

触发器

触发器 [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1) TriggerTemplate [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1) EventListener [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1)

触发绑定 [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1) 拦截器 [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1) 集群触发器绑定 [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1)

集群拦截器 [\[triggers.tekton.dev/v1beta1\]](https://triggers.tekton.dev/v1beta1) ScheduledTrigger [\[tekton.alaudadevops.io/v1beta1\]](https://tekton.alaudadevops.io/v1beta1)

触发器 [triggers.tekton.dev/v1beta1]

triggers.tekton.dev group

Trigger defines a mapping of an input event to parameters. This is used to extract information from events to be passed to TriggerTemplates within a Trigger.

v1alpha1 version

▼ spec object

Spec holds the desired state of the Trigger

▼ bindings []object required

▼ apiversion string

APIVersion of the binding ref

▼ kind string

Kind can only be provided if Ref is also provided. Defaults to TriggerBinding

▼ name string

Name is the name of the binding param Mutually exclusive with Ref

▼ ref string

Ref is a reference to a TriggerBinding kind. Mutually exclusive with Name

▼ **value** `string`

Value is the value of the binding param. Can contain JSONPath Has to be pointer since "" is a valid value Required if Name is also specified.

▼ **interceptors** `[]object`

TriggerInterceptor provides a hook to intercept and pre-process events

▼ **bitbucket** `object`

BitbucketInterceptor provides a webhook to intercept and pre-process events

▼ **eventTypes** `[]string`

▼ **secretRef** `object`

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ **secretKey** `string`

▼ **secretName** `string`

▼ cel `object`

CELInterceptor provides a webhook to intercept and pre-process events

▼ filter `string`

▼ overlays `[]object`

CELOverlay provides a way to modify the request body using
DeprecatedCEL expressions

▼ expression `string`

▼ key `string`

▼ github `object`

Deprecated old fields below

▼ eventTypes `[]string`

▼ secretRef `object`

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ **secretKey** `string`

▼ **secretName** `string`

▼ **gitlab** `object`

GitLabInterceptor provides a webhook to intercept and pre-process events

▼ **eventTypes** `[]string`

▼ **secretRef** `object`

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ **secretKey** `string`

▼ **secretName** `string`

▼ **name** `string`

Optional name to identify the current interceptor configuration

▼ params **[]object**

InterceptorParams defines a key-value pair that can be passed on an interceptor

▼ name **string** **required****▼ value** **required****▼ ref** **object** **required**

Ref refers to the Interceptor to use

▼ apiVersion **string**

API version of the referent

▼ kind **string**

InterceptorKind indicates the kind of the Interceptor, namespaced or cluster scoped.

▼ name **string**

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ webhook **object**

WebhookInterceptor refers to an old style webhook interceptor service

▼ header `[]object`

Param declares an ParamValues to use for the parameter called name.

▼ name `string` `required`**▼ value** `required`**▼ objectRef** `object`

ObjectRef is a reference to an object that will resolve to a cluster DNS name to use as the EventInterceptor. Either objectRef or url can be specified

▼ apiVersion `string`

API version of the referent.

▼ fieldPath `string`

If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as `desiredState.manifest.containers[2]`. For example, if the object reference is to a container within a pod, this would take on a value like: `"spec.containers{name}"` (where "name" refers to the name of the container that triggered the event) or if no container name is specified `"spec.containers[2]"` (container with index 2 in this pod).

This syntax is chosen only to have some well-defined way of referencing a part of an object.

▼ kind `string`

Kind of the referent. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds> ↗

▼ **name** `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **namespace** `string`

Namespace of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/> ↗

▼ **resourceVersion** `string`

Specific resourceVersion to which this reference is made, if any.

More info: <https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency> ↗

▼ **uid** `string`

UID of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids> ↗

▼ **url** `string`

▼ **name** `string`

▼ **serviceAccountName** `string`

ServiceAccountName optionally associates credentials with each trigger; Unlike EventListeners, this should be scoped to the same namespace as the Trigger itself

▼ **template** `object` required

▼ **apiversion** `string`

▼ **ref** `string`

▼ **spec** `object`

TriggerTemplateSpec holds the desired state of TriggerTemplate

▼ **params** `[]object`

ParamSpec defines an arbitrary named input whose value can be supplied by a `Param`.

▼ **default** `string`

Default is the value a parameter takes if no input value via a Param is supplied.

▼ **description** `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ **name** **string** **required**

Name declares the name by which a parameter is referenced.

▼ **resourcetemplates** **[]object**

TriggerResourceTemplate describes a resource to create

Trigger defines a mapping of an input event to parameters. This is used to extract information from events to be passed to TriggerTemplates within a Trigger.

v1beta1

version

▼ **spec** **object**

Spec holds the desired state of the Trigger

▼ **bindings** **[]object** **required**

▼ **apiversion** **string**

APIVersion of the binding ref

▼ **kind** **string**

Kind can only be provided if Ref is also provided. Defaults to TriggerBinding

▼ **name** **string**

Name is the name of the binding param Mutually exclusive with Ref

▼ ref string

Ref is a reference to a TriggerBinding kind. Mutually exclusive with Name

▼ value string

Value is the value of the binding param. Can contain JSONPath Has to be pointer since "" is a valid value Required if Name is also specified.

▼ interceptors []object

TriggerInterceptor provides a hook to intercept and pre-process events

▼ name string

Optional name to identify the current interceptor configuration

▼ params []object

InterceptorParams defines a key-value pair that can be passed on an interceptor

▼ name string required

▼ value required

▼ ref object required

Ref refers to the Interceptor to use

▼ apiVersion string

API version of the referent

▼ kind string

InterceptorKind indicates the kind of the Interceptor, namespaced or cluster scoped.

▼ name string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ webhook object

WebhookInterceptor refers to an old style webhook interceptor service

▼ header []object

Param declares an ParamValues to use for the parameter called name.

▼ name string required**▼ value** required**▼ objectRef** object

ObjectRef is a reference to an object that will resolve to a cluster DNS name to use as the EventInterceptor. Either objectRef or url can be specified

▼ apiVersion `string`

API version of the referent.

▼ fieldPath `string`

If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as `desiredState.manifest.containers[2]`. For example, if the object reference is to a container within a pod, this would take on a value like: `"spec.containers{name}"` (where "name" refers to the name of the container that triggered the event) or if no container name is specified `"spec.containers[2]"` (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

▼ kind `string`

Kind of the referent. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds> ↗

▼ name `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ namespace `string`

Namespace of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/> ↗

▼ resourceVersion string

Specific resourceVersion to which this reference is made, if any.

More info: <https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency> ↗

▼ uid string

UID of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids> ↗

▼ url string**▼ name** string**▼ serviceAccountName** string

ServiceAccountName optionally associates credentials with each trigger; Unlike EventListeners, this should be scoped to the same namespace as the Trigger itself

▼ template object required**▼ apiversion** string**▼ ref** string

▼ spec **object**

TriggerTemplateSpec holds the desired state of TriggerTemplate

▼ params **[]object**

ParamSpec defines an arbitrary named input whose value can be supplied by a **Param**.

▼ default **string**

Default is the value a parameter takes if no input value via a Param is supplied.

▼ description **string**

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ name **string** *required*

Name declares the name by which a parameter is referenced.

▼ resourcetemplates **[]object**

TriggerResourceTemplate describes a resource to create



TriggerTemplate

[triggers.tekton.dev/v1beta1]

`triggers.tekton.dev``group`

TriggerTemplate takes parameters and uses them to create CRDs

`v1alpha1``version`

▼ spec `object`

Spec holds the desired state of the TriggerTemplate from the client

▼ params `[]object`

ParamSpec defines an arbitrary named input whose value can be supplied by a `Param`.

▼ default `string`

Default is the value a parameter takes if no input value via a Param is supplied.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ name `string` `required`

Name declares the name by which a parameter is referenced.

▼ resourcetemplates []object

TriggerResourceTemplate describes a resource to create

▼ status object

TriggerTemplateStatus describes the desired state of TriggerTemplate

TriggerTemplate takes parameters and uses them to create CRDs

v1beta1

version

▼ spec object

Spec holds the desired state of the TriggerTemplate from the client

▼ params []object

ParamSpec defines an arbitrary named input whose value can be supplied by a Param.

▼ default string

Default is the value a parameter takes if no input value via a Param is supplied.

▼ description string

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ name string required

Name declares the name by which a parameter is referenced.

▼ resourcetemplates []object

TriggerResourceTemplate describes a resource to create

▼ status object

TriggerTemplateStatus describes the desired state of TriggerTemplate

EventListener [triggers.tekton.dev/v1beta1]

triggers.tekton.dev group

EventListener exposes a service to accept HTTP event payloads.

v1alpha1 version

▼ spec object

Spec holds the desired state of the EventListener from the client

▼ labelSelector object

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

▼ 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.

▼ namespaceSelector `object`

NamespaceSelector is a selector for selecting either all namespaces or a list of namespaces.

▼ matchNames `[]string`

List of namespace names.

▼ resources `object`**▼ customResource** `object`**▼ kubernetesResource** `object`

▼ replicas **integer****▼ serviceType** **string**

Service Type string describes ingress methods for a service

▼ spec **object**

WithPodSpec is the shell around the PodSpecable within WithPod.

▼ template **object**

PodSpecable is implemented by types containing a PodTemplateSpec in the manner of ReplicaSet, Deployment, DaemonSet, StatefulSet.

▼ metadata **object**

Standard object's metadata. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata> ↗

▼ spec **object**

Specification of the desired behavior of the pod. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status> ↗

▼ activeDeadlineSeconds **integer**

Optional duration in seconds the pod may be active on the node relative to StartTime before the system will actively try to mark it failed and kill

associated containers. Value must be a positive integer.

▼ affinity **object**

If specified, the pod's scheduling constraints

▼ 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

g

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

g

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,
DoesNotE
xist. Gt,
and Lt.

▼ values

[]string

g

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
DoesNotE
xist, 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

t

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



key

s

tr

in
g

required

key
y
is
th
e
la
be
l
key
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
otl
n,

Exists and Does Not Exist.

▼
values

[
]s
tr
in
g

values is an array of string values

. If the operator is In or Not on, the values are array must be non-empty. If the operator at

or
is
Exist
ists
s
or
Does
oes
s
Not
ot
Exist
ist
,
th
e
valu
lu
es
ar
ra
y
m
us
t
be
e
m
pt
y.
Th
is
ar
ra
y
is
re

placed 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 matchExpr

essions,
whose key
field is
"key", the
operator is
"In", and
the values
array
contains
only
"value".
The
requireme
nts 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**t**

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

**key****s****tr****in****g**

required

key
y
is
th
e
la
be
l
ke
y
th

at the selector applies to.

▼
operator

s
tr
in
g

required

operator represents a key's relationship

to
a
se
t
of
va
lu
es
. Valid
lid
op
er
at
or
s
ar
e
In,
N
otl
n,
Ex
ist
s
an
d
D
oe
s
N
ot
Ex
ist
.



values

[
]s
tr
in
g

values
is
an
array
of
string
values
. If
the
operator
is
In
or
Not
on,
th

evaluates array of must-be non-empty y. If the operator is Exist or Does Not Exist,

these values are array must be empty. This array is replaced during ga strategic merge

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 requireme

nts 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 namespaceSelect or. null or empty namespaces list and null namespaceSelect or 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 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

t

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



key

s

tr

in

g

required

key is the label being

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

ist

.



values

[

]s

tr

in

g

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

otl
n,
th
e
va
lu
es
ar
ra
y
m
us
t
be
no
n-
e
m
pt
y.
If
th
e
op
er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot

Exist, the values are array must be empty. This array is replaced during gastrategic

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
requireme
nts 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` not 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 `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

s
tr
in
g

required

key
is
the
label
key
that
selects
or
applies
to.



operator

string

required

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

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

▼
values
[
]s
tr
in
g
va
lu
es
is
an

array of strings values. If the operator is In or NotIn, then the values are array must be non-

empty.
If the operator
is Exists
or Does
Not 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.

▼ 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 namespaceSelect or. null or empty namespaces list and null namespaceSelect or 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 relationship to a set

of values.

Valid
operators
are In,
NotIn,
Exists and
DoesNotExist.

▼ values

`[]string`
g

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.

▼ automountServiceAccountToken

boolean

AutomountServiceAccountToken indicates whether a service account token should be automatically mounted.

▼ containers []object required

A single application container that you want to run within a pod.

▼ args []string

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ command []string

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ env []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.](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces)

[io/docs/concepts/o](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces)

[verview/working-](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces)

[with-](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces)

[objects/names/#n](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces)

[ames](https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces) ↗

▼ 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> ↗

▼ optional**boolean**

Specify whether the Secret or its key must be defined

▼ envFrom []object

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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 must be defined

▼ prefix **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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 Secret must be defined

▼ image **string**

Container image name. 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.

▼ imagePullPolicy **string**

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise.

Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle **object**

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart **object**

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy.

Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET 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.

▼ sleep **object**

Sleep represents a duration that the container should sleep.

▼ seconds**integer**

required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop **object**

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will

eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET 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.

▼ **sleep** **object**

Sleep represents a duration that the container should sleep.

▼ **seconds**

integer

required

Seconds is the number of

seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ livenessProbe **object**

Periodic probe of container liveness.

Container will be restarted if the probe

fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to

execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ **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 an HTTP GET 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/concept
s/workloads/pods/pod-
lifecycle#container-probes ↗](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 a connection to 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> ↗

▼ **name** **string** required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ **ports** **[]object**

ContainerPort represents a network port in a single container.

▼ **containerPort** **integer**

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ **hostIP** **string**

What host IP to bind the external port to.

▼ **hostPort** **integer**

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ **name** `string`

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ **protocol** `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ **readinessProbe** `object`

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ **exec** `object`

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **resizePolicy** **[]object**

`ContainerResizePolicy` represents resource resize policy for the container.

▼ resourceName string

required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ restartPolicy string

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources object

Compute Resources required by this container. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

▼ 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/>

[s/configuration/manage-resources-containers/ ↗](#)

▼ restartPolicy string

RestartPolicy defines the restart behavior of individual containers in a pod. This field may only be set for init containers, and the only allowed value is "Always". For non-init containers or when this field is not specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext object

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override

the equivalent fields of

PodSecurityContext. More info:

<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/> ↗

▼ 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.

▼ startupProbe **object**

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state

operation. This cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ **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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ **terminationMessagePath**

string

Optional: Path at which the file to which the container's termination message will

be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to `/dev/termination-log`. Cannot be updated.

▼ **terminationMessagePolicy**

string

Indicate how the termination message should be populated. File will use the contents of `terminationMessagePath` to populate the container status message on both success and failure.

`FallbackToLogsOnError` will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to `File`. Cannot be updated.

▼ **tty** **boolean**

Whether this container should allocate a TTY for itself, also requires `'stdin'` to be true. Default is false.

▼ **volumeDevices** **[]object**

`volumeDevice` describes a mapping of a raw block device within a container.

▼ devicePath **string****required**

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts **[]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.

▼ **workingDir** `string`

Container's working directory. If not specified, the container runtime's default

will be used, which might be configured in the container image. Cannot be updated.

▼ dnsConfig **object**

Specifies the DNS parameters of a pod.

Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

▼ nameservers **[]string**

A list of DNS name server IP addresses.

This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

▼ options **[]object**

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name **string**

Name is this DNS resolver option's name. Required.

▼ value **string**

Value is this DNS resolver option's value.

▼ searches **[]string**

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

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirstWithHostNet', 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy. To have DNS options set along with hostNetwork, you have to specify DNS policy explicitly to 'ClusterFirstWithHostNet'.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ ephemeralContainers `[]object`

An EphemeralContainer is a temporary container that you may add to an existing Pod for user-initiated activities such as debugging. Ephemeral containers have no resource or scheduling guarantees, and they will not be restarted when they exit or when a Pod is removed or restarted. The kubelet may evict a Pod if an ephemeral

container causes the Pod to exceed its resource allocation.

To add an ephemeral container, use the `ephemeralcontainers` subresource of an existing Pod. Ephemeral containers may not be removed or restarted.

▼ args `[]string`

Arguments to the entrypoint. The image's CMD is used if this is not provided.

Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info:

<https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ command `[]string`

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ env []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> ↗

▼ optional**boolean**

Specify whether the Secret or its key must be defined

▼ envFrom []object

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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/workin-g-with-objects/names/#names> ↗

▼ optional **boolean**

Specify whether the ConfigMap must be defined

▼ prefix **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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 Secret must be defined

▼ image **string**

Container image name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ imagePullPolicy **string**

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle **object**

Lifecycle is not allowed for ephemeral containers.

▼ postStart **object**

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy.

Other management of the container blocks until the hook completes. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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.

▼ sleep **object**

Sleep represents a duration that the container should sleep.

▼ seconds**integer****required**

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop object

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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

E.

▼ scheme

string

Scheme to use for connecting to the host. Defaults to HTTP.

▼ sleep object

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket object

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle

hooks will fail at runtime when it is specified.

▼ 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.

▼ livenessProbe **object**

Probes are not allowed for ephemeral containers.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **name** **string** required

Name of the ephemeral container specified as a DNS_LABEL. This name must be

unique among all containers, init containers and ephemeral containers.

▼ ports []object

ContainerPort represents a network port in a single container.

▼ containerPort integer

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ hostIP string

What host IP to bind the external port to.

▼ hostPort integer

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name string

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name.

Name for the port that can be referred to by services.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

Probes are not allowed for ephemeral containers.

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ `resizePolicy` `[]object`

ContainerResizePolicy represents resource resize policy for the container.

▼ `resourceName` `string`

required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ `restartPolicy` `string`

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources **object**

Resources are not allowed for ephemeral containers. Ephemeral containers use spare resources already allocated to 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/> ↗

▼ restartPolicy **string**

Restart policy for the container to manage the restart behavior of each container within a pod. This may only be set for init containers. You cannot set this field on ephemeral containers.

▼ securityContext **object**

Optional: SecurityContext defines the security options the ephemeral container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext.

▼ 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.

▼ startupProbe **object**

Probes are not allowed for ephemeral containers.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the

container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ **targetContainerName** **string**

If set, the name of the container from PodSpec that this ephemeral container targets. The ephemeral container will be run in the namespaces (IPC, PID, etc) of this container. If not set then the ephemeral container uses the namespaces configured in the Pod spec.

The container runtime must implement support for this feature. If the runtime does not support namespace targeting then the result of setting this field is undefined.

▼ terminationMessagePath

string

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to `/dev/termination-log`. Cannot be updated.

▼ terminationMessagePolicy

string

Indicate how the termination message should be populated. File will use the contents of `terminationMessagePath` to populate the container status message on both success and failure.

`FallbackToLogsOnError` will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to `File`.

Cannot be updated.

▼ tty `boolean`

Whether this container should allocate a TTY for itself, also requires `'stdin'` to be `true`. Default is `false`.

▼ volumeDevices []object

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath string

required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []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.

▼ workingDir `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ hostAliases `[]object`

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

▼ hostnames `[]string`

Hostnames for the above IP address.

▼ ip `string` required

IP address of the host file entry.

▼ hostIPC `boolean`

Use the host's ipc namespace. Optional: Default to false.

▼ hostNetwork `boolean`

Host networking requested for this pod. Use the host's network namespace. If this option is set, the ports that will be used must be specified. Default to false.

▼ hostPID `boolean`

Use the host's pid namespace. Optional: Default to false.

▼ hostUsers `boolean`

Use the host's user namespace. Optional: Default to true. If set to true or not present, the pod will be run in the host user namespace, useful for when the pod needs a feature only available to the host user namespace, such as loading a kernel module with CAP_SYS_MODULE. When set to false, a new users is created for the pod. Setting false is useful for mitigating container breakout vulnerabilities even allowing users to run their containers as root without actually having root privileges on the host. This field is alpha-level and is only honored by servers that enable the UserNamespacesSupport feature.

▼ hostname `string`

Specifies the hostname of the Pod If not specified, the pod's hostname will be set to a system-defined value.

▼ imagePullSecrets `[]object`

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

▼ 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> ↗

▼ initContainers `[]object`

A single application container that you want to run within a pod.

▼ args `[]string`

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info:

<https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ **command** []string

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ **env** []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/o](https://kubernetes.io/docs/concepts/o)

verview/working-
with-
objects/names/#n
ames ↗

▼ optional

boolean

Specify whether
the Secret or its
key must be
defined

▼ envFrom **[]object**

EnvFromSource represents the source of
a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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/workin](https://kubernetes.io/docs/concepts/overview/workin)

[g-with-objects/names/#names ↗](#)

▼ **optional** **boolean**

Specify whether the ConfigMap must be defined

▼ **prefix** **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** **object**

The Secret to select from

▼ **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/workin>

[g-with-objects/names/#names ↗](#)

▼ optional `boolean`

Specify whether the Secret must be defined

▼ image `string`

Container image name. 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.

▼ imagePullPolicy `string`

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle `object`

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart `object`

PostStart is called immediately after a container is created. If the

handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET 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.

▼ sleep object

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket object

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop **object**

PreStop is called immediately before a container is terminated due to an API request or

management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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.

▼ **sleep** object

Sleep represents a duration that the container should sleep.

▼ **seconds**

integer

required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ **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.

▼ livenessProbe object

Periodic probe of container liveness.

Container will be restarted if the probe

fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec object

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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>

▼ **name** **string** required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ **ports** **[]object**

ContainerPort represents a network port in a single container.

▼ **containerPort** **integer**

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ hostIP **string**

What host IP to bind the external port to.

▼ hostPort **integer**

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name **string**

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ protocol **string**

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe **object**

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ **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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **resizePolicy** []object

ContainerResizePolicy represents resource resize policy for the container.

▼ **resourceName** string

required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ **restartPolicy** string

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ **resources** object

Compute Resources required by this container. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/> ↗

▼ **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/> ↗

▼ restartPolicy **string**

RestartPolicy defines the restart behavior of individual containers in a pod. This field may only be set for init containers, and the only allowed value is "Always". For non-init containers or when this field is not specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next

init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext **object**

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/> ↗

▼ 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.

▼ startupProbe object

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If `stdinOnce` is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is

restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath

string

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.

▼ terminationMessagePolicy

string

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure.

FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ tty boolean

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices []object

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath string

required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []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.

▼ **workingDir** `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ **nodeName** `string`

`nodeName` indicates in which node this pod is scheduled. If empty, this pod is a candidate for scheduling by the scheduler defined in `schedulerName`. Once this field is set, the kubelet for this node becomes responsible for the lifecycle of this pod. This field should not be used to express a desire for the pod to be scheduled on a specific node.

<https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#nodename> ↗

▼ **nodeSelector** `object`

`NodeSelector` is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info:

<https://kubernetes.io/docs/concepts/configuration/assign-pod-node/> ↗

▼ os object

Specifies the OS of the containers in the pod.

Some pod and container fields are restricted if this is set.

If the OS field is set to linux, the following fields must be unset: -securityContext.windowsOptions

If the OS field is set to windows, following fields must be unset:

- spec.hostPID
- spec.hostIPC
- spec.hostUsers
- spec.securityContext.appArmorProfile
- spec.securityContext.seLinuxOptions
- spec.securityContext.seccompProfile
- spec.securityContext.fsGroup
- spec.securityContext.fsGroupChangePolicy
- spec.securityContext.sysctls
- spec.shareProcessNamespace
- spec.securityContext.runAsUser
- spec.securityContext.runAsGroup
- spec.securityContext.supplementalGroups
- spec.securityContext.supplementalGroupsPolicy
- spec.containers[*].securityContext.appArmorProfile
- spec.containers[*].securityContext.seLinuxOptions

- `spec.containers[*].securityContext.seccompProfile`
- `spec.containers[*].securityContext.capabilities`
- `spec.containers[*].securityContext.readOnlyRootFilesystem`
- `spec.containers[*].securityContext.privileged`
- `spec.containers[*].securityContext.allowPrivilegeEscalation`
- `spec.containers[*].securityContext.procMount`
- `spec.containers[*].securityContext.runAsUser`
- `spec.containers[*].securityContext.runAsGroup`

▼ **name** `string` required

Name is the name of the operating system.

The currently supported values are linux and windows. Additional value may be defined in future and can be one of:

<https://github.com/opencontainers/runtime-spec/blob/master/config.md#platform-specific-configuration> ↗ Clients should expect to handle additional values and treat unrecognized values in this field as os: null

▼ **overhead** `object`

Overhead represents the resource overhead associated with running a pod for a given

RuntimeClass. This field will be autopopulated at admission time by the RuntimeClass admission controller. If the RuntimeClass admission controller is enabled, overhead must not be set in Pod create requests. The RuntimeClass admission controller

will reject Pod create requests which have the overhead already set. If RuntimeClass is configured and selected in the PodSpec, Overhead will be set to the value defined in the corresponding RuntimeClass, otherwise it will remain unset and treated as zero. More info: <https://git.k8s.io/enhancements/keps/sig-node/688-pod-overhead/README.md> ↗

▼ **preemptionPolicy** `string`

PreemptionPolicy is the Policy for preempting pods with lower priority. One of Never, PreemptLowerPriority. Defaults to PreemptLowerPriority if unset.

▼ **priority** `integer`

The priority value. Various system components use this field to find the priority of the pod. When Priority Admission Controller is enabled, it prevents users from setting this field. The admission controller populates this field from PriorityClassName. The higher the value, the higher the priority.

▼ **priorityClassName** `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified,

the pod priority will be default or zero if there is no default.

▼ readinessGates `[]object`

PodReadinessGate contains the reference to a pod condition

▼ conditionType `string` required

ConditionType refers to a condition in the pod's condition list with matching type.

▼ resourceClaims `[]object`

PodResourceClaim references exactly one ResourceClaim, either directly or by naming a ResourceClaimTemplate which is then turned into a ResourceClaim for the pod.

It adds a name to it that uniquely identifies the ResourceClaim inside the Pod. Containers that need access to the ResourceClaim reference it with this name.

▼ name `string` required

Name uniquely identifies this resource claim inside the pod. This must be a DNS_LABEL.

▼ resourceClaimName `string`

ResourceClaimName is the name of a ResourceClaim object in the same

namespace as this pod.

Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

▼ resourceClaimTemplateName

string

ResourceClaimTemplateName is the name of a ResourceClaimTemplate object in the same namespace as this pod.

The template will be used to create a new ResourceClaim, which will be bound to this pod. When this pod is deleted, the ResourceClaim will also be deleted. The pod name and resource name, along with a generated component, will be used to form a unique name for the ResourceClaim, which will be recorded in pod.status.resourceClaimStatuses.

This field is immutable and no changes will be made to the corresponding ResourceClaim by the control plane after creating the ResourceClaim.

Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

▼ resources object

Resources is the total amount of CPU and Memory resources required by all containers in the pod. It supports specifying Requests and Limits for "cpu"

and "memory" resource names only.

ResourceClaims are not supported.

This field enables fine-grained control over resource allocation for the entire pod, allowing resource sharing among containers in a pod.

This is an alpha field and requires enabling the PodLevelResources feature gate.

▼ 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/> ↗

▼ restartPolicy **string**

Restart policy for all containers within the pod. One of Always, OnFailure, Never. In some contexts, only a subset of those values may be permitted.

Default to Always. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/#restart-policy> ↗

▼ runtimeClassName **string**

RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit class with an empty definition that uses the default runtime handler.

More info:

<https://git.k8s.io/enhancements/keps/sig-node/585-runtime-class> ↗

▼ schedulerName `string`

If specified, the pod will be dispatched by specified scheduler. If not specified, the pod will be dispatched by default scheduler.

▼ schedulingGates `[]object`

PodSchedulingGate is associated to a Pod to guard its scheduling.

▼ name `string` required

Name of the scheduling gate. Each scheduling gate must have a unique name field.

▼ securityContext `object`

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field.

▼ 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.

▼ seLinuxChangePolicy string

seLinuxChangePolicy defines how the container's SELinux label is applied to all volumes used by the Pod. It has no effect on nodes that do not support SELinux or to volumes does not support SELinux. Valid values are "MountOption" and "Recursive". "Recursive" means relabeling of all files on all Pod volumes by the container runtime. This may be slow for large volumes, but allows mixing privileged and unprivileged Pods sharing the same volume on the same node.

"MountOption" mounts all eligible Pod volumes with `-o context` mount option. This requires all Pods that share the same volume to use the same SELinux label. It is not possible to share the same volume among privileged and unprivileged Pods. Eligible volumes are in-tree FibreChannel and iSCSI volumes, and all CSI volumes whose CSI driver announces SELinux support by setting `spec.selinuxMount: true` in their CSIDriver instance. Other volumes are always re-labelled recursively.

"MountOption" value is allowed only when SELinuxMount feature gate is enabled.

If not specified and SELinuxMount feature gate is enabled, "MountOption" is used. If not specified and SELinuxMount feature gate is disabled, "MountOption" is used for ReadWriteOncePod volumes and "Recursive" for all other volumes.

This field affects only Pods that have SELinux label set, either in PodSecurityContext or in SecurityContext of all containers.

All Pods that use the same volume should use the same `seLinuxChangePolicy`, otherwise some pods can get stuck in ContainerCreating state. 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.

▼ sysctl []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 string

DeprecatedServiceAccount is a deprecated alias for ServiceAccountName. Deprecated: Use serviceAccountName instead.

▼ serviceAccountName string

ServiceAccountName is the name of the ServiceAccount to use to run this pod. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/> ↗

▼ setHostnameAsFQDN boolean

If true the pod's hostname will be configured as the pod's FQDN, rather than the leaf name (the default). In Linux containers, this means setting the FQDN in the hostname field of the kernel (the nodename field of struct utsname). In Windows containers, this means setting the registry value of hostname for the registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters to FQDN. If a pod does not have FQDN, this has no effect. Default to false.

▼ shareProcessNamespace boolean

Share a single process namespace between all of the containers in a pod. When this is set containers will be able to view and signal processes from other containers in the same pod, and the first process in each container will not be assigned PID 1. HostPID and ShareProcessNamespace cannot both be set. Optional: Default to false.

▼ subdomain string

If specified, the fully qualified Pod hostname will be "...svc.". If not specified, the pod will not have a domainname at all.

▼ terminationGracePeriodSeconds

integer

Optional duration in seconds the pod needs to terminate gracefully. May be decreased in delete request. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). If this value is nil, the default grace period will be used instead. 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. Defaults to 30 seconds.

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ labelSelector **object**

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

▼ 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 the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be

ignored. A null or empty list means only match against labelSelector.

This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

▼ maxSkew integer required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. | zone1 | zone2 | zone3 | | P P | P P | P |

- if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1).
- if MaxSkew is 2, incoming pod can be scheduled onto any zone. When `whenUnsatisfiable=ScheduleAnyway`, it is used to give higher precedence to topologies that satisfy it. It's a

required field. Default value is 1 and 0 is not allowed.

▼ minDomains integer

MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule.

For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: | zone1 | zone2 | zone3 | | P P | P P | P P | The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is

scheduled to any of the three zones, it will violate MaxSkew.

▼ nodeAffinityPolicy `string`

NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are:

- Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations.
- Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations.

If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.

▼ nodeTaintsPolicy `string`

NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are:

- Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included.
- Ignore: node taints are ignored. All nodes are included.

If this value is nil, the behavior is equivalent to the Ignore policy. This is a

beta-level feature default enabled by the `NodeInclusionPolicyInPodTopologySpread` feature flag.

▼ **topologyKey** `string` required

`TopologyKey` is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each `<key, value>` as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of `nodeAffinityPolicy` and `nodeTaintsPolicy`. e.g. If `TopologyKey` is `"kubernetes.io/hostname"`, each Node is a domain of that topology. And, if `TopologyKey` is `"topology.kubernetes.io/zone"`, each zone is a domain of that topology. It's a required field.

▼ **whenUnsatisfiable** `string` required

`WhenUnsatisfiable` indicates how to deal with a pod if it doesn't satisfy the spread constraint.

- `DoNotSchedule` (default) tells the scheduler not to schedule it.
- `ScheduleAnyway` tells the scheduler to schedule the pod in any location, but

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

▼ volumes []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. Deprecated: AWSElasticBlockStore is deprecated. All operations for the in-tree awsElasticBlockStore type are redirected to the ebs.csi.aws.com CSI driver. 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> ↗

[s/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. Deprecated: AzureDisk is deprecated. All operations for the in-tree azureDisk type are redirected to the disk.csi.azure.com CSI driver.

▼ **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. Deprecated: AzureFile is deprecated. All operations for the in-tree azureFile type are redirected to the file.csi.azure.com CSI driver.

▼ **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. Deprecated: CephFS is deprecated and the in-tree cephfs type is no longer supported.

▼ 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>

[ephfs/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/c>

[ephfs/README.md#how-to-use-it](#)



▼ cinder **object**

cinder represents a cinder volume attached and mounted on kubelets host machine. Deprecated: Cinder is deprecated. All operations for the in-tree cinder type are redirected to the [cinder.csi.openstack.org](https://examples.k8s.io/mysql-cinder-pd/README.md) CSI driver. 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.

▼ 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/e/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 be 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 PersistentVolume Claim 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

Namespac
e is the
namespac
e of
resource
being
referenced

Note that
when a
namespac
e is
specified,
a
gateway.n
etworking.
k8s.io/Ref
erenceGra
nt object is
required in
the
referent
namespac
e to allow
that
namespac
e's owner
to accept
the
reference.

See the Reference Grant 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/management-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://kub
ernetes.io/
docs/conc
epts/confi
guration/m
anage-
resources-](https://kubernetes.io/docs/concepts/configuration/management-resources-)

containers

/

▼ selector

object

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



matchExpressions

[]objec

t

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



key

s

trigger

required

key is the label key that the selector or applies to.



operator

string

required

operator relationship to a set of values.
Valid operators are:
In, N

otl

n,

Ex

ist

s

an

d

D

oe

s

N

ot

Ex

ist

.



values

[

]s

tr

in

g

va

lu

es

is

an

ar

ra

y

of

str

in

g

va

values. If the operator is In or Not on, the values are array must be non-empty. If the op

er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot
Ex
ist
,
th
e
va
lu
es
ar
ra
y
m
us
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
matchExpr
essions,
whose key
field is
"key", the
operator is
"In", and
the values
array
contains
only
"value".
The
requireme
nts are
ANDed.



storageClassName

string

storageClassNam
e is the name of
the StorageClass
required by the
claim. More info:
[https://kubernetes.
io/docs/concepts/s
torage/persistent-
volumes#class-1](https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1)



volumeAttributesClassName

string

volumeAttributesClassName may be used to set the VolumeAttributesClassName used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClassName. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string value means that no VolumeAttributesClassName 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 VolumeAttributesClassName

lass will be set by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesC lass does not exist, this PersistentVolume Claim 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 VolumeAttributesC lass 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. Deprecated: FlexVolume is deprecated. Consider using a CSIDriver instead.

▼ 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. Deprecated: Flocker is deprecated and the in-tree flocker type is no longer supported.

▼ 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. Deprecated: GCEPersistentDisk is deprecated. All operations for the in-tree gcePersistentDisk type are redirected to the pd.csi.storage.gke.io CSI driver. 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>
sk ↗

▼ **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>
sk ↗

▼ **readOnly** **boolean**

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

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

▼ **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. **Deprecated:** Glusterfs is deprecated and the in-tree glusterfs type is no longer

supported. 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#persistent-volumeclaims>

▼ **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#persistent-volumeclaims>

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.
Deprecated: PhotonPersistentDisk is deprecated and the in-tree photonPersistentDisk type is no longer supported.

▼ 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. Deprecated:

PortworxVolume is deprecated. All operations for the in-tree portworxVolume type are redirected to the pxd.portworx.com CSI driver when the CSIMigrationPortworx feature-gate is on.

▼ 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

t

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



key

s

tr

in

g

required

key
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

presentations as key's relationship to a set of values.
Valid operators are In, NotIn, Exists

and does not exist.

▼
values

[
]s
tr
in
g

values is an array of string values. If the

operator is in order. Not only, the values are array must be non-empty. If the operator is Ex

ists, the values are arrays must be empty. This array is replaced

during strategic merger package.



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

e 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

0

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
annotation
s, labels,
name,
namespac
e and uid
are
supported.



apiVersion

s
tr
in
g

Ve
rsi
on
of
th
e
sc
he
m
a
th

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

▼
fieldPath
string

required

Path of the

efileld to select in the specified API version.

▼ mode

integer

r

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

s
tr
in
g

C
on
tai
ne
r
na
m
e:
re
qu
ire
d
for
vo
lu
m
es

,
op
tio
na
l
for
en
v
va
rs

▼
divisor

S
pe
cif
ie
s
th
e
ou
tp
ut
for
m
at
of
th
e
ex
po
se
d
re
so

ur
ce
s,
de
fa
ult
s
to
"1
"

▼
resource
s
tr
in
g

required

R
eq
uir
ed
:
re
so
ur
ce
to
se
le
ct

▼ 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

r

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.
Deprecated: Quobyte is deprecated and the in-tree quobyte type is no longer supported.

▼ 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 serviceaccount 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. Deprecated: RBD is deprecated and the in-tree rbd type is no longer supported. 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. Deprecated: ScaleIO is deprecated and the in-tree scaleIO type is no longer supported.

▼ 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/workin-g-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. Deprecated: StorageOS is deprecated and the in-tree storageos type is no longer supported.

▼ 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/workin-g-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. Deprecated: VsphereVolume is deprecated. All operations for the in-tree vsphereVolume type are redirected to the csi.vsphere.vmware.com CSI driver.

▼ 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

▼ **serviceAccountName** `string`

▼ **triggers** `[]object` `required`

EventListenerTrigger represents a connection between TriggerBinding, Params, and TriggerTemplate; TriggerBinding provides extracted values for TriggerTemplate to then

create resources from. TriggerRef can also be provided instead of TriggerBinding, Interceptors and TriggerTemplate

▼ bindings `[]object`

▼ apiVersion `string`

APIVersion of the binding ref

▼ kind `string`

Kind can only be provided if Ref is also provided. Defaults to TriggerBinding

▼ name `string`

Name is the name of the binding param Mutually exclusive with Ref

▼ ref `string`

Ref is a reference to a TriggerBinding kind. Mutually exclusive with Name

▼ value `string`

Value is the value of the binding param. Can contain JSONPath Has to be pointer since "" is a valid value Required if Name is also specified.

▼ interceptors `[]object`

TriggerInterceptor provides a hook to intercept and pre-process events

▼ bitbucket **object**

BitbucketInterceptor provides a webhook to intercept and pre-process events

▼ eventTypes **[]string****▼ secretRef** **object**

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ secretKey **string****▼ secretName** **string****▼ cel** **object**

CELInterceptor provides a webhook to intercept and pre-process events

▼ filter **string****▼ overlays** **[]object**

CELOverlay provides a way to modify the request body using DeprecatedCEL expressions

▼ **expression** `string`

▼ **key** `string`

▼ **github** `object`

Deprecated old fields below

▼ **eventTypes** `[]string`

▼ **secretRef** `object`

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ **secretKey** `string`

▼ **secretName** `string`

▼ **gitlab** `object`

GitLabInterceptor provides a webhook to intercept and pre-process events

▼ eventTypes `[]string`**▼ secretRef** `object`

SecretRef contains the information required to reference a single secret string This is needed because the other secretRef types are not cross-namespace and do not actually contain the "SecretName" field, which allows us to access a single secret value.

▼ secretKey `string`**▼ secretName** `string`**▼ name** `string`

Optional name to identify the current interceptor configuration

▼ params `[]object`

InterceptorParams defines a key-value pair that can be passed on an interceptor

▼ name `string` required**▼ value** required

▼ ref **object** required

Ref refers to the Interceptor to use

▼ apiVersion **string**

API version of the referent

▼ kind **string**

InterceptorKind indicates the kind of the Interceptor, namespaced or cluster scoped.

▼ name **string**

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names>

▼ webhook **object**

WebhookInterceptor refers to an old style webhook interceptor service

▼ header **[]object**

Param declares an ParamValues to use for the parameter called name.

▼ name **string** required**▼ value** required

▼ objectRef **object**

ObjectRef is a reference to an object that will resolve to a cluster DNS name to use as the EventInterceptor. Either objectRef or url can be specified

▼ apiVersion **string**

API version of the referent.

▼ fieldPath **string**

If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as `desiredState.manifest.containers[2]`. For example, if the object reference is to a container within a pod, this would take on a value like: `"spec.containers{name}"` (where "name" refers to the name of the container that triggered the event) or if no container name is specified `"spec.containers[2]"` (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

▼ kind **string**

Kind of the referent. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds> ↗

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ namespace **string**

Namespace of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/> ↗

▼ resourceVersion **string**

Specific resourceVersion to which this reference is made, if any. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency> ↗

▼ uid **string**

UID of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids> ↗

▼ url **string**

▼ name **string**

▼ serviceAccountName **string**

ServiceAccountName optionally associates credentials with each trigger; more granular authorization for who is allowed to utilize the associated pipeline vs. defaulting to whatever permissions are associated with the entire EventListener and associated sink facilitates multi-tenant model based scenarios

▼ template `object`

▼ apiversion `string`

▼ ref `string`

▼ spec `object`

TriggerTemplateSpec holds the desired state of TriggerTemplate

▼ params `[]object`

ParamSpec defines an arbitrary named input whose value can be supplied by a `Param`.

▼ default `string`

Default is the value a parameter takes if no input value via a `Param` is supplied.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ **resourcetemplates** **[]object**

TriggerResourceTemplate describes a resource to create

▼ **triggerRef** **string**

▼ **status** **object**

EventListenerStatus holds the status of the EventListener

▼ **address** **object**

Address is a single Addressable address. If Addresses is present, Address will be ignored by clients.

▼ **CACerts** **string**

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ **hostname** **string**

▼ **name** **string**

Name is the name of the address.

▼ url `string`**▼ addresses** `[]object`

Addressable provides a generic mechanism for a custom resource definition to indicate a destination for message delivery.

Addressable is the schema for the destination information. This is typically stored in the object's `status`, as this information may be generated by the controller.

▼ CACerts `string`

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ hostname `string`**▼ name** `string`

Name is the name of the address.

▼ url `string`**▼ annotations** `object`

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

▼ conditions []object

Condition defines a readiness condition for a Knative resource. See:

<https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties> ↗

▼ lastTransitionTime string

LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

▼ message string

A human readable message indicating details about the transition.

▼ reason string

The reason for the condition's last transition.

▼ severity string

Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

▼ status string required

Status of the condition, one of True, False, Unknown.

▼ type string required

Type of condition.

▼ configuration **object** required

Configuration stores configuration for the EventListener service

▼ generatedName **string** required

GeneratedResourceName is the name given to all resources reconciled by the EventListener

▼ observedGeneration **integer**

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

EventListener exposes a service to accept HTTP event payloads.

v1beta1 version**▼ spec** **object**

Spec holds the desired state of the EventListener from the client

▼ cloudEventURI **string****▼ labelSelector** **object**

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

▼ 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.

▼ **namespaceSelector** `object`

NamespaceSelector is a selector for selecting either all namespaces or a list of namespaces.

▼ **matchNames** `[]string`

List of namespace names.

▼ resources **object**

▼ customResource **object**

▼ kubernetesResource **object**

▼ replicas **integer**

▼ serviceLoadBalancerClass **string**

▼ servicePort **integer**

▼ serviceType **string**

Service Type string describes ingress methods for a service

▼ spec **object**

WithPodSpec is the shell around the PodSpecable within WithPod.

▼ template **object**

PodSpecable is implemented by types containing a PodTemplateSpec in the manner of ReplicaSet, Deployment, DaemonSet, StatefulSet.

▼ metadata **object**

Standard object's metadata. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata> ↗

▼ spec **object**

Specification of the desired behavior of the pod. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status> ↗

▼ activeDeadlineSeconds **integer**

Optional duration in seconds the pod may be active on the node relative to StartTime before the system will actively try to mark it failed and kill associated containers. Value must be a positive integer.

▼ affinity **object**

If specified, the pod's scheduling constraints

▼ 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

g

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,
DoesNotE
xist. Gt,
and Lt.

▼ values

[]string

g

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
DoesNotE
xist, 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

Represent
s a key's
relationshi
p to a set
of values.

Valid
operators
are In,
NotIn,
Exists,
DoesNotE
xist. Gt,
and Lt.

▼ values

[]string

g

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
t**

A label selector requirement is a selector

that
contains
values, a
key, and
an
operator
that
relates the
key and
values.



key

s

tr

in

g

required

key
y
is
th
e
la
be
l
key
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
otl
n,
Ex
ist
s
an
d
D
oe
s
N
ot
Ex
ist
.

▼
values
[
]
s
tr
in
g

values is an array of string values. If the operator is In or NotIn, then the values are array m

us
t
be
no
n-
e
m
pt
y.
If
th
e
op
er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot
Ex
ist
,
th
e
va
lu
es
ar
ra
y

must be empty. This array is replaced during gating strategy merge path.



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 MatchLabelKeysIn PodAffinity 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

t

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



key

s
tr
in
g

required

key is the label being used as the selector or applies to.



operator

s
tr
in
g

required

operator relationship to a set of values.
Valid operators are:
In, N

otl
n,
Ex
ist
s
an
d
D
oe
s
N
ot
Ex
ist
.

▼
values

[
]s
tr
in
g

va
lu
es
is
an
ar
ra
y
of
str
in
g
va

values. If the operator is In or Not on, the values are array must be non-empty. If the op

er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot
Ex
ist
,
th
e
va
lu
es
ar
ra
y
m
us
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
matchExpr
essions,
whose key
field is
"key", the
operator is
"In", and
the values
array
contains
only
"value".
The
requireme
nts 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
namespaceSelect
or. null or empty
namespaces list
and null

namespaceSelect
or 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

DoesNotE

xist.

▼ values

[]string

g

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 `keynotin (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
DoesNotE
xist.

▼ 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
DoesNotE
xist, 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

t

A label selector requirement is a selector that contains values, a key, and

an operator that relates the key and values.



key

s

tr

in

g

required

key

y

is

th

e

la

be

I

key

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
.

Valid operators are In, Not, Exists and Does Not Exist.

▼ values
[]s
tring
value
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
otl
n,
th
e
va
lu
es
ar
ra
y
m
us
t

be
no
n-
e
m
pt
y.
If
th
e
op
er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot
Ex
ist
,
th
e
va
lu
es
ar
ra
y
m
us

to 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 MatchLabelKeysIn PodAffinity 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

t

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



key

s

tr

in
g

required

key
y
is
th
e
la
be
l
key
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
otl
n,

Exists and Does Not Exist.

▼
values

[
]s
tr
in
g

values is an array of strings.

. If the operator is in or Not on, the values are array must be non-empty. If the operator at

or
is
Exist
ists
s
or
Does
oes
s
Not
ot
Exist
ist
,
th
e
valu
lu
es
ar
ra
y
m
us
t
be
e
m
pt
y.
Th
is
ar
ra
y
is
re

placed 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 matchExpr

essions,
whose key
field is
"key", the
operator is
"In", and
the values
array
contains
only
"value".
The
requireme
nts 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
namespaceSelect
or. null or empty
namespaces list
and null
namespaceSelect
or 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

g

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

DoesNotExist.
xist.

▼ 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.

▼ automountServiceAccountToken

boolean

AutomountServiceAccountToken indicates whether a service account token should be automatically mounted.

▼ containers **[]object** required

A single application container that you want to run within a pod.

▼ args `[]string`

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ command `[]string`

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ env []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> ↗

▼ optional

boolean

Specify whether the Secret or its key must be defined

▼ envFrom **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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 must be defined

▼ prefix **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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 Secret must be defined

▼ image **string**

Container image name. 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.

▼ imagePullPolicy **string**

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle **object**

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart **object**

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET 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
canonicali
zed upon
output, so
case-
variant
names will
be
understoo
d 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.

▼ sleep **object**

Sleep represents a duration that the container should sleep.

▼ seconds**integer****required**

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop `object`

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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

E.

▼ scheme

string

Scheme to use for connecting to the host. Defaults to HTTP.

▼ sleep object

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket object

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle

hooks will fail at runtime
when it is specified.

▼ 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_NAM
E.

▼ livenessProbe object

Periodic probe of container liveness.
Container will be restarted if the probe
fails. Cannot be updated. More info:
<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec object

Exec specifies a command to execute in the container.

▼ **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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **name** **string** required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ ports []object

ContainerPort represents a network port in a single container.

▼ containerPort integer

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ hostIP string

What host IP to bind the external port to.

▼ hostPort integer

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name string

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ resizePolicy `[]object`

ContainerResizePolicy represents resource resize policy for the container.

▼ resourceName `string`

required

Name of the resource to which this resource resize policy applies. Supported values: cpu, memory.

▼ restartPolicy `string`

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources object

Compute Resources required by this container. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/> ↗

▼ 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/> ↗

▼ restartPolicy **string**

RestartPolicy defines the restart behavior
of individual containers in a pod. This field
may only be set for init containers, and the
only allowed value is "Always". For non-init
containers or when this field is not

specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext **object**

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/> ↗

▼ 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
e field.



gmsaCredentialSpecName

string

GMSACredentialSpecName
e 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.

▼ startupProbe object

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec object

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the

container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ **terminationMessagePath**

string

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to `/dev/termination-log`. Cannot be updated.

▼ terminationMessagePolicy

string

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure.

FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File.

Cannot be updated.

▼ tty boolean

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices []object

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath string

required

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts **[]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.

▼ workingDir `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ dnsConfig `object`

Specifies the DNS parameters of a pod.

Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

▼ **nameservers** `[]string`

A list of DNS name server IP addresses.

This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

▼ **options** `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ **name** `string`

Name is this DNS resolver option's name. Required.

▼ **value** `string`

Value is this DNS resolver option's value.

▼ **searches** `[]string`

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

DNSPolicy. Duplicated search paths will be removed.

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirstWithHostNet', 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy. To have DNS options set along with hostNetwork, you have to specify DNS policy explicitly to 'ClusterFirstWithHostNet'.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ ephemeralContainers `[]object`

An EphemeralContainer is a temporary container that you may add to an existing Pod for user-initiated activities such as debugging. Ephemeral containers have no resource or scheduling guarantees, and they will not be restarted when they exit or when a Pod is removed or restarted. The kubelet may evict a Pod if an ephemeral container causes the Pod to exceed its resource allocation.

To add an ephemeral container, use the ephemeralcontainers subresource of an existing

Pod. Ephemeral containers may not be removed or restarted.

▼ args `[]string`

Arguments to the entrypoint. The image's CMD is used if this is not provided.

Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info:

<https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ command `[]string`

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ env []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> ↗

▼ optional

boolean

Specify whether the Secret or its key must be defined

▼ envFrom **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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 must be defined

▼ prefix **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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 Secret must be defined

▼ image **string**

Container image name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ imagePullPolicy **string**

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle **object**

Lifecycle is not allowed for ephemeral containers.

▼ postStart **object**

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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.

▼ **sleep** **object**

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop object

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec object

Exec specifies a command to execute in the container.

▼ **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.

▼ **httpGet** **object**

HTTPGet specifies an HTTP GET 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

E.

▼ scheme

string

Scheme to use for connecting to the host. Defaults to HTTP.

▼ sleep object

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket object

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle

hooks will fail at runtime when it is specified.

▼ 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.

▼ livenessProbe **object**

Probes are not allowed for ephemeral containers.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **name** **string** required

Name of the ephemeral container specified as a DNS_LABEL. This name must be

unique among all containers, init containers and ephemeral containers.

▼ ports []object

ContainerPort represents a network port in a single container.

▼ containerPort integer

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ hostIP string

What host IP to bind the external port to.

▼ hostPort integer

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name string

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name.

Name for the port that can be referred to by services.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

Probes are not allowed for ephemeral containers.

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **resizePolicy** **[]object**

ContainerResizePolicy represents resource resize policy for the container.

▼ **resourceName** **string**

required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ **restartPolicy** **string**

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources **object**

Resources are not allowed for ephemeral containers. Ephemeral containers use spare resources already allocated to 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/> ↗

▼ restartPolicy string

Restart policy for the container to manage the restart behavior of each container within a pod. This may only be set for init containers. You cannot set this field on ephemeral containers.

▼ securityContext object

Optional: SecurityContext defines the security options the ephemeral container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext.

▼ 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.

▼ startupProbe **object**

Probes are not allowed for ephemeral containers.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the

container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ **targetContainerName** **string**

If set, the name of the container from PodSpec that this ephemeral container targets. The ephemeral container will be run in the namespaces (IPC, PID, etc) of this container. If not set then the ephemeral container uses the namespaces configured in the Pod spec.

The container runtime must implement support for this feature. If the runtime does not support namespace targeting then the result of setting this field is undefined.

▼ terminationMessagePath

string

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.

▼ terminationMessagePolicy

string

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure.

FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File.

Cannot be updated.

▼ tty boolean

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices []object

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath string

required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []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.

▼ workingDir `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ hostAliases `[]object`

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

▼ hostnames `[]string`

Hostnames for the above IP address.

▼ ip `string` required

IP address of the host file entry.

▼ hostIPC `boolean`

Use the host's ipc namespace. Optional: Default to false.

▼ hostNetwork `boolean`

Host networking requested for this pod. Use the host's network namespace. If this option is set, the ports that will be used must be specified. Default to false.

▼ **hostPID** `boolean`

Use the host's pid namespace. Optional: Default to false.

▼ **hostUsers** `boolean`

Use the host's user namespace. Optional: Default to true. If set to true or not present, the pod will be run in the host user namespace, useful for when the pod needs a feature only available to the host user namespace, such as loading a kernel module with CAP_SYS_MODULE. When set to false, a new users is created for the pod. Setting false is useful for mitigating container breakout vulnerabilities even allowing users to run their containers as root without actually having root privileges on the host. This field is alpha-level and is only honored by servers that enable the UserNamespacesSupport feature.

▼ **hostname** `string`

Specifies the hostname of the Pod If not specified, the pod's hostname will be set to a system-defined value.

▼ **imagePullSecrets** `[]object`

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

▼ 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> ↗

▼ initContainers `[]object`

A single application container that you want to run within a pod.

▼ args `[]string`

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. 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. Cannot be updated. More info:

<https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ **command** []string

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. 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. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell> ↗

▼ **env** []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/o](https://kubernetes.io/docs/concepts/o)

verview/working-
with-
objects/names/#n
ames ↗

▼ optional

boolean

Specify whether
the Secret or its
key must be
defined

▼ envFrom **[]object**

EnvFromSource represents the source of
a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ 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/workin](https://kubernetes.io/docs/concepts/overview/workin)

[g-with-objects/names/#names ↗](#)

▼ **optional** **boolean**

Specify whether the ConfigMap must be defined

▼ **prefix** **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** **object**

The Secret to select from

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

[g-with-objects/names/#names ↗](#)

▼ optional `boolean`

Specify whether the Secret must be defined

▼ image `string`

Container image name. 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.

▼ imagePullPolicy `string`

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle `object`

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart `object`

PostStart is called immediately after a container is created. If the

handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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.

▼ sleep object

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket object

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ preStop **object**

PreStop is called immediately before a container is terminated due to an API request or

management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: <https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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.

▼ httpGet object

HTTPGet specifies an HTTP GET 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.

▼ **sleep** **object**

Sleep represents a duration that the container should sleep.

▼ seconds

integer

required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

▼ 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.

▼ livenessProbe **object**

Periodic probe of container liveness.

Container will be restarted if the probe

fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to

execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **name** **string** required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ **ports** **[]object**

ContainerPort represents a network port in a single container.

▼ **containerPort** **integer**

required

Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.

▼ hostIP **string**

What host IP to bind the external port to.

▼ hostPort **integer**

Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name **string**

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ protocol **string**

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe **object**

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **resizePolicy** []object

ContainerResizePolicy represents resource resize policy for the container.

▼ **resourceName** string

required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ **restartPolicy** string

required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ **resources** object

Compute Resources required by this container. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/> ↗

▼ **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/> ↗

▼ restartPolicy **string**

RestartPolicy defines the restart behavior of individual containers in a pod. This field may only be set for init containers, and the only allowed value is "Always". For non-init containers or when this field is not specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next

init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext **object**

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/> ↗

▼ 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.

▼ startupProbe object

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ 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 a GRPC HealthCheckRequest.

▼ 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 an HTTP GET 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 a connection to 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> ↗

▼ **stdin** **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If `stdinOnce` is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is

restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath

string

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.

▼ terminationMessagePolicy

string

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure.

FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ tty boolean

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices []object

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath string

required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []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.

▼ **workingDir** `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ **nodeName** `string`

`nodeName` indicates in which node this pod is scheduled. If empty, this pod is a candidate for scheduling by the scheduler defined in `schedulerName`. Once this field is set, the kubelet for this node becomes responsible for the lifecycle of this pod. This field should not be used to express a desire for the pod to be scheduled on a specific node.

<https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#nodename> ↗

▼ **nodeSelector** `object`

`NodeSelector` is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info:

<https://kubernetes.io/docs/concepts/configuration/assign-pod-node/> ↗

▼ os object

Specifies the OS of the containers in the pod.

Some pod and container fields are restricted if this is set.

If the OS field is set to linux, the following fields must be unset: -securityContext.windowsOptions

If the OS field is set to windows, following fields must be unset:

- spec.hostPID
- spec.hostIPC
- spec.hostUsers
- spec.securityContext.appArmorProfile
- spec.securityContext.seLinuxOptions
- spec.securityContext.seccompProfile
- spec.securityContext.fsGroup
- spec.securityContext.fsGroupChangePolicy
- spec.securityContext.sysctls
- spec.shareProcessNamespace
- spec.securityContext.runAsUser
- spec.securityContext.runAsGroup
- spec.securityContext.supplementalGroups
- spec.securityContext.supplementalGroupsPolicy
- spec.containers[*].securityContext.appArmorProfile
- spec.containers[*].securityContext.seLinuxOptions

- `spec.containers[*].securityContext.seccompProfile`
- `spec.containers[*].securityContext.capabilities`
- `spec.containers[*].securityContext.readOnlyRootFilesystem`
- `spec.containers[*].securityContext.privileged`
- `spec.containers[*].securityContext.allowPrivilegeEscalation`
- `spec.containers[*].securityContext.procMount`
- `spec.containers[*].securityContext.runAsUser`
- `spec.containers[*].securityContext.runAsGroup`

▼ **name** `string` required

Name is the name of the operating system.

The currently supported values are linux and windows. Additional value may be defined in future and can be one of:

<https://github.com/opencontainers/runtime-spec/blob/master/config.md#platform-specific-configuration> ↗ Clients should expect to handle additional values and treat unrecognized values in this field as os: null

▼ **overhead** `object`

Overhead represents the resource overhead associated with running a pod for a given

RuntimeClass. This field will be autopopulated at admission time by the RuntimeClass admission controller. If the RuntimeClass admission controller is enabled, overhead must not be set in Pod create requests. The RuntimeClass admission controller

will reject Pod create requests which have the overhead already set. If RuntimeClass is configured and selected in the PodSpec, Overhead will be set to the value defined in the corresponding RuntimeClass, otherwise it will remain unset and treated as zero. More info: <https://git.k8s.io/enhancements/keps/sig-node/688-pod-overhead/README.md> ↗

▼ **preemptionPolicy** `string`

PreemptionPolicy is the Policy for preempting pods with lower priority. One of Never, PreemptLowerPriority. Defaults to PreemptLowerPriority if unset.

▼ **priority** `integer`

The priority value. Various system components use this field to find the priority of the pod. When Priority Admission Controller is enabled, it prevents users from setting this field. The admission controller populates this field from PriorityClassName. The higher the value, the higher the priority.

▼ **priorityClassName** `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified,

the pod priority will be default or zero if there is no default.

▼ readinessGates `[]object`

PodReadinessGate contains the reference to a pod condition

▼ conditionType `string` required

ConditionType refers to a condition in the pod's condition list with matching type.

▼ resourceClaims `[]object`

PodResourceClaim references exactly one ResourceClaim, either directly or by naming a ResourceClaimTemplate which is then turned into a ResourceClaim for the pod.

It adds a name to it that uniquely identifies the ResourceClaim inside the Pod. Containers that need access to the ResourceClaim reference it with this name.

▼ name `string` required

Name uniquely identifies this resource claim inside the pod. This must be a DNS_LABEL.

▼ resourceClaimName `string`

ResourceClaimName is the name of a ResourceClaim object in the same

namespace as this pod.

Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

▼ resourceClaimTemplateName

string

ResourceClaimTemplateName is the name of a ResourceClaimTemplate object in the same namespace as this pod.

The template will be used to create a new ResourceClaim, which will be bound to this pod. When this pod is deleted, the ResourceClaim will also be deleted. The pod name and resource name, along with a generated component, will be used to form a unique name for the ResourceClaim, which will be recorded in pod.status.resourceClaimStatuses.

This field is immutable and no changes will be made to the corresponding ResourceClaim by the control plane after creating the ResourceClaim.

Exactly one of ResourceClaimName and ResourceClaimTemplateName must be set.

▼ resources object

Resources is the total amount of CPU and Memory resources required by all containers in the pod. It supports specifying Requests and Limits for "cpu"

and "memory" resource names only.

ResourceClaims are not supported.

This field enables fine-grained control over resource allocation for the entire pod, allowing resource sharing among containers in a pod.

This is an alpha field and requires enabling the PodLevelResources feature gate.

▼ 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/> ↗

▼ restartPolicy string

Restart policy for all containers within the pod. One of Always, OnFailure, Never. In some contexts, only a subset of those values may be permitted.

Default to Always. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/#restart-policy> ↗

▼ runtimeClassName string

RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit class with an empty definition that uses the default runtime handler.

More info:

<https://git.k8s.io/enhancements/keps/sig-node/585-runtime-class> ↗

▼ schedulerName `string`

If specified, the pod will be dispatched by specified scheduler. If not specified, the pod will be dispatched by default scheduler.

▼ schedulingGates `[]object`

PodSchedulingGate is associated to a Pod to guard its scheduling.

▼ name `string` required

Name of the scheduling gate. Each scheduling gate must have a unique name field.

▼ securityContext `object`

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field.

▼ 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.

▼ seLinuxChangePolicy string

seLinuxChangePolicy defines how the container's SELinux label is applied to all volumes used by the Pod. It has no effect on nodes that do not support SELinux or to volumes does not support SELinux. Valid values are "MountOption" and "Recursive". "Recursive" means relabeling of all files on all Pod volumes by the container runtime. This may be slow for large volumes, but allows mixing privileged and unprivileged Pods sharing the same volume on the same node.

"MountOption" mounts all eligible Pod volumes with `-o context` mount option. This requires all Pods that share the same volume to use the same SELinux label. It is not possible to share the same volume among privileged and unprivileged Pods. Eligible volumes are in-tree FibreChannel and iSCSI volumes, and all CSI volumes whose CSI driver announces SELinux support by setting `spec.selinuxMount: true` in their CSIDriver instance. Other volumes are always re-labelled recursively.

"MountOption" value is allowed only when SELinuxMount feature gate is enabled.

If not specified and SELinuxMount feature gate is enabled, "MountOption" is used. If not specified and SELinuxMount feature gate is disabled, "MountOption" is used for ReadWriteOncePod volumes and "Recursive" for all other volumes.

This field affects only Pods that have SELinux label set, either in PodSecurityContext or in SecurityContext of all containers.

All Pods that use the same volume should use the same `seLinuxChangePolicy`, otherwise some pods can get stuck in ContainerCreating state. 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.

▼ sysctl []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 string

DeprecatedServiceAccount is a deprecated alias for ServiceAccountName. Deprecated: Use serviceAccountName instead.

▼ serviceAccountName string

ServiceAccountName is the name of the ServiceAccount to use to run this pod. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/> ↗

▼ setHostnameAsFQDN boolean

If true the pod's hostname will be configured as the pod's FQDN, rather than the leaf name (the default). In Linux containers, this means setting the FQDN in the hostname field of the kernel (the nodename field of struct utsname). In Windows containers, this means setting the registry value of hostname for the registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters to FQDN. If a pod does not have FQDN, this has no effect. Default to false.

▼ shareProcessNamespace boolean

Share a single process namespace between all of the containers in a pod. When this is set containers will be able to view and signal processes from other containers in the same pod, and the first process in each container will not be assigned PID 1. HostPID and ShareProcessNamespace cannot both be set. Optional: Default to false.

▼ subdomain string

If specified, the fully qualified Pod hostname will be "...svc.". If not specified, the pod will not have a domainname at all.

▼ terminationGracePeriodSeconds

integer

Optional duration in seconds the pod needs to terminate gracefully. May be decreased in delete request. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). If this value is nil, the default grace period will be used instead. 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. Defaults to 30 seconds.

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ labelSelector **object**

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

▼ 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 the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be

ignored. A null or empty list means only match against labelSelector.

This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

▼ maxSkew integer required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. | zone1 | zone2 | zone3 | | P P | P P | P |

- if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1).
- if MaxSkew is 2, incoming pod can be scheduled onto any zone. When `whenUnsatisfiable=ScheduleAnyway`, it is used to give higher precedence to topologies that satisfy it. It's a

required field. Default value is 1 and 0 is not allowed.

▼ minDomains integer

MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule.

For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: | zone1 | zone2 | zone3 | | P P | P P | P P | The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is

scheduled to any of the three zones, it will violate MaxSkew.

▼ nodeAffinityPolicy `string`

NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are:

- Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations.
- Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations.

If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.

▼ nodeTaintsPolicy `string`

NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are:

- Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included.
- Ignore: node taints are ignored. All nodes are included.

If this value is nil, the behavior is equivalent to the Ignore policy. This is a

beta-level feature default enabled by the `NodeInclusionPolicyInPodTopologySpread` feature flag.

▼ **topologyKey** `string` required

`TopologyKey` is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each `<key, value>` as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of `nodeAffinityPolicy` and `nodeTaintsPolicy`. e.g. If `TopologyKey` is `"kubernetes.io/hostname"`, each Node is a domain of that topology. And, if `TopologyKey` is `"topology.kubernetes.io/zone"`, each zone is a domain of that topology. It's a required field.

▼ **whenUnsatisfiable** `string` required

`WhenUnsatisfiable` indicates how to deal with a pod if it doesn't satisfy the spread constraint.

- `DoNotSchedule` (default) tells the scheduler not to schedule it.
- `ScheduleAnyway` tells the scheduler to schedule the pod in any location, but

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

▼ volumes []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. Deprecated:
AWSElasticBlockStore is deprecated. All operations for the in-tree awsElasticBlockStore type are redirected to the ebs.csi.aws.com CSI driver. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#awselasticsearchstore> ↗

▼ 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#awselasticsearchstore> ↗

▼ 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>

[s/storage/volumes#awselasticbloc](#)
[kstore ↗](#)

▼ **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 ↗](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. Deprecated: AzureDisk is deprecated. All operations for the in-tree azureDisk type are redirected to the disk.csi.azure.com CSI driver.

▼ **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. Deprecated: AzureFile is deprecated. All operations for the in-tree azureFile type are redirected to the file.csi.azure.com CSI driver.

▼ **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. Deprecated: CephFS is deprecated and the in-tree cephfs type is no longer supported.

▼ 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>

[ephfs/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/c>

[ephfs/README.md#how-to-use-it](#)



▼ cinder **object**

cinder represents a cinder volume attached and mounted on kubelets host machine. Deprecated: Cinder is deprecated. All operations for the in-tree cinder type are redirected to the [cinder.csi.openstack.org](https://examples.k8s.io/mysql-cinder-pd/README.md) CSI driver. 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.

▼ 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/e/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 be 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 PersistentVolume Claim 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

Namespac
e is the
namespac
e of
resource
being
referenced

Note that
when a
namespac
e is
specified,
a
gateway.n
etworking.
k8s.io/Ref
erenceGra
nt object is
required in
the
referent
namespac
e to allow
that
namespac
e's owner
to accept
the
reference.

See the Reference Grant 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/management-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://kub
ernetes.io/
docs/conc
epts/confi
guration/m
anage-
resources-](https://kubernetes.io/docs/concepts/configuration/management-resources-)

containers



▼ selector

object

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



matchExpressions

[]object

t

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



key

s

trigger

required

key is the label key that the selector or applies to.



operator

string

required

operator relationship to a set of values.
Valid operators are In, N

otl
n,
Ex
ist
s
an
d
D
oe
s
N
ot
Ex
ist
.

▼
values

[
]s
tr
in
g

va
lu
es
is
an
ar
ra
y
of
str
in
g
va

values. If the operator is In or Not on, the values are array must be non-empty. If the op

er
at
or
is
Ex
ist
s
or
D
oe
s
N
ot
Ex
ist
,
th
e
va
lu
es
ar
ra
y
m
us
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.



storageClassName

string

storageClassName is the name of the StorageClass required by the claim. More info: <https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1>



volumeAttributesClassName

string

volumeAttributesClassName may be used to set the VolumeAttributesClassName used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClassName. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string value means that no VolumeAttributesClassName 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 VolumeAttributesC

lass will be set by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesC lass does not exist, this PersistentVolume Claim 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 VolumeAttributesC lass 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. Deprecated: FlexVolume is deprecated. Consider using a CSIDriver instead.

▼ 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. Deprecated: Flocker is deprecated and the in-tree flocker type is no longer supported.

▼ 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. Deprecated: GCEPersistentDisk is deprecated. All operations for the in-tree gcePersistentDisk type are redirected to the pd.csi.storage.gke.io CSI driver. 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. **Deprecated:** Glusterfs is deprecated and the in-tree glusterfs type is no longer

supported. 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#persistent-volumeclaims> ↗

▼ **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#persistent-volumeclaims>

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.
Deprecated: PhotonPersistentDisk is deprecated and the in-tree photonPersistentDisk type is no longer supported.

▼ 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. Deprecated:

PortworxVolume is deprecated. All operations for the in-tree portworxVolume type are redirected to the pxd.portworx.com CSI driver when the CSIMigrationPortworx feature-gate is on.

▼ 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

t

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



key

s

tr

in

g

required

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

▼
operator
s
tr
in
g

required

op
er
at
or
re

presentations as key's relationship to a set of values.
Valid operators are In, NotIn, Exists

and does not exist.

▼
values

[
]s
tr
in
g

values is an array of string values. If the

operator is in order. Not only, the values are array must be non-empty. If the operator is Ex

ists, the values are arrays must be empty. This array is replaced

during strategic merger package.



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

e 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

0

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
annotation
s, labels,
name,
namespac
e and uid
are
supported.



apiVersion

s
tr
in
g

Ve
rsi
on
of
th
e
sc
he
m
a
th

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

▼
fieldPath
string

required

Path of the

efileld to select in the specified API version.

▼ mode

integer

r

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

s
tr
in
g

C
on
tai
ne
r
na
m
e:
re
qu
ire
d
for
vo
lu
m
es

,
op
tio
na
l
for
en
v
va
rs

▼
divisor

S
pe
cif
ie
s
th
e
ou
tp
ut
for
m
at
of
th
e
ex
po
se
d
re
so

ur
ce
s,
de
fa
ult
s
to
"1
"

▼
resource
s
tr
in
g

required

R
eq
uir
ed
:
re
so
ur
ce
to
se
le
ct

▼ 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

r

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.
Deprecated: Quobyte is deprecated and the in-tree quobyte type is no longer supported.

▼ 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 serviceaccount 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. Deprecated: RBD is deprecated and the in-tree rbd type is no longer supported. 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. Deprecated: ScaleIO is deprecated and the in-tree scaleIO type is no longer supported.

▼ 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/workin-g-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. Deprecated: StorageOS is deprecated and the in-tree storageos type is no longer supported.

▼ 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. Deprecated: VsphereVolume is deprecated. All operations for the in-tree vsphereVolume type are redirected to the csi.vsphere.vmware.com CSI driver.

▼ 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 vmrk

▼ **serviceAccountName** `string`

▼ **triggerGroups** `[]object`

EventListenerTriggerGroup defines a group of Triggers that share a common set of interceptors

▼ interceptors []object required

TriggerInterceptor provides a hook to intercept and pre-process events

▼ name string

Optional name to identify the current interceptor configuration

▼ params []object

InterceptorParams defines a key-value pair that can be passed on an interceptor

▼ name string required

▼ value required

▼ ref object required

Ref refers to the Interceptor to use

▼ apiVersion string

API version of the referent

▼ kind string

InterceptorKind indicates the kind of the Interceptor, namespaced or cluster scoped.

▼ name **string**

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ webhook **object**

WebhookInterceptor refers to an old style webhook interceptor service

▼ header **[]object**

Param declares an ParamValues to use for the parameter called name.

▼ name **string** **required****▼ value** **required****▼ objectRef** **object**

ObjectRef is a reference to an object that will resolve to a cluster DNS name to use as the EventInterceptor. Either objectRef or url can be specified

▼ apiVersion **string**

API version of the referent.

▼ fieldPath **string**

If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as `desiredState.manifest.containers[2]`. For example, if the object reference is to a container within a pod, this would take on a value like: `"spec.containers{name}"` (where "name" refers to the name of the container that triggered the event) or if no container name is specified `"spec.containers[2]"` (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

▼ **kind** `string`

Kind of the referent. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds> ↗

▼ **name** `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **namespace** `string`

Namespace of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/> ↗

▼ **resourceVersion** `string`

Specific resourceVersion to which this reference is made, if any. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency> ↗

▼ uid **string**

UID of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids> ↗

▼ url **string**

▼ name **string** required

▼ triggerSelector **object** required

EventListenerTriggerSelector defines ways to select a group of triggers using their metadata

▼ labelSelector **object**

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

▼ 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.

▼ **namespaceSelector** `object`

NamespaceSelector is a selector for selecting either all namespaces or a list of namespaces.

▼ **matchNames** `[]string`

List of namespace names.

▼ triggers []object

EventListenerTrigger represents a connection between TriggerBinding, Params, and TriggerTemplate; TriggerBinding provides extracted values for TriggerTemplate to then create resources from. TriggerRef can also be provided instead of TriggerBinding, Interceptors and TriggerTemplate

▼ bindings []object

▼ apiversion string

APIVersion of the binding ref

▼ kind string

Kind can only be provided if Ref is also provided. Defaults to TriggerBinding

▼ name string

Name is the name of the binding param Mutually exclusive with Ref

▼ ref string

Ref is a reference to a TriggerBinding kind. Mutually exclusive with Name

▼ value string

Value is the value of the binding param. Can contain JSONPath Has to be pointer since "" is a valid value Required if Name is also specified.

▼ interceptors `[]object`

TriggerInterceptor provides a hook to intercept and pre-process events

▼ name `string`

Optional name to identify the current interceptor configuration

▼ params `[]object`

InterceptorParams defines a key-value pair that can be passed on an interceptor

▼ name `string` required

▼ value required

▼ ref `object` required

Ref refers to the Interceptor to use

▼ apiVersion `string`

API version of the referent

▼ kind `string`

InterceptorKind indicates the kind of the Interceptor, namespaced or cluster scoped.

▼ name string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ webhook object

WebhookInterceptor refers to an old style webhook interceptor service

▼ header []object

Param declares an ParamValues to use for the parameter called name.

▼ name string required**▼ value** required**▼ objectRef** object

ObjectRef is a reference to an object that will resolve to a cluster DNS name to use as the EventInterceptor. Either objectRef or url can be specified

▼ apiVersion string

API version of the referent.

▼ fieldPath string

If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as `desiredState.manifest.containers[2]`. For example, if the object reference is to a container within a pod, this would take on a value like: `"spec.containers{name}"` (where "name" refers to the name of the container that triggered the event) or if no container name is specified `"spec.containers[2]"` (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

▼ **kind** `string`

Kind of the referent. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds> ↗

▼ **name** `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **namespace** `string`

Namespace of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/> ↗

▼ **resourceVersion** `string`

Specific resourceVersion to which this reference is made, if any. More info:

<https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency> ↗

▼ uid `string`

UID of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids> ↗

▼ url `string`

▼ name `string`

▼ serviceAccountName `string`

ServiceAccountName optionally associates credentials with each trigger; more granular authorization for who is allowed to utilize the associated pipeline vs. defaulting to whatever permissions are associated with the entire EventListener and associated sink facilitates multi-tenant model based scenarios

▼ template `object`

▼ apiversion `string`

▼ ref `string`

▼ spec **object**

TriggerTemplateSpec holds the desired state of TriggerTemplate

▼ params **[]object**

ParamSpec defines an arbitrary named input whose value can be supplied by a **Param**.

▼ default **string**

Default is the value a parameter takes if no input value via a Param is supplied.

▼ description **string**

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ name **string** **required**

Name declares the name by which a parameter is referenced.

▼ resourcetemplates **[]object**

TriggerResourceTemplate describes a resource to create

▼ triggerRef **string**

▼ status **object**

EventListenerStatus holds the status of the EventListener

▼ address **object**

Address is a single Addressable address. If Addresses is present, Address will be ignored by clients.

▼ CACerts **string**

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ name **string**

Name is the name of the address.

▼ url **string**

▼ addresses **[]object**

Addressable provides a generic mechanism for a custom resource definition to indicate a destination for message delivery.

Addressable is the schema for the destination information. This is typically stored in the object's **status**, as this information may be generated by the controller.

▼ CACerts **string**

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ name `string`

Name is the name of the address.

▼ url `string`**▼ annotations** `object`

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

▼ conditions `[]object`

Condition defines a readiness condition for a Knative resource. See:

<https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties> ↗

▼ lastTransitionTime `string`

LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

▼ message `string`

A human readable message indicating details about the transition.

▼ reason `string`

The reason for the condition's last transition.

▼ severity `string`

Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

▼ status `string` `required`

Status of the condition, one of True, False, Unknown.

▼ type `string` `required`

Type of condition.

▼ configuration `object` `required`

Configuration stores configuration for the EventListener service

▼ generatedName `string` `required`

GeneratedResourceName is the name given to all resources reconciled by the EventListener

▼ observedGeneration `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

触发绑定 [triggers.tekton.dev/v1beta1]

triggers.tekton.dev group

TriggerBinding defines a mapping of an input event to parameters. This is used to extract information from events to be passed to TriggerTemplates within a Trigger.

v1alpha1 version

▼ spec object

Spec holds the desired state of the TriggerBinding

▼ params []object

Param defines a string value to be used for a ParamSpec with the same name.

▼ name string required

▼ value string required

▼ status object

TriggerBindingStatus defines the observed state of TriggerBinding.

TriggerBinding defines a mapping of an input event to parameters. This is used to extract information from events to be passed to TriggerTemplates within a Trigger.

v1beta1

version

▼ spec object

Spec holds the desired state of the TriggerBinding

▼ params []object

Param defines a string value to be used for a ParamSpec with the same name.

▼ name string required**▼ value** string required**▼ status** object

TriggerBindingStatus defines the observed state of TriggerBinding.



拦截器 [triggers.tekton.dev/v1alpha1]

`triggers.tekton.dev``group`

Interceptor describes a pluggable interceptor including configuration such as the fields it accepts and its deployment address. The type is based on the Validating/MutatingWebhookConfiguration types for configuring AdmissionWebhooks

`v1alpha1``version`

▼ spec `object` `required`

InterceptorSpec describes the Spec for an Interceptor

▼ clientConfig `object` `required`

ClientConfig describes how a client can communicate with the Interceptor

▼ caBundle `string`

CaBundle is a PEM encoded CA bundle which will be used to validate the clusterinterceptor server certificate

▼ service `object`

Service is a reference to a Service object where the interceptor is running Mutually exclusive with URL

▼ name `string` `required`

Name is the name of the service

▼ namespace **string** required

Namespace is the namespace of the service

▼ path **string**

Path is an optional URL path

▼ port **integer**

Port is a valid port number

▼ url **string**

URL is a fully formed URL pointing to the interceptor Mutually exclusive with Service

▼ status **object**

InterceptorStatus holds the status of the Interceptor

▼ address **object**

Address is a single Addressable address. If Addresses is present, Address will be ignored by clients.

▼ CACerts **string**

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ audience **string**

Audience is the OIDC audience for this address.

▼ **name** `string`

Name is the name of the address.

▼ **url** `string`

▼ **addresses** `[]object`

Addressable provides a generic mechanism for a custom resource definition to indicate a destination for message delivery.

Addressable is the schema for the destination information. This is typically stored in the object's `status`, as this information may be generated by the controller.

▼ **CACerts** `string`

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468> ↗.

▼ **audience** `string`

Audience is the OIDC audience for this address.

▼ **name** `string`

Name is the name of the address.

▼ **url** `string`

▼ annotations `object`

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

▼ conditions `[]object`

Condition defines a readiness condition for a Knative resource. See:

<https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties> ↗

▼ lastTransitionTime `string`

LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

▼ message `string`

A human readable message indicating details about the transition.

▼ reason `string`

The reason for the condition's last transition.

▼ severity `string`

Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

▼ status `string` required

Status of the condition, one of True, False, Unknown.

▼ **type** `string` required

Type of condition.

▼ **observedGeneration** `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

集群触发器绑定

[triggers.tekton.dev/v1beta1]

`triggers.tekton.dev` `group`

ClusterTriggerBinding is a TriggerBinding with a cluster scope. ClusterTriggerBindings are used to represent TriggerBindings that should be publicly addressable from any namespace in the cluster.

`v1alpha1` `version`

▼ spec `object`

Spec holds the desired state of the ClusterTriggerBinding from the client

▼ params `[]object`

Param defines a string value to be used for a ParamSpec with the same name.

▼ name `string` `required`

▼ value `string` `required`

▼ status `object`

TriggerBindingStatus defines the observed state of TriggerBinding.

ClusterTriggerBinding is a TriggerBinding with a cluster scope. ClusterTriggerBindings are used to represent TriggerBindings that should be publicly addressable from any namespace in the cluster.

v1beta1

version

▼ spec object

Spec holds the desired state of the ClusterTriggerBinding from the client

▼ params []object

Param defines a string value to be used for a ParamSpec with the same name.

▼ name string required**▼ value** string required**▼ status** object

TriggerBindingStatus defines the observed state of TriggerBinding.

集群拦截器 [triggers.tekton.dev/v1alpha1]

triggers.tekton.dev group

ClusterInterceptor describes a pluggable interceptor including configuration such as the fields it accepts and its deployment address. The type is based on the Validating/MutatingWebhookConfiguration types for configuring AdmissionWebhooks

v1alpha1 version

▼ spec object required

ClusterInterceptorSpec describes the Spec for an ClusterInterceptor

▼ clientConfig object required

ClientConfig describes how a client can communicate with the Interceptor

▼ caBundle string

CaBundle is a PEM encoded CA bundle which will be used to validate the clusterinterceptor server certificate

▼ service object

Service is a reference to a Service object where the interceptor is running Mutually exclusive with URL

▼ name string required

Name is the name of the service

▼ namespace **string** required

Namespace is the namespace of the service

▼ path **string**

Path is an optional URL path

▼ port **integer**

Port is a valid port number

▼ url **string**

URL is a fully formed URL pointing to the interceptor Mutually exclusive with Service

▼ status **object**

ClusterInterceptorStatus holds the status of the ClusterInterceptor

▼ address **object**

Address is a single Addressable address. If Addresses is present, Address will be ignored by clients.

▼ CACerts **string**

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468>.

▼ audience **string**

Audience is the OIDC audience for this address.

▼ **name** `string`

Name is the name of the address.

▼ **url** `string`

▼ **addresses** `[]object`

Addressable provides a generic mechanism for a custom resource definition to indicate a destination for message delivery.

Addressable is the schema for the destination information. This is typically stored in the object's `status`, as this information may be generated by the controller.

▼ **CACerts** `string`

CACerts is the Certification Authority (CA) certificates in PEM format according to <https://www.rfc-editor.org/rfc/rfc7468> ↗.

▼ **audience** `string`

Audience is the OIDC audience for this address.

▼ **name** `string`

Name is the name of the address.

▼ **url** `string`

▼ annotations **object**

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

▼ conditions **[]object**

Condition defines a readiness condition for a Knative resource. See:

<https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties> ↗

▼ lastTransitionTime **string**

LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

▼ message **string**

A human readable message indicating details about the transition.

▼ reason **string**

The reason for the condition's last transition.

▼ severity **string**

Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

▼ status **string** required

Status of the condition, one of True, False, Unknown.

▼ **type** `string` required

Type of condition.

▼ **observedGeneration** `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

ScheduledTrigger

[tekton.alaudadevops.io/v1alpha1]

tekton.alaudadevops.io group

ScheduledTrigger is the Schema for the scheduledtriggers API

v1alpha1 version

▼ spec object required

spec defines the desired state of ScheduledTrigger

▼ params []object

Param defines a string value to be used for a ParamSpec with the same name.

▼ name string required

▼ value string required

▼ schedule string

The schedule in Cron format, see <https://en.wikipedia.org/wiki/Cron>.

▼ timeZone string

The time zone name for the given schedule, see

https://en.wikipedia.org/wiki/List_of_tz_database_time_zones ↗. If not specified, this will default to the time zone of the kube-controller-manager process. The set of valid time zone names and the time zone offset is loaded from the system-wide time zone database by the API server during CronJob validation and the controller manager during execution. If no system-wide time zone database can be found a bundled version of the database is used instead. If the time zone name becomes invalid during the lifetime of a CronJob or due to a change in host configuration, the controller will stop creating new new Jobs and will create a system event with the reason UnknownTimeZone. More information can be found in <https://kubernetes.io/docs/concepts/workloads/controllers/cron-jobs/#time-zones> ↗

▼ triggerTemplate object required

▼ apiversion string

▼ ref string

▼ spec object

TriggerTemplateSpec holds the desired state of TriggerTemplate

▼ params []object

ParamSpec defines an arbitrary named input whose value can be supplied by a Param .

▼ default string

Default is the value a parameter takes if no input value via a Param is supplied.

▼ description string

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ **name** `string` required

Name declares the name by which a parameter is referenced.

▼ **resourcetemplates** `[]object`

TriggerResourceTemplate describes a resource to create

▼ **status** `object`

status defines the observed state of ScheduledTrigger

▼ **lastScheduleTime** `string`

Information when was the last time the job was successfully scheduled.