ReplicationController

ReplicationController (RC) is an older Kubernetes object used to ensure that a specified number of Pods are always running.

But today, it's largely replaced by ReplicaSet (which is used under the hood by Deployments).

Purpose of ReplicationController

- Keeps a specific number of Pods running at all times
- If a Pod crashes or is deleted, a new one is started

ReplicationController YAML – Sections Checklist

```yaml
apiVersion: v1
kind: ReplicationController
metadata:
  name: nginx-rc
spec:
  replicas: 2
  selector:
    app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - containerPort: 80
```

Block diagram represenation of above yaml

```
+-------------------------------------------------+
|         ReplicationController (nginx-rc)        |
|-------------------------------------------------|
|  kind: ReplicationController                    |
|  replicas: 2                                    |
|  selector: app=nginx                            |
|  template:                                      |
|    metadata.labels.app: nginx                   |
+-------------------------------------------------+
                |
                | creates/ensures
                ▼
     +---------------------------+       +---------------------------+
     |           Pod 1           |       |           Pod 2           |
```

```
      |--------------------------|       |--------------------------|
      | metadata:                |       | metadata:                |
      |   labels:                |       |   labels:                |
      |     app: nginx           |       |     app: nginx           |
      | spec:                    |       | spec:                    |
      |   containers:            |       |   containers:            |
      |     - name: nginx        |       |     - name: nginx        |
      |       image: nginx:latest|       |       image: nginx:latest|
      |       port: 80           |       |       port: 80           |
     +--------------------------+      +--------------------------+
```

Part	Purpose
`apiVersion`, `kind`	Define the resource type (`ReplicationController`)
`metadata.name`	Unique name of the controller
`spec.replicas`	Desired number of Pods
`spec.selector`	What labels to match in Pods
`spec.template`	The **Pod definition** to create and manage
Pod `containers`	Image, name, ports — same as in standalone Pod YAML

ReplicaSet

A ReplicaSet is a Kubernetes controller that ensures a specified number of Pods are always running. It is more powerful than ReplicationController because it supports label selectors with set-based matching, and it's the default backend controller used by Deployments.

Why Use ReplicaSet

- Ensures desired number of Pods are running
- Replaces failed or deleted Pods
- Matches Pods using advanced label filters
- For updates, use Deployment, not raw RS

replicas in ReplicaSet

- replicas specifies how many Pods you want the ReplicaSet to maintain at all times.
- If the actual number of Pods is less than this, the ReplicaSet creates new Pods.
- If it’s more, it deletes extra Pods to match the desired count.
- This ensures your app stays highly available and scalable.

selector in ReplicaSet

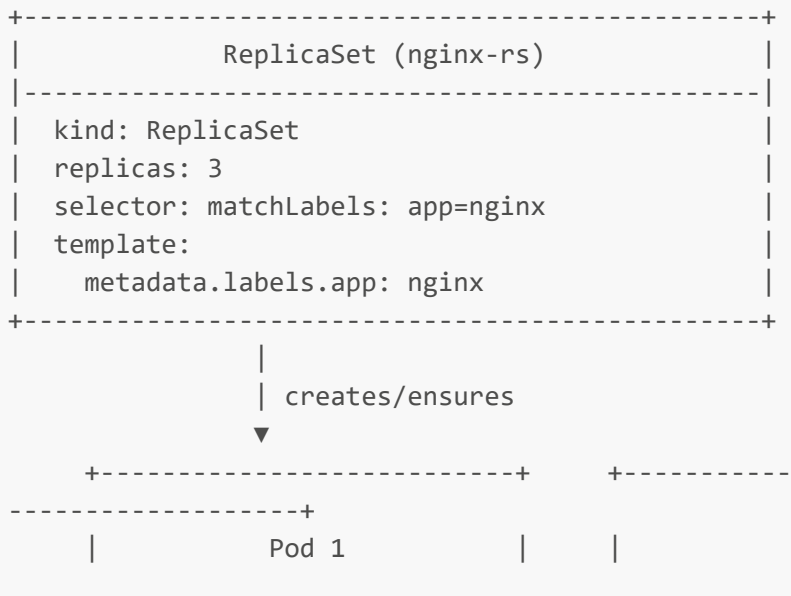
- The selector tells the ReplicaSet which Pods it should manage.
- It matches Pods by their labels — specifically, the labels that Pods must have to be considered “owned” by this ReplicaSet.
- In ReplicaSet, selectors usually use matchLabels or matchExpressions to filter Pods.

- The selector must match exactly with the labels defined in the Pod template (template.metadata.labels), or else the ReplicaSet will not manage those Pods.

ReplicaSet YAML – Sections Checklist

```yaml
apiVersion: apps/v1                  # API group for ReplicaSet (modern and required)
kind: ReplicaSet                     # Type of Kubernetes object
metadata:
  name: nginx-rs                     # Name of the ReplicaSet

spec:
  replicas: 3                        # Desired number of Pods to run

  selector:                          # How the ReplicaSet finds matching Pods
    matchLabels:                     # Must match labels in the pod template below
      app: nginx                     # Label used to select/manage the Pods

  template:                          # Pod template - defines what Pods to create
    metadata:
      labels:
        app: nginx                   # Labels assigned to Pods (must match selector above)

    spec:
      containers:                    # List of containers inside each Pod
        - name: nginx                # Name of the container
          image: nginx:latest        # Docker image to use for the container
          ports:
            - containerPort: 80      # Port exposed by the container (for service access)
```

Block diagram represents the ReplicaSet



```
Pod 3           |
     |--------------------------|     |--------------------------|     |--------
------------------|
     | metadata:                |     | metadata:                |     |
metadata:                 |
     |   labels: app=nginx      |     |   labels: app=nginx      |     |
labels: app=nginx       |
     | spec:                    |     | spec:                    |     | spec:
|
     |   containers:            |     |   containers:            |     |
containers:             |
     |     - name: nginx        |     |     - name: nginx        |     |     -
name: nginx         |
     |       image: nginx:latest |     |       image: nginx:latest |     |
image: nginx:latest |
     |       port: 80           |     |       port: 80           |     |
port: 80            |
     +--------------------------+     +--------------------------+     +--------
------------------+
```

Commands to write replicaSet

```
kubectl get rs                    # List all ReplicaSets in current namespace
kubectl describe rs <rs-name>    # Show detailed info about a specific ReplicaSet
kubectl apply -f replicaset.yaml # Create or update ReplicaSet from YAML file
kubectl delete rs <rs-name>      # Delete a ReplicaSet
kubectl scale rs <rs-name> --replicas=3  # Scale ReplicaSet to desired number of
pods
kubectl get pods -l app=nginx    # List Pods managed by ReplicaSet (filter by
label)
```