Kubernetes with Containers and DevOps Workshop

Hands-on lab step-by-step

Aralık 2018

## 5.CI/CD Automation using Jenkins Pipeline

**Install Helm and Tiller**

Install Helm

cd /

curl https://raw.githubusercontent.com/helm/helm/master/scripts/get | bash

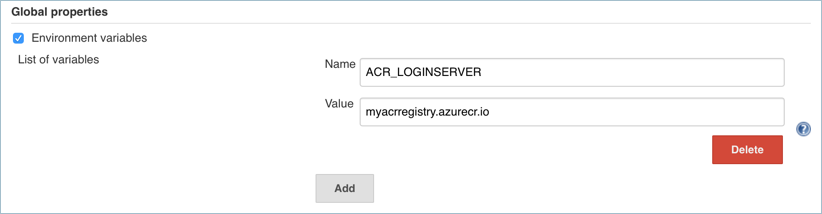
Run helm init to configure helm

helm init

**Create a Jenkins Environment Variable**

A Jenkins environment variable is used to hold the ACR login server name. This variable is referenced during the Jenkins build job. To create this environment variable, complete the following steps:

* On the left-hand side of the Jenkins portal, select Manage Jenkins > Configure System
* Under Global Properties, select Environment variables. Add a variable with the name **ACR\_LOGINSERVER** and the value of **your ACR login server** such as mycontainerregistry.azurecr.io



* When complete, click Save at the bottom of the Jenkins configuration page.

**Create a Jenkins Credential for ACR**

To allow Jenkins to build and then push updated container images to ACR, you need to specify credentials for ACR. This authentication can use Azure Active Directory service principals. In the pre-requisites, you configured the service principal for your AKS cluster with Reader permissions to your ACR registry. These permissions allow the AKS cluster to pull images from the ACR registry. During the CI/CD process, Jenkins builds new container images based on application updates, and needs to then push those images to the ACR registry. For separation of roles and permissions, now configure a service principal for Jenkins with Contributor permissions to your ACR registry.

**Create a Service Principal for Jenkins to use ACR**

* First, create SP:

az ad sp create-for-rbac --skip-assignment

* Make a note of the appId and password shown in your output. These values are used in following steps to configure the credential resource in Jenkins. (Replace 626dd8ea-042d-4043-a8df-4ef56273670f with your own appId)

APP\_ID=626dd8ea-042d-4043-a8df-4ef56273670f

* Get the resource ID of your ACR registry using the **az acr show** command, and store it as a variable. Do not forget to change ACR name.

ACR\_ID=$(az acr show --resource-group mstrdevops-rg --name mstrdevopsacregistry --query "id" --output tsv)

* Now create a role assignment to assign the service principal Contributor rights to the ACR registry. In the following example, provide your own appId noted before.

az role assignment create --assignee $APP\_ID --role Contributor --scope $ACR\_ID

**Create a Credential Resource in Jenkins for the ACR service principal**

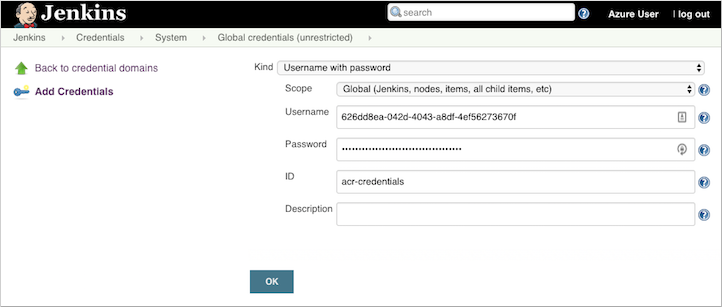
With the role assignment created in Azure, now store your ACR credentials in a Jenkins credential object. These credentials are referenced during the Jenkins build job.

Back on the left-hand side of the Jenkins portal, click Credentials > Jenkins > Global credentials (unrestricted) > Add Credentials

Ensure that the credential kind is Username with password and enter the following items:

* **Username:** The *appId* of the service principal created for authentication with your ACR registry.
* **Password:** The *password* of the service principal created for authentication with your ACR registry.
* **ID:** Credential identifier such as acr-credentials.

When complete, the credentials form looks like the following example:

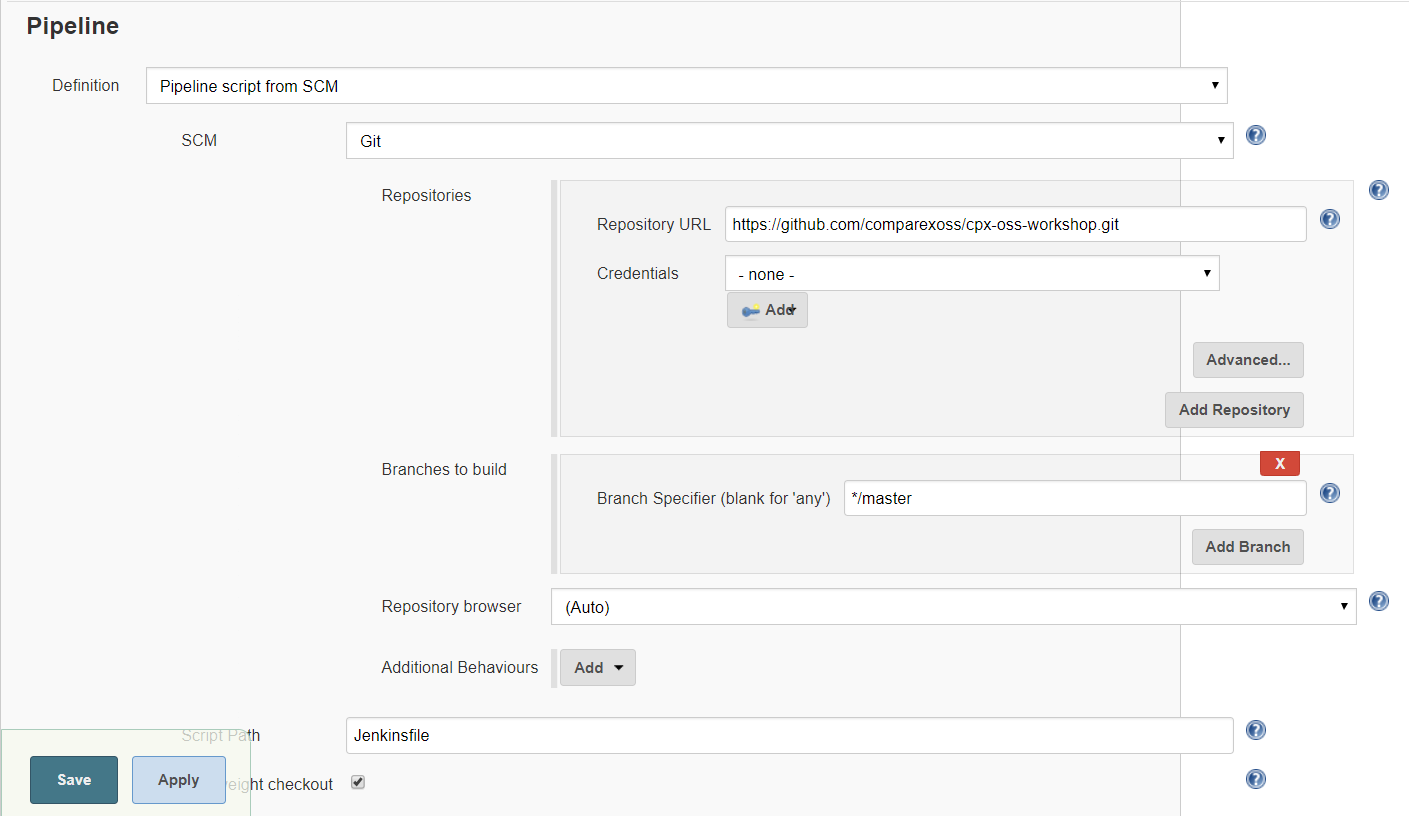


Click OK and return to the Jenkins portal.

**Create a Jenkins Project**

From the home page of your Jenkins portal, select **New item** on the left-hand side:

* Enter *superhero-ratings-pipe* as job name. Choose Pipeline project, then select OK.
* Under the **General** section, select **GitHub** project and enter your forked repo URL, such as <https://github.com/comparexoss/cpx-oss-workshop>
* Check “Do not allow concurrent builds”
* Under the **Build Triggers** section, select **GitHub hook trigger for GITscm polling**.
* Under the **Pipeline** section, select **Pipeline script from SCM** for the defitinion.



* Enter your GitHub project repository url. Do not forget to change github address with your own address.
* Select **Jenkinsfile** under the **Script Path** section.
* Once completed, click Save and build the project that you have created.