**Kubernetes Deployments - Study Notes (Day 34)**

**Course: Complete DevOps Course**  
**Instructor: Abhishek**

**1. Introduction**

**Recap: Kubernetes Journey So Far**

✅ **Day 30:** Kubernetes vs. Docker  
✅ **Day 31:** Kubernetes Architecture  
✅ **Day 32:** Kubernetes in Production  
✅ **Day 33:** Deploying Pods

**Why Deployments?**

* Pods **lack auto-healing/scaling** capabilities.
* Deployments provide:
  + Zero-downtime updates
  + Rollback functionality
  + Replica management

**2. Key Concepts**

**1. Kubernetes Resource Hierarchy**

| **Resource** | **Purpose** | **Production Use?** |
| --- | --- | --- |
| **Container** | Runs application (Docker/runc) | ❌ No (Low-level) |
| **Pod** | Wraps 1+ containers (shared networking/storage) | ❌ Rarely direct |
| **Deployment** | Manages Pods with auto-healing/scaling | ✅ Yes |

**2. How Deployments Work**

1. **User creates deployment.yaml** (declares desired state).
2. **Deployment creates ReplicaSet** (controller).
3. **ReplicaSet ensures N Pods** are always running (auto-healing).

**3. Hands-On: Creating a Deployment**

**Step 1: deployment.yaml**

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

replicas: 3 # Desired number of Pods

selector:

matchLabels:

app: nginx

template: # Pod template

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

**Step 2: Apply the Deployment**

kubectl apply -f deployment.yaml

**Step 3: Verify Resources**

kubectl get deploy # Check Deployment

kubectl get rs # Check ReplicaSet

kubectl get pods -o wide # Check Pods (should show 3)

**Step 4: Test Auto-Healing**

kubectl delete pod <pod-name> # Delete a Pod

kubectl get pods -w # Watch ReplicaSet recreate it

**Step 5: Scale the Deployment**

kubectl scale deploy nginx-deployment --replicas=5

# OR edit deployment.yaml → `replicas: 5` → `kubectl apply -f deployment.yaml`

**4. Key Features of Deployments**

**1. Auto-Healing**

* ReplicaSet ensures **desired Pod count** is maintained.
* Example: If a Pod crashes, a new one is created automatically.

**2. Rolling Updates**

* Update Pods **without downtime** (gradually replaces old Pods).

kubectl set image deploy nginx-deployment nginx=nginx:1.16.1

**3. Rollbacks**

* Revert to a previous version if something goes wrong.

kubectl rollout undo deploy nginx-deployment

**5. Interview Questions**

**Q1: Pod vs. Deployment?**

* **Pod:** Lowest deployable unit (1+ containers). No auto-healing.
* **Deployment:** Manages Pods with auto-healing/scaling.

**Q2: What is a ReplicaSet?**

* A **controller** that ensures N Pod replicas are running.
* Created automatically by Deployments.

**Q3: How to debug a Deployment?**

kubectl describe deploy <name> # Check events/errors

kubectl logs <pod-name> # Check container logs

kubectl get events # Cluster-wide events

**6. Summary**

* **Deployments > Pods** for production (auto-healing, scaling).
* **ReplicaSet** is the "worker" ensuring Pod count.
* **Always use Deployments** (not direct Pods) in real-world scenarios.

🚀 **Assignment:**

1. Deploy a custom app (e.g., Python/Go) using a Deployment.
2. Test auto-healing by deleting Pods.
3. Scale from 3 → 5 replicas.

📌 **Pro Tip:** Bookmark the [Kubernetes Deployment Docs](https://kubernetes.io/docs/concepts/workloads/controllers/deployment/).

📢 **Feedback?** Comment below! 👍 **Like & Share** if this helped!

**End of Notes** 🎉

# \*\*Kubernetes Deployments - Study Notes (Day 34)\*\*

\*\*Course: Complete DevOps Course\*\*

\*\*Instructor: Abhishek\*\*

---

## \*\*1. Introduction\*\*

### \*\*Recap: Kubernetes Journey So Far\*\*

✅ \*\*Day 30:\*\* Kubernetes vs. Docker

✅ \*\*Day 31:\*\* Kubernetes Architecture

✅ \*\*Day 32:\*\* Kubernetes in Production

✅ \*\*Day 33:\*\* Deploying Pods

### \*\*Why Deployments?\*\*

- Pods \*\*lack auto-healing/scaling\*\* capabilities.

- Deployments provide:

- Zero-downtime updates

- Rollback functionality

- Replica management

---

## \*\*2. Key Concepts\*\*

### \*\*1. Kubernetes Resource Hierarchy\*\*

| \*\*Resource\*\* | \*\*Purpose\*\* | \*\*Production Use?\*\* |

|----------------|---------------------------------------------|---------------------|

| \*\*Container\*\* | Runs application (Docker/runc) | ❌ No (Low-level) |

| \*\*Pod\*\* | Wraps 1+ containers (shared networking/storage) | ❌ Rarely direct |

| \*\*Deployment\*\* | Manages Pods with auto-healing/scaling | ✅ Yes |

### \*\*2. How Deployments Work\*\*

1. \*\*User creates `deployment.yaml`\*\* (declares desired state).

2. \*\*Deployment creates ReplicaSet\*\* (controller).

3. \*\*ReplicaSet ensures N Pods\*\* are always running (auto-healing).

![Deployment Flow](https://d33wubrfki0l68.cloudfront.net/152c845f25df8e69dd24dd7b0836a289747e258a/4a1d2/docs/tutorials/kubernetes-basics/public/images/module\_02\_first\_app.svg)

---

## \*\*3. Hands-On: Creating a Deployment\*\*

### \*\*Step 1: `deployment.yaml`\*\*

```yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

replicas: 3 # Desired number of Pods

selector:

matchLabels:

app: nginx

template: # Pod template

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

```

### \*\*Step 2: Apply the Deployment\*\*

```sh

kubectl apply -f deployment.yaml

```

### \*\*Step 3: Verify Resources\*\*

```sh

kubectl get deploy # Check Deployment

kubectl get rs # Check ReplicaSet

kubectl get pods -o wide # Check Pods (should show 3)

```

### \*\*Step 4: Test Auto-Healing\*\*

```sh

kubectl delete pod <pod-name> # Delete a Pod

kubectl get pods -w # Watch ReplicaSet recreate it

```

### \*\*Step 5: Scale the Deployment\*\*

```sh

kubectl scale deploy nginx-deployment --replicas=5

# OR edit deployment.yaml → `replicas: 5` → `kubectl apply -f deployment.yaml`

```

---

## \*\*4. Key Features of Deployments\*\*

### \*\*1. Auto-Healing\*\*

- ReplicaSet ensures \*\*desired Pod count\*\* is maintained.

- Example: If a Pod crashes, a new one is created automatically.

### \*\*2. Rolling Updates\*\*

- Update Pods \*\*without downtime\*\* (gradually replaces old Pods).

```sh

kubectl set image deploy nginx-deployment nginx=nginx:1.16.1

```

### \*\*3. Rollbacks\*\*

- Revert to a previous version if something goes wrong.

```sh

kubectl rollout undo deploy nginx-deployment

```

---

## \*\*5. Interview Questions\*\*

### \*\*Q1: Pod vs. Deployment?\*\*

- \*\*Pod:\*\* Lowest deployable unit (1+ containers). No auto-healing.

- \*\*Deployment:\*\* Manages Pods with auto-healing/scaling.

### \*\*Q2: What is a ReplicaSet?\*\*

- A \*\*controller\*\* that ensures N Pod replicas are running.

- Created automatically by Deployments.

### \*\*Q3: How to debug a Deployment?\*\*

```sh

kubectl describe deploy <name> # Check events/errors

kubectl logs <pod-name> # Check container logs

kubectl get events # Cluster-wide events

```

---

## \*\*6. Summary\*\*

- \*\*Deployments > Pods\*\* for production (auto-healing, scaling).

- \*\*ReplicaSet\*\* is the "worker" ensuring Pod count.

- \*\*Always use Deployments\*\* (not direct Pods) in real-world scenarios.

🚀 \*\*Assignment:\*\*

1. Deploy a custom app (e.g., Python/Go) using a Deployment.

2. Test auto-healing by deleting Pods.

3. Scale from 3 → 5 replicas.

📌 \*\*Pro Tip:\*\* Bookmark the [Kubernetes Deployment Docs](https://kubernetes.io/docs/concepts/workloads/controllers/deployment/).

📢 \*\*Feedback?\*\* Comment below! 👍 \*\*Like & Share\*\* if this helped!

---

\*\*End of Notes\*\* 🎉