Operator APIs

AmlCluster [amlclusters.aml.dev/v1alpha1]

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`	
`status`	`object`	

.spec

Type

`object`

Required

`values`

Property	Type	Description
`amlExperimental`	`object`	
`components`	`object`	
`values`	`object`	

.spec.amlExperimental

Type

`object`

.spec.components

Type

`object`

Property	Type	Description
`knativeServing`	`object`	Knative-Serving component configuration. Requires KServeless operator to be installed. Requires Service Mesh (istio).
`kserve`	`object`	Kserve component configuration. Requires KServeless operator to be installed.

.spec.components.knativeServing

Description

Knative-Serving component configuration. Requires KServeless operator to be installed. Requires Service Mesh (istio).

Type

`object`

Property	Type	Description
`ingressGateway`	`object`	IngressGateway allows to customize some parameters for the Istio Ingress Gateway that is bound to KNative-Serving.
`managementState`	`string`	
`namespace`	`string`	
`values`	`object`	

.spec.components.knativeServing.ingressGateway

Description

IngressGateway allows to customize some parameters for the Istio Ingress Gateway that is bound to KNative-Serving.

Type

`object`

Required

`domain`

Property	Type	Description
`certificate`	`object`	Certificate specifies configuration of the TLS certificate securing communication for the gateway.
`domain`	`string`	Domain specifies the host name for intercepting incoming requests. Most likely, you will want to use a wildcard name, like *.example.com. If you choose to generate a certificate, this is the domain used for the certificate request.

.spec.components.knativeServing.ingressGateway.certificate

Description

Certificate specifies configuration of the TLS certificate securing communication for the gateway.

Type

`object`

Property	Type	Description
`secretName`	`string`	SecretName specifies the name of the Kubernetes Secret resource that contains a TLS certificate secure HTTP communications for the KNative network.
`type`	`string`	Type specifies if the TLS certificate should be generated automatically, or if the certificate is provided by the user. Allowed values are: • SelfSigned: A certificate is going to be generated using an own private key. • Provided: Pre-existence of the TLS Secret (see SecretName) with a valid certificate is assumed. • ACPDefaultIngress: Default ingress certificate configured for ACP (ALB)

.spec.components.knativeServing.values

Type

`object`

.spec.components.kserve

Description

Kserve component configuration. Requires KServeless operator to be installed.

Type

`object`

Property	Type	Description
`managementState`	`string`	
`namespace`	`string`	
`values`	`object`	

.spec.components.kserve.values

Type

`object`

.spec.values

Type

`object`

Property	Type	Description
`amlApiServer`	`object`	
`amlServer`	`object`	
`amlService`	`object`	
`buildkitd`	`object`	
`experimentalFeatures`	`object`	
`global`	`object`	
`mysql`	`object`	

.spec.values.amlApiServer

Type

`object`

Property	Type	Description
`tolerations`	`array`	

.spec.values.amlApiServer.tolerations

Type

`array`

.spec.values.amlApiServer.tolerations[]

Description

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

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.

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

.spec.values.amlServer

Type

`object`

Property	Type	Description
`nodeSelector`	`array`	
`tolerations`	`array`	

.spec.values.amlServer.nodeSelector

Type

`array`

.spec.values.amlServer.nodeSelector[]

Type

`string`

.spec.values.amlServer.tolerations

Type

`array`

.spec.values.amlServer.tolerations[]

Description

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

Type

`object`

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

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

.spec.values.amlService

Type

`object`

Property	Type	Description
`appsBaseImage`	`string`	应用镜像构建的基础镜像
`appsGradioVersions`	`string`	应用镜像构建可选的 Gradio 的版本列表
`appsPipMirror`	`string`	应用镜像构建使用的 pip 镜像源
`appsStreamlitVersions`	`string`	应用镜像构建可选的 Streamlit 的版本列表
`gitDatasetUpdateCallbackUrl`	`string`	gitlab 回调 amlservice 的访问地址，用于在数据集更新之后，触发自动更新 preview 数据。
`imageBuildPvcWorkspaceSize`	`string`	镜像构建任务用于存放工作区文件的空间大小，比如存放模型文件
`imageBuildUsedPvcName`	`string`	镜像构建服务使用的pvc名称（非NFS），需提前创建好，建议申请大一点的空间用于缓存构建镜像，比如：150G
`nodeSelector`	`array`	
`notebookStorageClass`	`string`	自动创建的 notebook 使用的存储类
`tolerations`	`array`	

Property	Type	Description
`trainingPvcShareInNamespace`	`string`	训练使用的 PVC 是否在命名空间内共享，即一个命名空间仅使用一个通用的 PVC 缓存训练时的模型、数据集
`trainingPvcSize`	`string`	训练 PVC 大小。如果是命名空间共享的缓存PVC，建议使用较大的空间，足够命名空间所有训练任务使用
`trainingPvcStorageClass`	`string`	训练 PVC 使用的存储类

.spec.values.amlService.nodeSelector

Type

`array`

.spec.values.amlService.nodeSelector[]

Type

`string`

.spec.values.amlService.tolerations

Type

`array`

.spec.values.amlService.tolerations[]

Description

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

Type

`object`

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

Property	Type	Description
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.values.buildkitd

Type

`object`

Property	Type	Description
`storage`	`object`	
`tolerations`	`array`	

.spec.values.buildkitd.storage

Type

`object`

Property	Type	Description
`emptyDir`	`object`	当 type 为 emptyDir 时，需要提供以下配置
`pvc`	`object`	当 type 为 pvc 时，需要提供以下配置
`type`	`string`	存储类型，支持：emptyDir,pvc

.spec.values.buildkitd.storage.emptyDir

Description

当 type 为 emptyDir 时，需要提供以下配置

Type

`object`

Property	Type	Description
`sizeLimit`	`integer`	

.spec.values.buildkitd.storage.pvc

Description

当 type 为 pvc 时，需要提供以下配置

Type

`object`

Property	Type	Description
`accessMode`	`string`	
`storageClassName`	`string`	
`storageSize`	`string`	

.spec.values.buildkitd.tolerations

Type

`array`

.spec.values.buildkitd.tolerations[]

Description

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

Type

`object`

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

Property	Type	Description
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.values.experimentalFeatures

Type

`object`

Property	Type	Description
`agents`	`boolean`	
`datasets`	`boolean`	
`imageBuilder`	`boolean`	
`tuneModels`	`boolean`	

.spec.values.global

Type

`object`

Required

`gitlabAdminTokenSecretRef` `gitlabBaseUrl`

Property	Type	Description
`defaultNamespace`	`string`	默认纳管的命名空间

Property	Type	Description
`defaultNodeSelector`	`array`	
`defaultTolerations`	`array`	
`deployFlavor`	`string`	DeployFlavor for single node or HA-cluster
`gitUserEmail`	`string`	后台 git push 使用的 email，仅在 git log 存储
`gitlabAdminTokenSecretRef`	`object`	Gitlab Admin 密钥地址
`gitlabBaseUrl`	`string`	Gitlab 访问地址
`imageRegistryPrefix`	`string`	harbor 项目名字
`images`	`object`	
`labelBaseDomain`	`string`	
`mysql`	`object`	MySQL configuration.
`registry`	`object`	

.spec.values.global.defaultNodeSelector

Type

`array`

.spec.values.global.defaultNodeSelector[]

Type

`string`

.spec.values.global.defaultTolerations

Type

`array`

.spec.values.global.defaultTolerations[]

Description

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

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.

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

.spec.values.global.gitlabAdminTokenSecretRef

Description

Gitlab Admin 密钥地址

Type

`object`

Required

`name` `namespace`

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

.spec.values.global.images

Type

`object`

.spec.values.global.mysql

Description

MySQL configuration.

Type

`object`

Required

`database` `host` `user`

Property	Type	Description
`database`	`string`	Database to use.
`host`	`string`	Host where the database server is located.

Property	Type	Description
`passwordSecretRef`	`object`	Password reference to a secret containing the password.
`port`	`integer`	MySQL port to use, default is usually OK. (default: 3306)
`user`	`string`	User to log in as.

.spec.values.global.mysql.passwordSecretRef

Description

Password reference to a secret containing the password.

Type

`object`

Required

`name` `namespace`

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

.spec.values.global.registry

Type

`object`

Property	Type	Description
`address`	`string`	

.spec.values.mysql

Type

`object`

Property	Type	Description
`pvc`	`object`	
`tolerations`	`array`	

.spec.values.mysql.pvc

Type

`object`

Property	Type	Description
`storageClassName`	`string`	
`storageSize`	`integer`	

.spec.values.mysql.tolerations

Type

`array`

.spec.values.mysql.tolerations[]

Description

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

Type

`object`

Property	Type	Description
`effect`	`string`	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
`key`	`string`	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
`operator`	`string`	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
`tolerationSeconds`	`integer`	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint

Property	Type	Description
		forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
`value`	`string`	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.status

Type

`object`

Property	Type	Description
`components`	`object`	Expose component's specific status
`conditions`	`array`	
`lastHandledReconcileAt`	`string`	LastHandledReconcileAt holds the value of the most recent reconcile request value, so a change of the annotation value can be detected.
`observedGeneration`	`integer`	The observed generation of the object. When this matches the object's metadata.generation, it indicates the status is up-to-date.

.status.components

Description

Expose component's specific status

Type

`object`

Property	Type	Description
`aml`	`object`	
`kServe`	`object`	
`knativeServing`	`object`	

.status.components.aml

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions is a set of Condition instances.
`deployedRelease`	`object`	
`managementState`	`string`	

.status.components.aml.conditions

Description

Conditions is a set of Condition instances.

Type

`array`

.status.components.aml.conditions[]

Description

Condition represents an observation of an object's state. Conditions are an extension mechanism intended to be used when the details of an observation are not a priori known or would not apply to all instances of a given Kind. Conditions should be added to explicitly convey properties that users and components care about rather than requiring those properties to be inferred from other observations. Once defined, the meaning of a Condition can not be changed arbitrarily - it becomes part of the API, and has the same backwards- and forwards-compatibility concerns of any other part of the API.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	ConditionReason is intended to be a one-word, CamelCase representation of the category of cause of the current status. It is intended to be used in concise output, such as one-line kubectl get output, and in summarizing occurrences of causes.
`status`	`string`	
`type`	`string`	ConditionType is the type of the condition and is typically a CamelCased word or short phrase.

Property	Type	Description
		Condition types should indicate state in the "abnormal-true" polarity. For example, if the condition indicates when a policy is invalid, the "is valid" case is probably the norm, so the condition should be called "Invalid".

.status.components.aml.deployedRelease

Type

`object`

Property	Type	Description
`name`	`string`	

.status.components.kServe

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions is a set of Condition instances.
`deployedRelease`	`object`	
`managementState`	`string`	

.status.components.kServe.conditions

Description

Conditions is a set of Condition instances.

Type

`array`

.status.components.kServe.conditions[]

Description

Condition represents an observation of an object's state. Conditions are an extension mechanism intended to be used when the details of an observation are not a priori known or would not apply to all instances of a given Kind. Conditions should be added to explicitly convey properties that users and components care about rather than requiring those properties to be inferred from other observations. Once defined, the meaning of a Condition can not be changed arbitrarily - it becomes part of the API, and has the same backwards- and forwards-compatibility concerns of any other part of the API.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	ConditionReason is intended to be a one-word, CamelCase representation of the category of cause of the current status. It is intended to be used in concise output, such as one-line kubectl get output, and in summarizing occurrences of causes.
`status`	`string`	

Property	Type	Description
`type`	`string`	ConditionType is the type of the condition and is typically a CamelCased word or short phrase. Condition types should indicate state in the "abnormal-true" polarity. For example, if the condition indicates when a policy is invalid, the "is valid" case is probably the norm, so the condition should be called "Invalid".

.status.components.kServe.deployedRelease

Type

`object`

Property	Type	Description
`name`	`string`	

.status.components.knativeServing

Type

`object`

Property	Type	Description
`conditions`	`array`	Conditions is a set of Condition instances.
`deployedRelease`	`object`	
`managementState`	`string`	

.status.components.knativeServing.conditions

Description

Conditions is a set of Condition instances.

Type

`array`

.status.components.knativeServing.conditions[]

Description

Condition represents an observation of an object's state. Conditions are an extension mechanism intended to be used when the details of an observation are not a priori known or would not apply to all instances of a given Kind. Conditions should be added to explicitly convey properties that users and components care about rather than requiring those properties to be inferred from other observations. Once defined, the meaning of a Condition can not be changed arbitrarily - it becomes part of the API, and has the same backwards- and forwards-compatibility concerns of any other part of the API.

Type

`object`

Required

`status` `type`

Property	Type	Description
`lastTransitionTime`	`string`	
`message`	`string`	
`reason`	`string`	ConditionReason is intended to be a one-word, CamelCase representation of the category of cause of the current status. It is intended to be used in concise output, such as one-line kubectl get output, and in summarizing occurrences of causes.

Property	Type	Description
`status`	`string`	
`type`	`string`	ConditionType is the type of the condition and is typically a CamelCased word or short phrase. Condition types should indicate state in the "abnormal-true" polarity. For example, if the condition indicates when a policy is invalid, the "is valid" case is probably the norm, so the condition should be called "Invalid".

.status.components.knativeServing.deployedRelease

Type

`object`

Property	Type	Description
`name`	`string`	

.status.conditions

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

Property	Type	Description
`reason`	`string`	reason contains a programmatic identifier indicating the re for the condition's last transition. Producers of specific cor types may define expected values and meanings for this f and whether the values are considered a guaranteed API value should be a CamelCase string. This field may not b empty.
`status`	`string`	status of the condition, one of True, False, Unknown.
`type`	`string`	# type of condition in CamelCase or in foo.example.com/CamelCa Many .condition.type values are consistent across resour like Available, but because arbitrary conditions can be use (see .node.status.conditions), the ability to deconflict is important. The regex it matches is (dns1123SubdomainFr (qualifiedNameFmt)

API Endpoints

The following API endpoints are available:

- `/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters`

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

- `/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters/{name}`
  - `DELETE`: delete the specified AmlCluster
  - `GET`: read the specified AmlCluster
  - `PATCH`: partially update the specified AmlCluster
  - `PUT`: replace the specified AmlCluster
- `/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters/{name}/status`
  - `GET`: read status of the specified AmlCluster
  - `PATCH`: partially update status of the specified AmlCluster
  - `PUT`: replace status of the specified AmlCluster

/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters

HTTP method

`DELETE`

Description

delete collection of AmlCluster

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind AmlCluster

HTTP responses

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

HTTP method

`POST`

Description

create a new AmlCluster

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

Parameter	Type	Description
		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`	`AmlCluster` schema	`application/json` formatted

HTTP responses

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

/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters/{name}

HTTP method

`DELETE`

Description

delete the specified AmlCluster

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 AmlCluster

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified AmlCluster

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	`AmlCluster` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified AmlCluster

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`	`AmlCluster` schema	`application/json` formatted

HTTP responses

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

/apis/amlclusters.aml.dev/v1alpha1/namespaces/{namespace}/amlclusters/{name}/status

HTTP method

`GET`

Description

read status of the specified AmlCluster

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified AmlCluster

Query parameters

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace status of the specified AmlCluster

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

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

HTTP responses

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