Workload Management

Archived: 2026-04-06 01:56:24 UTC

Kubernetes provides several built-in APIs for declarative management of your workloads and the components of those workloads.

Ultimately, your applications run as containers inside Pods; however, managing individual Pods would be a lot of effort. For example, if a Pod fails, you probably want to run a new Pod to replace it. Kubernetes can do that for you.

You use the Kubernetes API to create a workload object that represents a higher abstraction level than a Pod, and then the Kubernetes control plane automatically manages Pod objects on your behalf, based on the specification for the workload object you defined.

The built-in APIs for managing workloads are:

Deployment (and, indirectly, ReplicaSet), the most common way to run an application on your cluster. Deployment is a good fit for managing a stateless application workload on your cluster, where any Pod in the Deployment is interchangeable and can be replaced if needed. (Deployments are a replacement for the legacy ReplicationController API).

A StatefulSet lets you manage one or more Pods – all running the same application code – where the Pods rely on having a distinct identity. This is different from a Deployment where the Pods are expected to be interchangeable. The most common use for a StatefulSet is to be able to make a link between its Pods and their persistent storage. For example, you can run a StatefulSet that associates each Pod with a PersistentVolume. If one of the Pods in the StatefulSet fails, Kubernetes makes a replacement Pod that is connected to the same PersistentVolume.

A DaemonSet defines Pods that provide facilities that are local to a specific node; for example, a driver that lets containers on that node access a storage system. You use a DaemonSet when the driver, or other node-level service, has to run on the node where it's useful. Each Pod in a DaemonSet performs a role similar to a system daemon on a classic Unix / POSIX server. A DaemonSet might be fundamental to the operation of your cluster, such as a plugin to let that node access cluster networking, it might help you to manage the node, or it could provide less essential facilities that enhance the container platform you are running. You can run DaemonSets (and their pods) across every node in your cluster, or across just a subset (for example, only install the GPU accelerator driver on nodes that have a GPU installed).

You can use a Job and / or a CronJob to define tasks that run to completion and then stop. A Job represents a one-off task, whereas each CronJob repeats according to a schedule.

Other topics in this section:

- Automatic Cleanup for Finished Jobs
- ReplicationController

Last modified January 14, 2024 at 2:20 PM PST: Rename concept section (6160a5e137)
