Openshift

Openshift cheatsheet

Here is a comprehensive **Openshift Container Platform cheatsheet** for Developers/Administrators.

Openshift Container Platform Login and Configuration

```
#login with a user
oc login https://192.168.99.100:8443 -u developer -p developer

#login as system admin
oc login -u system:admin

#User Information
oc whoami

#View your configuration
oc config view

#Update the current context to have users login to the desired namespace:
oc config set-context `oc config current-context` --namespace=<project_name>
```

Openshift Container Platform Basic Commands

```
#Use specific template
oc new-app https://github.com/name/project --template=<template>

#New app from a different branch
oc new-app --name=html-dev nginx:1.10~https://github.com/joe-speedboat/openshift.html.dev
ops.git#mybranch

#Create objects from a file:
oc create -f myobject.yaml -n myproject
```

```
#Delete objects contained in a file:
oc delete -f myobject.yaml -n myproject
```

```
#Create or merge objects from file
oc apply -f myobject.yaml -n myproject

#Update existing object
oc patch svc mysvc --type merge --patch '{"spec":{"ports":[{"port": 8080, "targetPort": 5
000 }]}}'

#Monitor Pod status
watch oc get pods

#Show labels
oc get pods --show-labels

#Gather information on a project's pod deployment with node information
oc get pods -o wide

#Hide inactive Pods
oc get pods --show-all=false

#Display all resources
oc get all,secret,configmap

#Get the Openshift Console Address
oc get -n openshift-console route console

#Get the Pod name from the Selector and rsh in it
POD=$(oc get pods -l app=myapp -o name) oc rsh -n $POD

#Exec single command in pod
oc exec $POD $COMMAND

#Copy file from myrunning-pod-2 path in the current location
oc rsync myrunning-pod-2:/tmp/LogginData_20180717220510.json .

#Read resource schema doc oc explain dc
```

Openshift Container Platform Image Streams

```
#List available IS for openshift project
oc get is -n openshift

#Import an image from an external registry
oc import-image --from=registry.access.redhat.com/jboss-amq-6/amq62-openshift -n openshif
t jboss-amq-62:1.3 --confirm

#List available IS and templates
oc new-app --list
```

Openshift Container Platform Templates

```
# Deploy resources contained in a template
oc process -f template.yaml | oc create -f -

#List parameters available in a template
oc process --parameters -f .template.yaml
```

WildFly application example on Openshift Container Platform

```
oc create -f https://raw.githubusercontent.com/wildfly/wildfly-s2i/wf-18.0/imagestreams/wildfly-centos7.json
oc new-app wildfly~https://github.com/fmarchioni/ocpdemos --context-dir=wildfly-basic --name=wildfly-basic
oc expose svc/wildfly-basic
```

Create app from a Project with Dockerfile

```
oc new-build --binary --name=mywildfly -l app=mywildfly

oc patch bc/mywildfly -p '{"spec":{"strategy":{"dockerStrategy":{"dockerfilePath":"Dockerfile"}}}}'

oc start-build mywildfly --from-dir=. --follow

oc new-app --image-stream=mywildfly

oc expose svc/mywildfly
```

Openshift Container Platform Nodes

```
#Get Nodes lits
oc get nodes

#Check on which Node your Pods are running
oc get pods -o wide

#Schedule an application to run on another Node
oc patch dc  myapp -p '{"spec":{"template":{"spec":{"nodeSelector":{"kubernetes.io/hostname

#List all pods which are running on a Node
oc adm manage-node node1.local --list-pods

#Add a label to a Node
oc label node node1.local mylabel=myvalue

#Remove a label from a Node
oc label node node1.local mylabel-
```

Openshift Container Platform Storage

```
#create a PersistentVolumeClaim (+update the DeploymentConfig to include a PV + update th
e DeploymentConfig to attach a volumemount into the specified mount-path)

oc set volume dc/file-uploader --add --name=my-shared-storage \
-t pvc --claim-mode=ReadWriteMany --claim-size=1Gi \
--claim-name=my-shared-storage --claim-class=ocs-storagecluster-cephfs \
--mount-path=/opt/app-root/src/uploaded \
-n my-shared-storage

#List storage classes
oc -n openshift-storage get sc
```

Openshift Container Platform Build

```
#Manual build from source
oc start-build ruby-ex

#Stop a build that is in progress
oc cancel-build <build_name>

#Changing the log level of a build:
oc set env bc/my-build-name BUILD_LOGLEVEL=[1-5]
```

Openshift Container Platform Deployment

```
#Manual deployment
$ oc rollout latest ruby-ex

#Pause automatic deployment rollout
oc rollout pause dc $DEPLOYMENT

# Resume automatic deployment rollout
oc rollout resume dc $DEPLOYMENT

#Define resource requests and limits in DeploymentConfig
oc set resources deployment nginx --limits=cpu=200m,memory=512Mi --requests=cpu=100m,memo
ry=256Mi

#Define livenessProve and readinessProve in DeploymentConfig
oc set probe dc/nginx --readiness --get-url=http://:8080/healthz --initial-delay-seconds=
10
oc set probe dc/nginx --liveness --get-url=http://:8080/healthz --initial-delay-seconds=1
0

#Define Horizontal Pod Autoscaler (hpa)
oc autoscale dc $DC_NAME --max=4 --cpu-percent=10
```

Openshift Container Platform Routes

```
#Create route
 $ oc expose service ruby-ex

#Read the Route Host attribute
oc get route my-route -o jsonpath --template="{.spec.host}"
```

Openshift Container Platform Services

```
#Make a service idle. When the service is next accessed will automatically boot up the po
ds again:
$ oc idle ruby-ex

#Read a Service IP
oc get services rook-ceph-mon-a --template='{{.spec.clusterIP}}'
```

Clean up resources

```
#Delete all resources
oc delete all --all

#Delete resources for one specific app
$ oc delete services -l app=ruby-ex
$ oc delete all -l app=ruby-ex

#CleanUp old docker images on nodes
#Keeping up to three tag revisions 1, and keeping resources (images, image streams and po
ds) younger than sixty minutes:
oc adm prune images --keep-tag-revisions=3 --keep-younger-than=60m

#Pruning every image that exceeds defined limits:
oc adm prune images --prune-over-size-limit
```

Openshift Container Platform Troubleshooting

```
#Check status of current project
oc status

#Get events for a project
oc get events --sort-by='{.lastTimestamp}'

# get the logs of the myrunning-pod-2-fdthn pod
oc logs myrunning-pod-2-fdthn<br />
# follow the logs of the myrunning-pod-2-fdthn pod
oc logs -f myrunning-pod-2-fdthn<br />
# tail the logs of the myrunning-pod-2-fdthn pod
oc logs myrunning-pod-2-fdthn --tail=50

#Check the integrated Docker registry logs:
oc logs docker-registry-n-{xxxxx} -n default | less

#run cluster diagnostics
oc adm diagnostics
```

Openshift Container Platform Security

```
#Create a secret from the CLI and mount it as a volume to a deployment config:
oc create secret generic oia-secret --from-literal=username=myuser
 --from-literal=password=mypassword
oc set volumes dc/myapp --add --name=secret-volume --mount-path=/opt/app-root/
 --secret-name=oia-secret
```

Openshift Container Platform Manage user roles

```
oc adm policy add-role-to-user admin oia -n python
oc adm policy add-cluster-role-to-user cluster-reader system:serviceaccount:monitoring:de
fault
oc adm policy add-scc-to-user anyuid -z default
```

Misc commands

```
#Manage node state
oc adm manage node <node> --schedulable=false

#List installed operators
oc get csv

#Export in a template the IS, BC, DC and SVC
oc export is,bc,dc,svc --as-template=app.yaml

#Show user in prompt
function ps1(){
   export PS1='[\u@\h($(oc whoami -c 2>/dev/null|cut -d/ -f3,1)) \W]\$ '
}

#backup openshift objects

oc get all --all-namespaces --no-headers=true | awk '{print $1","$2}' | while read obj
do
  NS=$(echo $obj | cut -d, -f1)
  OBJ=$(echo $obj | cut -d, -f2)
  FILE=$(echo $obj | sed 's/\//-/g;s/,/-/g')
  echo $NS $OBJ $FILE; oc export -n $NS $OBJ -o yaml > $FILE.yml
done
```