Joget Workflow on OpenShift

Red Hat OpenShift is a container application platform that brings Docker and Kubernetes to the enterprise. As a cloud native computing platform, OpenShift allows teams to automate the build, deployment, and management of applications. They can focus on developing apps without worrying about the underlying infrastructure, whether on physical/virtual servers, or on public/ private/hybrid cloud environments.

As an open source no-code/low-code platform to visually build enterprise web apps for coders and non-coders, Joget is an ideal complement to OpenShift to fill the gap. The Joget platform is now available as a Red Hat Certified Container, so it is trusted, secure and commercially supported on OpenShift. This article describes the steps in deploying the certified Joget container image running with the MySQL database.

blocked URL blocked URL blocked URL

Deploy Joget using the OpenShift Web Console

Step 1: Create OpenShift Project

Access the OpenShift Web Console and login

blocked URL

Create a project using the **Create Project** button and key in the desired **Name**, **Display Name** and **Description**.

blocked URL

Step 2: Deploy MySQL Database

Under the selected project, select **Add to Project** > **Browse Catalog** and select the **MySQL** image.

blocked URL

Key in the appropriate settings and click on **Create** e.g.

Namespace	openshift
Database Service Name	jogetdb
MySQL Connection Username	joget
MySQL Connection Password	joget
MySQL Database Name	jwdb

blocked URL

Step 3: Deploy Joget Platform

To access the Red Hat Container Catalog, a valid username and password that is used to log in to the **Red Hat Customer Portal** is required.

If you do not have an account, you can acquire one by registering for one of the following options:

- Red Hat Developer Program. This account gives you access to developer tools and programs.
- 30-day Trial Subscription. This account gives you a 30-day trial subscription with access to select Red Hat software products.

Under the selected project, select **Add to Project** > **Deploy Image** and click on the create an image pull secret link. In the ensuing popup, key in the Red Hat login details for the registry.

Secret Name	registry.connect.redhat.com

Authentication Type	Image Registry Credentials
Image Registry Server Address	registry.connect.redhat.com
Username	Red Hat account username
Password	Red Hat account password
Email	Red Hat account email
Link secret to a service account	Yes
Service Account	default

blocked URL

Once the secret has been created, select **Add to Project** > **Deploy Image**, select the **Image Name** option and key in registry.connect.redhat.com/joget/joget-v6-enterprise. Click on the **small search icon** to load the image details from the registry, key in the desired **Name** then click **Deploy**.

Image Name	registry.connect.redhat.com/joget/joget-v6-enterprise
Name	joget-v6-enterprise

blocked URL

Step 4: Add Persistent Storage

The next step is to add persistent storage to the container for storing configuration files and persistent file uploads. Under **Applications** > **Deployments**, select the application, then the **Configuration** tab. Scroll down to the **Volumes** section and click on the **Add Storage** link.

blocked URL

In the **Add storage** page, click on the small **create storage** link. Create the new storage with the desired values e.g.

Name	joget-data
Access Mode	Shared Access (RWX)
Size	1GB (or as required)

blocked URL

Back in the **Add Storage** page, select the newly created storage, set the **Mount Path** to **/opt/joget/wflow**, then **Add**.

Mount Path	/opt/joget/wflow

blocked URL

Step 5: Create Service Route for External Access

At this point, the service is not accessible from an external web browser so you will need to create a route, which exposes a service at a host name.

Click on **Applications** > **Services** and select joget-v6-enterprise.

blocked URL

Select **Actions** > **Create Route**, set the **Path** to **/jw** and click on **Create**.

Path	/jw

blocked URL

Return to the **Overview** and click on the URL next to the application name to launch the Joget platform.

blocked URL

Deploy Joget using the OpenShift CLI

For those more inclined to use a non-graphical command line interface, it is actually faster to use the OpenShift command line interface (CLI) to deploy the Joget platform.

The following is a Linux script to accomplish a similar Joget platform deployment. Before running this script, install the CLI and login. Change the environment variables in the script accordingly, at least the four values below:

PROJECT_NAME	The desired project name
REGISTRY_USERNAME	Red Hat account username
REGISTRY_PASSWORD	Red Hat account password
REGISTRY_EMAIL	Red Hat account email

```
    #!/bin/sh

export PROJECT_NAME=joget-openshift
export REGISTRY_USERNAME=email@domain
export REGISTRY_PASSWORD=password
export REGISTRY_EMAIL=email@domain
export REGISTRY_SERVER=registry.connect.redhat.com
export IMAGE_NAMESPACE=joget
export IMAGE_NAME=joget-v6-enterprise
export IMAGE_TAG=latest
export APP_NAME=joget-v6-enterprise
export DB_APP_NAME=joget-mysql
export STORAGE_NAME=joget-data
export MYSQL_DATABASE=jwdb
export MYSQL_USER=joget
export MYSQL_PASSWORD=joget

echo === deploy Joget on OpenShift ===
echo PROJECT_NAME: $PROJECT_NAME
echo REGISTRY_SERVER: $REGISTRY_SERVER
echo REGISTRY_USERNAME: $REGISTRY_USERNAME
echo REGISTRY_EMAIL: $REGISTRY_EMAIL
echo IMAGE_NAMESPACE: $IMAGE_NAMESPACE
echo IMAGE_NAME: $IMAGE_NAME
echo IMAGE_TAG $IMAGE_TAG
echo IMAGE_NAME: $IMAGE_NAME
echo APP_NAME: $APP_NAME
echo DB_APP_NAME: $DB_APP_NAME
echo STORAGE_NAME: $STORAGE_NAME
echo MYSQL_DATABASE: $MYSQL_DATABASE
echo MYSQL_USER: $MYSQL_USER
echo MYSQL_PASSWORD: $MYSQL_PASSWORD

echo === create project ===
oc new-project $PROJECT_NAME

echo === deploy MySQL ===
oc new-app openshift/mysql:5.7 --name $DB_APP_NAME -e MYSQL_USER=$MYSQL_USER -e MYSQL_PASSWORD=$MYSQL_PASSWORD -
e MYSQL_DATABASE=$MYSQL_DATABASE

echo === create and bind secret to pull Joget image ===
oc create secret docker-registry $REGISTRY_SERVER --docker-server=$REGISTRY_SERVER --docker-
username=$REGISTRY_USERNAME docker-password=$REGISTRY_PASSWORD --docker-email=$REGISTRY_EMAIL
```

```
oc secrets link default $REGISTRY_SERVER --for=pull

echo === import Joget image ===
oc import-image $PROJECT_NAME/$IMAGE_NAME --from=$REGISTRY_SERVER/$IMAGE_NAMESPACE/$IMAGE_NAME --confirm

#echo === grant pull permission if from different project ===
#oc policy add-role-to-user system:image-puller system:serviceaccount:$PROJECT_NAME:default -n $IMAGE_NAMESPACE

echo === deploy Joget ===
oc new-app $PROJECT_NAME/$IMAGE_NAME:$IMAGE_TAG -e GC_MAX_METASPACE_SIZE=300 --name $APP_NAME

echo === create persistent storage claim ===
cat <<EOF > pvc.yaml
apiVersion: "v1"
kind: "PersistentVolumeClaim"
metadata:
name: "$STORAGE_NAME"
spec:
accessModes:
  - "ReadWriteMany"
resources:
  requests:
    storage: "1Gi"
EOF
oc create -f pvc.yaml
rm pvc.yaml

echo === mount storage ===
oc set volume dc/$APP_NAME --add --type=persistentVolumeClaim --claim-name=$STORAGE_NAME --mount-path=/opt/joget
/wflow

echo === expose service route for external excess ===
oc expose svc $APP_NAME --path=/jw
oc annotate route $APP_NAME --overwrite haproxy.router.openshift.io/timeout=60s
oc get route
echo URL: http://$(oc get route $APP_NAME --template='{{ .spec.host }}')/jw
```

Setup Database

The first time the Joget platform is accessed, the Database Setup page will be displayed. Configure the database settings using the values defined when deploying the MySQL database previously e.g.

Database Host	jogetdb
Database Port	3306
Database Name	jwdb
Database User	joget
Database Password	joget

blocked URL

Upon successful configuration, the Joget App Center will be loaded.

blocked URL

NOTE: If you encounter a **504 Gateway Timeout** during the database setup, it is caused by the database initialization taking longer than the default OpenShift Route timeout. You can actually ignore the error and wait a couple of minutes before accessing the Application URL from the **Overview** page again.

You can also increase the route timeout using the OpenShift CLI i.e.

oc annotate route $APP_NAME --overwrite haproxy.router.openshift.io/timeout=60s