ModulePlugin APIs

ModuleConfig [moduleconfigs.

ModuleInfo [moduleinfoes.clus

ModulePlug

ModuleConfig [moduleconfigs.cluster.alauda.io/v1alpha1]

Description

ModuleConfig is the Schema for the moduleconfigs API

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ModuleConfigSpec defines the desired state of ModuleConfig
`status`	`object`	ModuleConfigStatus defines the observed state of ModuleConfig

.spec

Description

ModuleConfigSpec defines the desired state of ModuleConfig

Type

`object`

Required

`deleteable` `description` `logo` `name`

Property	Type	Description
`affinity`	`object`	Affinity is the affinity of the plugin.

Property	Type	Description
`appReleases`	`array`	AppReleases is the app releases of the module.
`cfgFilename`	`string`	CfgFilename is the plugin config filename in chart.
`config`	`object`	Config is the config of the module.
`crossClusterDependencies`	`array`	CrossClusterDependencies indicates the cross-cluster dependencies of this module. Deprecated
`deleteBlockDescription`	`string`	DeleteBlockDescription is the delete block description of the module.
`deleteBlockDescriptionEn`	`string`	DeleteBlockDescriptionEn is the delete block description of the module in English.
`deleteRiskDescription`	`string`	DeleteRiskDescription is the delete risk description of the module.
`deleteRiskDescriptionEn`	`string`	DeleteRiskDescriptionEn is the delete risk description of the module in English.

Property	Type	Description
`deleteable`	`boolean`	Deleteable indicates whether the module can be deleted.
`dependencies`	`array`	Dependencies is the dependencies of the module.
`deployDescriptors`	`array`	DeployDescriptors is the descriptors of the module for deployment.
`description`	`object`	Description is the description of the module.
`editDescriptors`	`array`	EditDescriptors is the descriptors of the module for edit.
`entrypointTemplate`	`string`	EntrypointTemplate is the entrypoint template of the module.
`labelCluster`	`string`	LabelCluster is the flag of whether label cluster resource.
`logo`	`string`	Logo is the logo of the module.

Property	Type	Description
`mainChart`	`string`	MainChart is the main chart of the module.
`mustUpgrade`	`boolean`	MustUpgrade is the flag of whether the module must upgrade.
`name`	`string`	Name is the name of the module.
`pluginDependencies`	`object`	PluginDependencies define the plugins dependent relationship
`resourcesBlockRemove`	`array`	ResourcesBlockRemove stands for the resources reference with can block remove this plugin, all of theses resources must be deleted before you can remove this plugin.
`skipPlatformConfig`	`boolean`	SkipPlatformConfig stands for skip reference the platform values when renderering the values for AppRelease.
`supportedUpgradeVersions`	`string`	SupportedUpgradeVersions is the supported old versions of the module for upgrade.

Property	Type	Description
`targetClusterVersions`	`array`	TargetClusterVersions is the target clusters of the module.
`transitTimeout`	`integer`	TransitTimeout is the timeout duration for upgrade.
`uiContext`	`object`	UIContext is the additional attributes for module config.
`upgradeBefore`	`array`	UpgradeBefore stands for need upgrade before these mouldes
`upgradeDescriptors`	`array`	UpgradeDescriptors is the descriptors of the module for upgrade.
`upgradeRiskDescription`	`string`	UpgradeRiskDescription is the upgrade risk description of the module.
`upgradeRiskDescriptionEn`	`string`	UpgradeRiskDescriptionEn is the upgrade risk description of the module in English.
`upgradeRiskLevel`	`string`	UpgradeRiskLevel is the upgrade risk level of the module.

Property	Type	Description
`valuesTemplates`	`object`	ValuesTemplates is the values templates of the module.
`version`	`string`	Version is the version of the module.

.spec.affinity

Description

Affinity is the affinity of the plugin.

Type

`object`

Property	Type	Description
`clusterAffinity`	`object`	ClusterAffinity defines the cluster affinity
`clusterAntiAffinity`	`object`	ClusterAntiAffinity defines the cluster anti-affinity
`pluginAntiAffinity`	`array`	PluginAntiAffinity defines the plugin anti-affinity

.spec.affinity.clusterAffinity

Description

ClusterAffinity defines the cluster affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.spec.affinity.clusterAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.spec.affinity.clusterAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

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

.spec.affinity.clusterAffinity.matchExpressions[].values

Description

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

Type

`array`

.spec.affinity.clusterAffinity.matchExpressions[].values[]

Type

`string`

.spec.affinity.clusterAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.spec.affinity.clusterAntiAffinity

Description

ClusterAntiAffinity defines the cluster anti-affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.spec.affinity.clusterAntiAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.spec.affinity.clusterAntiAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.affinity.clusterAntiAffinity.matchExpressions[].values

Description

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

Type

`array`

.spec.affinity.clusterAntiAffinity.matchExpressions[].values[]

Type

`string`

.spec.affinity.clusterAntiAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.spec.affinity.pluginAntiAffinity

Description

PluginAntiAffinity defines the plugin anti-affinity

Type

`array`

.spec.affinity.pluginAntiAffinity[]

Type

`string`

.spec.appReleases

Description

AppReleases is the app releases of the module.

Type

`array`

.spec.appReleases[]

Description

AppRelease is the release of an application

Type

`object`

Required

`chartVersions` `name`

Property	Type	Description
`chartVersions`	`array`	ChartVersions stands for the app's chart versions
`installCondition`	`string`	InstallCondition is the condition of installing
`name`	`string`	Name stands for the release name
`namespace`	`string`	Namespace stands for the release namespace

.spec.appReleases[].chartVersions

Description

ChartVersions stands for the app's chart versions

Type

`array`

.spec.appReleases[].chartVersions[]

Description

ChartVersion is the version of the chart

Type

`object`

Required

`name` `releaseName` `version`

Property	Type	Description
`name`	`string`	Name stands for the chart's name
`releaseName`	`string`	ReleaseName stands for the release name
`version`	`string`	Version stands for the chart's version

.spec.config

Description

Config is the config of the module.

Type

`object`

.spec.crossClusterDependencies

Description

CrossClusterDependencies indicates the cross-cluster dependencies of this module.
Deprecated

Type

`array`

.spec.crossClusterDependencies[]

Type

`string`

.spec.dependencies

Description

Dependencies is the dependencies of the module.

Type

`array`

.spec.dependencies[]

Description

ModuleDependency defines the module dependency

Type

`object`

Required

`moduleName`

Property	Type	Description
`instanceName`	`string`	InstanceName stands for the dependent module's ModuleInfo CR's name
`moduleName`	`string`	ModuleName stands for the dependent module's name

Property	Type	Description
`versionsMatch`	`array`	VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

.spec.dependencies[].versionsMatch

Description

VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

Type

`array`

.spec.dependencies[].versionsMatch[]

Type

`string`

.spec.deployDescriptors

Description

DeployDescriptors is the descriptors of the module for deployment.

Type

`array`

.spec.deployDescriptors[]

Description

Descriptors defines OLM descriptors http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.deployDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.deployDescriptors[].x-descriptors[]

Type

`string`

.spec.description

Description

Description is the description of the module.

Type

`object`

Property	Type	Description
`en`	`string`	En is the English description
`zh`	`string`	Zh is the Chinese description

.spec.editDescriptors

Description

EditDescriptors is the descriptors of the module for edit.

Type

`array`

.spec.editDescriptors[]

Description

Descriptors defines OLM descriptors
http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.editDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.editDescriptors[].x-descriptors[]

Type

`string`

.spec.pluginDependencies

Description

PluginDependencies define the plugins dependent relationship

Type

`object`

Property	Type	Description
`crossCluster`	`object`	

.spec.pluginDependencies.crossCluster

Type

`object`

Property	Type	Description
`oneOf`	`array`	

.spec.pluginDependencies.crossCluster.oneOf

Type

`array`

.spec.pluginDependencies.crossCluster.oneOf[]

Type

`object`

Required

`clusterName` `pluginName`

Property	Type	Description
`clusterName`	`string`	
`pluginName`	`string`	

.spec.resourcesBlockRemove

Description

ResourcesBlockRemove stands for the resources reference with can block remove this plugin, all of theses resources must be deleted before you can remove this plugin.

Type

`array`

.spec.resourcesBlockRemove[]

Description

ResourceReference defines the resource reference

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`kind`	`string`	Kind of the referent.
`name`	`string`	Name of the referent.
`namespace`	`string`	Namespace of the referent.

.spec.targetClusterVersions

Description

TargetClusterVersions is the target clusters of the module.

Type

`array`

.spec.targetClusterVersions[]

Type

`string`

.spec.uiContext

Description

UIContext is the additional attributes for module config.

Type

`object`

Property	Type	Description
`sizeSettings`	`object`	SizeSettings defines a list of size config

.spec.uiContext.sizeSettings

Description

SizeSettings defines a list of size config

Type

`object`

Property	Type	Description
`default`	`string`	Default is name for default size

Property	Type	Description
`quotaTip`	`object`	QuotaTip is quota tip
`sizes`	`array`	Sizes defines a list of size config

.spec.uiContext.sizeSettings.quotaTip

Description

QuotaTip is quota tip

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.uiContext.sizeSettings.sizes

Description

Sizes defines a list of size config

Type

`array`

.spec.uiContext.sizeSettings.sizes[]

Description

SizeConfig defines a list of components config

Type

`object`

Required

`components` `description` `name` `size`

Property	Type	Description
`components`	`object`	Components defines a list of component config
`description`	`object`	Description defines description for this size config
`name`	`object`	Name defines name for this size config
`size`	`string`	Size is name for this config

.spec.uiContext.sizeSettings.sizes[].components

Description

Components defines a list of component config

Type

`object`

.spec.uiContext.sizeSettings.sizes[].description

Description

Description defines description for this size config

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.uiContext.sizeSettings.sizes[].name

Description

Name defines name for this size config

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.upgradeBefore

Description

UpgradeBefore stands for need upgrade before these mouldes

Type

`array`

.spec.upgradeBefore[]

Description

ModuleRef defines the reference for the module

Type

`object`

Required

`moduleName`

Property	Type	Description
`moduleName`	`string`	ModuleName stands for the dependent module's name

.spec.upgradeDescriptors

Description

UpgradeDescriptors is the descriptors of the module for upgrade.

Type

`array`

.spec.upgradeDescriptors[]

Description

Descriptors defines OLM descriptors
http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.upgradeDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.upgradeDescriptors[].x-descriptors[]

Type

`string`

.spec.valuesTemplates

Description

ValuesTemplates is the values templates of the module.

Type

`object`

.status

Description

ModuleConfigStatus defines the observed state of ModuleConfig

Type

`object`

Property	Type	Description
`readyForDeploy`	`boolean`	ReadyForDeploy indicates whether the module is ready for deployment.
`targetClusterVersions`	`array`	TargetClusterVersions is the target clusters of the module.

.status.targetClusterVersions

Description

TargetClusterVersions is the target clusters of the module.

Type

`array`

.status.targetClusterVersions[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/cluster.alauda.io/v1alpha1/moduleconfigs`
  - `DELETE`: delete collection of ModuleConfig
  - `GET`: list objects of kind ModuleConfig
  - `POST`: create a new ModuleConfig
- `/apis/cluster.alauda.io/v1alpha1/moduleconfigs/{name}`
  - `DELETE`: delete the specified ModuleConfig
  - `GET`: read the specified ModuleConfig
  - `PATCH`: partially update the specified ModuleConfig
  - `PUT`: replace the specified ModuleConfig
- `/apis/cluster.alauda.io/v1alpha1/moduleconfigs/{name}/status`
  - `GET`: read status of the specified ModuleConfig
  - `PATCH`: partially update status of the specified ModuleConfig
  - `PUT`: replace status of the specified ModuleConfig

/apis/cluster.alauda.io/v1alpha1/moduleconfigs

HTTP method

`DELETE`

Description

delete collection of ModuleConfig

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind ModuleConfig

HTTP responses

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

HTTP method

`POST`

Description

create a new ModuleConfig

Query parameters

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

Parameter	Type	Description
		fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

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

HTTP responses

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

/apis/cluster.alauda.io/v1alpha1/moduleconfigs/{name}

HTTP method

`DELETE`

Description

delete the specified ModuleConfig

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

Parameter	Type	Description
		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 ModuleConfig

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified ModuleConfig

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

HTTP responses

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

HTTP method

`PUT`

Description

replace the specified ModuleConfig

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`	`ModuleConfig` schema	`application/json` formatted

HTTP responses

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

HTTP code	Response body
401 - Unauthorized	Empty

/apis/cluster.alauda.io/v1alpha1/moduleconfigs/{name}/status

HTTP method

`GET`

Description

read status of the specified ModuleConfig

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ModuleConfig

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

Parameter	Type	Description
		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	`ModuleConfig` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ModuleConfig

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`	`ModuleConfig` schema	`application/json` formatted

HTTP responses

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

HTTP code	Response body
401 - Unauthorized	Empty

ModuleInfo [moduleinfoes.cluster.alauda.io/v1alpha1]

Description

ModuleInfo is the Schema for the moduleinfos API

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ModuleInfoSpec defines the spec of ModuleInfo
`status`	`object`	ModuleInfoStatus defines the observed state of ModuleInfo

.spec

Description

ModuleInfoSpec defines the spec of ModuleInfo

Type

`object`

Required

`version`

Property	Type	Description
`config`	`object`	Config stands for the user configs of the module instance

Property	Type	Description
`entrypoint`	`string`	Entrypoint stands for the entrypoint on the web GUI for this module instance
`selectedCluster`	`object`	SelectedCluster stands for the selected cluster of the module instance
`valuesOverride`	`object`	ValuesOverride stands for the overrides for charts
`version`	`string`	Version stands for the target version of the module

.spec.config

Description

Config stands for the user configs of the module instance

Type

`object`

.spec.selectedCluster

Description

SelectedCluster stands for the selected cluster of the module instance

Type

`object`

.spec.valuesOverride

Description

ValuesOverride stands for the overrides for charts

Type

`object`

.status

Description

ModuleInfoStatus defines the observed state of ModuleInfo

Type

`object`

Required

`phase` `version`

Property	Type	Description
`appReleases`	`array`	AppReleases stands for the module's app releases
`availableVersions`	`array`	AvailableVersions stands for the available versions of the module can upgrade to.
`blockReasons`	`array`	BlockReasons stands for the reasons blocked upgrade
`conditions`	`array`	Conditions stands for the moudle's conditions

Property	Type	Description
`moduleconfigSpecHash`	`string`	Hash stands for the has of ModuleConfig Spec
`phase`	`string`	Phase stands for the module's status
`removeIsPrevented`	`boolean`	RemoveIsPrevented stands for you can not remove this plugin right now for some reason.
`specHash`	`string`	Hash stands for the has of ModuleInfo Spec
`transitTimeout`	`integer`	TransitTimeout is the timeout duration for upgrade.
`transitTimestamp`	`string`	TransitTimestamp stands for the timestamp of the version changed
`transitVersion`	`string`	TransitVersion stands for the module's transit version used for upgrade
`version`	`string`	Version stands for the module's current version

.status.appReleases

Description

AppReleases stands for the module's app releases

Type

`array`

.status.appReleases[]

Description

AppReleaseStatus defines the status of the app release

Type

`object`

Required

`failed` `name` `ready` `synced`

Property	Type	Description
`chartVersions`	`array`	
`failed`	`boolean`	
`lastProbeTime`	`string`	
`message`	`string`	
`name`	`string`	
`namespace`	`string`	
`ready`	`boolean`	
`reason`	`string`	
`synced`	`boolean`	

.status.appReleases[].chartVersions

Type

`array`

.status.appReleases[].chartVersions[]

Description

ChartVersion stands for chart version

Type

`object`

Required

`name` `releaseName` `version`

Property	Type	Description
`name`	`string`	Name stands for the chart's name
`releaseName`	`string`	ReleaseName stands for the release name
`version`	`string`	Version stands for the chart's version

.status.availableVersions

Description

AvailableVersions stands for the available versions of the module can upgrade to.

Type

`array`

.status.availableVersions[]

Description

ModuleAvailableVersion defines module's available upgrade version

Type

`object`

Required

`description` `riskLevel` `version`

Property	Type	Description
`dependencies`	`array`	Dependencies stands for the dependent modules
`description`	`string`	Description stands for the description of uprading to this version
`descriptionEn`	`string`	DescriptionEn stands for the description of uprading to this version in English
`options`	`object`	Options stands for the options of uprading to this version
`riskLevel`	`string`	RiskLevel stands for the risk level of uprading to this version
`upgradeBefore`	`array`	UpgradeBefore stands for need upgrade before these mouldes
`version`	`string`	Version stands for the version of the module can upgrade to

.status.availableVersions[].dependencies

Description

Dependencies stands for the dependent modules

Type

`array`

.status.availableVersions[].dependencies[]

Description

ModuleDependency defines the module's dependency

Type

`object`

Required

`moduleName`

Property	Type	Description
`instanceName`	`string`	InstanceName stands for the dependent module's ModuleInfo CR's name
`moduleName`	`string`	ModuleName stands for the dependent module's name
`versionsMatch`	`array`	VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

.status.availableVersions[].dependencies[].versionsMatch

Description

VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

Type

`array`

.status.availableVersions[].dependencies[].versionsMatch[]

Type

`string`

.status.availableVersions[].options

Description

Options stands for the options of uprading to this version

Type

`object`

.status.availableVersions[].upgradeBefore

Description

UpgradeBefore stands for need upgrade before these mouldes

Type

`array`

.status.availableVersions[].upgradeBefore[]

Description

ModuleRef defines the reference for the module

Type

`object`

Required

`moduleName`

Property	Type	Description
`moduleName`	`string`	ModuleName stands for the dependent module's name

.status.blockReasons

Description

BlockReasons stands for the reasons blocked upgrade

Type

`array`

.status.blockReasons[]

Type

`string`

.status.conditions

Description

Conditions stands for the moudle's conditions

Type

`array`

.status.conditions[]

Description

ModuleCondition defines the module's conditions

Type

`object`

Required

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

Property	Type	Description
`lastProbeTime`	`string`	LastProbeTime stands for the timestamp to probe status
`lastTransitionTime`	`string`	LastTransitionTime stands for the timestamp of the status changed
`message`	`string`	Message contains the detail information why in the current status
`reason`	`string`	Reason contains the reason why in the current status
`status`	`string`	Status stands for condition's status
`type`	`string`	Type stands for condition's type

API Endpoints

The following API endpoints are available:

- `/apis/cluster.alauda.io/v1alpha1/moduleinfos`

- `DELETE` : delete collection of ModuleInfo
- `GET` : list objects of kind ModuleInfo
- `POST` : create a new ModuleInfo

- `/apis/cluster.alauda.io/v1alpha1/moduleinfos/{name}`
  - `DELETE` : delete the specified ModuleInfo
  - `GET` : read the specified ModuleInfo
  - `PATCH` : partially update the specified ModuleInfo
  - `PUT` : replace the specified ModuleInfo
- `/apis/cluster.alauda.io/v1alpha1/moduleinfos/{name}/status`
  - `GET` : read status of the specified ModuleInfo
  - `PATCH` : partially update status of the specified ModuleInfo
  - `PUT` : replace status of the specified ModuleInfo

/apis/cluster.alauda.io/v1alpha1/moduleinfos

HTTP method

`DELETE`

Description

delete collection of ModuleInfo

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind ModuleInfo

HTTP responses

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

HTTP method

`POST`

Description

create a new ModuleInfo

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.

Body parameters

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

HTTP responses

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

/apis/cluster.alauda.io/v1alpha1/moduleinfos/{name}

HTTP method

`DELETE`

Description

delete the specified ModuleInfo

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 ModuleInfo

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified ModuleInfo

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	`ModuleInfo` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ModuleInfo

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`	`ModuleInfo` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ModuleInfo` schema

HTTP code	Response body
201 - Created	`ModuleInfo` schema
401 - Unauthorized	Empty

/apis/cluster.alauda.io/v1alpha1/moduleinfos/{name}/status

HTTP method

`GET`

Description

read status of the specified ModuleInfo

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ModuleInfo

Query parameters

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace status of the specified ModuleInfo

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

Body parameters

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

HTTP responses

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

ModulePlugin [moduleplugins.cluster.alauda.io/v1alpha1]

Description

ModuleConfig is the Schema for the moduleconfigs API

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ModuleConfigSpec defines the desired state of ModuleConfig
`status`	`object`	ModulePluginStatus defines the observed state of ModulePlugin

.spec

Description

ModuleConfigSpec defines the desired state of ModuleConfig

Type

`object`

Required

`deleteable` `description` `logo` `name`

Property	Type	Description
`affinity`	`object`	Affinity is the affinity of the plugin.

Property	Type	Description
`appReleases`	`array`	AppReleases is the app releases of the module.
`cfgFilename`	`string`	CfgFilename is the plugin config filename in chart.
`config`	`object`	Config is the config of the module.
`crossClusterDependencies`	`array`	CrossClusterDependencies indicates the cross-cluster dependencies of this module. Deprecated
`deleteBlockDescription`	`string`	DeleteBlockDescription is the delete block description of the module.
`deleteBlockDescriptionEn`	`string`	DeleteBlockDescriptionEn is the delete block description of the module in English.
`deleteRiskDescription`	`string`	DeleteRiskDescription is the delete risk description of the module.
`deleteRiskDescriptionEn`	`string`	DeleteRiskDescriptionEn is the delete risk description of the module in English.

Property	Type	Description
`deleteable`	`boolean`	Deleteable indicates whether the module can be deleted.
`dependencies`	`array`	Dependencies is the dependencies of the module.
`deployDescriptors`	`array`	DeployDescriptors is the descriptors of the module for deployment.
`description`	`object`	Description is the description of the module.
`editDescriptors`	`array`	EditDescriptors is the descriptors of the module for edit.
`entrypointTemplate`	`string`	EntrypointTemplate is the entrypoint template of the module.
`labelCluster`	`string`	LabelCluster is the flag of whether label cluster resource.
`logo`	`string`	Logo is the logo of the module.

Property	Type	Description
`mainChart`	`string`	MainChart is the main chart of the module.
`mustUpgrade`	`boolean`	MustUpgrade is the flag of whether the module must upgrade.
`name`	`string`	Name is the name of the module.
`pluginDependencies`	`object`	PluginDependencies define the plugins dependent relationship
`resourcesBlockRemove`	`array`	ResourcesBlockRemove stands for the resources reference with can block remove this plugin, all of theses resources must be deleted before you can remove this plugin.
`skipPlatformConfig`	`boolean`	SkipPlatformConfig stands for skip reference the platform values when renderering the values for AppRelease.
`supportedUpgradeVersions`	`string`	SupportedUpgradeVersions is the supported old versions of the module for upgrade.

Property	Type	Description
`targetClusterVersions`	`array`	TargetClusterVersions is the target clusters of the module.
`transitTimeout`	`integer`	TransitTimeout is the timeout duration for upgrade.
`uiContext`	`object`	UIContext is the additional attributes for module config.
`upgradeBefore`	`array`	UpgradeBefore stands for need upgrade before these mouldes
`upgradeDescriptors`	`array`	UpgradeDescriptors is the descriptors of the module for upgrade.
`upgradeRiskDescription`	`string`	UpgradeRiskDescription is the upgrade risk description of the module.
`upgradeRiskDescriptionEn`	`string`	UpgradeRiskDescriptionEn is the upgrade risk description of the module in English.
`upgradeRiskLevel`	`string`	UpgradeRiskLevel is the upgrade risk level of the module.

Property	Type	Description
`valuesTemplates`	`object`	ValuesTemplates is the values templates of the module.
`version`	`string`	Version is the version of the module.

.spec.affinity

Description

Affinity is the affinity of the plugin.

Type

`object`

Property	Type	Description
`clusterAffinity`	`object`	ClusterAffinity defines the cluster affinity
`clusterAntiAffinity`	`object`	ClusterAntiAffinity defines the cluster anti-affinity
`pluginAntiAffinity`	`array`	PluginAntiAffinity defines the plugin anti-affinity

.spec.affinity.clusterAffinity

Description

ClusterAffinity defines the cluster affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.spec.affinity.clusterAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.spec.affinity.clusterAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

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

.spec.affinity.clusterAffinity.matchExpressions[].values

Description

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

Type

`array`

.spec.affinity.clusterAffinity.matchExpressions[].values[]

Type

`string`

.spec.affinity.clusterAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.spec.affinity.clusterAntiAffinity

Description

ClusterAntiAffinity defines the cluster anti-affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.spec.affinity.clusterAntiAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.spec.affinity.clusterAntiAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.affinity.clusterAntiAffinity.matchExpressions[].values

Description

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

Type

`array`

.spec.affinity.clusterAntiAffinity.matchExpressions[].values[]

Type

`string`

.spec.affinity.clusterAntiAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.spec.affinity.pluginAntiAffinity

Description

PluginAntiAffinity defines the plugin anti-affinity

Type

`array`

.spec.affinity.pluginAntiAffinity[]

Type

`string`

.spec.appReleases

Description

AppReleases is the app releases of the module.

Type

`array`

.spec.appReleases[]

Description

AppRelease is the release of an application

Type

`object`

Required

`chartVersions` `name`

Property	Type	Description
`chartVersions`	`array`	ChartVersions stands for the app's chart versions
`installCondition`	`string`	InstallCondition is the condition of installing
`name`	`string`	Name stands for the release name
`namespace`	`string`	Namespace stands for the release namespace

.spec.appReleases[].chartVersions

Description

ChartVersions stands for the app's chart versions

Type

`array`

.spec.appReleases[].chartVersions[]

Description

ChartVersion is the version of the chart

Type

`object`

Required

`name` `releaseName` `version`

Property	Type	Description
`name`	`string`	Name stands for the chart's name
`releaseName`	`string`	ReleaseName stands for the release name
`version`	`string`	Version stands for the chart's version

.spec.config

Description

Config is the config of the module.

Type

`object`

.spec.crossClusterDependencies

Description

CrossClusterDependencies indicates the cross-cluster dependencies of this module.
Deprecated

Type

`array`

.spec.crossClusterDependencies[]

Type

`string`

.spec.dependencies

Description

Dependencies is the dependencies of the module.

Type

`array`

.spec.dependencies[]

Description

ModuleDependency defines the module dependency

Type

`object`

Required

`moduleName`

Property	Type	Description
`instanceName`	`string`	InstanceName stands for the dependent module's ModuleInfo CR's name
`moduleName`	`string`	ModuleName stands for the dependent module's name

Property	Type	Description
`versionsMatch`	`array`	VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

.spec.dependencies[].versionsMatch

Description

VersionMatch stands for the version requirements for the dependent module, any match will be acceptable.

Type

`array`

.spec.dependencies[].versionsMatch[]

Type

`string`

.spec.deployDescriptors

Description

DeployDescriptors is the descriptors of the module for deployment.

Type

`array`

.spec.deployDescriptors[]

Description

Descriptors defines OLM descriptors http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.deployDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.deployDescriptors[].x-descriptors[]

Type

`string`

.spec.description

Description

Description is the description of the module.

Type

`object`

Property	Type	Description
`en`	`string`	En is the English description
`zh`	`string`	Zh is the Chinese description

.spec.editDescriptors

Description

EditDescriptors is the descriptors of the module for edit.

Type

`array`

.spec.editDescriptors[]

Description

Descriptors defines OLM descriptors
http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.editDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.editDescriptors[].x-descriptors[]

Type

`string`

.spec.pluginDependencies

Description

PluginDependencies define the plugins dependent relationship

Type

`object`

Property	Type	Description
`crossCluster`	`object`	

.spec.pluginDependencies.crossCluster

Type

`object`

Property	Type	Description
`oneOf`	`array`	

.spec.pluginDependencies.crossCluster.oneOf

Type

`array`

.spec.pluginDependencies.crossCluster.oneOf[]

Type

`object`

Required

`clusterName` `pluginName`

Property	Type	Description
`clusterName`	`string`	
`pluginName`	`string`	

.spec.resourcesBlockRemove

Description

ResourcesBlockRemove stands for the resources reference with can block remove this plugin, all of theses resources must be deleted before you can remove this plugin.

Type

`array`

.spec.resourcesBlockRemove[]

Description

ResourceReference defines the resource reference

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`kind`	`string`	Kind of the referent.
`name`	`string`	Name of the referent.
`namespace`	`string`	Namespace of the referent.

.spec.targetClusterVersions

Description

TargetClusterVersions is the target clusters of the module.

Type

`array`

.spec.targetClusterVersions[]

Type

`string`

.spec.uiContext

Description

UIContext is the additional attributes for module config.

Type

`object`

Property	Type	Description
`sizeSettings`	`object`	SizeSettings defines a list of size config

.spec.uiContext.sizeSettings

Description

SizeSettings defines a list of size config

Type

`object`

Property	Type	Description
`default`	`string`	Default is name for default size

Property	Type	Description
`quotaTip`	`object`	QuotaTip is quota tip
`sizes`	`array`	Sizes defines a list of size config

.spec.uiContext.sizeSettings.quotaTip

Description

QuotaTip is quota tip

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.uiContext.sizeSettings.sizes

Description

Sizes defines a list of size config

Type

`array`

.spec.uiContext.sizeSettings.sizes[]

Description

SizeConfig defines a list of components config

Type

`object`

Required

`components` `description` `name` `size`

Property	Type	Description
`components`	`object`	Components defines a list of component config
`description`	`object`	Description defines description for this size config
`name`	`object`	Name defines name for this size config
`size`	`string`	Size is name for this config

.spec.uiContext.sizeSettings.sizes[].components

Description

Components defines a list of component config

Type

`object`

.spec.uiContext.sizeSettings.sizes[].description

Description

Description defines description for this size config

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.uiContext.sizeSettings.sizes[].name

Description

Name defines name for this size config

Type

`object`

Property	Type	Description
`en`	`string`	En is the English text
`zh`	`string`	Zh is the Chinese text

.spec.upgradeBefore

Description

UpgradeBefore stands for need upgrade before these mouldes

Type

`array`

.spec.upgradeBefore[]

Description

ModuleRef defines the reference for the module

Type

`object`

Required

`moduleName`

Property	Type	Description
`moduleName`	`string`	ModuleName stands for the dependent module's name

.spec.upgradeDescriptors

Description

UpgradeDescriptors is the descriptors of the module for upgrade.

Type

`array`

.spec.upgradeDescriptors[]

Description

Descriptors defines OLM descriptors
http://confluence.alauda.cn/display/cp/OLM+Descriptor

Type

`object`

Required

`path` `x-descriptors`

Property	Type	Description
`description`	`string`	Description is the description of the descriptor
`displayName`	`string`	DisplayName is the display name of the descriptor
`path`	`string`	Path is the path of the descriptor
`x-descriptors`	`array`	X-Descriptors is the list of descriptors

.spec.upgradeDescriptors[].x-descriptors

Description

X-Descriptors is the list of descriptors

Type

`array`

.spec.upgradeDescriptors[].x-descriptors[]

Type

`string`

.spec.valuesTemplates

Description

ValuesTemplates is the values templates of the module.

Type

`object`

.status

Description

ModulePluginStatus defines the observed state of ModulePlugin

Type

`object`

Property	Type	Description
`conditions`	`array`	conditions is the conditions on this plugin
`installed`	`array`	installed are the installed instances of this plugin
`latestVersion`	`string`	latestVersion is the latest version of the ModuleConfig
`moduleConfigs`	`array`	moduleConfigs is the module config versions of the ModulePlugin
`specHash`	`string`	SpecHash is the hash of the spec
`targetClusterVersions`	`object`	targetClusterVersions is the target cluster versions of the ModuleConfig

.status.conditions

Description

conditions is the conditions on this plugin

Type

`array`

.status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource. --- This struct is intended for direct use as an array at the field path .status.conditions. For example, type FooStatus struct{ // Represents the observations of a foo's current state. // Known .status.conditions.type are: "Available", "Progressing", and "Degraded" // +patchMergeKey=type // +patchStrategy=merge // +listType=map // +listMapKey=type Conditions []metav1.Condition `json:"conditions,omitempty" patchStrategy:"merge" patchMergeKey:"type" protobuf:"bytes,1,rep,name=conditions"` // other fields }

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.

Property	Type	Description
`observedGeneration`	`integer`	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
`reason`	`string`	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	type of condition in CamelCase or in foo.example.com/CamelCase. --- Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important. The regex it matches is (dns1123SubdomainFmt/)?(qualifiedNameFmt)

.status.installed

Description

installed are the installed instances of this plugin

Type

`array`

.status.installed[]

Description

InstalledPlugin defines the installed plugin's info

Type

`object`

Required

`cluster` `name` `phase` `version`

Property	Type	Description
`cluster`	`string`	
`name`	`string`	
`phase`	`string`	
`version`	`string`	

.status.moduleConfigs

Description

moduleConfigs is the module config versions of the ModulePlugin

Type

`array`

.status.moduleConfigs[]

Description

ModuleConfigStatus defines the module config of the plugin

Type

`object`

Required

`name` `readyForDeploy` `targetClusterVersions` `version`

Property	Type	Description
`affinity`	`object`	Affinity is the affinity of the plugin
`name`	`string`	Name is the name of the module config
`readyForDeploy`	`boolean`	ReadyForDeploy indicates whether the module is ready for deployment.
`targetClusterVersions`	`array`	TargetClusterVersions is the target cluster versions of the module config
`version`	`string`	Version is the version of the module config

.status.moduleConfigs[].affinity

Description

Affinity is the affinity of the plugin

Type

`object`

Property	Type	Description
`clusterAffinity`	`object`	ClusterAffinity defines the cluster affinity
`clusterAntiAffinity`	`object`	ClusterAntiAffinity defines the cluster anti-affinity
`pluginAntiAffinity`	`array`	PluginAntiAffinity defines the plugin anti-affinity

.status.moduleConfigs[].affinity.clusterAffinity

Description

ClusterAffinity defines the cluster affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.status.moduleConfigs[].affinity.clusterAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.status.moduleConfigs[].affinity.clusterAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.status.moduleConfigs[].affinity.clusterAffinity.matchExpressions[].values

Description

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

Type

`array`

.status.moduleConfigs[].affinity.clusterAffinity.matchExpressions[].values[]

Type

`string`

.status.moduleConfigs[].affinity.clusterAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.status.moduleConfigs[].affinity.clusterAntiAffinity

Description

ClusterAntiAffinity defines the cluster anti-affinity

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	MatchExpressions is the list of match expressions
`matchLabels`	`object`	MatchLabels is the cluster selector

.status.moduleConfigs[].affinity.clusterAntiAffinity.matchExpressions

Description

MatchExpressions is the list of match expressions

Type

`array`

.status.moduleConfigs[].affinity.clusterAntiAffinity.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

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

.status.moduleConfigs[].affinity.clusterAntiAffinity.matchExpressions[].values

Description

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

Type

`array`

.status.moduleConfigs[].affinity.clusterAntiAffinity.matchExpressions[].values[]

Type

`string`

.status.moduleConfigs[].affinity.clusterAntiAffinity.matchLabels

Description

MatchLabels is the cluster selector

Type

`object`

.status.moduleConfigs[].affinity.pluginAntiAffinity

Description

PluginAntiAffinity defines the plugin anti-affinity

Type

`array`

.status.moduleConfigs[].affinity.pluginAntiAffinity[]

Type

`string`

.status.moduleConfigs[].targetClusterVersions

Description

TargetClusterVersions is the target cluster versions of the module config

Type

`array`

.status.moduleConfigs[].targetClusterVersions[]

Type

`string`

.status.targetClusterVersions

Description

targetClusterVersions is the target cluster versions of the ModuleConfig

Type

`object`

API Endpoints

The following API endpoints are available:

- `/apis/cluster.alauda.io/v1alpha1/moduleplugins`
  - `DELETE`: delete collection of ModulePlugin
  - `GET`: list objects of kind ModulePlugin
  - `POST`: create a new ModulePlugin
- `/apis/cluster.alauda.io/v1alpha1/moduleplugins/{name}`
  - `DELETE`: delete the specified ModulePlugin
  - `GET`: read the specified ModulePlugin
  - `PATCH`: partially update the specified ModulePlugin
  - `PUT`: replace the specified ModulePlugin
- `/apis/cluster.alauda.io/v1alpha1/moduleplugins/{name}/status`
  - `GET`: read status of the specified ModulePlugin
  - `PATCH`: partially update status of the specified ModulePlugin
  - `PUT`: replace status of the specified ModulePlugin

/apis/cluster.alauda.io/v1alpha1/moduleplugins

HTTP method

`DELETE`

Description

delete collection of ModulePlugin

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind ModulePlugin

HTTP responses

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

HTTP method

`POST`

Description

create a new ModulePlugin

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`	`ModulePlugin` schema	`application/json` formatted

HTTP responses

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

/apis/cluster.alauda.io/v1alpha1/moduleplugins/{name}

HTTP method

`DELETE`

Description

delete the specified ModulePlugin

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 ModulePlugin

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified ModulePlugin

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	`ModulePlugin` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ModulePlugin

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`	`ModulePlugin` schema	`application/json` formatted

HTTP responses

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

/apis/cluster.alauda.io/v1alpha1/moduleplugins/{name}/status

HTTP method

`GET`

Description

read status of the specified ModulePlugin

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ModulePlugin

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

HTTP responses

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

HTTP method

`PUT`

Description

replace status of the specified ModulePlugin

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`	`ModulePlugin` schema	`application/json` formatted

HTTP responses

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

HTTP code	Response body
401 - Unauthorized	Empty