**Creating a Release in Helm**

We've got to created a Helm Chart, create a version, then create a second version, all so we can get the hang of updating releases and rolling back to previous ones. Let's jump right in with both feet.

Log In to the Kube Master

Before we can do anything in Kubernetes or Helm, we've got to log in to the Kube Master. Get its public IP from the lab page, as well as the username and password, and log in with SSH.

Modify and Release a Version

Update values.yaml

Locate the nginx chart that is in the home directory:

$ cd ~

$ ls -l

We should see a directory named nginx there. Get into that, then look around again, and we should see values.yaml.

$ cd nginx

$ ls -l

Edit values.yaml file (with whatever editor you like -- we're using Vim here by running vim ./values.yaml. Find the section of the file that currently reads:

index: >-

<h1>Hello</h1>

<p>This is a test</p>

Let's change it a bit, to read:

index: >-

<h1>Hello</h1>

<p>This is version 1</p>

We've also got to set the service to NodePort and the HTTP port to 30080. Change those two lines so that the whole section matches this:

service:

annotations: {}

clusterIP: ""

externalIPs: []

loadBalancerIP: ""

loadBalancerSourceRanges: []

type: NodePort

port: 8888

nodePort: "30080"

Update the Version Number

We'll have to edit Charts.yaml in order to set a version number. Use whatever editor you like and update the *version* value. It should look something like this:

version: 0.2.0

Release the Chart

Now we're ready to release our Chart. We need to get back to our home directory and run a helm command:

$ cd ~

$ helm install nginx

Confirm the Version Number of the Released Chart

Once that helm command has completed, we need to run a couple more. This one returns the nginx release name:

$ helm ls --short

And this one shows the version number of the released Chart:

$ helm history <release name>

Obtain the IP address of one of the cluster nodes (on the same page where we got the Kube Master's public IP) and navigate to *IP:Port* in a web browser (*http://(node\_IP)"30080"*).

We should see the *Hello This is version 1* message that we typed in the values.yaml file.

Confirm that the index page is correct.

Modify and Release a Newer Version

We're going to do this updating and releasing again, so that we can practice rolling back afterwards. Let's get right into it.

Update the Index Data and Increment the Version Number

Let's update the contents of the index section like we did before. It doesn't matter what we put there, just as long as it's different than it was before. This would work:

index: >-

<h1>Hello</h1>

<p>This is some new text</p>

Then let's increment the version in Chart.yaml so that the line reads:

version: 0.3.0

Upgrade the Release

Just like before, copy the release name that we get from a helm ls command and then upgrade using that name:

$ helm ls --short

$ helm upgrade <release name> ~/nginx

Verify the new version of the release by checking the chart name in the helm history:

$ helm history <release name>

Load the index page again (using the node's public IP like we did last time) and confirm the content reflects what we changed in values.yaml. It may take a couple of minutes for changes to show up over there.

Rollback the Release to the Previous Version

We're going roll back our current release to a previous one (which is why we made two in the first place).

View History

One more time, let's get the nginx release name with a helm ls --short, then use it to see the available revisions we can choose from:

$ helm ls --short

$ helm history <release name>

The output is a table of data. We can see which one is *DEPLOYED*, and which one is *SUPERCEDED*.

Roll It Back

We're going to roll back to that *SUPERCEDED* one. In the *REVISION* column, we can see that it's **#1**. Here's how we're going to do it:

$ helm rollback <release name> 1

Just remember, down the road, that we'd substitute 1 with the correct version number of whatever project we're dealing with.

Confirm That the Correct Version is Now Deployed

Let's prove that we actually rolled back. Use the same command we've been running to look at the history:

$ helm history <release name>

And verify the correct version is listed as *DEPLOYED*

If we allow the pods volumes a few minutes to update, then check the web page content, we should see the first batch of content we changed.

Conclusion

We've created a Helm Chart, created a version, then created *another* version, and managed to roll back to the first one.