Workbench APIs

Workspace Kind [kubeflow.org/

Workspace [kubeflow.org/v1beta1]

Workspace Kind [kubeflow.org/v1beta1]

`kubeflow.org` group

WorkspaceKind is the Schema for the WorkspaceKinds API

`v1beta1` version

▼ **spec** `object`

WorkspaceKindSpec defines the desired state of WorkspaceKind

▼ **podTemplate** `object` required

podTemplate is the PodTemplate used to spawn Pods to run Workspaces of this WorkspaceKind

▼ **containerSecurityContext** `object`

container security context for Workspace Pods (MUTABLE)

▼ **allowPrivilegeEscalation** `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is:

1. run as Privileged
2. has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.

▼ **appArmorProfile** `object`

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

▼ **localhostProfile** `string`

localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".

▼ **type** `string` required

type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

▼ **capabilities** `object`

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **privileged** `boolean`

Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when spec.os.name is windows.

▼ **procMount** `string`

procMount denotes the type of proc mount to use for the containers. The default value is Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.

▼ **readOnlyRootFilesystem** `boolean`

Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.

▼ **runAsGroup** `integer`

The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ **runAsNonRoot** `boolean`

Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ runAsUser `integer`

The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ seLinuxOptions `object`

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ level `string`

Level is SELinux level label that applies to the container.

▼ role `string`

Role is a SELinux role label that applies to the container.

▼ type `string`

Type is a SELinux type label that applies to the container.

▼ **user** `string`

User is a SELinux user label that applies to the container.

▼ **seccompProfile** `object`

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

▼ **localhostProfile** `string`

localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.

▼ **type** `string` required

type indicates which kind of seccomp profile will be applied. Valid options are:

Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

▼ **windowsOptions** `object`

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in

SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

▼ **gmsaCredentialSpec** `string`

GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName field.

▼ **gmsaCredentialSpecName** `string`

GMSACredentialSpecName is the name of the GMSA credential spec to use.

▼ **hostProcess** `boolean`

HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.

▼ **runAsUserName** `string`

The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ **culling** `object`

culling configs for pausing inactive Workspaces (MUTABLE)

▼ **activityProbe** `object` required

the probe used to determine if the Workspace is active

▼ **exec** `object`

a shell command probe

- if the Workspace had activity in the last 60 seconds this command should return status 0, otherwise it should return status 1

▼ **command** `[]string` required

the command to run

▼ **jupyter** `object`

a Jupyter-specific probe

- will poll the `/api/status` endpoint of the Jupyter API, and use the `last_activity` field
- note, users need to be careful that their other probes don't trigger a "last_activity" update e.g. they should only check the health of Jupyter using the `/api/status` endpoint

▼ **lastActivity** `boolean` required

if the Jupyter-specific probe is enabled

▼ **enabled** `boolean`

if the culling feature is enabled

▼ **maxInactiveSeconds** `integer`

the maximum number of seconds a Workspace can be inactive

▼ **extraEnv** `[]object`

EnvVar represents an environment variable present in a Container.

▼ **name** `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ **value** `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ **valueFrom** `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ **configMapKeyRef** `object`

Selects a key of a ConfigMap.

▼ **key** `string` required

The key to select.

▼ **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 its key must be defined

▼ **fieldRef** `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

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

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

Selects a key of a secret in the pod's namespace

▼ **key** `string` required

The key of the secret to select from. Must be a valid secret key.

▼ **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 ↗](https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names)

▼ **optional** `boolean`

Specify whether the Secret or its key must be defined

▼ **extraVolumeMounts** `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ **mountPath** `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ **mountPropagation** `string`

mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10. When RecursiveReadOnly is set to IfPossible or to Enabled, MountPropagation must be None or unspecified (which defaults to None).

▼ name `string` required

This must match the Name of a Volume.

▼ readOnly `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.

▼ recursiveReadOnly `string`

RecursiveReadOnly specifies whether read-only mounts should be handled recursively.

If ReadOnly is false, this field has no meaning and must be unspecified.

If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason.

If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None).

If this field is not specified, it is treated as an equivalent of Disabled.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references $(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

▼ extraVolumes `[]object`

Volume represents a named volume in a pod that may be accessed by any container in the pod.

▼ awsElasticBlockStore `object`

awsElasticBlockStore represents an AWS Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ fsType `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ partition `integer`

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty).

▼ readOnly `boolean`

readOnly value true will force the readOnly setting in VolumeMounts. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ **volumeID** `string` required

volumeID is unique ID of the persistent disk resource in AWS (Amazon EBS volume). More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ **azureDisk** `object`

azureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.

▼ **cachingMode** `string`

cachingMode is the Host Caching mode: None, Read Only, Read Write.

▼ **diskName** `string` required

diskName is the Name of the data disk in the blob storage

▼ **diskURI** `string` required

diskURI is the URI of data disk in the blob storage

▼ **fsType** `string`

fsType is Filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **kind** `string`

kind expected values are Shared: multiple blob disks per storage account Dedicated: single blob disk per storage account Managed: azure managed data disk (only in managed availability set). defaults to shared

▼ **readOnly** `boolean`

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **azureFile** `object`

azureFile represents an Azure File Service mount on the host and bind mount to the pod.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretName** `string` required

secretName is the name of secret that contains Azure Storage Account Name and Key

▼ **shareName** `string` required

shareName is the azure share Name

▼ **cephfs** `object`

cephFS represents a Ceph FS mount on the host that shares a pod's lifetime

▼ **monitors** `[]string` required

monitors is Required: Monitors is a collection of Ceph monitors More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

▼ **path** `string`

path is Optional: Used as the mounted root, rather than the full Ceph tree, default is /

▼ **readOnly** `boolean`

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

▼ **secretFile** `string`

secretFile is Optional: SecretFile is the path to key ring for User, default is /etc/ceph/user.secret More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

▼ **secretRef** `object`

secretRef is Optional: SecretRef is reference to the authentication secret for User, default is empty. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

▼ **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

▼ **user** `string`

user is optional: User is the rados user name, default is admin More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

▼ **cinder** `object`

cinder represents a cinder volume attached and mounted on kubelets host machine. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

▼ **secretRef** `object`

secretRef is optional: points to a secret object containing parameters used to connect to OpenStack.

▼ **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

▼ **volumeID** `string` required

volumeID used to identify the volume in cinder. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

▼ **configMap** `object`

configMap represents a configMap that should populate this volume

▼ **defaultMode** `integer`

defaultMode is optional: mode bits used to set permissions on created files by default. 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. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is 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 of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **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`

optional specify whether the ConfigMap or its keys must be defined

▼ **csi** `object`

csi (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers (Beta feature).

▼ **driver** `string` required

driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.

▼ **fsType** `string`

fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.

▼ **nodePublishSecretRef** `object`

nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

▼ **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 ↗

▼ **readOnly** `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ **volumeAttributes** `object`

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

▼ **downwardAPI** `object`

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

▼ **defaultMode** `integer`

Optional: mode bits to use on created files by default. Must be a Optional: mode bits used to set permissions on created files by default. 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. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **items** `[]object`

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ **fieldRef** `object`

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

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **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

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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **emptyDir** `object`

emptyDir represents a temporary directory that shares a pod's lifetime. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

▼ **medium** `string`

medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

▼ **sizeLimit**

sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

▼ **ephemeral** `object`

ephemeral represents a volume that is handled by a cluster storage driver. The volume's lifecycle is tied to the pod that defines it - it will be created before the pod starts, and deleted when the pod is removed.

Use this if: a) the volume is only needed while the pod runs, b) features of normal volumes like restoring from snapshot or capacity tracking are needed, c) the storage driver is specified through a storage class, and d) the storage driver supports dynamic volume provisioning through a PersistentVolumeClaim (see EphemeralVolumeSource for more information on the connection between this volume type and PersistentVolumeClaim).

Use PersistentVolumeClaim or one of the vendor-specific APIs for volumes that persist for longer than the lifecycle of an individual pod.

Use CSI for light-weight local ephemeral volumes if the CSI driver is meant to be used that way - see the documentation of the driver for more information.

A pod can use both types of ephemeral volumes and persistent volumes at the same time.

▼ **volumeClaimTemplate** `object`

Will be used to create a stand-alone PVC to provision the volume. The pod in which this EphemeralVolumeSource is embedded will be the owner of the PVC, i.e. the PVC will be deleted together with the pod. The name of the PVC will be `<pod name>-<volume name>` where `<volume name>` is the name from the `PodSpec.Volumes` array entry. Pod validation will reject the pod if the concatenated name is not valid for a PVC (for example, too long).

An existing PVC with that name that is not owned by the pod will *not* be used for the pod to avoid using an unrelated volume by mistake. Starting the pod is then blocked until the unrelated PVC is removed. If such a pre-created PVC is meant to be used by the pod, the PVC has to updated with an owner reference to the pod once the pod exists. Normally this should not be necessary, but it may be useful when manually reconstructing a broken cluster.

This field is read-only and no changes will be made by Kubernetes to the PVC after it has been created.

Required, must not be nil.

▼ **metadata** `object`

May contain labels and annotations that will be copied into the PVC when creating it. No other fields are allowed and will be rejected during validation.

▼ **spec** `object` required

The specification for the PersistentVolumeClaim. The entire content is copied unchanged into the PVC that gets created from this template. The same fields as in a PersistentVolumeClaim are also valid here.

▼ **accessModes** `[]string`

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 copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

▼ **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` required

Kind is the type of resource being referenced

▼ **name** `string` required

Name is the name of resource being referenced

▼ **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.

▼ **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` required

Kind is the type of resource being referenced

▼ **name** `string` required

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.

▼ **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 ↗

▼ **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/ ↗

▼ **selector** `object`

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

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values.

Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ 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.

▼ 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 ↗](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 value means that no VolumeAttributesClass will be applied to the claim but it's not allowed to reset this field to empty string once it is set. If unspecified and the PersistentVolumeClaim is unbound, the default VolumeAttributesClass will be set by the persistentvolume controller if it exists. 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/ (Beta) Using this field requires the VolumeAttributesClass feature gate to be enabled (off by default).

▼ 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.

▼ **fc** `object`

fc represents a Fibre Channel resource that is attached to a kubelet's host machine and then exposed to the pod.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **lun** `integer`

lun is Optional: FC target lun number

▼ **readOnly** `boolean`

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **targetWWNs** `[]string`

targetWWNs is Optional: FC target worldwide names (WWNs)

▼ **wwids** `[]string`

wwids Optional: FC volume world wide identifiers (wwids) Either wwids or combination of targetWWNs and lun must be set, but not both simultaneously.

▼ **flexVolume** `object`

flexVolume represents a generic volume resource that is provisioned/attached using an exec based plugin.

▼ **driver** `string` required

driver is the name of the driver to use for this volume.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". The default filesystem depends on FlexVolume script.

▼ **options** `object`

options is Optional: this field holds extra command options if any.

▼ **readOnly** `boolean`

readOnly is Optional: defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretRef** `object`

secretRef is Optional: secretRef is reference to the secret object containing sensitive information to pass to the plugin scripts. This may be empty if no secret object is specified. If the secret object contains more than one secret, all secrets are passed to the plugin scripts.

▼ **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

▼ flocker `object`

flocker represents a Flocker volume attached to a kubelet's host machine. This depends on the Flocker control service being running

▼ datasetName `string`

datasetName is Name of the dataset stored as metadata -> name on the dataset for Flocker should be considered as deprecated

▼ datasetUUID `string`

datasetUUID is the UUID of the dataset. This is unique identifier of a Flocker dataset

▼ gcePersistentDisk `object`

gcePersistentDisk represents a GCE Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk

▼ fsType `string`

fsType is filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info:

https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ **partition** `integer`

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty). More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ **pdName** `string` required

pdName is unique name of the PD resource in GCE. Used to identify the disk in GCE. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ **gitRepo** `object`

gitRepo represents a git repository at a particular revision. DEPRECATED: GitRepo is deprecated. To provision a container with a git repo, mount an EmptyDir into an InitContainer that clones the repo using git, then mount the EmptyDir into the Pod's container.

▼ **directory** `string`

directory is the target directory name. Must not contain or start with '..'. If '.' is supplied, the volume directory will be the git repository. Otherwise, if specified, the volume will contain the git repository in the subdirectory with the given name.

▼ **repository** `string` required

repository is the URL

▼ **revision** `string`

revision is the commit hash for the specified revision.

▼ **glusterfs** `object`

glusterfs represents a Glusterfs mount on the host that shares a pod's lifetime. More info: https://examples.k8s.io/volumes/glusterfs/README.md ↗

▼ **endpoints** `string` required

endpoints is the endpoint name that details Glusterfs topology. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod ↗

▼ **path** `string` required

path is the Glusterfs volume path. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod ↗

▼ **readOnly** `boolean`

readOnly here will force the Glusterfs volume to be mounted with read-only permissions. Defaults to false. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod

▼ **hostPath** `object`

hostPath represents a pre-existing file or directory on the host machine that is directly exposed to the container. This is generally used for system agents or other privileged things that are allowed to see the host machine. Most containers will NOT need this. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

▼ **path** `string` required

path of the directory on the host. If the path is a symlink, it will follow the link to the real path. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

▼ **type** `string`

type for HostPath Volume Defaults to "" More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

▼ **image** `object`

image represents an OCI object (a container image or artifact) pulled and mounted on the kubelet's host machine. The volume is resolved at pod startup depending on which PullPolicy value is provided:

- Always: the kubelet always attempts to pull the reference. Container creation will fail If the pull fails.
- Never: the kubelet never pulls the reference and only uses a local image or artifact. Container creation will fail if the reference isn't present.
- IfNotPresent: the kubelet pulls if the reference isn't already present on disk. Container creation will fail if the reference isn't present and the pull fails.

The volume gets re-resolved if the pod gets deleted and recreated, which means that new remote content will become available on pod recreation. A failure to resolve or pull the image during pod startup will block containers from starting and may add significant latency. Failures will be retried using normal volume backoff and will be reported on the pod reason and message. The types of objects that may be mounted by this volume are defined by the container runtime implementation on a host machine and at minimum must include all valid types supported by the container image field. The OCI object gets mounted in a single directory (spec.containers[*].volumeMounts.mountPath) by merging the manifest layers in the same way as for container images. The volume will be mounted read-only (ro) and non-executable files (noexec). Sub path mounts for containers are not supported (spec.containers[*].volumeMounts.subpath). The field spec.securityContext.fsGroupChangePolicy has no effect on this volume type.

▼ **pullPolicy** `string`

Policy for pulling OCI objects. Possible values are: Always: the kubelet always attempts to pull the reference. Container creation will fail If the pull fails. Never: the kubelet never pulls the reference and only uses a local image or artifact. Container creation will fail if the reference isn't present. IfNotPresent: the kubelet pulls if the reference isn't already present on disk. Container creation will fail if the reference isn't present and the pull fails. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise.

▼ reference `string`

Required: Image or artifact reference to be used. Behaves in the same way as pod.spec.containers[*].image. Pull secrets will be assembled in the same way as for the container image by looking up node credentials, SA image pull secrets, and pod spec image pull secrets. More info: https://kubernetes.io/docs/concepts/containers/images This field is optional to allow higher level config management to default or override container images in workload controllers like Deployments and StatefulSets.

▼ iscsi `object`

iscsi represents an ISCSI Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://examples.k8s.io/volumes/iscsi/README.md

▼ chapAuthDiscovery `boolean`

chapAuthDiscovery defines whether support iSCSI Discovery CHAP authentication

▼ chapAuthSession `boolean`

chapAuthSession defines whether support iSCSI Session CHAP authentication

▼ fsType `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred

to be "ext4" if unspecified. More info:
https://kubernetes.io/docs/concepts/storage/volumes#iscsi

▼ **initiatorName** `string`

initiatorName is the custom iSCSI Initiator Name. If initiatorName is specified with iscsiInterface simultaneously, new iSCSI interface : will be created for the connection.

▼ **iqn** `string` required

iqn is the target iSCSI Qualified Name.

▼ **iscsiInterface** `string`

iscsiInterface is the interface Name that uses an iSCSI transport. Defaults to 'default' (tcp).

▼ **lun** `integer` required

lun represents iSCSI Target Lun number.

▼ **portals** `[]string`

portals is the iSCSI Target Portal List. The portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false.

▼ **secretRef** `object`

secretRef is the CHAP Secret for iSCSI target and initiator authentication

▼ **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

▼ **targetPortal** `string` required

targetPortal is iSCSI Target Portal. The Portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

▼ **name** `string` required

name of the volume. Must be a DNS_LABEL and unique within the pod. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

▼ **nfs** `object`

nfs represents an NFS mount on the host that shares a pod's lifetime More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs

▼ **path** `string` required

path that is exported by the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs

▼ **readOnly** `boolean`

readOnly here will force the NFS export to be mounted with read-only permissions. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs

▼ **server** `string` required

server is the hostname or IP address of the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs

▼ **persistentVolumeClaim** `object`

persistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims

▼ **claimName** `string` required

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.

▼ **photonPersistentDisk** `object`

photonPersistentDisk represents a PhotonController persistent disk attached and mounted on kubelets host machine

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **pdID** `string` required

pdID is the ID that identifies Photon Controller persistent disk

▼ **portworxVolume** `object`

portworxVolume represents a portworx volume attached and mounted on kubelets host machine

▼ **fsType** `string`

fSType represents the filesystem type to mount Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs". Implicitly inferred to be "ext4" if unspecified.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **volumeID** `string` required

volumeID uniquely identifies a Portworx volume

▼ **projected** `object`

projected items for all in one resources secrets, configmaps, and downward API

▼ **defaultMode** `integer`

defaultMode are the mode bits used to set permissions on created files by default. 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. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **sources** `[]object`

Projection that may be projected along with other supported volume types. Exactly one of these fields must be set.

▼ **clusterTrustBundle** `object`

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file.

Alpha, gated by the ClusterTrustBundleProjection feature gate.

ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector.

Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

▼ **labelSelector** `object`

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **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.

▼ **name** `string`

Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.

▼ **optional** `boolean`

If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.

▼ **path** `string` required

Relative path from the volume root to write the bundle.

▼ **signerName** `string`

Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all selected ClusterTrustBundles will be unified and deduplicated.

▼ **configMap** `object`

configMap information about the configMap data to project

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is 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 of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **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`

optional specify whether the ConfigMap or its keys must be defined

▼ **downwardAPI** `object`

downwardAPI information about the downwardAPI data to project

▼ **items** `[]object`

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ **fieldRef** `object`

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

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string`

required

Path of the field to select in the specified API version.

▼ **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

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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secret** `object`

secret information about the secret data to project

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is 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 of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **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`

optional field specify whether the Secret or its key must be defined

▼ **serviceAccountToken** `object`

serviceAccountToken is information about the serviceAccountToken data to project

▼ **audience** `string`

audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.

▼ **expirationSeconds** `integer`

expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token.

The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.

▼ **path** `string` required

path is the path relative to the mount point of the file to project the token into.

▼ **quobyte** `object`

quobyte represents a Quobyte mount on the host that shares a pod's lifetime

▼ **group** `string`

group to map volume access to Default is no group

▼ **readOnly** `boolean`

readOnly here will force the Quobyte volume to be mounted with read-only permissions. Defaults to false.

▼ **registry** `string` required

registry represents a single or multiple Quobyte Registry services specified as a string as host:port pair (multiple entries are separated with commas) which acts as the central registry for volumes

▼ **tenant** `string`

tenant owning the given Quobyte volume in the Backend Used with dynamically provisioned Quobyte volumes, value is set by the plugin

▼ **user** `string`

user to map volume access to Defaults to serivceaccount user

▼ **volume** `string` required

volume is a string that references an already created Quobyte volume by name.

▼ **rbd** `object`

rbd represents a Rados Block Device mount on the host that shares a pod's lifetime. More info: https://examples.k8s.io/volumes/rbd/README.md ↗

▼ **fsType** `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#rbd ↗

▼ **image** `string` required

image is the rados image name. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ **keyring** `string`

keyring is the path to key ring for RBDUser. Default is /etc/ceph/keyring. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

▼ **monitors** `[]string` required

monitors is a collection of Ceph monitors. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

▼ **pool** `string`

pool is the rados pool name. Default is rbd. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

▼ **secretRef** `object`

secretRef is name of the authentication secret for RBDUser. If provided overrides keyring. Default is nil. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

▼ **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

user `string`

user is the rados user name. Default is admin. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

scaleIO `object`

scaleIO represents a ScaleIO persistent volume attached and mounted on Kubernetes nodes.

fsType `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Default is "xfs".

gateway `string` required

gateway is the host address of the ScaleIO API Gateway.

protectionDomain `string`

protectionDomain is the name of the ScaleIO Protection Domain for the configured storage.

readOnly `boolean`

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretRef** `object` required

secretRef references to the secret for ScaleIO user and other sensitive information. If this is not provided, Login operation will fail.

▼ **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 ↗

▼ **sslEnabled** `boolean`

sslEnabled Flag enable/disable SSL communication with Gateway, default false

▼ **storageMode** `string`

storageMode indicates whether the storage for a volume should be ThickProvisioned or ThinProvisioned. Default is ThinProvisioned.

▼ **storagePool** `string`

storagePool is the ScaleIO Storage Pool associated with the protection domain.

▼ **system** `string` required

system is the name of the storage system as configured in ScaleIO.

▼ **volumeName** `string`

volumeName is the name of a volume already created in the ScaleIO system that is associated with this volume source.

▼ **secret** `object`

secret represents a secret that should populate this volume. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

▼ **defaultMode** `integer`

defaultMode is Optional: mode bits used to set permissions on created files by default. 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. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is 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 of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **optional** `boolean`

optional field specify whether the Secret or its keys must be defined

▼ **secretName** `string`

secretName is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

▼ **storageos** `object`

storageOS represents a StorageOS volume attached and mounted on Kubernetes nodes.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretRef** `object`

secretRef specifies the secret to use for obtaining the StorageOS API credentials. If not specified, default values will be attempted.

▼ **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 ↗

▼ **volumeName** `string`

volumeName is the human-readable name of the StorageOS volume. Volume names are only unique within a namespace.

▼ **volumeNamespace** `string`

volumeNamespace specifies the scope of the volume within StorageOS. If no namespace is specified then the Pod's namespace will be used. This allows the Kubernetes name scoping to be mirrored within StorageOS for tighter integration. Set VolumeName to any name to override the default behaviour. Set to "default" if you are not using namespaces within StorageOS. Namespaces that do not pre-exist within StorageOS will be created.

▼ vsphereVolume `object`

vsphereVolume represents a vSphere volume attached and mounted on kubelets host machine

▼ fsType `string`

fsType is filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ storagePolicyID `string`

storagePolicyID is the storage Policy Based Management (SPBM) profile ID associated with the StoragePolicyName.

▼ storagePolicyName `string`

storagePolicyName is the storage Policy Based Management (SPBM) profile name.

▼ volumePath `string` required

volumePath is the path that identifies vSphere volume vmdk

▼ httpProxy `object`

http proxy configs (MUTABLE)

▼ removePathPrefix `boolean`

if the path prefix is stripped from incoming HTTP requests

- if true, the '/workspace/{profile_name}/{workspace_name}/' path prefix is stripped from incoming requests, the application sees the request as if it was made to '/...'
- this only works if the application serves RELATIVE URLs for its assets

▼ requestHeaders `object`

header manipulation rules for incoming HTTP requests

- sets the `spec.http[].headers.request` of the Istio VirtualService https://istio.io/latest/docs/reference/config/networking/virtual-service/#Headers-HeaderOperations↗
- the following string templates are available:
  - `.PathPrefix`: the path prefix of the Workspace (e.g. '/workspace/{profile_name}/{workspace_name}/')

▼ add `object`

append the given values to the headers specified by keys (will create a comma-separated list of values)

▼ remove `[]string`

remove the specified headers

▼ set `object`

overwrite the headers specified by key with the given values

▼ **options** `object` required

options are the user-selectable fields, they determine the PodSpec of the Workspace

▼ **imageConfig** `object` required

imageConfig options

▼ **spawner** `object` required

spawner ui configs

▼ **default** `string` required

the id of the default option

- this will be selected by default in the spawner ui

▼ **values** `[]object` required

▼ **id** `string` required

the id of this image config

▼ **redirect** `object`

redirect configs

▼ **message** `object`

information about the redirect

▼ **level** `string` required

the importance level of the message

▼ **text** `string` required

the text of the message to show

▼ **to** `string` required

the id of the option to redirect to

▼ **spawner** `object` required

information for the spawner ui

▼ **description** `string`

a description of the option

▼ **displayName** `string` required

the display name of the option

▼ **hidden** `boolean`

if this option should be hidden from the Workspace Spawner UI

▼ **labels** `[]object`

▼ **key** `string` required

the key of the label

▼ **value** `string` required

the value of the label

▼ **spec** `object` required

the spec of the image config

▼ **image** `string` required

the container image to use

▼ **imagePullPolicy** `string`

the pull policy for the container image

▼ **ports** `[]object` required

▼ **displayName** `string` required

the display name of the port

▼ **id** `string` required

the id of the port

- this is NOT used as the Container or Service port name, but as part of the HTTP path

▼ **port** `integer` required

the port number

▼ **protocol** `string` required

the protocol of the port

▼ **podConfig** `object` required

podConfig options

▼ **spawner** `object` required

spawner ui configs

▼ **default** `string` required

the id of the default option

- this will be selected by default in the spawner ui

▼ **values** `[]object` required

▼ **id** `string` required

the id of this pod config

▼ **redirect** `object`

redirect configs

▼ **message** `object`

information about the redirect

▼ **level** `string` required

the importance level of the message

▼ **text** `string` required

the text of the message to show

▼ **to** `string` required

the id of the option to redirect to

▼ **spawner** `object` required

information for the spawner ui

▼ **description** `string`

a description of the option

▼ **displayName** `string` required

the display name of the option

▼ **hidden** `boolean`

if this option should be hidden from the Workspace Spawner UI

▼ **labels** `[]object`

▼ **key** `string` required

the key of the label

▼ **value** `string` required

the value of the label

▼ **spec** `object` required

the spec of the pod config

▼ **affinity** `object`

affinity configs for the pod

▼ **nodeAffinity** `object`

Describes node affinity scheduling rules for the pod.

▼ **preferredDuringSchedulingIgnoredDuringExecution**

`[]object`

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).

▼ **preference**

`object` required

A node selector term, associated with the corresponding weight.

▼ **matchExpressions**

`[]object`

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

▼ **key**

`string` required

The label key that the

selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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.

▼ **matchFields** `[]object`

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

▼ **key**
`string`
required

The label key that the selector applies to.

▼ **operator**
`string`
required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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.

▼ **weight** `integer` required

Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

▼ **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.

▼ **nodeSelectorTerms** `[]object` required

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.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

The label key that the selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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.

▼ **matchFields**

`[]object`

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

▼ **key** `string` required

The label key that the selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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.

▼ **podAffinity** `object`

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

▼ **preferredDuringSchedulingIgnoredDuringExecution** `[]object`

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

▼ **podAffinityTerm** `object` required

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

▼ **labelSelector** `object`

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

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label

key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relati

												onship to a set of values. Valid operators are In, NotIn, Exists and DoesNotEx												

ist
.

▼ **values** **`[]string`**

values is an array of string values. If the operator is In or N

												otIn, the values array must be non-empty. If the operator is Exists or DoesNot												

												Exist, the values array must be empty. This array is replaced during a strategic												

merge patch.

▼ **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.

▼ **matchLabelKeys** **`[]string`**

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **mismatchLabelKeys** `[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **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.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid op

er
at
or
s
ar
e
In,
N
otI
n,
Ex
ist
s
an
d
D
oe
s
N
ot
Ex
ist
.

▼ **values** **`[]string`**

va
lu
es
is
an

ar
ra
y
of
str
in
g
va
lu
es
. If
th
e
op
er
at
or
is
In
or
N
otI
n,
th
e
va
lu
es
ar
ra
y
m
us
t
be
no
n-

												empty. If the operator is Exists or DoesNotExist, the values array must be e												

m
pt
y.
Th
is
ar
ra
y
is
re
pl
ac
ed
du
rin
g
a
str
at
eg
ic
m
er
ge
pa
tc
h.

▼
matchLabels
`object`

matchLab
els 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.

▼ **namespaces** `[]string`

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` required

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.

▼ **weight** `integer`

required

weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

▼ **requiredDuringSchedulingIgnoredDuringExecution** `[]object`

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 matches that of any node on which a pod of the set of pods is running

▼ **labelSelector** `object`

A label query over a set of resources, in this case pods. If it's null, this

PodAffinityTerm matches with no Pods.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values.

Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ **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.

▼ **matchLabelKeys**

`[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffi nity feature gate (enabled by default).

▼ **mismatchLabelKeys** **`[]string`**

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **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.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **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.

▼ **namespaces**

`[]string`

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` required

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.

▼ **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)).

▼ **preferredDuringSchedulingIgnoredDuringExecution**

`[]object`

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

▼ podAffinityTerm

`object` required

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

▼ labelSelector

`object`

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

▼ matchExpressions

`[]object`

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

▼ **key** `string` required

key is the label key that the selector appli

es
to.

▼ **operator** `string` required

op
er
at
or
re
pr
es
en
ts
a
ke
y's
rel
ati
on
sh
ip
to
a
se
t
of
va
lu
es
.

Va lid op er at or s ar e In, N otI n, Ex ist s an d D oe s N ot Ex ist .

▼ **values** `[]string`

va lu

												es 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 mus												

t be e m pt y. Th is ar ra y is re pl ac ed du rin g a str at eg ic m er ge pa tc h.

▼ **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.

▼ **matchLabelKeys** `[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **mismatchLabelKeys** `[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **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.

▼ **matchExpressions** `[]object`

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

▼ **key** `str`

`in` `g` required

ke y is th e la be l ke y th at th e se le ct or ap pli es to.

▼ **operator** `s` `tr` `in` `g` required

op er

at
or
re
pr
es
en
ts
a
ke
y's
rel
ati
on
sh
ip
to
a
se
t
of
va
lu
es
.
Va
lid
op
er
at
or
s
ar
e
In,
N
otI
n,

Exists and DoesNotExist.

▼ **values** `[]string`

values is an array of string values

												. If the operator is In or NotIn, the values array must be non-empty. If the operat												

												or is Exists or DoesNotExist, the values array must be empty. This array is re												

pl
ac
ed
du
rin
g
a
str
at
eg
ic
m
er
ge
pa
tc
h.

▼ **matchLabels** `object`

matchLab
els is a
map of
{key,value
} pairs. A
single
{key,value
} in the
matchLab
els map is
equivalent
to an
element of
matchExpr

essions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

▼ **namespaces**

`[]string`

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` required

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.

▼ **weight** `integer` required

weight associated with matching the

corresponding podAffinityTerm, in the range 1-100.

▼ **requiredDuringSchedulingIgnoredDuringExecution** `[]object`

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 matches that of any node on which a pod of the set of pods is running

▼ **labelSelector** `object`

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

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **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.

▼ **matchLabelKeys** `[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼

mismatchLabelKeys `[]string`

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. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

▼ **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.

▼ **matchExpressions** `[]object`

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

▼ **key**

`string`

required

key is the label key that the selector applies to.

▼ **operator**

`string`

required

operator represents a key's relationshi p to a set of values. Valid operators are In, NotIn, Exists and

DoesNotExist.

▼ **values** **`[]string`**

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.

▼ 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.

▼ namespaces

`[]string`

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` required

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.

▼ **nodeSelector** `object`

node selector configs for the pod

▼ **resources** `object`

resource configs for the "main" container in the pod

▼ **claims** `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** `string` required

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.

▼ **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/

▼ **tolerations** `[]object`

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

▼ **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.

▼ **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.

▼ **podMetadata** `object`

metadata for Workspace Pods (MUTABLE)

▼ **annotations** `object`

annotations to be applied to the Pod resource

▼ **labels** `object`

labels to be applied to the Pod resource

▼ **probes** `object`

standard probes to determine Container health (MUTABLE)

▼ **livenessProbe** `object`

the liveness probe for the main container

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗).

If this is not specified, the default behavior is defined by gRPC.

▼ **httpGet** `object`

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-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 and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

▼ **readinessProbe** `object`

the readiness probe for the main container

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md↗).

If this is not specified, the default behavior is defined by gRPC.

▼ **httpGet** `object`

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-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 and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate.

Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ timeoutSeconds `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

▼ startupProbe `object`

the startup probe for the main container

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md).

If this is not specified, the default behavior is defined by gRPC.

▼ **httpGet** `object`

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-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 and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ timeoutSeconds `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

▼ securityContext `object`

security context for Workspace Pods (MUTABLE)

▼ appArmorProfile `object`

appArmorProfile is the AppArmor options to use by the containers in this pod. Note that this field cannot be set when spec.os.name is windows.

▼ localhostProfile `string`

localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".

▼ type `string` required

type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node.

RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

▼ fsGroup `integer`

A special supplemental group that applies to all containers in a pod. Some volume types allow the Kubelet to change the ownership of that volume to be owned by the pod:

1. The owning GID will be the FSGroup
2. The setgid bit is set (new files created in the volume will be owned by FSGroup)
3. The permission bits are OR'd with rw-rw----

If unset, the Kubelet will not modify the ownership and permissions of any volume. Note that this field cannot be set when spec.os.name is windows.

▼ fsGroupChangePolicy `string`

fsGroupChangePolicy defines behavior of changing ownership and permission of the volume before being exposed inside Pod. This field will only apply to volume types which support fsGroup based ownership(and permissions). It will have no effect on ephemeral volume types such as: secret, configmaps and emptydir. Valid values are "OnRootMismatch" and "Always". If not specified, "Always" is used. Note that this field cannot be set when spec.os.name is windows.

▼ runAsGroup `integer`

The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container. Note that this field cannot be set when spec.os.name is windows.

▼ **runAsNonRoot** `boolean`

Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ **runAsUser** `integer`

The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container. Note that this field cannot be set when spec.os.name is windows.

▼ **seLinuxOptions** `object`

The SELinux context to be applied to all containers. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in SecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence for that container. Note that this field cannot be set when spec.os.name is windows.

▼ **level** `string`

Level is SELinux level label that applies to the container.

▼ **role** `string`

Role is a SELinux role label that applies to the container.

▼ **type** `string`

Type is a SELinux type label that applies to the container.

▼ **user** `string`

User is a SELinux user label that applies to the container.

▼ **seccompProfile** `object`

The seccomp options to use by the containers in this pod. Note that this field cannot be set when spec.os.name is windows.

▼ **localhostProfile** `string`

localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.

▼ **type** `string` required

type indicates which kind of seccomp profile will be applied. Valid options are:

Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

▼ **supplementalGroups** `[]integer`

A list of groups applied to the first process run in each container, in addition to the container's primary GID and fsGroup (if specified). If the SupplementalGroupsPolicy feature is enabled, the supplementalGroupsPolicy field determines whether these are in addition to or instead of any group memberships defined in the container image. If unspecified, no additional groups are added, though group memberships defined in the container image may still be used, depending on the supplementalGroupsPolicy field. Note that this field cannot be set when spec.os.name is windows.

▼ **supplementalGroupsPolicy** `string`

Defines how supplemental groups of the first container processes are calculated. Valid values are "Merge" and "Strict". If not specified, "Merge" is used. (Alpha) Using the field requires the SupplementalGroupsPolicy feature gate to be enabled and the container runtime must implement support for this feature. Note that this field cannot be set when spec.os.name is windows.

▼ **sysctls** `[]object`

Sysctl defines a kernel parameter to be set

▼ **name** `string` required

Name of a property to set

▼ **value** `string` required

Value of a property to set

▼ **windowsOptions** `object`

The Windows specific settings applied to all containers. If unspecified, the options within a container's SecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

▼ **gmsaCredentialSpec** `string`

GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName field.

▼ **gmsaCredentialSpecName** `string`

GMSACredentialSpecName is the name of the GMSA credential spec to use.

▼ **hostProcess** `boolean`

HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.

▼ **runAsUserName** `string`

The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both

SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ **serviceAccount** `object` required

service account configs for Workspace Pods

▼ **name** `string` required

the name of the ServiceAccount (NOT MUTABLE)

- this Service Account MUST already exist in the Namespace of the Workspace, the controller will NOT create it
- we will not show this WorkspaceKind in the Spawner UI if the SA does not exist in the Namespace

▼ **volumeMounts** `object` required

volume mount paths

▼ **home** `string` required

the path to mount the home PVC (NOT MUTABLE)

▼ **spawner** `object` required

spawner config determines how the WorkspaceKind is displayed in the Workspace Spawner UI

▼ **deprecated** `boolean`

if this WorkspaceKind is deprecated

▼ deprecationMessage `string`

a message to show in Workspace Spawner UI when the WorkspaceKind is deprecated

▼ description `string` required

the description of the WorkspaceKind

▼ displayName `string` required

the display name of the WorkspaceKind

▼ hidden `boolean`

if this WorkspaceKind should be hidden from the Workspace Spawner UI

▼ icon `object` required

the icon of the WorkspaceKind

- a small (favicon-sized) icon used in the Workspace Spawner UI

▼ configMap `object`

▼ key `string` required

▼ name `string` required

▼ url `string`

▼ logo `object` required

the logo of the WorkspaceKind

- a 1:1 (card size) logo used in the Workspace Spawner UI

▼ configMap `object`

▼ key `string` required

▼ name `string` required

▼ url `string`

▼ status `object`

WorkspaceKindStatus defines the observed state of WorkspaceKind

▼ podTemplateOptions `object` required

metrics for podTemplate options

▼ imageConfig `[]object` required

▼ id `string` required

the id of the option

▼ workspaces `integer` required

the number of Workspaces currently using the option

▼ podConfig `[]object` required

▼ id `string` required

the id of the option

▼ workspaces `integer` required

the number of Workspaces currently using the option

▼ workspaces `integer` required

the number of Workspaces that are using this WorkspaceKind

Workspace [kubeflow.org/v1beta1]

`kubeflow.org` group

Workspace is the Schema for the Workspaces API

`v1beta1` version

▼ **spec** `object`

WorkspaceSpec defines the desired state of Workspace

▼ **deferUpdates** `boolean`

if true, pending updates are NOT applied when the Workspace is paused if false, pending updates are applied when the Workspace is paused

▼ **kind** `string` required

the WorkspaceKind to use

▼ **paused** `boolean`

if the workspace is paused (no pods running)

▼ **podTemplate** `object` required

options for "podTemplate"-type WorkspaceKinds

▼ **options** `object` required

the selected podTemplate options

▼ **imageConfig** `string` required

the id of an imageConfig option

- options are defined in WorkspaceKind under `spec.podTemplate.options.imageConfig.values[]`

▼ **podConfig** `string` required

the id of a podConfig option

- options are defined in WorkspaceKind under `spec.podTemplate.options.podConfig.values[]`

▼ **podMetadata** `object`

metadata to be applied to the Pod resource

▼ **annotations** `object`

annotations to be applied to the Pod resource

▼ **labels** `object`

labels to be applied to the Pod resource

▼ **volumes** `object` required

volume configs

▼ **data** `[]object`

▼ **mountPath** `string` required

the mount path for the PVC

▼ **pvcName** `string` required

the name of the PVC to mount

▼ **readOnly** `boolean`

if the PVC should be mounted as ReadOnly

▼ **home** `string`

the name of the PVC to mount as the home volume

- this PVC must already exist in the Namespace
- this PVC must be RWX (ReadWriteMany, ReadWriteOnce)
- the mount path is defined in the WorkspaceKind under `spec.podTemplate.volumeMounts.home`

▼ **status** `object`

WorkspaceStatus defines the observed state of Workspace

▼ **activity** `object` required

activity information for the Workspace, used to determine when to cull

▼ **lastActivity** `integer` required

the last time activity was observed on the Workspace (UNIX epoch)

▼ **lastUpdate** `integer` required

the last time we checked for activity on the Workspace (UNIX epoch)

▼ **pauseTime** `integer` required

the time when the Workspace was paused (UNIX epoch)

- set to 0 when the Workspace is NOT paused

▼ **pendingRestart** `boolean` required

if the current Pod does not reflect the current "desired" state

- true if any `spec.podTemplate.options` have a redirect and so will be patched on the next restart
- true if the WorkspaceKind has changed one of its common `podTemplate` fields like `podMetadata`, `probes`, `extraEnv`, or `containerSecurityContext`

▼ **podTemplateOptions** `object` required

information about the current podTemplate options (only set for WorkspaceKind of podTemplate kind)

▼ **imageConfig** `object` required

info about the current imageConfig option

▼ **desired** `string`

the option id which will take effect after the next restart

▼ **redirectChain** `[]object`

▼ **source** `string` required

the source option id

▼ **target** `string` required

the target option id

▼ **podConfig** `object` required

info about the current podConfig option

▼ **desired** `string`

the option id which will take effect after the next restart

▼ **redirectChain** `[]object`

▼ **source** `string` required

the source option id

▼ **target** `string` required

the target option id

▼ **podTemplatePod** `object` required

information about the Pod managed by this Workspace (only set for WorkspaceKind of podTemplate kind)

▼ **containers** `[]object`

▼ **name** `string` required

the name of the container

▼ **initContainers** `[]object`

▼ **name** `string` required

the name of the container

▼ **name** `string` required

the name of the Pod resource

▼ **state** `string` required

the current state of the Workspace

▼ **stateMessage** `string` required

a human-readable message about the state of the Workspace

- WARNING: this field is NOT FOR MACHINE USE, subject to change without notice