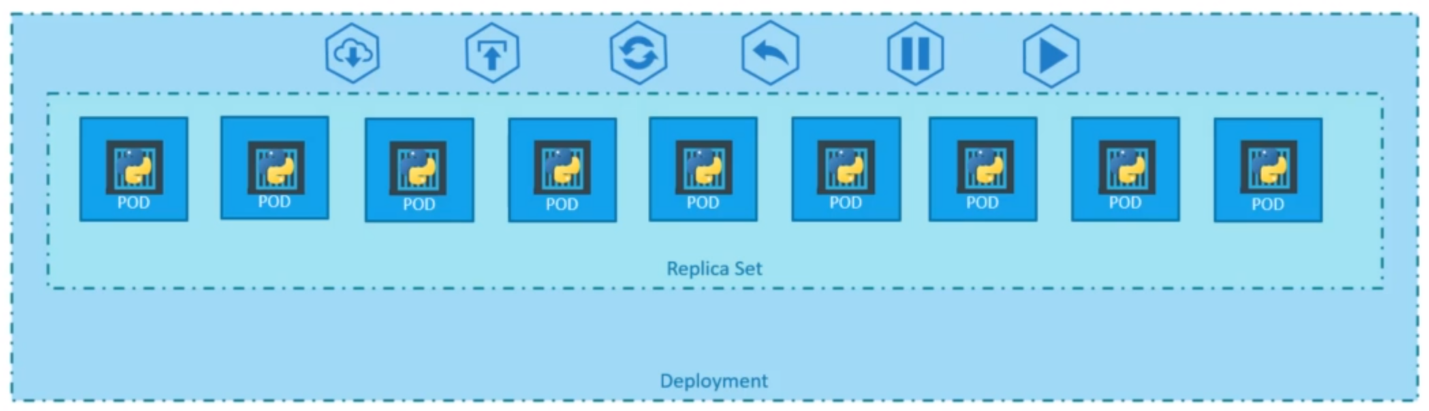
**How you can deploy your application in a production environment?**

For example, you have a web server that needs to be deployed in a production environment. You need not one but many such instances of the web server running for obvious reasons. Secondly, whenever newer versions of application builds become available on the Docker registry, you would like to upgrade your Docker instances seamlessly.

However, when you upgrade your instances, you do not want to upgrade all of them at once because this may impact users accessing your applications so you might want to upgrade them one after the other and that kind of update is known as rolling updates. Suppose one of the upgrades you performed resulted in an unexpected error and you’re asked to undo the recent change, you would like to be able to roll back the changes that were recently carried out.

Finally, let’s say you would like to make multiple changes to your environment such as upgrading the underlying Web Server versions as well as scaling your environment and also modifying the resource allocations etc., you do not want to apply each change immediately after the command is run, instead you like to apply a pause to your environment, make the changes and then resumes so that all the changes are rolled out together. All of these capabilities are available with the Kubernetes deployment.

So far, we are aware about pods which deploy single instance of our application such as the web application. Each container is encapsulated in pods, multiple such pods are deployed using replication controllers or replica sets and then comes deployment which is a Kubernetes object that comes higher in hierarchy, the deployment provides us with the capability to upgrade the underlying instances seamlessly using rolling updates, undo changes, pause and resume changes as required.



**How do we create a deployment?**

We have to first create a deployment definition file, the contents of the deployment definition file are exactly similar to the replica set definition file, except for the “kind” which is now going to be “deployment”. Once the file is ready run the Kubectl create command and specify the deployment definition file. Then, run the Kubectl get deployments command to see the newly created deployment. The deployment automatically creates a replica set.



So, if you run the Kubectl get replica set command you will see a new replica set in the name of the deployment. The replica set ultimately create pods so, if you run the Kubectl get pods command you will see the pod with the name of the deployment and the replica set. So far, there hasn’t been much of a difference between replica set and deployments, except for the fact that deployment created a new Kubernetes objects called deployments.

To see all the created objects at once run the below command:

