RBAC APIs

ClusterRole [rbac.authorization ClusterRoleBinding [rbac.authc Role [rbac.a

RoleBinding [rbac.authorizatio RoleTemplate [auth.alauda.io/v1beta1]

ClusterRole [rbac.authorization.k8s.io/v1]

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/clusterroles

Common Parameters

- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

list or watch objects of kind ClusterRole

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.
- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query

result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.

- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `resourceVersion` (*in query*): `string`

resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.

Defaults to unset

- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.
    - `resourceVersionMatch` set to any other value or unset Invalid error is returned.

  Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.
- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`

Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` ClusterRoleList: OK
- `401` : Unauthorized

`post`

create a ClusterRole

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

ClusterRole required

Response

- `200` ClusterRole: OK
- `201` ClusterRole: Created
- `202` ClusterRole: Accepted
- `401` : Unauthorized

`delete`

delete collection of ClusterRole

Parameters

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `dryRun` (*in query*): `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
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.

- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.
- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If

`resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.

- `resourceVersionMatch` set to any other value or unset Invalid error is returned.

Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.

- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `401` : Unauthorized

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/clusterroles/{name}

Common Parameters

- `name` (*in path*): `string` required
  name of the ClusterRole
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

read the specified ClusterRole

Response

- `200` ClusterRole: OK
- `401` : Unauthorized

`put`

replace the specified ClusterRole

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

ClusterRole required

Response

- `200` ClusterRole: OK
- `201` ClusterRole: Created
- `401` : Unauthorized

`delete`

delete a ClusterRole

Parameters

- `dryRun` (*in query*): `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
- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `orphanDependents` (*in query*): `boolean`
  Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added

to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `202` Status: Accepted
- `401` : Unauthorized

`patch`

partially update the specified ClusterRole

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗. This field is required for apply requests (application/apply-patch) but optional for non-apply patch types (JsonPatch, MergePatch, StrategicMergePatch).

- `fieldValidation` (*in query*): `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.
- `force` (*in query*): `boolean`
  Force is going to "force" Apply requests. It means user will re-acquire conflicting fields owned by other people. Force flag must be unset for non-apply patch requests.

Request Body

Patch required

Response

- `200` ClusterRole: OK
- `201` ClusterRole: Created
- `401` : Unauthorized

ClusterRoleList

ClusterRoleList is a collection of ClusterRoles

- `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 ↗

- `items` : `[]ClusterRole`
  Items is a list of ClusterRoles
- `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 ↗
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

ClusterRole

ClusterRole is a cluster level, logical grouping of PolicyRules that can be referenced as a unit by a RoleBinding or ClusterRoleBinding.

- `aggregationRule` : AggregationRule
  AggregationRule describes how to locate ClusterRoles to aggregate into the ClusterRole
- `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 ↗
- `metadata` : ObjectMeta
  ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
- `rules` : `[]PolicyRule`
  Rules holds all the PolicyRules for this ClusterRole

AggregationRule

AggregationRule describes how to locate ClusterRoles to aggregate into the ClusterRole

- `clusterRoleSelectors` : `[]LabelSelector`
  ClusterRoleSelectors holds a list of selectors which will be used to find ClusterRoles and create the rules. If any of the selectors match, then the ClusterRole's permissions will be added

LabelSelector

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

- `matchExpressions` : `[]LabelSelectorRequirement`
  matchExpressions is a list of label selector requirements. The requirements are ANDed.
- `matchLabels` : `map[string]string`
  matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

LabelSelectorRequirement

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

- `key` : `string`
  key is the label key that the selector applies to.
- `operator` : `string`
  operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
- `values` : `[]string`

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

PolicyRule

PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

- `apiGroups`: `[]string`
  APIGroups is the name of the APIGroup that contains the resources. If multiple API groups are specified, any action requested against one of the enumerated resources in any API group will be allowed. "" represents the core API group and "*" represents all API groups.
- `nonResourceURLs`: `[]string`
  NonResourceURLs is a set of partial urls that a user should have access to. *s are allowed, but only as the full, final step in the path Since non-resource URLs are not namespaced, this field is only applicable for ClusterRoles referenced from a ClusterRoleBinding. Rules can either apply to API resources (such as "pods" or "secrets") or non-resource URL paths (such as "/api"), but not both.
- `resourceNames`: `[]string`
  ResourceNames is an optional white list of names that the rule applies to. An empty set means that everything is allowed.
- `resources`: `[]string`
  Resources is a list of resources this rule applies to. '*' represents all resources.
- `verbs`: `[]string`
  Verbs is a list of Verbs that apply to ALL the ResourceKinds contained in this rule. '*' represents all verbs.

ListMeta

ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

- `continue`: `string`

continue may be set if the user set a limit on the number of items returned, and indicates that the server has more data available. The value is opaque and may be used to issue another request to the endpoint that served this list to retrieve the next set of available objects. Continuing a consistent list may not be possible if the server configuration has changed or more than a few minutes have passed. The resourceVersion field returned when using this continue value will be identical to the value in the first response, unless you have received this token from an error message.

- `remainingItemCount` : `integer`
  remainingItemCount is the number of subsequent items in the list which are not included in this list response. If the list request contained label or field selectors, then the number of remaining items is unknown and the field will be left unset and omitted during serialization. If the list is complete (either because it is not chunking or because this is the last chunk), then there are no more remaining items and this field will be left unset and omitted during serialization. Servers older than v1.15 do not set this field. The intended use of the remainingItemCount is *estimating* the size of a collection. Clients should not rely on the remainingItemCount to be set or to be exact.
- `resourceVersion` : `string`
  String that identifies the server's internal version of this object that can be used by clients to determine when objects have changed. Value must be treated as opaque by clients and passed unmodified back to the server. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
- `selfLink` : `string`
  Deprecated: selfLink is a legacy read-only field that is no longer populated by the system.

Status

Status is a return value for calls that don't return other objects.

- `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 ↗
- `code` : `integer`

Suggested HTTP return code for this status, 0 if not set.

- `details` : StatusDetails
  StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.
- `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 ↗
- `message` : `string`
  A human-readable description of the status of this operation.
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.
- `reason` : `string`
  A machine-readable description of why this operation is in the "Failure" status. If this value is empty there is no information available. A Reason clarifies an HTTP status code but does not override it.
- `status` : `string`
  Status of the operation. One of: "Success" or "Failure". More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status ↗

StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `causes` : `[]StatusCause`

The Causes array includes more details associated with the StatusReason failure. Not all StatusReasons may provide detailed causes.

- `group` : `string`
  The group attribute of the resource associated with the status StatusReason.
- `kind` : `string`
  The kind attribute of the resource associated with the status StatusReason. On some operations may differ from the requested resource Kind. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
- `name` : `string`
  The name attribute of the resource associated with the status StatusReason (when there is a single name which can be described).
- `retryAfterSeconds` : `integer`
  If specified, the time in seconds before the operation should be retried. Some errors may indicate the client must take an alternate action - for those errors this field may indicate how long to wait before taking the alternate action.
- `uid` : `string`
  UID of the resource. (when there is a single resource which can be described). More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#uids ↗

StatusCause

StatusCause provides more information about an api.Status failure, including cases when multiple errors are encountered.

- `field` : `string`
  The field of the resource that has caused this error, as named by its JSON serialization. May include dot and postfix notation for nested attributes. Arrays are zero-indexed. Fields may appear more than once in an array of causes due to fields having multiple errors. Optional.
  Examples: "name" - the field "name" on the current resource "items[0].name" - the field "name" on the first array entry in "items"
- `message` : `string`

A human-readable description of the cause of the error. This field may be presented as-is to a reader.

- `reason` : `string`
  A machine-readable description of the cause of the error. If this value is empty there is no information available.

Patch

Patch is provided to give a concrete name and type to the Kubernetes PATCH request body.

ClusterRoleBinding [rbac.authorization.k8s.io/v1]

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/clusterrolebindings

Common Parameters

- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

list or watch objects of kind ClusterRoleBinding

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.
- `continue` (*in query*): `string`

The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.

- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.

- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.
    - `resourceVersionMatch` set to any other value or unset Invalid error is returned.

  Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.
- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`

Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` ClusterRoleBindingList: OK
- `401` : Unauthorized

`post`

create a ClusterRoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

ClusterRoleBinding required

Response

- `200` ClusterRoleBinding: OK
- `201` ClusterRoleBinding: Created
- `202` ClusterRoleBinding: Accepted
- `401` : Unauthorized

`delete`

delete collection of ClusterRoleBinding

Parameters

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `dryRun` (*in query*): `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
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.

- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.
- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If

`resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.

- `resourceVersionMatch` set to any other value or unset Invalid error is returned.

Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.

- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `401` : Unauthorized

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/clusterrolebindings/{name}

Common Parameters

- `name` (*in path*): `string` required
  name of the ClusterRoleBinding
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

read the specified ClusterRoleBinding

Response

- `200` ClusterRoleBinding: OK
- `401` : Unauthorized

`put`

replace the specified ClusterRoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

ClusterRoleBinding required

Response

- `200` ClusterRoleBinding: OK
- `201` ClusterRoleBinding: Created
- `401` : Unauthorized

`delete`

delete a ClusterRoleBinding

Parameters

- `dryRun` (*in query*): `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
- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `orphanDependents` (*in query*): `boolean`
  Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added

to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `202` Status: Accepted
- `401` : Unauthorized

`patch`

partially update the specified ClusterRoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗. This field is required for apply requests (application/apply-patch) but optional for non-apply patch types (JsonPatch, MergePatch, StrategicMergePatch).

- `fieldValidation` (*in query*): `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.
- `force` (*in query*): `boolean`
  Force is going to "force" Apply requests. It means user will re-acquire conflicting fields owned by other people. Force flag must be unset for non-apply patch requests.

Request Body

Patch required

Response

- `200` ClusterRoleBinding: OK
- `201` ClusterRoleBinding: Created
- `401` : Unauthorized

ClusterRoleBindingList

ClusterRoleBindingList is a collection of ClusterRoleBindings

- `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 ↗

- `items` : `[]ClusterRoleBinding`
  Items is a list of ClusterRoleBindings
- `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 ↗
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

ClusterRoleBinding

ClusterRoleBinding references a ClusterRole, but not contain it. It can reference a ClusterRole in the global namespace, and adds who information via Subject.

- `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 ↗
- `metadata` : ObjectMeta
  ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
- `roleRef` : RoleRef
  RoleRef contains information that points to the role being used
- `subjects` : `[]Subject`
  Subjects holds references to the objects the role applies to.

RoleRef

RoleRef contains information that points to the role being used

- `apiGroup` : `string`
  APIGroup is the group for the resource being referenced
- `kind` : `string`
  Kind is the type of resource being referenced
- `name` : `string`
  Name is the name of resource being referenced

Subject

Subject contains a reference to the object or user identities a role binding applies to. This can either hold a direct API object reference, or a value for non-objects such as user and group names.

- `apiGroup` : `string`
  APIGroup holds the API group of the referenced subject. Defaults to "" for ServiceAccount subjects. Defaults to "rbac.authorization.k8s.io" for User and Group subjects.
- `kind` : `string`
  Kind of object being referenced. Values defined by this API group are "User", "Group", and "ServiceAccount". If the Authorizer does not recognized the kind value, the Authorizer should report an error.
- `name` : `string`
  Name of the object being referenced.
- `namespace` : `string`
  Namespace of the referenced object. If the object kind is non-namespace, such as "User" or "Group", and this value is not empty the Authorizer should report an error.

ListMeta

ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

- `continue` : `string`
  continue may be set if the user set a limit on the number of items returned, and indicates that the server has more data available. The value is opaque and may be used to issue another request to the endpoint that served this list to retrieve the next set of available objects. Continuing a consistent list may not be possible if the server configuration has changed or more than a few minutes have passed. The resourceVersion field returned when using this continue value will be identical to the value in the first response, unless you have received this token from an error message.
- `remainingItemCount` : `integer`
  remainingItemCount is the number of subsequent items in the list which are not included in this list response. If the list request contained label or field selectors, then the number of remaining items is unknown and the field will be left unset and omitted during serialization. If the list is complete (either because it is not chunking or because this is the last chunk), then there are no more remaining items and this field will be left unset and omitted during serialization. Servers older than v1.15 do not set this field. The intended use of the remainingItemCount is *estimating* the size of a collection. Clients should not rely on the remainingItemCount to be set or to be exact.
- `resourceVersion` : `string`
  String that identifies the server's internal version of this object that can be used by clients to determine when objects have changed. Value must be treated as opaque by clients and passed unmodified back to the server. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
- `selfLink` : `string`
  Deprecated: selfLink is a legacy read-only field that is no longer populated by the system.

Status

Status is a return value for calls that don't return other objects.

- `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 ↗

- `code` : `integer`
  Suggested HTTP return code for this status, 0 if not set.
- `details` : StatusDetails
  StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.
- `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 ↗
- `message` : `string`
  A human-readable description of the status of this operation.
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.
- `reason` : `string`
  A machine-readable description of why this operation is in the "Failure" status. If this value is empty there is no information available. A Reason clarifies an HTTP status code but does not override it.
- `status` : `string`
  Status of the operation. One of: "Success" or "Failure". More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status ↗

StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `causes` : `[]StatusCause`
  The Causes array includes more details associated with the StatusReason failure. Not all StatusReasons may provide detailed causes.
- `group` : `string`
  The group attribute of the resource associated with the status StatusReason.
- `kind` : `string`
  The kind attribute of the resource associated with the status StatusReason. On some operations may differ from the requested resource Kind. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
- `name` : `string`
  The name attribute of the resource associated with the status StatusReason (when there is a single name which can be described).
- `retryAfterSeconds` : `integer`
  If specified, the time in seconds before the operation should be retried. Some errors may indicate the client must take an alternate action - for those errors this field may indicate how long to wait before taking the alternate action.
- `uid` : `string`
  UID of the resource. (when there is a single resource which can be described). More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#uids ↗

StatusCause

StatusCause provides more information about an api.Status failure, including cases when multiple errors are encountered.

- `field` : `string`
  The field of the resource that has caused this error, as named by its JSON serialization. May include dot and postfix notation for nested attributes. Arrays are zero-indexed. Fields

may appear more than once in an array of causes due to fields having multiple errors. Optional.
Examples: "name" - the field "name" on the current resource "items[0].name" - the field "name" on the first array entry in "items"

- `message` : `string`
  A human-readable description of the cause of the error. This field may be presented as-is to a reader.
- `reason` : `string`
  A machine-readable description of the cause of the error. If this value is empty there is no information available.

Patch

Patch is provided to give a concrete name and type to the Kubernetes PATCH request body.

Role [rbac.authorization.k8s.io/v1]

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles

Common Parameters

- `namespace` (*in path*): `string` required
  object name and auth scope, such as for teams and projects
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

list or watch objects of kind Role

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.

- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.
    - `resourceVersionMatch` set to any other value or unset Invalid error is returned.

  Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.
- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`

Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` RoleList: OK
- `401` : Unauthorized

`post`

create a Role

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

Role required

Response

- `200` Role: OK
- `201` Role: Created
- `202` Role: Accepted
- `401` : Unauthorized

`delete`

delete collection of Role

Parameters

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `dryRun` (*in query*): `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
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.

- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.
- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If

`resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.

- `resourceVersionMatch` set to any other value or unset Invalid error is returned.

Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.

- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `401` : Unauthorized

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles/{name}

Common Parameters

- `name` (*in path*): `string` required
  name of the Role
- `namespace` (*in path*): `string` required
  object name and auth scope, such as for teams and projects
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

read the specified Role

Response

- `200` Role: OK
- `401` : Unauthorized

`put`

replace the specified Role

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

Role required

Response

- `200` Role: OK
- `201` Role: Created
- `401` : Unauthorized

`delete`

delete a Role

Parameters

- `dryRun` (*in query*): `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
- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `202` Status: Accepted
- `401` : Unauthorized

`patch`

partially update the specified Role

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗. This field is required for apply

requests (application/apply-patch) but optional for non-apply patch types (JsonPatch, MergePatch, StrategicMergePatch).

- `fieldValidation` (*in query*): `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.
- `force` (*in query*): `boolean`
  Force is going to "force" Apply requests. It means user will re-acquire conflicting fields owned by other people. Force flag must be unset for non-apply patch requests.

Request Body

Patch `required`

Response

- `200` Role: OK
- `201` Role: Created
- `401` : Unauthorized

RoleList

RoleList is a collection of Roles

- `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 ↗

- `items`: `[]Role`
  Items is a list of Roles
- `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 ↗
- `metadata`: ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

Role

Role is a namespaced, logical grouping of PolicyRules that can be referenced as a unit by a RoleBinding.

- `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 ↗
- `metadata`: ObjectMeta
  ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
- `rules`: `[]PolicyRule`
  Rules holds all the PolicyRules for this Role

PolicyRule

PolicyRule holds information that describes a policy rule, but does not contain information about who the rule applies to or which namespace the rule applies to.

- `apiGroups` : `[]string`
  APIGroups is the name of the APIGroup that contains the resources. If multiple API groups are specified, any action requested against one of the enumerated resources in any API group will be allowed. "" represents the core API group and "*" represents all API groups.
- `nonResourceURLs` : `[]string`
  NonResourceURLs is a set of partial urls that a user should have access to. *s are allowed, but only as the full, final step in the path Since non-resource URLs are not namespaced, this field is only applicable for ClusterRoles referenced from a ClusterRoleBinding. Rules can either apply to API resources (such as "pods" or "secrets") or non-resource URL paths (such as "/api"), but not both.
- `resourceNames` : `[]string`
  ResourceNames is an optional white list of names that the rule applies to. An empty set means that everything is allowed.
- `resources` : `[]string`
  Resources is a list of resources this rule applies to. '*' represents all resources.
- `verbs` : `[]string`
  Verbs is a list of Verbs that apply to ALL the ResourceKinds contained in this rule. '*' represents all verbs.

ListMeta

ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

- `continue` : `string`
  continue may be set if the user set a limit on the number of items returned, and indicates that the server has more data available. The value is opaque and may be used to issue another request to the endpoint that served this list to retrieve the next set of available objects. Continuing a consistent list may not be possible if the server configuration has

changed or more than a few minutes have passed. The resourceVersion field returned when using this continue value will be identical to the value in the first response, unless you have received this token from an error message.

- `remainingItemCount` : `integer`
  remainingItemCount is the number of subsequent items in the list which are not included in this list response. If the list request contained label or field selectors, then the number of remaining items is unknown and the field will be left unset and omitted during serialization. If the list is complete (either because it is not chunking or because this is the last chunk), then there are no more remaining items and this field will be left unset and omitted during serialization. Servers older than v1.15 do not set this field. The intended use of the remainingItemCount is *estimating* the size of a collection. Clients should not rely on the remainingItemCount to be set or to be exact.
- `resourceVersion` : `string`
  String that identifies the server's internal version of this object that can be used by clients to determine when objects have changed. Value must be treated as opaque by clients and passed unmodified back to the server. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
- `selfLink` : `string`
  Deprecated: selfLink is a legacy read-only field that is no longer populated by the system.

Status

Status is a return value for calls that don't return other objects.

- `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 ↗
- `code` : `integer`
  Suggested HTTP return code for this status, 0 if not set.
- `details` : StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `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 ↗
- `message` : `string`
  A human-readable description of the status of this operation.
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.
- `reason` : `string`
  A machine-readable description of why this operation is in the "Failure" status. If this value is empty there is no information available. A Reason clarifies an HTTP status code but does not override it.
- `status` : `string`
  Status of the operation. One of: "Success" or "Failure". More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status ↗

StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `causes` : `[]StatusCause`
  The Causes array includes more details associated with the StatusReason failure. Not all StatusReasons may provide detailed causes.
- `group` : `string`

The group attribute of the resource associated with the status StatusReason.

- `kind` : `string`
  The kind attribute of the resource associated with the status StatusReason. On some operations may differ from the requested resource Kind. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
- `name` : `string`
  The name attribute of the resource associated with the status StatusReason (when there is a single name which can be described).
- `retryAfterSeconds` : `integer`
  If specified, the time in seconds before the operation should be retried. Some errors may indicate the client must take an alternate action - for those errors this field may indicate how long to wait before taking the alternate action.
- `uid` : `string`
  UID of the resource. (when there is a single resource which can be described). More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#uids ↗

StatusCause

StatusCause provides more information about an api.Status failure, including cases when multiple errors are encountered.

- `field` : `string`
  The field of the resource that has caused this error, as named by its JSON serialization. May include dot and postfix notation for nested attributes. Arrays are zero-indexed. Fields may appear more than once in an array of causes due to fields having multiple errors. Optional.
  Examples: "name" - the field "name" on the current resource "items[0].name" - the field "name" on the first array entry in "items"
- `message` : `string`
  A human-readable description of the cause of the error. This field may be presented as-is to a reader.
- `reason` : `string`

A machine-readable description of the cause of the error. If this value is empty there is no information available.

Patch

Patch is provided to give a concrete name and type to the Kubernetes PATCH request body.

RoleBinding [rbac.authorization.k8s.io/v1]

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings

Common Parameters

- `namespace` (*in path*): `string` required
  object name and auth scope, such as for teams and projects
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

list or watch objects of kind RoleBinding

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.

- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.
    - `resourceVersionMatch` set to any other value or unset Invalid error is returned.

  Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.
- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`

Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` RoleBindingList: OK
- `401` : Unauthorized

`post`

create a RoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

RoleBinding required

Response

- `200` RoleBinding: OK
- `201` RoleBinding: Created
- `202` RoleBinding: Accepted
- `401` : Unauthorized

`delete`

delete collection of RoleBinding

Parameters

- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.
- `dryRun` (*in query*): `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
- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.

- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.
- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If

`resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.

- `resourceVersionMatch` set to any other value or unset Invalid error is returned.

Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.

- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `401` : Unauthorized

/kubernetes/{cluster}/apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings/{name}

Common Parameters

- `name` (*in path*): `string` required
  name of the RoleBinding
- `namespace` (*in path*): `string` required
  object name and auth scope, such as for teams and projects
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

read the specified RoleBinding

Response

- `200` RoleBinding: OK
- `401` : Unauthorized

`put`

replace the specified RoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

RoleBinding required

Response

- `200` RoleBinding: OK
- `201` RoleBinding: Created
- `401` : Unauthorized

`delete`

delete a RoleBinding

Parameters

- `dryRun` (*in query*): `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
- `gracePeriodSeconds` (*in query*): `integer`
  The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.
- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `orphanDependents` (*in query*): `boolean`

Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.

- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `202` Status: Accepted
- `401` : Unauthorized

`patch`

partially update the specified RoleBinding

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗. This field is required for apply

requests (application/apply-patch) but optional for non-apply patch types (JsonPatch, MergePatch, StrategicMergePatch).

- `fieldValidation` (*in query*): `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.
- `force` (*in query*): `boolean`
  Force is going to "force" Apply requests. It means user will re-acquire conflicting fields owned by other people. Force flag must be unset for non-apply patch requests.

Request Body

Patch required

Response

- `200` RoleBinding: OK
- `201` RoleBinding: Created
- `401` : Unauthorized

RoleBindingList

RoleBindingList is a collection of RoleBindings

- `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 ↗

- `items`: `[]RoleBinding`
  Items is a list of RoleBindings
- `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 ↗
- `metadata`: ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

RoleBinding

RoleBinding references a role, but does not contain it. It can reference a Role in the same namespace or a ClusterRole in the global namespace. It adds who information via Subjects and namespace information by which namespace it exists in. RoleBindings in a given namespace only have effect in that namespace.

- `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 ↗
- `metadata`: ObjectMeta
  ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
- `roleRef`: RoleRef

RoleRef contains information that points to the role being used

- `subjects` : `[]Subject`
  Subjects holds references to the objects the role applies to.

RoleRef

RoleRef contains information that points to the role being used

- `apiGroup` : `string`
  APIGroup is the group for the resource being referenced
- `kind` : `string`
  Kind is the type of resource being referenced
- `name` : `string`
  Name is the name of resource being referenced

Subject

Subject contains a reference to the object or user identities a role binding applies to. This can either hold a direct API object reference, or a value for non-objects such as user and group names.

- `apiGroup` : `string`
  APIGroup holds the API group of the referenced subject. Defaults to "" for ServiceAccount subjects. Defaults to "rbac.authorization.k8s.io" for User and Group subjects.
- `kind` : `string`
  Kind of object being referenced. Values defined by this API group are "User", "Group", and "ServiceAccount". If the Authorizer does not recognized the kind value, the Authorizer should report an error.
- `name` : `string`
  Name of the object being referenced.
- `namespace` : `string`
  Namespace of the referenced object. If the object kind is non-namespace, such as "User" or "Group", and this value is not empty the Authorizer should report an error.

ListMeta

ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

- `continue` : `string`
  continue may be set if the user set a limit on the number of items returned, and indicates that the server has more data available. The value is opaque and may be used to issue another request to the endpoint that served this list to retrieve the next set of available objects. Continuing a consistent list may not be possible if the server configuration has changed or more than a few minutes have passed. The resourceVersion field returned when using this continue value will be identical to the value in the first response, unless you have received this token from an error message.
- `remainingItemCount` : `integer`
  remainingItemCount is the number of subsequent items in the list which are not included in this list response. If the list request contained label or field selectors, then the number of remaining items is unknown and the field will be left unset and omitted during serialization. If the list is complete (either because it is not chunking or because this is the last chunk), then there are no more remaining items and this field will be left unset and omitted during serialization. Servers older than v1.15 do not set this field. The intended use of the remainingItemCount is *estimating* the size of a collection. Clients should not rely on the remainingItemCount to be set or to be exact.
- `resourceVersion` : `string`
  String that identifies the server's internal version of this object that can be used by clients to determine when objects have changed. Value must be treated as opaque by clients and passed unmodified back to the server. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
- `selfLink` : `string`
  Deprecated: selfLink is a legacy read-only field that is no longer populated by the system.

Status

Status is a return value for calls that don't return other objects.

- `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 ↗
- `code` : `integer`
  Suggested HTTP return code for this status, 0 if not set.
- `details` : StatusDetails
  StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.
- `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 ↗
- `message` : `string`
  A human-readable description of the status of this operation.
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.
- `reason` : `string`
  A machine-readable description of why this operation is in the "Failure" status. If this value is empty there is no information available. A Reason clarifies an HTTP status code but does not override it.
- `status` : `string`
  Status of the operation. One of: "Success" or "Failure". More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status ↗

StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `causes` : `[]StatusCause`
  The Causes array includes more details associated with the StatusReason failure. Not all StatusReasons may provide detailed causes.
- `group` : `string`
  The group attribute of the resource associated with the status StatusReason.
- `kind` : `string`
  The kind attribute of the resource associated with the status StatusReason. On some operations may differ from the requested resource Kind. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
- `name` : `string`
  The name attribute of the resource associated with the status StatusReason (when there is a single name which can be described).
- `retryAfterSeconds` : `integer`
  If specified, the time in seconds before the operation should be retried. Some errors may indicate the client must take an alternate action - for those errors this field may indicate how long to wait before taking the alternate action.
- `uid` : `string`
  UID of the resource. (when there is a single resource which can be described). More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#uids ↗

StatusCause

StatusCause provides more information about an api.Status failure, including cases when multiple errors are encountered.

- `field` : `string`
  The field of the resource that has caused this error, as named by its JSON serialization. May include dot and postfix notation for nested attributes. Arrays are zero-indexed. Fields

may appear more than once in an array of causes due to fields having multiple errors. Optional.
Examples: "name" - the field "name" on the current resource "items[0].name" - the field "name" on the first array entry in "items"

- `message` : `string`
  A human-readable description of the cause of the error. This field may be presented as-is to a reader.
- `reason` : `string`
  A machine-readable description of the cause of the error. If this value is empty there is no information available.

Patch

Patch is provided to give a concrete name and type to the Kubernetes PATCH request body.

RoleTemplate [auth.alauda.io/v1beta1]

/kubernetes/{cluster}/apis/auth.alauda.io/v1beta1/roletemplates

Common Parameters

- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

list objects of kind RoleTemplate

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.
- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query

result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.

- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `resourceVersion` (*in query*): `string`

resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.

Defaults to unset

- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
  When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.
    - `resourceVersionMatch` set to any other value or unset Invalid error is returned.

  Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.
- `timeoutSeconds` (*in query*): `integer`
  Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`

Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` RoleTemplateList: OK
- `401` : Unauthorized

`post`

create a RoleTemplate

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint ↗.
- `fieldValidation` (*in query*): `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.

Request Body

RoleTemplate required

Response

- `200` RoleTemplate: OK
- `201` RoleTemplate: Created
- `202` RoleTemplate: Accepted
- `401` : Unauthorized

`delete`

delete collection of RoleTemplate

Parameters

- `allowWatchBookmarks` (*in query*): `boolean`
  allowWatchBookmarks requests watch events with type "BOOKMARK". Servers that do not implement bookmarks may ignore this flag and bookmarks are sent at the server's discretion. Clients should not assume bookmarks are returned at any specific interval, nor may they assume the server will send any BOOKMARK event during a session. If this is not a watch, this field is ignored.
- `continue` (*in query*): `string`
  The continue option should be set when retrieving more results from the server. Since this value is server defined, clients may only use the continue value from a previous query result with identical query parameters (except for the value of continue) and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, the server will respond with a 410 ResourceExpired error together with a continue token. If the client needs a consistent list, it must restart their list without the continue field. Otherwise, the client may send another list request with the token received with the 410 error, the server will respond with a list starting from the next key, but from the latest snapshot, which is inconsistent from the previous list results - objects that are created, modified, or deleted after the first list request will be included in the response, as long as their keys are after the "next key".
  This field is not supported when watch is true. Clients may start a watch from the last resourceVersion value returned by the server and not miss any modifications.

- `fieldSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their fields. Defaults to everything.
- `labelSelector` (*in query*): `string`
  A selector to restrict the list of returned objects by their labels. Defaults to everything.
- `limit` (*in query*): `integer`
  limit is a maximum number of responses to return for a list call. If more items exist, the server will set the `continue` field on the list metadata to a value that can be used with the same initial query to retrieve the next set of results. Setting a limit may return fewer than the requested amount of items (up to zero items) in the event all requested objects are filtered out and clients should only use the presence of the continue field to determine whether more results are available. Servers may choose not to support the limit argument and will return all of the available results. If limit is specified and the continue field is empty, clients may assume that no more results are available. This field is not supported if watch is true.
  The server guarantees that the objects returned when using continue will be identical to issuing a single list call without a limit - that is, no objects created, modified, or deleted after the first request is issued will be included in any subsequent continued requests. This is sometimes referred to as a consistent snapshot, and ensures that a client that is using limit to receive smaller chunks of a very large result can ensure they see all possible objects. If objects are updated during a chunked list the version of the object that was present at the time the first list result was calculated is returned.
- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `resourceVersionMatch` (*in query*): `string`
  resourceVersionMatch determines how resourceVersion is applied to list calls. It is highly recommended that resourceVersionMatch be set for list calls where resourceVersion is set See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset
- `sendInitialEvents` (*in query*): `boolean`
  `sendInitialEvents=true` may be set together with `watch=true`. In that case, the watch stream will begin with synthetic events to produce the current state of objects in the

collection. Once all such events have been sent, a synthetic "Bookmark" event will be sent. The bookmark will report the ResourceVersion (RV) corresponding to the set of objects, and be marked with `"k8s.io/initial-events-end": "true"` annotation. Afterwards, the watch stream will proceed as usual, sending watch events corresponding to changes (subsequent to the RV) to objects watched.
When `sendInitialEvents` option is set, we require `resourceVersionMatch` option to also be set. The semantic of the watch request is as following: - `resourceVersionMatch` = NotOlderThan is interpreted as "data at least as new as the provided `resourceVersion`" and the bookmark event is send when the state is synced to a `resourceVersion` at least as fresh as the one provided by the ListOptions. If `resourceVersion` is unset, this is interpreted as "consistent read" and the bookmark event is send when the state is synced at least to the moment when request started being processed.

- `resourceVersionMatch` set to any other value or unset Invalid error is returned.

Defaults to true if `resourceVersion=""` or `resourceVersion="0"` (for backward compatibility reasons) and to false otherwise.

- `timeoutSeconds` (*in query*): `integer`
Timeout for the list/watch call. This limits the duration of the call, regardless of any activity or inactivity.
- `watch` (*in query*): `boolean`
Watch for changes to the described resources and return them as a stream of add, update, and remove notifications. Specify resourceVersion.

Response

- `200` Status: OK
- `401` : Unauthorized

/kubernetes/{cluster}/apis/auth.alauda.io/v1beta1/roletemplates/{name}

Common Parameters

- `name` (*in path*): `string` required
  name of the RoleTemplate
- `pretty` (*in query*): `string`
  If 'true', then the output is pretty printed. Defaults to 'false' unless the user-agent indicates a browser or command-line HTTP tool (curl and wget).

`get`

read the specified RoleTemplate

Parameters

- `resourceVersion` (*in query*): `string`
  resourceVersion sets a constraint on what resource versions a request may be served from. See https://kubernetes.io/docs/reference/using-api/api-concepts/#resource-versions ↗ for details.
  Defaults to unset

Response

- `200` RoleTemplate: OK
- `401` : Unauthorized

`put`

replace the specified RoleTemplate

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters,

as defined by https://golang.org/pkg/unicode/#IsPrint ↗.

- `fieldValidation` (*in query*): `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.

Request Body

RoleTemplate required

Response

- `200` RoleTemplate: OK
- `201` RoleTemplate: Created
- `401` : Unauthorized

`delete`

delete a RoleTemplate

Parameters

- `dryRun` (*in query*): `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
- `gracePeriodSeconds` (*in query*): `integer`

The duration in seconds before the object should be deleted. Value must be non-negative integer. The value zero indicates delete immediately. If this value is nil, the default grace period for the specified type will be used. Defaults to a per object value if not specified. zero means delete immediately.

- `ignoreStoreReadErrorWithClusterBreakingPotential` (*in query*): `boolean`
  if set to true, it will trigger an unsafe deletion of the resource in case the normal deletion flow fails with a corrupt object error. A resource is considered corrupt if it can not be retrieved from the underlying storage successfully because of a) its data can not be transformed e.g. decryption failure, or b) it fails to decode into an object. NOTE: unsafe deletion ignores finalizer constraints, skips precondition checks, and removes the object from the storage. WARNING: This may potentially break the cluster if the workload associated with the resource being unsafe-deleted relies on normal deletion flow. Use only if you REALLY know what you are doing. The default value is false, and the user must opt in to enable it
- `orphanDependents` (*in query*): `boolean`
  Deprecated: please use the PropagationPolicy, this field will be deprecated in 1.7. Should the dependent objects be orphaned. If true/false, the "orphan" finalizer will be added to/removed from the object's finalizers list. Either this field or PropagationPolicy may be set, but not both.
- `propagationPolicy` (*in query*): `string`
  Whether and how garbage collection will be performed. Either this field or OrphanDependents may be set, but not both. The default policy is decided by the existing finalizer set in the metadata.finalizers and the resource-specific default policy. Acceptable values are: 'Orphan' - orphan the dependents; 'Background' - allow the garbage collector to delete the dependents in the background; 'Foreground' - a cascading policy that deletes all dependents in the foreground.

Request Body

DeleteOptions

Response

- `200` Status: OK
- `202` Status: Accepted
- `401` : Unauthorized

`patch`

partially update the specified RoleTemplate

Parameters

- `dryRun` (*in query*): `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
- `fieldManager` (*in query*): `string`
  fieldManager is a name associated with the actor or entity that is making these changes. The value must be less than or 128 characters long, and only contain printable characters, as defined by https://golang.org/pkg/unicode/#IsPrint↗. This field is required for apply requests (application/apply-patch) but optional for non-apply patch types (JsonPatch, MergePatch, StrategicMergePatch).
- `fieldValidation` (*in query*): `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.
- `force` (*in query*): `boolean`
  Force is going to "force" Apply requests. It means user will re-acquire conflicting fields owned by other people. Force flag must be unset for non-apply patch requests.

Request Body

Patch required

Response

- `200` RoleTemplate: OK
- `401` : Unauthorized

RoleTemplateList

RoleTemplateList is a list of RoleTemplate

- `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 ↗
- `items` : `[]RoleTemplate`
  List of roletemplates. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md ↗
- `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 ↗
- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

RoleTemplate

RoleTemplate is the Schema for the roletemplates API

- `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 ↗
- `metadata` : ObjectMeta
  ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
- `spec` : `object`
  RoleTemplateSpec defines the desired state of RoleTemplate
- `status` : `object`
  RoleTemplateStatus defines the observed state of RoleTemplate

ListMeta

ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.

- `continue` : `string`
  continue may be set if the user set a limit on the number of items returned, and indicates that the server has more data available. The value is opaque and may be used to issue another request to the endpoint that served this list to retrieve the next set of available objects. Continuing a consistent list may not be possible if the server configuration has changed or more than a few minutes have passed. The resourceVersion field returned when using this continue value will be identical to the value in the first response, unless you have received this token from an error message.
- `remainingItemCount` : `integer`
  remainingItemCount is the number of subsequent items in the list which are not included in this list response. If the list request contained label or field selectors, then the number of remaining items is unknown and the field will be left unset and omitted during serialization. If the list is complete (either because it is not chunking or because this is the last chunk), then there are no more remaining items and this field will be left unset and omitted during

serialization. Servers older than v1.15 do not set this field. The intended use of the remainingItemCount is *estimating* the size of a collection. Clients should not rely on the remainingItemCount to be set or to be exact.

- `resourceVersion` : `string`
  String that identifies the server's internal version of this object that can be used by clients to determine when objects have changed. Value must be treated as opaque by clients and passed unmodified back to the server. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
- `selfLink` : `string`
  Deprecated: selfLink is a legacy read-only field that is no longer populated by the system.

Status

Status is a return value for calls that don't return other objects.

- `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 ↗
- `code` : `integer`
  Suggested HTTP return code for this status, 0 if not set.
- `details` : StatusDetails
  StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.
- `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 ↗
- `message` : `string`

A human-readable description of the status of this operation.

- `metadata` : ListMeta
  ListMeta describes metadata that synthetic resources must have, including lists and various status objects. A resource may have only one of {ObjectMeta, ListMeta}.
- `reason` : `string`
  A machine-readable description of why this operation is in the "Failure" status. If this value is empty there is no information available. A Reason clarifies an HTTP status code but does not override it.
- `status` : `string`
  Status of the operation. One of: "Success" or "Failure". More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status ↗

StatusDetails

StatusDetails is a set of additional properties that MAY be set by the server to provide additional information about a response. The Reason field of a Status object defines what attributes will be set. Clients must ignore fields that do not match the defined type of each attribute, and should assume that any attribute may be empty, invalid, or under defined.

- `causes` : `[]StatusCause`
  The Causes array includes more details associated with the StatusReason failure. Not all StatusReasons may provide detailed causes.
- `group` : `string`
  The group attribute of the resource associated with the status StatusReason.
- `kind` : `string`
  The kind attribute of the resource associated with the status StatusReason. On some operations may differ from the requested resource Kind. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
- `name` : `string`
  The name attribute of the resource associated with the status StatusReason (when there is a single name which can be described).
- `retryAfterSeconds` : `integer`

If specified, the time in seconds before the operation should be retried. Some errors may indicate the client must take an alternate action - for those errors this field may indicate how long to wait before taking the alternate action.

- `uid` : `string`
  UID of the resource. (when there is a single resource which can be described). More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#uids ↗

StatusCause

StatusCause provides more information about an api.Status failure, including cases when multiple errors are encountered.

- `field` : `string`
  The field of the resource that has caused this error, as named by its JSON serialization. May include dot and postfix notation for nested attributes. Arrays are zero-indexed. Fields may appear more than once in an array of causes due to fields having multiple errors. Optional.
  Examples: "name" - the field "name" on the current resource "items[0].name" - the field "name" on the first array entry in "items"
- `message` : `string`
  A human-readable description of the cause of the error. This field may be presented as-is to a reader.
- `reason` : `string`
  A machine-readable description of the cause of the error. If this value is empty there is no information available.

Patch

Patch is provided to give a concrete name and type to the Kubernetes PATCH request body.