❹ Naming rule for creating the plural form, used for specifying a list of those objects

❺ Scope—whether the resource can be created cluster-wide or is specific to a namespace

❻ Version of the CRD

❼ OpenAPI V3 schema for validation (not shown here)

An OpenAPI V3 schema can also be specified to allow Kubernetes to validate a custom resource. For simple use cases, this schema can be omitted, but for production-grade CRDs, the schema should be provided so that configuration errors can be detected early.

Additionally, Kubernetes allows us to specify two possible subresources for our CRD via the spec field `subresources`:

scale
: With this property, a CRD can specify how it manages its replica count. This field can be used to declare the JSON path, where the number of desired replicas of this custom resource is specified: the path to the property that holds the actual number of running replicas and an optional path to a label selector that can be used to find copies of custom resource instances. This label selector is usually optional, but is required if you want to use this custom resource with the HorizontalPodAutoscaler explained in Chapter 24, *Elastic Scale*.

status
: When we set this property, a new API call is available that only allows you to change the status. This API call can be secured individually and allows for status updates from outside the controller. On the other hand, when we update a custom resource as a whole, the `status` section is ignored as for standard Kubernetes resources.

Example 23-2 shows a potential subresource path as is also used for a regular Pod.

Example 23-2. Subresource definition for a CustomResourceDefinition

```
kind: CustomResourceDefinition
# ...
spec:
  subresources:
    status: {}
    scale:
      specReplicasPath: .spec.replicas ❶
```