Deployment

A Deployment provides declarative updates for Pods and ReplicaSets. We can make use of many Deployment features.

This chapter covers **Updating a Deployment**, **rolling back a Deployment**, **Scalling a Deployment**, **Pausing and Resuming a Deployment** and **Deployment Status**.

First, we create a Deployment.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deploymnet
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web_server
  template:
    metadata:
      labels:
        app: web_server
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
```

- **Updating a Deployment**

  Refet to this Link: https://yeliansong.github.io/2020/04/22/K8S-Deployment-Strategies/

  It covers the rollout and rollback the deployment.

- **Scalling a Deployment**

  You can scale a Deployment by using the following command according to your requirements.

  ```
  kubectl scale deployment nginx-deploymnet --replicas=10
  ```

  Assuming horizontal Pod autoscaling is enabled in your cluster, you can setup an autoscaler for your Deployment and choose the minimum and maximum number of Pods you want to run based on the CPU utilization of your existing Pods.

  ```
  kubectl autoscale deployment nginx-deploymnet --min=10 --max=15 --cpu-percent=20
  ```

- **Pausing and Resuming a Deployment**

  You can pause a Deployment before triggering one or more updates and then resume it. This allows you to apply multiple fixes in between pausing and resuming without triggering unnecessary rollouts.

  ```
  kubectl rollout pause deployment nginx-deploymnet
  ```

  This will pause the deployment, then you can patch the changes for this deployment.

  ```
  kubectl set image deployment nginx-deploymnet nginx=nginx:1.16.1
  ```

  After that, you can resume the the Deployment.

  ```
  kubectl rollout resume deployment nginx-deploymnet
  ```

- **Deployment Status**

  The Deployment enters various status during its lifecycle. It can be progressing, complete or fail to progress.

  Deployment enter the progressing status when one of the following tasks is performed.

- The Deployment creates a new ReplicaSet
- The Deployment is scaling up its newest ReplicaSet
- The Deployment is scaling down its older ReplicaSet
- New Pods become ready or available

You can monitor the progress by using kubectl rollout status.

Enter the complete when it has the following characteristics.

- All the replicas have been updated to the latest version you've specified.
- All the replicas are available
- No old replicas for the Deployment are running.

The reasons that fail to progress when match below one.

- Insufficient quota
- Readiness probe failures
- Image pull errors
- Insufficient permissions
- Limit ranges
- Application runtime misconfiguration

The following command sets the spec with progressDeadlineSeconds to make the controller report lack of progress for a Deployment after the time you defined.

```
Kubectl patch deployment nginx-deployment -p '{"spec":{"progressDeadlineSeconds":600}}'
```