Create an Azure Kubernetes Service (AKS) cluster with jx

In the lesson we will discuss how to create a AKS cluster with jx.

We'll cover the following

Creating an Azure Kubernetes Service (AKS) cluster with `jx`

We'll create an AKS cluster with all the tools installed and configured.

Cluster name, resource group, location and nodes

We'll name the cluster with a unique value (`--cluster-name`) and let it reside inside its own group `jxrocks-group` (`--resource-group-name`). It'll run inside the `eastus` location (`--location`) and on `Standard_D2s_v3` (2 CPUs and 8GB RAM) machines (`--node-vm-size`). The number of nodes will be set to three (`--nodes`).

Default admin password

We’ll also set the default Jenkins X password to `admin` (`--default-admin-password`). Otherwise, the process will create a random one.

Default environment prefix

Finally, we’ll set `jx-rocks` as the default environment prefix (`--default-environment-prefix`). A part of the process, we will create a few repositories (one for staging and the other for production), and that prefix will be used to form their names. We won’t go into much detail about those environments and repositories just yet as that is reserved for one of the follow-up chapters.

Feel free to change any of the values in the command that follows to suit your needs better. After all, this is only a practice, and you’ll be able to destroy the cluster and recreate it later with different values.

```
# Please replace [...] with a unique name (e.g., your GitHub user and a day and month).
# Otherwise, it might fail to create a registry.
# The name of the cluster must conform to the following pattern: '^[a-zA-Z0-9]*$'.
CLUSTER_NAME=[...]

jx create cluster aks \
    --cluster-name $CLUSTER_NAME \
    --resource-group-name jxrocks-group \
    --location eastus \
    --node-vm-size Standard_D2s_v3 \
    --nodes 3 \
    --default-admin-password admin \
    --default-environment-prefix jx-rocks
```

What do we get from this command?

Let’s explore what we’re getting with that command. You should be able to correlate my explanation with the console output.

Azure authentication

First, the Azure authentication screen should open asking you to confirm that you are indeed who you claim you are. If that does not happen, please open the link provided in the output manually.

Once we’re authenticated, `jx` will create a cluster. It should take around ten minutes.

Once the AKS cluster is up and running, the process will create a `jx` Namespace. It

Once the AKS cluster is up and running, the process will create a `jx` Namespace. It will also modify your local `kubectl` context.

Jenkins X and other installations

Next, the installation of Jenkins X itself and a few other applications (e.g., ChartMuseum for storing Helm charts) will start. The exact list of apps that will be installed depends on the Kubernetes flavor, the type of setup, and the hosting vendor. But, before it proceeds, it'll ask us a few other questions: What kind do we want to install? Static or serverless? Please answer with `Serverless Jenkins X Pipelines with Tekton`. Even though Jenkins X started its history with Jenkins, the preferred pipeline engine is Tekton, which is available through the serverless flavor of Jenkins X. We'll discuss Tekton and the reasons Jenkins X is using it later. At this point, `jx` will try to deduce your Git name and email. If it fails to do so, it'll ask you for that info.

Ingress installation

The next in line is Ingress. The process will try to find it inside the `kube-system` Namespace and install it if it's not there. The process installs it through a Helm chart. As a result, Ingress will create a load balancer that will provide an entry point into the cluster.

Custom domain name

Once the load balancer is created, `jx` will use its hostname to deduce its IP.

Since we did not specify a custom domain for our cluster, the process will combine that IP with the nip.io service to create a fully qualified domain, and we'll be asked whether we want to proceed using it. Type `y` or merely press the enter key to continue.

Long-term logs

You might be asked *to enable long-term logs storage*, make sure to answer with *n*. We will not need it for our exercises and, at the time of this writing, it is still in the "experimental" phase.

Git and GitHub settings

Next, we'll be asked a few questions related to Git and GitHub. You should be able to answer those. In most cases, all you have to do is confirm the suggested answer by pressing the enter key. As a result, `jx` will store the credentials internally so by pressing the enter key. As a result, `jx` will store the credentials internally so that it can continue interacting with GitHub on our behalf. It will also install the

software necessary for the correct functioning of those environments (Namespaces) inside our cluster.

Organization

We're almost done. Only one question is pending: `Select the organization where you want to create the environment repository?` Choose one from the list.

GitOps

The process will create two GitHub repositories: `environment-jx-rocks-staging` that describes the staging environment and `environment-jx-rocks-production` for production. Those repositories will hold the definitions of those environments. For example, when you decide to promote a release to production, your pipelines will not install anything directly. Instead, they will push changes to `environment-jx-rocks-production` which, in turn, will trigger another job that will comply with the updated definition of the environment.

That's GitOps.

Nothing is done without recording a change in Git. Of course, for that process to work, we need new jobs in Jenkins X, so the process created two that correspond to those repositories. We'll discuss the environments in greater detail later.

`kubectl` context change

Finally, the `kubectl` context was changed to point to the `jx` Namespace, instead of `default`.

We'll get back to the new cluster and the tools that were installed and configured in the What Did We Get? section. Feel free to jump there if you have no interest in how to install Jenkins X inside an existing cluster.