(ED-0017) How-To: Setup Docker Compose & Deploy and Debug Banyan POC

Engineering How-To Document

Versions:			Author:	Valerie Arena		Date:	23 Apr 2020
Product Version (i.e. MH-CURE)	Banyan		Reviewed by:			Reviewed date:	
	XXX		Last revised by:			Revised date:	

Table of Contents

Overview

The purpose of this document is to provide the steps for setting up Docker Compose in IntelliJ. Once Docker Compose is set up, developers can deploy and debug the Banyan POC, which is made up of two Spring Boot applications each running in their own Docker containers.

Step-by-step Guide

Prerequisites

Before following the steps below, the steps listed in (ED-0019) How-To: Setup IntelliJ for Banyan POC must be completed.

Confirm Docker Compose is Installed

Docker Compose is used to deploy multiple docker containers. For Macs, Docker Compose is installed when the Docker Desktop for Mac is installed.

1. Confirm that Docker Compose is installed:

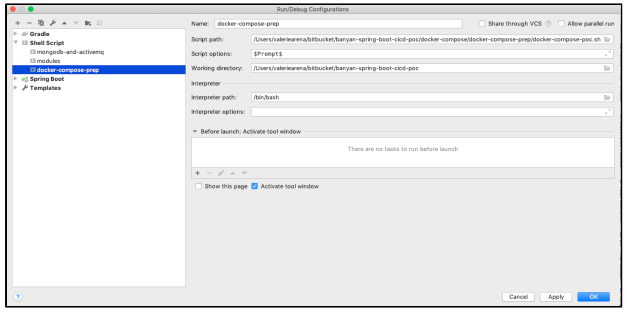
```
docker-compose version
```

Create docker-compose-prep Run Config

The docker-compose-prep run config, will do the following:

- Configure the env file for docker-compose, which will contain the environment variables.
- Automate the process of building docker images for the modules in the project.

1. Go to Run Edit Configurations
2. Click + to Add New Configuration and select Shell Script.
3. Name the run config docker-compose-prep and select docker-compose-poc.sh from /docker-compose/docker-compose-prep/.
4. Add $Prompt$ to Script options.
5. Confirm that your run config looks like the following:

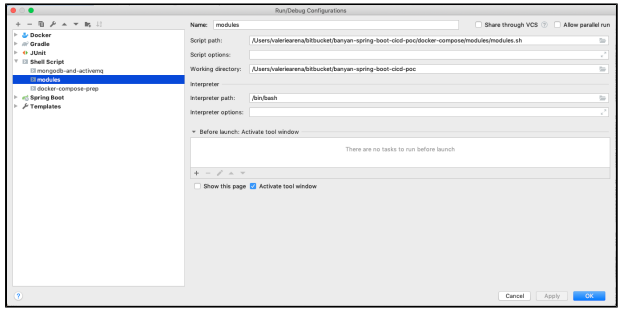


Create modules Run Config

The modules run config will do the following:

- The 'start' option will created and start the containers for MongoDB, ActiveMQ, and the modules in the project.
- The 'stop' option will stop and remove the containers for MongoDB, ActiveMQ, and the modules in the project.
- The 'update' option will update the containers for the modules in the project if any docker images have changed.

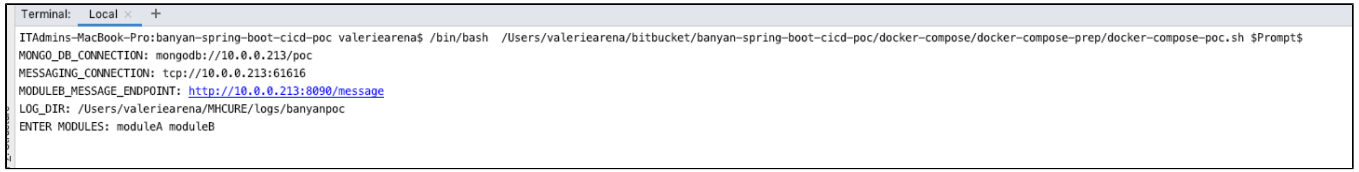
1. Go to Run Edit Configurations
2. Click + to Add New Configuration and select Shell Script.
3. Name the run config modules and select modules.sh from /docker-compose/modules/
4. Confirm that your run config looks like the following:



Set Environment Variables and Build Docker Images

Write environment variables to docker-compose.env and build docker images.

1. Copy /docker-compose/docker-compose-prep/docker-compose.env to your home directory.
2. Select the docker-compose-prep run config and run it.
3. In IntelliJ Terminal (View Tool Windows Terminal), enter the names for 'moduleA' and 'moduleB':

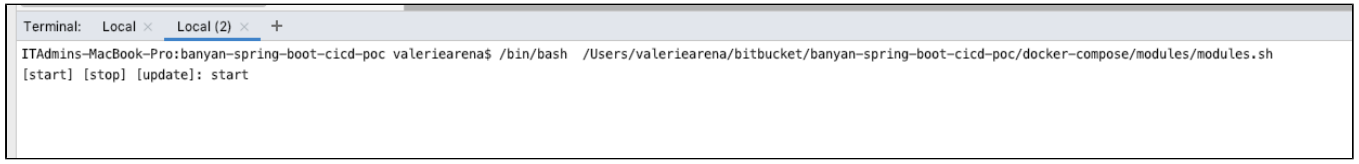


4. Hit Enter. You should see output for building docker images for moduleA and moduleB. Both builds should be successful.

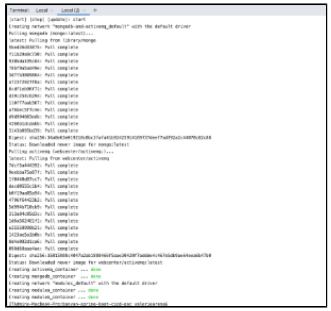
Start Containers

The docker images for MongoDB and ActiveMQ are downloaded in step #3.

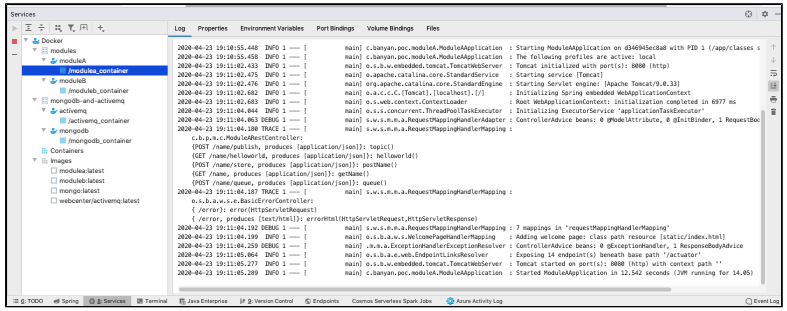
1. Select the modules run config and run it.
2. In IntelliJ Terminal (View Tool Windows Terminal), enter 'start' in Terminal:



3. When starting for the very first time, the docker images for MongoDB and ActiveMQ will be downloaded. When the images have been downloaded and all the containers started, you should see the following:

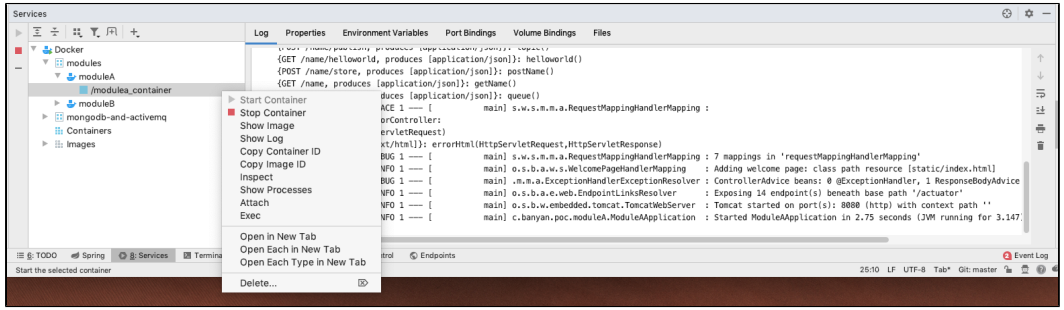


4. In IntelliJ Terminal (View Tool Windows Terminal), connect to Docker. Confirm that you see the following:



Debug Modules

1. In the Services window, select container that you want to debug. Right click and stop the container:



2. Execute the Spring Boot debug config for the module that you want to debug.

3. Add a breakpoint and execute a cURL.

Update Modules

1. Make a code change to moduleA and/or moduleB
2. Select the docker-compose-prep run config and run it.
3. In IntelliJ Terminal (View Tool Windows Terminal), enter the name of the module that you modified. A new docker image will be created containing your code change.
4. Select the modules run config and run it.
5. In IntelliJ Terminal (View Tool Windows Terminal), enter 'update' in Terminal.
6. Confirm that only the container for the image with the code change is recreated.

```
ITAdmins-MacBook-Pro:banyan-spring-boot-cicd-poc valeriearena$ /bin/bash  /Users/valeriearena/bitbucket/banyan-spring-boot-cicd-poc/docker-compose/modules/modules.sh
[start] [stop] [update]: update
Recreating modulea_container ...
Recreating modulea_container ... done
ITAdmins-MacBook-Pro:banyan-spring-boot-cicd-poc valeriearena$
```

Stop Containers

1. Select the modules run config and run it.
2. In IntelliJ Terminal (View Tool Windows Terminal), enter stop in Terminal.
3. Confirm that you see the following:

```
ITAdmins-MacBook-Pro:banyan-spring-boot-cicd-poc valeriearena$ /bin/bash  /Users/valeriearena/bitbucket/banyan-spring-boot-cicd-poc/docker-compose/modules/modules.sh
[start] [stop] [update]: stop
Stopping moduleb_container ... done
Stopping modulea_container ... done
Removing moduleb_container ... done
Removing modulea_container ... done
Removing network modules_default
Stopping mongodb_container  ... done
Stopping activemq_container ... done
Removing mongodb_container  ... done
Removing activemq_container ... done
Removing network mongodb-and-activemq_default
ITAdmins-MacBook-Pro:banyan-spring-boot-cicd-poc valeriearena$
```

4. In IntelliJ Terminal (View Tool Windows Terminal), confirm that the Docker containers have been removed.