Openshift

Introduction to the Side Car

Laurent Valeyre

Orange

Auguste 2018

Apache status

Module to enable the output statistic of *Apache*.

```
<Location /server-status>
SetHandler server-status
Order deny,allow
Allow from all
</Location> ExtendedStatus On>
```

Figure: status.conf

This module will be copied in the */etc/apache2/mods-enabled/* directory.

Dockerfile

The *Dockerfile* include the copy of the *Apache* module.
Important to add the switching between *root* and *1001* user

```
FROM ubuntu:latest
USER root
...
RUN a2enmod status
COPY status.conf /etc/apache2/mods-enabled/
EXPOSE 8080
USER 1001
CMD ["/usr/sbin/apache2ctl", "-DFOREGROUND"]
```

Figure: Dockerfile

Secret Access

And because the credential of *GITLAB*

```
apiVersion: v1
kind: Secret
metadata:
  name: github-secret
  namespace: sidecar
type: kubernetes.io/basic-auth
data:
  username: c3Bpa2U=
  password: dmFsZW50aW5l
```

Figure: gitlab-secret.yaml

Secret Access

The *username* and *password* are coded with this method and we load the new *secret*

```
$ echo -n 'spike' | base64
c3Bpa2U=
$ echo -n 'valentine' | base64
dmFsZW50aW5l
$ oc create -f gitlab-secret.yaml
```

New Project

It's time to create our new project *sidecar*, similar to a namespace

```
$ oc new-project sidecar \
--display-name='Side Car Project' \
--description='Side Car Project'
```

New Build

We build our new image *faye*, linked to the new secret and we restart the build process

```
$ oc new-build http://192.168.0.8:8880/spike/faye.git \
--name faye
$ oc set build-secret --source bc/faye github-secret
$ oc start-build faye
```

New Build more friendly

A other solution concists to build the new image with a same command line

```
$ oc new-build http://192.168.0.8:8880/spike/faye.git \
--source-secret github-secret \
--name faye
```

New Application

It's time to create our application based on the new image *faye*

```
$ oc new-app faye \
--name fayeapp
$ oc status
$ oc expose service faye
$ oc get pod
$ oc get all name --selector app=cdnselect
```

Export

We export the new application to have a base for the next process
The final application will be based on the export.

```
$ oc get --export is,bc,dc,svc -o yaml > export.yaml
```

Item To Modify

4 parts will be modified to adapted to our application

- ImageStream
- BuildConfig
- DeploymentConfig
- Service

ImageStream

We delete resourceVersion, selfLink and uid. In status, we keep dockerImageRepository (set to "")

BuildConfig

We delete resourceVersion, selfLink and uid. We delete in spec.triggers.imageChange lastTriggeredImageID

BuildConfig

We replace spec.template.spec.containers.image by faye in the first container
We add in spec.template.spec.container

```
- name: apache-exporter
image: previousnext/apache-exporter
command: [ "apache_exporter", \
"-scrape_uri", \
"http://127.0.0.1:8080/server-status/?auto" ]
ports:
- containerPort: 9117
```

Service

We add in spec.ports

```
- name: 9117-tcp
port: 9117
protocol: TCP
targetPort: 9117
```

Finally

We finally create our new application from this *yaml* file

```
$ oc create -f export.yaml
```

Et voila...