k8s에서 가장 널리 사용되는 Object이다. ReplicaSet을 이용해 Pod를 업데이트하고 이력을 관리하여 롤백하거나 특정버전으로 돌아갈 수 있다.

deployment.yml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: echo-deploy  spec:  replicas: 4  selector:  matchLabels:  app: echo  tier: app  template:  metadata:  labels:  app: echo  tier: app  spec:  containers:  - name: echo  image: ghcr.io/subicura/echo:v1 |

그리고 적용해주자.

|  |
| --- |
| $ kubectl apply -f deployment.yml |

확인 해보면

|  |
| --- |
| $ kubectl get po,rs,deploy |



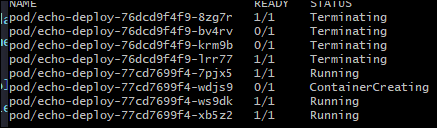
replicaset, pod, deploy 모두 생성된 것을 볼 수 있다.

그리고 버전을 바꿔 배포를 해주자.

deployment.yml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: echo-deploy  spec:  replicas: 4  selector:  matchLabels:  app: echo  tier: app  template:  metadata:  labels:  app: echo  tier: app  spec:  containers:  - name: echo  image: ghcr.io/subicura/echo:v2 |

적용하고 확인 해보면 알와 같이 기존에 있던 pod는 종료되고 새로운 podㅇ ㅣ생성 되는 것을 볼 수 있다.



**Version 관리**

|  |
| --- |
| $ kubectl rollout history deploy/echo-deploy *# 히스토리 확인*  $ kubectl rollout history deploy/echo-deploy --revision=1 *# revision 1에 대한 히스토리 상세*  $ kubectl rollout undo deploy/echo-deploy *# 바로 전 버전으로 롤백*  $ kubectl rollout undo deploy/echo-deploy --to-revision=2 *# 특정 버전으로 롤 백* |

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: echo-deploy-st  spec:  replicas: 4  selector:  matchLabels:  app: echo  tier: app  minReadySeconds: 5  **strategy:**  **type: RollingUpdate**  **rollingUpdate:**  **maxSurge: 3**  **maxUnavailable: 3**  template:  metadata:  labels:  app: echo  tier: app  spec:  containers:  - name: echo  image: ghcr.io/subicura/echo:v1  livenessProbe:  httpGet:  path: /  port: 3000 |

위와 같이 배포 전략을 작성하면 해당 모든 pod을 한 번에 끄고 다시 새로운 pod를 실행한다.