Cluster Version APIs

ProductManifest [config.alauda
ClusterVersionShadow [config.alauda.io/v1]

ProductManifest [config.alauda.io/v1]

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec selects the product manifest source to resolve.
`status`	`object`	status reports resolved artifacts for the requested manifest source.

.spec

Description

spec selects the product manifest source to resolve.

Type

`object`

Property	Type	Description
`image`	`string`	image is a container image reference that contains the update payload.
`version`	`string`	version is the container image tag that identifies the update payload.

.status

Description

status reports resolved artifacts for the requested manifest source.

Type

`object`

Property	Type	Description
`artifacts`	`array`	artifacts reports channel-level artifact metadata and availability.
`conditions`	`array`	conditions reports reconciliation state for this ProductManifest.
`observedGeneration`	`integer`	observedGeneration of the object. When this matches the object's metadata.generation, it indicates the status is up-to-date.
`resolvedImageDigest`	`string`	resolvedImageDigest is the digest resolved from the desired release image.

.status.artifacts

Description

artifacts reports channel-level artifact metadata and availability.

Type

`array`

.status.artifacts[]

Description

Artifact describes one manifest component and its channels.

Type

`object`

Required

`name`

Property	Type	Description
`channels`	`array`	channels lists all channels reported by the manifest for this component.
`name`	`string`	name is the manifest component name.
`packageType`	`string`	packageType is the manifest package type.

.status.artifacts[].channels

Description

channels lists all channels reported by the manifest for this component.

Type

`array`

.status.artifacts[].channels[]

Description

ArtifactChannel describes one artifact channel from product.yaml.

Type

`object`

Required

`channel` `repository` `tag`

Property	Type	Description
`artifactStatus`	`string`	artifactStatus is the channel artifact probe result.
`brief`	`object`	brief is the localized short channel summary.
`catalogSource`	`string`	catalogSource is the OLM catalog source name.
`categories`	`array`	categories is the category set of this channel.
`channel`	`string`	channel is the channel name in product.yaml.
`default`	`boolean`	default indicates whether this channel is the default channel.
`description`	`object`	description is the localized channel description.

Property	Type	Description
`digest`	`string`	digest is the optional OCI digest of this channel package.
`displayName`	`object`	displayName is the localized channel display name.
`hidden`	`boolean`	hidden indicates this channel should be hidden from normal display.
`packIgnore`	`boolean`	packIgnore controls whether this channel is excluded by packaging workflows.
`provider`	`object`	provider is the localized provider name.
`providerType`	`string`	providerType is the provider classification.
`repository`	`string`	repository is the OCI repository of this channel package.
`supportedArchitectures`	`array`	supportedArchitectures is the architecture set this channel supports.

Property	Type	Description
`supportedPlatformVersions`	`array`	supportedPlatformVersions is the ACP platform version set this channel supports.
`supportedProtocolStacks`	`array`	supportedProtocolStacks is the protocol stack set this channel supports.
`tag`	`string`	tag is the OCI tag of this channel package.

.status.artifacts[].channels[].brief

Description

brief is the localized short channel summary.

Type

`object`

Property	Type	Description
`en`	`string`	en is the English variant.
`zh`	`string`	zh is the Simplified Chinese variant.

.status.artifacts[].channels[].categories

Description

categories is the category set of this channel.

Type

`array`

.status.artifacts[].channels[].categories[]

Type

`string`

.status.artifacts[].channels[].description

Description

description is the localized channel description.

Type

`object`

Property	Type	Description
`en`	`string`	en is the English variant.
`zh`	`string`	zh is the Simplified Chinese variant.

.status.artifacts[].channels[].displayName

Description

displayName is the localized channel display name.

Type

`object`

Property	Type	Description
`en`	`string`	en is the English variant.
`zh`	`string`	zh is the Simplified Chinese variant.

.status.artifacts[].channels[].provider

Description

provider is the localized provider name.

Type

`object`

Property	Type	Description
`en`	`string`	en is the English variant.
`zh`	`string`	zh is the Simplified Chinese variant.

.status.artifacts[].channels[].supportedArchitectures

Description

supportedArchitectures is the architecture set this channel supports.

Type

`array`

.status.artifacts[].channels[].supportedArchitectures[]

Type

`string`

.status.artifacts[].channels[].supportedPlatformVersions

Description

supportedPlatformVersions is the ACP platform version set this channel supports.

Type

`array`

.status.artifacts[].channels[].supportedPlatformVersions[]

Type

`string`

.status.artifacts[].channels[].supportedProtocolStacks

Description

supportedProtocolStacks is the protocol stack set this channel supports.

Type

`array`

.status.artifacts[].channels[].supportedProtocolStacks[]

Type

`string`

.status.conditions

Description

conditions reports reconciliation state for this ProductManifest.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.

Property	Type	Description
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

API Endpoints

The following API endpoints are available:

- `/apis/config.alauda.io/v1/productmanifests`
  - `DELETE`: delete collection of ProductManifest
  - `GET`: list objects of kind ProductManifest
  - `POST`: create a new ProductManifest
- `/apis/config.alauda.io/v1/productmanifests/{name}`
  - `DELETE`: delete the specified ProductManifest
  - `GET`: read the specified ProductManifest
  - `PATCH`: partially update the specified ProductManifest

- `PUT`: replace the specified ProductManifest

- `/apis/config.alauda.io/v1/productmanifests/{name}/status`
  - `GET`: read status of the specified ProductManifest
  - `PATCH`: partially update status of the specified ProductManifest
  - `PUT`: replace status of the specified ProductManifest

/apis/config.alauda.io/v1/productmanifests

HTTP method

`DELETE`

Description

delete collection of ProductManifest

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ProductManifest

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifestList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ProductManifest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
201 - Created	`ProductManifest` schema
202 - Accepted	`ProductManifest` schema
401 - Unauthorized	Empty

/apis/config.alauda.io/v1/productmanifests/{name}

HTTP method

`DELETE`

Description

delete the specified ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ProductManifest

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ProductManifest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
201 - Created	`ProductManifest` schema
401 - Unauthorized	Empty

/apis/config.alauda.io/v1/productmanifests/{name}/status

HTTP method

`GET`

Description

read status of the specified ProductManifest

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ProductManifest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ProductManifest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ProductManifest` schema
201 - Created	`ProductManifest` schema
401 - Unauthorized	Empty

ClusterVersionShadow [config.alauda.io/v1]

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	spec is the desired state of the cluster upgrade.
`status`	`object`	status contains information about cluster upgrade.

.spec

Description

spec is the desired state of the cluster upgrade.

Type

`object`

Property	Type	Description
`desiredUpdate`	`object`	desiredUpdate is an optional field that indicates the desired value of the cluster version. Setting this value will trigger an upgrade (if the current version does not match the desired version).

.spec.desiredUpdate

Description

desiredUpdate is an optional field that indicates the desired value of the cluster version. Setting this value will trigger an upgrade (if the current version does not match the desired version).

Type

`object`

Property	Type	Description
`force`	`boolean`	force allows an administrator to update to an image that has failed checks that are designed to keep the cluster safe. Use only if: • testing • working around a known bug
`image`	`string`	image is a container image reference that contains the update.
`version`	`string`	version is the container image tag that identifies the update.

.status

Description

status contains information about cluster upgrade.

Type

`object`

Property	Type	Description
`availableUpdates`	`array`	availableUpdates contains available updates. This list may be
`conditions`	`array`	conditions provides information about the cluster version. The condition "Ready" is set to true if the desiredUpdate has been reached. The condition "Reconciling" is set to true if an update is being applied. The condition "Degraded" is set to true if an update is currently blocked by a temporary or permanent error. Conditions are only valid for the current desiredUpdate when metadata.generation is equal to status.generation.
`current`	`object`	current is the currently applied release version and image of the cluster.
`desired`	`object`	desired is the version that the cluster is reconciling towards. If the cluster is not yet fully initialized, desired will be set with the information available, which may be an image or a tag.
`history`	`array`	history contains a list of the most recent versions applied to the cluster. This value may be empty during cluster startup, and then will be updated when a new update is being applied. The newest update is first in the list and it is ordered by recency. Updates in the history have state

Property	Type	Description
		Completed if the rollout completed - if an update was failing or halfway applied the state will be Partial. Only a limited amount of update history is preserved.
`observedGeneration`	`integer`	observedGeneration of the object. When this matches the object's metadata.generation, it indicates the status is up-to-date.
`preflight`	`object`	preflight records information used during an in-progress upgrade to support deterministic preflight reentry behavior.
`stages`	`array`	stages expose the execution stages and per-operation DAG for the current update. Stages are executed sequentially; operations within a stage form a DAG.

.status.availableUpdates

Description

availableUpdates contains available updates. This list may be

Type

`array`

.status.availableUpdates[]

Description

Release represents an ACP release image and associated metadata.

Type

`object`

Property	Type	Description
`image`	`string`	image is a container image location that contains the update. When this field is part of spec, image is optional if version is specified.
`version`	`string`	version is a semantic version identifying the update version. When this field is part of spec, version is optional if image is specified.

.status.conditions

Description

conditions provides information about the cluster version. The condition "Ready" is set to true if the desiredUpdate has been reached. The condition "Reconciling" is set to true if an update is being applied. The condition "Degraded" is set to true if an update is currently blocked by a temporary or permanent error. Conditions are only valid for the current desiredUpdate when metadata.generation is equal to status.generation.

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

`object`

Required

`lastTransitionTime` `message` `reason` `status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
`message`	`string`	message is a human readable message indicating details about the transition. This may be an empty string.
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.

Property	Type	Description
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase.

.status.current

Description

current is the currently applied release version and image of the cluster.

Type

`object`

Property	Type	Description
`image`	`string`	image is a container image location that contains the update. When this field is part of spec, image is optional if version is specified.
`version`	`string`	version is a semantic version identifying the update version. When this field is part of spec, version is optional if image is specified.

.status.desired

Description

desired is the version that the cluster is reconciling towards. If the cluster is not yet fully initialized, desired will be set with the information available, which may be an image or a tag.

Type

`object`

Property	Type	Description
`image`	`string`	image is a container image location that contains the update. When this field is part of spec, image is optional if version is specified.
`version`	`string`	version is a semantic version identifying the update version. When this field is part of spec, version is optional if image is specified.

.status.history

Description

history contains a list of the most recent versions applied to the cluster. This value may be empty during cluster startup, and then will be updated when a new update is being applied. The newest update is first in the list and it is ordered by recency. Updates in the history have state Completed if the rollout completed - if an update was failing or halfway applied the state will be Partial. Only a limited amount of update history is preserved.

Type

`array`

.status.history[]

Description

UpdateHistory is a single attempted update to the cluster.

Type

`object`

Required

`image` `startedTime` `state`

Property	Type	Description
`acceptedRisks`	`string`	acceptedRisks records risks which were accepted to initiate the update.
`completionTime`	`string`	completionTime if set, is when the update was fully applied. The update that is currently being applied will have a null completion time. Completion time will always be set for entries that are not the current update (usually to the started time of the next update).
`image`	`string`	image is a container image location that contains the update. This value is always populated.
`startedTime`	`string`	startedTime is the time at which the update was started.
`state`	`string`	state reflects whether the update was fully applied. The Partial state indicates the update is not fully applied, while the Completed state indicates the update was successfully rolled out at least once (all parts of the update successfully applied).
`version`	`string`	version is a semantic version identifying the update version. If the requested image does not define a

Property	Type	Description
		version, or if a failure occurs retrieving the image, this value may be empty.

.status.preflight

Description

preflight records information used during an in-progress upgrade to support deterministic preflight reentry behavior.

Type

`object`

Property	Type	Description
`checks`	`array`	checks records the latest evaluation result for each preflight check.
`observedAt`	`string`	observedAt is the time when preflight was last evaluated.

.status.preflight.checks

Description

checks records the latest evaluation result for each preflight check.

Type

`array`

.status.preflight.checks[]

Description

PreflightCheckStatus records the latest result of a single preflight check.

Type

`object`

Required

`name`

Property	Type	Description
`message`	`string`	message contains human-readable details for the last result.
`name`	`string`	name is the preflight check name.
`policy`	`string`	policy indicates how the check behaves on reentry.
`reason`	`string`	reason is a programmatic identifier for the last result.
`state`	`string`	state is the latest result state for the check.

.status.stages

Description

stages expose the execution stages and per-operation DAG for the current update. Stages are executed sequentially; operations within a stage form a DAG.

Type

`array`

.status.stages[]

Type

`object`

Required

`name`

Property	Type	Description
`name`	`string`	name is the stage display name.
`operations`	`array`	operations are the DAG nodes in this stage.
`phase`	`string`	phase is the lifecycle status of this stage.
`priority`	`integer`	priority is the stage execution order (ascending).

.status.stages[].operations

Description

operations are the DAG nodes in this stage.

Type

`array`

.status.stages[].operations[]

Type

`object`

Required

`name`

Property	Type	Description
`action`	`string`	action is Install/Upgrade/Delete (omit for no-op/unknown).
`currentVersion`	`string`	currentVersion is the observed value at build/progress time.
`dependsOn`	`array`	dependsOn lists direct predecessor operation names within this stage.
`moduleInfo`	`string`	moduleInfo is the backing ModuleInfo name (if any).
`name`	`string`	name is the operation/component identifier (stable ID).
`phase`	`string`	phase is the current operation phase string (e.g. Running/UpgradePending/Failed).
`targetVersion`	`string`	targetVersion is the desired value at build/progress time.

.status.stages[].operations[].dependsOn

Description

dependsOn lists direct predecessor operation names within this stage.

Type

`array`

.status.stages[].operations[].dependsOn[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows`
  - `DELETE`: delete collection of ClusterVersionShadow
  - `GET`: list objects of kind ClusterVersionShadow
  - `POST`: create a new ClusterVersionShadow
- `/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows/{name}`
  - `DELETE`: delete the specified ClusterVersionShadow
  - `GET`: read the specified ClusterVersionShadow
  - `PATCH`: partially update the specified ClusterVersionShadow
  - `PUT`: replace the specified ClusterVersionShadow
- `/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows/{name}/status`
  - `GET`: read status of the specified ClusterVersionShadow
  - `PATCH`: partially update status of the specified ClusterVersionShadow
  - `PUT`: replace status of the specified ClusterVersionShadow

/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows

HTTP method

`DELETE`

Description

delete collection of ClusterVersionShadow

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ClusterVersionShadow

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadowList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ClusterVersionShadow` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
201 - Created	`ClusterVersionShadow` schema

HTTP code	Response body
202 - Accepted	`ClusterVersionShadow` schema
401 - Unauthorized	Empty

/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows/{name}

HTTP method

`DELETE`

Description

delete the specified ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ClusterVersionShadow

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ClusterVersionShadow` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
201 - Created	`ClusterVersionShadow` schema
401 - Unauthorized	Empty

/apis/config.alauda.io/v1/namespaces/{namespace}/clusterversionshadows/{name}/status

HTTP method

`GET`

Description

read status of the specified ClusterVersionShadow

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ClusterVersionShadow

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ClusterVersionShadow` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ClusterVersionShadow` schema
201 - Created	`ClusterVersionShadow` schema
401 - Unauthorized	Empty