原文地址：https://blog.csdn.net/lxwh0323/category\_11505875.html

**deployment的yaml文件**

apiVersion: apps/v1 # 版本号

kind: Deployment # 类型

metadata: # 元数据

name: # rs名称

namespace: # 所属命名空间

labels: #标签

controller: deploy

spec: # 详情描述

replicas: 3 # 副本数量

revisionHistoryLimit: 3 # 保留历史版本

paused: false # 暂停部署，默认是false

progressDeadlineSeconds: 600 # 部署超时时间（s），默认是600

strategy: # 策略

type: RollingUpdate # 滚动更新策略

rollingUpdate: # 滚动更新

maxSurge: 30% # 最大额外可以存在的副本数，可以为百分比，也可以为整数

maxUnavailable: 30% # 最大不可用状态的 Pod 的最大值，可以为百分比，也可以为整数

selector: # 选择器，通过它指定该控制器管理哪些pod

matchLabels: # Labels匹配规则

app: nginx-pod

matchExpressions: # Expressions匹配规则

- {key: app, operator: In, values: [nginx-pod]}

template: # 模板，当副本数量不足时，会根据下面的模板创建pod副本

metadata:

labels:

app: nginx-pod

spec:

containers:

- name: nginx

image: nginx:1.17.1

ports:

- containerPort: 80

**创建deployment**

我们创建pc-deployment.yaml文件

apiVersion: apps/v1

kind: Deployment

metadata:

name: pc-deployment

namespace: dev

spec:

replicas: 3 # 副本数量

selector:

matchLabels:

app: nginx-pod

template: # 模板，当副本数量不足时，会根据下面的模板创建pod副本

metadata:

labels:

app: nginx-pod

spec:

containers: # 使用的容器镜像

- name: nginx

image: nginx:1.17.1

执行命令

[root@k8s-master01 ~]# kubectl create -f pc-deployment.yaml

deployment.apps/pc-deployment created

查询deployment可以看到我们创建的deployment

root@k8s-master:~# kubectl get deployments -n dev

NAME READY UP-TO-DATE AVAILABLE AGE

pc-deployment 3/3 3 3 66s

**删除deployment**

kubectl delete deployments/pc-deployment -n dev

**编辑deployment yaml**

使用如下命令编辑已存在的deployment

kubectl edit deployments/pc-deployment -n dev

**扩容**

我们使用命令直接扩容

root@k8s-master:~# kubectl scale deployment pc-deployment --replicas=5 -n dev

deployment.apps/pc-deployment scaled

使用如下命令查询扩容情况

root@k8s-master:~# kubectl get pods -l app="nginx-pod" -n dev

NAME READY STATUS RESTARTS AGE

pc-deployment-6f7f65b46d-6mwh5 1/1 Running 0 4m57s

pc-deployment-6f7f65b46d-bblzt 1/1 Running 0 17m

pc-deployment-6f7f65b46d-jmztl 1/1 Running 0 17m

pc-deployment-6f7f65b46d-xgfls 1/1 Running 0 17m

pc-deployment-6f7f65b46d-zflhn 1/1 Running 0 4m57s

**版本更新策略**

Deployment的版本更新策略使用strategy字段控制的

# 指定新的Pod替换旧的Pod的策略，支持两个属性:

strategy:

  # 指定策略类型，支持两种策略，

  # Recreate: 在创建出新的Pod之前会先杀掉所有已存在的Pod，

  # RollingUpdate: 滚动更新，就是杀死一部分，就启动一部分，在更新过程中，存在两个版本Pod

  type: RollingUpdate

  # 当type为RollingUpdate时生效，用于为RollingUpdate设置参数，支持两个属性

  rollingUpdate:

    # 用来指定在升级过程中不可用Pod的最大数量，默认为25%。

    maxUnavailable: 25%

    # 用来指定在升级过程中可以超过期望的Pod的最大数量，默认为25%。

    maxSurge: 25%

示例1，编辑pc-deployment.yaml文件

spec:

strategy: # 策略

type: Recreate # 重建更新

示例2，编辑pc-deployment.yaml文件

spec:

strategy: # 策略

type: RollingUpdate # 滚动更新策略

rollingUpdate:

maxSurge: 25%

maxUnavailable: 25%

**版本更新**

我们可以使用如下命令更新pod的镜像

# kubectl set image deployment pc-deployment nginx=nginx:1.17.2 -n dev --record

我们可以使用如下命令查看更新记录

root@k8s-master:~# kubectl rollout history deploy pc-deployment -n dev

deployment.apps/pc-deployment

REVISION CHANGE-CAUSE

1 <none>

2 kubectl set image deployment pc-deployment nginx=nginx:1.17.2 --namespace=dev --record=true

3 kubectl set image deployment pc-deployment nginx=nginx:1.17.3 --namespace=dev --record=true

我们可以使用如下命令退回某个版本

# 这里直接使用--to-revision=1回滚到了1版本， 如果省略这个选项，就是回退到上个版本，就是2版本

[root@k8s-master01 ~]# kubectl rollout undo deployment pc-deployment --to-revision=2 -n dev

查看deployment信息可以看到版本已退回

root@k8s-master:~# kubectl get deploy -n dev -o wide

NAME READY UP-TO-DATE AVAILABLE AGE CONTAINERS IMAGES SELECTOR

nginx 3/3 3 3 5d23h nginx nginx:latest app=nginx

pc-deployment 4/4 4 4 12m nginx nginx:1.17.2 app=nginx-pod