CUSTOM RESOURCE DEFINITIONS

Custom Resource Definitions

Custom Resources

We have been working with built-in resources, or API endpoints. The flexibility of Kubernetes allows for the dynamic addition of new resources as well. Once these Custom Resources have been added, the objects can be created and accessed using standard calls and commands, like **kubectl**. The creation of a new object stores new structured data in the etcd database and allows access via kube-apiserver.

To make a new custom resource part of a declarative API, there needs to be a controller to retrieve the structured data continually and act to meet and maintain the declared state. This controller, or operator, is an agent that creates and manages one or more instances of a specific stateful application. We have worked with built-in controllers such as Deployments, DaemonSets and other resources.

The functions encoded into a custom operator should be all the tasks a human would need to perform if deploying the application outside of Kubernetes. The details of building a custom controller are outside the scope of this course, and thus, not included.

There are two ways to add custom resources to your Kubernetes cluster. The easiest way, but less flexible, is by adding a Custom Resource Definition (CRD) to the cluster. The second way, which is more flexible, is the use of Aggregated APIs (AA), which requires a new API server to be written and added to the cluster.

Either way of adding a new object to the cluster, as distinct from a built-in resource, is called a Custom Resource.

If you are using RBAC for authorization, you probably will need to grant access to the new CRD resource and controller. If using an Aggregated API, you can use the same or a different authentication process.