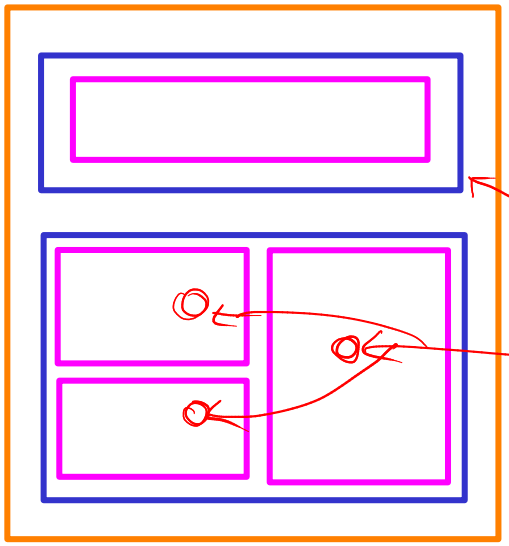
POD



Node
(Kind of 1 bare metal server)

Pod

Containers

Note] For multi container Pod, all have same network namespace
→ Same IP

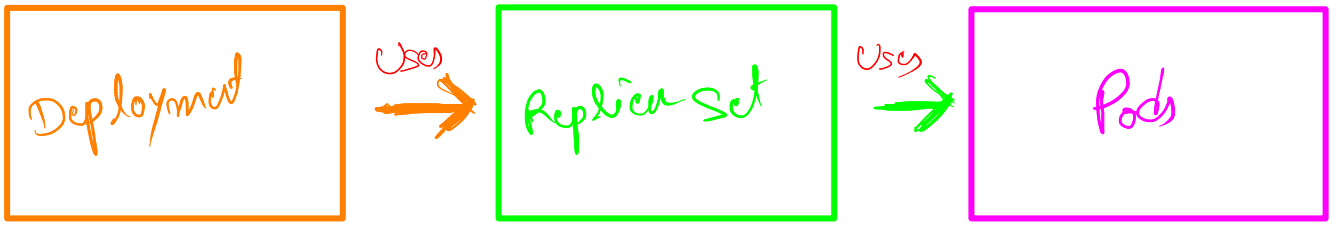
Container's IP ≡ Pod's IP

Replication Controller is deprecated

ReplicaSet → Will maintain the no of healthy pods in overall system

Deployment → Use replicaset for managing versioning

- Recreate → ① Scale down existing replicaset to 0
  ② Create new replicaset
- Rolling Update
  Do until existing one becomes 0
  ① Scale down existing replicaset by 1
  ② Increase new replicaset by 1



← Imperative Commands [check here]

for deployment, Kind : Deployment

ReplicaSet & Deployment

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: myapp-replicaset
  labels:
    app: myapp
    type: front-end
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
        type: front-end
    spec:
      containers:
        - name: nginx-container
          image: nginx
  strategy:
    type: RollingUpdate
  replicas: 3
  selector:
    matchLabels:
      type: front-end
```

Can be deployment also

→ metadata of Replicaset

→ Pod definition mainly

strategy → RollingUpdate, Recreate (Default)

→ No of replicas

→ Selector for Pod

Important Command -

```
kubectl scale --replicas=<new_no> replicaset <name>

kubectl rollout status <name>

kubectl rollout history <name>

kubectl rollout undo <name>
```

Static Pod

- This does not required Kubectl or Kube API Server
- Used in bootstrapping cluster
- We can see details in Kube API Server, but can't Control
- Managed by Kubelet
- Store Pod definition in /etc/kubernetes/manifests

Daemonsets

- When a new node get added, Daemonset will be deployed on the node
- K8S will ensure that the Pod is available on each node
