马哥教育

IT人的高薪职业学院

kubernetes

讲师： 张士杰/杰哥(QQ:2973707860)

http://www.magedu.com

目录

01 k8s资源管理核心概念

02 K8s命令使用

03 k8s的牛鼻子—API

01

k8s资源管理的核心概念

k8s的设计理念—分层架构

http://docs.kubernetes.org.cn/251.html

k8s的设计理念—API设计原则

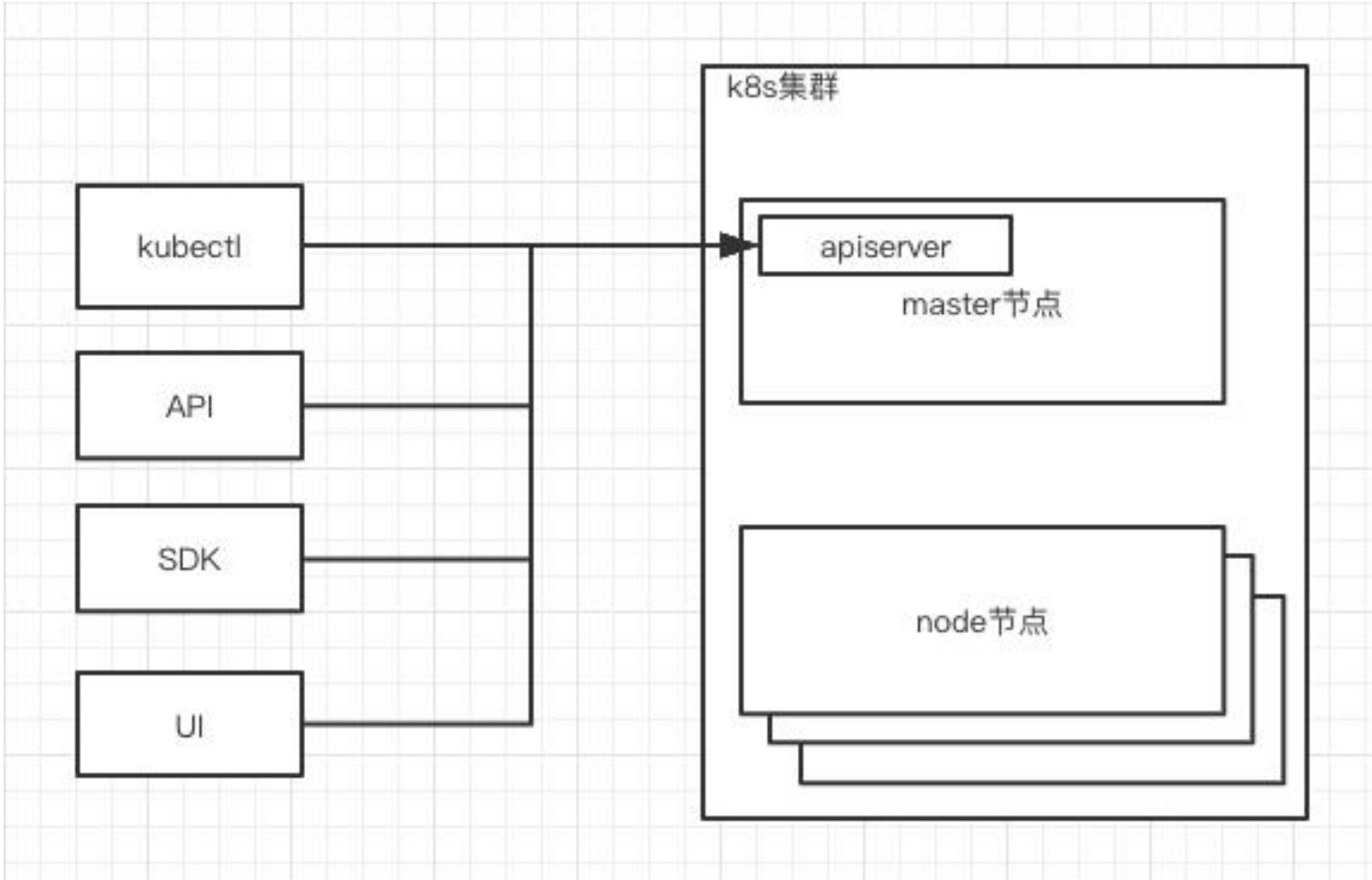
- https://www.kubernetes.org.cn/kubernetes%e8%ae%be%e8%ae%a1%e7%90%86%e5%bf%b5
- 所有API应该是声明式的。
- API对象是彼此互补而且可组合的。
- 高层API以操作意图为基础设计。
- 低层API根据高层API的控制需要设计。
- 尽量避免简单封装，不要有在外部API无法显式知道的内部隐藏的机制。
- API操作复杂度与对象数量成正比。
- API对象状态不能依赖于网络连接状态。
- 尽量避免让操作机制依赖于全局状态，因为在分布式系统中要保证全局状态的同步是非常困难的。

k8s资源管理核心

API：对象

——是K8s集群中的管理操作单元

HOW?



WHAT?

类别	名称
资源对象	Pod、ReplicaSet、ReplicationController、Deployment、StatefulSet、DaemonSet、Job、CronJob、HorizontalPodAutoscaling、Node、Namespace、Service、Ingress、Label、CustomResourceDefinition
存储对象	Volume、PersistentVolume、Secret、ConfigMap
策略对象	SecurityContext、ResourceQuota、LimitRange
身份对象	ServiceAccount、Role、ClusterRole

02

k8s命令使用

https://kubernetes.io/zh/docs/concepts/workloads/controllers/deployment/

命令集	命令	用途
基础命令	create/delete/edit/get/describe/logs/exec/scale	增删改查
	explain	命令说明
配置命令	Label：给node标记label，实现亲pod与node亲和性	标签管理
	apply	动态配置
集群管理命令	cluster-info/top	集群状态
	cordon：警戒线，标记node不被调度 uncordon：取消警戒标记为cordon的node drain：驱逐node上的pod,用于node下线等场景 taint：给node标记污点，实现反亲pod与node反亲和性	node节点管理
	api-resources/api-versions/version	api资源
	config	客户端kube-config配置

03

k8s的牛鼻子—API

k8s的几个重要概念

- 对象　用k8s是和什么打交道？ K8s 声明式API
- yaml文件　怎么打交道？ 调用声明式API
- 必需字段　怎么声明？
    1. apiVersion - 创建该对象所使用的 Kubernetes API 的版本
    2. kind - 想要创建的对象的类型
    3. metadata - 帮助识别对象唯一性的数据，包括一个 name 名称 、可选的 namespace
    4. spec
    5. status（Pod创建完成后k8s自动生成status状态）

yaml文件及必需字段
每个API对象都有3大类属性：元数据metadata、规范spec和状态status。

```yaml
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

spec和status的区别：

spec是期望状态

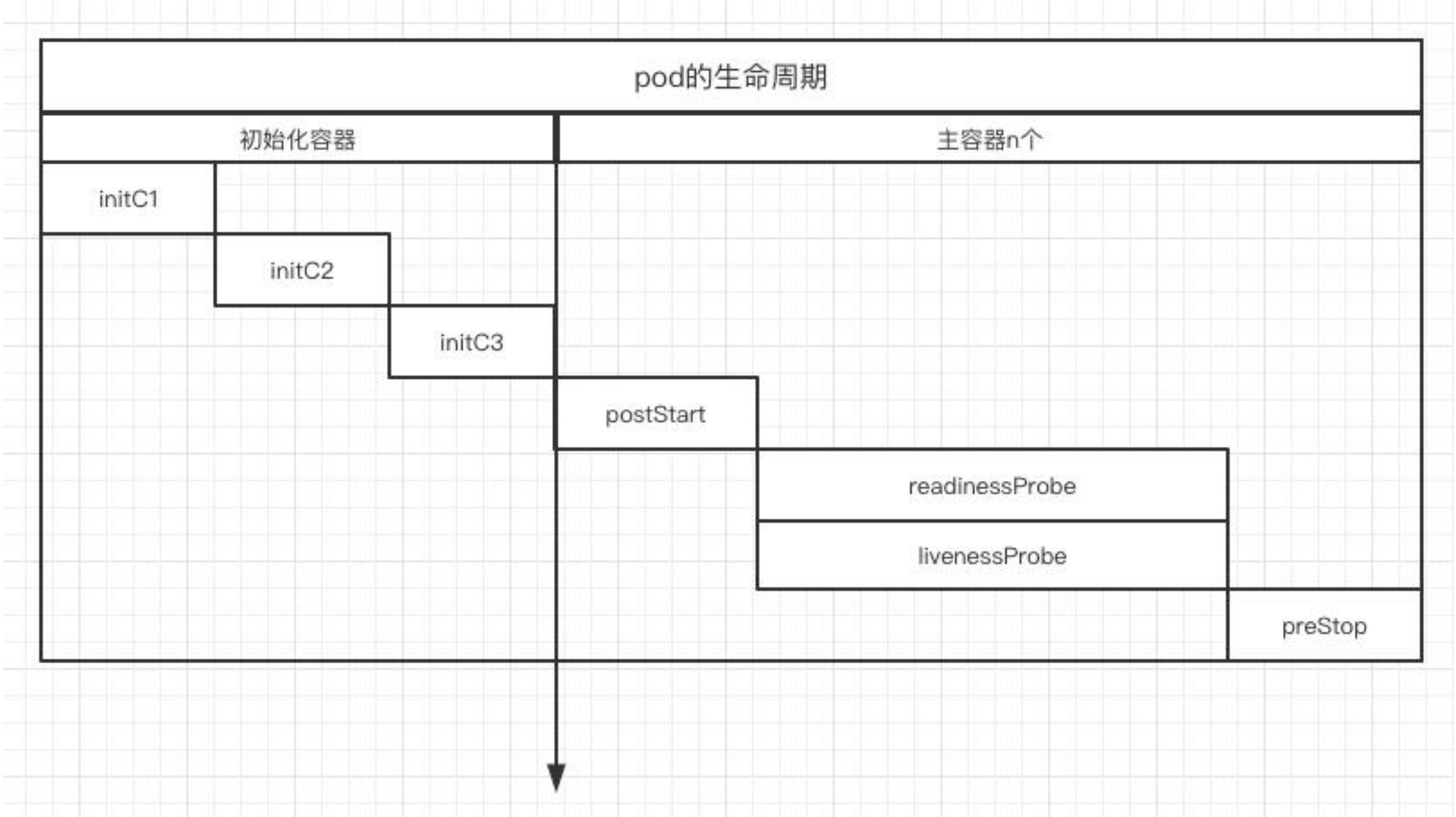
status是实际状态

Pod

概述：

1. pod是k8s中的最小单元
2. 一个pod中可以运行一个容器，也可以运行多个容器
3. 运行多个容器的话，这些容器是一起被调度的
4. Pod的生命周期是短暂的，不会自愈，是用完就销毁的实体
5. 一般我们是通过Controller来创建和管理pod的

Pod生命周期：
初始化容器、启动前操作、就绪探针、存活探针、删除pod操作



livenessProbe和readinessProbe

- livenessProbe：存活探针

  检测应用发生故障时使用，不能提供服务、超时等

  检测失败重启pod
- readinessProbe：就绪探针

  检测pod启动之后应用是否就绪，是否可以提供服务

  检测成功，pod才开始接收流量

Controller：控制器

- Replication Controller #第一代pod副本控制器
- ReplicaSet #第二代pod副本控制器
- Deployment #第三代pod控制器

Rc，Rs和Deployment

- Replication Controller：副本控制器（selector = !=）
  - https://kubernetes.io/zh/docs/concepts/workloads/controllers/replicationcontroller/
    - https://kubernetes.io/zh/docs/concepts/overview/working-with-objects/labels/

- ReplicaSet：副本控制集，和副本控制器的区别是：对选择器的支持（selector 还支持in notin）
  - https://kubernetes.io/zh/docs/concepts/workloads/controllers/replicaset/

- Deployment：比rs更高一级的控制器，除了有rs的功能之外，还有很多高级功能,，比如说最重要的：滚动升级、回滚等
  - https://kubernetes.io/zh/docs/concepts/workloads/controllers/deployment/

Service

- Why：pod重建之后ip就变了，pod之间直接访问会有问题
- What：解耦了服务和应用。
- How：声明一个service对象

一般常用的有两种：

- k8s集群内的service：selector指定pod，自动创建Endpoints
- k8s集群外的service：手动创建Endpoints，指定外部服务的ip，端口和协议

kube-proxy和service的关系：
kube-proxy——————> k8s-apiserver
watch

kube-proxy监听着k8s-apiserver，一旦service资源发生变化（调k8s-api修改service信息），
kube-proxy就会生成对应的负载调度的调整，这样就保证service的最新状态。

kube-proxy有三种调度模型：

- userspace：k8s1.1之前
- iptables：1.2-k8s1.11之前
- ipvs：k8s 1.11之后，如果没有开启ipvs，则自动降级为iptables

Volume

- Why：数据和镜像解耦，以及容器间的数据共享
- What：k8s抽象出的一个对象，用来保存数据，做存储用
- 常用的几种卷：

  emptyDir：**本地临时卷**

  hostPath：**本地卷**

  nfs等：**共享卷**

  configmap: **配置文件**

https://kubernetes.io/zh/docs/concepts/storage/volumes/

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。
- /var/lib/kubelet/pods/$ID/volumes/kubernetes.io~empty-dir/cache-volume/$FILE

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，pod删除的时候，卷不会被删除

nfs等共享存储

- nfs 卷允许将现有的 NFS（网络文件系统）共享挂载到您的容器中。不像 emptyDir，当删除 Pod 时，nfs 卷的内容被保留，卷仅仅是被卸载。这意味着 NFS 卷可以预填充数据，并且可以在 pod 之间“切换”数据。 NFS 可以被多个写入者同时挂载。

  - 创建多个pod测试挂载同一个NFS
  - 创建多个pod测试每个pod挂载多个NFS

Configmap

- Why：配置信息和镜像解耦
- What：将配置信息放到configmap对象中，然后在pod的对象中导入configmap对象，实现导入配置的操作
- How：声明一个ConfigMap的对象，作为Volume挂载到pod中

PV/PVC

- Why：实现pod和storage的解耦，这样我们修改storage的时候不需要修改pod，也可以实现存储和应用权限的隔离
- What：PersistentVolume 和 PersistentVolumeClaim

PV/PVC

- PersistentVolume（PV）是由管理员设置的存储，它是群集的一部分。就像节点是集群中的资源一样，PV 也是集群中的资源。PV 是 Volume 之类的卷插件，但具有独立于使用 PV 的 Pod 的生命周期。此 API 对象包含存储实现的细节，即 NFS、iSCSI 或特定于云供应商的存储系统。
- PersistentVolumeClaim（PVC）是用户存储的请求。它与 Pod 相似。Pod 消耗节点资源，PVC 消耗 PV 资源。Pod 可以请求特定级别的资源（CPU 和内存）。声明可以请求特定的大小和访问模式（例如，可以以读/写一次或 只读多次模式挂载）。

Statefulset

- Why：为了解决有状态服务的问题
- What：它所管理的Pod拥有固定的Pod名称，主机名，启停顺序
- How：

创建一个statefulset类型的pod，并指定serviceName，创建headless类型的svc

https://kubernetes.io/zh/docs/concepts/workloads/controllers/statefulset/ #后期演示

DaemonSet

DaemonSet 在当前集群中每个节点运行同一个pod，当有新的节点加入集群时也会为新的节点配置相同的pod，当节点从集群中移除时其pod也会被kubernetes回收，但是删除DaemonSet 将删除其创建的所有的pod。

- https://kubernetes.io/zh/docs/concepts/workloads/controllers/daemonset/

马哥教育

IT人的高薪职业学院

kubernetes

讲师： 张士杰/杰哥(QQ:2973707860)

http://www.magedu.com