Cheat sheet

OpenShift command line essentials

As you progress in your journey of mastering OpenShift, you will find yourself using the command line more and more. Given that scripting a solution is the key component of creating automated and reproducible results, command-line expertise becomes paramount to elegant solutions.

In short: The command line makes life easier.

This cheat sheet covers 12 essential commands for the OpenShift command line. There are activities that you will come back to again and again, such as copying files to a pod or a Persistent Volume Claim (PVC). Others are more obscure operations, such as editing a deployment from your local machine. Some commands are more developer-centric, like an endless `curl` loop that lets you watch a rolling update in real time.

While this cheat sheet is by no means exhaustive, it has several "must-have" commands and—more importantly—can serve to inspire you to create your own library of commands and scripts.

Install the OpenShift command-line interface

Red Hat OpenShift features a web-based dashboard for configuration and operation. It also supports a command-line interface, `oc`, that allows you to control clusters via a terminal session.

The power of the `oc` command means you have complete control over your cluster, objects, and more. Combining that power with other scripting elements gives developers and operations the ability to create reproducible results via commands and scripts.

All of this begins when you install the `oc` command on your PC, whether it be on Linux, macOS, or Windows.

You will need to log into your OpenShift sandbox in your browser. This lands you at the dashboard.

Before diving into the OpenShift command-line interface (CLI), the CLI must be installed on your PC. This, ironically, is best done by starting at the OpenShift web-based dashboard. After opening the dashboard, follow these steps:

1. Find the help icon (the question mark) in the upper-right corner and click it.
2. From the drop-down menu, select the **Command Line Tools** option.
3. Select the appropriate download for your operating system and hardware and save the file to a convenient location.

With the installation file downloaded, expand the compressed ZIP file and move the executable file into a directory that is in your system path.

For **Windows PowerShell**, run as Administrator and use the following commands:

```
1. Expand-Archive oc.zip
2. cd .\oc\
3. mkdir $ENV:ProgramFiles\OpenShift-Client
4. Move-Item oc.exe $ENV:ProgramFiles\OpenShift-Client\
5. $current_path = (Get-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Session
   Manager\Environment' -Name "PATH").path
6. $new_path = "$current_path;$ENV:ProgramFiles\OpenShift-Client"
7. Set-ItemProperty -Path 'Registry::HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session
   Manager\Environment' -Name path -Value $new_path
```

Close the terminal and open a new session to fetch the new system Path.

For **macOS**, use the following commands:

```
unzip oc.zip
sudo mv oc /usr/local/bin
```

For **Linux**, use the following commands:

```
tar xvf oc.tar
sudo mv oc /usr/bin
```

You can now run the `oc version` command to prove that the installation was successful, such as the following example:

```
PS C:\Users\foo> oc version
Client Version: 4.14.0-202310201027.p0.g0c63f9d.assembly.stream-0c63f9d
Kustomize Version: v5.0.1
error: You must be logged in to the server (Unauthorized)
```

Log in to the Developer Sandbox

In your web-based dashboard, locate and click on your project name in the upper-right corner; then, select **Copy login command**. If prompted, click **DevSandbox** to continue.

You are now presented with one option: **Display token**. Click the link to display your login command (Figure 1).

Figure 1: Copy the oc login command to your copy and paste buffer.

Copy the line labeled **Log in with this token** to your machine's copy-and-paste buffer.

Switch to your command line, paste the command, and click **Enter** to log in, as in the following example:

```
PS C:\Users\foo> oc login
–token=sha256~YQZ2YHfoUmaN3BEo-ZlRama8VCKYnogtbG9bDFWaHZU
–server=https://api.sandbox-m3.1530.p1.openshiftapps.com:6443
Logged into “https://api.sandbox-m3.1530.p1.openshiftapps.com:6443” as “rhn-engineering-dsch” using the token provided.
You have one project on this server:“rhn-engineering-dsch-dev”
Using project “rhn-engineering-dsch-dev”.
```

You are now logged into your cluster at the command line and can use commands such as `oc` and `kubectl` to work with your cluster.

Create a new app from an image

Using the Developer Sandbox for Red Hat OpenShift and the `oc` command, it is possible to create an application from the command line using an existing image.

Before launching the command, think about what you will name your app and which labels you wish to assign to your app. Note that the name and labels do not need to agree with each other, nor do they need to align with the name of the image. This presents an opportunity where poor planning can create confusion; plan ahead and proceed carefully.

Using the image at `quay.io/rhdevelopers/filecontents:1.0.0` , here's the command to create it:

```
oc new-app  –image=quay.io/rhdevelopers/filecontents:1.0.0
–name=filecontents –labels=sandbox=filecontents
```

Then, expose it to the world via a URL with this command:

```
oc expose service/filecontents
```

You can see the results in your OpenShift dashboard.

Create a new app from source code

Using the Developer Sandbox for Red Hat OpenShift and the oc command, it is possible to create an application from the command line using source code that exists in a Git repo.

Use the `oc new-app` command to create an app from source code. Here's an example that creates an app written in Go:

```
oc new-app https://github.com/redhat-developer-demos/qotd.git –name=qotd –labels=sandbox=qotd
```

Then, expose it to the world via a URL with this command:

```
oc expose service/qotd
```

You can best view the results using an endless `curl` loop.

Get the route for a deployment

You'll often need to get the route (i.e., the public URL) for an app in your OpenShift cluster.
Using your own project name (i.e., the `-n myproject` parameter) and your deployment name, run the following command:

```
oc get route deploymentnamegoeshere –n myproject -o jsonpath='{"http://"}{.spec.host}'
```

Example:

```
PS C:> oc get route –n rhn-engineering-dsch-dev resource-pressurizer -o jsonpath='{"http://"}{.spec.host}'
http://resource-pressurizer-rhn-engineering-dsch-dev.apps.sandbox-m3.1530.p1.openshiftapps.com
```

Get pods for a deployment

Here's how to get the pods for an app in your OpenShift cluster with a simple command.
Using your deployment name, run:

```
oc get pods –selector deployment=deploymentnamegoeshere -o custom-columns=NAME:.metadata.name –no-headers
```

Example:

```
PS C:> oc get pods –selector deployment=resource-pressurizer -o custom-columns=NAME:.metadata.name –no-headers
resource-pressurizer-85b577f48d-pnc6g
```

Edit a deployment with oc edit

Sometimes, especially when developing and debugging, being able to edit an object while it is in OpenShift is valuable. The `oc edit` command enables this.

Using your deployment name, run the following command:

```
oc edit deploy/deploymentnamegoeshere
```

The default editor of your PC will open with the deployment YAML.

Near the top, under the `labels` section, add a new label (suggestion: `editedby: yourname`). Here's an example:

```
labels:
  app: filecontents
  app.kubernetes.io/component: filecontents
  app.kubernetes.io/component: filecontents
  editedby: donschenck
```

Save the file and exit.

Check your work using this command:

```
oc get deploy/deploymentnamegoeshere -o jsonpath='{.metadata.labels}'
```

Here's an example, where you can see the editedby: donschenck label is, in fact, now added:

```
PS C:\> oc get deploy/filecontents -o jsonpath='{.metadata.labels}'
{"app":"filecontents","app.kubernetes.io/component":"filecontents","app.kubernetes.io/instance":
"filecontents","editedby":"donschenck","sandbox":"filecontents"}
```

Enter debug mode

Sometimes, especially when developing and debugging, it's valuable to be able to open a command line inside a pod before the code starts running in OpenShift. The `oc debug` command enables this.

Using the name of an existing pod, run the following command:

```
oc debug pod/yourpodnamegoeshere
```

The command line that will appear is inside the pod, but the code is not executing. You can move about and view file contents. Use the `exit` command to leave the pod's command line and return to your PC's command line.

Run a command-line infinite curl loop

Sometimes, especially when developing and debugging, being able to run a continuous `curl` command loop against a URI running in OpenShift is valuable.

If using **PowerShell**, use the following command with your own URL:

```
while ($true) {curl http://qotd-rhn-engineering-dsch-dev.apps.sandbox-m3.1530.p1.openshiftapps.com/quotes/random;sleep 1;}
```

If using **Bash**, use the following command with your own URL:

```
while true; do curl http://qotd-rhn-engineering-dsch-dev.apps.sandbox-m3.1530.p1.openshiftapps.com/quotes/random;
sleep 1; done
```

You can adjust the delay between `curl` commands by adjusting the sleep value.

Copy a file to a pod

It is often necessary to add files to a pod. Whether using ephemeral or persistent storage, the method of copying a file is the same. Using your own file name and pod name, run the following command:

```
oc cp .\myowndata.txt filecontents-58b86d6c87-lnw7j:/tmp/myowndata.txt
```

You can now open a terminal session inside your pod to see that the file exists.

See all objects with a specific label

Using your own label, run the following command:

```
oc get all -l label_tag=label_value
```

Example:

```
C:\> oc get all -l editedby=donschenck
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/filecontents   1/1     1            1           11d
```

Delete all objects with a specific label

Using your own label, run the following command:

```
oc delete all -l label_tag=label_value
```

Example:

```
PS C:\Users\dschenck\src> oc delete all -l editedby=donschenck
deployment.apps “filecontents” deleted
```