Virtualization APIs

VirtualMachine [virtualmachine
VirtualMachineInstance [virtual
VirtualMach
[virtualmac

VirtualMachineClone [virtualma
VirtualMachineSnapshot
[virtualmachinesnapshots.snap
VirtualMach
[virtualmac

DataVolume [datavolumes.cdi.k
DataSource [datasources.cdi.ku
MigrationPo

VirtualMachineStorageMigratio
[virtualmachinestoragemigratio
VirtualMachineStorageMigratio
[virtualmachinestoragemigratio
VirtualMach

VirtualMachineClusterInstancet
[virtualmachineclusterinstance
VirtualMachineClusterPreference
[virtualmachineclusterpreferences.instancety

VirtualMachine [virtualmachines.kubevirt.io/v1]

Description

VirtualMachine handles the VirtualMachines that are not running or are in a stopped state The VirtualMachine contains the template to create the VirtualMachineInstance. It also mirrors the running state of the created VirtualMachineInstance in its status.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	Spec contains the specification of VirtualMachineInstance created
`status`	`object`	Status holds the current state of the controller and brief information about its associated VirtualMachineInstance

.spec

Description

Spec contains the specification of VirtualMachineInstance created

Type

`object`

Required

`template`

Property	Type	Description
`dataVolumeTemplates`	`array`	dataVolumeTemplates is a list of dataVolumes that the VirtualMachineInstance template can reference. DataVolumes in this list are dynamically created for the VirtualMachine and are tied to the VirtualMachine's life-cycle.
`instancetype`	`object`	InstancetypeMatcher references a instancetype that is used to fill fields in Template
`preference`	`object`	PreferenceMatcher references a set of preference that is used to fill fields in Template
`runStrategy`	`string`	Running state indicates the requested running state of the VirtualMachineInstance mutually exclusive with Running
`running`	`boolean`	Running controls whether the associatied VirtualMachineInstance is created or not Mutually exclusive with RunStrategy Deprecated: VirtualMachineInstance field "Running" is now deprecated, please use RunStrategy instead.
`template`	`object`	Template is the direct specification of VirtualMachineInstance

Property	Type	Description
`updateVolumesStrategy`	`string`	UpdateVolumesStrategy is the strategy to apply on volumes updates

.spec.dataVolumeTemplates

Description

dataVolumeTemplates is a list of dataVolumes that the VirtualMachineInstance template can reference. DataVolumes in this list are dynamically created for the VirtualMachine and are tied to the VirtualMachine's life-cycle.

Type

`array`

.spec.dataVolumeTemplates[]

Type

`object`

Required

`spec`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataVolumeSpec contains the DataVolume specification.
`status`	`object`	DataVolumeTemplateDummyStatus is here simply for backwards compatibility with a previous API.

.spec.dataVolumeTemplates[].spec

Description

DataVolumeSpec contains the DataVolume specification.

Type

`object`

Property	Type	Description
`checkpoints`	`array`	Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Property	Type	Description
`contentType`	`string`	DataVolumeContentType options: "kubevirt", "archive"
`finalCheckpoint`	`boolean`	FinalCheckpoint indicates whether the current DataVolumeCheckpoint is the final checkpoint.
`preallocation`	`boolean`	Preallocation controls whether storage for DataVolumes should be allocated in advance.
`priorityClassName`	`string`	PriorityClassName for Importer, Cloner and Uploader pod
`pvc`	`object`	PVC is the PVC specification
`source`	`object`	Source is the src of the data for the requested DataVolume
`sourceRef`	`object`	SourceRef is an indirect reference to the source of data for the requested DataVolume
`storage`	`object`	Storage is the requested storage specification

.spec.dataVolumeTemplates[].spec.checkpoints

Description

Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Type

`array`

.spec.dataVolumeTemplates[].spec.checkpoints[]

Description

DataVolumeCheckpoint defines a stage in a warm migration.

Type

`object`

Required

`current` `previous`

Property	Type	Description
`current`	`string`	Current is the identifier of the snapshot created for this checkpoint.
`previous`	`string`	Previous is the identifier of the snapshot from the previous checkpoint.

.spec.dataVolumeTemplates[].spec.pvc

Description

PVC is the PVC specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which populate the volume with data, if a non-empty v is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as both fields are non-empty, they must have the s

Property	Type	Description
		value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any names (Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.

Property	Type	Description
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume v attributes defined in the corresponding VolumeAttributesClass. This has a different pur than storageClassName, it can be changed afte claim is created. An empty string value means t VolumeAttributesClass will be applied to the cla it's not allowed to reset this field to empty string is set. If unspecified and the PersistentVolumeC unbound, the default VolumeAttributesClass wil by the persistentvolume controller if it exists. If t

Property	Type	Description
		resource referred to by volumeAttributesClass d not exist, this PersistentVolumeClaim will be se Pending state, as reflected by the modifyVolum field, until such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ (Beta) Using this field requ the VolumeAttributesClass feature gate to be en (off by default).
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.spec.dataVolumeTemplates[].spec.pvc.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.dataVolumeTemplates[].spec.pvc.accessModes[]

Type

`string`

.spec.dataVolumeTemplates[].spec.pvc.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.dataVolumeTemplates[].spec.pvc.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.dataVolumeTemplates[].spec.pvc.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

Property	Type	Description
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.dataVolumeTemplates[].spec.pvc.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataVolumeTemplates[].spec.pvc.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataVolumeTemplates[].spec.pvc.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.dataVolumeTemplates[].spec.pvc.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.dataVolumeTemplates[].spec.pvc.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.dataVolumeTemplates[].spec.pvc.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.dataVolumeTemplates[].spec.pvc.selector.matchExpressions[].values[]

Type

`string`

.spec.dataVolumeTemplates[].spec.pvc.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.dataVolumeTemplates[].spec.source

Description

Source is the src of the data for the requested DataVolume

Type

`object`

Property	Type	Description
`blank`	`object`	DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC
`gcs`	`object`	DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source
`http`	`object`	DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and

Property	Type	Description
		password, and an optional configmap containing additional CAs
`imageio`	`object`	DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`registry`	`object`	DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source
`s3`	`object`	DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot
`upload`	`object`	DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source
`vddk`	`object`	DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

.spec.dataVolumeTemplates[].spec.source.blank

Description

DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC

Type

`object`

.spec.dataVolumeTemplates[].spec.source.gcs

Description

DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source

Type

`object`

Required

`url`

Property	Type	Description
`secretRef`	`string`	SecretRef provides the secret reference needed to access the GCS source
`url`	`string`	URL is the url of the GCS source

.spec.dataVolumeTemplates[].spec.source.http

Description

DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`extraHeaders`	`array`	ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests
`secretExtraHeaders`	`array`	SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information
`secretRef`	`string`	SecretRef A Secret reference, the secret should contain accessKeyId (user name) base64 encoded, and secretKey (password) also base64 encoded
`url`	`string`	URL is the URL of the http(s) endpoint

.spec.dataVolumeTemplates[].spec.source.http.extraHeaders

Description

ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests

Type

`array`

.spec.dataVolumeTemplates[].spec.source.http.extraHeaders[]

Type

`string`

.spec.dataVolumeTemplates[].spec.source.http.secretExtraHeaders

Description

SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information

Type

`array`

.spec.dataVolumeTemplates[].spec.source.http.secretExtraHeaders[]

Type

`string`

.spec.dataVolumeTemplates[].spec.source.imageio

Description

DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source

Type

`object`

Required

`diskId` `url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the CA cert
`diskId`	`string`	DiskID provides id of a disk to be imported
`secretRef`	`string`	SecretRef provides the secret reference needed to access the ovirt-engine
`url`	`string`	URL is the URL of the ovirt-engine

.spec.dataVolumeTemplates[].spec.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC
`namespace`	`string`	The namespace of the source PVC

.spec.dataVolumeTemplates[].spec.source.registry

Description

DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source

Type

`object`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the Registry certs
`imageStream`	`string`	ImageStream is the name of image stream for import
`platform`	`object`	Platform describes the minimum runtime requirements of the image
`pullMethod`	`string`	PullMethod can be either "pod" (default import), or "node" (node docker cache based import)

Property	Type	Description
`secretRef`	`string`	SecretRef provides the secret reference needed to access the Registry source
`url`	`string`	URL is the url of the registry source (starting with the scheme: docker, oci-archive)

.spec.dataVolumeTemplates[].spec.source.registry.platform

Description

Platform describes the minimum runtime requirements of the image

Type

`object`

Property	Type	Description
`architecture`	`string`	Architecture specifies the image target CPU architecture

.spec.dataVolumeTemplates[].spec.source.s3

Description

DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`secretRef`	`string`	SecretRef provides the secret reference needed to access the S3 source
`url`	`string`	URL is the url of the S3 source

.spec.dataVolumeTemplates[].spec.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot
`namespace`	`string`	The namespace of the source VolumeSnapshot

.spec.dataVolumeTemplates[].spec.source.upload

Description

DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source

Type

`object`

.spec.dataVolumeTemplates[].spec.source.vddk

Description

DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

Type

`object`

Property	Type	Description
`backingFile`	`string`	BackingFile is the path to the virtual hard disk to migrate from vCenter/ESXi
`extraArgs`	`string`	ExtraArgs is a reference to a ConfigMap containing extra arguments to pass directly to the VDDK library
`initImageURL`	`string`	InitImageURL is an optional URL to an image containing an extracted VDDK library, overrides v2v-vmware config map
`secretRef`	`string`	SecretRef provides a reference to a secret containing the username and password needed to access the vCenter or

Property	Type	Description
		ESXi host
`thumbprint`	`string`	Thumbprint is the certificate thumbprint of the vCenter or ESXi host
`url`	`string`	URL is the URL of the vCenter or ESXi host with the VM to migrate
`uuid`	`string`	UUID is the UUID of the virtual machine that the backing file is attached to in vCenter/ESXi

.spec.dataVolumeTemplates[].spec.sourceRef

Description

SourceRef is an indirect reference to the source of data for the requested DataVolume

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	The kind of the source reference, currently only "DataSource" is supported

Property	Type	Description
`name`	`string`	The name of the source reference
`namespace`	`string`	The namespace of the source reference, defaults to the DataVolume namespace

.spec.dataVolumeTemplates[].spec.storage

Description

Storage is the requested storage specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`dataSource`	`object`	This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support

Property	Type	Description
		the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.
`dataSourceRef`	`object`	Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: • While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.

Property	Type	Description
`resources`	`object`	Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗
`selector`	`object`	A label query over volumes to consider for binding.
`storageClassName`	`string`	Name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.
`volumeName`	`string`	VolumeName is the binding reference to the PersistentVolume backing this claim.

.spec.dataVolumeTemplates[].spec.storage.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.dataVolumeTemplates[].spec.storage.accessModes[]

Type

`string`

.spec.dataVolumeTemplates[].spec.storage.dataSource

Description

This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced

.spec.dataVolumeTemplates[].spec.storage.dataSourceRef

Description

Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: * While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.

Property	Type	Description
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.dataVolumeTemplates[].spec.storage.resources

Description

Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

Property	Type	Description
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.dataVolumeTemplates[].spec.storage.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataVolumeTemplates[].spec.storage.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataVolumeTemplates[].spec.storage.selector

Description

A label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.dataVolumeTemplates[].spec.storage.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.dataVolumeTemplates[].spec.storage.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.dataVolumeTemplates[].spec.storage.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.dataVolumeTemplates[].spec.storage.selector.matchExpressions[].values[]

Type

`string`

.spec.dataVolumeTemplates[].spec.storage.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.dataVolumeTemplates[].status

Description

DataVolumeTemplateDummyStatus is here simply for backwards compatibility with a previous API.

Type

`object`

.spec.instancetype

Description

InstancetypeMatcher references a instancetype that is used to fill fields in Template

Type

`object`

Property	Type	Description
`inferFromVolume`	`string`	InferFromVolume lists the name of a volume that should be used to infer or

Property	Type	Description
		discover the instancetype to be used through known annotations on the underlying resource. Once applied to the InstancetypeMatcher this field is removed.
`inferFromVolumeFailurePolicy`	`string`	InferFromVolumeFailurePolicy controls what should happen on failure when inferring the instancetype. Allowed values are: "RejectInferFromVolumeFailure" and "IgnoreInferFromVolumeFailure". If not specified, "RejectInferFromVolumeFailure" is used by default.
`kind`	`string`	Kind specifies which instancetype resource is referenced. Allowed values are: "VirtualMachineInstancetype" and "VirtualMachineClusterInstancetype". If not specified, "VirtualMachineClusterInstancetype" is used by default.
`name`	`string`	Name is the name of the VirtualMachineInstancetype or VirtualMachineClusterInstancetype
`revisionName`	`string`	RevisionName specifies a ControllerRevision containing a

Property	Type	Description
		specific copy of the VirtualMachineInstancetype or VirtualMachineClusterInstancetype to be used. This is initially captured the first time the instancetype is applied to the VirtualMachineInstance.

.spec.preference

Description

PreferenceMatcher references a set of preference that is used to fill fields in Template

Type

`object`

Property	Type	Description
`inferFromVolume`	`string`	InferFromVolume lists the name of a volume that should be used to infer or discover the preference to be used through known annotations on the underlying resource. Once applied to the PreferenceMatcher this field is removed.
`inferFromVolumeFailurePolicy`	`string`	InferFromVolumeFailurePolicy controls what should happen on failure when preference the instancetype. Allowed values are: "RejectInferFromVolumeFailure" and "IgnoreInferFromVolumeFailure". If not specified,

Property	Type	Description
		"RejectInferFromVolumeFailure" is used by default.
`kind`	`string`	Kind specifies which preference resource is referenced. Allowed values are: "VirtualMachinePreference" and "VirtualMachineClusterPreference". If not specified, "VirtualMachineClusterPreference" is used by default.
`name`	`string`	Name is the name of the VirtualMachinePreference or VirtualMachineClusterPreference
`revisionName`	`string`	RevisionName specifies a ControllerRevision containing a specific copy of the VirtualMachinePreference or VirtualMachineClusterPreference to be used. This is initially captured the first time the instancetype is applied to the VirtualMachineInstance.

.spec.template

Description

Template is the direct specification of VirtualMachineInstance

Type

`object`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineInstance Spec contains the VirtualMachineInstance specification.

.spec.template.spec

Description

VirtualMachineInstance Spec contains the VirtualMachineInstance specification.

Type

`object`

Required

`domain`

Property	Type	Description
`accessCredentials`	`array`	Specifies a set of public keys to inject into
`affinity`	`object`	If affinity is specifies, obey all the affinity ru
`architecture`	`string`	Specifies the architecture of the vm guest y architecture of the KubeVirt components

Property	Type	Description
`dnsConfig`	`object`	Specifies the DNS parameters of a pod. Pa generated DNS configuration based on DN
`dnsPolicy`	`string`	Set DNS policy for the pod. Defaults to "Cl 'ClusterFirstWithHostNet', 'ClusterFirst', 'De DNSConfig will be merged with the policy s along with hostNetwork, you have to speci
`domain`	`object`	Specification of the desired behavior of the
`evictionStrategy`	`string`	EvictionStrategy describes the strategy to t options are: • "None": No action will be taken, accordi VirtualMachine will be restarted or shut • "LiveMigrate": the VirtualMachineInstan • "LiveMigrateIfPossible": the same as "L Migratable, otherwise it will behave as " • "External": the VirtualMachineInstance v 'vmi.Status.EvacuationNodeName' will b api-provider-kubevirt (capk) which need signal capk that eviction has been calle tearing the VMI down. Details can be fo https://github.com/kubevirt/kubevirt/com ↗.

Property	Type	Description
`hostname`	`string`	Specifies the hostname of the vmi If not sp vmi, if dhcp or cloud-init is configured prop
`livenessProbe`	`object`	Periodic probe of VirtualMachineInstance l the probe fails. Cannot be updated. More i https://kubernetes.io/docs/concepts/worklo
`networks`	`array`	List of networks that can be attached to a v
`nodeSelector`	`object`	NodeSelector is a selector which must be t must match a node's labels for the vmi to b https://kubernetes.io/docs/concepts/configu
`priorityClassName`	`string`	If specified, indicates the pod's priority. If n there is no default.
`readinessProbe`	`object`	Periodic probe of VirtualMachineInstance s removed from service endpoints if the prob https://kubernetes.io/docs/concepts/worklo
`resourceClaims`	`array`	ResourceClaims define which ResourceCl VMI, hence virt-launcher pod is allowed to domain which consumes them by name. This is an alpha field and requires enablinç kubernetes https://kubernetes.io/docs/conc

Property	Type	Description
		allocation/↗ This field should only be confi HostDevicesWithDRA is enabled. This feat
`schedulerName`	`string`	If specified, the VMI will be dispatched by s dispatched by default scheduler.
`startStrategy`	`string`	StartStrategy can be set to "Paused" if Virt
`subdomain`	`string`	If specified, the fully qualified vmi hostnam have a domainname at all. The DNS entry can pick up a hostname.
`terminationGracePeriodSeconds`	`integer`	Grace period observed after signalling a Vi VirtualMachineInstance is force terminated
`tolerations`	`array`	If toleration is specified, obey all the tolerat
`topologySpreadConstraints`	`array`	TopologySpreadConstraints describes how topology domains. K8s scheduler will sche constraints.
`volumes`	`array`	List of volumes that can be mounted by dis

.spec.template.spec.accessCredentials

Description

Specifies a set of public keys to inject into the vm guest

Type

`array`

.spec.template.spec.accessCredentials[]

Description

AccessCredential represents a credential source that can be used to authorize remote access to the vm guest Only one of its members may be specified.

Type

`object`

Property	Type	Description
`sshPublicKey`	`object`	SSHPublicKey represents the source and method of applying a ssh public key into a guest virtual machine.
`userPassword`	`object`	UserPassword represents the source and method for applying a guest user's password

.spec.template.spec.accessCredentials[].sshPublicKey

Description

SSHPublicKey represents the source and method of applying a ssh public key into a guest virtual machine.

Type

`object`

Required

`propagationMethod` `source`

Property	Type	Description
`propagationMethod`	`object`	PropagationMethod represents how the public key is injected into the vm guest.
`source`	`object`	Source represents where the public keys are pulled from

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod

Description

PropagationMethod represents how the public key is injected into the vm guest.

Type

`object`

Property	Type	Description
`configDrive`	`object`	ConfigDrivePropagation means that the ssh public keys are injected into the VM using metadata using the configDrive cloud-init provider
`noCloud`	`object`	NoCloudPropagation means that the ssh public keys are injected into the VM using metadata using the noCloud cloud-init provider
`qemuGuestAgent`	`object`	QemuGuestAgentAccessCredentailPropagation means ssh public keys are dynamically injected into the vm at

Property	Type	Description
		runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod.configDrive

Description

ConfigDrivePropagation means that the ssh public keys are injected into the VM using metadata using the configDrive cloud-init provider

Type

`object`

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod.noCloud

Description

NoCloudPropagation means that the ssh public keys are injected into the VM using metadata using the noCloud cloud-init provider

Type

`object`

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent

Description

QemuGuestAgentAccessCredentailPropagation means ssh public keys are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

Type

`object`

Required

`users`

Property	Type	Description
`users`	`array`	Users represents a list of guest users that should have the ssh public keys added to their authorized_keys file.

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent.users

Description

Users represents a list of guest users that should have the ssh public keys added to their authorized_keys file.

Type

`array`

.spec.template.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent.users[]

Type

`string`

.spec.template.spec.accessCredentials[].sshPublicKey.source

Description

Source represents where the public keys are pulled from

Type

`object`

Property	Type	Description
`secret`	`object`	Secret means that the access credential is pulled from a kubernetes secret

.spec.template.spec.accessCredentials[].sshPublicKey.source.secret

Description

Secret means that the access credential is pulled from a kubernetes secret

Type

`object`

Required

`secretName`

Property	Type	Description
`secretName`	`string`	SecretName represents the name of the secret in the VMI's namespace

.spec.template.spec.accessCredentials[].userPassword

Description

UserPassword represents the source and method for applying a guest user's password

Type

`object`

Required

`propagationMethod` `source`

Property	Type	Description
`propagationMethod`	`object`	propagationMethod represents how the user passwords are injected into the vm guest.
`source`	`object`	Source represents where the user passwords are pulled from

.spec.template.spec.accessCredentials[].userPassword.propagationMethod

Description

propagationMethod represents how the user passwords are injected into the vm guest.

Type

`object`

Property	Type	Description
`qemuGuestAgent`	`object`	QemuGuestAgentAccessCredentailPropagation means passwords are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

.spec.template.spec.accessCredentials[].userPassword.propagationMethod.qemuGuestAgent

Description

QemuGuestAgentAccessCredentailPropagation means passwords are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

Type

`object`

.spec.template.spec.accessCredentials[].userPassword.source

Description

Source represents where the user passwords are pulled from

Type

`object`

Property	Type	Description
`secret`	`object`	Secret means that the access credential is pulled from a kubernetes secret

.spec.template.spec.accessCredentials[].userPassword.source.secret

Description

Secret means that the access credential is pulled from a kubernetes secret

Type

`object`

Required

`secretName`

Property	Type	Description
`secretName`	`string`	SecretName represents the name of the secret in the VMI's namespace

.spec.template.spec.affinity

Description

If affinity is specifies, obey all the affinity rules

Type

`object`

Property	Type	Description
`nodeAffinity`	`object`	Describes node affinity scheduling rules for the pod.
`podAffinity`	`object`	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
`podAntiAffinity`	`object`	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

.spec.template.spec.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the affinity expressions specified by this field, it may choose a node violates one or more the expressions. The node that is most preferred is the one v the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched affinity expressions, e compute a sum by iterating through the elements of this field adding "weight" to the sum if the node matc the corresponding matchExpressions; th node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`object`	If the affinity requirem specified by this field

Property	Type	Description
		not met at scheduling time, the pod will not scheduled onto the n If the affinity requirem specified by this field cease to be met at sc point during pod execution (e.g. due to update), the system r or may not try to eventually evict the p from its node.

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

`object`

Required

`preference` `weight`

Property	Type	Description
`preference`	`object`	A node selector term, associated with the corresponding weight.
`weight`	`integer`	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference

Description

A node selector term, associated with the corresponding weight.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFiel

ds

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single

Property	Type	Description
		element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values[]

Type

`string`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to

be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

`object`

Required

`nodeSelectorTerms`

Property	Type	Description
`nodeSelectorTerms`	`array`	Required. A list of node selector terms. The terms are ORed.

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms

Description

Required. A list of node selector terms. The terms are ORed.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[]

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].ma

tchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.

Property	Type	Description
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values[]

Type

`string`

.spec.template.spec.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the affinity expressions specified by this field, it may choose a node violates one or more the expressions. The node that is most preferred is the one w the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched affinity expressions, e compute a sum by iterating through the elements of this field adding "weight" to the sum if the node has p which matches the corresponding podAffinityTerm; the

Property	Type	Description
		node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the affinity requirem specified by this field not met at scheduling time, the pod will not scheduled onto the n If the affinity requirem specified by this field cease to be met at sc point during pod execution (e.g. due to pod label update), the system may or may n try to eventually evict pod from its node. Wh there are multiple elements, the lists of nodes corresponding each podAffinityTerm intersected, i.e. all ter must be satisfied.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights,

i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to

Property	Type	Description
		select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.name

spaceSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.template.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Property	Type	Description
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of

existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is

Property	Type	Description
		"key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.template.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the anti-affinity expressio specified by this field, it may choose a node violates one or more the expressions. The node that is most preferred is the one v the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched anti-affinity expressio etc.), compute a sum iterating through the elements of this field adding "weight" to the sum if the node has p which matches the corresponding podAffinityTerm; the node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the anti-affinity requirements specifie this field are not met scheduling time, the p

Property	Type	Description
		will not be scheduled the node. If the anti-affinity requirements specified by this field cease to be met at sc point during pod execution (e.g. due tc pod label update), the system may or may n try to eventually evict pod from its node. Wl there are multiple elements, the lists of nodes corresponding each podAffinityTerm intersected, i.e. all ter must be satisfied.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Property	Type	Description
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.na

mespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Property	Type	Description
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelecto

r.matchExpressions[].values[]

Type

`string`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.template.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.template.spec.dnsConfig

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
`nameservers`	`array`	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.
`options`	`array`	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
`searches`	`array`	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

.spec.template.spec.dnsConfig.nameservers

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

.spec.template.spec.dnsConfig.nameservers[]

Type

`string`

.spec.template.spec.dnsConfig.options

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

`array`

.spec.template.spec.dnsConfig.options[]

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

`object`

Property	Type	Description
`name`	`string`	Name is this DNS resolver option's name. Required.
`value`	`string`	Value is this DNS resolver option's value.

.spec.template.spec.dnsConfig.searches

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

`array`

.spec.template.spec.dnsConfig.searches[]

Type

`string`

.spec.template.spec.domain

Description

Specification of the desired behavior of the VirtualMachineInstance on the host.

Type

`object`

Required

`devices`

Property	Type	Description
`chassis`	`object`	Chassis specifies the chassis info passed to the domain.
`clock`	`object`	Clock sets the clock and timers of the vmi.

Property	Type	Description
`cpu`	`object`	CPU allow specified the detailed CPU topology inside the vmi.
`devices`	`object`	Devices allows adding disks, network interfaces, and others
`features`	`object`	Features like acpi, apic, hyperv, smm.
`firmware`	`object`	Firmware.
`ioThreads`	`object`	IOThreads specifies the IOThreads options.
`ioThreadsPolicy`	`string`	Controls whether or not disks will share IOThreads. Omitting IOThreadsPolicy disables use of IOThreads. One of: shared, auto, supplementalPool
`launchSecurity`	`object`	Launch Security setting of the vmi.
`machine`	`object`	Machine type.
`memory`	`object`	Memory allow specifying the VMI memory features.

Property	Type	Description
`resources`	`object`	Resources describes the Compute Resources required by this vmi.

.spec.template.spec.domain.chassis

Description

Chassis specifies the chassis info passed to the domain.

Type

`object`

Property	Type	Description
`asset`	`string`	
`manufacturer`	`string`	
`serial`	`string`	
`sku`	`string`	
`version`	`string`	

.spec.template.spec.domain.clock

Description

Clock sets the clock and timers of the vmi.

Type

`object`

Property	Type	Description
`timer`	`object`	Timer specifies whih timers are attached to the vmi.
`timezone`	`string`	Timezone sets the guest clock to the specified timezone. Zone name follows the TZ environment variable format (e.g. 'America/New_York').
`utc`	`object`	UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

.spec.template.spec.domain.clock.timer

Description

Timer specifies whih timers are attached to the vmi.

Type

`object`

Property	Type	Description
`hpet`	`object`	HPET (High Precision Event Timer) - multiple timers with periodic interrupts.
`hyperv`	`object`	Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.

Property	Type	Description
`kvm`	`object`	KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.
`pit`	`object`	PIT (Programmable Interval Timer) - a timer with periodic interrupts.
`rtc`	`object`	RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

.spec.template.spec.domain.clock.timer.hpet

Description

HPET (High Precision Event Timer) - multiple timers with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "merge", "discard".

.spec.template.spec.domain.clock.timer.hyperv

Description

Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.template.spec.domain.clock.timer.kvm

Description

KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.template.spec.domain.clock.timer.pit

Description

PIT (Programmable Interval Timer) - a timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "discard".

.spec.template.spec.domain.clock.timer.rtc

Description

RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup".

Property	Type	Description
`track`	`string`	Track the guest or the wall clock.

.spec.template.spec.domain.clock.utc

Description

UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

Type

`object`

Property	Type	Description
`offsetSeconds`	`integer`	OffsetSeconds specifies an offset in seconds, relative to UTC. If set, guest changes to the clock will be kept during reboots and not reset.

.spec.template.spec.domain.cpu

Description

CPU allow specified the detailed CPU topology inside the vmi.

Type

`object`

Property	Type	Description
`cores`	`integer`	Cores specifies the number of cores inside the vmi. Must be a value greater or equal 1.

Property	Type	Description
`dedicatedCpuPlacement`	`boolean`	DedicatedCPUPlacement requests the scheduler to place the VirtualMachineInstance on a node with enough dedicated pCPUs and pin the vCPUs to it.
`features`	`array`	Features specifies the CPU features list inside the VMI.
`isolateEmulatorThread`	`boolean`	IsolateEmulatorThread requests one more dedicated pCPU to be allocated for the VMI to place the emula thread on it.
`maxSockets`	`integer`	MaxSockets specifies the maximum amount of sock that can be hotplugged
`model`	`string`	Model specifies the CPU model inside the VMI. List available models https://github.com/libvirt/libvirt/tree/master/src/cpu_m ↗. It is possible to specify special cases like "host-passthrough" to get the same CPU as the node and "host-model" to get CPU closest to the node one. Defaults to host-model.
`numa`	`object`	NUMA allows specifying settings for the guest NUM topology

Property	Type	Description
`realtime`	`object`	Realtime instructs the virt-launcher to tune the VMI f lower latency, optional for real time workloads
`sockets`	`integer`	Sockets specifies the number of sockets inside the vmi. Must be a value greater or equal 1.
`threads`	`integer`	Threads specifies the number of threads inside the vmi. Must be a value greater or equal 1.

.spec.template.spec.domain.cpu.features

Description

Features specifies the CPU features list inside the VMI.

Type

`array`

.spec.template.spec.domain.cpu.features[]

Description

CPUFeature allows specifying a CPU feature.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name of the CPU feature
`policy`	`string`	Policy is the CPU feature attribute which can have the following attributes: force - The virtual CPU will claim the feature is supported regardless of it being supported by host CPU. require - Guest creation will fail unless the feature is supported by the host CPU or the hypervisor is able to emulate it. optional - The feature will be supported by virtual CPU if and only if it is supported by host CPU. disable - The feature will not be supported by virtual CPU. forbid - Guest creation will fail if the feature is supported by host CPU. Defaults to require

.spec.template.spec.domain.cpu.numa

Description

NUMA allows specifying settings for the guest NUMA topology

Type

`object`

Property	Type	Description
`guestMappingPassthrough`	`object`	GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on the virtual numa nodes never cross boundaries of host numa nodes.

.spec.template.spec.domain.cpu.numa.guestMappingPassthrough

Description

GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on the virtual numa nodes never cross boundaries of host numa nodes.

Type

`object`

.spec.template.spec.domain.cpu.realtime

Description

Realtime instructs the virt-launcher to tune the VMI for lower latency, optional for real time workloads

Type

`object`

Property	Type	Description
`mask`	`string`	Mask defines the vcpu mask expression that defines which vcpus are used for realtime. Format matches libvirt's expressions. Example: "0-3,^1","0,2,3","2-3"

.spec.template.spec.domain.devices

Description

Devices allows adding disks, network interfaces, and others

Type

`object`

Property	Type	Description
`autoattachGraphicsDevice`	`boolean`	Whether to attach the default graphics device or not. VNC will not be available if set to false. Defaults to true.
`autoattachInputDevice`	`boolean`	Whether to attach an Input Device. Defaults to false.
`autoattachMemBalloon`	`boolean`	Whether to attach the Memory balloon device with default period. Period can be adjusted in virt-config. Defaults to true.
`autoattachPodInterface`	`boolean`	Whether to attach a pod network interface. Defaults to true.
`autoattachSerialConsole`	`boolean`	Whether to attach the default virtio-serial console or not. Serial console access will not be available if set to false. Defaults to true.
`autoattachVSOCK`	`boolean`	Whether to attach the VSOCK CID to the VM or not. VSOCK access will be available if set to true. Defaults to false.

Property	Type	Description
`blockMultiQueue`	`boolean`	Whether or not to enable virtio multi-queue for block devices. Defaults to false.
`clientPassthrough`	`object`	To configure and access client devices such as redirecting USB
`disableHotplug`	`boolean`	DisableHotplug disabled the ability to hotplug disks.
`disks`	`array`	Disks describes disks, cdroms and luns which are connected to the vmi.
`downwardMetrics`	`object`	DownwardMetrics creates a virtio serials for exposing the downward metrics to the vmi.
`filesystems`	`array`	Filesystems describes filesystem which is connected to the vmi.
`gpus`	`array`	Whether to attach a GPU device to the vmi.

Property	Type	Description
`hostDevices`	`array`	Whether to attach a host device to the vmi.
`inputs`	`array`	Inputs describe input devices
`interfaces`	`array`	Interfaces describe network interfaces which are added to the vmi.
`logSerialConsole`	`boolean`	Whether to log the auto-attached default serial console or not. Serial console logs will be collect to a file and then streamed from a named 'guest-console-log'. Not relevant if autoattachSerialConsole is disabled. Defaults to cluster wide setting on VirtualMachineOptions.
`networkInterfaceMultiqueue`	`boolean`	If specified, virtual network interfaces configured with a virtio bus will also enable the vhost multiqueue feature for network devices. The number of queues created depends on additional factors of the VirtualMachineInstance, like the number of guest CPUs.
`panicDevices`	`array`	PanicDevices provides additional crash information when a guest crashes.

Property	Type	Description
`rng`	`object`	Whether to have random number generator from host
`sound`	`object`	Whether to emulate a sound device.
`tpm`	`object`	Whether to emulate a TPM device.
`useVirtioTransitional`	`boolean`	Fall back to legacy virtio 0.9 support if virtio bus is selected on devices. This is helpful for old machines like CentOS6 or RHEL6 which do not understand virtio_non_transitional (virtio 1.0).
`video`	`object`	Video describes the video device configuration for the vmi.
`watchdog`	`object`	Watchdog describes a watchdog device which can be added to the vmi.

.spec.template.spec.domain.devices.clientPassthrough

Description

To configure and access client devices such as redirecting USB

Type

`object`

.spec.template.spec.domain.devices.disks

Description

Disks describes disks, cdroms and luns which are connected to the vmi.

Type

`array`

.spec.template.spec.domain.devices.disks[]

Type

`object`

Required

`name`

Property	Type	Description
`blockSize`	`object`	If specified, the virtual disk will be presented with the given block sizes.
`bootOrder`	`integer`	BootOrder is an integer value > 0, used to determine ordering of boot devices. Lower values take precedence. Each disk or interface that has a boot order must have a unique value. Disks without a boot order are not tried if a disk with a boot order exists.
`cache`	`string`	Cache specifies which kvm disk cache mode should be used. Supported values are: none: Guest I/O not cached on the host, but may be kept in a disk cache. writethrough: Guest I/O cached on the host but written through to the

Property	Type	Description
		physical medium. Slowest but with most guarantees. writeback: Guest I/O cached on the host. Defaults to none if the storage supports O_DIRECT, otherwise writethrough.
`cdrom`	`object`	Attach a volume as a cdrom to the vmi.
`changedBlockTracking`	`boolean`	ChangedBlockTracking indicates this disk should have CBT option Defaults to false.
`dedicatedIOThread`	`boolean`	dedicatedIOThread indicates this disk should have an exclusive IO Thread. Enabling this implies useIOThreads = true. Defaults to false.
`disk`	`object`	Attach a volume as a disk to the vmi.
`errorPolicy`	`string`	If specified, it can change the default error policy (stop) for the disk
`io`	`string`	IO specifies which QEMU disk IO mode should be used. Supported values are: native, default, threads.
`lun`	`object`	Attach a volume as a LUN to the vmi.

Property	Type	Description
`name`	`string`	Name is the device name
`serial`	`string`	Serial provides the ability to specify a serial number for the disk device.
`shareable`	`boolean`	If specified the disk is made sharable and multiple write from different VMs are permitted
`tag`	`string`	If specified, disk address and its tag will be provided to the guest via config drive metadata

.spec.template.spec.domain.devices.disks[].blockSize

Description

If specified, the virtual disk will be presented with the given block sizes.

Type

`object`

Property	Type	Description
`custom`	`object`	CustomBlockSize represents the desired logical and physical block size for a VM disk.
`matchVolume`	`object`	Represents if a feature is enabled or disabled.

.spec.template.spec.domain.devices.disks[].blockSize.custom

Description

CustomBlockSize represents the desired logical and physical block size for a VM disk.

Type

`object`

Property	Type	Description
`discardGranularity`	`integer`	
`logical`	`integer`	
`physical`	`integer`	

.spec.template.spec.domain.devices.disks[].blockSize.matchVolume

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.devices.disks[].cdrom

Description

Attach a volume as a cdrom to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to true.
`tray`	`string`	Tray indicates if the tray of the device is open or closed. Allowed values are "open" and "closed". Defaults to closed.

.spec.template.spec.domain.devices.disks[].disk

Description

Attach a volume as a disk to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi, usb.

Property	Type	Description
`pciAddress`	`string`	If specified, the virtual disk will be placed on the guests pci address with the specified PCI address. For example: 0000:81:01.10
`readonly`	`boolean`	ReadOnly. Defaults to false.

.spec.template.spec.domain.devices.disks[].lun

Description

Attach a volume as a LUN to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to false.
`reservation`	`boolean`	Reservation indicates if the disk needs to support the persistent reservation for the SCSI disk

.spec.template.spec.domain.devices.downwardMetrics

Description

DownwardMetrics creates a virtio serials for exposing the downward metrics to the vmi.

Type

`object`

.spec.template.spec.domain.devices.filesystems

Description

Filesystems describes filesystem which is connected to the vmi.

Type

`array`

.spec.template.spec.domain.devices.filesystems[]

Type

`object`

Required

`name` `virtiofs`

Property	Type	Description
`name`	`string`	Name is the device name
`virtiofs`	`object`	Virtiofs is supported

.spec.template.spec.domain.devices.filesystems[].virtiofs

Description

Virtiofs is supported

Type

`object`

.spec.template.spec.domain.devices.gpus

Description

Whether to attach a GPU device to the vmi.

Type

`array`

.spec.template.spec.domain.devices.gpus[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	Name of the GPU device as exposed by a device plugin
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where

Property	Type	Description
		this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive
`virtualGPUOptions`	`object`	

.spec.template.spec.domain.devices.gpus[].virtualGPUOptions

Type

`object`

Property	Type	Description
`display`	`object`	

.spec.template.spec.domain.devices.gpus[].virtualGPUOptions.display

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if a display addapter backed by a vGPU should be enabled or disabled on the guest. Defaults to true.

Property	Type	Description
`ramFB`	`object`	Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

.spec.template.spec.domain.devices.gpus[].virtualGPUOptions.display.ramFB

Description

Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.devices.hostDevices

Description

Whether to attach a host device to the vmi.

Type

`array`

.spec.template.spec.domain.devices.hostDevices[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive

.spec.template.spec.domain.devices.inputs

Description

Inputs describe input devices

Type

`array`

.spec.template.spec.domain.devices.inputs[]

Type

`object`

Required

`name` `type`

Property	Type	Description
`bus`	`string`	Bus indicates the bus of input device to emulate. Supported values: virtio, usb.
`name`	`string`	Name is the device name
`type`	`string`	Type indicated the type of input device. Supported values: tablet.

.spec.template.spec.domain.devices.interfaces

Description

Interfaces describe network interfaces which are added to the vmi.

Type

`array`

.spec.template.spec.domain.devices.interfaces[]

Type

`object`

Required

`name`

Property	Type	Description
`acpiIndex`	`integer`	If specified, the ACPI index is used to provide network interface device naming, that is stable across changes in PCI addresses assigned to the device. This value is required to be unique across all devices and be between 1 and (16*1024-1).
`binding`	`object`	Binding specifies the binding plugin that will be used to connect the interface to the guest. It provides an alternative to InterfaceBindingMethod. version: 1alphav1
`bootOrder`	`integer`	BootOrder is an integer value > 0, used to determine ordering of boot devices. Lower values take precedence. Each interface or disk that has a boot order must have a unique value. Interfaces without a boot order are not tried.
`bridge`	`object`	InterfaceBridge connects to a given network via a linux bridge.
`dhcpOptions`	`object`	If specified the network interface will pass additional DHCP options to the VMI
`macAddress`	`string`	Interface MAC address. For example: de:ad:00:00:be:af or DE-AD-00-00-BE-AF.
`macvtap`	`object`	DeprecatedMacvtap is an alias to the deprecated Macvtap interface, please refer to Kubevirt user guide for

Property	Type	Description
		alternatives. Deprecated: Removed in v1.3
`masquerade`	`object`	InterfaceMasquerade connects to a given network using netfilter rules to nat the traffic.
`model`	`string`	Interface model. One of: e1000, e1000e, igb, ne2k_pci, pcnet, rtl8139, virtio. Defaults to virtio.
`name`	`string`	Logical name of the interface as well as a reference to the associated networks. Must match the Name of a Network.
`passt`	`object`	DeprecatedPasst is an alias to the deprecated Passt interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3
`pciAddress`	`string`	If specified, the virtual network interface will be placed on the guests pci address with the specified PCI address. For example: 0000:81:01.10
`ports`	`array`	List of ports to be forwarded to the virtual machine.
`slirp`	`object`	DeprecatedSlirp is an alias to the deprecated Slirp interface Deprecated: Removed in v1.3

Property	Type	Description
`sriov`	`object`	InterfaceSRIOV connects to a given network by passing-through an SR-IOV PCI device via vfio.
`state`	`string`	State represents the requested operational state of the interface. The supported values are: 'absent', expressing a request to remove the interface. 'down', expressing a request to set the link down. 'up', expressing a request to set the link up. Empty value functions as 'up'.
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive

.spec.template.spec.domain.devices.interfaces[].binding

Description

Binding specifies the binding plugin that will be used to connect the interface to the guest. It provides an alternative to InterfaceBindingMethod. version: 1alphav1

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name references to the binding name as denined in the kubevirt CR. version: 1alphav1

.spec.template.spec.domain.devices.interfaces[].bridge

Description

InterfaceBridge connects to a given network via a linux bridge.

Type

`object`

.spec.template.spec.domain.devices.interfaces[].dhcpOptions

Description

If specified the network interface will pass additional DHCP options to the VMI

Type

`object`

Property	Type	Description
`bootFileName`	`string`	If specified will pass option 67 to interface's DHCP server
`ntpServers`	`array`	If specified will pass the configured NTP server to the VM via DHCP option 042.
`privateOptions`	`array`	If specified will pass extra DHCP options for private use, range: 224-254
`tftpServerName`	`string`	If specified will pass option 66 to interface's DHCP server

.spec.template.spec.domain.devices.interfaces[].dhcpOptions.ntpServers

Description

If specified will pass the configured NTP server to the VM via DHCP option 042.

Type

`array`

.spec.template.spec.domain.devices.interfaces[].dhcpOptions.ntpServers[]

Type

`string`

.spec.template.spec.domain.devices.interfaces[].dhcpOptions.privateOptions

Description

If specified will pass extra DHCP options for private use, range: 224-254

Type

`array`

.spec.template.spec.domain.devices.interfaces[].dhcpOptions.privateOptions[]

Description

DHCPExtraOptions defines Extra DHCP options for a VM.

Type

`object`

Required

`option` `value`

Property	Type	Description
`option`	`integer`	Option is an Integer value from 224-254 Required.
`value`	`string`	Value is a String value for the Option provided Required.

.spec.template.spec.domain.devices.interfaces[].macvtap

Description

DeprecatedMacvtap is an alias to the deprecated Macvtap interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3

Type

`object`

.spec.template.spec.domain.devices.interfaces[].masquerade

Description

InterfaceMasquerade connects to a given network using netfilter rules to nat the traffic.

Type

`object`

.spec.template.spec.domain.devices.interfaces[].passt

Description

DeprecatedPasst is an alias to the deprecated Passt interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3

Type

`object`

.spec.template.spec.domain.devices.interfaces[].ports

Description

List of ports to be forwarded to the virtual machine.

Type

`array`

.spec.template.spec.domain.devices.interfaces[].ports[]

Description

Port represents a port to expose from the virtual machine. Default protocol TCP. The port field is mandatory

Type

`object`

Required

`port`

Property	Type	Description
`name`	`string`	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
`port`	`integer`	Number of port to expose for the virtual machine. This must be a valid port number, 0 < x < 65536.

Property	Type	Description
`protocol`	`string`	Protocol for port. Must be UDP or TCP. Defaults to "TCP".

.spec.template.spec.domain.devices.interfaces[].slirp

Description

DeprecatedSlirp is an alias to the deprecated Slirp interface Deprecated: Removed in v1.3

Type

`object`

.spec.template.spec.domain.devices.interfaces[].sriov

Description

InterfaceSRIOV connects to a given network by passing-through an SR-IOV PCI device via vfio.

Type

`object`

.spec.template.spec.domain.devices.panicDevices

Description

PanicDevices provides additional crash information when a guest crashes.

Type

`array`

.spec.template.spec.domain.devices.panicDevices[]

Type

`object`

Property	Type	Description
`model`	`string`	Model specifies what type of panic device is provided. The panic model used when this attribute is missing depends on the hypervisor and guest arch. One of: isa, hyperv, pvpanic.

.spec.template.spec.domain.devices.rng

Description

Whether to have random number generator from host

Type

`object`

.spec.template.spec.domain.devices.sound

Description

Whether to emulate a sound device.

Type

`object`

Required

`name`

Property	Type	Description
`model`	`string`	We only support ich9 or ac97. If SoundDevice is not set: No sound card is emulated. If SoundDevice is set but Model is not: ich9

Property	Type	Description
`name`	`string`	User's defined name for this sound device

.spec.template.spec.domain.devices.tpm

Description

Whether to emulate a TPM device.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled allows a user to explicitly disable the vTPM even when one is enabled by a preference referenced by the VirtualMachine Defaults to True
`persistent`	`boolean`	Persistent indicates the state of the TPM device should be kept accross reboots Defaults to false

.spec.template.spec.domain.devices.video

Description

Video describes the video device configuration for the vmi.

Type

`object`

Property	Type	Description
`type`	`string`	Type specifies the video device type (e.g., virtio, vga, bochs, ramfb). If not specified, the default is architecture-dependent (VGA for BIOS-based VMs, Bochs for EFI-based VMs on AMD64; virtio for Arm and s390x).

.spec.template.spec.domain.devices.watchdog

Description

Watchdog describes a watchdog device which can be added to the vmi.

Type

`object`

Required

`name`

Property	Type	Description
`diag288`	`object`	diag288 watchdog device (specific to s390x architecture).
`i6300esb`	`object`	i6300esb watchdog device.
`name`	`string`	Name of the watchdog.

.spec.template.spec.domain.devices.watchdog.diag288

Description

diag288 watchdog device (specific to s390x architecture).

Type

`object`

Property	Type	Description
`action`	`string`	The action to take. Valid values are poweroff, reset, shutdown. Defaults to reset.

.spec.template.spec.domain.devices.watchdog.i6300esb

Description

i6300esb watchdog device.

Type

`object`

Property	Type	Description
`action`	`string`	The action to take. Valid values are poweroff, reset, shutdown. Defaults to reset.

.spec.template.spec.domain.features

Description

Features like acpi, apic, hyperv, smm.

Type

`object`

Property	Type	Description
`acpi`	`object`	ACPI enables/disables ACPI inside the guest. Defaults to enabled.
`apic`	`object`	Defaults to the machine type setting.
`hyperv`	`object`	Defaults to the machine type setting.
`hypervPassthrough`	`object`	This enables all supported hyperv flags automatically. Bear in mind that if this enabled hyperV features cannot be enabled explicitly. In addition, a Virtual Machine using it will be non-migratable.
`kvm`	`object`	Configure how KVM presence is exposed to the guest.
`pvspinlock`	`object`	Notify the guest that the host supports paravirtual spinlocks. For older kernels this feature should be explicitly disabled.
`smm`	`object`	SMM enables/disables System Management Mode. TSEG not yet implemented.

.spec.template.spec.domain.features.acpi

Description

ACPI enables/disables ACPI inside the guest. Defaults to enabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.apic

Description

Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`endOfInterrupt`	`boolean`	EndOfInterrupt enables the end of interrupt notification in the guest. Defaults to false.

.spec.template.spec.domain.features.hyperv

Description

Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`evmcs`	`object`	EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.
`frequencies`	`object`	Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.
`ipi`	`object`	IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.
`reenlightenment`	`object`	Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.
`relaxed`	`object`	Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.
`reset`	`object`	Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.

Property	Type	Description
`runtime`	`object`	Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.
`spinlocks`	`object`	Spinlocks allows to configure the spinlock retry attempts.
`synic`	`object`	SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.
`synictimer`	`object`	SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.
`tlbflush`	`object`	TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.
`vapic`	`object`	VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.
`vendorid`	`object`	VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.

Property	Type	Description
`vpindex`	`object`	VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

.spec.template.spec.domain.features.hyperv.evmcs

Description

EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.frequencies

Description

Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.ipi

Description

IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.reenlightenment

Description

Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.relaxed

Description

Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.reset

Description

Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.runtime

Description

Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.spinlocks

Description

Spinlocks allows to configure the spinlock retry attempts.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

Property	Type	Description
`spinlocks`	`integer`	Retries indicates the number of retries. Must be a value greater or equal 4096. Defaults to 4096.

.spec.template.spec.domain.features.hyperv.synic

Description

SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.synictimer

Description

SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`direct`	`object`	Represents if a feature is enabled or disabled.

Property	Type	Description
`enabled`	`boolean`	

.spec.template.spec.domain.features.hyperv.synictimer.direct

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.tlbflush

Description

TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.vapic

Description

VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hyperv.vendorid

Description

VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`vendorid`	`string`	VendorID sets the hypervisor vendor id, visible to the vmi. String up to twelve characters.

.spec.template.spec.domain.features.hyperv.vpindex

Description

VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.hypervPassthrough

Description

This enables all supported hyperv flags automatically. Bear in mind that if this enabled hyperV features cannot be enabled explicitly. In addition, a Virtual Machine using it will be non-migratable.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	

.spec.template.spec.domain.features.kvm

Description

Configure how KVM presence is exposed to the guest.

Type

`object`

Property	Type	Description
`hidden`	`boolean`	Hide the KVM hypervisor from standard MSR based discovery. Defaults to false

.spec.template.spec.domain.features.pvspinlock

Description

Notify the guest that the host supports paravirtual spinlocks. For older kernels this feature should be explicitly disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.features.smm

Description

SMM enables/disables System Management Mode. TSEG not yet implemented.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.template.spec.domain.firmware

Description

Firmware.

Type

`object`

Property	Type	Description
`acpi`	`object`	Information that can be set in the ACPI table
`bootloader`	`object`	Settings to control the bootloader that is used.
`kernelBoot`	`object`	Settings to set the kernel for booting.
`serial`	`string`	The system-serial-number in SMBIOS
`uuid`	`string`	UUID reported by the vmi bios. Defaults to a random generated uid.

.spec.template.spec.domain.firmware.acpi

Description

Information that can be set in the ACPI table

Type

`object`

Property	Type	Description
`msdmNameRef`	`string`	Similar to SlicNameRef, another ACPI entry that is used in more recent Windows versions. The above points to the spec of MSDM too.
`slicNameRef`	`string`	SlicNameRef should match the volume name of a secret object. The data in the secret should be a binary blob that follows the ACPI SLIC standard, see: https://learn.microsoft.com/en-us/previous-versions/windows/hardware/design/dn653305(v=vs.85) ↗

.spec.template.spec.domain.firmware.bootloader

Description

Settings to control the bootloader that is used.

Type

`object`

Property	Type	Description
`bios`	`object`	If set (default), BIOS will be used.
`efi`	`object`	If set, EFI will be used instead of BIOS.

.spec.template.spec.domain.firmware.bootloader.bios

Description

If set (default), BIOS will be used.

Type

`object`

Property	Type	Description
`useSerial`	`boolean`	If set, the BIOS output will be transmitted over serial

.spec.template.spec.domain.firmware.bootloader.efi

Description

If set, EFI will be used instead of BIOS.

Type

`object`

Property	Type	Description
`persistent`	`boolean`	If set to true, Persistent will persist the EFI NVRAM across reboots. Defaults to false
`secureBoot`	`boolean`	If set, SecureBoot will be enabled and the OVMF roms will be swapped for SecureBoot-enabled ones. Requires SMM to be enabled. Defaults to true

.spec.template.spec.domain.firmware.kernelBoot

Description

Settings to set the kernel for booting.

Type

`object`

Property	Type	Description
`container`	`object`	Container defines the container that containes kernel artifacts
`kernelArgs`	`string`	Arguments to be passed to the kernel at boot time

.spec.template.spec.domain.firmware.kernelBoot.container

Description

Container defines the container that containes kernel artifacts

Type

`object`

Required

`image`

Property	Type	Description
`image`	`string`	Image that contains initrd / kernel files.
`imagePullPolicy`	`string`	Image pull policy. One of Always, Never, IfNotPresent. Defa to Always if :latest tag is specified, or IfNotPresent otherwis Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#upd images ↗

Property	Type	Description
`imagePullSecret`	`string`	ImagePullSecret is the name of the Docker registry secret required to pull the image. The secret must already exist.
`initrdPath`	`string`	the fully-qualified path to the ramdisk image in the host OS
`kernelPath`	`string`	The fully-qualified path to the kernel image in the host OS

.spec.template.spec.domain.ioThreads

Description

IOThreads specifies the IOThreads options.

Type

`object`

Property	Type	Description
`supplementalPoolThreadCount`	`integer`	SupplementalPoolThreadCount specifies how many iothreads are allocated for the supplementalPool policy.

.spec.template.spec.domain.launchSecurity

Description

Launch Security setting of the vmi.

Type

`object`

Property	Type	Description
`sev`	`object`	AMD Secure Encrypted Virtualization (SEV).
`snp`	`object`	AMD SEV-SNP flags defined by the SEV-SNP specifications.
`tdx`	`object`	Intel Trust Domain Extensions (TDX).

.spec.template.spec.domain.launchSecurity.sev

Description

AMD Secure Encrypted Virtualization (SEV).

Type

`object`

Property	Type	Description
`attestation`	`object`	If specified, run the attestation process for a vmi.
`dhCert`	`string`	Base64 encoded guest owner's Diffie-Hellman key.

Property	Type	Description
`policy`	`object`	Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.
`session`	`string`	Base64 encoded session blob.

.spec.template.spec.domain.launchSecurity.sev.attestation

Description

If specified, run the attestation process for a vmi.

Type

`object`

.spec.template.spec.domain.launchSecurity.sev.policy

Description

Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.

Type

`object`

Property	Type	Description
`encryptedState`	`boolean`	SEV-ES is required. Defaults to false.

.spec.template.spec.domain.launchSecurity.snp

Description

AMD SEV-SNP flags defined by the SEV-SNP specifications.

Type

`object`

.spec.template.spec.domain.launchSecurity.tdx

Description

Intel Trust Domain Extensions (TDX).

Type

`object`

.spec.template.spec.domain.machine

Description

Machine type.

Type

`object`

Property	Type	Description
`type`	`string`	QEMU machine type is the actual chipset of the VirtualMachineInstance.

.spec.template.spec.domain.memory

Description

Memory allow specifying the VMI memory features.

Type

`object`

Property	Type	Description
`guest`	`integer|string`	Guest allows to specifying the amount of memory which is visible inside the Guest OS. The Guest must lie between Requests and Limits from the resources section. Defaults to the requested memory in the resources section if not specified.
`hugepages`	`object`	Hugepages allow to use hugepages for the VirtualMachineInstance instead of regular memory.
`maxGuest`	`integer|string`	MaxGuest allows to specify the maximum amount of memory which is visible inside the Guest OS. The delta between MaxGuest and Guest is the amount of memory that can be hot(un)plugged.

.spec.template.spec.domain.memory.hugepages

Description

Hugepages allow to use hugepages for the VirtualMachineInstance instead of regular memory.

Type

`object`

Property	Type	Description
`pageSize`	`string`	PageSize specifies the hugepage size, for x86_64 architecture valid values are 1Gi and 2Mi.

.spec.template.spec.domain.resources

Description

Resources describes the Compute Resources required by this vmi.

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. Valid resource keys are "memory" and "cpu".
`overcommitGuestOverhead`	`boolean`	Don't ask the scheduler to take the guest-management overhead into account. Instead put the overhead only into the container's memory limit. This can lead to crashes if all memory is in use on a node. Defaults to false.
`requests`	`object`	Requests is a description of the initial vmi resources. Valid resource keys are "memory" and "cpu".

.spec.template.spec.domain.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. Valid resource keys are "memory" and "cpu".

Type

`object`

.spec.template.spec.domain.resources.requests

Description

Requests is a description of the initial vmi resources. Valid resource keys are "memory" and "cpu".

Type

`object`

.spec.template.spec.livenessProbe

Description

Periodic probe of VirtualMachineInstance liveness. VirtualmachineInstances will be stopped if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Type

`object`

Property	Type	Description
`exec`	`object`	One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Property	Type	Description
`failureThreshold`	`integer`	Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3 Minimum value is 1.
`guestAgentPing`	`object`	GuestAgentPing contacts the qemu-guest-agent for availability checks.
`httpGet`	`object`	HTTPGet specifies the http request to perform.
`initialDelaySeconds`	`integer`	Number of seconds after the VirtualMachineInstance h started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗
`periodSeconds`	`integer`	How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.
`successThreshold`	`integer`	Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1 Must be 1 for liveness. Minimum value is 1.
`tcpSocket`	`object`	TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Property	Type	Description
`timeoutSeconds`	`integer`	Number of seconds after which the probe times out. Fo exec probes the timeout fails the probe but does not terminate the command running on the guest. This means a blocking command can result in an increasing load on the guest. A small buffer will be added to the resulting workload exec probe to compensate for delay caused by the qemu guest exec mechanism. Defaults 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/po lifecycle#container-probes ↗

.spec.template.spec.livenessProbe.exec

Description

One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Type

`object`

Property	Type	Description
`command`	`array`	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

.spec.template.spec.livenessProbe.exec.command

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

.spec.template.spec.livenessProbe.exec.command[]

Type

`string`

.spec.template.spec.livenessProbe.guestAgentPing

Description

GuestAgentPing contacts the qemu-guest-agent for availability checks.

Type

`object`

.spec.template.spec.livenessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
`httpHeaders`	`array`	Custom headers to set in the request. HTTP allows repeated headers.
`path`	`string`	Path to access on the HTTP server.
`port`	`integer|string`	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
`scheme`	`string`	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.template.spec.livenessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

.spec.template.spec.livenessProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
`name`	`string`	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
`value`	`string`	The header field value

.spec.template.spec.livenessProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Optional: Host name to connect to, defaults to the pod IP.
`port`	`integer|string`	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must

Property	Type	Description
		be an IANA_SVC_NAME.

.spec.template.spec.networks

Description

List of networks that can be attached to a vm's virtual interface.

Type

`array`

.spec.template.spec.networks[]

Description

Network represents a network type and a resource that should be connected to the vm.

Type

`object`

Required

`name`

Property	Type	Description
`multus`	`object`	Represents the multus cni network.
`name`	`string`	Network name. Must be a DNS_LABEL and unique within the vm. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗

Property	Type	Description
`pod`	`object`	Represents the stock pod network interface.

.spec.template.spec.networks[].multus

Description

Represents the multus cni network.

Type

`object`

Required

`networkName`

Property	Type	Description
`default`	`boolean`	Select the default network and add it to the multus-cni.io/default-network annotation.
`networkName`	`string`	References to a NetworkAttachmentDefinition CRD object. Format: , /. If namespace is not specified, VMI namespace is assumed.

.spec.template.spec.networks[].pod

Description

Represents the stock pod network interface.

Type

`object`

Property	Type	Description
`vmIPv6NetworkCIDR`	`string`	IPv6 CIDR for the vm network. Defaults to fd10:0:2::/120 if not specified.
`vmNetworkCIDR`	`string`	CIDR for vm network. Default 10.0.2.0/24 if not specified.

.spec.template.spec.nodeSelector

Description

NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/

Type

`object`

.spec.template.spec.readinessProbe

Description

Periodic probe of VirtualMachineInstance service readiness. VirtualmachineInstances will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Type

`object`

Property	Type	Description
`exec`	`object`	One and only one of the following should be specified. Exec specifies the action to take, it will be executed on

Property	Type	Description
		the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.
`failureThreshold`	`integer`	Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3 Minimum value is 1.
`guestAgentPing`	`object`	GuestAgentPing contacts the qemu-guest-agent for availability checks.
`httpGet`	`object`	HTTPGet specifies the http request to perform.
`initialDelaySeconds`	`integer`	Number of seconds after the VirtualMachineInstance h started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗
`periodSeconds`	`integer`	How often (in seconds) to perform the probe. Default t 10 seconds. Minimum value is 1.
`successThreshold`	`integer`	Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1 Must be 1 for liveness. Minimum value is 1.

Property	Type	Description
`tcpSocket`	`object`	TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported
`timeoutSeconds`	`integer`	Number of seconds after which the probe times out. Fo exec probes the timeout fails the probe but does not terminate the command running on the guest. This means a blocking command can result in an increasing load on the guest. A small buffer will be added to the resulting workload exec probe to compensate for delay caused by the qemu guest exec mechanism. Defaults 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗

.spec.template.spec.readinessProbe.exec

Description

One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Type

`object`

Property	Type	Description
`command`	`array`	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to

Property	Type	Description
		explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

.spec.template.spec.readinessProbe.exec.command

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

.spec.template.spec.readinessProbe.exec.command[]

Type

`string`

.spec.template.spec.readinessProbe.guestAgentPing

Description

GuestAgentPing contacts the qemu-guest-agent for availability checks.

Type

`object`

.spec.template.spec.readinessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
`httpHeaders`	`array`	Custom headers to set in the request. HTTP allows repeated headers.
`path`	`string`	Path to access on the HTTP server.
`port`	`integer|string`	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
`scheme`	`string`	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.template.spec.readinessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

.spec.template.spec.readinessProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
`name`	`string`	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
`value`	`string`	The header field value

.spec.template.spec.readinessProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Optional: Host name to connect to, defaults to the pod IP.
`port`	`integer|string`	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.template.spec.resourceClaims

Description

ResourceClaims define which ResourceClaims must be allocated and reserved before the VMI, hence virt-launcher pod is allowed to start. The resources will be made available to the domain which consumes them by name. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate in kubernetes https://kubernetes.io/docs/concepts/scheduling-eviction/dynamic-resource-allocation/ This field should only be configured if one of the feature-gates GPUsWithDRA or HostDevicesWithDRA is enabled. This feature is in alpha.

Type

`array`

.spec.template.spec.resourceClaims[]

Description

PodResourceClaim references exactly one ResourceClaim, either directly or by naming a ResourceClaimTemplate which is then turned into a ResourceClaim for the pod. It adds a name to it that uniquely identifies the ResourceClaim inside the Pod. Containers that need access to the ResourceClaim reference it with this name.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name uniquely identifies this resource claim inside the pod. This must be a DNS_LABEL.
`resourceClaimName`	`string`	ResourceClaimName is the name of a ResourceClaim object in the same namespace as this pod. Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.
`resourceClaimTemplateName`	`string`	ResourceClaimTemplateName is the name of a ResourceClaimTemplate object in the same namespace as this pod. The template will be used to create a new ResourceClaim, which will be bound to this pod. When this pod is deleted, the ResourceClaim will also be deleted. The pod name and resource name, along with a generated component, will be used to form a unique name for the ResourceClaim, which will be recorded in pod.status.resourceClaimStatuses. This field is immutable and no changes will be made to the corresponding

Property	Type	Description
		ResourceClaim by the control plane after creating the ResourceClaim. Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

.spec.template.spec.tolerations

Description

If toleration is specified, obey all the toleration rules.

Type

`array`

.spec.template.spec.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.

Property	Type	Description
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
`operator`	`string`	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
`tolerationSeconds`	`integer`	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.template.spec.topologySpreadConstraints

Description

TopologySpreadConstraints describes how a group of VMIs will be spread across a given topology domains. K8s scheduler will schedule VMI pods in a way which abides by the constraints.

Type

`array`

.spec.template.spec.topologySpreadConstraints[]

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

`object`

Required

`maxSkew` `topologyKey` `whenUnsatisfiable`

Property	Type	Description
`labelSelector`	`object`	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector.

Property	Type	Description
		This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
`maxSkew`	`integer`	MaxSkew describes the degree to which pods may be unevenly distributed. When 'whenUnsatisfiable=DoNotSchedule', it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. | zone1 | zone2 | zone3 | | P P | P P | P | • if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). • if MaxSkew is 2, incoming pod can be scheduled onto any zone. When 'whenUnsatisfiable=ScheduleAnyway', it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
`minDomains`	`integer`	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than

Property	Type	Description
		minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule. For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: | zone1 | zone2 | zone3 | | P P | P P | P P | The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
`nodeAffinityPolicy`	`string`	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: • Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. • Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations.

Property	Type	Description
		If this value is nil, the behavior is equivalent to the Honor policy.
`nodeTaintsPolicy`	`string`	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: • Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included. • Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy.
`topologyKey`	`string`	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.

Property	Type	Description
`whenUnsatisfiable`	`string`	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. • DoNotSchedule (default) tells the scheduler not to schedule it. • ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: | zone1 | zone2 | zone3 | | P P P | P | P | If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it *more* imbalanced. It's a required field.

.spec.template.spec.topologySpreadConstraints[].labelSelector

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.spec.topologySpreadConstraints[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.template.spec.topologySpreadConstraints[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.template.spec.topologySpreadConstraints[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.template.spec.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.spec.topologySpreadConstraints[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.template.spec.topologySpreadConstraints[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Type

`array`

.spec.template.spec.topologySpreadConstraints[].matchLabelKeys[]

Type

`string`

.spec.template.spec.volumes

Description

List of volumes that can be mounted by disks belonging to the vmi.

Type

`array`

.spec.template.spec.volumes[]

Description

Volume represents a named volume in a vmi.

Type

`object`

Required

`name`

Property	Type	Description
`cloudInitConfigDrive`	`object`	CloudInitConfigDrive represents a cloud-init Config D The Config Drive data will be added as a disk to the installation is required inside the guest. More info: https://cloudinit.readthedocs.io/en/latest/topics/datas ↗
`cloudInitNoCloud`	`object`	CloudInitNoCloud represents a cloud-init NoCloud u NoCloud data will be added as a disk to the vmi. A p installation is required inside the guest. More info: http://cloudinit.readthedocs.io/en/latest/topics/datasc

Property	Type	Description
`configMap`	`object`	ConfigMapSource represents a reference to a Confi namespace. More info: https://kubernetes.io/docs/ta container/configure-pod-configmap/ ↗
`containerDisk`	`object`	ContainerDisk references a docker image, embeddi More info: https://kubevirt.gitbooks.io/user-guide/reg
`dataVolume`	`object`	DataVolume represents the dynamic creation a PVC as the process of populating that PVC with a disk im
`downwardAPI`	`object`	DownwardAPI represents downward API about the p populate this volume
`downwardMetrics`	`object`	DownwardMetrics adds a very small disk to VMIs wh view of host and guest metrics. The disk content is c vhostmd (https://github.com/vhostmd/vhostmd ↗) an
`emptyDisk`	`object`	EmptyDisk represents a temporary disk which share More info: https://kubevirt.gitbooks.io/user-guide/disl
`ephemeral`	`object`	Ephemeral is a special volume source that "wraps" s provides copy-on-write image on top of it.

Property	Type	Description
`hostDisk`	`object`	HostDisk represents a disk created on the cluster lev
`memoryDump`	`object`	MemoryDump is attached to the virt launcher and is memory dump of the vmi
`name`	`string`	Volume's name. Must be a DNS_LABEL and unique info: https://kubernetes.io/docs/concepts/overview/w objects/names/#names ↗
`persistentVolumeClaim`	`object`	PersistentVolumeClaimVolumeSource represents a PersistentVolumeClaim in the same namespace. Dir vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persiste volumes#persistentvolumeclaims ↗
`secret`	`object`	SecretVolumeSource represents a reference to a se namespace. More info: https://kubernetes.io/docs/concepts/configuration/se
`serviceAccount`	`object`	ServiceAccountVolumeSource represents a referenc There can only be one volume of this type! More info https://kubernetes.io/docs/tasks/configure-pod-conta account/ ↗
`sysprep`	`object`	Represents a Sysprep volume source.

.spec.template.spec.volumes[].cloudInitConfigDrive

Description

CloudInitConfigDrive represents a cloud-init Config Drive user-data source. The Config Drive data will be added as a disk to the vmi. A proper cloud-init installation is required inside the guest. More info: https://cloudinit.readthedocs.io/en/latest/topics/datasources/configdrive.html

Type

`object`

Property	Type	Description
`networkData`	`string`	NetworkData contains config drive inline cloud-init networkdata.
`networkDataBase64`	`string`	NetworkDataBase64 contains config drive cloud-init networkdata as a base64 encoded string.
`networkDataSecretRef`	`object`	NetworkDataSecretRef references a k8s secret that contains config drive networkdata.
`secretRef`	`object`	UserDataSecretRef references a k8s secret that contains config drive userdata.
`userData`	`string`	UserData contains config drive inline cloud-init userdata.

Property	Type	Description
`userDataBase64`	`string`	UserDataBase64 contains config drive cloud-init userdata as a base64 encoded string.

.spec.template.spec.volumes[].cloudInitConfigDrive.networkDataSecretRef

Description

NetworkDataSecretRef references a k8s secret that contains config drive networkdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.volumes[].cloudInitConfigDrive.secretRef

Description

UserDataSecretRef references a k8s secret that contains config drive userdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.volumes[].cloudInitNoCloud

Description

CloudInitNoCloud represents a cloud-init NoCloud user-data source. The NoCloud data will be added as a disk to the vmi. A proper cloud-init installation is required inside the guest. More info: http://cloudinit.readthedocs.io/en/latest/topics/datasources/nocloud.html

Type

`object`

Property	Type	Description
`networkData`	`string`	NetworkData contains NoCloud inline cloud-init networkdata.
`networkDataBase64`	`string`	NetworkDataBase64 contains NoCloud cloud-init networkdata as a base64 encoded string.
`networkDataSecretRef`	`object`	NetworkDataSecretRef references a k8s secret that contains NoCloud networkdata.

Property	Type	Description
`secretRef`	`object`	UserDataSecretRef references a k8s secret that contains NoCloud userdata.
`userData`	`string`	UserData contains NoCloud inline cloud-init userdata.
`userDataBase64`	`string`	UserDataBase64 contains NoCloud cloud-init userdata as a base64 encoded string.

.spec.template.spec.volumes[].cloudInitNoCloud.networkDataSecretRef

Description

NetworkDataSecretRef references a k8s secret that contains NoCloud networkdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.volumes[].cloudInitNoCloud.secretRef

Description

UserDataSecretRef references a k8s secret that contains NoCloud userdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.volumes[].configMap

Description

ConfigMapSource represents a reference to a ConfigMap in the same namespace. More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`optional`	`boolean`	Specify whether the ConfigMap or it's keys must be defined
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.template.spec.volumes[].containerDisk

Description

ContainerDisk references a docker image, embedding a qcow or raw disk. More info: https://kubevirt.gitbooks.io/user-guide/registry-disk.html

Type

`object`

Required

`image`

Property	Type	Description
`image`	`string`	Image is the name of the image with the embedded disk.
`imagePullPolicy`	`string`	Image pull policy. One of Always, Never, IfNotPresent. Defa to Always if :latest tag is specified, or IfNotPresent otherwis

Property	Type	Description
		Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#upd images ↗
`imagePullSecret`	`string`	ImagePullSecret is the name of the Docker registry secret required to pull the image. The secret must already exist.
`path`	`string`	Path defines the path to disk file in the container

.spec.template.spec.volumes[].dataVolume

Description

DataVolume represents the dynamic creation a PVC for this volume as well as the process of populating that PVC with a disk image.

Type

`object`

Required

`name`

Property	Type	Description
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`name`	`string`	Name of both the DataVolume and the PVC in the same namespace.

.spec.template.spec.volumes[].downwardAPI

Description

DownwardAPI represents downward API about the pod that should populate this volume

Type

`object`

Property	Type	Description
`fields`	`array`	Fields is a list of downward API volume file
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.template.spec.volumes[].downwardAPI.fields

Description

Fields is a list of downward API volume file

Type

`array`

.spec.template.spec.volumes[].downwardAPI.fields[]

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

`object`

Required

`path`

Property	Type	Description
`fieldRef`	`object`	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.
`mode`	`integer`	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
`path`	`string`	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
`resourceFieldRef`	`object`	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

.spec.template.spec.volumes[].downwardAPI.fields[].fieldRef

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
`apiVersion`	`string`	Version of the schema the FieldPath is written in terms of, defaults to "v1".
`fieldPath`	`string`	Path of the field to select in the specified API version.

.spec.template.spec.volumes[].downwardAPI.fields[].resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
`containerName`	`string`	Container name: required for volumes, optional for env vars
`divisor`	`integer|string`	Specifies the output format of the exposed resources, defaults to "1"
`resource`	`string`	Required: resource to select

.spec.template.spec.volumes[].downwardMetrics

Description

DownwardMetrics adds a very small disk to VMIs which contains a limited view of host and guest metrics. The disk content is compatible with vhostmd (https://github.com/vhostmd/vhostmd) and vm-dump-metrics.

Type

`object`

.spec.template.spec.volumes[].emptyDisk

Description

EmptyDisk represents a temporary disk which shares the vmis lifecycle. More info: https://kubevirt.gitbooks.io/user-guide/disks-and-volumes.html

Type

`object`

Required

`capacity`

Property	Type	Description
`capacity`	`integer|string`	Capacity of the sparse disk.

.spec.template.spec.volumes[].ephemeral

Description

Ephemeral is a special volume source that "wraps" specified source and provides copy-on-write image on top of it.

Type

`object`

Property	Type	Description
`persistentVolumeClaim`	`object`	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistevolumes#persistentvolumeclaims ↗

.spec.template.spec.volumes[].ephemeral.persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.template.spec.volumes[].hostDisk

Description

HostDisk represents a disk created on the cluster level

Type

`object`

Required

`path` `type`

Property	Type	Description
`capacity`	`integer|string`	Capacity of the sparse disk
`path`	`string`	The path to HostDisk image located on the cluster

Property	Type	Description
`shared`	`boolean`	Shared indicate whether the path is shared between nodes
`type`	`string`	Contains information if disk.img exists or should be created allowed options are 'Disk' and 'DiskOrCreate'

.spec.template.spec.volumes[].memoryDump

Description

MemoryDump is attached to the virt launcher and is populated with a memory dump of the vmi

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.

Property	Type	Description
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.template.spec.volumes[].persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.

Property	Type	Description
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.template.spec.volumes[].secret

Description

SecretVolumeSource represents a reference to a secret data in the same namespace. More info: https://kubernetes.io/docs/concepts/configuration/secret/

Type

`object`

Property	Type	Description
`optional`	`boolean`	Specify whether the Secret or it's keys must be defined
`secretName`	`string`	Name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret ↗
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.template.spec.volumes[].serviceAccount

Description

ServiceAccountVolumeSource represents a reference to a service account. There can only be one volume of this type! More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/

Type

`object`

Property	Type	Description
`serviceAccountName`	`string`	Name of the service account in the pod's namespace to use. More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/ ↗

.spec.template.spec.volumes[].sysprep

Description

Represents a Sysprep volume source.

Type

`object`

Property	Type	Description
`configMap`	`object`	ConfigMap references a ConfigMap that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.
`secret`	`object`	Secret references a k8s Secret that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

.spec.template.spec.volumes[].sysprep.configMap

Description

ConfigMap references a ConfigMap that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.template.spec.volumes[].sysprep.secret

Description

Secret references a k8s Secret that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.status

Description

Status holds the current state of the controller and brief information about its associated VirtualMachineInstance

Type

`object`

Property	Type	Description
`changedBlockTracking`	`object`	ChangedBlockTracking represents the status of the changedBlockTracking
`conditions`	`array`	Hold the state information of the VirtualMachine and its VirtualMachineInstance
`created`	`boolean`	Created indicates if the virtual machine is created in the cluster
`desiredGeneration`	`integer`	DesiredGeneration is the generation which is desired for the VMI. This will be used in comparisons with ObservedGeneration to understand when the VMI is out of sync. This will be changed at the same time as ObservedGeneration to remove errors which

Property	Type	Description
		could occur if Generation is updated through an Update() before ObservedGeneration in Status.
`instancetypeRef`	`object`	InstancetypeRef captures the state of any referenced instance type from the VirtualMachine
`memoryDumpRequest`	`object`	MemoryDumpRequest tracks memory dump request phase and info of getting a memory dump to the given pvc
`observedGeneration`	`integer`	ObservedGeneration is the generation observed by the vmi when started.
`preferenceRef`	`object`	PreferenceRef captures the state of any referenced preference from the VirtualMachine
`printableStatus`	`string`	PrintableStatus is a human readable, high-level representation of the status of the virtual machine
`ready`	`boolean`	Ready indicates if the virtual machine is running and ready

Property	Type	Description
`restoreInProgress`	`string`	RestoreInProgress is the name of the VirtualMachineRestore currently executing
`runStrategy`	`string`	RunStrategy tracks the last recorded RunStrategy used by the VM. This is needed to correctly process the next strategy (for now only the RerunOnFailure)
`snapshotInProgress`	`string`	SnapshotInProgress is the name of the VirtualMachineSnapshot currently executing
`startFailure`	`object`	StartFailure tracks consecutive VMI startup failures for the purposes of crash loop backoffs
`stateChangeRequests`	`array`	StateChangeRequests indicates a list of actions that should be taken on a VMI e.g. stop a specific VMI then start a new one.
`volumeRequests`	`array`	VolumeRequests indicates a list of volumes add or remove from the VMI template and hotplug on an active running VMI.

Property	Type	Description
`volumeSnapshotStatuses`	`array`	VolumeSnapshotStatuses indicates a list of statuses whether snapshotting is supported by each volume.
`volumeUpdateState`	`object`	VolumeUpdateState contains the information about the volumes set updates related to the volumeUpdateStrategy

.status.changedBlockTracking

Description

ChangedBlockTracking represents the status of the changedBlockTracking

Type

`object`

Required

`state`

Property	Type	Description
`state`	`string`	State represents the current CBT state

.status.conditions

Description

Hold the state information of the VirtualMachine and its VirtualMachineInstance

Type

`array`

.status.conditions[]

Description

VirtualMachineCondition represents the state of VirtualMachine

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	

.status.instancetypeRef

Description

InstancetypeRef captures the state of any referenced instance type from the VirtualMachine

Type

`object`

Property	Type	Description
`controllerRevisionRef`	`object`	ControllerRef specifies the ControllerRevision storing a copy of

Property	Type	Description
		the object captured when it is first seen by the VirtualMachine controller
`inferFromVolume`	`string`	InferFromVolume lists the name of a volume that should be used to infer or discover the resource
`inferFromVolumeFailurePolicy`	`string`	InferFromVolumeFailurePolicy controls what should happen on failure when inferring the resource
`kind`	`string`	Kind specifies the kind of resource
`name`	`string`	Name is the name of resource

.status.instancetypeRef.controllerRevisionRef

Description

ControllerRef specifies the ControllerRevision storing a copy of the object captured when it is first seen by the VirtualMachine controller

Type

`object`

Property	Type	Description
`name`	`string`	Name of the ControllerRevision

.status.memoryDumpRequest

Description

MemoryDumpRequest tracks memory dump request phase and info of getting a memory dump to the given pvc

Type

`object`

Required

`claimName` `phase`

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the pvc that will contain the memory dump
`endTimestamp`	`string`	EndTimestamp represents the time the memory dump was completed
`fileName`	`string`	FileName represents the name of the output file
`message`	`string`	Message is a detailed message about failure of the memory dump
`phase`	`string`	Phase represents the memory dump phase
`remove`	`boolean`	Remove represents request of dissociating the memory dump pvc

Property	Type	Description
`startTimestamp`	`string`	StartTimestamp represents the time the memory dump started

.status.preferenceRef

Description

PreferenceRef captures the state of any referenced preference from the VirtualMachine

Type

`object`

Property	Type	Description
`controllerRevisionRef`	`object`	ControllerRef specifies the ControllerRevision storing a copy of the object captured when it is first seen by the VirtualMachine controller
`inferFromVolume`	`string`	InferFromVolume lists the name of a volume that should be used to infer or discover the resource
`inferFromVolumeFailurePolicy`	`string`	InferFromVolumeFailurePolicy controls what should happen on failure when inferring the resource
`kind`	`string`	Kind specifies the kind of resource

Property	Type	Description
`name`	`string`	Name is the name of resource

.status.preferenceRef.controllerRevisionRef

Description

ControllerRef specifies the ControllerRevision storing a copy of the object captured when it is first seen by the VirtualMachine controller

Type

`object`

Property	Type	Description
`name`	`string`	Name of the ControllerRevision

.status.startFailure

Description

StartFailure tracks consecutive VMI startup failures for the purposes of crash loop backoffs

Type

`object`

Property	Type	Description
`consecutiveFailCount`	`integer`	
`lastFailedVMIUID`	`string`	UID is a type that holds unique ID values, including UUIDs. Because we don't ONLY use UUIDs, this is an alias to string. Being a type

Property	Type	Description
		captures intent and helps make sure that UIDs and names do not get conflated.
`retryAfterTimestamp`	`string`	

.status.stateChangeRequests

Description

StateChangeRequests indicates a list of actions that should be taken on a VMI e.g. stop a specific VMI then start a new one.

Type

`array`

.status.stateChangeRequests[]

Type

`object`

Required

`action`

Property	Type	Description
`action`	`string`	Indicates the type of action that is requested. e.g. Start or Stop
`data`	`object`	Provides additional data in order to perform the Action

Property	Type	Description
`uid`	`string`	Indicates the UUID of an existing Virtual Machine Instance that this change request applies to -- if applicable

.status.stateChangeRequests[].data

Description

Provides additional data in order to perform the Action

Type

`object`

.status.volumeRequests

Description

VolumeRequests indicates a list of volumes add or remove from the VMI template and hotplug on an active running VMI.

Type

`array`

.status.volumeRequests[]

Type

`object`

Property	Type	Description
`addVolumeOptions`	`object`	AddVolumeOptions when set indicates a volume should be added. The details within this field specify how to add the volume

Property	Type	Description
`removeVolumeOptions`	`object`	RemoveVolumeOptions when set indicates a volume should be removed. The details within this field specify how to add the volume

.status.volumeRequests[].addVolumeOptions

Description

AddVolumeOptions when set indicates a volume should be added. The details within this field specify how to add the volume

Type

`object`

Required

`disk` `name` `volumeSource`

Property	Type	Description
`disk`	`object`	Disk represents the hotplug disk that will be plugged into the running VMI
`dryRun`	`array`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: • All: all dry run stages will be processed
`name`	`string`	Name represents the name that will be used to map the disk to the corresponding volume. This overrides any

Property	Type	Description
		name set inside the Disk struct itself.
`volumeSource`	`object`	VolumeSource represents the source of the volume to map to the disk.

.status.volumeRequests[].addVolumeOptions.disk

Description

Disk represents the hotplug disk that will be plugged into the running VMI

Type

`object`

Required

`name`

Property	Type	Description
`blockSize`	`object`	If specified, the virtual disk will be presented with the given block sizes.
`bootOrder`	`integer`	BootOrder is an integer value > 0, used to determine ordering of boot devices. Lower values take precedence. Each disk or interface that has a boot order must have a unique value. Disks without a boot order are not tried if a disk with a boot order exists.
`cache`	`string`	Cache specifies which kvm disk cache mode should be used. Supported values are: none:

Property	Type	Description
		Guest I/O not cached on the host, but may be kept in a disk cache. writethrough: Guest I/O cached on the host but written through to the physical medium. Slowest but with most guarantees. writeback: Guest I/O cached on the host. Defaults to none if the storage supports O_DIRECT, otherwise writethrough.
`cdrom`	`object`	Attach a volume as a cdrom to the vmi.
`changedBlockTracking`	`boolean`	ChangedBlockTracking indicates this disk should have CBT option Defaults to false.
`dedicatedIOThread`	`boolean`	dedicatedIOThread indicates this disk should have an exclusive IO Thread. Enabling this implies useIOThreads = true. Defaults to false.
`disk`	`object`	Attach a volume as a disk to the vmi.
`errorPolicy`	`string`	If specified, it can change the default error policy (stop) for the disk
`io`	`string`	IO specifies which QEMU disk IO mode should be used. Supported values are: native, default, threads.

Property	Type	Description
`lun`	`object`	Attach a volume as a LUN to the vmi.
`name`	`string`	Name is the device name
`serial`	`string`	Serial provides the ability to specify a serial number for the disk device.
`shareable`	`boolean`	If specified the disk is made sharable and multiple write from different VMs are permitted
`tag`	`string`	If specified, disk address and its tag will be provided to the guest via config drive metadata

.status.volumeRequests[].addVolumeOptions.disk.blockSize

Description

If specified, the virtual disk will be presented with the given block sizes.

Type

`object`

Property	Type	Description
`custom`	`object`	CustomBlockSize represents the desired logical and physical block size for a VM disk.
`matchVolume`	`object`	Represents if a feature is enabled or disabled.

.status.volumeRequests[].addVolumeOptions.disk.blockSize.custom

Description

CustomBlockSize represents the desired logical and physical block size for a VM disk.

Type

`object`

Property	Type	Description
`discardGranularity`	`integer`	
`logical`	`integer`	
`physical`	`integer`	

.status.volumeRequests[].addVolumeOptions.disk.blockSize.matchVolume

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.status.volumeRequests[].addVolumeOptions.disk.cdrom

Description

Attach a volume as a cdrom to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to true.
`tray`	`string`	Tray indicates if the tray of the device is open or closed. Allowed values are "open" and "closed". Defaults to closed.

.status.volumeRequests[].addVolumeOptions.disk.disk

Description

Attach a volume as a disk to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi, usb.
`pciAddress`	`string`	If specified, the virtual disk will be placed on the guests pci address with the specified PCI address. For example: 0000:81:01.10
`readonly`	`boolean`	ReadOnly. Defaults to false.

.status.volumeRequests[].addVolumeOptions.disk.lun

Description

Attach a volume as a LUN to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to false.
`reservation`	`boolean`	Reservation indicates if the disk needs to support the persistent reservation for the SCSI disk

.status.volumeRequests[].addVolumeOptions.dryRun

Description

When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Type

`array`

.status.volumeRequests[].addVolumeOptions.dryRun[]

Type

`string`

.status.volumeRequests[].addVolumeOptions.volumeSource

Description

VolumeSource represents the source of the volume to map to the disk.

Type

`object`

Property	Type	Description
`dataVolume`	`object`	DataVolume represents the dynamic creation a PVC for this volume as well as the process of populating that PVC with a disk image.
`persistentVolumeClaim`	`object`	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persiste volumes#persistentvolumeclaims ↗

.status.volumeRequests[].addVolumeOptions.volumeSource.dataVolume

Description

DataVolume represents the dynamic creation a PVC for this volume as well as the process of populating that PVC with a disk image.

Type

`object`

Required

`name`

Property	Type	Description
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`name`	`string`	Name of both the DataVolume and the PVC in the same namespace.

.status.volumeRequests[].addVolumeOptions.volumeSource.persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info:

https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.status.volumeRequests[].removeVolumeOptions

Description

RemoveVolumeOptions when set indicates a volume should be removed. The details within this field specify how to add the volume

Type

`object`

Required

`name`

Property	Type	Description
`dryRun`	`array`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: • All: all dry run stages will be processed
`name`	`string`	Name represents the name that maps to both the disk and volume that should be removed

.status.volumeRequests[].removeVolumeOptions.dryRun

Description

When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Type

`array`

.status.volumeRequests[].removeVolumeOptions.dryRun[]

Type

`string`

.status.volumeSnapshotStatuses

Description

VolumeSnapshotStatuses indicates a list of statuses whether snapshotting is supported by each volume.

Type

`array`

.status.volumeSnapshotStatuses[]

Type

`object`

Required

`enabled` `name`

Property	Type	Description
`enabled`	`boolean`	True if the volume supports snapshotting
`name`	`string`	Volume name
`reason`	`string`	Empty if snapshotting is enabled, contains reason otherwise

.status.volumeUpdateState

Description

VolumeUpdateState contains the information about the volumes set updates related to the volumeUpdateStrategy

Type

`object`

Property	Type	Description
`volumeMigrationState`	`object`	VolumeMigrationState tracks the information related to the volume migration

.status.volumeUpdateState.volumeMigrationState

Description

VolumeMigrationState tracks the information related to the volume migration

Type

`object`

Property	Type	Description
`migratedVolumes`	`array`	MigratedVolumes lists the source and destination volumes during the volume migration

.status.volumeUpdateState.volumeMigrationState.migratedVolumes

Description

MigratedVolumes lists the source and destination volumes during the volume migration

Type

`array`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[]

Description

StorageMigratedVolumeInfo tracks the information about the source and destination volumes during the volume migration

Type

`object`

Required

`volumeName`

Property	Type	Description
`destinationPVCInfo`	`object`	DestinationPVCInfo contains the information about the destination PVC
`sourcePVCInfo`	`object`	SourcePVCInfo contains the information about the source PVC
`volumeName`	`string`	VolumeName is the name of the volume that is being migrated

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].destinationPVCInfo

Description

DestinationPVCInfo contains the information about the destination PVC

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`capacity`	`object`	Capacity represents the capacity set on the corresponding PVC status
`claimName`	`string`	ClaimName is the name of the PVC
`filesystemOverhead`	`string`	Percentage of filesystem's size to be reserved when resizing the PVC
`preallocated`	`boolean`	Preallocated indicates if the PVC's storage is preallocated or not
`requests`	`object`	Requests represents the resources requested by the corresponding PVC spec
`volumeMode`	`string`	VolumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].destinationPVCInfo.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].destinationPVCInfo.accessModes[]

Type

`string`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].destinationPVCInfo.capacity

Description

Capacity represents the capacity set on the corresponding PVC status

Type

`object`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].destinationPVCInfo.requests

Description

Requests represents the resources requested by the corresponding PVC spec

Type

`object`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].sourcePVCInfo

Description

SourcePVCInfo contains the information about the source PVC

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`capacity`	`object`	Capacity represents the capacity set on the corresponding PVC status
`claimName`	`string`	ClaimName is the name of the PVC
`filesystemOverhead`	`string`	Percentage of filesystem's size to be reserved when resizing the PVC
`preallocated`	`boolean`	Preallocated indicates if the PVC's storage is preallocated or not

Property	Type	Description
`requests`	`object`	Requests represents the resources requested by the corresponding PVC spec
`volumeMode`	`string`	VolumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].sourcePVCInfo.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].sourcePVCInfo.accessModes[]

Type

`string`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].sourcePVCInfo.capacity

Description

Capacity represents the capacity set on the corresponding PVC status

Type

`object`

.status.volumeUpdateState.volumeMigrationState.migratedVolumes[].sourcePVCInfo.requests

Description

Requests represents the resources requested by the corresponding PVC spec

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines`
  - `DELETE`: delete collection of VirtualMachine
  - `GET`: list objects of kind VirtualMachine
  - `POST`: create a new VirtualMachine
- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines/{name}`
  - `DELETE`: delete the specified VirtualMachine
  - `GET`: read the specified VirtualMachine
  - `PATCH`: partially update the specified VirtualMachine
  - `PUT`: replace the specified VirtualMachine
- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines/{name}/status`
  - `GET`: read status of the specified VirtualMachine
  - `PATCH`: partially update status of the specified VirtualMachine
  - `PUT`: replace status of the specified VirtualMachine

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines

HTTP method

`DELETE`

Description

delete collection of VirtualMachine

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachine

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
201 - Created	`VirtualMachine` schema

HTTP code	Response body
202 - Accepted	`VirtualMachine` schema
401 - Unauthorized	Empty

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachine

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
201 - Created	`VirtualMachine` schema
401 - Unauthorized	Empty

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachines/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachine

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachine

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachine` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachine` schema
201 - Created	`VirtualMachine` schema
401 - Unauthorized	Empty

VirtualMachineInstance [virtualmachineinstances.kubevirt.io/v1]

Description

VirtualMachineInstance is *the* VirtualMachineInstance Definition. It represents a virtual machine in the runtime environment of kubernetes.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineInstance Spec contains the VirtualMachineInstance specification.
`status`	`object`	Status is the high level overview of how the VirtualMachineInstance is doing. It contains information available to controllers and users.

.spec

Description

VirtualMachineInstance Spec contains the VirtualMachineInstance specification.

Type

`object`

Required

`domain`

Property	Type	Description
`accessCredentials`	`array`	Specifies a set of public keys to inject into
`affinity`	`object`	If affinity is specifies, obey all the affinity ru
`architecture`	`string`	Specifies the architecture of the vm guest y architecture of the KubeVirt components
`dnsConfig`	`object`	Specifies the DNS parameters of a pod. Pa generated DNS configuration based on DN
`dnsPolicy`	`string`	Set DNS policy for the pod. Defaults to "Cl 'ClusterFirstWithHostNet', 'ClusterFirst', 'De DNSConfig will be merged with the policy s along with hostNetwork, you have to speci
`domain`	`object`	Specification of the desired behavior of the
`evictionStrategy`	`string`	EvictionStrategy describes the strategy to t options are: • "None": No action will be taken, accordi VirtualMachine will be restarted or shut • "LiveMigrate": the VirtualMachineInstan • "LiveMigrateIfPossible": the same as "L Migratable, otherwise it will behave as "

Property	Type	Description
		• "External": the VirtualMachineInstance v 'vmi.Status.EvacuationNodeName' will l api-provider-kubevirt (capk) which need signal capk that eviction has been calle tearing the VMI down. Details can be fo https://github.com/kubevirt/kubevirt/com ↗.
`hostname`	`string`	Specifies the hostname of the vmi If not sp vmi, if dhcp or cloud-init is configured prop
`livenessProbe`	`object`	Periodic probe of VirtualMachineInstance l the probe fails. Cannot be updated. More i https://kubernetes.io/docs/concepts/worklo
`networks`	`array`	List of networks that can be attached to a v
`nodeSelector`	`object`	NodeSelector is a selector which must be t must match a node's labels for the vmi to b https://kubernetes.io/docs/concepts/configu
`priorityClassName`	`string`	If specified, indicates the pod's priority. If n there is no default.

Property	Type	Description
`readinessProbe`	`object`	Periodic probe of VirtualMachineInstance s removed from service endpoints if the prob https://kubernetes.io/docs/concepts/worklo
`resourceClaims`	`array`	ResourceClaims define which ResourceCl VMI, hence virt-launcher pod is allowed to domain which consumes them by name. This is an alpha field and requires enablinç kubernetes https://kubernetes.io/docs/conc allocation/↗ This field should only be confi HostDevicesWithDRA is enabled. This feat
`schedulerName`	`string`	If specified, the VMI will be dispatched by s dispatched by default scheduler.
`startStrategy`	`string`	StartStrategy can be set to "Paused" if Virt
`subdomain`	`string`	If specified, the fully qualified vmi hostnam have a domainname at all. The DNS entry can pick up a hostname.
`terminationGracePeriodSeconds`	`integer`	Grace period observed after signalling a Vi VirtualMachineInstance is force terminated

Property	Type	Description
`tolerations`	`array`	If toleration is specified, obey all the tolerat
`topologySpreadConstraints`	`array`	TopologySpreadConstraints describes how topology domains. K8s scheduler will sche constraints.
`volumes`	`array`	List of volumes that can be mounted by dis

.spec.accessCredentials

Description

Specifies a set of public keys to inject into the vm guest

Type

`array`

.spec.accessCredentials[]

Description

AccessCredential represents a credential source that can be used to authorize remote access to the vm guest Only one of its members may be specified.

Type

`object`

Property	Type	Description
`sshPublicKey`	`object`	SSHPublicKey represents the source and method of applying a ssh public key into a guest virtual machine.
`userPassword`	`object`	UserPassword represents the source and method for applying a guest user's password

.spec.accessCredentials[].sshPublicKey

Description

SSHPublicKey represents the source and method of applying a ssh public key into a guest virtual machine.

Type

`object`

Required

`propagationMethod` `source`

Property	Type	Description
`propagationMethod`	`object`	PropagationMethod represents how the public key is injected into the vm guest.
`source`	`object`	Source represents where the public keys are pulled from

.spec.accessCredentials[].sshPublicKey.propagationMethod

Description

PropagationMethod represents how the public key is injected into the vm guest.

Type

`object`

Property	Type	Description
`configDrive`	`object`	ConfigDrivePropagation means that the ssh public keys are injected into the VM using metadata using the configDrive cloud-init provider
`noCloud`	`object`	NoCloudPropagation means that the ssh public keys are injected into the VM using metadata using the noCloud cloud-init provider
`qemuGuestAgent`	`object`	QemuGuestAgentAccessCredentailPropagation means ssh public keys are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

.spec.accessCredentials[].sshPublicKey.propagationMethod.configDrive

Description

ConfigDrivePropagation means that the ssh public keys are injected into the VM using metadata using the configDrive cloud-init provider

Type

`object`

.spec.accessCredentials[].sshPublicKey.propagationMethod.noCloud

Description

NoCloudPropagation means that the ssh public keys are injected into the VM using metadata using the noCloud cloud-init provider

Type

`object`

.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent

Description

QemuGuestAgentAccessCredentailPropagation means ssh public keys are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

Type

`object`

Required

`users`

Property	Type	Description
`users`	`array`	Users represents a list of guest users that should have the ssh public keys added to their authorized_keys file.

.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent.users

Description

Users represents a list of guest users that should have the ssh public keys added to their authorized_keys file.

Type

`array`

.spec.accessCredentials[].sshPublicKey.propagationMethod.qemuGuestAgent.users[]

Type

`string`

.spec.accessCredentials[].sshPublicKey.source

Description

Source represents where the public keys are pulled from

Type

`object`

Property	Type	Description
`secret`	`object`	Secret means that the access credential is pulled from a kubernetes secret

.spec.accessCredentials[].sshPublicKey.source.secret

Description

Secret means that the access credential is pulled from a kubernetes secret

Type

`object`

Required

`secretName`

Property	Type	Description
`secretName`	`string`	SecretName represents the name of the secret in the VMI's namespace

.spec.accessCredentials[].userPassword

Description

UserPassword represents the source and method for applying a guest user's password

Type

`object`

Required

`propagationMethod` `source`

Property	Type	Description
`propagationMethod`	`object`	propagationMethod represents how the user passwords are injected into the vm guest.
`source`	`object`	Source represents where the user passwords are pulled from

.spec.accessCredentials[].userPassword.propagationMethod

Description

propagationMethod represents how the user passwords are injected into the vm guest.

Type

`object`

Property	Type	Description
`qemuGuestAgent`	`object`	QemuGuestAgentAccessCredentailPropagation means passwords are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

.spec.accessCredentials[].userPassword.propagationMethod.qemuGuestAgent

Description

QemuGuestAgentAccessCredentailPropagation means passwords are dynamically injected into the vm at runtime via the qemu guest agent. This feature requires the qemu guest agent to be running within the guest.

Type

`object`

.spec.accessCredentials[].userPassword.source

Description

Source represents where the user passwords are pulled from

Type

`object`

Property	Type	Description
`secret`	`object`	Secret means that the access credential is pulled from a kubernetes secret

.spec.accessCredentials[].userPassword.source.secret

Description

Secret means that the access credential is pulled from a kubernetes secret

Type

`object`

Required

`secretName`

Property	Type	Description
`secretName`	`string`	SecretName represents the name of the secret in the VMI's namespace

.spec.affinity

Description

If affinity is specifies, obey all the affinity rules

Type

`object`

Property	Type	Description
`nodeAffinity`	`object`	Describes node affinity scheduling rules for the pod.
`podAffinity`	`object`	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
`podAntiAffinity`	`object`	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

.spec.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the affinity expressions specified by this field, it may choose a node violates one or more the expressions. The node that is most

Property	Type	Description
		preferred is the one v the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched affinity expressions, e compute a sum by iterating through the elements of this field adding "weight" to the sum if the node matc the corresponding matchExpressions; th node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`object`	If the affinity requirem specified by this field not met at scheduling time, the pod will not scheduled onto the n If the affinity requirem specified by this field cease to be met at so point during pod execution (e.g. due to update), the system r or may not try to eventually evict the p from its node.

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

`object`

Required

`preference` `weight`

Property	Type	Description
`preference`	`object`	A node selector term, associated with the corresponding weight.

Property	Type	Description
`weight`	`integer`	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference

Description

A node selector term, associated with the corresponding weight.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values[]

Type

`string`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values[]

Type

`string`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

`object`

Required

`nodeSelectorTerms`

Property	Type	Description
`nodeSelectorTerms`	`array`	Required. A list of node selector terms. The terms are ORed.

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms

Description

Required. A list of node selector terms. The terms are ORed.

Type

`array`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[]

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressio

ns[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values[]

Type

`string`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values[]

Type

`string`

.spec.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the affinity expressions specified by this field, it may choose a node violates one or more the expressions. The node that is most preferred is the one v the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched affinity expressions, e

Property	Type	Description
		compute a sum by iterating through the elements of this field adding "weight" to the sum if the node has p which matches the corresponding podAffinityTerm; the node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the affinity requirem specified by this field not met at scheduling time, the pod will not scheduled onto the n If the affinity requirem specified by this field cease to be met at sc point during pod execution (e.g. due to pod label update), the system may or may n try to eventually evict pod from its node. Wh there are multiple elements, the lists of nodes corresponding each podAffinityTerm intersected, i.e. all ter must be satisfied.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.

Property	Type	Description
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both

Property	Type	Description
		matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null

Property	Type	Description
		or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is

Property	Type	Description
		"key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.

matchExpressions[].values[]

Type

`string`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with

Property	Type	Description
		no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Property	Type	Description
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].val

ues[]

Type

`string`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpression

s[].values[]

Type

`string`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will pr to schedule pods to nodes that satisfy the anti-affinity expressio specified by this field, it may choose a node violates one or more the expressions. The node that is most preferred is the one v the greatest sum of weights, i.e. for each node that meets all o scheduling requireme (resource request, requiredDuringSched anti-affinity expressio etc.), compute a sum iterating through the elements of this field adding "weight" to the sum if the node has p which matches the corresponding podAffinityTerm; the

Property	Type	Description
		node(s) with the high sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the anti-affinity requirements specifie this field are not met scheduling time, the p will not be scheduled the node. If the anti-affinity requirements specified by this field cease to be met at sc point during pod execution (e.g. due to pod label update), the system may or may n try to eventually evict pod from its node. Wh there are multiple elements, the lists of nodes corresponding each podAffinityTerm intersected, i.e. all ter must be satisfied.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the

expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to

Property	Type	Description
		select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelec

tor.matchExpressions[].values[]

Type

`string`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with

Property	Type	Description
		no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Property	Type	Description
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

.values[]

Type

`string`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key in (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with 'labelSelector' as 'key notin (value)' to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpres

sions[].values[]

Type

`string`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.dnsConfig

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
`nameservers`	`array`	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.
`options`	`array`	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
`searches`	`array`	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

.spec.dnsConfig.nameservers

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

.spec.dnsConfig.nameservers[]

Type

`string`

.spec.dnsConfig.options

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

`array`

.spec.dnsConfig.options[]

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

`object`

Property	Type	Description
`name`	`string`	Name is this DNS resolver option's name. Required.
`value`	`string`	Value is this DNS resolver option's value.

.spec.dnsConfig.searches

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

`array`

.spec.dnsConfig.searches[]

Type

`string`

.spec.domain

Description

Specification of the desired behavior of the VirtualMachineInstance on the host.

Type

`object`

Required

`devices`

Property	Type	Description
`chassis`	`object`	Chassis specifies the chassis info passed to the domain.
`clock`	`object`	Clock sets the clock and timers of the vmi.
`cpu`	`object`	CPU allow specified the detailed CPU topology inside the vmi.

Property	Type	Description
`devices`	`object`	Devices allows adding disks, network interfaces, and others
`features`	`object`	Features like acpi, apic, hyperv, smm.
`firmware`	`object`	Firmware.
`ioThreads`	`object`	IOThreads specifies the IOThreads options.
`ioThreadsPolicy`	`string`	Controls whether or not disks will share IOThreads. Omitting IOThreadsPolicy disables use of IOThreads. One of: shared, auto, supplementalPool
`launchSecurity`	`object`	Launch Security setting of the vmi.
`machine`	`object`	Machine type.
`memory`	`object`	Memory allow specifying the VMI memory features.
`resources`	`object`	Resources describes the Compute Resources required by this vmi.

.spec.domain.chassis

Description

Chassis specifies the chassis info passed to the domain.

Type

`object`

Property	Type	Description
`asset`	`string`	
`manufacturer`	`string`	
`serial`	`string`	
`sku`	`string`	
`version`	`string`	

.spec.domain.clock

Description

Clock sets the clock and timers of the vmi.

Type

`object`

Property	Type	Description
`timer`	`object`	Timer specifies whih timers are attached to the vmi.

Property	Type	Description
`timezone`	`string`	Timezone sets the guest clock to the specified timezone. Zone name follows the TZ environment variable format (e.g. 'America/New_York').
`utc`	`object`	UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

.spec.domain.clock.timer

Description

Timer specifies whih timers are attached to the vmi.

Type

`object`

Property	Type	Description
`hpet`	`object`	HPET (High Precision Event Timer) - multiple timers with periodic interrupts.
`hyperv`	`object`	Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.
`kvm`	`object`	KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.

Property	Type	Description
`pit`	`object`	PIT (Programmable Interval Timer) - a timer with periodic interrupts.
`rtc`	`object`	RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

.spec.domain.clock.timer.hpet

Description

HPET (High Precision Event Timer) - multiple timers with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "merge", "discard".

.spec.domain.clock.timer.hyperv

Description

Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.domain.clock.timer.kvm

Description

KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.domain.clock.timer.pit

Description

PIT (Programmable Interval Timer) - a timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "discard".

.spec.domain.clock.timer.rtc

Description

RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup".
`track`	`string`	Track the guest or the wall clock.

.spec.domain.clock.utc

Description

UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

Type

`object`

Property	Type	Description
`offsetSeconds`	`integer`	OffsetSeconds specifies an offset in seconds, relative to UTC. If set, guest changes to the clock will be kept during reboots and not reset.

.spec.domain.cpu

Description

CPU allow specified the detailed CPU topology inside the vmi.

Type

`object`

Property	Type	Description
`cores`	`integer`	Cores specifies the number of cores inside the vmi. Must be a value greater or equal 1.
`dedicatedCpuPlacement`	`boolean`	DedicatedCPUPlacement requests the scheduler to place the VirtualMachineInstance on a node with enough dedicated pCPUs and pin the vCPUs to it.

Property	Type	Description
`features`	`array`	Features specifies the CPU features list inside the VMI.
`isolateEmulatorThread`	`boolean`	IsolateEmulatorThread requests one more dedicated pCPU to be allocated for the VMI to place the emula thread on it.
`maxSockets`	`integer`	MaxSockets specifies the maximum amount of sock that can be hotplugged
`model`	`string`	Model specifies the CPU model inside the VMI. List available models https://github.com/libvirt/libvirt/tree/master/src/cpu_m ↗. It is possible to specify special cases like "host-passthrough" to get the same CPU as the node and "host-model" to get CPU closest to the node one. Defaults to host-model.
`numa`	`object`	NUMA allows specifying settings for the guest NUMA topology
`realtime`	`object`	Realtime instructs the virt-launcher to tune the VMI f lower latency, optional for real time workloads

Property	Type	Description
`sockets`	`integer`	Sockets specifies the number of sockets inside the vmi. Must be a value greater or equal 1.
`threads`	`integer`	Threads specifies the number of threads inside the vmi. Must be a value greater or equal 1.

.spec.domain.cpu.features

Description

Features specifies the CPU features list inside the VMI.

Type

`array`

.spec.domain.cpu.features[]

Description

CPUFeature allows specifying a CPU feature.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name of the CPU feature

Property	Type	Description
`policy`	`string`	Policy is the CPU feature attribute which can have the following attributes: force - The virtual CPU will claim the feature is supported regardless of it being supported by host CPU. require - Guest creation will fail unless the feature is supported by the host CPU or the hypervisor is able to emulate it. optional - The feature will be supported by virtual CPU if and only if it is supported by host CPU. disable - The feature will not be supported by virtual CPU. forbid - Guest creation will fail if the feature is supported by host CPU. Defaults to require

.spec.domain.cpu.numa

Description

NUMA allows specifying settings for the guest NUMA topology

Type

`object`

Property	Type	Description
`guestMappingPassthrough`	`object`	GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on the virtual numa nodes never cross boundaries of host numa nodes.

.spec.domain.cpu.numa.guestMappingPassthrough

Description

GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on the virtual numa nodes never cross boundaries of host numa nodes.

Type

`object`

.spec.domain.cpu.realtime

Description

Realtime instructs the virt-launcher to tune the VMI for lower latency, optional for real time workloads

Type

`object`

Property	Type	Description
`mask`	`string`	Mask defines the vcpu mask expression that defines which vcpus are used for realtime. Format matches libvirt's expressions. Example: "0-3,^1","0,2,3","2-3"

.spec.domain.devices

Description

Devices allows adding disks, network interfaces, and others

Type

`object`

Property	Type	Description
`autoattachGraphicsDevice`	`boolean`	Whether to attach the default graphics device or not. VNC will not be available if set to false. Defaults to true.
`autoattachInputDevice`	`boolean`	Whether to attach an Input Device. Defaults to false.
`autoattachMemBalloon`	`boolean`	Whether to attach the Memory balloon device with default period. Period can be adjusted in virt-config. Defaults to true.
`autoattachPodInterface`	`boolean`	Whether to attach a pod network interface. Defaults to true.
`autoattachSerialConsole`	`boolean`	Whether to attach the default virtio-serial console or not. Serial console access will not be available if set to false. Defaults to true.
`autoattachVSOCK`	`boolean`	Whether to attach the VSOCK CID to the VM or not. VSOCK access will be available if set to true. Defaults to false.

Property	Type	Description
`blockMultiQueue`	`boolean`	Whether or not to enable virtio multi-queue for block devices. Defaults to false.
`clientPassthrough`	`object`	To configure and access client devices such as redirecting USB
`disableHotplug`	`boolean`	DisableHotplug disabled the ability to hotplug disks.
`disks`	`array`	Disks describes disks, cdroms and luns which are connected to the vmi.
`downwardMetrics`	`object`	DownwardMetrics creates a virtio serials for exposing the downward metrics to the vmi.
`filesystems`	`array`	Filesystems describes filesystem which is connected to the vmi.
`gpus`	`array`	Whether to attach a GPU device to the vmi.

Property	Type	Description
`hostDevices`	`array`	Whether to attach a host device to the vmi.
`inputs`	`array`	Inputs describe input devices
`interfaces`	`array`	Interfaces describe network interfaces which are added to the vmi.
`logSerialConsole`	`boolean`	Whether to log the auto-attached default serial console or not. Serial console logs will be collect to a file and then streamed from a named 'guest-console-log'. Not relevant if autoattachSerialConsole is disabled. Defaults to cluster wide setting on VirtualMachineOptions.
`networkInterfaceMultiqueue`	`boolean`	If specified, virtual network interfaces configured with a virtio bus will also enable the vhost multiqueue feature for network devices. The number of queues created depends on additional factors of the VirtualMachineInstance, like the number of guest CPUs.
`panicDevices`	`array`	PanicDevices provides additional crash information when a guest crashes.

Property	Type	Description
`rng`	`object`	Whether to have random number generator from host
`sound`	`object`	Whether to emulate a sound device.
`tpm`	`object`	Whether to emulate a TPM device.
`useVirtioTransitional`	`boolean`	Fall back to legacy virtio 0.9 support if virtio bus is selected on devices. This is helpful for old machines like CentOS6 or RHEL6 which do not understand virtio_non_transitional (virtio 1.0).
`video`	`object`	Video describes the video device configuration for the vmi.
`watchdog`	`object`	Watchdog describes a watchdog device which can be added to the vmi.

.spec.domain.devices.clientPassthrough

Description

To configure and access client devices such as redirecting USB

Type

`object`

.spec.domain.devices.disks

Description

Disks describes disks, cdroms and luns which are connected to the vmi.

Type

`array`

.spec.domain.devices.disks[]

Type

`object`

Required

`name`

Property	Type	Description
`blockSize`	`object`	If specified, the virtual disk will be presented with the given block sizes.
`bootOrder`	`integer`	BootOrder is an integer value > 0, used to determine ordering of boot devices. Lower values take precedence. Each disk or interface that has a boot order must have a unique value. Disks without a boot order are not tried if a disk with a boot order exists.
`cache`	`string`	Cache specifies which kvm disk cache mode should be used. Supported values are: none: Guest I/O not cached on the host, but may be kept in a disk cache. writethrough: Guest I/O cached on the host but written through to the

Property	Type	Description
		physical medium. Slowest but with most guarantees. writeback: Guest I/O cached on the host. Defaults to none if the storage supports O_DIRECT, otherwise writethrough.
`cdrom`	`object`	Attach a volume as a cdrom to the vmi.
`changedBlockTracking`	`boolean`	ChangedBlockTracking indicates this disk should have CBT option Defaults to false.
`dedicatedIOThread`	`boolean`	dedicatedIOThread indicates this disk should have an exclusive IO Thread. Enabling this implies useIOThreads = true. Defaults to false.
`disk`	`object`	Attach a volume as a disk to the vmi.
`errorPolicy`	`string`	If specified, it can change the default error policy (stop) for the disk
`io`	`string`	IO specifies which QEMU disk IO mode should be used. Supported values are: native, default, threads.
`lun`	`object`	Attach a volume as a LUN to the vmi.

Property	Type	Description
`name`	`string`	Name is the device name
`serial`	`string`	Serial provides the ability to specify a serial number for the disk device.
`shareable`	`boolean`	If specified the disk is made sharable and multiple write from different VMs are permitted
`tag`	`string`	If specified, disk address and its tag will be provided to the guest via config drive metadata

.spec.domain.devices.disks[].blockSize

Description

If specified, the virtual disk will be presented with the given block sizes.

Type

`object`

Property	Type	Description
`custom`	`object`	CustomBlockSize represents the desired logical and physical block size for a VM disk.
`matchVolume`	`object`	Represents if a feature is enabled or disabled.

.spec.domain.devices.disks[].blockSize.custom

Description

CustomBlockSize represents the desired logical and physical block size for a VM disk.

Type

`object`

Property	Type	Description
`discardGranularity`	`integer`	
`logical`	`integer`	
`physical`	`integer`	

.spec.domain.devices.disks[].blockSize.matchVolume

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.devices.disks[].cdrom

Description

Attach a volume as a cdrom to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to true.
`tray`	`string`	Tray indicates if the tray of the device is open or closed. Allowed values are "open" and "closed". Defaults to closed.

.spec.domain.devices.disks[].disk

Description

Attach a volume as a disk to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi, usb.
`pciAddress`	`string`	If specified, the virtual disk will be placed on the guests pci address with the specified PCI address. For example: 0000:81:01.10

Property	Type	Description
`readonly`	`boolean`	ReadOnly. Defaults to false.

.spec.domain.devices.disks[].lun

Description

Attach a volume as a LUN to the vmi.

Type

`object`

Property	Type	Description
`bus`	`string`	Bus indicates the type of disk device to emulate. supported values: virtio, sata, scsi.
`readonly`	`boolean`	ReadOnly. Defaults to false.
`reservation`	`boolean`	Reservation indicates if the disk needs to support the persistent reservation for the SCSI disk

.spec.domain.devices.downwardMetrics

Description

DownwardMetrics creates a virtio serials for exposing the downward metrics to the vmi.

Type

`object`

.spec.domain.devices.filesystems

Description

Filesystems describes filesystem which is connected to the vmi.

Type

`array`

.spec.domain.devices.filesystems[]

Type

`object`

Required

`name` `virtiofs`

Property	Type	Description
`name`	`string`	Name is the device name
`virtiofs`	`object`	Virtiofs is supported

.spec.domain.devices.filesystems[].virtiofs

Description

Virtiofs is supported

Type

`object`

.spec.domain.devices.gpus

Description

Whether to attach a GPU device to the vmi.

Type

`array`

.spec.domain.devices.gpus[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	Name of the GPU device as exposed by a device plugin
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config

Property	Type	Description
		drive
`virtualGPUOptions`	`object`	

.spec.domain.devices.gpus[].virtualGPUOptions

Type

`object`

Property	Type	Description
`display`	`object`	

.spec.domain.devices.gpus[].virtualGPUOptions.display

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if a display addapter backed by a vGPU should be enabled or disabled on the guest. Defaults to true.
`ramFB`	`object`	Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

.spec.domain.devices.gpus[].virtualGPUOptions.display.ramFB

Description

Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.devices.hostDevices

Description

Whether to attach a host device to the vmi.

Type

`array`

.spec.domain.devices.hostDevices[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated

Property	Type	Description
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive

.spec.domain.devices.inputs

Description

Inputs describe input devices

Type

`array`

.spec.domain.devices.inputs[]

Type

`object`

Required

`name` `type`

Property	Type	Description
`bus`	`string`	Bus indicates the bus of input device to emulate. Supported values: virtio, usb.
`name`	`string`	Name is the device name
`type`	`string`	Type indicated the type of input device. Supported values: tablet.

.spec.domain.devices.interfaces

Description

Interfaces describe network interfaces which are added to the vmi.

Type

`array`

.spec.domain.devices.interfaces[]

Type

`object`

Required

`name`

Property	Type	Description
`acpiIndex`	`integer`	If specified, the ACPI index is used to provide network interface device naming, that is stable across changes in PCI addresses assigned to the device. This value is

Property	Type	Description
		required to be unique across all devices and be between 1 and (16*1024-1).
`binding`	`object`	Binding specifies the binding plugin that will be used to connect the interface to the guest. It provides an alternative to InterfaceBindingMethod. version: 1alphav1
`bootOrder`	`integer`	BootOrder is an integer value > 0, used to determine ordering of boot devices. Lower values take precedence. Each interface or disk that has a boot order must have a unique value. Interfaces without a boot order are not tried.
`bridge`	`object`	InterfaceBridge connects to a given network via a linux bridge.
`dhcpOptions`	`object`	If specified the network interface will pass additional DHCP options to the VMI
`macAddress`	`string`	Interface MAC address. For example: de:ad:00:00:be:af or DE-AD-00-00-BE-AF.
`macvtap`	`object`	DeprecatedMacvtap is an alias to the deprecated Macvtap interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3

Property	Type	Description
`masquerade`	`object`	InterfaceMasquerade connects to a given network using netfilter rules to nat the traffic.
`model`	`string`	Interface model. One of: e1000, e1000e, igb, ne2k_pci, pcnet, rtl8139, virtio. Defaults to virtio.
`name`	`string`	Logical name of the interface as well as a reference to the associated networks. Must match the Name of a Network.
`passt`	`object`	DeprecatedPasst is an alias to the deprecated Passt interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3
`pciAddress`	`string`	If specified, the virtual network interface will be placed on the guests pci address with the specified PCI address. For example: 0000:81:01.10
`ports`	`array`	List of ports to be forwarded to the virtual machine.
`slirp`	`object`	DeprecatedSlirp is an alias to the deprecated Slirp interface Deprecated: Removed in v1.3

Property	Type	Description
`sriov`	`object`	InterfaceSRIOV connects to a given network by passing-through an SR-IOV PCI device via vfio.
`state`	`string`	State represents the requested operational state of the interface. The supported values are: 'absent', expressing a request to remove the interface. 'down', expressing a request to set the link down. 'up', expressing a request to set the link up. Empty value functions as 'up'.
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive

.spec.domain.devices.interfaces[].binding

Description

Binding specifies the binding plugin that will be used to connect the interface to the guest. It provides an alternative to InterfaceBindingMethod. version: 1alphav1

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name references to the binding name as denined in the kubevirt CR. version: 1alphav1

.spec.domain.devices.interfaces[].bridge

Description

InterfaceBridge connects to a given network via a linux bridge.

Type

`object`

.spec.domain.devices.interfaces[].dhcpOptions

Description

If specified the network interface will pass additional DHCP options to the VMI

Type

`object`

Property	Type	Description
`bootFileName`	`string`	If specified will pass option 67 to interface's DHCP server
`ntpServers`	`array`	If specified will pass the configured NTP server to the VM via DHCP option 042.
`privateOptions`	`array`	If specified will pass extra DHCP options for private use, range: 224-254
`tftpServerName`	`string`	If specified will pass option 66 to interface's DHCP server

.spec.domain.devices.interfaces[].dhcpOptions.ntpServers

Description

If specified will pass the configured NTP server to the VM via DHCP option 042.

Type

`array`

.spec.domain.devices.interfaces[].dhcpOptions.ntpServers[]

Type

`string`

.spec.domain.devices.interfaces[].dhcpOptions.privateOptions

Description

If specified will pass extra DHCP options for private use, range: 224-254

Type

`array`

.spec.domain.devices.interfaces[].dhcpOptions.privateOptions[]

Description

DHCPExtraOptions defines Extra DHCP options for a VM.

Type

`object`

Required

`option` `value`

Property	Type	Description
`option`	`integer`	Option is an Integer value from 224-254 Required.
`value`	`string`	Value is a String value for the Option provided Required.

.spec.domain.devices.interfaces[].macvtap

Description

DeprecatedMacvtap is an alias to the deprecated Macvtap interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3

Type

`object`

.spec.domain.devices.interfaces[].masquerade

Description

InterfaceMasquerade connects to a given network using netfilter rules to nat the traffic.

Type

`object`

.spec.domain.devices.interfaces[].passt

Description

DeprecatedPasst is an alias to the deprecated Passt interface, please refer to Kubevirt user guide for alternatives. Deprecated: Removed in v1.3

Type

`object`

.spec.domain.devices.interfaces[].ports

Description

List of ports to be forwarded to the virtual machine.

Type

`array`

.spec.domain.devices.interfaces[].ports[]

Description

Port represents a port to expose from the virtual machine. Default protocol TCP. The port field is mandatory

Type

`object`

Required

`port`

Property	Type	Description
`name`	`string`	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
`port`	`integer`	Number of port to expose for the virtual machine. This must be a valid port number, 0 < x < 65536.
`protocol`	`string`	Protocol for port. Must be UDP or TCP. Defaults to "TCP".

.spec.domain.devices.interfaces[].slirp

Description

DeprecatedSlirp is an alias to the deprecated Slirp interface Deprecated: Removed in v1.3

Type

`object`

.spec.domain.devices.interfaces[].sriov

Description

InterfaceSRIOV connects to a given network by passing-through an SR-IOV PCI device via vfio.

Type

`object`

.spec.domain.devices.panicDevices

Description

PanicDevices provides additional crash information when a guest crashes.

Type

`array`

.spec.domain.devices.panicDevices[]

Type

`object`

Property	Type	Description
`model`	`string`	Model specifies what type of panic device is provided. The panic model used when this attribute is missing depends

Property	Type	Description
		on the hypervisor and guest arch. One of: isa, hyperv, pvpanic.

.spec.domain.devices.rng

Description

Whether to have random number generator from host

Type

`object`

.spec.domain.devices.sound

Description

Whether to emulate a sound device.

Type

`object`

Required

`name`

Property	Type	Description
`model`	`string`	We only support ich9 or ac97. If SoundDevice is not set: No sound card is emulated. If SoundDevice is set but Model is not: ich9
`name`	`string`	User's defined name for this sound device

.spec.domain.devices.tpm

Description

Whether to emulate a TPM device.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled allows a user to explicitly disable the vTPM even when one is enabled by a preference referenced by the VirtualMachine Defaults to True
`persistent`	`boolean`	Persistent indicates the state of the TPM device should be kept accross reboots Defaults to false

.spec.domain.devices.video

Description

Video describes the video device configuration for the vmi.

Type

`object`

Property	Type	Description
`type`	`string`	Type specifies the video device type (e.g., virtio, vga, bochs, ramfb). If not specified, the default is architecture-dependent (VGA for BIOS-based VMs, Bochs for EFI-based VMs on AMD64; virtio for Arm and s390x).

.spec.domain.devices.watchdog

Description

Watchdog describes a watchdog device which can be added to the vmi.

Type

`object`

Required

`name`

Property	Type	Description
`diag288`	`object`	diag288 watchdog device (specific to s390x architecture).
`i6300esb`	`object`	i6300esb watchdog device.
`name`	`string`	Name of the watchdog.

.spec.domain.devices.watchdog.diag288

Description

diag288 watchdog device (specific to s390x architecture).

Type

`object`

Property	Type	Description
`action`	`string`	The action to take. Valid values are poweroff, reset, shutdown. Defaults to reset.

.spec.domain.devices.watchdog.i6300esb

Description

i6300esb watchdog device.

Type

`object`

Property	Type	Description
`action`	`string`	The action to take. Valid values are poweroff, reset, shutdown. Defaults to reset.

.spec.domain.features

Description

Features like acpi, apic, hyperv, smm.

Type

`object`

Property	Type	Description
`acpi`	`object`	ACPI enables/disables ACPI inside the guest. Defaults to enabled.
`apic`	`object`	Defaults to the machine type setting.
`hyperv`	`object`	Defaults to the machine type setting.

Property	Type	Description
`hypervPassthrough`	`object`	This enables all supported hyperv flags automatically. Bear in mind that if this enabled hyperV features cannot be enabled explicitly. In addition, a Virtual Machine using it will be non-migratable.
`kvm`	`object`	Configure how KVM presence is exposed to the guest.
`pvspinlock`	`object`	Notify the guest that the host supports paravirtual spinlocks. For older kernels this feature should be explicitly disabled.
`smm`	`object`	SMM enables/disables System Management Mode. TSEG not yet implemented.

.spec.domain.features.acpi

Description

ACPI enables/disables ACPI inside the guest. Defaults to enabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.apic

Description

Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`endOfInterrupt`	`boolean`	EndOfInterrupt enables the end of interrupt notification in the guest. Defaults to false.

.spec.domain.features.hyperv

Description

Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`evmcs`	`object`	EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.
`frequencies`	`object`	Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.
`ipi`	`object`	IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.
`reenlightenment`	`object`	Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.
`relaxed`	`object`	Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.
`reset`	`object`	Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.
`runtime`	`object`	Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.

Property	Type	Description
`spinlocks`	`object`	Spinlocks allows to configure the spinlock retry attempts.
`synic`	`object`	SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.
`synictimer`	`object`	SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.
`tlbflush`	`object`	TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.
`vapic`	`object`	VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.
`vendorid`	`object`	VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.
`vpindex`	`object`	VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

.spec.domain.features.hyperv.evmcs

Description

EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.frequencies

Description

Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.ipi

Description

IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.reenlightenment

Description

Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.relaxed

Description

Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.reset

Description

Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.runtime

Description

Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.spinlocks

Description

Spinlocks allows to configure the spinlock retry attempts.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`spinlocks`	`integer`	Retries indicates the number of retries. Must be a value greater or equal 4096. Defaults to 4096.

.spec.domain.features.hyperv.synic

Description

SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.synictimer

Description

SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`direct`	`object`	Represents if a feature is enabled or disabled.
`enabled`	`boolean`	

.spec.domain.features.hyperv.synictimer.direct

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.tlbflush

Description

TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.vapic

Description

VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hyperv.vendorid

Description

VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`vendorid`	`string`	VendorID sets the hypervisor vendor id, visible to the vmi. String up to twelve characters.

.spec.domain.features.hyperv.vpindex

Description

VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.hypervPassthrough

Description

This enables all supported hyperv flags automatically. Bear in mind that if this enabled hyperV features cannot be enabled explicitly. In addition, a Virtual Machine using it will be non-migratable.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	

.spec.domain.features.kvm

Description

Configure how KVM presence is exposed to the guest.

Type

`object`

Property	Type	Description
`hidden`	`boolean`	Hide the KVM hypervisor from standard MSR based discovery. Defaults to false

.spec.domain.features.pvspinlock

Description

Notify the guest that the host supports paravirtual spinlocks. For older kernels this feature should be explicitly disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.features.smm

Description

SMM enables/disables System Management Mode. TSEG not yet implemented.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.domain.firmware

Description

Firmware.

Type

`object`

Property	Type	Description
`acpi`	`object`	Information that can be set in the ACPI table
`bootloader`	`object`	Settings to control the bootloader that is used.
`kernelBoot`	`object`	Settings to set the kernel for booting.
`serial`	`string`	The system-serial-number in SMBIOS
`uuid`	`string`	UUID reported by the vmi bios. Defaults to a random generated uid.

.spec.domain.firmware.acpi

Description

Information that can be set in the ACPI table

Type

`object`

Property	Type	Description
`msdmNameRef`	`string`	Similar to SlicNameRef, another ACPI entry that is used in more recent Windows versions. The above points to the spec of MSDM too.

Property	Type	Description
`slicNameRef`	`string`	SlicNameRef should match the volume name of a secret object. The data in the secret should be a binary blob that follows the ACPI SLIC standard, see: https://learn.microsoft.com/en-us/previous-versions/windows/hardware/design/dn653305(v=vs.85) ↗

.spec.domain.firmware.bootloader

Description

Settings to control the bootloader that is used.

Type

`object`

Property	Type	Description
`bios`	`object`	If set (default), BIOS will be used.
`efi`	`object`	If set, EFI will be used instead of BIOS.

.spec.domain.firmware.bootloader.bios

Description

If set (default), BIOS will be used.

Type

`object`

Property	Type	Description
`useSerial`	`boolean`	If set, the BIOS output will be transmitted over serial

.spec.domain.firmware.bootloader.efi

Description

If set, EFI will be used instead of BIOS.

Type

`object`

Property	Type	Description
`persistent`	`boolean`	If set to true, Persistent will persist the EFI NVRAM across reboots. Defaults to false
`secureBoot`	`boolean`	If set, SecureBoot will be enabled and the OVMF roms will be swapped for SecureBoot-enabled ones. Requires SMM to be enabled. Defaults to true

.spec.domain.firmware.kernelBoot

Description

Settings to set the kernel for booting.

Type

`object`

Property	Type	Description
`container`	`object`	Container defines the container that containes kernel artifacts
`kernelArgs`	`string`	Arguments to be passed to the kernel at boot time

.spec.domain.firmware.kernelBoot.container

Description

Container defines the container that containes kernel artifacts

Type

`object`

Required

`image`

Property	Type	Description
`image`	`string`	Image that contains initrd / kernel files.
`imagePullPolicy`	`string`	Image pull policy. One of Always, Never, IfNotPresent. Defa to Always if :latest tag is specified, or IfNotPresent otherwis Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#upd images ↗

Property	Type	Description
`imagePullSecret`	`string`	ImagePullSecret is the name of the Docker registry secret required to pull the image. The secret must already exist.
`initrdPath`	`string`	the fully-qualified path to the ramdisk image in the host OS
`kernelPath`	`string`	The fully-qualified path to the kernel image in the host OS

.spec.domain.ioThreads

Description

IOThreads specifies the IOThreads options.

Type

`object`

Property	Type	Description
`supplementalPoolThreadCount`	`integer`	SupplementalPoolThreadCount specifies how many iothreads are allocated for the supplementalPool policy.

.spec.domain.launchSecurity

Description

Launch Security setting of the vmi.

Type

`object`

Property	Type	Description
`sev`	`object`	AMD Secure Encrypted Virtualization (SEV).
`snp`	`object`	AMD SEV-SNP flags defined by the SEV-SNP specifications.
`tdx`	`object`	Intel Trust Domain Extensions (TDX).

.spec.domain.launchSecurity.sev

Description

AMD Secure Encrypted Virtualization (SEV).

Type

`object`

Property	Type	Description
`attestation`	`object`	If specified, run the attestation process for a vmi.
`dhCert`	`string`	Base64 encoded guest owner's Diffie-Hellman key.

Property	Type	Description
`policy`	`object`	Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.
`session`	`string`	Base64 encoded session blob.

.spec.domain.launchSecurity.sev.attestation

Description

If specified, run the attestation process for a vmi.

Type

`object`

.spec.domain.launchSecurity.sev.policy

Description

Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.

Type

`object`

Property	Type	Description
`encryptedState`	`boolean`	SEV-ES is required. Defaults to false.

.spec.domain.launchSecurity.snp

Description

AMD SEV-SNP flags defined by the SEV-SNP specifications.

Type

`object`

.spec.domain.launchSecurity.tdx

Description

Intel Trust Domain Extensions (TDX).

Type

`object`

.spec.domain.machine

Description

Machine type.

Type

`object`

Property	Type	Description
`type`	`string`	QEMU machine type is the actual chipset of the VirtualMachineInstance.

.spec.domain.memory

Description

Memory allow specifying the VMI memory features.

Type

`object`

Property	Type	Description
`guest`	`integer|string`	Guest allows to specifying the amount of memory which is visible inside the Guest OS. The Guest must lie between Requests and Limits from the resources section. Defaults to the requested memory in the resources section if not specified.
`hugepages`	`object`	Hugepages allow to use hugepages for the VirtualMachineInstance instead of regular memory.
`maxGuest`	`integer|string`	MaxGuest allows to specify the maximum amount of memory which is visible inside the Guest OS. The delta between MaxGuest and Guest is the amount of memory that can be hot(un)plugged.

.spec.domain.memory.hugepages

Description

Hugepages allow to use hugepages for the VirtualMachineInstance instead of regular memory.

Type

`object`

Property	Type	Description
`pageSize`	`string`	PageSize specifies the hugepage size, for x86_64 architecture valid values are 1Gi and 2Mi.

.spec.domain.resources

Description

Resources describes the Compute Resources required by this vmi.

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. Valid resource keys are "memory" and "cpu".
`overcommitGuestOverhead`	`boolean`	Don't ask the scheduler to take the guest-management overhead into account. Instead put the overhead only into the container's memory limit. This can lead to crashes if all memory is in use on a node. Defaults to false.
`requests`	`object`	Requests is a description of the initial vmi resources. Valid resource keys are "memory" and "cpu".

.spec.domain.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. Valid resource keys are "memory" and "cpu".

Type

`object`

.spec.domain.resources.requests

Description

Requests is a description of the initial vmi resources. Valid resource keys are "memory" and "cpu".

Type

`object`

.spec.livenessProbe

Description

Periodic probe of VirtualMachineInstance liveness. VirtualmachineInstances will be stopped if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Type

`object`

Property	Type	Description
`exec`	`object`	One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Property	Type	Description
`failureThreshold`	`integer`	Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3 Minimum value is 1.
`guestAgentPing`	`object`	GuestAgentPing contacts the qemu-guest-agent for availability checks.
`httpGet`	`object`	HTTPGet specifies the http request to perform.
`initialDelaySeconds`	`integer`	Number of seconds after the VirtualMachineInstance h started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗
`periodSeconds`	`integer`	How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.
`successThreshold`	`integer`	Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1 Must be 1 for liveness. Minimum value is 1.
`tcpSocket`	`object`	TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Property	Type	Description
`timeoutSeconds`	`integer`	Number of seconds after which the probe times out. Fo exec probes the timeout fails the probe but does not terminate the command running on the guest. This means a blocking command can result in an increasing load on the guest. A small buffer will be added to the resulting workload exec probe to compensate for delay caused by the qemu guest exec mechanism. Defaults 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/po lifecycle#container-probes ↗

.spec.livenessProbe.exec

Description

One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Type

`object`

Property	Type	Description
`command`	`array`	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

.spec.livenessProbe.exec.command

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

.spec.livenessProbe.exec.command[]

Type

`string`

.spec.livenessProbe.guestAgentPing

Description

GuestAgentPing contacts the qemu-guest-agent for availability checks.

Type

`object`

.spec.livenessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
`httpHeaders`	`array`	Custom headers to set in the request. HTTP allows repeated headers.
`path`	`string`	Path to access on the HTTP server.
`port`	`integer|string`	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
`scheme`	`string`	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.livenessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

.spec.livenessProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
`name`	`string`	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
`value`	`string`	The header field value

.spec.livenessProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Optional: Host name to connect to, defaults to the pod IP.
`port`	`integer|string`	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must

Property	Type	Description
		be an IANA_SVC_NAME.

.spec.networks

Description

List of networks that can be attached to a vm's virtual interface.

Type

`array`

.spec.networks[]

Description

Network represents a network type and a resource that should be connected to the vm.

Type

`object`

Required

`name`

Property	Type	Description
`multus`	`object`	Represents the multus cni network.
`name`	`string`	Network name. Must be a DNS_LABEL and unique within the vm. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗

Property	Type	Description
`pod`	`object`	Represents the stock pod network interface.

.spec.networks[].multus

Description

Represents the multus cni network.

Type

`object`

Required

`networkName`

Property	Type	Description
`default`	`boolean`	Select the default network and add it to the multus-cni.io/default-network annotation.
`networkName`	`string`	References to a NetworkAttachmentDefinition CRD object. Format: , /. If namespace is not specified, VMI namespace is assumed.

.spec.networks[].pod

Description

Represents the stock pod network interface.

Type

`object`

Property	Type	Description
`vmIPv6NetworkCIDR`	`string`	IPv6 CIDR for the vm network. Defaults to fd10:0:2::/120 if not specified.
`vmNetworkCIDR`	`string`	CIDR for vm network. Default 10.0.2.0/24 if not specified.

.spec.nodeSelector

Description

NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/

Type

`object`

.spec.readinessProbe

Description

Periodic probe of VirtualMachineInstance service readiness. VirtualmachineInstances will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Type

`object`

Property	Type	Description
`exec`	`object`	One and only one of the following should be specified. Exec specifies the action to take, it will be executed on

Property	Type	Description
		the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.
`failureThreshold`	`integer`	Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3 Minimum value is 1.
`guestAgentPing`	`object`	GuestAgentPing contacts the qemu-guest-agent for availability checks.
`httpGet`	`object`	HTTPGet specifies the http request to perform.
`initialDelaySeconds`	`integer`	Number of seconds after the VirtualMachineInstance h started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗
`periodSeconds`	`integer`	How often (in seconds) to perform the probe. Default t 10 seconds. Minimum value is 1.
`successThreshold`	`integer`	Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1 Must be 1 for liveness. Minimum value is 1.

Property	Type	Description
`tcpSocket`	`object`	TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported
`timeoutSeconds`	`integer`	Number of seconds after which the probe times out. Fo exec probes the timeout fails the probe but does not terminate the command running on the guest. This means a blocking command can result in an increasing load on the guest. A small buffer will be added to the resulting workload exec probe to compensate for delay caused by the qemu guest exec mechanism. Defaults 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pc lifecycle#container-probes ↗

.spec.readinessProbe.exec

Description

One and only one of the following should be specified. Exec specifies the action to take, it will be executed on the guest through the qemu-guest-agent. If the guest agent is not available, this probe will fail.

Type

`object`

Property	Type	Description
`command`	`array`	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to

Property	Type	Description
		explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

.spec.readinessProbe.exec.command

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

.spec.readinessProbe.exec.command[]

Type

`string`

.spec.readinessProbe.guestAgentPing

Description

GuestAgentPing contacts the qemu-guest-agent for availability checks.

Type

`object`

.spec.readinessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
`httpHeaders`	`array`	Custom headers to set in the request. HTTP allows repeated headers.
`path`	`string`	Path to access on the HTTP server.
`port`	`integer|string`	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
`scheme`	`string`	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.readinessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

.spec.readinessProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
`name`	`string`	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
`value`	`string`	The header field value

.spec.readinessProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port. TCP hooks not yet supported

Type

`object`

Required

`port`

Property	Type	Description
`host`	`string`	Optional: Host name to connect to, defaults to the pod IP.
`port`	`integer|string`	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.resourceClaims

Description

ResourceClaims define which ResourceClaims must be allocated and reserved before the VMI, hence virt-launcher pod is allowed to start. The resources will be made available to the domain which consumes them by name. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate in kubernetes https://kubernetes.io/docs/concepts/scheduling-eviction/dynamic-resource-allocation/ This field should only be configured if one of the feature-gates GPUsWithDRA or HostDevicesWithDRA is enabled. This feature is in alpha.

Type

`array`

.spec.resourceClaims[]

Description

PodResourceClaim references exactly one ResourceClaim, either directly or by naming a ResourceClaimTemplate which is then turned into a ResourceClaim for the pod. It adds a name to it that uniquely identifies the ResourceClaim inside the Pod. Containers that need access to the ResourceClaim reference it with this name.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name uniquely identifies this resource claim inside the pod. This must be a DNS_LABEL.
`resourceClaimName`	`string`	ResourceClaimName is the name of a ResourceClaim object in the same namespace as this pod. Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.
`resourceClaimTemplateName`	`string`	ResourceClaimTemplateName is the name of a ResourceClaimTemplate object in the same namespace as this pod. The template will be used to create a new ResourceClaim, which will be bound to this pod. When this pod is deleted, the ResourceClaim will also be deleted. The pod name and resource name, along with a generated component, will be used to form a unique name for the ResourceClaim, which will be recorded in pod.status.resourceClaimStatuses. This field is immutable and no changes will be made to the corresponding

Property	Type	Description
		ResourceClaim by the control plane after creating the ResourceClaim. Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

.spec.tolerations

Description

If toleration is specified, obey all the toleration rules.

Type

`array`

.spec.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.

Property	Type	Description
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
`operator`	`string`	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
`tolerationSeconds`	`integer`	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.topologySpreadConstraints

Description

TopologySpreadConstraints describes how a group of VMIs will be spread across a given topology domains. K8s scheduler will schedule VMI pods in a way which abides by the constraints.

Type

`array`

.spec.topologySpreadConstraints[]

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

`object`

Required

`maxSkew` `topologyKey` `whenUnsatisfiable`

Property	Type	Description
`labelSelector`	`object`	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector.

Property	Type	Description
		This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
`maxSkew`	`integer`	MaxSkew describes the degree to which pods may be unevenly distributed. When 'whenUnsatisfiable=DoNotSchedule', it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. | zone1 | zone2 | zone3 | | P P | P P | P | • if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). • if MaxSkew is 2, incoming pod can be scheduled onto any zone. When 'whenUnsatisfiable=ScheduleAnyway', it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
`minDomains`	`integer`	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than

Property	Type	Description
		minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule. For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: | zone1 | zone2 | zone3 | | P P | P P | P P | The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
`nodeAffinityPolicy`	`string`	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: • Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. • Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations.

Property	Type	Description
		If this value is nil, the behavior is equivalent to the Honor policy.
`nodeTaintsPolicy`	`string`	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: • Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included. • Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy.
`topologyKey`	`string`	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.

Property	Type	Description
`whenUnsatisfiable`	`string`	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. • DoNotSchedule (default) tells the scheduler not to schedule it. • ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: | zone1 | zone2 | zone3 | | P P P | P | P | If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it *more* imbalanced. It's a required field.

.spec.topologySpreadConstraints[].labelSelector

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.topologySpreadConstraints[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.topologySpreadConstraints[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.topologySpreadConstraints[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.topologySpreadConstraints[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.topologySpreadConstraints[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Type

`array`

.spec.topologySpreadConstraints[].matchLabelKeys[]

Type

`string`

.spec.volumes

Description

List of volumes that can be mounted by disks belonging to the vmi.

Type

`array`

.spec.volumes[]

Description

Volume represents a named volume in a vmi.

Type

`object`

Required

`name`

Property	Type	Description
`cloudInitConfigDrive`	`object`	CloudInitConfigDrive represents a cloud-init Config D The Config Drive data will be added as a disk to the installation is required inside the guest. More info: https://cloudinit.readthedocs.io/en/latest/topics/datas ↗
`cloudInitNoCloud`	`object`	CloudInitNoCloud represents a cloud-init NoCloud u NoCloud data will be added as a disk to the vmi. A p installation is required inside the guest. More info: http://cloudinit.readthedocs.io/en/latest/topics/datasc
`configMap`	`object`	ConfigMapSource represents a reference to a Confi namespace. More info: https://kubernetes.io/docs/ta container/configure-pod-configmap/ ↗

Property	Type	Description
`containerDisk`	`object`	ContainerDisk references a docker image, embeddi More info: https://kubevirt.gitbooks.io/user-guide/reg
`dataVolume`	`object`	DataVolume represents the dynamic creation a PVC as the process of populating that PVC with a disk im
`downwardAPI`	`object`	DownwardAPI represents downward API about the p populate this volume
`downwardMetrics`	`object`	DownwardMetrics adds a very small disk to VMIs wh view of host and guest metrics. The disk content is c vhostmd (https://github.com/vhostmd/vhostmd ↗) an
`emptyDisk`	`object`	EmptyDisk represents a temporary disk which share More info: https://kubevirt.gitbooks.io/user-guide/disl
`ephemeral`	`object`	Ephemeral is a special volume source that "wraps" s provides copy-on-write image on top of it.
`hostDisk`	`object`	HostDisk represents a disk created on the cluster le
`memoryDump`	`object`	MemoryDump is attached to the virt launcher and is memory dump of the vmi

Property	Type	Description
`name`	`string`	Volume's name. Must be a DNS_LABEL and unique info: https://kubernetes.io/docs/concepts/overview/w objects/names/#names ↗
`persistentVolumeClaim`	`object`	PersistentVolumeClaimVolumeSource represents a PersistentVolumeClaim in the same namespace. Dir vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persiste volumes#persistentvolumeclaims ↗
`secret`	`object`	SecretVolumeSource represents a reference to a se namespace. More info: https://kubernetes.io/docs/concepts/configuration/se
`serviceAccount`	`object`	ServiceAccountVolumeSource represents a referenc There can only be one volume of this type! More infc https://kubernetes.io/docs/tasks/configure-pod-conta account/ ↗
`sysprep`	`object`	Represents a Sysprep volume source.

.spec.volumes[].cloudInitConfigDrive

Description

CloudInitConfigDrive represents a cloud-init Config Drive user-data source. The Config Drive data will be added as a disk to the vmi. A proper cloud-init installation is required

inside the guest. More info: https://cloudinit.readthedocs.io/en/latest/topics/datasources/configdrive.html

Type

`object`

Property	Type	Description
`networkData`	`string`	NetworkData contains config drive inline cloud-init networkdata.
`networkDataBase64`	`string`	NetworkDataBase64 contains config drive cloud-init networkdata as a base64 encoded string.
`networkDataSecretRef`	`object`	NetworkDataSecretRef references a k8s secret that contains config drive networkdata.
`secretRef`	`object`	UserDataSecretRef references a k8s secret that contains config drive userdata.
`userData`	`string`	UserData contains config drive inline cloud-init userdata.
`userDataBase64`	`string`	UserDataBase64 contains config drive cloud-init userdata as a base64 encoded string.

.spec.volumes[].cloudInitConfigDrive.networkDataSecretRef

Description

NetworkDataSecretRef references a k8s secret that contains config drive networkdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].cloudInitConfigDrive.secretRef

Description

UserDataSecretRef references a k8s secret that contains config drive userdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].cloudInitNoCloud

Description

CloudInitNoCloud represents a cloud-init NoCloud user-data source. The NoCloud data will be added as a disk to the vmi. A proper cloud-init installation is required inside the guest. More info: http://cloudinit.readthedocs.io/en/latest/topics/datasources/nocloud.html

Type

`object`

Property	Type	Description
`networkData`	`string`	NetworkData contains NoCloud inline cloud-init networkdata.
`networkDataBase64`	`string`	NetworkDataBase64 contains NoCloud cloud-init networkdata as a base64 encoded string.
`networkDataSecretRef`	`object`	NetworkDataSecretRef references a k8s secret that contains NoCloud networkdata.
`secretRef`	`object`	UserDataSecretRef references a k8s secret that contains NoCloud userdata.
`userData`	`string`	UserData contains NoCloud inline cloud-init userdata.
`userDataBase64`	`string`	UserDataBase64 contains NoCloud cloud-init userdata as a base64 encoded string.

.spec.volumes[].cloudInitNoCloud.networkDataSecretRef

Description

NetworkDataSecretRef references a k8s secret that contains NoCloud networkdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].cloudInitNoCloud.secretRef

Description

UserDataSecretRef references a k8s secret that contains NoCloud userdata.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].configMap

Description

ConfigMapSource represents a reference to a ConfigMap in the same namespace. More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`optional`	`boolean`	Specify whether the ConfigMap or it's keys must be defined
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.volumes[].containerDisk

Description

ContainerDisk references a docker image, embedding a qcow or raw disk. More info: https://kubevirt.gitbooks.io/user-guide/registry-disk.html

Type

`object`

Required

`image`

Property	Type	Description
`image`	`string`	Image is the name of the image with the embedded disk.
`imagePullPolicy`	`string`	Image pull policy. One of Always, Never, IfNotPresent. Defa to Always if :latest tag is specified, or IfNotPresent otherwis Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#upd images ↗
`imagePullSecret`	`string`	ImagePullSecret is the name of the Docker registry secret required to pull the image. The secret must already exist.
`path`	`string`	Path defines the path to disk file in the container

.spec.volumes[].dataVolume

Description

DataVolume represents the dynamic creation a PVC for this volume as well as the process of populating that PVC with a disk image.

Type

`object`

Required

`name`

Property	Type	Description
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`name`	`string`	Name of both the DataVolume and the PVC in the same namespace.

.spec.volumes[].downwardAPI

Description

DownwardAPI represents downward API about the pod that should populate this volume

Type

`object`

Property	Type	Description
`fields`	`array`	Fields is a list of downward API volume file
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.volumes[].downwardAPI.fields

Description

Fields is a list of downward API volume file

Type

`array`

.spec.volumes[].downwardAPI.fields[]

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

`object`

Required

`path`

Property	Type	Description
`fieldRef`	`object`	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.
`mode`	`integer`	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
`path`	`string`	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..'

Property	Type	Description
		path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
`resourceFieldRef`	`object`	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

.spec.volumes[].downwardAPI.fields[].fieldRef

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
`apiVersion`	`string`	Version of the schema the FieldPath is written in terms of, defaults to "v1".
`fieldPath`	`string`	Path of the field to select in the specified API version.

.spec.volumes[].downwardAPI.fields[].resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
`containerName`	`string`	Container name: required for volumes, optional for env vars
`divisor`	`integer|string`	Specifies the output format of the exposed resources, defaults to "1"
`resource`	`string`	Required: resource to select

.spec.volumes[].downwardMetrics

Description

DownwardMetrics adds a very small disk to VMIs which contains a limited view of host and guest metrics. The disk content is compatible with vhostmd (https://github.com/vhostmd/vhostmd) and vm-dump-metrics.

Type

`object`

.spec.volumes[].emptyDisk

Description

EmptyDisk represents a temporary disk which shares the vmis lifecycle. More info: https://kubevirt.gitbooks.io/user-guide/disks-and-volumes.html

Type

`object`

Required

`capacity`

Property	Type	Description
`capacity`	`integer|string`	Capacity of the sparse disk.

.spec.volumes[].ephemeral

Description

Ephemeral is a special volume source that "wraps" specified source and provides copy-on-write image on top of it.

Type

`object`

Property	Type	Description
`persistentVolumeClaim`	`object`	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistevolumes#persistentvolumeclaims ↗

.spec.volumes[].ephemeral.persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.volumes[].hostDisk

Description

HostDisk represents a disk created on the cluster level

Type

`object`

Required

`path` `type`

Property	Type	Description
`capacity`	`integer|string`	Capacity of the sparse disk
`path`	`string`	The path to HostDisk image located on the cluster
`shared`	`boolean`	Shared indicate whether the path is shared between nodes
`type`	`string`	Contains information if disk.img exists or should be created allowed options are 'Disk' and 'DiskOrCreate'

.spec.volumes[].memoryDump

Description

MemoryDump is attached to the virt launcher and is populated with a memory dump of the vmi

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.volumes[].persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Directly attached to the vmi via qemu. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Required

`claimName`

Property	Type	Description
`claimName`	`string`	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

Property	Type	Description
`hotpluggable`	`boolean`	Hotpluggable indicates whether the volume can be hotplugged and hotunplugged.
`readOnly`	`boolean`	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

.spec.volumes[].secret

Description

SecretVolumeSource represents a reference to a secret data in the same namespace. More info: https://kubernetes.io/docs/concepts/configuration/secret/

Type

`object`

Property	Type	Description
`optional`	`boolean`	Specify whether the Secret or it's keys must be defined
`secretName`	`string`	Name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret ↗

Property	Type	Description
`volumeLabel`	`string`	The volume label of the resulting disk inside the VMI. Different bootstrapping mechanisms require different values. Typical values are "cidata" (cloud-init), "config-2" (cloud-init) or "OEMDRV" (kickstart).

.spec.volumes[].serviceAccount

Description

ServiceAccountVolumeSource represents a reference to a service account. There can only be one volume of this type! More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/

Type

`object`

Property	Type	Description
`serviceAccountName`	`string`	Name of the service account in the pod's namespace to use. More info: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/ ↗

.spec.volumes[].sysprep

Description

Represents a Sysprep volume source.

Type

`object`

Property	Type	Description
`configMap`	`object`	ConfigMap references a ConfigMap that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.
`secret`	`object`	Secret references a k8s Secret that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

.spec.volumes[].sysprep.configMap

Description

ConfigMap references a ConfigMap that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].sysprep.secret

Description

Secret references a k8s Secret that contains Sysprep answer file named autounattend.xml that should be attached as disk of CDROM type.

Type

`object`

Property	Type	Description
`name`	`string`	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.status

Description

Status is the high level overview of how the VirtualMachineInstance is doing. It contains information available to controllers and users.

Type

`object`

Property	Type	Description
`VSOCKCID`	`integer`	VSOCKCID is used to track the allocated VSOCK CID in the VM.
`activePods`	`object`	ActivePods is a mapping of pod UID to noc name. It is possible for multiple pods to be running for a single VMI during migration.

Property	Type	Description
`changedBlockTracking`	`object`	ChangedBlockTracking represents the stat the changedBlockTracking
`conditions`	`array`	Conditions are specific points in VirtualMachineInstance's pod runtime.
`currentCPUTopology`	`object`	CurrentCPUTopology specifies the current topology used by the VM workload. Curren topology may differ from the desired topolc the spec while CPU hotplug takes place.
`deviceStatus`	`object`	DeviceStatus reflects the state of devices requested in spec.domain.devices. This is optional field available only when DRA feat gate is enabled This field will only be popu if one of the feature-gates GPUsWithDRA HostDevicesWithDRA is enabled. This feat is in alpha.
`evacuationNodeName`	`string`	EvacuationNodeName is used to track the eviction process of a VMI. It stores the nan the node that we want to evacuate. It is me to be used by KubeVirt core components o and can't be set or modified by users.
`fsFreezeStatus`	`string`	FSFreezeStatus indicates whether a freeze operation was requested for the guest

Property	Type	Description
		filesystem. It will be set to "frozen" if the request was made, or unset otherwise. Thi does not reflect the actual state of the gues filesystem.
`guestOSInfo`	`object`	Guest OS Information
`interfaces`	`array`	Interfaces represent the details of available network interfaces.
`kernelBootStatus`	`object`	KernelBootStatus contains info about the kernelBootContainer
`launcherContainerImageVersion`	`string`	LauncherContainerImageVersion indicates what container image is currently active for vmi.
`machine`	`object`	Machine shows the final resulting qemu machine type. This can be different than th machine type selected in the spec, due to qemus machine type alias mechanism.
`memory`	`object`	Memory shows various informations about VirtualMachine memory.

Property	Type	Description
`migratedVolumes`	`array`	MigratedVolumes lists the source and destination volumes during the volume migration
`migrationMethod`	`string`	Represents the method using which the vn can be migrated: live migration or block migration
`migrationState`	`object`	Represents the status of a live migration
`migrationTransport`	`string`	This represents the migration transport
`nodeName`	`string`	NodeName is the name where the VirtualMachineInstance is currently running
`phase`	`string`	Phase is the status of the VirtualMachineInstance in kubernetes worl is not the VirtualMachineInstance status, b partially correlates to it.
`phaseTransitionTimestamps`	`array`	PhaseTransitionTimestamp is the timestam when the last phase change occurred
`qosClass`	`string`	The Quality of Service (QOS) classificatior assigned to the virtual machine instance ba

Property	Type	Description
		on resource requirements See PodQOSCl type for available QOS classes More info: https://git.k8s.io/community/contributors/de proposals/node/resource-qos.md ↗
`reason`	`string`	A brief CamelCase message indicating det about why the VMI is in this state. e.g. 'NodeUnresponsive'
`runtimeUser`	`integer`	RuntimeUser is used to determine what us will be used in launcher
`selinuxContext`	`string`	SELinuxContext is the actual SELinux cont of the virt-launcher pod
`topologyHints`	`object`	
`virtualMachineRevisionName`	`string`	VirtualMachineRevisionName is used to ge vm revision of the vmi when doing an onlin snapshot
`volumeStatus`	`array`	VolumeStatus contains the statuses of all t volumes

.status.activePods

Description

ActivePods is a mapping of pod UID to node name. It is possible for multiple pods to be running for a single VMI during migration.

Type

`object`

.status.changedBlockTracking

Description

ChangedBlockTracking represents the status of the changedBlockTracking

Type

`object`

Required

`state`

Property	Type	Description
`state`	`string`	State represents the current CBT state

.status.conditions

Description

Conditions are specific points in VirtualMachineInstance's pod runtime.

Type

`array`

.status.conditions[]

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	

.status.currentCPUTopology

Description

CurrentCPUTopology specifies the current CPU topology used by the VM workload. Current topology may differ from the desired topology in the spec while CPU hotplug takes place.

Type

`object`

Property	Type	Description
`cores`	`integer`	Cores specifies the number of cores inside the vmi. Must be a value greater or equal 1.
`sockets`	`integer`	Sockets specifies the number of sockets inside the vmi. Must be a value greater or equal 1.

Property	Type	Description
`threads`	`integer`	Threads specifies the number of threads inside the vmi. Must be a value greater or equal 1.

.status.deviceStatus

Description

DeviceStatus reflects the state of devices requested in spec.domain.devices. This is an optional field available only when DRA feature gate is enabled This field will only be populated if one of the feature-gates GPUsWithDRA or HostDevicesWithDRA is enabled. This feature is in alpha.

Type

`object`

Property	Type	Description
`gpuStatuses`	`array`	GPUStatuses reflects the state of GPUs requested in spec.domain.devices.gpus
`hostDeviceStatuses`	`array`	HostDeviceStatuses reflects the state of GPUs requested in spec.domain.devices.hostDevices DRA

.status.deviceStatus.gpuStatuses

Description

GPUStatuses reflects the state of GPUs requested in spec.domain.devices.gpus

Type

`array`

.status.deviceStatus.gpuStatuses[]

Type

`object`

Required

`name`

Property	Type	Description
`deviceResourceClaimStatus`	`object`	DeviceResourceClaimStatus reflects the DRA related information for the device
`name`	`string`	Name of the device as specified in spec.domain.devices.gpus.name or spec.domain.devices.hostDevices.name

.status.deviceStatus.gpuStatuses[].deviceResourceClaimStatus

Description

DeviceResourceClaimStatus reflects the DRA related information for the device

Type

`object`

Property	Type	Description
`attributes`	`object`	Attributes are properties of the device that could be used by kubevirt and other copmonents to learn

Property	Type	Description
		more about the device, like pciAddress or mdevUUID
`name`	`string`	Name is the name of actual device on the host provisioned by the driver as reflected in resourceclaim.status
`resourceClaimName`	`string`	ResourceClaimName is the name of the resource claims object used to provision this resource

.status.deviceStatus.gpuStatuses[].deviceResourceClaimStatus.attributes

Description

Attributes are properties of the device that could be used by kubevirt and other copmonents to learn more about the device, like pciAddress or mdevUUID

Type

`object`

Property	Type	Description
`mDevUUID`	`string`	MDevUUID is the mediated device uuid of the allocated device
`pciAddress`	`string`	PCIAddress is the PCIe bus address of the allocated device

.status.deviceStatus.hostDeviceStatuses

Description

HostDeviceStatuses reflects the state of GPUs requested in spec.domain.devices.hostDevices DRA

Type

`array`

.status.deviceStatus.hostDeviceStatuses[]

Type

`object`

Required

`name`

Property	Type	Description
`deviceResourceClaimStatus`	`object`	DeviceResourceClaimStatus reflects the DRA related information for the device
`name`	`string`	Name of the device as specified in spec.domain.devices.gpus.name or spec.domain.devices.hostDevices.name

.status.deviceStatus.hostDeviceStatuses[].deviceResourceClaimStatus

Description

DeviceResourceClaimStatus reflects the DRA related information for the device

Type

`object`

Property	Type	Description
`attributes`	`object`	Attributes are properties of the device that could be used by kubevirt and other copmonents to learn more about the device, like pciAddress or mdevUUID
`name`	`string`	Name is the name of actual device on the host provisioned by the driver as reflected in resourceclaim.status
`resourceClaimName`	`string`	ResourceClaimName is the name of the resource claims object used to provision this resource

.status.deviceStatus.hostDeviceStatuses[].deviceResourceClaimStatus.attributes

Description

Attributes are properties of the device that could be used by kubevirt and other copmonents to learn more about the device, like pciAddress or mdevUUID

Type

`object`

Property	Type	Description
`mDevUUID`	`string`	MDevUUID is the mediated device uuid of the allocated device

Property	Type	Description
`pciAddress`	`string`	PCIAddress is the PCIe bus address of the allocated device

.status.guestOSInfo

Description

Guest OS Information

Type

`object`

Property	Type	Description
`id`	`string`	Guest OS Id
`kernelRelease`	`string`	Guest OS Kernel Release
`kernelVersion`	`string`	Kernel version of the Guest OS
`machine`	`string`	Machine type of the Guest OS
`name`	`string`	Name of the Guest OS
`prettyName`	`string`	Guest OS Pretty Name

Property	Type	Description
`version`	`string`	Guest OS Version
`versionId`	`string`	Version ID of the Guest OS

.status.interfaces

Description

Interfaces represent the details of available network interfaces.

Type

`array`

.status.interfaces[]

Type

`object`

Property	Type	Description
`infoSource`	`string`	Specifies the origin of the interface data collected. values: domain, guest-agent, multus-status.
`interfaceName`	`string`	The interface name inside the Virtual Machine
`ipAddress`	`string`	IP address of a Virtual Machine interface. It is always the first item of IPs

Property	Type	Description
`ipAddresses`	`array`	List of all IP addresses of a Virtual Machine interface
`linkState`	`string`	LinkState Reports the current operational link state'. values: up, down.
`mac`	`string`	Hardware address of a Virtual Machine interface
`name`	`string`	Name of the interface, corresponds to name of the network assigned to the interface
`podInterfaceName`	`string`	PodInterfaceName represents the name of the pod network interface
`queueCount`	`integer`	Specifies how many queues are allocated by MultiQueue

.status.interfaces[].ipAddresses

Description

List of all IP addresses of a Virtual Machine interface

Type

`array`

.status.interfaces[].ipAddresses[]

Type

`string`

.status.kernelBootStatus

Description

KernelBootStatus contains info about the kernelBootContainer

Type

`object`

Property	Type	Description
`initrdInfo`	`object`	InitrdInfo show info about the initrd file
`kernelInfo`	`object`	KernelInfo show info about the kernel image

.status.kernelBootStatus.initrdInfo

Description

InitrdInfo show info about the initrd file

Type

`object`

Property	Type	Description
`checksum`	`integer`	Checksum is the checksum of the initrd file

.status.kernelBootStatus.kernelInfo

Description

KernelInfo show info about the kernel image

Type

`object`

Property	Type	Description
`checksum`	`integer`	Checksum is the checksum of the kernel image

.status.machine

Description

Machine shows the final resulting qemu machine type. This can be different than the machine type selected in the spec, due to qemus machine type alias mechanism.

Type

`object`

Property	Type	Description
`type`	`string`	QEMU machine type is the actual chipset of the VirtualMachineInstance.

.status.memory

Description

Memory shows various informations about the VirtualMachine memory.

Type

`object`

Property	Type	Description
`guestAtBoot`	`integer|string`	GuestAtBoot specifies with how much memory the VirtualMachine intiallly booted with.
`guestCurrent`	`integer|string`	GuestCurrent specifies how much memory is currently available for the VirtualMachine.
`guestRequested`	`integer|string`	GuestRequested specifies how much memory was requested (hotplug) for the VirtualMachine.

.status.migratedVolumes

Description

MigratedVolumes lists the source and destination volumes during the volume migration

Type

`array`

.status.migratedVolumes[]

Description

StorageMigratedVolumeInfo tracks the information about the source and destination volumes during the volume migration

Type

`object`

Required

`volumeName`

Property	Type	Description
`destinationPVCInfo`	`object`	DestinationPVCInfo contains the information about the destination PVC
`sourcePVCInfo`	`object`	SourcePVCInfo contains the information about the source PVC
`volumeName`	`string`	VolumeName is the name of the volume that is being migrated

.status.migratedVolumes[].destinationPVCInfo

Description

DestinationPVCInfo contains the information about the destination PVC

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`capacity`	`object`	Capacity represents the capacity set on the corresponding PVC status

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the PVC
`filesystemOverhead`	`string`	Percentage of filesystem's size to be reserved when resizing the PVC
`preallocated`	`boolean`	Preallocated indicates if the PVC's storage is preallocated or not
`requests`	`object`	Requests represents the resources requested by the corresponding PVC spec
`volumeMode`	`string`	VolumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

.status.migratedVolumes[].destinationPVCInfo.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.migratedVolumes[].destinationPVCInfo.accessModes[]

Type

`string`

.status.migratedVolumes[].destinationPVCInfo.capacity

Description

Capacity represents the capacity set on the corresponding PVC status

Type

`object`

.status.migratedVolumes[].destinationPVCInfo.requests

Description

Requests represents the resources requested by the corresponding PVC spec

Type

`object`

.status.migratedVolumes[].sourcePVCInfo

Description

SourcePVCInfo contains the information about the source PVC

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`capacity`	`object`	Capacity represents the capacity set on the corresponding PVC status
`claimName`	`string`	ClaimName is the name of the PVC
`filesystemOverhead`	`string`	Percentage of filesystem's size to be reserved when resizing the PVC
`preallocated`	`boolean`	Preallocated indicates if the PVC's storage is preallocated or not
`requests`	`object`	Requests represents the resources requested by the corresponding PVC spec
`volumeMode`	`string`	VolumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

.status.migratedVolumes[].sourcePVCInfo.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.migratedVolumes[].sourcePVCInfo.accessModes[]

Type

`string`

.status.migratedVolumes[].sourcePVCInfo.capacity

Description

Capacity represents the capacity set on the corresponding PVC status

Type

`object`

.status.migratedVolumes[].sourcePVCInfo.requests

Description

Requests represents the resources requested by the corresponding PVC spec

Type

`object`

.status.migrationState

Description

Represents the status of a live migration

Type

`object`

Property	Type	Description
`abortRequested`	`boolean`	Indicates that the migration has been requested to abort
`abortStatus`	`string`	Indicates the final status of the live migration abortion
`completed`	`boolean`	Indicates the migration completed
`endTimestamp`	`string`	The time the migration action ended
`failed`	`boolean`	Indicates that the migration failed
`failureReason`	`string`	Contains the reason why the migration failed
`migrationConfiguration`	`object`	Migration configurations to apply
`migrationNetworkType`	`string`	The type of migration network, either 'pod' or 'migration'
`migrationPolicyName`	`string`	Name of the migration policy. If string is empty, no policy is matched

Property	Type	Description
`migrationUid`	`string`	The VirtualMachineInstanceMigration object associated with this migration
`mode`	`string`	Lets us know if the vmi is currently running pre or post copy migration
`sourceNode`	`string`	The source node that the VMI originated on
`sourcePersistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here
`sourcePod`	`string`	
`sourceState`	`object`	SourceState contains migration state managed by the source virt handler
`startTimestamp`	`string`	The time the migration action began
`targetAttachmentPodUID`	`string`	The UID of the target attachment pod for hotplug volumes

Property	Type	Description
`targetCPUSet`	`array`	If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node
`targetDirectMigrationNodePorts`	`object`	The list of ports opened for live migration on the destination node
`targetNode`	`string`	The target node that the VMI is moving to
`targetNodeAddress`	`string`	The address of the target node to use for the migration
`targetNodeDomainDetected`	`boolean`	The Target Node has seen the Domain Start Event
`targetNodeDomainReadyTimestamp`	`string`	The timestamp at which the target node detects the domain is active
`targetNodeTopology`	`string`	If the VMI requires dedicated CPUs, this field will hold the numa topology on the target node
`targetPersistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-

Property	Type	Description
		storage), its target PVC name is saved here
`targetPod`	`string`	The target pod that the VMI is moving to
`targetState`	`object`	TargetState contains migration state managed by the target virt handler

.status.migrationState.migrationConfiguration

Description

Migration configurations to apply

Type

`object`

Property	Type	Description
`allowAutoConverge`	`boolean`	AllowAutoConverge allows the platform to compromise performance/availability of VMIs to guarantee successful VMI live migrations. Defaults to false
`allowPostCopy`	`boolean`	AllowPostCopy enables post-copy live migrations. Such migrations allow even the busiest VMIs to successfully live-migrate. Howeve

Property	Type	Description
		events like a network failure can cause a VMI crash. If set to true, migrations will still start in pre-copy but switch to post-copy when CompletionTimeoutPerGiB triggers Defaults to false
`allowWorkloadDisruption`	`boolean`	AllowWorkloadDisruption indicates that the migration shouldn't be canceled after acceptableCompletionTime is exceeded. Instead, if permitted, migration will be switched to post-copy or the VMI will be paused to allow the migration to complete
`bandwidthPerMigration`	`integer|string`	BandwidthPerMigration limits the amount of network bandwidth live migrations are allowed to use. The value is in quantity per second. Defaults to 0 (no limit)
`completionTimeoutPerGiB`	`integer`	CompletionTimeoutPerGiB is the maximum number of seconds per GiB a migration is allowed to take. the timeout is reached, the migrati will be either paused, switched to post-copy or cancelled depending on other settings. Defaults to 150

Property	Type	Description
`disableTLS`	`boolean`	When set to true, DisableTLS will disable the additional layer of live migration encryption provided by KubeVirt. This is usually a bad idea Defaults to false
`matchSELinuxLevelOnMigration`	`boolean`	By default, the SELinux level of target virt-launcher pods is forced t the level of the source virt-launche When set to true, MatchSELinuxLevelOnMigration le the CRI auto-assign a random leve to the target. That will ensure the target virt-launcher doesn't share categories with another pod on the node. However, migrations will fail when using RWX volumes that don automatically deal with SELinux levels.
`network`	`string`	Network is the name of the CNI network to use for live migrations. By default, migrations go through the pod network.

Property	Type	Description
`nodeDrainTaintKey`	`string`	NodeDrainTaintKey defines the tai key that indicates a node should b drained. Note: this option relies on the deprecated node taint feature. Default: kubevirt.io/drain
`parallelMigrationsPerCluster`	`integer`	ParallelMigrationsPerCluster is the total number of concurrent live migrations allowed cluster-wide. Defaults to 5
`parallelOutboundMigrationsPerNode`	`integer`	ParallelOutboundMigrationsPerNo is the maximum number of concurrent outgoing live migrations allowed per node. Defaults to 2
`progressTimeout`	`integer`	ProgressTimeout is the maximum number of seconds a live migration is allowed to make no progress. Hitting this timeout means a migration transferred 0 data for tha many seconds. The migration is then considered stuck and therefo cancelled. Defaults to 150
`unsafeMigrationOverride`	`boolean`	UnsafeMigrationOverride allows liv migrations to occur even if the compatibility check indicates the

Property	Type	Description
		migration will be unsafe to the gue Defaults to false

.status.migrationState.sourceState

Description

SourceState contains migration state managed by the source virt handler

Type

`object`

Property	Type	Description
`domainName`	`string`	The name of the domain on the source libvirt domain
`domainNamespace`	`string`	Namespace used in the name of the source libvirt domain. Can be used to find and modify paths in the domain
`migrationUID`	`string`	The Source VirtualMachineInstanceMigration object associated with this migration
`node`	`string`	The source node that the VMI originated on

Property	Type	Description
`nodeSelectors`	`object`	Node selectors needed by the target to start the receiving pod.
`persistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here
`pod`	`string`	The source pod that the VMI is originated on
`selinuxContext`	`string`	SELinuxContext is the actual SELinux context of the pod
`syncAddress`	`string`	The ip address/fqdn:port combination to use to synchronize the VMI with the target.
`virtualMachineInstanceUID`	`string`	VirtualMachineInstanceUID is the UID of the target virtual machine instance

.status.migrationState.sourceState.nodeSelectors

Description

Node selectors needed by the target to start the receiving pod.

Type

`object`

.status.migrationState.targetCPUSet

Description

If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node

Type

`array`

.status.migrationState.targetCPUSet[]

Type

`integer`

.status.migrationState.targetDirectMigrationNodePorts

Description

The list of ports opened for live migration on the destination node

Type

`object`

.status.migrationState.targetState

Description

TargetState contains migration state managed by the target virt handler

Type

`object`

Property	Type	Description
`attachmentPodUID`	`string`	The UID of the target attachment pod for hotplug volumes

Property	Type	Description
`cpuSet`	`array`	If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node
`directMigrationNodePorts`	`object`	The list of ports opened for live migration on the destination node
`domainDetected`	`boolean`	The Target Node has seen the Domain Start Event
`domainName`	`string`	The name of the domain on the source libvirt domain
`domainNamespace`	`string`	Namespace used in the name of the source libvirt domain. Can be used to find and modify paths in the domain
`domainReadyTimestamp`	`string`	The timestamp at which the target node detects the domain is active
`migrationUID`	`string`	The Source VirtualMachineInstanceMigration object associated with this migration

Property	Type	Description
`node`	`string`	The source node that the VMI originated on
`nodeAddress`	`string`	The address of the target node to use for the migration
`nodeTopology`	`string`	If the VMI requires dedicated CPUs, this field will hold the numa topology on the target node
`persistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here
`pod`	`string`	The source pod that the VMI is originated on
`selinuxContext`	`string`	SELinuxContext is the actual SELinux context of the pod
`syncAddress`	`string`	The ip address/fqdn:port combination to use to synchronize the VMI with the target.

Property	Type	Description
`virtualMachineInstanceUID`	`string`	VirtualMachineInstanceUID is the UID of the target virtual machine instance

.status.migrationState.targetState.cpuSet

Description

If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node

Type

`array`

.status.migrationState.targetState.cpuSet[]

Type

`integer`

.status.migrationState.targetState.directMigrationNodePorts

Description

The list of ports opened for live migration on the destination node

Type

`object`

.status.phaseTransitionTimestamps

Description

PhaseTransitionTimestamp is the timestamp of when the last phase change occurred

Type

`array`

.status.phaseTransitionTimestamps[]

Description

VirtualMachineInstancePhaseTransitionTimestamp gives a timestamp in relation to when a phase is set on a vmi

Type

`object`

Property	Type	Description
`phase`	`string`	Phase is the status of the VirtualMachineInstance in kubernetes world. It is not the VirtualMachineInstance status, but partially correlates to it.
`phaseTransitionTimestamp`	`string`	PhaseTransitionTimestamp is the timestamp of when the phase change occurred

.status.topologyHints

Type

`object`

Property	Type	Description
`tscFrequency`	`integer`	

.status.volumeStatus

Description

VolumeStatus contains the statuses of all the volumes

Type

`array`

.status.volumeStatus[]

Description

VolumeStatus represents information about the status of volumes attached to the VirtualMachineInstance.

Type

`object`

Required

`name` `target`

Property	Type	Description
`containerDiskVolume`	`object`	ContainerDiskVolume shows info about the containerdisk, if the volume is a containerdisk
`hotplugVolume`	`object`	If the volume is hotplug, this will contain the hotplug status.
`memoryDumpVolume`	`object`	If the volume is memorydump volume, this will contain the memorydump info.

Property	Type	Description
`message`	`string`	Message is a detailed message about the current hotplug volume phase
`name`	`string`	Name is the name of the volume
`persistentVolumeClaimInfo`	`object`	PersistentVolumeClaimInfo is information about the PVC that handler requires during start flow
`phase`	`string`	Phase is the phase
`reason`	`string`	Reason is a brief description of why we are in the current hotplug volume phase
`size`	`integer`	Represents the size of the volume
`target`	`string`	Target is the target name used when adding the volume to the VM, eg: vda

.status.volumeStatus[].containerDiskVolume

Description

ContainerDiskVolume shows info about the containerdisk, if the volume is a containerdisk

Type

`object`

Property	Type	Description
`checksum`	`integer`	Checksum is the checksum of the rootdisk or kernel artifacts inside the containerdisk

.status.volumeStatus[].hotplugVolume

Description

If the volume is hotplug, this will contain the hotplug status.

Type

`object`

Property	Type	Description
`attachPodName`	`string`	AttachPodName is the name of the pod used to attach the volume to the node.
`attachPodUID`	`string`	AttachPodUID is the UID of the pod used to attach the volume to the node.

.status.volumeStatus[].memoryDumpVolume

Description

If the volume is memorydump volume, this will contain the memorydump info.

Type

`object`

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the pvc the memory was dumped to
`endTimestamp`	`string`	EndTimestamp is the time when the memory dump completed
`startTimestamp`	`string`	StartTimestamp is the time when the memory dump started
`targetFileName`	`string`	TargetFileName is the name of the memory dump output

.status.volumeStatus[].persistentVolumeClaimInfo

Description

PersistentVolumeClaimInfo is information about the PVC that handler requires during start flow

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗

Property	Type	Description
`capacity`	`object`	Capacity represents the capacity set on the corresponding PVC status
`claimName`	`string`	ClaimName is the name of the PVC
`filesystemOverhead`	`string`	Percentage of filesystem's size to be reserved when resizing the PVC
`preallocated`	`boolean`	Preallocated indicates if the PVC's storage is preallocated or not
`requests`	`object`	Requests represents the resources requested by the corresponding PVC spec
`volumeMode`	`string`	VolumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

.status.volumeStatus[].persistentVolumeClaimInfo.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.volumeStatus[].persistentVolumeClaimInfo.accessModes[]

Type

`string`

.status.volumeStatus[].persistentVolumeClaimInfo.capacity

Description

Capacity represents the capacity set on the corresponding PVC status

Type

`object`

.status.volumeStatus[].persistentVolumeClaimInfo.requests

Description

Requests represents the resources requested by the corresponding PVC spec

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstances`
  - `DELETE`: delete collection of VirtualMachineInstance

- `GET`: list objects of kind VirtualMachineInstance
- `POST`: create a new VirtualMachineInstance

- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstances/{name}`
  - `DELETE`: delete the specified VirtualMachineInstance
  - `GET`: read the specified VirtualMachineInstance
  - `PATCH`: partially update the specified VirtualMachineInstance
  - `PUT`: replace the specified VirtualMachineInstance

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstances

HTTP method

`DELETE`

Description

delete collection of VirtualMachineInstance

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineInstance

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceList` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineInstance

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineInstance` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstance` schema
201 - Created	`VirtualMachineInstance` schema
202 - Accepted	`VirtualMachineInstance` schema
401 - Unauthorized	Empty

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstances/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineInstance

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineInstance

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstance` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineInstance

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstance` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineInstance

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineInstance` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstance` schema
201 - Created	`VirtualMachineInstance` schema
401 - Unauthorized	Empty

VirtualMachineInstanceMigration [virtualmachineinstancemigrations.kubevirt.io/v1]

Description

VirtualMachineInstanceMigration represents the object tracking a VMI's migration to another host in the cluster

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	VirtualMachineInstanceMigration reprents information pertaining to a VMI's migration.

.spec

Type

`object`

Property	Type	Description
`addedNodeSelector`	`object`	AddedNodeSelector is an additional selector that can be used to complement a NodeSelector or NodeAffinity as set on the VM to restrict the set of allowed target nodes for a migration. In case of key collisions, values set on the VM objects are going to be preserved to ensure that addedNodeSelector can only restrict but not bypass constraints already set on the VM object.
`priority`	`string`	Priority of the migration. This can be one of 'system-critical', 'user-triggered', 'system-maintenance'.
`receive`	`object`	If receieve is specified, this VirtualMachineInstanceMigration will be considered the target
`sendTo`	`object`	If sendTo is specified, this VirtualMachineInstanceMigration will be considered the source
`vmiName`	`string`	The name of the VMI to perform the migration on. VMI must exist in the migration objects namespace

.spec.addedNodeSelector

Description

AddedNodeSelector is an additional selector that can be used to complement a NodeSelector or NodeAffinity as set on the VM to restrict the set of allowed target nodes for a migration. In case of key collisions, values set on the VM objects are going to be preserved to ensure that addedNodeSelector can only restrict but not bypass constraints already set on the VM object.

Type

`object`

.spec.receive

Description

If receieve is specified, this VirtualMachineInstanceMigration will be considered the target

Type

`object`

Required

`migrationID`

Property	Type	Description
`migrationID`	`string`	A unique identifier to identify this migration.

.spec.sendTo

Description

If sendTo is specified, this VirtualMachineInstanceMigration will be considered the source

Type

`object`

Required

`connectURL` `migrationID`

Property	Type	Description
`connectURL`	`string`	The synchronization controller URL to connect to.
`migrationID`	`string`	A unique identifier to identify this migration.

.status

Description

VirtualMachineInstanceMigration reprents information pertaining to a VMI's migration.

Type

`object`

Property	Type	Description
`conditions`	`array`	
`migrationState`	`object`	Represents the status of a live migration
`phase`	`string`	VirtualMachineInstanceMigrationPhase is a label for the condition of a VirtualMachineInstanceMigration at the current time.
`phaseTransitionTimestamps`	`array`	PhaseTransitionTimestamp is the timestamp of when the last phase change occurred

Property	Type	Description
`synchronizationAddresses`	`array`	The synchronization addresses one can use to connect to the synchronization controller, includes the port, if multiple addresses are available, the first one is reported in the synchronizationAddress field.

.status.conditions

Type

`array`

.status.conditions[]

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	

.status.migrationState

Description

Represents the status of a live migration

Type

`object`

Property	Type	Description
`abortRequested`	`boolean`	Indicates that the migration has been requested to abort
`abortStatus`	`string`	Indicates the final status of the live migration abortion
`completed`	`boolean`	Indicates the migration completed
`endTimestamp`	`string`	The time the migration action ended
`failed`	`boolean`	Indicates that the migration failed
`failureReason`	`string`	Contains the reason why the migration failed
`migrationConfiguration`	`object`	Migration configurations to apply

Property	Type	Description
`migrationNetworkType`	`string`	The type of migration network, either 'pod' or 'migration'
`migrationPolicyName`	`string`	Name of the migration policy. If string is empty, no policy is matched
`migrationUid`	`string`	The VirtualMachineInstanceMigration object associated with this migration
`mode`	`string`	Lets us know if the vmi is currently running pre or post copy migration
`sourceNode`	`string`	The source node that the VMI originated on
`sourcePersistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here
`sourcePod`	`string`	

Property	Type	Description
`sourceState`	`object`	SourceState contains migration state managed by the source virt handler
`startTimestamp`	`string`	The time the migration action began
`targetAttachmentPodUID`	`string`	The UID of the target attachment pod for hotplug volumes
`targetCPUSet`	`array`	If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node
`targetDirectMigrationNodePorts`	`object`	The list of ports opened for live migration on the destination node
`targetNode`	`string`	The target node that the VMI is moving to
`targetNodeAddress`	`string`	The address of the target node to use for the migration

Property	Type	Description
`targetNodeDomainDetected`	`boolean`	The Target Node has seen the Domain Start Event
`targetNodeDomainReadyTimestamp`	`string`	The timestamp at which the target node detects the domain is active
`targetNodeTopology`	`string`	If the VMI requires dedicated CPUs, this field will hold the numa topology on the target node
`targetPersistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its target PVC name is saved here
`targetPod`	`string`	The target pod that the VMI is moving to
`targetState`	`object`	TargetState contains migration state managed by the target virt handler

.status.migrationState.migrationConfiguration

Description

Migration configurations to apply

Type

`object`

Property	Type	Description
`allowAutoConverge`	`boolean`	AllowAutoConverge allows the platform to compromise performance/availability of VMIs to guarantee successful VMI live migrations. Defaults to false
`allowPostCopy`	`boolean`	AllowPostCopy enables post-copy live migrations. Such migrations allow even the busiest VMIs to successfully live-migrate. However events like a network failure can cause a VMI crash. If set to true, migrations will still start in pre-copy but switch to post-copy when CompletionTimeoutPerGiB triggers Defaults to false
`allowWorkloadDisruption`	`boolean`	AllowWorkloadDisruption indicates that the migration shouldn't be canceled after acceptableCompletionTime is exceeded. Instead, if permitted, migration will be switched to post-copy or the VMI will be paused to allow the migration to complete

Property	Type	Description
`bandwidthPerMigration`	`integer|string`	BandwidthPerMigration limits the amount of network bandwidth live migrations are allowed to use. The value is in quantity per second. Defaults to 0 (no limit)
`completionTimeoutPerGiB`	`integer`	CompletionTimeoutPerGiB is the maximum number of seconds per GiB a migration is allowed to take. the timeout is reached, the migrati will be either paused, switched to post-copy or cancelled depending on other settings. Defaults to 150
`disableTLS`	`boolean`	When set to true, DisableTLS will disable the additional layer of live migration encryption provided by KubeVirt. This is usually a bad idea Defaults to false
`matchSELinuxLevelOnMigration`	`boolean`	By default, the SELinux level of target virt-launcher pods is forced the level of the source virt-launche When set to true, MatchSELinuxLevelOnMigration le the CRI auto-assign a random leve to the target. That will ensure the target virt-launcher doesn't share categories with another pod on the node. However, migrations will fail

Property	Type	Description
		when using RWX volumes that do automatically deal with SELinux levels.
`network`	`string`	Network is the name of the CNI network to use for live migrations. By default, migrations go through the pod network.
`nodeDrainTaintKey`	`string`	NodeDrainTaintKey defines the tai key that indicates a node should b drained. Note: this option relies on the deprecated node taint feature. Default: kubevirt.io/drain
`parallelMigrationsPerCluster`	`integer`	ParallelMigrationsPerCluster is the total number of concurrent live migrations allowed cluster-wide. Defaults to 5
`parallelOutboundMigrationsPerNode`	`integer`	ParallelOutboundMigrationsPerNo is the maximum number of concurrent outgoing live migrations allowed per node. Defaults to 2
`progressTimeout`	`integer`	ProgressTimeout is the maximum number of seconds a live migration is allowed to make no progress.

Property	Type	Description
		Hitting this timeout means a migration transferred 0 data for tha many seconds. The migration is then considered stuck and therefo cancelled. Defaults to 150
`unsafeMigrationOverride`	`boolean`	UnsafeMigrationOverride allows liv migrations to occur even if the compatibility check indicates the migration will be unsafe to the gue Defaults to false

.status.migrationState.sourceState

Description

SourceState contains migration state managed by the source virt handler

Type

`object`

Property	Type	Description
`domainName`	`string`	The name of the domain on the source libvirt domain
`domainNamespace`	`string`	Namespace used in the name of the source libvirt domain. Can be used to find and modify paths in the domain

Property	Type	Description
`migrationUID`	`string`	The Source VirtualMachineInstanceMigration object associated with this migration
`node`	`string`	The source node that the VMI originated on
`nodeSelectors`	`object`	Node selectors needed by the target to start the receiving pod.
`persistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here
`pod`	`string`	The source pod that the VMI is originated on
`selinuxContext`	`string`	SELinuxContext is the actual SELinux context of the pod
`syncAddress`	`string`	The ip address/fqdn:port combination to use to synchronize the VMI with the target.

Property	Type	Description
`virtualMachineInstanceUID`	`string`	VirtualMachineInstanceUID is the UID of the target virtual machine instance

.status.migrationState.sourceState.nodeSelectors

Description

Node selectors needed by the target to start the receiving pod.

Type

`object`

.status.migrationState.targetCPUSet

Description

If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node

Type

`array`

.status.migrationState.targetCPUSet[]

Type

`integer`

.status.migrationState.targetDirectMigrationNodePorts

Description

The list of ports opened for live migration on the destination node

Type

`object`

.status.migrationState.targetState

Description

TargetState contains migration state managed by the target virt handler

Type

`object`

Property	Type	Description
`attachmentPodUID`	`string`	The UID of the target attachment pod for hotplug volumes
`cpuSet`	`array`	If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node
`directMigrationNodePorts`	`object`	The list of ports opened for live migration on the destination node
`domainDetected`	`boolean`	The Target Node has seen the Domain Start Event
`domainName`	`string`	The name of the domain on the source libvirt domain

Property	Type	Description
`domainNamespace`	`string`	Namespace used in the name of the source libvirt domain. Can be used to find and modify paths in the domain
`domainReadyTimestamp`	`string`	The timestamp at which the target node detects the domain is active
`migrationUID`	`string`	The Source VirtualMachineInstanceMigration object associated with this migration
`node`	`string`	The source node that the VMI originated on
`nodeAddress`	`string`	The address of the target node to use for the migration
`nodeTopology`	`string`	If the VMI requires dedicated CPUs, this field will hold the numa topology on the target node
`persistentStatePVCName`	`string`	If the VMI being migrated uses persistent features (backend-storage), its source PVC name is saved here

Property	Type	Description
`pod`	`string`	The source pod that the VMI is originated on
`selinuxContext`	`string`	SELinuxContext is the actual SELinux context of the pod
`syncAddress`	`string`	The ip address/fqdn:port combination to use to synchronize the VMI with the target.
`virtualMachineInstanceUID`	`string`	VirtualMachineInstanceUID is the UID of the target virtual machine instance

.status.migrationState.targetState.cpuSet

Description

If the VMI requires dedicated CPUs, this field will hold the dedicated CPU set on the target node

Type

`array`

.status.migrationState.targetState.cpuSet[]

Type

`integer`

.status.migrationState.targetState.directMigrationNodePorts

Description

The list of ports opened for live migration on the destination node

Type

`object`

.status.phaseTransitionTimestamps

Description

PhaseTransitionTimestamp is the timestamp of when the last phase change occurred

Type

`array`

.status.phaseTransitionTimestamps[]

Description

VirtualMachineInstanceMigrationPhaseTransitionTimestamp gives a timestamp in relation to when a phase is set on a vmi

Type

`object`

Property	Type	Description
`phase`	`string`	Phase is the status of the VirtualMachineInstanceMigrationPhase in kubernetes world. It is not the VirtualMachineInstanceMigrationPhase status, but partially correlates to it.

Property	Type	Description
`phaseTransitionTimestamp`	`string`	PhaseTransitionTimestamp is the timestamp of when the phase change occurred

.status.synchronizationAddresses

Description

The synchronization addresses one can use to connect to the synchronization controller, includes the port, if multiple addresses are available, the first one is reported in the synchronizationAddress field.

Type

`array`

.status.synchronizationAddresses[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations`
  - `DELETE`: delete collection of VirtualMachineInstanceMigration
  - `GET`: list objects of kind VirtualMachineInstanceMigration
  - `POST`: create a new VirtualMachineInstanceMigration

- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations/{name}`
  - `DELETE`: delete the specified VirtualMachineInstanceMigration
  - `GET`: read the specified VirtualMachineInstanceMigration
  - `PATCH`: partially update the specified VirtualMachineInstanceMigration
  - `PUT`: replace the specified VirtualMachineInstanceMigration
- `/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations/{name}/status`
  - `GET`: read status of the specified VirtualMachineInstanceMigration
  - `PATCH`: partially update status of the specified VirtualMachineInstanceMigration
  - `PUT`: replace status of the specified VirtualMachineInstanceMigration

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations

HTTP method

`DELETE`

Description

delete collection of VirtualMachineInstanceMigration

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineInstanceMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigrationList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineInstanceMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
201 - Created	`VirtualMachineInstanceMigration` schema
202 - Accepted	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineInstanceMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineInstanceMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
201 - Created	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

/apis/kubevirt.io/v1/namespaces/{namespace}/virtualmachineinstancemigrations/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineInstanceMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineInstanceMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineInstanceMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineInstanceMigration` schema
201 - Created	`VirtualMachineInstanceMigration` schema
401 - Unauthorized	Empty

VirtualMachineClone [virtualmachineclones.clone.kubevirt.io/v1beta1]

Description

VirtualMachineClone is a CRD that clones one VM into another.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	

.spec

Type

`object`

Required

`source`

Property	Type	Description
`annotationFilters`	`array`	Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters ↗.

Property	Type	Description
`labelFilters`	`array`	Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters ↗.
`newMacAddresses`	`object`	NewMacAddresses manually sets that target interfaces' mac addresses. The key is the interface name and the value is the new mac address. If this field is not specified, a new MAC address will be generated automatically, as for any interface that is not included in this map.
`newSMBiosSerial`	`string`	NewSMBiosSerial manually sets that target's SMbios serial. If this field is not specified, a new serial will be generated automatically.
`patches`	`array`	Patches holds JSON patches to apply to target. Patches should fit the target's Kind. Example: '{"op": "add", "path": "/spec/template/metadata/labels/example", "value": "new-label"}'
`source`	`object`	Source is the object that would be cloned. Currently supported source types are: VirtualMachine of kubevirt.io API group, VirtualMachineSnapshot of snapshot.kubevirt.io API group

Property	Type	Description
`target`	`object`	Target is the outcome of the cloning process. Currently supported source types are: • VirtualMachine of kubevirt.io API group • Empty (nil). If the target is not provided, the target type would default to VirtualMachine and a random name would be generated for the target. The target's name can be viewed by inspecting status "TargetName" field below.
`template`	`object`	For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters ↗.

.spec.annotationFilters

Description

Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters.

Type

`array`

.spec.annotationFilters[]

Type

`string`

.spec.labelFilters

Description

Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters.

Type

`array`

.spec.labelFilters[]

Type

`string`

.spec.newMacAddresses

Description

NewMacAddresses manually sets that target interfaces' mac addresses. The key is the interface name and the value is the new mac address. If this field is not specified, a new MAC address will be generated automatically, as for any interface that is not included in this map.

Type

`object`

.spec.patches

Description

Patches holds JSON patches to apply to target. Patches should fit the target's Kind. Example: '{"op": "add", "path": "/spec/template/metadata/labels/example", "value": "new-label"}'

Type

`array`

.spec.patches[]

Type

`string`

.spec.source

Description

Source is the object that would be cloned. Currently supported source types are: VirtualMachine of kubevirt.io API group, VirtualMachineSnapshot of snapshot.kubevirt.io API group

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.target

Description

Target is the outcome of the cloning process. Currently supported source types are: - VirtualMachine of kubevirt.io API group - Empty (nil). If the target is not provided, the target

type would default to VirtualMachine and a random name would be generated for the target. The target's name can be viewed by inspecting status "TargetName" field below.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.template

Description

For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters.

Type

`object`

Property	Type	Description
`annotationFilters`	`array`	Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters ↗.
`labelFilters`	`array`	Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters ↗.

.spec.template.annotationFilters

Description

Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters.

Type

`array`

.spec.template.annotationFilters[]

Type

`string`

.spec.template.labelFilters

Description

Example use: "!some/key*". For a detailed description, please refer to https://kubevirt.io/user-guide/operations/clone_api/#label-annotation-filters.

Type

`array`

.spec.template.labelFilters[]

Type

`string`

.status

Type

`object`

Property	Type	Description
`conditions`	`array`	
`creationTime`	`string`	
`phase`	`string`	
`restoreName`	`string`	
`snapshotName`	`string`	
`targetName`	`string`	

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines conditions

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	ConditionType is the const type for Conditions

API Endpoints

The following API endpoints are available:

- `/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones`
  - `DELETE`: delete collection of VirtualMachineClone
  - `GET`: list objects of kind VirtualMachineClone
  - `POST`: create a new VirtualMachineClone
- `/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones/{name}`
  - `DELETE`: delete the specified VirtualMachineClone
  - `GET`: read the specified VirtualMachineClone

- `PATCH`: partially update the specified VirtualMachineClone
- `PUT`: replace the specified VirtualMachineClone

- `/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones/{name}/status`
  - `GET`: read status of the specified VirtualMachineClone
  - `PATCH`: partially update status of the specified VirtualMachineClone
  - `PUT`: replace status of the specified VirtualMachineClone

/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones

HTTP method

`DELETE`

Description

delete collection of VirtualMachineClone

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineClone

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineCloneList` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
201 - Created	`VirtualMachineClone` schema
202 - Accepted	`VirtualMachineClone` schema
401 - Unauthorized	Empty

/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineClone

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
201 - Created	`VirtualMachineClone` schema
401 - Unauthorized	Empty

/apis/clone.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineclones/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineClone

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineClone

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClone` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClone` schema
201 - Created	`VirtualMachineClone` schema
401 - Unauthorized	Empty

VirtualMachineSnapshot [virtualmachinesnapshots.snapshot.kubevirt.io/v1beta1]

Description

VirtualMachineSnapshot defines the operation of snapshotting a VM

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineSnapshotSpec is the spec for a VirtualMachineSnapshot resource
`status`	`object`	VirtualMachineSnapshotStatus is the status for a VirtualMachineSnapshot resource

.spec

Description

VirtualMachineSnapshotSpec is the spec for a VirtualMachineSnapshot resource

Type

`object`

Required

`source`

Property	Type	Description
`deletionPolicy`	`string`	DeletionPolicy defines that to do with VirtualMachineSnapshot when VirtualMachineSnapshot is deleted
`failureDeadline`	`string`	This time represents the number of seconds we permit the vm snapshot to take. In case we pass this deadline we mark this snapshot as failed. Defaults to DefaultFailureDeadline - 5min
`source`	`object`	TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

.spec.source

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in

Property	Type	Description
		the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.status

Description

VirtualMachineSnapshotStatus is the status for a VirtualMachineSnapshot resource

Type

`object`

Property	Type	Description
`conditions`	`array`	
`creationTime`	`string`	
`error`	`object`	Error is the last error encountered during the snapshot/restore
`indications`	`array`	Deprecated: Use SourceIndications instead. This field will be removed in a future version.

Property	Type	Description
`phase`	`string`	VirtualMachineSnapshotPhase is the current phase of the VirtualMachineSnapshot
`readyToUse`	`boolean`	
`snapshotVolumes`	`object`	SnapshotVolumesLists includes the list of volumes which were included in the snapshot and volumes which were excluded from the snapshot
`sourceIndications`	`array`	
`sourceUID`	`string`	UID is a type that holds unique ID values, including UUIDs. Because we don't ONLY use UUIDs, this is an alias to string. Being a type captures intent and helps make sure that UIDs and names do not get conflated.
`virtualMachineSnapshotContentName`	`string`	

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines conditions

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	ConditionType is the const type for Conditions

.status.error

Description

Error is the last error encountered during the snapshot/restore

Type

`object`

Property	Type	Description
`message`	`string`	
`time`	`string`	

.status.indications

Description

Deprecated: Use SourceIndications instead. This field will be removed in a future version.

Type

`array`

.status.indications[]

Description

Indication is a way to indicate the state of the vm when taking the snapshot

Type

`string`

.status.snapshotVolumes

Description

SnapshotVolumesLists includes the list of volumes which were included in the snapshot and volumes which were excluded from the snapshot

Type

`object`

Property	Type	Description
`excludedVolumes`	`array`	
`includedVolumes`	`array`	

.status.snapshotVolumes.excludedVolumes

Type

`array`

.status.snapshotVolumes.excludedVolumes[]

Type

`string`

.status.snapshotVolumes.includedVolumes

Type

`array`

.status.snapshotVolumes.includedVolumes[]

Type

`string`

.status.sourceIndications

Type

`array`

.status.sourceIndications[]

Description

SourceIndication provides an indication of the source VM with its description message

Type

`object`

Required

`indication` `message`

Property	Type	Description
`indication`	`string`	Indication is the indication type
`message`	`string`	Message provides a description message of the indication

API Endpoints

The following API endpoints are available:

- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots`
  - `DELETE`: delete collection of VirtualMachineSnapshot
  - `GET`: list objects of kind VirtualMachineSnapshot
  - `POST`: create a new VirtualMachineSnapshot
- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots/{name}`
  - `DELETE`: delete the specified VirtualMachineSnapshot
  - `GET`: read the specified VirtualMachineSnapshot
  - `PATCH`: partially update the specified VirtualMachineSnapshot
  - `PUT`: replace the specified VirtualMachineSnapshot
- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots/{name}/status`
  - `GET`: read status of the specified VirtualMachineSnapshot
  - `PATCH`: partially update status of the specified VirtualMachineSnapshot
  - `PUT`: replace status of the specified VirtualMachineSnapshot

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots

HTTP method

`DELETE`

Description

delete collection of VirtualMachineSnapshot

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineSnapshot

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshotList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineSnapshot` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
201 - Created	`VirtualMachineSnapshot` schema

HTTP code	Response body
202 - Accepted	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineSnapshot

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineSnapshot` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
201 - Created	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinesnapshots/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineSnapshot

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineSnapshot

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineSnapshot` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineSnapshot` schema
201 - Created	`VirtualMachineSnapshot` schema
401 - Unauthorized	Empty

VirtualMachineRestore [virtualmachinerestores.snapshot.kubevirt.io/v1beta1]

Description

VirtualMachineRestore defines the operation of restoring a VM

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineRestoreSpec is the spec for a VirtualMachineRestore resource
`status`	`object`	VirtualMachineRestoreStatus is the status for a VirtualMachineRestore resource

.spec

Description

VirtualMachineRestoreSpec is the spec for a VirtualMachineRestore resource

Type

`object`

Required

`target` `virtualMachineSnapshotName`

Property	Type	Description
`patches`	`array`	If the target for the restore does not exist, it will be created. Patches holds JSON patches that would be applied to the target manifest before it's created. Patches should fit the target's Kind. Example for a patch: {"op": "replace", "path": "/metadata/name", "value": "new-vm-name"}
`target`	`object`	initially only VirtualMachine type supported
`targetReadinessPolicy`	`string`	TargetReadinessPolicy defines how to handle the restore in case the target is not ready
`virtualMachineSnapshotName`	`string`	
`volumeOwnershipPolicy`	`string`	VolumeOwnershipPolicy defines what owns volumes once they're restored
`volumeRestoreOverrides`	`array`	VolumeRestoreOverrides gives the option to change properties of each restored volume For example, specifying the name of the restored volume, or adding labels/annotations to it

Property	Type	Description
`volumeRestorePolicy`	`string`	VolumeRestorePolicy defines how to handle the restore of snapshotted volumes

.spec.patches

Description

If the target for the restore does not exist, it will be created. Patches holds JSON patches that would be applied to the target manifest before it's created. Patches should fit the target's Kind. Example for a patch: {"op": "replace", "path": "/metadata/name", "value": "new-vm-name"}

Type

`array`

.spec.patches[]

Type

`string`

.spec.target

Description

initially only VirtualMachine type supported

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.volumeRestoreOverrides

Description

VolumeRestoreOverrides gives the option to change properties of each restored volume For example, specifying the name of the restored volume, or adding labels/annotations to it

Type

`array`

.spec.volumeRestoreOverrides[]

Description

VolumeRestoreOverride specifies how a volume should be restored from a VirtualMachineSnapshot

Type

`object`

Property	Type	Description
`annotations`	`object`	

Property	Type	Description
`labels`	`object`	
`restoreName`	`string`	
`volumeName`	`string`	

.spec.volumeRestoreOverrides[].annotations

Type

`object`

.spec.volumeRestoreOverrides[].labels

Type

`object`

.status

Description

VirtualMachineRestoreStatus is the status for a VirtualMachineRestore resource

Type

`object`

Property	Type	Description
`complete`	`boolean`	
`conditions`	`array`	
`deletedDataVolumes`	`array`	
`restoreTime`	`string`	

Property	Type	Description
`restores`	`array`	

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines conditions

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	ConditionType is the const type for Conditions

.status.deletedDataVolumes

Type

`array`

.status.deletedDataVolumes[]

Type

`string`

.status.restores

Type

`array`

.status.restores[]

Description

VolumeRestore contains the data needed to restore a PVC

Type

`object`

Required

`persistentVolumeClaim` `volumeName` `volumeSnapshotName`

Property	Type	Description
`dataVolumeName`	`string`	
`persistentVolumeClaim`	`string`	
`volumeName`	`string`	
`volumeSnapshotName`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores`
  - `DELETE`: delete collection of VirtualMachineRestore
  - `GET`: list objects of kind VirtualMachineRestore
  - `POST`: create a new VirtualMachineRestore
- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores/{name}`
  - `DELETE`: delete the specified VirtualMachineRestore
  - `GET`: read the specified VirtualMachineRestore
  - `PATCH`: partially update the specified VirtualMachineRestore
  - `PUT`: replace the specified VirtualMachineRestore
- `/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores/{name}/status`
  - `GET`: read status of the specified VirtualMachineRestore
  - `PATCH`: partially update status of the specified VirtualMachineRestore
  - `PUT`: replace status of the specified VirtualMachineRestore

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores

HTTP method

`DELETE`

Description

delete collection of VirtualMachineRestore

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineRestore

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestoreList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
201 - Created	`VirtualMachineRestore` schema
202 - Accepted	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineRestore

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
201 - Created	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

/apis/snapshot.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachinerestores/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineRestore

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineRestore

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineRestore` schema
201 - Created	`VirtualMachineRestore` schema

HTTP code	Response body
401 - Unauthorized	Empty

DataVolume [datavolumes.cdi.kubevirt.io/v1beta1]

Description

DataVolume is an abstraction on top of PersistentVolumeClaims to allow easy population of those PersistentVolumeClaims with relation to VirtualMachines

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataVolumeSpec defines the DataVolume type specification
`status`	`object`	DataVolumeStatus contains the current status of the DataVolume

.spec

Description

DataVolumeSpec defines the DataVolume type specification

Type

`object`

Property	Type	Description
`checkpoints`	`array`	Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.
`contentType`	`string`	DataVolumeContentType options: "kubevirt", "archive"
`finalCheckpoint`	`boolean`	FinalCheckpoint indicates whether the current DataVolumeCheckpoint is the final checkpoint.
`preallocation`	`boolean`	Preallocation controls whether storage for DataVolumes should be allocated in advance.
`priorityClassName`	`string`	PriorityClassName for Importer, Cloner and Uploader pod
`pvc`	`object`	PVC is the PVC specification
`source`	`object`	Source is the src of the data for the requested DataVolume
`sourceRef`	`object`	SourceRef is an indirect reference to the source of data for the requested DataVolume

Property	Type	Description
`storage`	`object`	Storage is the requested storage specification

.spec.checkpoints

Description

Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Type

`array`

.spec.checkpoints[]

Description

DataVolumeCheckpoint defines a stage in a warm migration.

Type

`object`

Required

`current` `previous`

Property	Type	Description
`current`	`string`	Current is the identifier of the snapshot created for this checkpoint.
`previous`	`string`	Previous is the identifier of the snapshot from the previous checkpoint.

.spec.pvc

Description

PVC is the PVC specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which populate the volume with data, if a non-empty v is desired. This may be any object from a non-e

Property	Type	Description
		API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any names (Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.
`resources`	`object`	resources represents the minimum resources th volume should have. If

Property	Type	Description
		RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume attributes defined in the corresponding VolumeAttributesClass. This has a different pur than storageClassName, it can be changed afte claim is created. An empty string value means t VolumeAttributesClass will be applied to the cla it's not allowed to reset this field to empty string is set. If unspecified and the PersistentVolumeC unbound, the default VolumeAttributesClass wil by the persistentvolume controller if it exists. If t resource referred to by volumeAttributesClass d not exist, this PersistentVolumeClaim will be se Pending state, as reflected by the modifyVolum

Property	Type	Description
		field, until such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ (Alpha) Using this field req the VolumeAttributesClass feature gate to be er
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.spec.pvc.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.pvc.accessModes[]

Type

`string`

.spec.pvc.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.pvc.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or

dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

Property	Type	Description
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.pvc.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to

Property	Type	Description
		an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.pvc.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.pvc.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.pvc.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.pvc.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.pvc.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.pvc.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.pvc.selector.matchExpressions[].values[]

Type

`string`

.spec.pvc.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.source

Description

Source is the src of the data for the requested DataVolume

Type

`object`

Property	Type	Description
`blank`	`object`	DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC
`gcs`	`object`	DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source
`http`	`object`	DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs
`imageio`	`object`	DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source

Property	Type	Description
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`registry`	`object`	DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source
`s3`	`object`	DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot
`upload`	`object`	DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source
`vddk`	`object`	DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

.spec.source.blank

Description

DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC

Type

`object`

.spec.source.gcs

Description

DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source

Type

`object`

Required

`url`

Property	Type	Description
`secretRef`	`string`	SecretRef provides the secret reference needed to access the GCS source
`url`	`string`	URL is the url of the GCS source

.spec.source.http

Description

DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`extraHeaders`	`array`	ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests
`secretExtraHeaders`	`array`	SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information
`secretRef`	`string`	SecretRef A Secret reference, the secret should contain accessKeyId (user name) base64 encoded, and secretKey (password) also base64 encoded
`url`	`string`	URL is the URL of the http(s) endpoint

.spec.source.http.extraHeaders

Description

ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests

Type

`array`

.spec.source.http.extraHeaders[]

Type

`string`

.spec.source.http.secretExtraHeaders

Description

SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information

Type

`array`

.spec.source.http.secretExtraHeaders[]

Type

`string`

.spec.source.imageio

Description

DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source

Type

`object`

Required

`diskId` `url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the CA cert

Property	Type	Description
`diskId`	`string`	DiskID provides id of a disk to be imported
`insecureSkipVerify`	`boolean`	InsecureSkipVerify is a flag to skip certificate verification
`secretRef`	`string`	SecretRef provides the secret reference needed to access the ovirt-engine
`url`	`string`	URL is the URL of the ovirt-engine

.spec.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC

Property	Type	Description
`namespace`	`string`	The namespace of the source PVC

.spec.source.registry

Description

DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source

Type

`object`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the Registry certs
`imageStream`	`string`	ImageStream is the name of image stream for import
`platform`	`object`	Platform describes the minimum runtime requirements of the image
`pullMethod`	`string`	PullMethod can be either "pod" (default import), or "node" (node docker cache based import)
`secretRef`	`string`	SecretRef provides the secret reference needed to access the Registry source

Property	Type	Description
`url`	`string`	URL is the url of the registry source (starting with the scheme: docker, oci-archive)

.spec.source.registry.platform

Description

Platform describes the minimum runtime requirements of the image

Type

`object`

Property	Type	Description
`architecture`	`string`	Architecture specifies the image target CPU architecture

.spec.source.s3

Description

DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64

Property	Type	Description
		encoded pem certificate
`secretRef`	`string`	SecretRef provides the secret reference needed to access the S3 source
`url`	`string`	URL is the url of the S3 source

.spec.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot
`namespace`	`string`	The namespace of the source VolumeSnapshot

.spec.source.upload

Description

DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source

Type

`object`

.spec.source.vddk

Description

DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

Type

`object`

Property	Type	Description
`backingFile`	`string`	BackingFile is the path to the virtual hard disk to migrate from vCenter/ESXi
`extraArgs`	`string`	ExtraArgs is a reference to a ConfigMap containing extra arguments to pass directly to the VDDK library
`initImageURL`	`string`	InitImageURL is an optional URL to an image containing an extracted VDDK library, overrides v2v-vmware config map
`secretRef`	`string`	SecretRef provides a reference to a secret containing the username and password needed to access the vCenter or ESXi host

Property	Type	Description
`thumbprint`	`string`	Thumbprint is the certificate thumbprint of the vCenter or ESXi host
`url`	`string`	URL is the URL of the vCenter or ESXi host with the VM to migrate
`uuid`	`string`	UUID is the UUID of the virtual machine that the backing file is attached to in vCenter/ESXi

.spec.sourceRef

Description

SourceRef is an indirect reference to the source of data for the requested DataVolume

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	The kind of the source reference, currently only "DataSource" is supported
`name`	`string`	The name of the source reference

Property	Type	Description
`namespace`	`string`	The namespace of the source reference, defaults to the DataVolume namespace

.spec.storage

Description

Storage is the requested storage specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`dataSource`	`object`	This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled,

Property	Type	Description
		this field will always have the same contents as the DataSourceRef field.
`dataSourceRef`	`object`	Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: • While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.
`resources`	`object`	Resources represents the minimum resources the volume should have. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗
`selector`	`object`	A label query over volumes to consider for binding.
`storageClassName`	`string`	Name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.
`volumeName`	`string`	VolumeName is the binding reference to the PersistentVolume backing this claim.

.spec.storage.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.storage.accessModes[]

Type

`string`

.spec.storage.dataSource

Description

This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.storage.dataSourceRef

Description

Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: * While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.storage.resources

Description

Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it

Property	Type	Description
		defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.storage.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.storage.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.storage.selector

Description

A label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.storage.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.storage.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.storage.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.storage.selector.matchExpressions[].values[]

Type

`string`

.spec.storage.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status

Description

DataVolumeStatus contains the current status of the DataVolume

Type

`object`

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the underlying PVC used by the DataVolume.
`conditions`	`array`	
`phase`	`string`	Phase is the current phase of the data volume
`progress`	`string`	DataVolumeProgress is the current progress of the DataVolume transfer operation. Value between 0 and 100 inclusive, N/A if not available
`restartCount`	`integer`	RestartCount is the number of times the pod populating the DataVolume has restarted

.status.conditions

Type

`array`

.status.conditions[]

Description

DataVolumeCondition represents the state of a data volume condition.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastHeartbeatTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	DataVolumeConditionType is the string representation of known condition types

API Endpoints

The following API endpoints are available:

- `/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes`
  - `DELETE`: delete collection of DataVolume
  - `GET`: list objects of kind DataVolume
  - `POST`: create a new DataVolume
- `/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes/{name}`
  - `DELETE`: delete the specified DataVolume
  - `GET`: read the specified DataVolume
  - `PATCH`: partially update the specified DataVolume
  - `PUT`: replace the specified DataVolume
- `/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes/{name}/status`
  - `GET`: read status of the specified DataVolume
  - `PATCH`: partially update status of the specified DataVolume
  - `PUT`: replace status of the specified DataVolume

/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes

HTTP method

`DELETE`

Description

delete collection of DataVolume

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DataVolume

HTTP responses

HTTP code	Response body
200 - OK	`DataVolumeList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataVolume` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
201 - Created	`DataVolume` schema
202 - Accepted	`DataVolume` schema
401 - Unauthorized	Empty

/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes/{name}

HTTP method

`DELETE`

Description

delete the specified DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DataVolume

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataVolume` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema

HTTP code	Response body
201 - Created	`DataVolume` schema
401 - Unauthorized	Empty

/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datavolumes/{name}/status

HTTP method

`GET`

Description

read status of the specified DataVolume

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DataVolume

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataVolume` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataVolume` schema
201 - Created	`DataVolume` schema
401 - Unauthorized	Empty

DataSource [datasources.cdi.kubevirt.io/v1beta1]

Description

DataSource references an import/clone source for a DataVolume

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataSourceSpec defines specification for DataSource
`status`	`object`	DataSourceStatus provides the most recently observed status of the DataSource

.spec

Description

DataSourceSpec defines specification for DataSource

Type

`object`

Required

`source`

Property	Type	Description
`source`	`object`	Source is the source of the data referenced by the DataSource

.spec.source

Description

Source is the source of the data referenced by the DataSource

Type

`object`

Property	Type	Description
`dataSource`	`object`	DataSourceRefSourceDataSource serves as a reference to another DataSource Can be resolved into a DataVolumeSourcePVC or a DataVolumeSourceSnapshot The maximum depth of a reference chain may not exceed 1.
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

.spec.source.dataSource

Description

DataSourceRefSourceDataSource serves as a reference to another DataSource Can be resolved into a DataVolumeSourcePVC or a DataVolumeSourceSnapshot The maximum depth of a reference chain may not exceed 1.

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source DataSource
`namespace`	`string`	The namespace of the source DataSource

.spec.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC

Property	Type	Description
`namespace`	`string`	The namespace of the source PVC

.spec.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot
`namespace`	`string`	The namespace of the source VolumeSnapshot

.status

Description

DataSourceStatus provides the most recently observed status of the DataSource

Type

`object`

Property	Type	Description
`conditions`	`array`	
`source`	`object`	Source is the current source of the data referenced by the DataSource

.status.conditions

Type

`array`

.status.conditions[]

Description

DataSourceCondition represents the state of a data source condition

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastHeartbeatTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	

Property	Type	Description
`type`	`string`	DataSourceConditionType is the string representation of known condition types

.status.source

Description

Source is the current source of the data referenced by the DataSource

Type

`object`

Property	Type	Description
`dataSource`	`object`	DataSourceRefSourceDataSource serves as a reference to another DataSource Can be resolved into a DataVolumeSourcePVC or a DataVolumeSourceSnapshot The maximum depth of a reference chain may not exceed 1.
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

.status.source.dataSource

Description

DataSourceRefSourceDataSource serves as a reference to another DataSource Can be resolved into a DataVolumeSourcePVC or a DataVolumeSourceSnapshot The maximum depth of a reference chain may not exceed 1.

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source DataSource
`namespace`	`string`	The namespace of the source DataSource

.status.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC

Property	Type	Description
`namespace`	`string`	The namespace of the source PVC

.status.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot
`namespace`	`string`	The namespace of the source VolumeSnapshot

API Endpoints

The following API endpoints are available:

- `/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datasources`
  - `DELETE`: delete collection of DataSource
  - `GET`: list objects of kind DataSource
  - `POST`: create a new DataSource

- `/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datasources/{name}`
  - `DELETE`: delete the specified DataSource
  - `GET`: read the specified DataSource
  - `PATCH`: partially update the specified DataSource
  - `PUT`: replace the specified DataSource

/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datasources

HTTP method

`DELETE`

Description

delete collection of DataSource

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind DataSource

HTTP responses

HTTP code	Response body
200 - OK	`DataSourceList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DataSource

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataSource` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataSource` schema
201 - Created	`DataSource` schema
202 - Accepted	`DataSource` schema
401 - Unauthorized	Empty

/apis/cdi.kubevirt.io/v1beta1/namespaces/{namespace}/datasources/{name}

HTTP method

`DELETE`

Description

delete the specified DataSource

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified DataSource

HTTP responses

HTTP code	Response body
200 - OK	`DataSource` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DataSource

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DataSource` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DataSource

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataSource` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataSource` schema
201 - Created	`DataSource` schema
401 - Unauthorized	Empty

MigrationPolicy [migrationpolicies.migrations.kubevirt.io/v1alpha1]

Description

MigrationPolicy holds migration policy (i.e. configurations) to apply to a VM or group of VMs

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	
`status`	`object`	

.spec

Type

`object`

Required

`selectors`

Property	Type	Description
`allowAutoConverge`	`boolean`	
`allowPostCopy`	`boolean`	
`allowWorkloadDisruption`	`boolean`	
`bandwidthPerMigration`	`integer|string`	

Property	Type	Description
`completionTimeoutPerGiB`	`integer`	
`selectors`	`object`	

.spec.selectors

Type

`object`

Property	Type	Description
`namespaceSelector`	`object`	
`virtualMachineInstanceSelector`	`object`	

.spec.selectors.namespaceSelector

Type

`object`

.spec.selectors.virtualMachineInstanceSelector

Type

`object`

.status

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies`
  - `DELETE`: delete collection of MigrationPolicy
  - `GET`: list objects of kind MigrationPolicy
  - `POST`: create a new MigrationPolicy
- `/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies/{name}`
  - `DELETE`: delete the specified MigrationPolicy
  - `GET`: read the specified MigrationPolicy
  - `PATCH`: partially update the specified MigrationPolicy
  - `PUT`: replace the specified MigrationPolicy
- `/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies/{name}/status`
  - `GET`: read status of the specified MigrationPolicy
  - `PATCH`: partially update status of the specified MigrationPolicy
  - `PUT`: replace status of the specified MigrationPolicy

/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies

HTTP method

`DELETE`

Description

delete collection of MigrationPolicy

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind MigrationPolicy

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicyList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MigrationPolicy` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
201 - Created	`MigrationPolicy` schema
202 - Accepted	`MigrationPolicy` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies/{name}

HTTP method

`DELETE`

Description

delete the specified MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified MigrationPolicy

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MigrationPolicy` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
201 - Created	`MigrationPolicy` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/migrationpolicies/{name}/status

HTTP method

`GET`

Description

read status of the specified MigrationPolicy

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

Parameter	Type	Description
		processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified MigrationPolicy

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`MigrationPolicy` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`MigrationPolicy` schema
201 - Created	`MigrationPolicy` schema

HTTP code	Response body
401 - Unauthorized	Empty

VirtualMachineStorageMigrationPlan [virtualmachinestoragemigrationplans.migrations.kubevirt.io/v1alpha1]

Description

VirtualMachineStorageMigrationPlan is the Schema for the virtualmachinestoragemigrationplans API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineStorageMigrationPlanSpec defines the desired state of VirtualMachineStorageMigrationPlan
`status`	`object`	VirtualMachineStorageMigrationPlanStatus defines the observed state of VirtualMachineStorageMigrationPlan

.spec

Description

VirtualMachineStorageMigrationPlanSpec defines the desired state of VirtualMachineStorageMigrationPlan

Type

`object`

Required

`virtualMachines`

Property	Type	Description
`retentionPolicy`	`string`	RetentionPolicy indicates whether to keep or delete the source DataVolume/PVC after each VM migration completes. When "keepSource" (default), the source is preserved. When "deleteSource", the source DataVolume is deleted if it exists, otherwise the source PVC is deleted.
`virtualMachines`	`array`	The virtual machines to migrate.

.spec.virtualMachines

Description

The virtual machines to migrate.

Type

`array`

.spec.virtualMachines[]

Description

VirtualMachineStorageMigrationPlanVirtualMachine defines the VirtualMachine to migrate and the PVCs to migrate.

Type

`object`

Required

`name` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.spec.virtualMachines[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.spec.virtualMachines[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.

Property	Type	Description
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.spec.virtualMachines[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage

Property	Type	Description
		class storage profile. If empty, it will be set to filesystem.

.spec.virtualMachines[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.spec.virtualMachines[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

.status

Description

VirtualMachineStorageMigrationPlanStatus defines the observed state of VirtualMachineStorageMigrationPlan

Type

`object`

Property	Type	Description
`completedMigrations`	`array`	The migrations that have been completed.
`completedOutOf`	`string`	The number of virtual machines that have been completed out of the total number of virtual machines.
`conditions`	`array`	
`failedMigrations`	`array`	The migrations that have failed.
`inProgressMigrations`	`array`	InProgress migrations are migrations that are in progress.
`invalidMigrations`	`array`	Invalid migrations are migrations that are invalid and cannot be started.
`readyMigrations`	`array`	Ready migrations are migrations that are ready to be started.
`suffix`	`string`	The suffix to automatically append to the source PVC name. If the target name is not provided. This will replace the suffix "-new" or "-mig-xxxx" if present on the source PVC name.

.status.completedMigrations

Description

The migrations that have been completed.

Type

`array`

.status.completedMigrations[]

Description

VirtualMachineStorageMigrationPlanStatusVirtualMachine defines the status of the VirtualMachine to migrate.

Type

`object`

Required

`name` `sourcePVCs` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`sourcePVCs`	`array`	A list of source PVCs currently used by the VirtualMachine.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.status.completedMigrations[].sourcePVCs

Description

A list of source PVCs currently used by the VirtualMachine.

Type

`array`

.status.completedMigrations[].sourcePVCs[]

Description

VirtualMachineStorageMigrationPlanSourcePVC defines the source PVC used by the VirtualMachine.

Type

`object`

Required

`name` `namespace` `sourcePVC` `volumeName`

Property	Type	Description
`name`	`string`	The name of the source PVC.
`namespace`	`string`	The namespace of the source PVC.
`sourcePVC`	`object`	The source PVC.
`volumeName`	`string`	The name of the volume in the VirtualMachine.

.status.completedMigrations[].sourcePVCs[].sourcePVC

Description

The source PVC.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

Property	Type	Description
`status`	`object`	status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec

Description

spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie

Property	Type	Description
		source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which t populate the volume with data, if a non-empty v is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified.

Property	Type	Description
		• While dataSource only allows local objects, dataSourceRef allows objects in any names (Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume v

Property	Type	Description
		attributes defined in the corresponding VolumeAttributesClass. This has a different pur than storageClassName, it can be changed afte claim is created. An empty string or nil value inc that no VolumeAttributesClass will be applied tc claim. If the claim enters an Infeasible error stat field can be reset to its previous value (including cancel the modification. If the resource referred volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending as reflected by the modifyVolumeStatus field, u such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.accessModes[]

Type

`string`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values[]

Type

`string`

.status.completedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status

Description

status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the actual acces volume backing the PVC has. More infc https://kubernetes.io/docs/concepts/sto volumes#access-modes-1 ↗
`allocatedResourceStatuses`	`object`	allocatedResourceStatuses stores statu being resized for the given PVC. Key n standard Kubernetes label syntax. Valic either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. ClaimResourceStatus can be in any of states: - ControllerResizeInProgress: S resize controller starts resizing the volu plane. - ControllerResizeFailed: State s has failed in resize controller with a terr NodeResizePending: State set when re has finished resizing the volume but fur volume is needed on the node. - NodeResizeInProgress: State set wher resizing the volume. - NodeResizeFaile when resizing has failed in kubelet with error. Transient errors don't set NodeRe example: if expanding a PVC for more field can be one of the following states: pvc.status.allocatedResourceStatus['st "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['st

Property	Type	Description
		"ControllerResizeFailed" - pvc.status.allocatedResourceStatus['st "NodeResizePending" - pvc.status.allocatedResourceStatus['st "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['st "NodeResizeFailed" When this field is r means that no resize operation is in pro given PVC. A controller that receives PVC update w unknown resourceName or ClaimReso should ignore the update for the purpos designed. For example - a controller tha responsible for resizing capacity of the ignore PVC updates that change other associated with PVC. This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`allocatedResources`	`object`	allocatedResources tracks the resource a PVC including its capacity. Key name standard Kubernetes label syntax. Valid either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. Capacity reported here may be larger th capacity when a volume expansion ope requested. For storage quota, the large

Property	Type	Description
		allocatedResources and PVC.spec.res If allocatedResources is not set, PVC.s alone is used for quota calculation. If a expansion capacity request is lowered, allocatedResources is only lowered if t expansion operations in progress and i volume capacity is equal or lower than capacity. A controller that receives PVC update v unknown resourceName should ignore the purpose it was designed. For exam controller that only is responsible for re of the volume, should ignore PVC upda change other valid resources associate This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`capacity`	`object`	capacity represents the actual resource underlying volume.
`conditions`	`array`	conditions is the current Condition of pe volume claim. If underlying persistent v resized then the Condition will be set to
`currentVolumeAttributesClassName`	`string`	currentVolumeAttributesClassName is name of the VolumeAttributesClass the When unset, there is no VolumeAttribut to this PersistentVolumeClaim

Property	Type	Description
`modifyVolumeStatus`	`object`	ModifyVolumeStatus represents the sta ControllerModifyVolume operation. Wh there is no ModifyVolume operation bei
`phase`	`string`	phase represents the current phase of PersistentVolumeClaim.

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.accessModes

Description

accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.accessModes[]

Type

`string`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.allocatedResourceStatuses

Description

allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys:

- storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.allocatedResources

Description

allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity

when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.capacity

Description

capacity represents the actual resources of the underlying volume.

Type

`object`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

`array`

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	lastProbeTime is the time we probed the condition.
`lastTransitionTime`	`string`	lastTransitionTime is the time the condition transitioned
`message`	`string`	message is the human-readable message indicating de
`reason`	`string`	reason is a unique, this should be a short, machine und the reason for condition's last transition. If it reports "Re underlying persistent volume is being resized.
`status`	`string`	Status is the status of the condition. Can be True, False https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-v1/#:~:text=state%20of%20pvc-,conditions.status,-(strin
`type`	`string`	Type is the type of the condition. More info: https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-

Property	Type	Description
		v1/#:~:text=set%20to%20%27ResizeStarted%27.-,Pers contains%20details%20about ↗

.status.completedMigrations[].sourcePVCs[].sourcePVC.status.modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation. When this is unset, there is no ModifyVolume operation being attempted.

Type

`object`

Required

`status`

Property	Type	Description
`status`	`string`	status is the status of the ControllerModifyVolume operation. It can be in any of following states: • Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. • InProgress InProgress indicates that the volume is being modified.

Property	Type	Description
		• Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately.
`targetVolumeAttributesClassName`	`string`	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

.status.completedMigrations[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.status.completedMigrations[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.status.completedMigrations[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.

Property	Type	Description
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage class storage profile. If empty, it will be set to filesystem.

.status.completedMigrations[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.status.completedMigrations[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition Type - The condition type. Status - The condition status. Reason - The reason for the condition. Message - The human readable description of the condition. Durable - The condition is not un-staged. Items - A list of `items` associated with the condition used to replace [] in `Message`. staging - A condition has been explicitly set/updated.

Type

`object`

Required

`category` `lastTransitionTime` `status` `type`

Property	Type	Description
`category`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	

.status.failedMigrations

Description

The migrations that have failed.

Type

`array`

.status.failedMigrations[]

Description

VirtualMachineStorageMigrationPlanStatusVirtualMachine defines the status of the VirtualMachine to migrate.

Type

`object`

Required

`name` `sourcePVCs` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`sourcePVCs`	`array`	A list of source PVCs currently used by the VirtualMachine.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.status.failedMigrations[].sourcePVCs

Description

A list of source PVCs currently used by the VirtualMachine.

Type

`array`

.status.failedMigrations[].sourcePVCs[]

Description

VirtualMachineStorageMigrationPlanSourcePVC defines the source PVC used by the VirtualMachine.

Type

`object`

Required

`name` `namespace` `sourcePVC` `volumeName`

Property	Type	Description
`name`	`string`	The name of the source PVC.
`namespace`	`string`	The namespace of the source PVC.
`sourcePVC`	`object`	The source PVC.
`volumeName`	`string`	The name of the volume in the VirtualMachine.

.status.failedMigrations[].sourcePVCs[].sourcePVC

Description

The source PVC.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
`status`	`object`	status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec

Description

spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which populate the volume with data, if a non-empty v

Property	Type	Description
		is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any names (Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.

Property	Type	Description
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume attributes defined in the corresponding VolumeAttributesClass. This has a different pur than storageClassName, it can be changed afte claim is created. An empty string or nil value inc that no VolumeAttributesClass will be applied to claim. If the claim enters an Infeasible error stat field can be reset to its previous value (including cancel the modification. If the resource referred volumeAttributesClass does not exist, this

Property	Type	Description
		PersistentVolumeClaim will be set to a Pending as reflected by the modifyVolumeStatus field, u such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.accessModes[]

Type

`string`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.

Property	Type	Description
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an

implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values[]

Type

`string`

.status.failedMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status

Description

status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the actual acces volume backing the PVC has. More infc https://kubernetes.io/docs/concepts/sto volumes#access-modes-1 ↗
`allocatedResourceStatuses`	`object`	allocatedResourceStatuses stores statu being resized for the given PVC. Key n standard Kubernetes label syntax. Valic either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. ClaimResourceStatus can be in any of states: - ControllerResizeInProgress: S resize controller starts resizing the volu plane. - ControllerResizeFailed: State s has failed in resize controller with a terr NodeResizePending: State set when re has finished resizing the volume but fur volume is needed on the node. - NodeResizeInProgress: State set wher resizing the volume. - NodeResizeFaile when resizing has failed in kubelet with error. Transient errors don't set NodeRe example: if expanding a PVC for more field can be one of the following states: pvc.status.allocatedResourceStatus['st "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['st

Property	Type	Description
		"ControllerResizeFailed" - pvc.status.allocatedResourceStatus['st "NodeResizePending" - pvc.status.allocatedResourceStatus['st "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['st "NodeResizeFailed" When this field is r means that no resize operation is in pro given PVC. A controller that receives PVC update w unknown resourceName or ClaimReso should ignore the update for the purpos designed. For example - a controller tha responsible for resizing capacity of the ignore PVC updates that change other associated with PVC. This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`allocatedResources`	`object`	allocatedResources tracks the resource a PVC including its capacity. Key name standard Kubernetes label syntax. Valid either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. Capacity reported here may be larger th capacity when a volume expansion ope requested. For storage quota, the large

Property	Type	Description
		allocatedResources and PVC.spec.res If allocatedResources is not set, PVC.s alone is used for quota calculation. If a expansion capacity request is lowered, allocatedResources is only lowered if t expansion operations in progress and i volume capacity is equal or lower than capacity. A controller that receives PVC update v unknown resourceName should ignore the purpose it was designed. For exam controller that only is responsible for re of the volume, should ignore PVC upda change other valid resources associate This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`capacity`	`object`	capacity represents the actual resource underlying volume.
`conditions`	`array`	conditions is the current Condition of pe volume claim. If underlying persistent v resized then the Condition will be set to
`currentVolumeAttributesClassName`	`string`	currentVolumeAttributesClassName is name of the VolumeAttributesClass the When unset, there is no VolumeAttribut to this PersistentVolumeClaim

Property	Type	Description
`modifyVolumeStatus`	`object`	ModifyVolumeStatus represents the sta ControllerModifyVolume operation. Wh there is no ModifyVolume operation bei
`phase`	`string`	phase represents the current phase of PersistentVolumeClaim.

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.accessModes

Description

accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.accessModes[]

Type

`string`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.allocatedResourceStatuses

Description

allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys:

- storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.allocatedResources

Description

allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity

when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.capacity

Description

capacity represents the actual resources of the underlying volume.

Type

`object`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

`array`

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	lastProbeTime is the time we probed the condition.
`lastTransitionTime`	`string`	lastTransitionTime is the time the condition transitioned
`message`	`string`	message is the human-readable message indicating de
`reason`	`string`	reason is a unique, this should be a short, machine und the reason for condition's last transition. If it reports "Re underlying persistent volume is being resized.
`status`	`string`	Status is the status of the condition. Can be True, False https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-v1/#:~:text=state%20of%20pvc-,conditions.status,-(strin
`type`	`string`	Type is the type of the condition. More info: https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-

Property	Type	Description
		v1/#:~:text=set%20to%20%27ResizeStarted%27.-,Pers contains%20details%20about ↗

.status.failedMigrations[].sourcePVCs[].sourcePVC.status.modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation. When this is unset, there is no ModifyVolume operation being attempted.

Type

`object`

Required

`status`

Property	Type	Description
`status`	`string`	status is the status of the ControllerModifyVolume operation. It can be in any of following states: • Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. • InProgress InProgress indicates that the volume is being modified.

Property	Type	Description
		• Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately.
`targetVolumeAttributesClassName`	`string`	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

.status.failedMigrations[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.status.failedMigrations[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.status.failedMigrations[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.

Property	Type	Description
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage class storage profile. If empty, it will be set to filesystem.

.status.failedMigrations[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.status.failedMigrations[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

.status.inProgressMigrations

Description

InProgress migrations are migrations that are in progress.

Type

`array`

.status.inProgressMigrations[]

Description

VirtualMachineStorageMigrationPlanStatusVirtualMachine defines the status of the VirtualMachine to migrate.

Type

`object`

Required

`name` `sourcePVCs` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`sourcePVCs`	`array`	A list of source PVCs currently used by the VirtualMachine.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.status.inProgressMigrations[].sourcePVCs

Description

A list of source PVCs currently used by the VirtualMachine.

Type

`array`

.status.inProgressMigrations[].sourcePVCs[]

Description

VirtualMachineStorageMigrationPlanSourcePVC defines the source PVC used by the VirtualMachine.

Type

`object`

Required

`name` `namespace` `sourcePVC` `volumeName`

Property	Type	Description
`name`	`string`	The name of the source PVC.
`namespace`	`string`	The namespace of the source PVC.
`sourcePVC`	`object`	The source PVC.
`volumeName`	`string`	The name of the volume in the VirtualMachine.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC

Description

The source PVC.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

Property	Type	Description
`status`	`object`	status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec

Description

spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie

Property	Type	Description
		source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which t populate the volume with data, if a non-empty v is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified.

Property	Type	Description
		• While dataSource only allows local objects, dataSourceRef allows objects in any names (Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume v

Property	Type	Description
		attributes defined in the corresponding VolumeAttributesClass. This has a different pur than storageClassName, it can be changed afte claim is created. An empty string or nil value inc that no VolumeAttributesClass will be applied tc claim. If the claim enters an Infeasible error stat field can be reset to its previous value (including cancel the modification. If the resource referred volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending as reflected by the modifyVolumeStatus field, u such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.accessModes[]

Type

`string`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values[]

Type

`string`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status

Description

status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the actual acces volume backing the PVC has. More infc https://kubernetes.io/docs/concepts/sto volumes#access-modes-1 ↗
`allocatedResourceStatuses`	`object`	allocatedResourceStatuses stores statu being resized for the given PVC. Key n standard Kubernetes label syntax. Valic either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. ClaimResourceStatus can be in any of states: - ControllerResizeInProgress: S resize controller starts resizing the volu plane. - ControllerResizeFailed: State s has failed in resize controller with a terr NodeResizePending: State set when re has finished resizing the volume but fur volume is needed on the node. - NodeResizeInProgress: State set wher resizing the volume. - NodeResizeFaile when resizing has failed in kubelet with error. Transient errors don't set NodeRe example: if expanding a PVC for more field can be one of the following states: pvc.status.allocatedResourceStatus['st "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['st

Property	Type	Description
		"ControllerResizeFailed" - pvc.status.allocatedResourceStatus['st "NodeResizePending" - pvc.status.allocatedResourceStatus['st "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['st "NodeResizeFailed" When this field is r means that no resize operation is in pro given PVC. A controller that receives PVC update w unknown resourceName or ClaimReso should ignore the update for the purpos designed. For example - a controller tha responsible for resizing capacity of the ignore PVC updates that change other associated with PVC. This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`allocatedResources`	`object`	allocatedResources tracks the resource a PVC including its capacity. Key name standard Kubernetes label syntax. Valid either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. Capacity reported here may be larger th capacity when a volume expansion ope requested. For storage quota, the large

Property	Type	Description
		allocatedResources and PVC.spec.res If allocatedResources is not set, PVC.s alone is used for quota calculation. If a expansion capacity request is lowered, allocatedResources is only lowered if t expansion operations in progress and i volume capacity is equal or lower than capacity. A controller that receives PVC update v unknown resourceName should ignore the purpose it was designed. For exam controller that only is responsible for re of the volume, should ignore PVC upda change other valid resources associate This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`capacity`	`object`	capacity represents the actual resource underlying volume.
`conditions`	`array`	conditions is the current Condition of pe volume claim. If underlying persistent v resized then the Condition will be set to
`currentVolumeAttributesClassName`	`string`	currentVolumeAttributesClassName is name of the VolumeAttributesClass the When unset, there is no VolumeAttribut to this PersistentVolumeClaim

Property	Type	Description
`modifyVolumeStatus`	`object`	ModifyVolumeStatus represents the sta ControllerModifyVolume operation. Wh there is no ModifyVolume operation bei
`phase`	`string`	phase represents the current phase of PersistentVolumeClaim.

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.accessModes

Description

accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.accessModes[]

Type

`string`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.allocatedResourceStatuses

Description

allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys:

- storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.allocatedResources

Description

allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity

when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.capacity

Description

capacity represents the actual resources of the underlying volume.

Type

`object`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

`array`

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	lastProbeTime is the time we probed the condition.
`lastTransitionTime`	`string`	lastTransitionTime is the time the condition transitioned
`message`	`string`	message is the human-readable message indicating de
`reason`	`string`	reason is a unique, this should be a short, machine und the reason for condition's last transition. If it reports "Re underlying persistent volume is being resized.
`status`	`string`	Status is the status of the condition. Can be True, False https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-v1/#:~:text=state%20of%20pvc-,conditions.status,-(strin
`type`	`string`	Type is the type of the condition. More info: https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-

Property	Type	Description
		v1/#:~:text=set%20to%20%27ResizeStarted%27.-,Pers contains%20details%20about ↗

.status.inProgressMigrations[].sourcePVCs[].sourcePVC.status.modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation. When this is unset, there is no ModifyVolume operation being attempted.

Type

`object`

Required

`status`

Property	Type	Description
`status`	`string`	status is the status of the ControllerModifyVolume operation. It can be in any of following states: • Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. • InProgress InProgress indicates that the volume is being modified.

Property	Type	Description
		• Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately.
`targetVolumeAttributesClassName`	`string`	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

.status.inProgressMigrations[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.status.inProgressMigrations[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.status.inProgressMigrations[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.

Property	Type	Description
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage class storage profile. If empty, it will be set to filesystem.

.status.inProgressMigrations[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.status.inProgressMigrations[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

.status.invalidMigrations

Description

Invalid migrations are migrations that are invalid and cannot be started.

Type

`array`

.status.invalidMigrations[]

Description

VirtualMachineStorageMigrationPlanStatusVirtualMachine defines the status of the VirtualMachine to migrate.

Type

`object`

Required

`name` `sourcePVCs` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`sourcePVCs`	`array`	A list of source PVCs currently used by the VirtualMachine.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.status.invalidMigrations[].sourcePVCs

Description

A list of source PVCs currently used by the VirtualMachine.

Type

`array`

.status.invalidMigrations[].sourcePVCs[]

Description

VirtualMachineStorageMigrationPlanSourcePVC defines the source PVC used by the VirtualMachine.

Type

`object`

Required

`name` `namespace` `sourcePVC` `volumeName`

Property	Type	Description
`name`	`string`	The name of the source PVC.
`namespace`	`string`	The namespace of the source PVC.
`sourcePVC`	`object`	The source PVC.
`volumeName`	`string`	The name of the volume in the VirtualMachine.

.status.invalidMigrations[].sourcePVCs[].sourcePVC

Description

The source PVC.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

Property	Type	Description
`status`	`object`	status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec

Description

spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be

Property	Type	Description
		to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which t populate the volume with data, if a non-empty v is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any names

Property	Type	Description
		(Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume v attributes defined in the corresponding VolumeAttributesClass. This has a different pur

Property	Type	Description
		than storageClassName, it can be changed afte claim is created. An empty string or nil value inc that no VolumeAttributesClass will be applied tc claim. If the claim enters an Infeasible error stat field can be reset to its previous value (includin cancel the modification. If the resource referred volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending as reflected by the modifyVolumeStatus field, u such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.accessModes[]

Type

`string`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values[]

Type

`string`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status

Description

status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the actual acces volume backing the PVC has. More infc https://kubernetes.io/docs/concepts/sto volumes#access-modes-1 ↗
`allocatedResourceStatuses`	`object`	allocatedResourceStatuses stores statu being resized for the given PVC. Key n standard Kubernetes label syntax. Valic either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. ClaimResourceStatus can be in any of states: - ControllerResizeInProgress: S resize controller starts resizing the volu plane. - ControllerResizeFailed: State s has failed in resize controller with a terr NodeResizePending: State set when re has finished resizing the volume but fur volume is needed on the node. - NodeResizeInProgress: State set wher resizing the volume. - NodeResizeFaile when resizing has failed in kubelet with error. Transient errors don't set NodeRe example: if expanding a PVC for more field can be one of the following states: pvc.status.allocatedResourceStatus['st "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['st

Property	Type	Description
		"ControllerResizeFailed" - pvc.status.allocatedResourceStatus['st "NodeResizePending" - pvc.status.allocatedResourceStatus['st "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['st "NodeResizeFailed" When this field is r means that no resize operation is in pro given PVC. A controller that receives PVC update w unknown resourceName or ClaimReso should ignore the update for the purpos designed. For example - a controller tha responsible for resizing capacity of the ignore PVC updates that change other associated with PVC. This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`allocatedResources`	`object`	allocatedResources tracks the resource a PVC including its capacity. Key name standard Kubernetes label syntax. Valid either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. Capacity reported here may be larger th capacity when a volume expansion ope requested. For storage quota, the large

Property	Type	Description
		allocatedResources and PVC.spec.res If allocatedResources is not set, PVC.s alone is used for quota calculation. If a expansion capacity request is lowered, allocatedResources is only lowered if t expansion operations in progress and i volume capacity is equal or lower than capacity. A controller that receives PVC update v unknown resourceName should ignore the purpose it was designed. For exam controller that only is responsible for re of the volume, should ignore PVC upda change other valid resources associate This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`capacity`	`object`	capacity represents the actual resource underlying volume.
`conditions`	`array`	conditions is the current Condition of pe volume claim. If underlying persistent v resized then the Condition will be set to
`currentVolumeAttributesClassName`	`string`	currentVolumeAttributesClassName is name of the VolumeAttributesClass the When unset, there is no VolumeAttribut to this PersistentVolumeClaim

Property	Type	Description
`modifyVolumeStatus`	`object`	ModifyVolumeStatus represents the sta ControllerModifyVolume operation. Wh there is no ModifyVolume operation bei
`phase`	`string`	phase represents the current phase of PersistentVolumeClaim.

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.accessModes

Description

accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.accessModes[]

Type

`string`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.allocatedResourceStatuses

Description

allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys:

- storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.allocatedResources

Description

allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity

when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.capacity

Description

capacity represents the actual resources of the underlying volume.

Type

`object`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

`array`

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	lastProbeTime is the time we probed the condition.
`lastTransitionTime`	`string`	lastTransitionTime is the time the condition transitioned
`message`	`string`	message is the human-readable message indicating de
`reason`	`string`	reason is a unique, this should be a short, machine und the reason for condition's last transition. If it reports "Re underlying persistent volume is being resized.
`status`	`string`	Status is the status of the condition. Can be True, False https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-v1/#:~:text=state%20of%20pvc-,conditions.status,-(strin
`type`	`string`	Type is the type of the condition. More info: https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-

Property	Type	Description
		v1/#:~:text=set%20to%20%27ResizeStarted%27.-,Pers contains%20details%20about ↗

.status.invalidMigrations[].sourcePVCs[].sourcePVC.status.modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation. When this is unset, there is no ModifyVolume operation being attempted.

Type

`object`

Required

`status`

Property	Type	Description
`status`	`string`	status is the status of the ControllerModifyVolume operation. It can be in any of following states: • Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. • InProgress InProgress indicates that the volume is being modified.

Property	Type	Description
		• Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately.
`targetVolumeAttributesClassName`	`string`	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

.status.invalidMigrations[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.status.invalidMigrations[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.status.invalidMigrations[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.

Property	Type	Description
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage class storage profile. If empty, it will be set to filesystem.

.status.invalidMigrations[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.status.invalidMigrations[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

.status.readyMigrations

Description

Ready migrations are migrations that are ready to be started.

Type

`array`

.status.readyMigrations[]

Description

VirtualMachineStorageMigrationPlanStatusVirtualMachine defines the status of the VirtualMachine to migrate.

Type

`object`

Required

`name` `sourcePVCs` `targetMigrationPVCs`

Property	Type	Description
`name`	`string`	The name of the virtual machine to migrate.
`sourcePVCs`	`array`	A list of source PVCs currently used by the VirtualMachine.
`targetMigrationPVCs`	`array`	A list of PVCs associated with the VirtualMachine to migrate.

.status.readyMigrations[].sourcePVCs

Description

A list of source PVCs currently used by the VirtualMachine.

Type

`array`

.status.readyMigrations[].sourcePVCs[]

Description

VirtualMachineStorageMigrationPlanSourcePVC defines the source PVC used by the VirtualMachine.

Type

`object`

Required

`name` `namespace` `sourcePVC` `volumeName`

Property	Type	Description
`name`	`string`	The name of the source PVC.
`namespace`	`string`	The namespace of the source PVC.
`sourcePVC`	`object`	The source PVC.
`volumeName`	`string`	The name of the volume in the VirtualMachine.

.status.readyMigrations[].sourcePVCs[].sourcePVC

Description

The source PVC.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata ↗
`spec`	`object`	spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

Property	Type	Description
`status`	`object`	status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec

Description

spec defines the desired characteristics of a volume requested by a pod author. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access mod volume should have. More info: https://kubernetes.io/docs/concepts/storage/per volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If provisioner or an external controller can sup the specified data source, it will create a nev volume based on the contents of the specifie source. When the AnyVolumeDataSource fe gate is enabled, dataSource contents will be

Property	Type	Description
		to dataSourceRef, and dataSourceRef conte be copied to dataSource when dataSourceRef.namespace is not specified. namespace is specified, then dataSourceRe not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which t populate the volume with data, if a non-empty v is desired. This may be any object from a non-e API group (non core object) or a PersistentVolumeClaim object. When this field i specified, volume binding will only succeed if th of the specified object matches some installed v populator or dynamic provisioner. This field will the functionality of the dataSource field and as s both fields are non-empty, they must have the s value. For backwards compatibility, when name isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to t same value automatically if one of them is empt the other is non-empty. When namespace is sp in dataSourceRef, dataSource isn't set to the sa value and must be empty. There are three impo differences between dataSource and dataSourc • While dataSource only allows two specific ty objects, dataSourceRef allows any non-core as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves a values, and generates an error if a disallowe value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any names

Property	Type	Description
		(Beta) Using this field requires the AnyVolumeDataSource feature gate to be en (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource featur to be enabled.
`resources`	`object`	resources represents the minimum resources th volume should have. If RecoverVolumeExpansionFailure feature is ena users are allowed to specify resource requireme that are lower than previous value but must still higher than capacity recorded in the status field claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consid binding.
`storageClassName`	`string`	storageClassName is the name of the StorageC required by the claim. More info: https://kubernetes.io/docs/concepts/storage/per volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to se VolumeAttributesClass used by this claim. If spe the CSI driver will create or update the volume v attributes defined in the corresponding VolumeAttributesClass. This has a different pur

Property	Type	Description
		than storageClassName, it can be changed afte claim is created. An empty string or nil value inc that no VolumeAttributesClass will be applied tc claim. If the claim enters an Infeasible error stat field can be reset to its previous value (including cancel the modification. If the resource referred volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending as reflected by the modifyVolumeStatus field, u such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/vol attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is rec by the claim. Value of Filesystem is implied whe included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.accessModes[]

Type

`string`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced

Property	Type	Description
`name`	`string`	Name is the name of resource being referenced

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchExpressions[].values[]

Type

`string`

.status.readyMigrations[].sourcePVCs[].sourcePVC.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status

Description

status represents the current information/status of a persistent volume claim. Read-only. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the actual acces volume backing the PVC has. More infc https://kubernetes.io/docs/concepts/sto volumes#access-modes-1 ↗
`allocatedResourceStatuses`	`object`	allocatedResourceStatuses stores statu being resized for the given PVC. Key n standard Kubernetes label syntax. Valic either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. ClaimResourceStatus can be in any of states: - ControllerResizeInProgress: S resize controller starts resizing the volu plane. - ControllerResizeFailed: State s has failed in resize controller with a terr NodeResizePending: State set when re has finished resizing the volume but fur volume is needed on the node. - NodeResizeInProgress: State set wher resizing the volume. - NodeResizeFaile when resizing has failed in kubelet with error. Transient errors don't set NodeRe example: if expanding a PVC for more field can be one of the following states: pvc.status.allocatedResourceStatus['st "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['st

Property	Type	Description
		"ControllerResizeFailed" - pvc.status.allocatedResourceStatus['st "NodeResizePending" - pvc.status.allocatedResourceStatus['st "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['st "NodeResizeFailed" When this field is r means that no resize operation is in pro given PVC. A controller that receives PVC update w unknown resourceName or ClaimReso should ignore the update for the purpos designed. For example - a controller tha responsible for resizing capacity of the ignore PVC updates that change other associated with PVC. This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`allocatedResources`	`object`	allocatedResources tracks the resource a PVC including its capacity. Key name standard Kubernetes label syntax. Valid either: * Un-prefixed keys: - storage - th the volume. * Custom resources must u implementation-defined prefixed names "example.com/my-custom-resource" Ap values - keys that are unprefixed or hav kubernetes.io prefix are considered res hence may not be used. Capacity reported here may be larger th capacity when a volume expansion ope requested. For storage quota, the large

Property	Type	Description
		allocatedResources and PVC.spec.res If allocatedResources is not set, PVC.s alone is used for quota calculation. If a expansion capacity request is lowered, allocatedResources is only lowered if t expansion operations in progress and i volume capacity is equal or lower than capacity. A controller that receives PVC update v unknown resourceName should ignore the purpose it was designed. For exam controller that only is responsible for re of the volume, should ignore PVC upda change other valid resources associate This is an alpha field and requires enab RecoverVolumeExpansionFailure featu
`capacity`	`object`	capacity represents the actual resource underlying volume.
`conditions`	`array`	conditions is the current Condition of pe volume claim. If underlying persistent v resized then the Condition will be set to
`currentVolumeAttributesClassName`	`string`	currentVolumeAttributesClassName is name of the VolumeAttributesClass the When unset, there is no VolumeAttribut to this PersistentVolumeClaim

Property	Type	Description
`modifyVolumeStatus`	`object`	ModifyVolumeStatus represents the sta ControllerModifyVolume operation. Wh there is no ModifyVolume operation bei
`phase`	`string`	phase represents the current phase of PersistentVolumeClaim.

.status.readyMigrations[].sourcePVCs[].sourcePVC.status .accessModes

Description

accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status .accessModes[]

Type

`string`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status .allocatedResourceStatuses

Description

allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys:

- storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status.allocatedResources

Description

allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity

when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status.capacity

Description

capacity represents the actual resources of the underlying volume.

Type

`object`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

`array`

.status.readyMigrations[].sourcePVCs[].sourcePVC.status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	lastProbeTime is the time we probed the condition.
`lastTransitionTime`	`string`	lastTransitionTime is the time the condition transitioned
`message`	`string`	message is the human-readable message indicating de
`reason`	`string`	reason is a unique, this should be a short, machine und the reason for condition's last transition. If it reports "Re underlying persistent volume is being resized.
`status`	`string`	Status is the status of the condition. Can be True, False https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-v1/#:~:text=state%20of%20pvc-,conditions.status,-(strin
`type`	`string`	Type is the type of the condition. More info: https://kubernetes.io/docs/reference/kubernetes-api/con resources/persistent-volume-claim-

Property	Type	Description
		v1/#:~:text=set%20to%20%27ResizeStarted%27.-,Pers contains%20details%20about ↗

.status.readyMigrations[].sourcePVCs[].sourcePVC.status .modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation. When this is unset, there is no ModifyVolume operation being attempted.

Type

`object`

Required

`status`

Property	Type	Description
`status`	`string`	status is the status of the ControllerModifyVolume operation. It can be in any of following states: • Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. • InProgress InProgress indicates that the volume is being modified.

Property	Type	Description
		• Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately.
`targetVolumeAttributesClassName`	`string`	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

.status.readyMigrations[].targetMigrationPVCs

Description

A list of PVCs associated with the VirtualMachine to migrate.

Type

`array`

.status.readyMigrations[].targetMigrationPVCs[]

Description

VirtualMachineStorageMigrationPlanTargetMigrationPVC defines the PVC to migrate to.

Type

`object`

Required

`destinationPVC` `volumeName`

Property	Type	Description
`destinationPVC`	`object`	The destination PVC to migrate to.
`volumeName`	`string`	The name of the volume in the VirtualMachine to migrate.

.status.readyMigrations[].targetMigrationPVCs[].destinationPVC

Description

The destination PVC to migrate to.

Type

`object`

Property	Type	Description
`accessModes`	`array`	The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.
`name`	`string`	The name of the destination PVC. If not provided, the PVC will be named after the source PVC with a "-mig-xxxx" suffix.

Property	Type	Description
`storageClassName`	`string`	The target storage class to use for the PVC. If not provided, the PVC will use the default storage class.
`volumeMode`	`string`	The volume mode to use for the PVC, if set to Auto, the volume mode will be looked up from the storage class storage profile. If empty, it will be set to filesystem.

.status.readyMigrations[].targetMigrationPVCs[].destinationPVC.accessModes

Description

The access modes to use for the PVC, if set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`array`

.status.readyMigrations[].targetMigrationPVCs[].destinationPVC.accessModes[]

Description

The access mode of the source PVC. If set to Auto, the access mode will be looked up from the storage class storage profile.

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans`
  - `DELETE`: delete collection of VirtualMachineStorageMigrationPlan
  - `GET`: list objects of kind VirtualMachineStorageMigrationPlan
  - `POST`: create a new VirtualMachineStorageMigrationPlan
- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans/{name}`
  - `DELETE`: delete the specified VirtualMachineStorageMigrationPlan
  - `GET`: read the specified VirtualMachineStorageMigrationPlan
  - `PATCH`: partially update the specified VirtualMachineStorageMigrationPlan
  - `PUT`: replace the specified VirtualMachineStorageMigrationPlan
- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans/{name}/status`
  - `GET`: read status of the specified VirtualMachineStorageMigrationPlan
  - `PATCH`: partially update status of the specified VirtualMachineStorageMigrationPlan
  - `PUT`: replace status of the specified VirtualMachineStorageMigrationPlan

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans

HTTP method

`DELETE`

Description

delete collection of VirtualMachineStorageMigrationPlan

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineStorageMigrationPlan

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlanList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigrationPlan` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
201 - Created	`VirtualMachineStorageMigrationPlan` schema
202 - Accepted	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineStorageMigrationPlan

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigrationPlan` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
201 - Created	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrationplans/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineStorageMigrationPlan

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineStorageMigrationPlan

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigrationPlan` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationPlan` schema
201 - Created	`VirtualMachineStorageMigrationPlan` schema

HTTP code	Response body
401 - Unauthorized	Empty

VirtualMachineStorageMigration [virtualmachinestoragemigrations.migrations.kubevirt.io/v1alpha1]

Description

VirtualMachineStorageMigration is the Schema for the virtualmachinestoragemigrations API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint

Property	Type	Description
		the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	MigMigrationSpec defines the desired state of MigMigration
`status`	`object`	VirtualMachineStorageMigrationStatus defines the observed state of VirtualMachineStorageMigration

.spec

Description

MigMigrationSpec defines the desired state of MigMigration

Type

`object`

Required

`virtualMachineStorageMigrationPlanRef`

Property	Type	Description
`virtualMachineStorageMigrationPlanRef`	`object`	ObjectReference contains enough information to let

Property	Type	Description
		you inspect or modify the referred object.

.spec.virtualMachineStorageMigrationPlanRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-

Property	Type	Description
		architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status

Description

VirtualMachineStorageMigrationStatus defines the observed state of VirtualMachineStorageMigration

Type

`object`

Property	Type	Description
`cancelledMigrations`	`array`	The cancelled migrations.
`completedMigrations`	`array`	The completed migrations.
`conditions`	`array`	
`errors`	`array`	The errors occurred during the migration.
`phase`	`string`	The current phase of the migration.
`runningMigrations`	`array`	The running migrations.

.status.cancelledMigrations

Description

The cancelled migrations.

Type

`array`

.status.cancelledMigrations[]

Type

`string`

.status.completedMigrations

Description

The completed migrations.

Type

`array`

.status.completedMigrations[]

Type

`string`

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition Type - The condition type. Status - The condition status. Reason - The reason for the condition. Message - The human readable description of the condition. Durable - The condition is not un-staged. Items - A list of `items` associated with the condition used to replace [] in `Message`. staging - A condition has been explicitly set/updated.

Type

`object`

Required

`category` `lastTransitionTime` `status` `type`

Property	Type	Description
`category`	`string`	
`lastTransitionTime`	`string`	

Property	Type	Description
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	

.status.errors

Description

The errors occurred during the migration.

Type

`array`

.status.errors[]

Type

`string`

.status.runningMigrations

Description

The running migrations.

Type

`array`

.status.runningMigrations[]

Description

RunningVirtualMachineMigration has the name of the VirtualMachine and the progress of the migration.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	The name of the VirtualMachine.
`progress`	`string`	The progress of the migration.

API Endpoints

The following API endpoints are available:

- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations`
  - `DELETE`: delete collection of VirtualMachineStorageMigration
  - `GET`: list objects of kind VirtualMachineStorageMigration
  - `POST`: create a new VirtualMachineStorageMigration
- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations/{name}`
  - `DELETE`: delete the specified VirtualMachineStorageMigration
  - `GET`: read the specified VirtualMachineStorageMigration
  - `PATCH`: partially update the specified VirtualMachineStorageMigration
  - `PUT`: replace the specified VirtualMachineStorageMigration

- `/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations/{name}/status`
  - `GET`: read status of the specified VirtualMachineStorageMigration
  - `PATCH`: partially update status of the specified VirtualMachineStorageMigration
  - `PUT`: replace status of the specified VirtualMachineStorageMigration

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations

HTTP method

`DELETE`

Description

delete collection of VirtualMachineStorageMigration

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineStorageMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigrationList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
201 - Created	`VirtualMachineStorageMigration` schema
202 - Accepted	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineStorageMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object,

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
201 - Created	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

/apis/migrations.kubevirt.io/v1alpha1/namespaces/{namespace}/virtualmachinestoragemigrations/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineStorageMigration

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineStorageMigration

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineStorageMigration` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineStorageMigration` schema
201 - Created	`VirtualMachineStorageMigration` schema
401 - Unauthorized	Empty

VirtualMachineExport [virtualmachineexports.export.kubevirt.io/v1beta1]

Description

VirtualMachineExport defines the operation of exporting a VM source

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VirtualMachineExportSpec is the spec for a VirtualMachineExport resource
`status`	`object`	VirtualMachineExportStatus is the status for a VirtualMachineExport resource

.spec

Description

VirtualMachineExportSpec is the spec for a VirtualMachineExport resource

Type

`object`

Required

`source`

Property	Type	Description
`source`	`object`	TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.
`tokenSecretRef`	`string`	TokenSecretRef is the name of the custom-defined secret that contains the token used by the export server pod
`ttlDuration`	`string`	ttlDuration limits the lifetime of an export If this field is set, after this duration has passed from counting from CreationTimestamp, the export is eligible to be automatically deleted. If this field is omitted, a reasonable default is applied.

.spec.source

Description

TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in

Property	Type	Description
		the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.status

Description

VirtualMachineExportStatus is the status for a VirtualMachineExport resource

Type

`object`

Property	Type	Description
`conditions`	`array`	
`links`	`object`	VirtualMachineExportLinks contains the links that point the exported VM resources
`phase`	`string`	VirtualMachineExportPhase is the current phase of the VirtualMachineExport
`serviceName`	`string`	ServiceName is the name of the service created associated with the Virtual Machine export. It will

Property	Type	Description
		be used to create the internal URLs for downloading the images
`tokenSecretRef`	`string`	TokenSecretRef is the name of the secret that contains the token used by the export server pod
`ttlExpirationTime`	`string`	The time at which the VM Export will be completely removed according to specified TTL Formula is CreationTimestamp + TTL
`virtualMachineName`	`string`	VirtualMachineName shows the name of the source virtual machine if the source is either a VirtualMachine or a VirtualMachineSnapshot. This is mainly to easily identify the source VirtualMachine in case of a VirtualMachineSnapshot

.status.conditions

Type

`array`

.status.conditions[]

Description

Condition defines conditions

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	ConditionType is the const type for Conditions

.status.links

Description

VirtualMachineExportLinks contains the links that point the exported VM resources

Type

`object`

Property	Type	Description
`external`	`object`	VirtualMachineExportLink contains a list of volumes available for export, as well as the URLs to obtain these volumes

Property	Type	Description
`internal`	`object`	VirtualMachineExportLink contains a list of volumes available for export, as well as the URLs to obtain these volumes

.status.links.external

Description

VirtualMachineExportLink contains a list of volumes available for export, as well as the URLs to obtain these volumes

Type

`object`

Required

`cert`

Property	Type	Description
`cert`	`string`	Cert is the public CA certificate base64 encoded
`manifests`	`array`	Manifests is a list of available manifests for the export
`volumes`	`array`	Volumes is a list of available volumes to export

.status.links.external.manifests

Description

Manifests is a list of available manifests for the export

Type

`array`

.status.links.external.manifests[]

Description

VirtualMachineExportManifest contains the type and URL of the exported manifest

Type

`object`

Required

`type` `url`

Property	Type	Description
`type`	`string`	Type is the type of manifest returned
`url`	`string`	Url is the url of the endpoint that returns the manifest

.status.links.external.volumes

Description

Volumes is a list of available volumes to export

Type

`array`

.status.links.external.volumes[]

Description

VirtualMachineExportVolume contains the name and available formats for the exported volume

Type

`object`

Required

`name`

Property	Type	Description
`formats`	`array`	
`name`	`string`	Name is the name of the exported volume

.status.links.external.volumes[].formats

Type

`array`

.status.links.external.volumes[].formats[]

Description

VirtualMachineExportVolumeFormat contains the format type and URL to get the volume in that format

Type

`object`

Required

`format` `url`

Property	Type	Description
`format`	`string`	Format is the format of the image at the specified URL

Property	Type	Description
`url`	`string`	Url is the url that contains the volume in the format specified

.status.links.internal

Description

VirtualMachineExportLink contains a list of volumes available for export, as well as the URLs to obtain these volumes

Type

`object`

Required

`cert`

Property	Type	Description
`cert`	`string`	Cert is the public CA certificate base64 encoded
`manifests`	`array`	Manifests is a list of available manifests for the export
`volumes`	`array`	Volumes is a list of available volumes to export

.status.links.internal.manifests

Description

Manifests is a list of available manifests for the export

Type

`array`

.status.links.internal.manifests[]

Description

VirtualMachineExportManifest contains the type and URL of the exported manifest

Type

`object`

Required

`type` `url`

Property	Type	Description
`type`	`string`	Type is the type of manifest returned
`url`	`string`	Url is the url of the endpoint that returns the manifest

.status.links.internal.volumes

Description

Volumes is a list of available volumes to export

Type

`array`

.status.links.internal.volumes[]

Description

VirtualMachineExportVolume contains the name and available formats for the exported volume

Type

`object`

Required

`name`

Property	Type	Description
`formats`	`array`	
`name`	`string`	Name is the name of the exported volume

.status.links.internal.volumes[].formats

Type

`array`

.status.links.internal.volumes[].formats[]

Description

VirtualMachineExportVolumeFormat contains the format type and URL to get the volume in that format

Type

`object`

Required

`format` `url`

Property	Type	Description
`format`	`string`	Format is the format of the image at the specified URL

Property	Type	Description
`url`	`string`	Url is the url that contains the volume in the format specified

API Endpoints

The following API endpoints are available:

- `/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports`
  - `DELETE`: delete collection of VirtualMachineExport
  - `GET`: list objects of kind VirtualMachineExport
  - `POST`: create a new VirtualMachineExport
- `/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports/{name}`
  - `DELETE`: delete the specified VirtualMachineExport
  - `GET`: read the specified VirtualMachineExport
  - `PATCH`: partially update the specified VirtualMachineExport
  - `PUT`: replace the specified VirtualMachineExport
- `/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports/{name}/status`
  - `GET`: read status of the specified VirtualMachineExport
  - `PATCH`: partially update status of the specified VirtualMachineExport
  - `PUT`: replace status of the specified VirtualMachineExport

/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports

HTTP method

`DELETE`

Description

delete collection of VirtualMachineExport

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineExport

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExportList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineExport` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
201 - Created	`VirtualMachineExport` schema

HTTP code	Response body
202 - Accepted	`VirtualMachineExport` schema
401 - Unauthorized	Empty

/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineExport

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineExport` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
201 - Created	`VirtualMachineExport` schema
401 - Unauthorized	Empty

/apis/export.kubevirt.io/v1beta1/namespaces/{namespace}/virtualmachineexports/{name}/status

HTTP method

`GET`

Description

read status of the specified VirtualMachineExport

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VirtualMachineExport

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineExport` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineExport` schema
201 - Created	`VirtualMachineExport` schema
401 - Unauthorized	Empty

HyperConverged [hyperconvergeds.hco.kubevirt.io/v1beta1]

Description

HyperConverged is the Schema for the hyperconvergeds API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.

Property	Type	Description
`spec`	`object`	HyperConvergedSpec defines the desired state of HyperConverged
`status`	`object`	HyperConvergedStatus defines the observed state of HyperConverged

.spec

Description

HyperConvergedSpec defines the desired state of HyperConverged

Type

`object`

Property	Type	Description
`CommonInstancetypesDeployment`	`object`	CommonInstancetypesDeployment holds the configuration of common-instancetypes deployment within KubeVirt.
`applicationAwareConfig`	`object`	ApplicationAwareConfig set the AAQ configurations
`certConfig`	`object`	certConfig holds the rotation policy for internal, self-signed certificates
`commonBootImageNamespace`	`string`	CommonBootImageNamespace override the default namespace of the common boot images, in order to hide them. If not set, HCO won't set any namespace, letting SSP to use the default. If set, use the namespace to create the DataImportCronTemplates and the common image streams, with this namespace. This field is not set by default.

Property	Type	Description
`commonTemplatesNamespace`	`string`	CommonTemplatesNamespace defines namespace in which common templates will be deployed. It overrides the default openshift namespace.
`dataImportCronTemplates`	`array`	DataImportCronTemplates holds list of data import cron templates (golden images)
`defaultCPUModel`	`string`	DefaultCPUModel defines a cluster default for CPU model: default CPU model is set when VMI doesn't have any CPU model. When VMI has CPU model set, then VMI's CPU model is preferred. When default CPU model is not set and VMI's CPU model is not set too, host-model will be set. Default CPU model can be changed when kubevirt is running.
`defaultRuntimeClass`	`string`	DefaultRuntimeClass defines a cluster default for the RuntimeClass to be used for VMIs pods if not set there. Default RuntimeClass can be changed when kubevirt is running, existing VMIs are not impacted till the next restart/live-migration when they are eventually going to consume the new default RuntimeClass.
`deployVmConsoleProxy`	`boolean`	deploy VM console proxy resources in SSP operator
`enableApplicationAwareQuota`	`boolean`	EnableApplicationAwareQuota if true, enables the Application Aware Quota feature
`enableCommonBootImageImport`	`boolean`	Opt-in to automatic delivery/updates of the common data import cron templates. There are two sources for the data import cron templates: hard coded list of common templates, and custom (user defined) templates that can be added to the dataImportCronTemplates field. This field only controls the common templates. It is possible to use

Property	Type	Description
		custom templates by adding them to the dataImportCronTemplates field.
`evictionStrategy`	`string`	EvictionStrategy defines at the cluster level if the VirtualMachineInstance should be migrated instead of shut-off in case of a node drain. If the VirtualMachineInstance specific field is set it overrides the cluster level one. Allowed values: • `None` no eviction strategy at cluster level. • `LiveMigrate` migrate the VM on eviction; a not live migratable VM with no specific strategy will block the drain of the node util manually evicted. • `LiveMigrateIfPossible` migrate the VM on eviction if live migration is possible, otherwise directly evict. • `External` block the drain, track eviction and notify an external controller. Defaults to LiveMigrate with multiple worker nodes, None on single worker clusters.
`featureGates`	`object`	featureGates is a map of feature gate flags. Setting a flag to `true` will enable the feature. Setting `false` or removing the feature gate, disables the feature.
`filesystemOverhead`	`object`	FilesystemOverhead describes the space reserved for overhead when using Filesystem volumes. A value is between 0 and 1, if not defined it is 0.055 (5.5 percent overhead)
`higherWorkloadDensity`	`object`	HigherWorkloadDensity holds configuration aimed to increase virtual machine density
`infra`	`object`	infra HyperConvergedConfig influences the pod configuration (currently only placement) for all the infra components needed on the virtualization enabled cluster

Property	Type	Description
		but not necessarily directly on each node running VMs/VMIs.
`instancetypeConfig`	`object`	InstancetypeConfig holds the configuration of instance type related functionality within KubeVirt.
`ksmConfiguration`	`object`	KSMConfiguration holds the information regarding the enabling the KSM in the nodes (if available).
`kubeMacPoolConfiguration`	`object`	KubeMacPoolConfiguration holds kubemacpool MAC address range configuration.
`kubeSecondaryDNSNameServerIP`	`string`	KubeSecondaryDNSNameServerIP defines name server IP used by KubeSecondaryDNS
`liveMigrationConfig`	`object`	Live migration limits and timeouts are applied so that migration processes do not overwhelm the cluster.
`liveUpdateConfiguration`	`object`	LiveUpdateConfiguration holds the cluster configuration for live update of virtual machines - max cpu sockets, max guest memory and max hotplug ratio. This setting can affect VM CPU and memory settings.
`localStorageClassName`	`string`	Deprecated: LocalStorageClassName the name of the local storage class.
`logVerbosityConfig`	`object`	LogVerbosityConfig configures the verbosity level of Kubevirt's different components. The higher the value - the higher the log verbosity.

Property	Type	Description
`mediatedDevicesConfiguration`	`object`	MediatedDevicesConfiguration holds information about MDEV types to be defined on nodes, if available
`networkBinding`	`object`	NetworkBinding defines the network binding plugins. Those bindings can be used when defining virtual machine interfaces.
`obsoleteCPUs`	`object`	ObsoleteCPUs allows avoiding scheduling of VMs for obsolete CPU models
`permittedHostDevices`	`object`	PermittedHostDevices holds information about devices allowed for passthrough
`resourceRequirements`	`object`	ResourceRequirements describes the resource requirements for the operand workloads.
`scratchSpaceStorageClass`	`string`	Override the storage class used for scratch space during transfer operations. The scratch space storage class is determined in the following order: value of scratchSpaceStorageClass, if that doesn't exist, use the default storage class, if there is no default storage class, use the storage class of the DataVolume, if no storage class specified, use no storage class for scratch space
`storageImport`	`object`	StorageImport contains configuration for importing containerized data

Property	Type	Description
`tektonPipelinesNamespace`	`string`	TektonPipelinesNamespace defines namespace in which example pipelines will be deployed. If unset, then the default value is the operator namespace. Deprecated: This field is ignored.
`tektonTasksNamespace`	`string`	TektonTasksNamespace defines namespace in which tekton tasks will be deployed. If unset, then the default value is the operator namespace. Deprecated: This field is ignored.
`tlsSecurityProfile`	`object`	TLSSecurityProfile specifies the settings for TLS connections to be propagated to all kubevirt-hyperconverged components. If unset, the hyperconverged cluster operator will consume the value set on the APIServer CR on OCP/OKD or Intermediate if on vanilla k8s. Note that only Old, Intermediate and Custom profiles are currently supported, and the maximum available MinTLSVersions is VersionTLS12.
`tuningPolicy`	`string`	TuningPolicy allows to configure the mode in which the RateLimits of kubevirt are set. If TuningPolicy is not present the default kubevirt values are used. It can be set to `annotation` for fine-tuning the kubevirt queryPerSeconds (qps) and burst values. Qps and burst values are taken from the annotation hco.kubevirt.io/tuningPolicy
`uninstallStrategy`	`string`	UninstallStrategy defines how to proceed on uninstall when workloads (VirtualMachines, DataVolumes) still exist. BlockUninstallIfWorkloadsExist will prevent the CR from being removed when workloads still exist. BlockUninstallIfWorkloadsExist is the safest choice to protect your workloads from accidental data loss, so it's strongly advised. RemoveWorkloads will cause all the workloads to be cascading deleted on uninstallation.

Property	Type	Description
		WARNING: please notice that RemoveWorkloads will cause your workloads to be deleted as soon as this CR will be, even accidentally, deleted. Please correctly consider the implications of this option before setting it. BlockUninstallIfWorkloadsExist is the default behaviour.
`vddkInitImage`	`string`	VDDK Init Image eventually used to import VMs from external providers Deprecated: please use the Migration Toolkit for Virtualization
`virtualMachineOptions`	`object`	VirtualMachineOptions holds the cluster level information regarding the virtual machine.
`vmStateStorageClass`	`string`	VMStateStorageClass is the name of the storage class to use for the PVCs created to preserve VM state, like TPM.
`workloadUpdateStrategy`	`object`	WorkloadUpdateStrategy defines at the cluster level how to handle automated workload updates
`workloads`	`object`	workloads HyperConvergedConfig influences the pod configuration (currently only placement) of components which need to be running on a node where virtualization workloads should be able to run. Changes to Workloads HyperConvergedConfig can be applied only without existing workload.

.spec.CommonInstancetypesDeployment

Description

CommonInstancetypesDeployment holds the configuration of common-instancetypes deployment within KubeVirt.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled controls the deployment of common-instancetypes resources, defaults to True.

.spec.applicationAwareConfig

Description

ApplicationAwareConfig set the AAQ configurations

Type

`object`

Property	Type	Description
`allowApplicationAwareClusterResourceQuota`	`boolean`	AllowApplicationAwareClusterResourceQuota if set to true, allows creation and management of ClusterAppsResourceQuota
`namespaceSelector`	`object`	NamespaceSelector determines in which namespaces scheduling gate will be added to pods..
`vmiCalcConfigName`	`string`	VmiCalcConfigName determine how resource allocation will be done with ApplicationsResourceQuota. allowed values are: VmiPodUsage, VirtualResources, DedicatedVirtualResources or IgnoreVmiCalculator

.spec.applicationAwareConfig.namespaceSelector

Description

NamespaceSelector determines in which namespaces scheduling gate will be added to pods..

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.applicationAwareConfig.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.applicationAwareConfig.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.applicationAwareConfig.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.applicationAwareConfig.namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.applicationAwareConfig.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.certConfig

Description

certConfig holds the rotation policy for internal, self-signed certificates

Type

`object`

Property	Type	Description
`ca`	`object`	CA configuration - CA certs are kept in the CA bundle as long as they are valid
`server`	`object`	Server configuration - Certs are rotated and discarded

.spec.certConfig.ca

Description

CA configuration - CA certs are kept in the CA bundle as long as they are valid

Type

`object`

Property	Type	Description
`duration`	`string`	The requested 'duration' (i.e. lifetime) of the Certificate. This should comply with golang's ParseDuration format (https://golang.org/pkg/time/#ParseDuration)
`renewBefore`	`string`	The amount of time before the currently issued certificate's `notAfter` time that we will begin to attempt to renew the certificate. This should comply with golang's ParseDuration format (https://golang.org/pkg/time/#ParseDuration)

.spec.certConfig.server

Description

Server configuration - Certs are rotated and discarded

Type

`object`

Property	Type	Description
`duration`	`string`	The requested 'duration' (i.e. lifetime) of the Certificate. This should comply with golang's ParseDuration format (https://golang.org/pkg/time/#ParseDuration ↗)
`renewBefore`	`string`	The amount of time before the currently issued certificate's `notAfter` time that we will begin to attempt to renew the certificate. This should comply with golang's ParseDuration format (https://golang.org/pkg/time/#ParseDuration ↗)

.spec.dataImportCronTemplates

Description

DataImportCronTemplates holds list of data import cron templates (golden images)

Type

`array`

.spec.dataImportCronTemplates[]

Description

DataImportCronTemplate defines the template type for DataImportCrons. It requires metadata.name to be specified while leaving namespace as optional.

Type

`object`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataImportCronSpec defines specification for DataImportCron

.spec.dataImportCronTemplates[].spec

Description

DataImportCronSpec defines specification for DataImportCron

Type

`object`

Required

`managedDataSource` `schedule` `template`

Property	Type	Description
`garbageCollect`	`string`	GarbageCollect specifies whether old PVCs should be cleaned up after a new PVC is imported. Options are currently "Outdated" and "Never", defaults to "Outdated".
`importsToKeep`	`integer`	Number of import PVCs to keep when garbage collecting. Default is 3.
`managedDataSource`	`string`	ManagedDataSource specifies the name of the corresponding DataSource this cron will manage. DataSource has to be in the same namespace.
`retentionPolicy`	`string`	RetentionPolicy specifies whether the created DataVolumes and DataSources are retained when their DataImportCron is deleted. Default is RatainAll.
`schedule`	`string`	Schedule specifies in cron format when and how often to look for new imports
`serviceAccountName`	`string`	ServiceAccountName is the name of the ServiceAccount for creating DataVolumes.
`template`	`object`	Template specifies template for the DVs to be created

.spec.dataImportCronTemplates[].spec.template

Description

Template specifies template for the DVs to be created

Type

`object`

Required

`spec`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataVolumeSpec defines the DataVolume type specification
`status`	`object`	DataVolumeStatus contains the current status of the DataVolume

.spec.dataImportCronTemplates[].spec.template.spec

Description

DataVolumeSpec defines the DataVolume type specification

Type

`object`

Property	Type	Description
`checkpoints`	`array`	Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.
`contentType`	`string`	DataVolumeContentType options: "kubevirt", "archive"
`finalCheckpoint`	`boolean`	FinalCheckpoint indicates whether the current DataVolumeCheckpoint is the final checkpoint.
`preallocation`	`boolean`	Preallocation controls whether storage for DataVolumes should be allocated in advance.
`priorityClassName`	`string`	PriorityClassName for Importer, Cloner and Uploader pod
`pvc`	`object`	PVC is the PVC specification
`source`	`object`	Source is the src of the data for the requested DataVolume
`sourceRef`	`object`	SourceRef is an indirect reference to the source of data for the requested DataVolume
`storage`	`object`	Storage is the requested storage specification

.spec.dataImportCronTemplates[].spec.template.spec.checkpoints

Description

Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.checkpoints[]

Description

DataVolumeCheckpoint defines a stage in a warm migration.

Type

`object`

Required

`current` `previous`

Property	Type	Description
`current`	`string`	Current is the identifier of the snapshot created for this checkpoint.
`previous`	`string`	Previous is the identifier of the snapshot from the previous checkpoint.

.spec.dataImportCronTemplates[].spec.template.spec.pvc

Description

PVC is the PVC specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of

Property	Type	Description
		the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: • While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Property	Type	Description
`resources`	`object`	resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consider for binding.
`storageClassName`	`string`	storageClassName is the name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClass. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string or nil value indicates that no VolumeAttributesClass will be applied to the claim. If the claim enters an Infeasible error state, this field can be reset to its previous value (including nil) to cancel the modification. If the resource referred to by volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/volume-attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

Property	Type	Description
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.spec.dataImportCronTemplates[].spec.template.spec.pvc.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.accessModes[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.dataImportCronTemplates[].spec.template.spec.pvc.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.dataImportCronTemplates[].spec.template.spec.pvc.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly

Property	Type	Description
		specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.dataImportCronTemplates[].spec.template.spec.pvc.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[].values[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.source

Description

Source is the src of the data for the requested DataVolume

Type

`object`

Property	Type	Description
`blank`	`object`	DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC
`gcs`	`object`	DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source
`http`	`object`	DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs
`imageio`	`object`	DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`registry`	`object`	DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source
`s3`	`object`	DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source

Property	Type	Description
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot
`upload`	`object`	DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source
`vddk`	`object`	DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

.spec.dataImportCronTemplates[].spec.template.spec.source.blank

Description

DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.source.gcs

Description

DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source

Type

`object`

Required

`url`

Property	Type	Description
`secretRef`	`string`	SecretRef provides the secret reference needed to access the GCS source
`url`	`string`	URL is the url of the GCS source

.spec.dataImportCronTemplates[].spec.template.spec.source.http

Description

DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`extraHeaders`	`array`	ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests
`secretExtraHeaders`	`array`	SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information
`secretRef`	`string`	SecretRef A Secret reference, the secret should contain accessKeyId (user name) base64 encoded, and secretKey (password) also base64 encoded
`url`	`string`	URL is the URL of the http(s) endpoint

.spec.dataImportCronTemplates[].spec.template.spec.source.http.extraHeaders

Description

ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.source.http.extraHeaders[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.source.http.secretExtraHeaders

Description

SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.source.http.secretExtraHeaders[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.source.imageio

Description

DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source

Type

`object`

Required

`diskId` `url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the CA cert

Property	Type	Description
`diskId`	`string`	DiskID provides id of a disk to be imported
`insecureSkipVerify`	`boolean`	InsecureSkipVerify is a flag to skip certificate verification
`secretRef`	`string`	SecretRef provides the secret reference needed to access the ovirt-engine
`url`	`string`	URL is the URL of the ovirt-engine

.spec.dataImportCronTemplates[].spec.template.spec.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC
`namespace`	`string`	The namespace of the source PVC

.spec.dataImportCronTemplates[].spec.template.spec.source.registry

Description

DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source

Type

`object`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the Registry certs
`imageStream`	`string`	ImageStream is the name of image stream for import
`platform`	`object`	Platform describes the minimum runtime requirements of the image
`pullMethod`	`string`	PullMethod can be either "pod" (default import), or "node" (node docker cache based import)
`secretRef`	`string`	SecretRef provides the secret reference needed to access the Registry source
`url`	`string`	URL is the url of the registry source (starting with the scheme: docker, oci-archive)

.spec.dataImportCronTemplates[].spec.template.spec.source.registry.platform

Description

Platform describes the minimum runtime requirements of the image

Type

`object`

Property	Type	Description
`architecture`	`string`	Architecture specifies the image target CPU architecture

.spec.dataImportCronTemplates[].spec.template.spec.source.s3

Description

DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`secretRef`	`string`	SecretRef provides the secret reference needed to access the S3 source
`url`	`string`	URL is the url of the S3 source

.spec.dataImportCronTemplates[].spec.template.spec.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot

Property	Type	Description
`namespace`	`string`	The namespace of the source VolumeSnapshot

.spec.dataImportCronTemplates[].spec.template.spec.source.upload

Description

DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.source.vddk

Description

DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

Type

`object`

Property	Type	Description
`backingFile`	`string`	BackingFile is the path to the virtual hard disk to migrate from vCenter/ESXi
`extraArgs`	`string`	ExtraArgs is a reference to a ConfigMap containing extra arguments to pass directly to the VDDK library
`initImageURL`	`string`	InitImageURL is an optional URL to an image containing an extracted VDDK library, overrides v2v-vmware config map
`secretRef`	`string`	SecretRef provides a reference to a secret containing the username and password needed to access the vCenter or ESXi host
`thumbprint`	`string`	Thumbprint is the certificate thumbprint of the vCenter or ESXi host

Property	Type	Description
`url`	`string`	URL is the URL of the vCenter or ESXi host with the VM to migrate
`uuid`	`string`	UUID is the UUID of the virtual machine that the backing file is attached to in vCenter/ESXi

.spec.dataImportCronTemplates[].spec.template.spec.sourceRef

Description

SourceRef is an indirect reference to the source of data for the requested DataVolume

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	The kind of the source reference, currently only "DataSource" is supported
`name`	`string`	The name of the source reference
`namespace`	`string`	The namespace of the source reference, defaults to the DataVolume namespace

.spec.dataImportCronTemplates[].spec.template.spec.storage

Description

Storage is the requested storage specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1 ↗
`dataSource`	`object`	This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.
`dataSourceRef`	`object`	Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: • While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.
`resources`	`object`	Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-

Property	Type	Description
		volumes#resources ↗
`selector`	`object`	A label query over volumes to consider for binding.
`storageClassName`	`string`	Name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.
`volumeName`	`string`	VolumeName is the binding reference to the PersistentVolume backing this claim.

.spec.dataImportCronTemplates[].spec.template.spec.storage.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.storage.accessModes[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.storage.dataSource

Description

This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.dataImportCronTemplates[].spec.template.spec.storage.dataSourceRef

Description

Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: * While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.spec.dataImportCronTemplates[].spec.template.spec.storage.resources

Description

Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-

Property	Type	Description
		containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.dataImportCronTemplates[].spec.template.spec.storage.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.storage.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector

Description

A label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[].values[]

Type

`string`

.spec.dataImportCronTemplates[].spec.template.spec.storage.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.dataImportCronTemplates[].spec.template.status

Description

DataVolumeStatus contains the current status of the DataVolume

Type

`object`

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the underlying PVC used by the DataVolume.
`conditions`	`array`	
`phase`	`string`	Phase is the current phase of the data volume
`progress`	`string`	DataVolumeProgress is the current progress of the DataVolume transfer operation. Value between 0 and 100 inclusive, N/A if not available
`restartCount`	`integer`	RestartCount is the number of times the pod populating the DataVolume has restarted

.spec.dataImportCronTemplates[].spec.template.status.conditions

Type

`array`

.spec.dataImportCronTemplates[].spec.template.status.conditions[]

Description

DataVolumeCondition represents the state of a data volume condition.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastHeartbeatTime`	`string`	
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	DataVolumeConditionType is the string representation of known condition types

.spec.featureGates

Description

featureGates is a map of feature gate flags. Setting a flag to `true` will enable the feature. Setting `false` or removing the feature gate, disables the feature.

Type

`object`

Property	Type	Description
`alignCPUs`	`boolean`	Enable KubeVirt to request up to two additional dedicated CPUs in order to complete the total CPU count to an even parity when using emulator thread isolation. Note: this feature is in Developer Preview.
`autoResourceLimits`	`boolean`	Deprecated: this field is ignored and will be removed in the next version of the API.

Property	Type	Description
`decentralizedLiveMigration`	`boolean`	DecentralizedLiveMigration enables the decentralized live migration (cross-cluster migration) feature. This feature allows live migration of VirtualMachineInstances between different clusters. This feature is in Developer Preview.
`declarativeHotplugVolumes`	`boolean`	DeclarativeHotplugVolumes enables the use of the declarative volume hotplug feature in KubeVirt. When set to true, the "DeclarativeHotplugVolumes" feature gate is enabled instead of "HotplugVolumes". When set to false or nil, the "HotplugVolumes" feature gate is enabled (default behavior). This feature is in Developer Preview.
`deployKubeSecondaryDNS`	`boolean`	Deploy KubeSecondaryDNS by CNAO
`deployKubevirtIpamController`	`boolean`	Deprecated: this field is ignored and will be removed in the next version of the API.
`deployTektonTaskResources`	`boolean`	Deprecated: This field is ignored and will be removed on the next version of the API.
`deployVmConsoleProxy`	`boolean`	Deprecated: This field is ignored and will be removed on the next version of the API. Use spec.deployVmConsoleProxy instead
`disableMDevConfiguration`	`boolean`	Disable mediated devices handling on KubeVirt
`downwardMetrics`	`boolean`	Allow to expose a limited set of host metrics to guests.

Property	Type	Description
`enableApplicationAwareQuota`	`boolean`	Deprecated: This field is ignored and will be removed on the next version of the API. Use spec.enableApplicationAwareQuota instead
`enableCommonBootImageImport`	`boolean`	Deprecated: This field is ignored. Use spec.enableCommonBootImageImport instead
`enableManagedTenantQuota`	`boolean`	Deprecated: This field is ignored and will be removed on the next version of the API.
`enableMultiArchBootImageImport`	`boolean`	EnableMultiArchBootImageImport allows the HCO to run on heterogeneous clusters with different CPU architectures. Setting this field to true will allow the HCO to create Golden Images for different CPU architectures. This feature is in Developer Preview.
`nonRoot`	`boolean`	Deprecated: // Deprecated: This field is ignored and will be removed on the next version of the API.
`objectGraph`	`boolean`	ObjectGraph enables the ObjectGraph VM and VMI subresource in KubeVirt. This subresource returns a structured list of k8s objects that are related to the specified VM or VMI, enabling better dependency tracking. Note: This feature is in Developer Preview.
`persistentReservation`	`boolean`	Enable persistent reservation of a LUN through the SCSI Persistent Reserve commands on Kubevirt. In order to issue privileged SCSI ioctls, the VM requires activation of the persistent reservation flag. Once this feature gate is enabled, then the additional container with the qemu-pr-helper is deployed inside the virt-

Property	Type	Description
		handler pod. Enabling (or removing) the feature gate causes the redeployment of the virt-handler pod.
`primaryUserDefinedNetworkBinding`	`boolean`	primaryUserDefinedNetworkBinding deploys the needed configurations for kubevirt users to be able to bind their VM to a UDN network on the VM's primary interface. Deprecated: this field is ignored and will be removed in the next version of the API.
`videoConfig`	`boolean`	EnableVideoDeviceConfiguration allows users to configure video device types for their virtual machines. This can be useful for workloads that require specific video capabilities or architectures. Note: This feature is in Tech Preview.
`withHostPassthroughCPU`	`boolean`	Deprecated: there is no such FG in KubeVirt. This field is ignored

.spec.filesystemOverhead

Description

FilesystemOverhead describes the space reserved for overhead when using Filesystem volumes. A value is between 0 and 1, if not defined it is 0.055 (5.5 percent overhead)

Type

`object`

Property	Type	Description
`global`	`string`	Global is how much space of a Filesystem volume should be reserved for overhead. This value is used unless overridden by a more specific value (per storageClass)
`storageClass`	`object`	StorageClass specifies how much space of a Filesystem volume should be reserved for safety. The keys are the storageClass and the values are the

Property	Type	Description
		overhead. This value overrides the global value

.spec.filesystemOverhead.storageClass

Description

StorageClass specifies how much space of a Filesystem volume should be reserved for safety. The keys are the storageClass and the values are the overhead. This value overrides the global value

Type

`object`

.spec.higherWorkloadDensity

Description

HigherWorkloadDensity holds configuration aimed to increase virtual machine density

Type

`object`

Property	Type	Description
`memoryOvercommitPercentage`	`integer`	MemoryOvercommitPercentage is the percentage of memory we want to give VMIs compared to the amount given to its parent pod (virt-launcher). For example, a value of 102 means the VMI will "see" 2% more memory than its parent pod. Values under 100 are effectively "undercommits". Overcommits can lead to memory exhaustion, which in turn can lead to crashes. Use carefully.

.spec.infra

Description

infra HyperConvergedConfig influences the pod configuration (currently only placement) for all the infra components needed on the virtualization enabled cluster but not necessarily directly on each node running VMs/VMIs.

Type

`object`

Property	Type	Description
`nodePlacement`	`object`	NodePlacement describes node scheduling configuration.

.spec.infra.nodePlacement

Description

NodePlacement describes node scheduling configuration.

Type

`object`

Property	Type	Description
`affinity`	`object`	affinity enables pod affinity/anti-affinity placement expanding the types of constraints that can be expressed with nodeSelector. affinity is going to be applied to the relevant kind of pods in parallel with nodeSelector See https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#affinity-and-anti-affinity ↗
`nodeSelector`	`object`	nodeSelector is the node selector applied to the relevant kind of pods It specifies a map of key-value pairs: for the pod to be eligible to run on a node, the node must have each of the indicated key-value pairs as labels (it can have additional labels as well). See https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#nodeselector ↗
`tolerations`	`array`	tolerations is a list of tolerations applied to the relevant kind of pods See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/ ↗ for more info. These are additional tolerations other than default ones.

.spec.infra.nodePlacement.affinity

Description

affinity enables pod affinity/anti-affinity placement expanding the types of constraints that can be expressed with nodeSelector. affinity is going to be applied to the relevant kind of pods in parallel with nodeSelector See

https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#affinity-and-anti-affinity

Type

`object`

Property	Type	Description
`nodeAffinity`	`object`	Describes node affinity scheduling rules for the pod.
`podAffinity`	`object`	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
`podAntiAffinity`	`object`	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

.spec.infra.nodePlacement.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if

Property	Type	Description
		the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`object`	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

`object`

Required

`preference` `weight`

Property	Type	Description
`preference`	`object`	A node selector term, associated with the corresponding weight.
`weight`	`integer`	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference

Description

A node selector term, associated with the corresponding weight.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be

Property	Type	Description
		empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

`object`

Required

`nodeSelectorTerms`

Property	Type	Description
`nodeSelectorTerms`	`array`	Required. A list of node selector terms. The terms are ORed.

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms

Description

Required. A list of node selector terms. The terms are ORed.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[]

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have

a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.

Property	Type	Description
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Property	Type	Description
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchEx

pressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running

on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Property	Type	Description
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.infra.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this

Property	Type	Description
		field and subtracting "weight" from the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and subtracting "weight" from the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's

Property	Type	Description
		pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Property	Type	Description
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the

Property	Type	Description
		label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into

consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.infra.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.infra.nodePlacement.nodeSelector

Description

nodeSelector is the node selector applied to the relevant kind of pods It specifies a map of key-value pairs: for the pod to be eligible to run on a node, the node must have each of the indicated key-value pairs as labels (it can have additional labels as well). See https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#nodeselector

Type

`object`

.spec.infra.nodePlacement.tolerations

Description

tolerations is a list of tolerations applied to the relevant kind of pods See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/ for more info. These are additional tolerations other than default ones.

Type

`array`

.spec.infra.nodePlacement.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.

Property	Type	Description
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
`operator`	`string`	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
`tolerationSeconds`	`integer`	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.instancetypeConfig

Description

InstancetypeConfig holds the configuration of instance type related functionality within KubeVirt.

Type

`object`

Property	Type	Description
`referencePolicy`	`string`	ReferencePolicy defines how an instance type or preference should be referenced by the VM after submission, supported values are: reference (default) - Where a copy of the original object is stashed in a ControllerRevision and referenced by the VM. expand - Where the instance type or preference are expanded into the VM if no revisionNames have been populated. expandAll - Where the instance type or preference are expanded into the VM regardless of revisionNames previously being populated.

.spec.ksmConfiguration

Description

KSMConfiguration holds the information regarding the enabling the KSM in the nodes (if available).

Type

`object`

Property	Type	Description
`nodeLabelSelector`	`object`	NodeLabelSelector is a selector that filters in which nodes the KSM will be enabled. Empty NodeLabelSelector will enable ksm for every node.

.spec.ksmConfiguration.nodeLabelSelector

Description

NodeLabelSelector is a selector that filters in which nodes the KSM will be enabled. Empty NodeLabelSelector will enable ksm for every node.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.ksmConfiguration.nodeLabelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.ksmConfiguration.nodeLabelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.ksmConfiguration.nodeLabelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.ksmConfiguration.nodeLabelSelector.matchExpressions[].values[]

Type

`string`

.spec.ksmConfiguration.nodeLabelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.kubeMacPoolConfiguration

Description

KubeMacPoolConfiguration holds kubemacpool MAC address range configuration.

Type

`object`

Property	Type	Description
`rangeEnd`	`string`	RangeEnd defines the last MAC address in the kubemacpool range. The MAC address format should be AA:BB:CC:DD:EE:FF.

Property	Type	Description
`rangeStart`	`string`	RangeStart defines the first MAC address in the kubemacpool range. The MAC address format should be AA:BB:CC:DD:EE:FF.

.spec.liveMigrationConfig

Description

Live migration limits and timeouts are applied so that migration processes do not overwhelm the cluster.

Type

`object`

Property	Type	Description
`allowAutoConverge`	`boolean`	AllowAutoConverge allows the platform to compromise performance/availability of VMIs to guarantee successful VMI live migrations. Defaults to false
`allowPostCopy`	`boolean`	When enabled, KubeVirt attempts to use post-copy live-migration in case it reaches its completion timeout while attempting pre-copy live-migration. Post-copy migrations allow even the busiest VMs to successfully live-migrate. However, events like a network failure or a failure in any of the source or destination nodes can cause the migrated VM to crash or reach inconsistency. Enable this option when evicting nodes is more important than keeping VMs alive. Defaults to false.
`bandwidthPerMigration`	`string`	Bandwidth limit of each migration, the value is quantity of bytes per second (e.g. 2048Mi = 2048MiB/sec)
`completionTimeoutPerGiB`	`integer`	If a migrating VM is big and busy, while the connection to the destination node is slow, migration may never converge. The completion timeout is calculated based

Property	Type	Description
		on completionTimeoutPerGiB times the size of the guest (both RAM and migrated disks, if any). For example, with completionTimeoutPerGiB set to 800, a virtual machine instance with 6GiB memory will timeout if it has not completed migration in 1h20m. Use a lower completionTimeoutPerGiB to induce quicker failure, so that another destination or post-copy is attempted. Use a higher completionTimeoutPerGiB to let workload with spikes in its memory dirty rate to converge. The format is a number.
`network`	`string`	The migrations will be performed over a dedicated multus network to minimize disruption to tenant workloads due to network saturation when VM live migrations are triggered.
`parallelMigrationsPerCluster`	`integer`	Number of migrations running in parallel in the cluster.
`parallelOutboundMigrationsPerNode`	`integer`	Maximum number of outbound migrations per node.
`progressTimeout`	`integer`	The migration will be canceled if memory copy fails to make progress in this time, in seconds.

.spec.liveUpdateConfiguration

Description

LiveUpdateConfiguration holds the cluster configuration for live update of virtual machines - max cpu sockets, max guest memory and max hotplug ratio. This setting can affect VM CPU and memory settings.

Type

`object`

Property	Type	Description
`maxCpuSockets`	`integer`	MaxCpuSockets provides a MaxSockets value for VMs that do not provide their own. For VMs with more sockets than maximum the MaxSockets will be set to equal number of sockets.
`maxGuest`	`integer|string`	MaxGuest defines the maximum amount memory that can be allocated to the guest using hotplug.
`maxHotplugRatio`	`integer`	MaxHotplugRatio is the ratio used to define the max amount of a hotplug resource that can be made available to a VM when the specific Max* setting is not defined (MaxCpuSockets, MaxGuest) Example: VM is configured with 512Mi of guest memory, if MaxGuest is not defined and MaxHotplugRatio is 2 then MaxGuest = 1Gi defaults to 4

.spec.logVerbosityConfig

Description

LogVerbosityConfig configures the verbosity level of Kubevirt's different components. The higher the value - the higher the log verbosity.

Type

`object`

Property	Type	Description
`cdi`	`integer`	CDI indicates the log verbosity level that controls the amount of information logged for CDI components.
`kubevirt`	`object`	Kubevirt is a struct that allows specifying the log verbosity level that controls the amount of information logged for each Kubevirt component.

.spec.logVerbosityConfig.kubevirt

Description

Kubevirt is a struct that allows specifying the log verbosity level that controls the amount of information logged for each Kubevirt component.

Type

`object`

Property	Type	Description
`nodeVerbosity`	`object`	NodeVerbosity represents a map of nodes with a specific verbosity level
`virtAPI`	`integer`	
`virtController`	`integer`	
`virtHandler`	`integer`	
`virtLauncher`	`integer`	
`virtOperator`	`integer`	
`virtSynchronizationController`	`integer`	

.spec.logVerbosityConfig.kubevirt.nodeVerbosity

Description

NodeVerbosity represents a map of nodes with a specific verbosity level

Type

`object`

.spec.mediatedDevicesConfiguration

Description

MediatedDevicesConfiguration holds information about MDEV types to be defined on nodes, if available

Type

`object`

Property	Type	Description
`mediatedDeviceTypes`	`array`	

Property	Type	Description
`mediatedDevicesTypes`	`array`	Deprecated: please use mediatedDeviceTypes instead.
`nodeMediatedDeviceTypes`	`array`	

.spec.mediatedDevicesConfiguration.mediatedDeviceTypes

Type

`array`

.spec.mediatedDevicesConfiguration.mediatedDeviceTypes[]

Type

`string`

.spec.mediatedDevicesConfiguration.mediatedDevicesTypes

Description

Deprecated: please use mediatedDeviceTypes instead.

Type

`array`

.spec.mediatedDevicesConfiguration.mediatedDevicesTypes[]

Type

`string`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes

Type

`array`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[]

Description

NodeMediatedDeviceTypesConfig holds information about MDEV types to be defined in a specific node that matches the NodeSelector field.

Type

`object`

Required

`nodeSelector`

Property	Type	Description
`mediatedDeviceTypes`	`array`	
`mediatedDevicesTypes`	`array`	Deprecated: please use mediatedDeviceTypes instead.
`nodeSelector`	`object`	NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/ ↗

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[].mediatedDeviceTypes

Type

`array`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[].mediatedDeviceTypes[]

Type

`string`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[].mediatedDevicesTypes

Description

Deprecated: please use mediatedDeviceTypes instead.

Type

`array`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[].mediatedDevicesTypes[]

Type

`string`

.spec.mediatedDevicesConfiguration.nodeMediatedDeviceTypes[].nodeSelector

Description

NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/

Type

`object`

.spec.networkBinding

Description

NetworkBinding defines the network binding plugins. Those bindings can be used when defining virtual machine interfaces.

Type

`object`

.spec.obsoleteCPUs

Description

ObsoleteCPUs allows avoiding scheduling of VMs for obsolete CPU models

Type

`object`

Property	Type	Description
`cpuModels`	`array`	CPUModels is a list of obsolete CPU models. When the node-labeller obtains the list of obsolete CPU models, it eliminates those CPU models and creates labels for valid CPU models. The default values for this field is nil, however,

Property	Type	Description
		HCO uses opinionated values, and adding values to this list will add them to the opinionated values.
`minCPUModel`	`string`	MinCPUModel is not in use Deprecated: This field is not in use and is ignored.

.spec.obsoleteCPUs.cpuModels

Description

CPUModels is a list of obsolete CPU models. When the node-labeller obtains the list of obsolete CPU models, it eliminates those CPU models and creates labels for valid CPU models. The default values for this field is nil, however, HCO uses opinionated values, and adding values to this list will add them to the opinionated values.

Type

`array`

.spec.obsoleteCPUs.cpuModels[]

Type

`string`

.spec.permittedHostDevices

Description

PermittedHostDevices holds information about devices allowed for passthrough

Type

`object`

Property	Type	Description
`mediatedDevices`	`array`	
`pciHostDevices`	`array`	
`usbHostDevices`	`array`	

.spec.permittedHostDevices.mediatedDevices

Type

`array`

.spec.permittedHostDevices.mediatedDevices[]

Description

MediatedHostDevice represents a host mediated device allowed for passthrough

Type

`object`

Required

`mdevNameSelector` `resourceName`

Property	Type	Description
`disabled`	`boolean`	HCO enforces the existence of several MediatedHostDevice objects. Set disabled field to true instead of remove these objects.
`externalResourceProvider`	`boolean`	indicates that this resource is being provided by an external device plugin
`mdevNameSelector`	`string`	name of a mediated device type required to identify a mediated device on a host
`resourceName`	`string`	name by which a device is advertised and being requested

.spec.permittedHostDevices.pciHostDevices

Type

`array`

.spec.permittedHostDevices.pciHostDevices[]

Description

PciHostDevice represents a host PCI device allowed for passthrough

Type

`object`

Required

`pciDeviceSelector` `resourceName`

Property	Type	Description
`disabled`	`boolean`	HCO enforces the existence of several PciHostDevice objects. Set disabled field to true instead of remove these objects.
`externalResourceProvider`	`boolean`	indicates that this resource is being provided by an external device plugin
`pciDeviceSelector`	`string`	a combination of a vendor_id:product_id required to identify a PCI device on a host.
`resourceName`	`string`	name by which a device is advertised and being requested

.spec.permittedHostDevices.usbHostDevices

Type

`array`

.spec.permittedHostDevices.usbHostDevices[]

Description

USBHostDevice represents a host USB device allowed for passthrough

Type

`object`

Required

`resourceName`

Property	Type	Description
`disabled`	`boolean`	HCO enforces the existence of several USBHostDevice objects. Set disabled field to true instead of remove these objects.
`externalResourceProvider`	`boolean`	If true, KubeVirt will leave the allocation and monitoring to an external device plugin
`resourceName`	`string`	Identifies the list of USB host devices. e.g: kubevirt.io/storage, kubevirt.io/bootable-usb, etc
`selectors`	`array`	

.spec.permittedHostDevices.usbHostDevices[].selectors

Type

`array`

.spec.permittedHostDevices.usbHostDevices[].selectors[]

Description

USBSelector represents a selector for a USB device allowed for passthrough

Type

`object`

Required

`product` `vendor`

Property	Type	Description
`product`	`string`	
`vendor`	`string`	

.spec.resourceRequirements

Description

ResourceRequirements describes the resource requirements for the operand workloads.

Type

`object`

Property	Type	Description
`autoCPULimitNamespaceLabelSelector`	`object`	When set, AutoCPULimitNamespaceLabelSelector will set a CPU limit on virt-launcher for VMIs running inside namespaces that match the label selector. The CPU limit will equal the number of requested vCPUs. This setting does not apply to VMIs with dedicated CPUs.
`storageWorkloads`	`object`	StorageWorkloads defines the resources requirements for storage workloads. It will propagate to the CDI custom resource
`vmiCPUAllocationRatio`	`integer`	VmiCPUAllocationRatio defines, for each requested virtual CPU, how much physical CPU to request per VMI from the hosting node. The value is in fraction of a CPU thread (or core on non-hyperthreaded nodes). VMI POD CPU request = number of vCPUs * 1/vmiCPUAllocationRatio For example, a value of 1 means 1 physical CPU thread per VMI CPU thread. A value of 100 would be 1% of a physical thread allocated for each requested VMI thread. This option has no effect on VMIs that request dedicated CPUs. Defaults to 10

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector

Description

When set, AutoCPULimitNamespaceLabelSelector will set a CPU limit on virt-launcher for VMIs running inside namespaces that match the label selector. The CPU limit will equal the number of requested vCPUs. This setting does not apply to VMIs with dedicated CPUs.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

Property	Type	Description
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector.matchExpressions[].values[]

Type

`string`

.spec.resourceRequirements.autoCPULimitNamespaceLabelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.resourceRequirements.storageWorkloads

Description

StorageWorkloads defines the resources requirements for storage workloads. It will propagate to the CDI custom resource

Type

`object`

Property	Type	Description
`claims`	`array`	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This field depends on the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.resourceRequirements.storageWorkloads.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This field depends on the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

`array`

.spec.resourceRequirements.storageWorkloads.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.
`request`	`string`	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

.spec.resourceRequirements.storageWorkloads.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.resourceRequirements.storageWorkloads.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.spec.storageImport

Description

StorageImport contains configuration for importing containerized data

Type

`object`

Property	Type	Description
`insecureRegistries`	`array`	InsecureRegistries is a list of image registries URLs that are not secured. Setting an insecure registry URL in this list allows pulling images from this registry.

.spec.storageImport.insecureRegistries

Description

InsecureRegistries is a list of image registries URLs that are not secured. Setting an insecure registry URL in this list allows pulling images from this registry.

Type

`array`

.spec.storageImport.insecureRegistries[]

Type

`string`

.spec.tlsSecurityProfile

Description

TLSSecurityProfile specifies the settings for TLS connections to be propagated to all kubevirt-hyperconverged components. If unset, the hyperconverged cluster operator will consume the value set on the APIServer CR on OCP/OKD or Intermediate if on vanilla k8s. Note that only Old, Intermediate and Custom profiles are currently supported, and the maximum available MinTLSVersions is VersionTLS12.

Type

`object`

Property	Type	Description

`custom`	`object`	custom is a user-defined TLS security profile. Be extremely careful using a custom p
		invalid configurations can be catastrophic. An example custom profile looks like this:
		
		ciphers:
		
		`- ECDHE-ECDSA-CHACHA20-POLY1305`
		`- ECDHE-RSA-CHACHA20-POLY1305`
		`- ECDHE-RSA-AES128-GCM-SHA256`
		`- ECDHE-ECDSA-AES128-GCM-SHA256`
		
		minTLSVersion: VersionTLS11

`intermediate`	`object`	intermediate is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Intermediate_compatibility_.28rec ↗ and looks like this (yaml): ciphers:

Property	Type	Description
		- TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 minTLSVersion: VersionTLS12
`modern`	`object`	modern is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Modern_compatibility ↗ and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 minTLSVersion: VersionTLS13
`old`	`object`	old is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Old_backward_compatibility ↗

Property	Type	Description
		and looks like this (yaml): ciphers:

Property	Type	Description

Property	Type	Description
		- TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 - DHE-RSA-CHACHA20-POLY1305 - ECDHE-ECDSA-AES128-SHA256 - ECDHE-RSA-AES128-SHA256 - ECDHE-ECDSA-AES128-SHA - ECDHE-RSA-AES128-SHA - ECDHE-ECDSA-AES256-SHA384 - ECDHE-RSA-AES256-SHA384 - ECDHE-ECDSA-AES256-SHA - ECDHE-RSA-AES256-SHA - DHE-RSA-AES128-SHA256 - DHE-RSA-AES256-SHA256 - AES128-GCM-SHA256 - AES256-GCM-SHA384 - AES128-SHA256

```
- AES256-SHA256
- AES128-SHA
- AES256-SHA
- DES-CBC3-SHA
```

minTLSVersion: VersionTLS10

`type`	`string`	type is one of Old, Intermediate, Modern or Custom. Custom provides the ability to s individual TLS security profile parameters. Old, Intermediate and Modern are TLS se based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Recommended_configurations ↗ The profiles are intent based, so they may change over time as new ciphers are dev existing ciphers are found to be insecure. Depending on precisely which ciphers are process, the list may be reduced. Note that the Modern profile is currently not supported because it is not yet well ado common software libraries.

.spec.tlsSecurityProfile.custom

Description

custom is a user-defined TLS security profile. Be extremely careful using a custom profile as invalid configurations can be catastrophic. An example custom profile looks like this: ciphers: - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 minTLSVersion: VersionTLS11

Type

`object`

Property	Type	Description
`ciphers`	`array`	ciphers is used to specify the cipher algorithms that are negotiated during the TLS handshake. Operators may remove entries their operands do not support. For example, to use DES-CBC3-SHA (yaml): ciphers: - DES-CBC3-SHA

Property	Type	Description
`minTLSVersion`	`string`	minTLSVersion is used to specify the minimal version of the TLS protocol that is negotiated during the TLS handshake. For example, to use TLS versions 1.1, 1.2 and 1.3 (yaml): minTLSVersion: VersionTLS11 NOTE: currently the highest minTLSVersion allowed is VersionTLS12

.spec.tlsSecurityProfile.custom.ciphers

Description

ciphers is used to specify the cipher algorithms that are negotiated during the TLS handshake. Operators may remove entries their operands do not support. For example, to use DES-CBC3-SHA (yaml): ciphers: - DES-CBC3-SHA

Type

`array`

.spec.tlsSecurityProfile.custom.ciphers[]

Type

`string`

.spec.tlsSecurityProfile.intermediate

Description

intermediate is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Intermediate_compatibility_.28recommended.29 and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 minTLSVersion: VersionTLS12

Type

`object`

.spec.tlsSecurityProfile.modern

Description

modern is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Modern_compatibility and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 minTLSVersion: VersionTLS13

Type

`object`

.spec.tlsSecurityProfile.old

Description

old is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Old_backward_compatibility and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 - DHE-RSA-CHACHA20-POLY1305 - ECDHE-ECDSA-AES128-SHA256 - ECDHE-RSA-AES128-SHA256 - ECDHE-ECDSA-AES128-SHA - ECDHE-RSA-AES128-SHA - ECDHE-ECDSA-AES256-SHA384 - ECDHE-RSA-AES256-SHA384 - ECDHE-ECDSA-AES256-SHA - ECDHE-RSA-AES256-SHA - DHE-RSA-AES128-SHA256 - DHE-RSA-AES256-SHA256 - AES128-GCM-SHA256 - AES256-GCM-SHA384 - AES128-SHA256 - AES256-SHA256 - AES128-SHA - AES256-SHA - DES-CBC3-SHA minTLSVersion: VersionTLS10

Type

`object`

.spec.virtualMachineOptions

Description

VirtualMachineOptions holds the cluster level information regarding the virtual machine.

Type

`object`

Property	Type	Description
`disableFreePageReporting`	`boolean`	DisableFreePageReporting disable the free page reporting of memory balloon device https://libvirt.org/formatdomain.html#memory-balloon-device↗. This will have effect only if AutoattachMemBalloon is not false and the vmi is not requesting any high performance feature

Property	Type	Description
		(dedicatedCPU/realtime/hugePages), in which free page reporting is always disabled.
`disableSerialConsoleLog`	`boolean`	DisableSerialConsoleLog disables logging the auto-attached default serial console. If not set, serial console logs will be written to a file and then streamed from a container named `guest-console-log`. The value can be individually overridden for each VM, not relevant if AutoattachSerialConsole is disabled for the VM.

.spec.workloadUpdateStrategy

Description

WorkloadUpdateStrategy defines at the cluster level how to handle automated workload updates

Type

`object`

Required

`workloadUpdateMethods`

Property	Type	Description
`batchEvictionInterval`	`string`	BatchEvictionInterval Represents the interval to wait before issuing the next batch of shutdowns
`batchEvictionSize`	`integer`	BatchEvictionSize Represents the number of VMIs that can be forced updated per the BatchShutdownInterval interval

Property	Type	Description
`workloadUpdateMethods`	`array`	WorkloadUpdateMethods defines the methods that can be used to disrupt workloads during automated workload updates. When multiple methods are present, the least disruptive method takes precedence over more disruptive methods. For example if both LiveMigrate and Evict methods are listed, only VMs which are not live migratable will be restarted/shutdown. An empty list defaults to no automated workload updating.

.spec.workloadUpdateStrategy.workloadUpdateMethods

Description

WorkloadUpdateMethods defines the methods that can be used to disrupt workloads during automated workload updates. When multiple methods are present, the least disruptive method takes precedence over more disruptive methods. For example if both LiveMigrate and Evict methods are listed, only VMs which are not live migratable will be restarted/shutdown. An empty list defaults to no automated workload updating.

Type

`array`

.spec.workloadUpdateStrategy.workloadUpdateMethods[]

Type

`string`

.spec.workloads

Description

workloads HyperConvergedConfig influences the pod configuration (currently only placement) of components which need to be running on a node where virtualization workloads should be able to run. Changes to Workloads HyperConvergedConfig can be applied only without existing workload.

Type

`object`

Property	Type	Description
`nodePlacement`	`object`	NodePlacement describes node scheduling configuration.

.spec.workloads.nodePlacement

Description

NodePlacement describes node scheduling configuration.

Type

`object`

Property	Type	Description
`affinity`	`object`	affinity enables pod affinity/anti-affinity placement expanding the types of constraints that can be expressed with nodeSelector. affinity is going to be applied to the relevant kind of pods in parallel with nodeSelector See https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#affinity-and-anti-affinity ↗
`nodeSelector`	`object`	nodeSelector is the node selector applied to the relevant kind of pods It specifies a map of key-value pairs: for the pod to be eligible to run on a node, the node must have each of the indicated key-value pairs as labels (it can have additional labels as well). See https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#nodeselector ↗
`tolerations`	`array`	tolerations is a list of tolerations applied to the relevant kind of pods See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/ ↗ for more info. These are additional tolerations other than default ones.

.spec.workloads.nodePlacement.affinity

Description

affinity enables pod affinity/anti-affinity placement expanding the types of constraints that can be expressed with nodeSelector. affinity is going to be applied to the relevant kind of pods in parallel with nodeSelector See https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#affinity-and-anti-affinity

Type

`object`

Property	Type	Description
`nodeAffinity`	`object`	Describes node affinity scheduling rules for the pod.
`podAffinity`	`object`	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
`podAntiAffinity`	`object`	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

.spec.workloads.nodePlacement.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`object`	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements

(resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

`object`

Required

`preference` `weight`

Property	Type	Description
`preference`	`object`	A node selector term, associated with the corresponding weight.
`weight`	`integer`	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference

Description

A node selector term, associated with the corresponding weight.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

`object`

Required

`nodeSelectorTerms`

Property	Type	Description
`nodeSelectorTerms`	`array`	Required. A list of node selector terms. The terms are ORed.

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms

Description

Required. A list of node selector terms. The terms are ORed.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[]

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of node selector requirements by node's labels.
`matchFields`	`array`	A list of node selector requirements by node's fields.

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions

Description

A list of node selector requirements by node's labels.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields

Description

A list of node selector requirements by node's fields.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	The label key that the selector applies to.
`operator`	`string`	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values

Description

An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matchFields[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches

Property	Type	Description
		the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be

Property	Type	Description
		ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic

merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be

Property	Type	Description
		ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.workloads.nodePlacement.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

`object`

Property	Type	Description
`preferredDuringSchedulingIgnoredDuringExecution`	`array`	The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and subtracting "weight" from the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
`requiredDuringSchedulingIgnoredDuringExecution`	`array`	If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and subtracting "weight" from the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

`object`

Required

`podAffinityTerm` `weight`

Property	Type	Description
`podAffinityTerm`	`object`	Required. A pod affinity term, associated with the corresponding weight.
`weight`	`integer`	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Property	Type	Description
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the

incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.matchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.mismatchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespace

Selector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespaces[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

`object`

Required

`topologyKey`

Property	Type	Description
`labelSelector`	`object`	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
`matchLabelKeys`	`array`	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.
`mismatchLabelKeys`	`array`	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to

Property	Type	Description
		exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.
`namespaceSelector`	`object`	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
`namespaces`	`array`	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
`topologyKey`	`string`	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key in (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].matchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys

Description

MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with `labelSelector` as `key notin (value)` to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].mismatchLabelKeys[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchExpressions[].values[]

Type

`string`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces

Description

namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Type

`array`

.spec.workloads.nodePlacement.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaces[]

Type

`string`

.spec.workloads.nodePlacement.nodeSelector

Description

nodeSelector is the node selector applied to the relevant kind of pods It specifies a map of key-value pairs: for the pod to be eligible to run on a node, the node must have each of the indicated key-value pairs as labels (it can have additional labels as well). See https://kubernetes.io/docs/concepts/configuration/assign-pod-node/#nodeselector

Type

`object`

.spec.workloads.nodePlacement.tolerations

Description

tolerations is a list of tolerations applied to the relevant kind of pods See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/ for more info. These are additional tolerations other than default ones.

Type

`array`

.spec.workloads.nodePlacement.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
`operator`	`string`	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
`tolerationSeconds`	`integer`	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.

Property	Type	Description
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.status

Description

HyperConvergedStatus defines the observed state of HyperConverged

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions describes the state of the HyperConverged resource.
`dataImportCronTemplates`	`array`	DataImportCronTemplates is a list of the actual DataImportCronTemplates as HCO update in the SSP CR. The list contains both the common and the custom templates, including any modification done by HCO.
`dataImportSchedule`	`string`	DataImportSchedule is the cron expression that is used in for the hard-coded data import cron templates. HCO generates the value of this field once and stored in the status field, so will survive restart.
`infrastructureHighlyAvailable`	`boolean`	InfrastructureHighlyAvailable describes whether the cluster has only one worker node (false) or more (true).
`nodeInfo`	`object`	NodeInfo holds information about the cluster nodes

Property	Type	Description
`observedGeneration`	`integer`	ObservedGeneration reflects the HyperConverged resource generation. If the ObservedGeneration is less than the resource generation in metadata, the status is out of date
`relatedObjects`	`array`	RelatedObjects is a list of objects created and maintained by this operator. Object references will be added to this list after they have been created AND found in the cluster.
`systemHealthStatus`	`string`	SystemHealthStatus reflects the health of HCO and its secondary resources, based on the aggregated conditions.
`versions`	`array`	Versions is a list of HCO component versions, as name/version pairs. The version with a name of "operator" is the HCO version itself, as described here: https://github.com/openshift/cluster-version-operator/blob/master/docs/dev/clusteroperator.md#version ↗

.status.conditions

Description

Conditions describes the state of the HyperConverged resource.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.dataImportCronTemplates

Description

DataImportCronTemplates is a list of the actual DataImportCronTemplates as HCO update in the SSP CR. The list contains both the common and the custom templates, including any modification done by HCO.

Type

`array`

.status.dataImportCronTemplates[]

Description

DataImportCronTemplateStatus is a copy of a dataImportCronTemplate as defined in the spec, or in the HCO image.

Type

`object`

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataImportCronSpec defines specification for DataImportCron
`status`	`object`	DataImportCronStatus is the status field of the DIC template

.status.dataImportCronTemplates[].spec

Description

DataImportCronSpec defines specification for DataImportCron

Type

`object`

Required

`managedDataSource` `schedule` `template`

Property	Type	Description
`garbageCollect`	`string`	GarbageCollect specifies whether old PVCs should be cleaned up after a new PVC is imported. Options are currently "Outdated" and "Never", defaults to "Outdated".
`importsToKeep`	`integer`	Number of import PVCs to keep when garbage collecting. Default is 3.

Property	Type	Description
`managedDataSource`	`string`	ManagedDataSource specifies the name of the corresponding DataSource this cron will manage. DataSource has to be in the same namespace.
`retentionPolicy`	`string`	RetentionPolicy specifies whether the created DataVolumes and DataSources are retained when their DataImportCron is deleted. Default is RatainAll.
`schedule`	`string`	Schedule specifies in cron format when and how often to look for new imports
`serviceAccountName`	`string`	ServiceAccountName is the name of the ServiceAccount for creating DataVolumes.
`template`	`object`	Template specifies template for the DVs to be created

.status.dataImportCronTemplates[].spec.template

Description

Template specifies template for the DVs to be created

Type

`object`

Required

`spec`

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataVolumeSpec defines the DataVolume type specification
`status`	`object`	DataVolumeStatus contains the current status of the DataVolume

.status.dataImportCronTemplates[].spec.template.spec

Description

DataVolumeSpec defines the DataVolume type specification

Type

`object`

Property	Type	Description
`checkpoints`	`array`	Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Property	Type	Description
`contentType`	`string`	DataVolumeContentType options: "kubevirt", "archive"
`finalCheckpoint`	`boolean`	FinalCheckpoint indicates whether the current DataVolumeCheckpoint is the final checkpoint.
`preallocation`	`boolean`	Preallocation controls whether storage for DataVolumes should be allocated in advance.
`priorityClassName`	`string`	PriorityClassName for Importer, Cloner and Uploader pod
`pvc`	`object`	PVC is the PVC specification
`source`	`object`	Source is the src of the data for the requested DataVolume
`sourceRef`	`object`	SourceRef is an indirect reference to the source of data for the requested DataVolume
`storage`	`object`	Storage is the requested storage specification

.status.dataImportCronTemplates[].spec.template.spec.checkpoints

Description

Checkpoints is a list of DataVolumeCheckpoints, representing stages in a multistage import.

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.checkpoints[]

Description

DataVolumeCheckpoint defines a stage in a warm migration.

Type

`object`

Required

`current` `previous`

Property	Type	Description
`current`	`string`	Current is the identifier of the snapshot created for this checkpoint.
`previous`	`string`	Previous is the identifier of the snapshot from the previous checkpoint.

.status.dataImportCronTemplates[].spec.template.spec.pvc

Description

PVC is the PVC specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1↗
`dataSource`	`object`	dataSource field can be used to specify either: • An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) • An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be

Property	Type	Description
		copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.
`dataSourceRef`	`object`	dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: • While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.
`resources`	`object`	resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than

Property	Type	Description
		capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗
`selector`	`object`	selector is a label query over volumes to consider for binding.
`storageClassName`	`string`	storageClassName is the name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeAttributesClassName`	`string`	volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClass. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string or nil value indicates that no VolumeAttributesClass will be applied to the claim. If the claim enters an Infeasible error state, this field can be reset to its previous value (including nil) to cancel the modification. If the resource referred to by volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/volume-attributes-classes/ ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.
`volumeName`	`string`	volumeName is the binding reference to the PersistentVolume backing this claim.

.status.dataImportCronTemplates[].spec.template.spec.pvc.accessModes

Description

accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.pvc.accessModes[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.pvc.dataSource

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.

Property	Type	Description
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.status.dataImportCronTemplates[].spec.template.spec.pvc.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.

Property	Type	Description
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.dataImportCronTemplates[].spec.template.spec.pvc.resources

Description

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.dataImportCronTemplates[].spec.template.spec.pvc.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.pvc.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector

Description

selector is a label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchExpressions[].values[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.pvc.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.source

Description

Source is the src of the data for the requested DataVolume

Type

`object`

Property	Type	Description
`blank`	`object`	DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC
`gcs`	`object`	DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source

Property	Type	Description
`http`	`object`	DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs
`imageio`	`object`	DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source
`pvc`	`object`	DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC
`registry`	`object`	DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source
`s3`	`object`	DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source
`snapshot`	`object`	DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot
`upload`	`object`	DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source
`vddk`	`object`	DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

.status.dataImportCronTemplates[].spec.template.spec.source.blank

Description

DataVolumeBlankImage provides the parameters to create a new raw blank image for the PVC

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.source.gcs

Description

DataVolumeSourceGCS provides the parameters to create a Data Volume from an GCS source

Type

`object`

Required

`url`

Property	Type	Description
`secretRef`	`string`	SecretRef provides the secret reference needed to access the GCS source
`url`	`string`	URL is the url of the GCS source

.status.dataImportCronTemplates[].spec.template.spec.source.http

Description

DataVolumeSourceHTTP can be either an http or https endpoint, with an optional basic auth user name and password, and an optional configmap containing additional CAs

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate

Property	Type	Description
`extraHeaders`	`array`	ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests
`secretExtraHeaders`	`array`	SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information
`secretRef`	`string`	SecretRef A Secret reference, the secret should contain accessKeyId (user name) base64 encoded, and secretKey (password) also base64 encoded
`url`	`string`	URL is the URL of the http(s) endpoint

.status.dataImportCronTemplates[].spec.template.spec.source.http.extraHeaders

Description

ExtraHeaders is a list of strings containing extra headers to include with HTTP transfer requests

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.source.http.extraHeaders[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.source.http.secretExtraHeaders

Description

SecretExtraHeaders is a list of Secret references, each containing an extra HTTP header that may include sensitive information

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.source.http.secretExtraHeaders[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.source.imageio

Description

DataVolumeSourceImageIO provides the parameters to create a Data Volume from an imageio source

Type

`object`

Required

`diskId` `url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the CA cert
`diskId`	`string`	DiskID provides id of a disk to be imported
`insecureSkipVerify`	`boolean`	InsecureSkipVerify is a flag to skip certificate verification
`secretRef`	`string`	SecretRef provides the secret reference needed to access the ovirt-engine
`url`	`string`	URL is the URL of the ovirt-engine

.status.dataImportCronTemplates[].spec.template.spec.source.pvc

Description

DataVolumeSourcePVC provides the parameters to create a Data Volume from an existing PVC

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source PVC
`namespace`	`string`	The namespace of the source PVC

.status.dataImportCronTemplates[].spec.template.spec.source.registry

Description

DataVolumeSourceRegistry provides the parameters to create a Data Volume from an registry source

Type

`object`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap provides a reference to the Registry certs
`imageStream`	`string`	ImageStream is the name of image stream for import
`platform`	`object`	Platform describes the minimum runtime requirements of the image

Property	Type	Description
`pullMethod`	`string`	PullMethod can be either "pod" (default import), or "node" (node docker cache based import)
`secretRef`	`string`	SecretRef provides the secret reference needed to access the Registry source
`url`	`string`	URL is the url of the registry source (starting with the scheme: docker, oci-archive)

.status.dataImportCronTemplates[].spec.template.spec.source.registry.platform

Description

Platform describes the minimum runtime requirements of the image

Type

`object`

Property	Type	Description
`architecture`	`string`	Architecture specifies the image target CPU architecture

.status.dataImportCronTemplates[].spec.template.spec.source.s3

Description

DataVolumeSourceS3 provides the parameters to create a Data Volume from an S3 source

Type

`object`

Required

`url`

Property	Type	Description
`certConfigMap`	`string`	CertConfigMap is a configmap reference, containing a Certificate Authority(CA) public key, and a base64 encoded pem certificate
`secretRef`	`string`	SecretRef provides the secret reference needed to access the S3 source
`url`	`string`	URL is the url of the S3 source

.status.dataImportCronTemplates[].spec.template.spec.source.snapshot

Description

DataVolumeSourceSnapshot provides the parameters to create a Data Volume from an existing VolumeSnapshot

Type

`object`

Required

`name` `namespace`

Property	Type	Description
`name`	`string`	The name of the source VolumeSnapshot
`namespace`	`string`	The namespace of the source VolumeSnapshot

.status.dataImportCronTemplates[].spec.template.spec.source.upload

Description

DataVolumeSourceUpload provides the parameters to create a Data Volume by uploading the source

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.source.vddk

Description

DataVolumeSourceVDDK provides the parameters to create a Data Volume from a Vmware source

Type

`object`

Property	Type	Description
`backingFile`	`string`	BackingFile is the path to the virtual hard disk to migrate from vCenter/ESXi
`extraArgs`	`string`	ExtraArgs is a reference to a ConfigMap containing extra arguments to pass directly to the VDDK library
`initImageURL`	`string`	InitImageURL is an optional URL to an image containing an extracted VDDK library, overrides v2v-vmware config map
`secretRef`	`string`	SecretRef provides a reference to a secret containing the username and password needed to access the vCenter or ESXi host
`thumbprint`	`string`	Thumbprint is the certificate thumbprint of the vCenter or ESXi host
`url`	`string`	URL is the URL of the vCenter or ESXi host with the VM to migrate
`uuid`	`string`	UUID is the UUID of the virtual machine that the backing file is attached to in vCenter/ESXi

.status.dataImportCronTemplates[].spec.template.spec.sourceRef

Description

SourceRef is an indirect reference to the source of data for the requested DataVolume

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	The kind of the source reference, currently only "DataSource" is supported
`name`	`string`	The name of the source reference
`namespace`	`string`	The namespace of the source reference, defaults to the DataVolume namespace

.status.dataImportCronTemplates[].spec.template.spec.storage

Description

Storage is the requested storage specification

Type

`object`

Property	Type	Description
`accessModes`	`array`	AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1
`dataSource`	`object`	This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is

Property	Type	Description
		enabled, this field will always have the same contents as the DataSourceRef field.
`dataSourceRef`	`object`	Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: • While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. • While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.
`resources`	`object`	Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗
`selector`	`object`	A label query over volumes to consider for binding.
`storageClassName`	`string`	Name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗
`volumeMode`	`string`	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

Property	Type	Description
`volumeName`	`string`	VolumeName is the binding reference to the PersistentVolume backing this claim.

.status.dataImportCronTemplates[].spec.template.spec.storage.accessModes

Description

AccessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.storage.accessModes[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.storage.dataSource

Description

This field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) * An existing custom resource that implements data population (Alpha) In order to use custom resource types that implement data population, the AnyVolumeDataSource feature gate must be enabled. If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. If the AnyVolumeDataSource feature gate is enabled, this field will always have the same contents as the DataSourceRef field.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.status.dataImportCronTemplates[].spec.template.spec.storage.dataSourceRef

Description

Specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any local object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the DataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, both fields (DataSource and DataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. There are two important differences between DataSource and DataSourceRef: * While DataSource only allows two specific types of objects, DataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While DataSource ignores disallowed values (dropping them), DataSourceRef preserves all values, and generates an error if a disallowed value is specified. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`apiGroup`	`string`	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.

Property	Type	Description
`kind`	`string`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced
`namespace`	`string`	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

.status.dataImportCronTemplates[].spec.template.spec.storage.resources

Description

Resources represents the minimum resources the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources

Type

`object`

Property	Type	Description
`limits`	`object`	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗
`requests`	`object`	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗

.status.dataImportCronTemplates[].spec.template.spec.storage.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.storage.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Type

`object`

.status.dataImportCronTemplates[].spec.template.spec.storage.selector

Description

A label query over volumes to consider for binding.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
`matchLabels`	`object`	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.status.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
`values`	`array`	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.status.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

.status.dataImportCronTemplates[].spec.template.spec.storage.selector.matchExpressions[].values[]

Type

`string`

.status.dataImportCronTemplates[].spec.template.spec.storage.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

`object`

.status.dataImportCronTemplates[].spec.template.status

Description

DataVolumeStatus contains the current status of the DataVolume

Type

`object`

Property	Type	Description
`claimName`	`string`	ClaimName is the name of the underlying PVC used by the DataVolume.
`conditions`	`array`	
`phase`	`string`	Phase is the current phase of the data volume
`progress`	`string`	DataVolumeProgress is the current progress of the DataVolume transfer operation. Value between 0 and 100 inclusive, N/A if not available
`restartCount`	`integer`	RestartCount is the number of times the pod populating the DataVolume has restarted

.status.dataImportCronTemplates[].spec.template.status.conditions

Type

`array`

.status.dataImportCronTemplates[].spec.template.status.conditions[]

Description

DataVolumeCondition represents the state of a data volume condition.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastHeartbeatTime`	`string`	

Property	Type	Description
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	
`status`	`string`	
`type`	`string`	DataVolumeConditionType is the string representation of known condition types

.status.dataImportCronTemplates[].status

Description

DataImportCronStatus is the status field of the DIC template

Type

`object`

Property	Type	Description
`commonTemplate`	`boolean`	CommonTemplate indicates whether this is a common template (true), or a custom one (false)
`conditions`	`array`	Conditions is a list of conditions that describe the state of the DataImportCronTemplate.
`modified`	`boolean`	Modified indicates if a common template was customized. Always false for custom templates.
`originalSupportedArchitectures`	`string`	OriginalSupportedArchitectures is a comma-separated list of CPU architectures that the original template supports.

.status.dataImportCronTemplates[].status.conditions

Description

Conditions is a list of conditions that describe the state of the DataImportCronTemplate.

Type

`array`

.status.dataImportCronTemplates[].status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.

Property	Type	Description
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.nodeInfo

Description

NodeInfo holds information about the cluster nodes

Type

`object`

Property	Type	Description
`controlPlaneArchitectures`	`array`	ControlPlaneArchitectures is a distinct list of the CPU architecture of the control-plane nodes.
`workloadsArchitectures`	`array`	WorkloadsArchitectures is a distinct list of the CPU architectures of the workloads nodes in the cluster.

.status.nodeInfo.controlPlaneArchitectures

Description

ControlPlaneArchitectures is a distinct list of the CPU architecture of the control-plane nodes.

Type

`array`

.status.nodeInfo.controlPlaneArchitectures[]

Type

`string`

.status.nodeInfo.workloadsArchitectures

Description

WorkloadsArchitectures is a distinct list of the CPU architectures of the workloads nodes in the cluster.

Type

`array`

.status.nodeInfo.workloadsArchitectures[]

Type

`string`

.status.relatedObjects

Description

RelatedObjects is a list of objects created and maintained by this operator. Object references will be added to this list after they have been created AND found in the cluster.

Type

`array`

.status.relatedObjects[]

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is

Property	Type	Description
		chosen only to have some well-defined way of referencing a part of an object.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.status.versions

Description

Versions is a list of HCO component versions, as name/version pairs. The version with a name of "operator" is the HCO version itself, as described here: https://github.com/openshift/cluster-version-operator/blob/master/docs/dev/clusteroperator.md#version

Type

`array`

.status.versions[]

Type

`object`

Property	Type	Description
`name`	`string`	
`version`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds`
  - `DELETE` : delete collection of HyperConverged
  - `GET` : list objects of kind HyperConverged
  - `POST` : create a new HyperConverged
- `/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds/{name}`
  - `DELETE` : delete the specified HyperConverged
  - `GET` : read the specified HyperConverged
  - `PATCH` : partially update the specified HyperConverged
  - `PUT` : replace the specified HyperConverged
- `/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds/{name}/status`
  - `GET` : read status of the specified HyperConverged
  - `PATCH` : partially update status of the specified HyperConverged
  - `PUT` : replace status of the specified HyperConverged

/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds

HTTP method

`DELETE`

Description

delete collection of HyperConverged

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind HyperConverged

HTTP responses

HTTP code	Response body
200 - OK	`HyperConvergedList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate

Parameter	Type	Description
		fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HyperConverged` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
201 - Created	`HyperConverged` schema
202 - Accepted	`HyperConverged` schema
401 - Unauthorized	Empty

/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds/{name}

HTTP method

`DELETE`

Description

delete the specified HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema

HTTP code	Response body
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified HyperConverged

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HyperConverged` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema

HTTP code	Response body
201 - Created	`HyperConverged` schema
401 - Unauthorized	Empty

/apis/hco.kubevirt.io/v1beta1/namespaces/{namespace}/hyperconvergeds/{name}/status

HTTP method

`GET`

Description

read status of the specified HyperConverged

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the

Parameter	Type	Description
		default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified HyperConverged

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`HyperConverged` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`HyperConverged` schema
201 - Created	`HyperConverged` schema
401 - Unauthorized	Empty

VirtualMachineClusterInstancetype [virtualmachineclusterinstancetypes.instancetype.kubevirt.io/v1beta1]

Description

VirtualMachineClusterInstancetype is a cluster scoped version of VirtualMachineInstancetype resource.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	Required spec describing the instancetype

.spec

Description

Required spec describing the instancetype

Type

`object`

Required

`cpu` `memory`

Property	Type	Description
`annotations`	`object`	Optionally defines the required Annotations to be used by the instance type and applied to the VirtualMachineInstance
`cpu`	`object`	Required CPU related attributes of the instancetype.
`gpus`	`array`	Optionally defines any GPU devices associated with the instancetype.
`hostDevices`	`array`	Optionally defines any HostDevices associated with the instancetype.
`ioThreads`	`object`	Optionally specifies the IOThreads options to be used by the instancetype.
`ioThreadsPolicy`	`string`	Optionally defines the IOThreadsPolicy to be used by the instancetype.
`launchSecurity`	`object`	Optionally defines the LaunchSecurity to be used by the instancetype.
`memory`	`object`	Required Memory related attributes of the instancetype.

Property	Type	Description
`nodeSelector`	`object`	NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/ ↗ NodeSelector is the name of the custom node selector for the instancetype.
`schedulerName`	`string`	If specified, the VMI will be dispatched by specified scheduler. If not specified, the VMI will be dispatched by default scheduler. SchedulerName is the name of the custom K8s scheduler for the instancetype.

.spec.annotations

Description

Optionally defines the required Annotations to be used by the instance type and applied to the VirtualMachineInstance

Type

`object`

.spec.cpu

Description

Required CPU related attributes of the instancetype.

Type

`object`

Required

`guest`

Property	Type	Description
`dedicatedCPUPlacement`	`boolean`	DedicatedCPUPlacement requests the scheduler to place the VirtualMachineInstance on a node with enough dedicated pCPUs and pin the vCPUs to it.
`guest`	`integer`	Required number of vCPUs to expose to the guest. The resulting CPU topology being derived from the optional PreferredCPUTopology attribute of CPUPreferences that itself defaults to PreferSockets
`isolateEmulatorThread`	`boolean`	IsolateEmulatorThread requests one more dedicated pCPU to be allocated for the VMI to place the emula thread on it.
`maxSockets`	`integer`	MaxSockets specifies the maximum amount of sock that can be hotplugged

Property	Type	Description
`model`	`string`	Model specifies the CPU model inside the VMI. List available models https://github.com/libvirt/libvirt/tree/master/src/cpu_m ↗. It is possible to specify special cases like "host-passthrough" to get the same CPU as the node and "host-model" to get CPU closest to the node one. Defaults to host-model.
`numa`	`object`	NUMA allows specifying settings for the guest NUMA topology
`realtime`	`object`	Realtime instructs the virt-launcher to tune the VMI f lower latency, optional for real time workloads

.spec.cpu.numa

Description

NUMA allows specifying settings for the guest NUMA topology

Type

`object`

Property	Type	Description
`guestMappingPassthrough`	`object`	GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on

Property	Type	Description
		the virtual numa nodes never cross boundaries of host numa nodes.

.spec.cpu.numa.guestMappingPassthrough

Description

GuestMappingPassthrough will create an efficient guest topology based on host CPUs exclusively assigned to a pod. The created topology ensures that memory and CPUs on the virtual numa nodes never cross boundaries of host numa nodes.

Type

`object`

.spec.cpu.realtime

Description

Realtime instructs the virt-launcher to tune the VMI for lower latency, optional for real time workloads

Type

`object`

Property	Type	Description
`mask`	`string`	Mask defines the vcpu mask expression that defines which vcpus are used for realtime. Format matches libvirt's expressions. Example: "0-3,^1","0,2,3","2-3"

.spec.gpus

Description

Optionally defines any GPU devices associated with the instancetype.

Type

`array`

.spec.gpus[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	Name of the GPU device as exposed by a device plugin
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config

Property	Type	Description
		drive
`virtualGPUOptions`	`object`	

.spec.gpus[].virtualGPUOptions

Type

`object`

Property	Type	Description
`display`	`object`	

.spec.gpus[].virtualGPUOptions.display

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if a display addapter backed by a vGPU should be enabled or disabled on the guest. Defaults to true.
`ramFB`	`object`	Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

.spec.gpus[].virtualGPUOptions.display.ramFB

Description

Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.hostDevices

Description

Optionally defines any HostDevices associated with the instancetype.

Type

`array`

.spec.hostDevices[]

Type

`object`

Required

`name`

Property	Type	Description
`claimName`	`string`	ClaimName needs to be provided from the list vmi.spec.resourceClaims[].name where this device is allocated

Property	Type	Description
`deviceName`	`string`	DeviceName is the name of the device provisioned by device-plugins
`name`	`string`	
`requestName`	`string`	RequestName needs to be provided from resourceClaim.spec.devices.requests[].name where this device is requested
`tag`	`string`	If specified, the virtual network interface address and its tag will be provided to the guest via config drive

.spec.ioThreads

Description

Optionally specifies the IOThreads options to be used by the instancetype.

Type

`object`

Property	Type	Description
`supplementalPoolThreadCount`	`integer`	SupplementalPoolThreadCount specifies how many iothreads are allocated for the supplementalPool policy.

.spec.launchSecurity

Description

Optionally defines the LaunchSecurity to be used by the instancetype.

Type

`object`

Property	Type	Description
`sev`	`object`	AMD Secure Encrypted Virtualization (SEV).
`snp`	`object`	AMD SEV-SNP flags defined by the SEV-SNP specifications.
`tdx`	`object`	Intel Trust Domain Extensions (TDX).

.spec.launchSecurity.sev

Description

AMD Secure Encrypted Virtualization (SEV).

Type

`object`

Property	Type	Description
`attestation`	`object`	If specified, run the attestation process for a vmi.
`dhCert`	`string`	Base64 encoded guest owner's Diffie-Hellman key.

Property	Type	Description
`policy`	`object`	Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.
`session`	`string`	Base64 encoded session blob.

.spec.launchSecurity.sev.attestation

Description

If specified, run the attestation process for a vmi.

Type

`object`

.spec.launchSecurity.sev.policy

Description

Guest policy flags as defined in AMD SEV API specification. Note: due to security reasons it is not allowed to enable guest debugging. Therefore NoDebug flag is not exposed to users and is always true.

Type

`object`

Property	Type	Description
`encryptedState`	`boolean`	SEV-ES is required. Defaults to false.

.spec.launchSecurity.snp

Description

AMD SEV-SNP flags defined by the SEV-SNP specifications.

Type

`object`

.spec.launchSecurity.tdx

Description

Intel Trust Domain Extensions (TDX).

Type

`object`

.spec.memory

Description

Required Memory related attributes of the instancetype.

Type

`object`

Required

`guest`

Property	Type	Description
`guest`	`integer|string`	Required amount of memory which is visible inside the guest OS.

Property	Type	Description
`hugepages`	`object`	Optionally enables the use of hugepages for the VirtualMachineInstance instead of regular memory.
`maxGuest`	`integer|string`	MaxGuest allows to specify the maximum amount of memory which is visible inside the Guest OS. The delta between MaxGuest and Guest is the amount of memory that can be hot(un)plugged.
`overcommitPercent`	`integer`	OvercommitPercent is the percentage of the guest memory which will be overcommitted. This means that the VMIs parent pod (virt-launcher) will request less physical memory by a factor specified by the OvercommitPercent. Overcommits can lead to memory exhaustion, which in turn can lead to crashes. Use carefully. Defaults to 0

.spec.memory.hugepages

Description

Optionally enables the use of hugepages for the VirtualMachineInstance instead of regular memory.

Type

`object`

Property	Type	Description
`pageSize`	`string`	PageSize specifies the hugepage size, for x86_64 architecture valid values are 1Gi and 2Mi.

.spec.nodeSelector

Description

NodeSelector is a selector which must be true for the vmi to fit on a node. Selector which must match a node's labels for the vmi to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/assign-pod-node/ NodeSelector is the name of the custom node selector for the instancetype.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterinstancetypes`
  - `DELETE`: delete collection of VirtualMachineClusterInstancetype
  - `GET`: list objects of kind VirtualMachineClusterInstancetype
  - `POST`: create a new VirtualMachineClusterInstancetype
- `/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterinstancetypes/{name}`
  - `DELETE`: delete the specified VirtualMachineClusterInstancetype
  - `GET`: read the specified VirtualMachineClusterInstancetype
  - `PATCH`: partially update the specified VirtualMachineClusterInstancetype
  - `PUT`: replace the specified VirtualMachineClusterInstancetype

/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterinstancetypes

HTTP method

`DELETE`

Description

delete collection of VirtualMachineClusterInstancetype

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineClusterInstancetype

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterInstancetypeList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineClusterInstancetype

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClusterInstancetype` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterInstancetype` schema
201 - Created	`VirtualMachineClusterInstancetype` schema

HTTP code	Response body
202 - Accepted	`VirtualMachineClusterInstancetype` schema
401 - Unauthorized	Empty

/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterinstancetypes/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineClusterInstancetype

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineClusterInstancetype

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterInstancetype` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineClusterInstancetype

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterInstancetype` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineClusterInstancetype

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClusterInstancetype` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterInstancetype` schema
201 - Created	`VirtualMachineClusterInstancetype` schema
401 - Unauthorized	Empty

VirtualMachineClusterPreference [virtualmachineclusterpreferences.instancetype.kubevirt.io/v1beta1]

Description

VirtualMachineClusterPreference is a cluster scoped version of the VirtualMachinePreference resource.

Type

`object`

Required

`spec`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	Required spec describing the preferences

.spec

Description

Required spec describing the preferences

Type

`object`

Property	Type	Description
`annotations`	`object`	Optionally defines preferred Annotations to be applied to the VirtualMachineInstance
`clock`	`object`	Clock optionally defines preferences associated with the Clock attribute of a VirtualMachineInstance DomainSpec
`cpu`	`object`	CPU optionally defines preferences associated with the CPU attribute of a VirtualMachineInstance DomainSpec
`devices`	`object`	Devices optionally defines preferences associated with the Devices attribute of a VirtualMachineInstance DomainSpec
`features`	`object`	Features optionally defines preferences associated with the Features attribute of a VirtualMachineInstance DomainSpec

Property	Type	Description
`firmware`	`object`	Firmware optionally defines preferences associated with the Firmware attribute of a VirtualMachineInstance DomainSpec
`machine`	`object`	Machine optionally defines preferences associated with the Machine attribute of a VirtualMachineInstance DomainSpec
`preferSpreadSocketToCoreRatio`	`integer`	PreferSpreadSocketToCoreRati defines the ratio to spread vCPUs between cores and sockets, it defaults to 2.
`preferredArchitecture`	`string`	PreferredArchitecture defines a prefeerred architecture for the VirtualMachine
`preferredSubdomain`	`string`	Subdomain of the VirtualMachineInstance
`preferredTerminationGracePeriodSeconds`	`integer`	Grace period observed after signalling a VirtualMachineInstance to stop

Property	Type	Description
		after which the VirtualMachineInstance is force terminated.
`requirements`	`object`	Requirements defines the minium amount of instance type defined resources required by a set of preferences
`volumes`	`object`	Volumes optionally defines preferences associated with the Volumes attribute of a VirtualMachineInstace DomainSpec

.spec.annotations

Description

Optionally defines preferred Annotations to be applied to the VirtualMachineInstance

Type

`object`

.spec.clock

Description

Clock optionally defines preferences associated with the Clock attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredClockOffset`	`object`	ClockOffset allows specifying the UTC offset or the timezone of the guest clock.
`preferredTimer`	`object`	Timer specifies whih timers are attached to the vmi.

.spec.clock.preferredClockOffset

Description

ClockOffset allows specifying the UTC offset or the timezone of the guest clock.

Type

`object`

Property	Type	Description
`timezone`	`string`	Timezone sets the guest clock to the specified timezone. Zone name follows the TZ environment variable format (e.g. 'America/New_York').
`utc`	`object`	UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

.spec.clock.preferredClockOffset.utc

Description

UTC sets the guest clock to UTC on each boot. If an offset is specified, guest changes to the clock will be kept during reboots and are not reset.

Type

`object`

Property	Type	Description
`offsetSeconds`	`integer`	OffsetSeconds specifies an offset in seconds, relative to UTC. If set, guest changes to the clock will be kept during reboots and not reset.

.spec.clock.preferredTimer

Description

Timer specifies whih timers are attached to the vmi.

Type

`object`

Property	Type	Description
`hpet`	`object`	HPET (High Precision Event Timer) - multiple timers with periodic interrupts.
`hyperv`	`object`	Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.
`kvm`	`object`	KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.

Property	Type	Description
`pit`	`object`	PIT (Programmable Interval Timer) - a timer with periodic interrupts.
`rtc`	`object`	RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

.spec.clock.preferredTimer.hpet

Description

HPET (High Precision Event Timer) - multiple timers with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "merge", "discard".

.spec.clock.preferredTimer.hyperv

Description

Hyperv (Hypervclock) - lets guests read the host’s wall clock time (paravirtualized). For windows guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.clock.preferredTimer.kvm

Description

KVM (KVM clock) - lets guests read the host’s wall clock time (paravirtualized). For linux guests.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.

.spec.clock.preferredTimer.pit

Description

PIT (Programmable Interval Timer) - a timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup", "discard".

.spec.clock.preferredTimer.rtc

Description

RTC (Real Time Clock) - a continuously running timer with periodic interrupts.

Type

`object`

Property	Type	Description
`present`	`boolean`	Enabled set to false makes sure that the machine type or a preset can't add the timer. Defaults to true.
`tickPolicy`	`string`	TickPolicy determines what happens when QEMU misses a deadline for injecting a tick to the guest. One of "delay", "catchup".
`track`	`string`	Track the guest or the wall clock.

.spec.cpu

Description

CPU optionally defines preferences associated with the CPU attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredCPUFeatures`	`array`	PreferredCPUFeatures optionally defines a slice of preferred CPU features.
`preferredCPUTopology`	`string`	PreferredCPUTopology optionally defines the preferred guest visible CPU topology, defaults to PreferSockets.
`spreadOptions`	`object`	

.spec.cpu.preferredCPUFeatures

Description

PreferredCPUFeatures optionally defines a slice of preferred CPU features.

Type

`array`

.spec.cpu.preferredCPUFeatures[]

Description

CPUFeature allows specifying a CPU feature.

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	Name of the CPU feature
`policy`	`string`	Policy is the CPU feature attribute which can have the following attributes: force - The virtual CPU will claim the feature is supported regardless of it being supported by host CPU. require - Guest creation will fail unless the feature is supported by the host CPU or the hypervisor is able to emulate it. optional - The feature will be supported by virtual CPU if and only if it is supported by host CPU. disable - The feature will not be supported by virtual CPU. forbid - Guest creation will fail if the feature is supported by host CPU. Defaults to require

.spec.cpu.spreadOptions

Type

`object`

Property	Type	Description
`across`	`string`	Across optionally defines how to spread vCPUs across the guest visible topology. Default: SocketsCores

Property	Type	Description
`ratio`	`integer`	Ratio optionally defines the ratio to spread vCPUs across the guest visible topology: CoresThreads - 1:2 - Controls the ratio of cores to threads. Only a ratio of 2 is currently accepted. SocketsCores - 1:N - Controls the ratio of socket to cores. SocketsCoresThreads - 1:N:2 - Controls the ratio of socket to cores. Each core providing 2 threads. Default: 2

.spec.devices

Description

Devices optionally defines preferences associated with the Devices attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredAutoattachGraphicsDevice`	`boolean`	PreferredAutoattachGraphicsDevice optionally defines the preferred valu of AutoattachGraphicsDevice
`preferredAutoattachInputDevice`	`boolean`	PreferredAutoattachInputDevice optionally defines the preferred valu of AutoattachInputDevice

Property	Type	Description
`preferredAutoattachMemBalloon`	`boolean`	PreferredAutoattachMemBalloon optionally defines the preferred valu of AutoattachMemBalloon
`preferredAutoattachPodInterface`	`boolean`	PreferredAutoattachPodInterface optionally defines the preferred valu of AutoattachPodInterface
`preferredAutoattachSerialConsole`	`boolean`	PreferredAutoattachSerialConsole optionally defines the preferred valu of AutoattachSerialConsole
`preferredBlockMultiQueue`	`boolean`	PreferredBlockMultiQueue optionall enables the vhost multiqueue featu for virtio disks.
`preferredCdromBus`	`string`	PreferredCdromBus optionally defines the preferred bus for Cdrom Disk devices.
`preferredDisableHotplug`	`boolean`	PreferredDisableHotplug optionally defines the preferred value of DisableHotplug

Property	Type	Description
`preferredDiskBlockSize`	`object`	PreferredBlockSize optionally defin the block size of Disk devices.
`preferredDiskBus`	`string`	PreferredDiskBus optionally defines the preferred bus for Disk Disk devices.
`preferredDiskCache`	`string`	PreferredCache optionally defines t DriverCache to be used by Disk devices.
`preferredDiskDedicatedIoThread`	`boolean`	PreferredDedicatedIoThread optionally enables dedicated IO threads for Disk devices using the virtio bus.
`preferredDiskIO`	`string`	PreferredIo optionally defines the QEMU disk IO mode to be used by Disk devices.
`preferredInputBus`	`string`	PreferredInputBus optionally define the preferred bus for Input devices.
`preferredInputType`	`string`	PreferredInputType optionally defin the preferred type for Input devices.

Property	Type	Description
`preferredInterfaceMasquerade`	`object`	PreferredInterfaceMasquerade optionally defines the preferred masquerade configuration to use w each network interface.
`preferredInterfaceModel`	`string`	PreferredInterfaceModel optionally defines the preferred model to be used by Interface devices.
`preferredLunBus`	`string`	PreferredLunBus optionally defines the preferred bus for Lun Disk devices.
`preferredNetworkInterfaceMultiQueue`	`boolean`	PreferredNetworkInterfaceMultiQue optionally enables the vhost multiqueue feature for virtio interfaces.
`preferredPanicDeviceModel`	`string`	PreferredPanicDeviceModel optionally defines the preferred pan device model to use with panic devices.
`preferredRng`	`object`	PreferredRng optionally defines the preferred rng device to be used.

Property	Type	Description
`preferredSoundModel`	`string`	PreferredSoundModel optionally defines the preferred model for Sound devices.
`preferredTPM`	`object`	PreferredTPM optionally defines the preferred TPM device to be used.
`preferredUseVirtioTransitional`	`boolean`	PreferredUseVirtioTransitional optionally defines the preferred value of UseVirtioTransitional
`preferredVirtualGPUOptions`	`object`	PreferredVirtualGPUOptions optionally defines the preferred value of VirtualGPUOptions

.spec.devices.preferredDiskBlockSize

Description

PreferredBlockSize optionally defines the block size of Disk devices.

Type

`object`

Property	Type	Description
`custom`	`object`	CustomBlockSize represents the desired logical and physical block size for a VM disk.

Property	Type	Description
`matchVolume`	`object`	Represents if a feature is enabled or disabled.

.spec.devices.preferredDiskBlockSize.custom

Description

CustomBlockSize represents the desired logical and physical block size for a VM disk.

Type

`object`

Property	Type	Description
`discardGranularity`	`integer`	
`logical`	`integer`	
`physical`	`integer`	

.spec.devices.preferredDiskBlockSize.matchVolume

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.devices.preferredInterfaceMasquerade

Description

PreferredInterfaceMasquerade optionally defines the preferred masquerade configuration to use with each network interface.

Type

`object`

.spec.devices.preferredRng

Description

PreferredRng optionally defines the preferred rng device to be used.

Type

`object`

.spec.devices.preferredTPM

Description

PreferredTPM optionally defines the preferred TPM device to be used.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled allows a user to explicitly disable the vTPM even when one is enabled by a preference referenced by the VirtualMachine Defaults to True
`persistent`	`boolean`	Persistent indicates the state of the TPM device should be kept accross reboots Defaults to false

.spec.devices.preferredVirtualGPUOptions

Description

PreferredVirtualGPUOptions optionally defines the preferred value of VirtualGPUOptions

Type

`object`

Property	Type	Description
`display`	`object`	

.spec.devices.preferredVirtualGPUOptions.display

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if a display addapter backed by a vGPU should be enabled or disabled on the guest. Defaults to true.
`ramFB`	`object`	Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

.spec.devices.preferredVirtualGPUOptions.display.ramFB

Description

Enables a boot framebuffer, until the guest OS loads a real GPU driver Defaults to true.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features

Description

Features optionally defines preferences associated with the Features attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredAcpi`	`object`	PreferredAcpi optionally enables the ACPI feature
`preferredApic`	`object`	PreferredApic optionally enables and configures the APIC feature
`preferredHyperv`	`object`	PreferredHyperv optionally enables and configures HyperV features
`preferredKvm`	`object`	PreferredKvm optionally enables and configures KVM features

Property	Type	Description
`preferredPvspinlock`	`object`	PreferredPvspinlock optionally enables the Pvspinlock feature
`preferredSmm`	`object`	PreferredSmm optionally enables the SMM feature

.spec.features.preferredAcpi

Description

PreferredAcpi optionally enables the ACPI feature

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredApic

Description

PreferredApic optionally enables and configures the APIC feature

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`endOfInterrupt`	`boolean`	EndOfInterrupt enables the end of interrupt notification in the guest. Defaults to false.

.spec.features.preferredHyperv

Description

PreferredHyperv optionally enables and configures HyperV features

Type

`object`

Property	Type	Description
`evmcs`	`object`	EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.
`frequencies`	`object`	Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.
`ipi`	`object`	IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Property	Type	Description
`reenlightenment`	`object`	Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.
`relaxed`	`object`	Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.
`reset`	`object`	Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.
`runtime`	`object`	Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.
`spinlocks`	`object`	Spinlocks allows to configure the spinlock retry attempts.
`synic`	`object`	SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.
`synictimer`	`object`	SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.

Property	Type	Description
`tlbflush`	`object`	TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.
`vapic`	`object`	VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.
`vendorid`	`object`	VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.
`vpindex`	`object`	VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

.spec.features.preferredHyperv.evmcs

Description

EVMCS Speeds up L2 vmexits, but disables other virtualization features. Requires vapic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.frequencies

Description

Frequencies improves the TSC clock source handling for Hyper-V on KVM. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.ipi

Description

IPI improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.reenlightenment

Description

Reenlightenment enables the notifications on TSC frequency changes. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.relaxed

Description

Relaxed instructs the guest OS to disable watchdog timeouts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.reset

Description

Reset enables Hyperv reboot/reset for the vmi. Requires synic. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.runtime

Description

Runtime improves the time accounting to improve scheduling in the guest. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.spinlocks

Description

Spinlocks allows to configure the spinlock retry attempts.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

Property	Type	Description
`spinlocks`	`integer`	Retries indicates the number of retries. Must be a value greater or equal 4096. Defaults to 4096.

.spec.features.preferredHyperv.synic

Description

SyNIC enables the Synthetic Interrupt Controller. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.synictimer

Description

SyNICTimer enables Synthetic Interrupt Controller Timers, reducing CPU load. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`direct`	`object`	Represents if a feature is enabled or disabled.

Property	Type	Description
`enabled`	`boolean`	

.spec.features.preferredHyperv.synictimer.direct

Description

Represents if a feature is enabled or disabled.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.tlbflush

Description

TLBFlush improves performances in overcommited environments. Requires vpindex. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.vapic

Description

VAPIC improves the paravirtualized handling of interrupts. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredHyperv.vendorid

Description

VendorID allows setting the hypervisor vendor id. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.
`vendorid`	`string`	VendorID sets the hypervisor vendor id, visible to the vmi. String up to twelve characters.

.spec.features.preferredHyperv.vpindex

Description

VPIndex enables the Virtual Processor Index to help windows identifying virtual processors. Defaults to the machine type setting.

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredKvm

Description

PreferredKvm optionally enables and configures KVM features

Type

`object`

Property	Type	Description
`hidden`	`boolean`	Hide the KVM hypervisor from standard MSR based discovery. Defaults to false

.spec.features.preferredPvspinlock

Description

PreferredPvspinlock optionally enables the Pvspinlock feature

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.features.preferredSmm

Description

PreferredSmm optionally enables the SMM feature

Type

`object`

Property	Type	Description
`enabled`	`boolean`	Enabled determines if the feature should be enabled or disabled on the guest. Defaults to true.

.spec.firmware

Description

Firmware optionally defines preferences associated with the Firmware attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredEfi`	`object`	PreferredEfi optionally enables EFI

Property	Type	Description
`preferredUseBios`	`boolean`	PreferredUseBios optionally enables BIOS
`preferredUseBiosSerial`	`boolean`	PreferredUseBiosSerial optionally transmitts BIOS output over the serial. Requires PreferredUseBios to be enabled.
`preferredUseEfi`	`boolean`	PreferredUseEfi optionally enables EFI Deprecated: Will be removed with v1beta2 or v1
`preferredUseSecureBoot`	`boolean`	PreferredUseSecureBoot optionally enables SecureBoot and the OVMF roms will be swapped for SecureBoot-enabled ones. Requires PreferredUseEfi and PreferredSmm to be enabled. Deprecated: Will be removed with v1beta2 or v1

.spec.firmware.preferredEfi

Description

PreferredEfi optionally enables EFI

Type

`object`

Property	Type	Description
`persistent`	`boolean`	If set to true, Persistent will persist the EFI NVRAM across reboots. Defaults to false
`secureBoot`	`boolean`	If set, SecureBoot will be enabled and the OVMF roms will be swapped for SecureBoot-enabled ones. Requires SMM to be enabled. Defaults to true

.spec.machine

Description

Machine optionally defines preferences associated with the Machine attribute of a VirtualMachineInstance DomainSpec

Type

`object`

Property	Type	Description
`preferredMachineType`	`string`	PreferredMachineType optionally defines the preferred machine type to use.

.spec.requirements

Description

Requirements defines the minium amount of instance type defined resources required by a set of preferences

Type

`object`

Property	Type	Description
`architecture`	`string`	Required Architecture of the VM referencing this preference
`cpu`	`object`	Required CPU related attributes of the instancetype.
`memory`	`object`	Required Memory related attributes of the instancetype.

.spec.requirements.cpu

Description

Required CPU related attributes of the instancetype.

Type

`object`

Required

`guest`

Property	Type	Description
`guest`	`integer`	Minimal number of vCPUs required by the preference.

.spec.requirements.memory

Description

Required Memory related attributes of the instancetype.

Type

`object`

Required

`guest`

Property	Type	Description
`guest`	`integer|string`	Minimal amount of memory required by the preference.

.spec.volumes

Description

Volumes optionally defines preferences associated with the Volumes attribute of a VirtualMachineInstace DomainSpec

Type

`object`

Property	Type	Description
`preferredStorageClassName`	`string`	PreffereedStorageClassName optionally defines the preferred storageClass

API Endpoints

The following API endpoints are available:

- `/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterpreferences`
  - `DELETE`: delete collection of VirtualMachineClusterPreference
  - `GET`: list objects of kind VirtualMachineClusterPreference
  - `POST`: create a new VirtualMachineClusterPreference

- `/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterpreferences/{name}`
  - `DELETE`: delete the specified VirtualMachineClusterPreference
  - `GET`: read the specified VirtualMachineClusterPreference
  - `PATCH`: partially update the specified VirtualMachineClusterPreference
  - `PUT`: replace the specified VirtualMachineClusterPreference

/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterpreferences

HTTP method

`DELETE`

Description

delete collection of VirtualMachineClusterPreference

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind VirtualMachineClusterPreference

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterPreferenceList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VirtualMachineClusterPreference

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClusterPreference` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterPreference` schema
201 - Created	`VirtualMachineClusterPreference` schema
202 - Accepted	`VirtualMachineClusterPreference` schema
401 - Unauthorized	Empty

/apis/instancetype.kubevirt.io/v1beta1/virtualmachineclusterpreferences/{name}

HTTP method

`DELETE`

Description

delete the specified VirtualMachineClusterPreference

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified VirtualMachineClusterPreference

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterPreference` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VirtualMachineClusterPreference

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterPreference` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VirtualMachineClusterPreference

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VirtualMachineClusterPreference` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VirtualMachineClusterPreference` schema
201 - Created	`VirtualMachineClusterPreference` schema
401 - Unauthorized	Empty