파드 생성

- kubectl run

```
kubectl run nginx-pod --image=nginx
```

- kubectl create 로 파드 생성하기 위해서는 deployment가 필요

```
kubectl create deployment dpy-nginx --image=nginx
```

차이점

- create로 파드를 생성하면 디폴로이먼트라는 관리 그룹 내에서 파드가 생성
- run으로 파드를 생성하면 디폴로이먼트가 없음
- 아래 코드는 오류

```
kubectl scale pod nginx-pod --replicas=3
```

- 정상적으로 실행

```
kubectl scale deplyment dpy-nginx --replicas=3
```

yaml

- filename : echo-hname.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: echo-hname
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
```

```
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: echo-hname
        image: sysnet4admin/echo-hname
```

```
kubectl create -f echo-hname.yaml
```

- 생성 뒤에는 yaml 파일의 replicas의 값을 6으로 변경

```
kubectl apply -f echo-hname.yaml
```

self-healing

- k8s는 거의 모든 부분이 자동 복구되도록 설계 됨
- 파드의 자동 복구 기술

노드 자원 보호

- k8s는 모든 워커 노드들에게 균등한 파드를 할당할려고 노력함
- 특정 노드가 문제 발생했을 경우 해당 노드의 기능을 잠시 정지시킬 수 있음

```
# cordon 명령을 실행하면 해당되는 노드는 SchedulingDisabled 상태로 변함
kubectl cordon [노드 이름]
```

- SchedulingDisabled된 노드는 스케줄링이 되지 않기 때문에 새로운 파드를 생성하지 않음

특정 노드에 있는 파드 종료

- 특정 노드가 문제 및 시스템 업그레이드 문제로 재가동을 해야할 경우 drain 명령어를 사용
- 지정한 노드의 파드들은 종료되며, 다른 워크 노드에 해당 파드가 생성
- DaemonSet은 각 노드에 1개만 존재하는 파드라서 drain으로는 삭제할 수 없음

```
# Daemonset을 무시하고 진행. 경고가 발생하지만 모든 파드가 다른 워크 노드로 이동(종료 후 생성)
kubectl drain [노드이름] --ignore-daemonsets
```

rolling update

- vim rolling-hname.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: rollout-nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.15.12
```

- rollling-hname.yaml 파일을 적용

```
kubectl rollout history deployment rollout-nginx
```

update

- record옵션을 사용하여 history 기록

```
kubectl set image deployment rollout-nginx nginx=nginx:1.16.0 --record

# rollout 상태 값 확인
kubectl rollout status deployment rollout-nginx
```

- 워커 노드의 인스턴스에 접속하여 파드의 ip에 아래와 같이 curl를 실행

```
# nginx의 버전이 업데이트 되어 있는걸 확인
curl -I --silent 192.168.72.31
```

update fail - rollback

- 도커 허브에 존재하지 않은 이미지의 tag 버전을 선택하여 고의적으로 오류를 발생 시킴

```
kubectl set image deployment rollout-nginx nginx=nginx:1.17.23 --record

# 상태 값 혹은 pods의 상태를 확인하면 이미지 오류가 발생하는 것을 확인할 수 있음
kubectl rollout status deployment rollout-nginx

# update의 기록을 확인
kubectl rollout history deployment rollout-nginx

# roll back
kubectl  rollout undo deployment rollout-nginx
```

create vs apply

- pod를 생성할 때 create, apply를 사용하여 생성할 수 있지만, 차이점도 존재함

	리소스가 존재하지 않을 때	리소스가 존재할 때
create	새로운 리소스 생성	Error 발생
apply	새로운 리소스 생성	리소스를 구성(부분적인 spec를 적용)