

Your Work / Kubernetes' Work



What Kubernetes Will Do

Create your objects (e.g. Pods) and manage them

Monitor Pods and re-create them, scale Pods etc.

Kubernetes utilizes the provided (cloud) resources to apply your configuration / goals



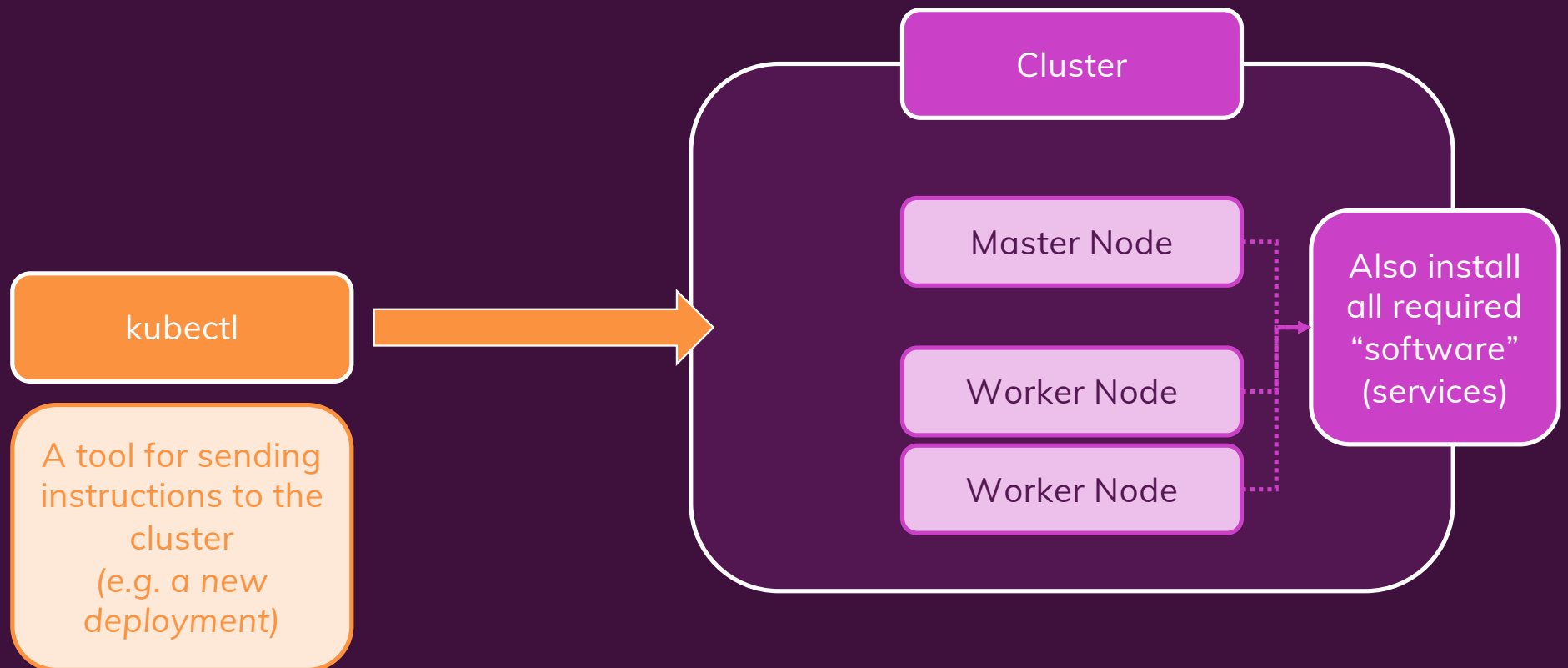
What You Need To Do / Setup *(i.e. what Kubernetes requires)*

Create the Cluster and the Node Instances (Worker + Master Nodes)

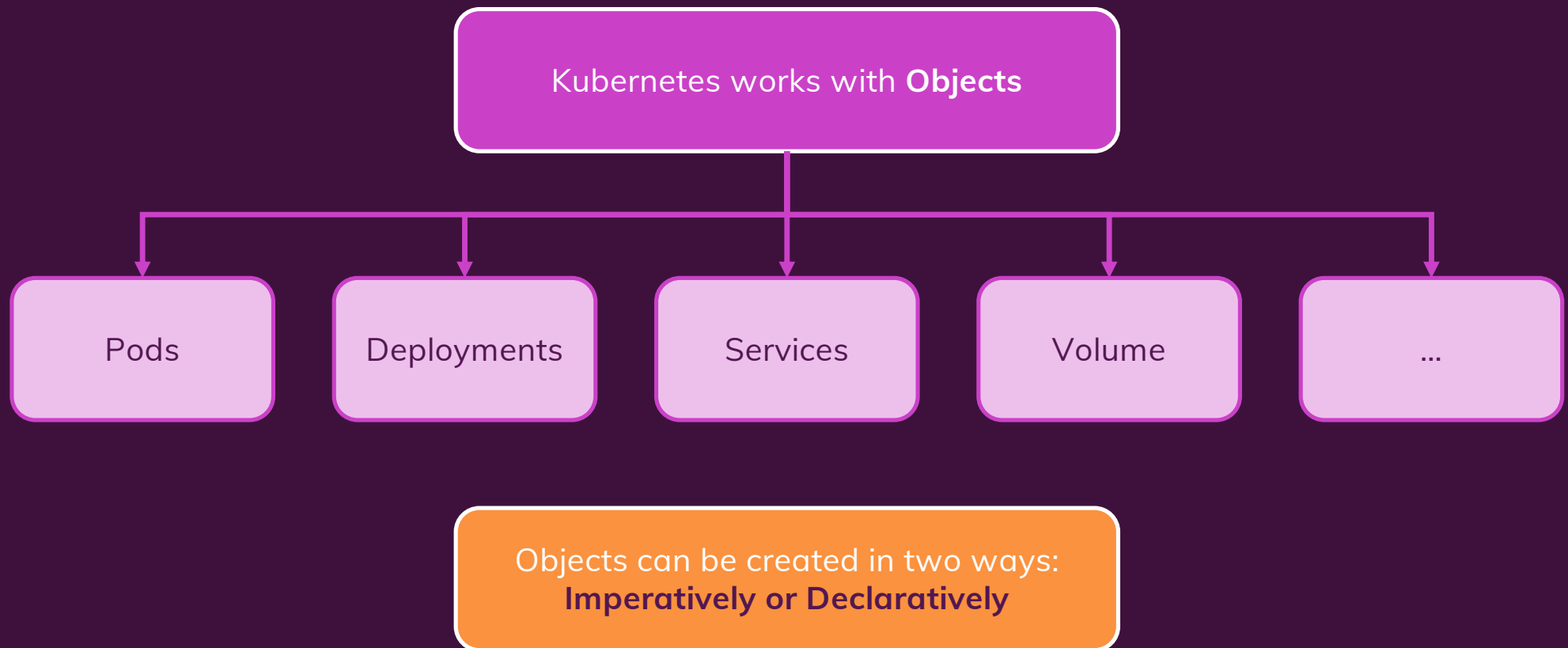
Setup API Server, kubelet and other Kubernetes services / software on Nodes

Create other (cloud) provider resources that might be needed (e.g. Load Balancer, Filesystems)

Installation



Understanding Kubernetes Objects



The “Pod” Object



The smallest “unit” Kubernetes interacts with

Contains and runs one or multiple containers

The most common use-case is “one container per Pod”

Pods contain shared resources (e.g. volumes) for all Pod containers

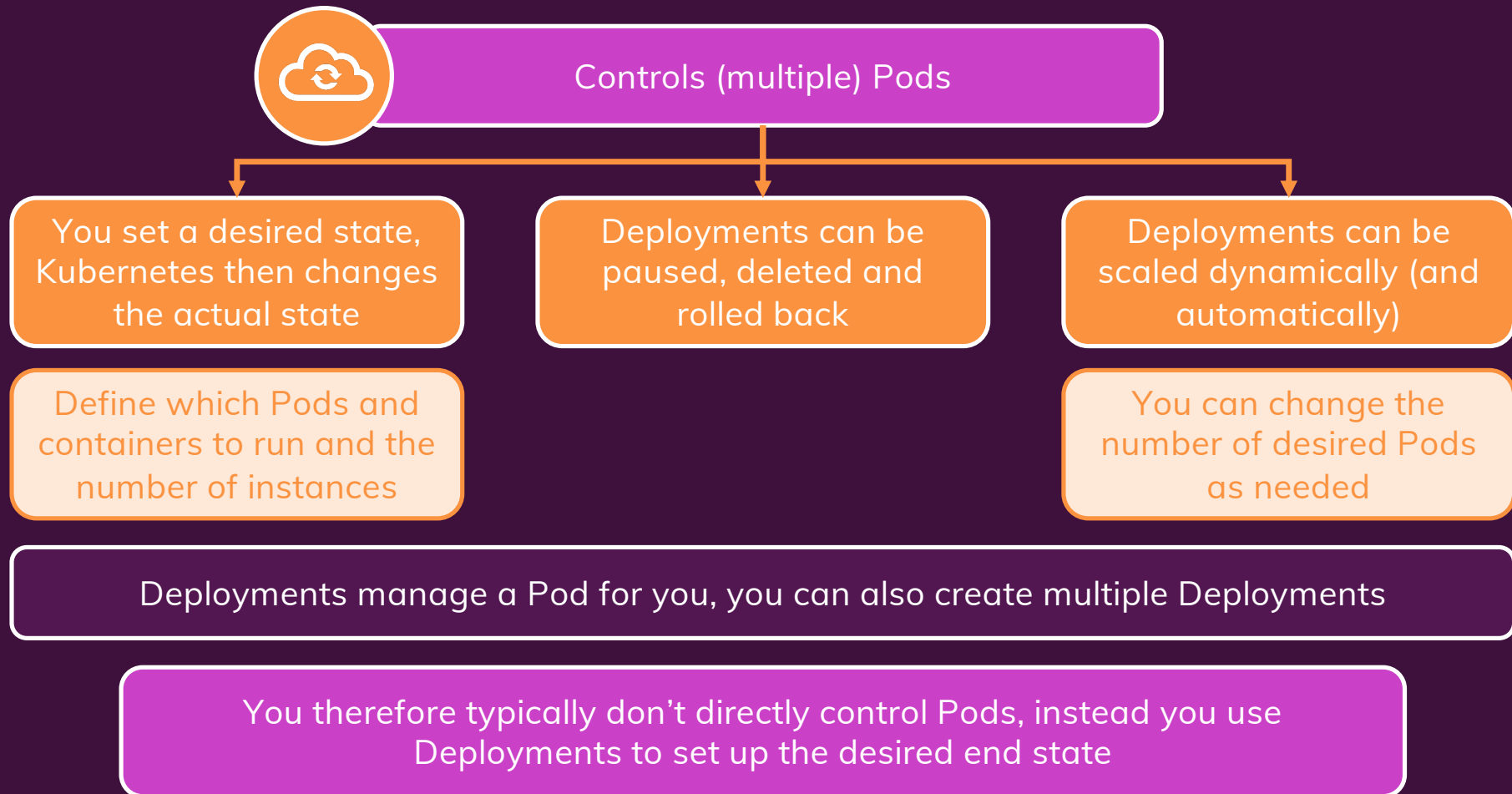
Has a cluster-internal IP by default

Containers inside a Pod can communicate via localhost

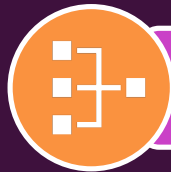
Pods are designed to be **ephemeral**: Kubernetes will start, stop and replace them as needed.

For Pods to be managed for you, you need a “**Controller**” (e.g. a “Deployment”)

The “Deployment” Object



The “Service” Object



Exposes Pods to the Cluster or Externally

Pods have an internal IP by default – it changes when a Pod is replaced

Finding Pods is hard if the IP changes all the time

Services group Pods with a shared IP

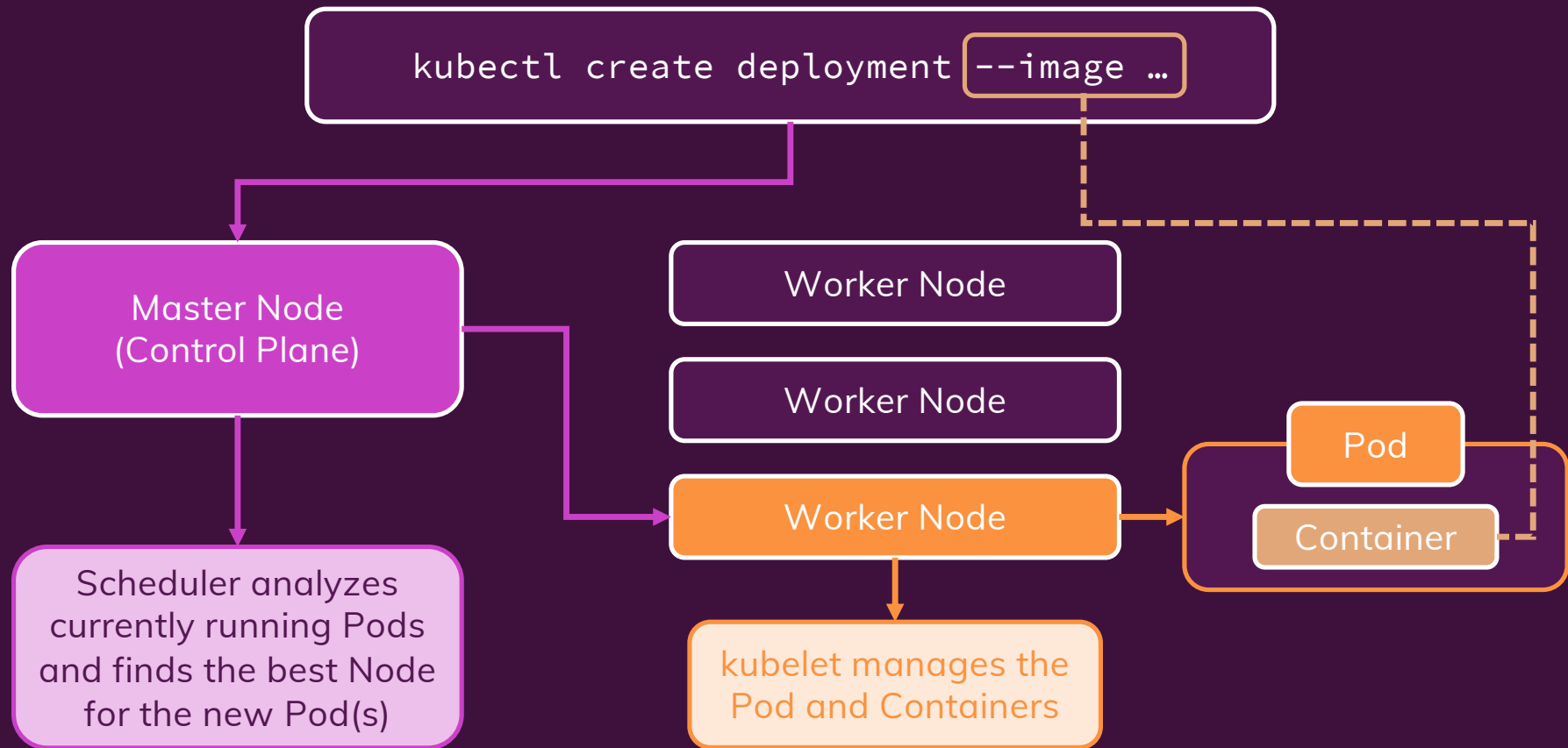
Services can allow external access to Pods

The default (internal only) can be overwritten

Without Services, Pods are very hard to reach and communication is difficult

Reaching a Pod from outside the Cluster is not possible at all without Services

Behind The Scenes



Imperative vs Declarative Kubernetes Usage

Imperative

```
kubectl create deployment ...
```

Individual commands are executed to trigger certain Kubernetes actions

Comparable to using **docker run** only

Declarative

```
kubectl apply -f config.yaml
```

A config file is defined and applied to change the desired state

Comparable to using **Docker Compose** with compose files

A Resource Definition

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: second-app
spec:
  selector:
    matchLabels:
      app: second-dummy
  replicas: 1
  template:
    metadata:
      labels:
        app: second-dummy
    spec:
      containers:
      - name: second-node
        image: "academind/kub-first-app"
```