容器磁盘上的文件的生命周期是短暂的，这就使得在容器中运行重要应用时会出现一些问题。首先，当容器崩溃时，kubelet 会重启它，但是容器中的文件将丢失——容器以干净的状态（镜像最初的状态）重新启动。其次，在 `Pod` 中同时运行多个容器时，这些容器之间通常需要共享文件。Kubernetes 中的 `Volume` 抽象就很好的解决了这些问题

背景

Kubernetes 中的卷有明确的寿命 —— 与封装它的 Pod 相同。所f以，卷的生命比 Pod 中的所有容器都长，当这个容器重启时数据仍然得以保存。当然，当 Pod 不再存在时，卷也将不复存在。也许更重要的是，Kubernetes 支持多种类型的卷，Pod 可以同时使用任意数量的卷

卷的类型

Kubernetes 支持以下类型的卷：

- `awsElasticBlockStore` `azureDisk` `azureFile` `cephfs` `csi` `downwardAPI` `emptyDir`
- `fc` `flocker` `gcePersistentDisk` `gitRepo` `glusterfs` `hostPath` `iscsi` `local` `nfs`
- `persistentVolumeClaim` `projected` `portworxVolume` `quobyte` `rbd` `scaleIO` `secret`
- `storageos` `vsphereVolume`

emptyDir

当 Pod 被分配给节点时，首先创建 `emptyDir` 卷，并且只要该 Pod 在该节点上运行，该卷就会存在。正如卷的名字所述，它最初是空的。Pod 中的容器可以读取和写入 `emptyDir` 卷中的相同文件，尽管该卷可以挂载到每个容器中的相同或不同路径上。当出于任何原因从节点中删除 Pod 时，`emptyDir` 中的数据将被永久删除

`emptyDir` **的用法有：**

- **暂存空间，例如用于基于磁盘的合并排序**
- **用作长时间计算崩溃恢复时的检查点**
- **Web服务器容器提供数据时，保存内容管理器容器提取的文件**

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:

  - image: k8s.gcr.io/test-webserver
```

```
    name: test-container
    volumeMounts:
    - mountPath: /cache
      name: cache-volume
  volumes:
  - name: cache-volume
    emptyDir: {}
```

hostPath

`hostPath` **卷将主机节点的文件系统中的文件或目录挂载到集群中**

`hostPath` **的用途如下：**

- **运行需要访问 Docker 内部的容器；使用** `/var/lib/docker` **的** `hostPath`
- **在容器中运行 cAdvisor；使用** `/dev/cgroups` **的** `hostPath`
- **允许 pod 指定给定的 hostPath 是否应该在 pod 运行之前存在，是否应该创建，以及它应该以什么形式存在**

除了所需的 `path` **属性之外，用户还可以为** `hostPath` **卷指定** `type`

值	行为
	空字符串（默认）用于向后兼容，这意味着在挂载 hostPath 卷之前不会执行任何检查。
`DirectoryOrCreate`	如果在给定的路径上没有任何东西存在，那么将根据需要在那里创建一个空目录，权限设置为 0755，与 Kubelet 具有相同的组和所有权。
`Directory`	给定的路径下必须存在目录
`FileOrCreate`	如果在给定的路径上没有任何东西存在，那么会根据需要创建一个空文件，权限设置为 0644，与 Kubelet 具有相同的组和所有权。
`File`	给定的路径下必须存在文件
`Socket`	给定的路径下必须存在 UNIX 套接字
`CharDevice`	给定的路径下必须存在字符设备
`BlockDevice`	给定的路径下必须存在块设备

使用这种卷类型是请注意，因为：

- **由于每个节点上的文件都不同，具有相同配置（例如从 podTemplate 创建的）的 pod 在不同节点上的行为可能会有所不同**
- **当 Kubernetes 按照计划添加资源感知调度时，将无法考虑** `hostPath` **使用的资源**

- **在底层主机上创建的文件或目录只能由 root 写入。您需要在特权容器中以 root 身份运行进程，或修改主机上的文件权限以便写入 `hostPath` 卷**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      # directory location on host
      path: /data
      # this field is optional
      type: Directory
```