RANCHER

Create and Use a ConfigMap and a Secret with a Workload

Lab 31

What are you Learning?

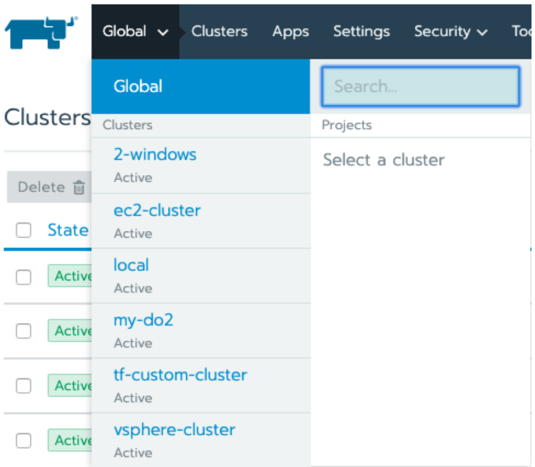
You will learn how to create and use both ConfigMaps and Secrets in your workloads.

Why is it important?

These objects allow you to dynamically inject state into your Kubernetes workloads. This allows you to keep your container images more generic and largely immutable instead of having to rebuild each time there is a configuration change. Separating "config" and "code" is also a principle of 12-factor app methodology.

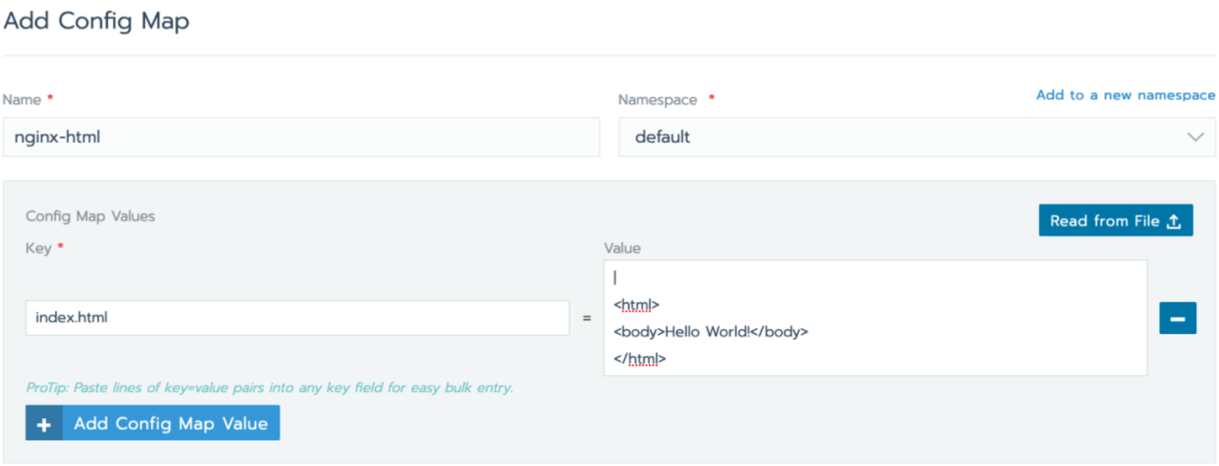
Activating Monitoring

1. In the Rancher UI Navigate to the cluster and project you would like to use for this exercise by selecting it from a list of clusters in the top left Menu. Select a project from the right part of the menu



2. The top menu bar now reflects the available options in the project context. Select "Resources" and then "Config". You can see the available ConfigMaps in the current project.
3. Create a new ConfigMap and call it "dev-parameters". Define a couple of important key value pairs we want our app to reference. These can later become environment variables or files in our container. Select "Add ConfigMap" in the top right and fill out the page accordingly. Define the following key pairs:
   a. DEBUG=TRUE
   b. PROFILE=DEV
4. We can also store long-form strings to be used as a file in our container. Create a new ConfigMap named "nginx-html" with a key called "index.html" and the following contents (the first character is a vertical bar, which indicates that a multi-line string follows):

```
|
<html>
<body>Hello World!</body>
</html>
```



Add Config Map

Name *: nginx-html

Namespace *: default

Config Map Values

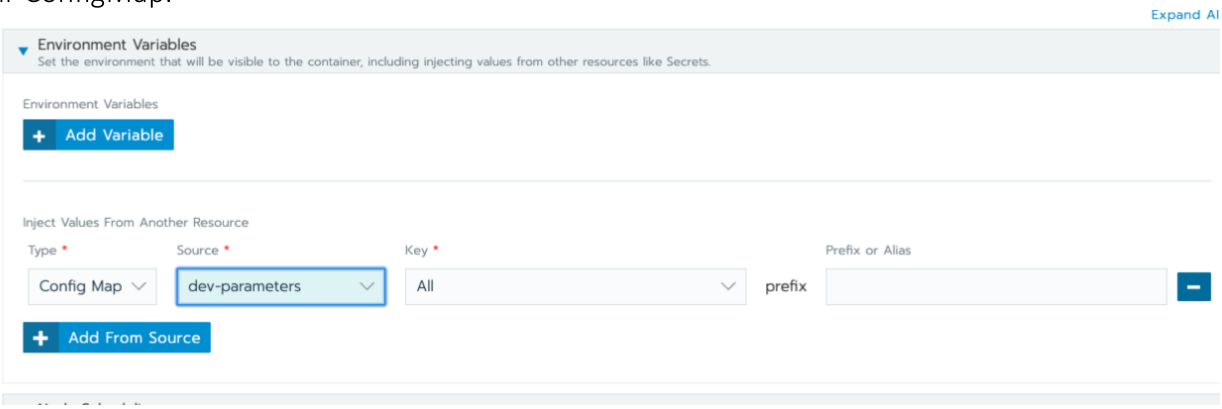
Key *: index.html

Value:

```
|
<html>
<body>Hello World!</body>
</html>
```

ProTip: Paste lines of key=value pairs into any key field for easy bulk entry.

5. Repeat this to create a Secret, which you can do by clicking Resources and then Secrets from the top menu. Create a Secret named “rancher-secret” with the following key and value:
    a. PASSWORD=RancherRox
6. Go back to our workload we previously created called nginx, and now add in our “dev-parameters” ConfigMap as environment variables. Select “Edit” on the workload, and then on the page that opens, expand the “Environment Variables Section” to define the variables. Then select “Add From Source” and select the ConfigMap we previously defined. If we set “Keys” to “All”, it will create an environment variable for each key in our ConfigMap:



Environment Variables

Set the environment that will be visible to the container, including injecting values from other resources like Secrets.

Environment Variables

Inject Values From Another Resource

Type *: Config Map

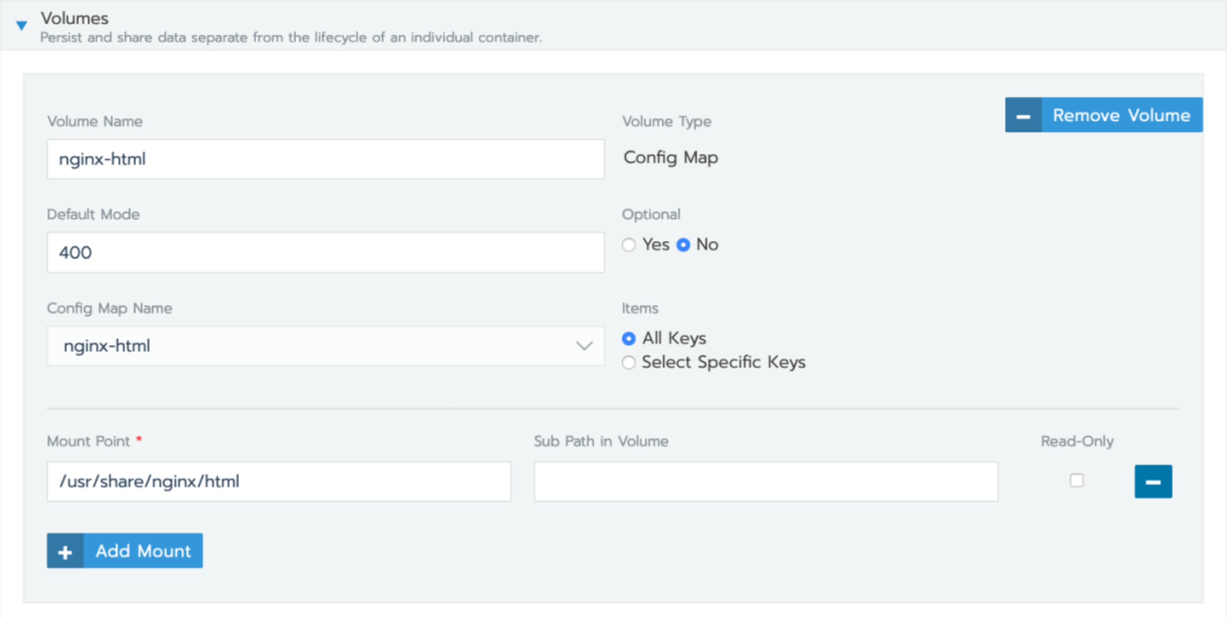
Source *: dev-parameters

Key *: All

Prefix or Alias: prefix

7. Repeat this step to add the Secret we created.
8. Add the “nginx-html” ConfigMap as a volume. Expand the “Volumes” section and add a Volume from a ConfigMap. Name it “nginx-html” and choose our “nginx-html”

ConfigMap that we created above. Set “/usr/share/nginx/html” as the mount point.



Volumes
Persist and share data separate from the lifecycle of an individual container.

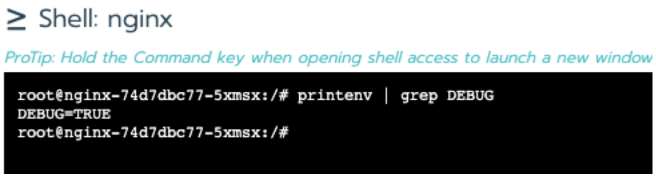
Volume Name: nginx-html
Volume Type: Config Map
Default Mode: 400
Optional: ○ Yes ● No
Config Map Name: nginx-html
Items: ● All Keys ○ Select Specific Keys
Mount Point *: /usr/share/nginx/html
Sub Path in Volume:
Read-Only: □

9. Save and let the workload update.

Testing That It Works

To test that this works, shell into one of the containers for the workload and inspect the environment and filesystem. For the environment variables, you can type “printenv” or “set” to get a list of environment variables.

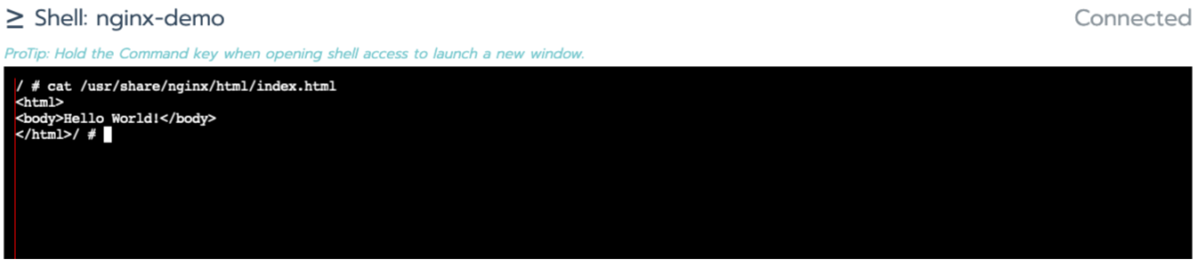
For example, to check for the “DEBUG” var we set previously, I could run “printenv | grep DEBUG”.



≥ Shell: nginx
ProTip: Hold the Command key when opening shell access to launch a new window

```
root@nginx-74d7dbc77-5xmsx:/# printenv | grep DEBUG
DEBUG=TRUE
root@nginx-74d7dbc77-5xmsx:/#
```

Look in /usr/share/nginx/html and you should see a file called “index.html” that has the contents of our ConfigMap.



≥ Shell: nginx-demo
Connected
ProTip: Hold the Command key when opening shell access to launch a new window.

```
/ # cat /usr/share/nginx/html/index.html
<html>
<body>Hello World!</body>
</html>/ #
```

Edit the HTML ConfigMap and change its contents. Go back and look at the file in the container. Did it update?

What if you change one of the environment variables? Do those also update in real time?

References

- ConfigMaps (Rancher Docs)- https://rancher.com/docs/rancher/v2.x/en/k8s-in-rancher/configmaps/
- ConfigMaps (Kubernetes Docs) - https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/
- 12-Factor Apps - https://12factor.net/