Installing Entando App Builder 5.0

Prerequisites

Installed Prerequisites:

- Java 1.8+ or later required
- npm 5.6.0+ (for ReactJS front ends)
- Maven 3.0.5+ (including Maven Central)
- Ant 1.8.0+
- git (to build from source)

Space and Hardware Requirements

Entando is a built to be a very light framework requiring limited runtime resources. A deployment of Entando can have a footprint as small as 58 megabytes. When extending Entando in a development setting, sizing for development environments that add to an Entando install is up to individual user requirements.

Creating a new Entando Project

You can set up an Entando application via a Maven archetype that will create the project and structure needed to develop and add to an application.

Steps:

1. Open your command line and navigate to the directory that will house your new project.
2. Run the following command:
   `mvn archetype:generate -Dfilter=entando-archetype-webapp-generic`
3. Select Option 1, displayed as follows:
   1: remote → org.entando.entando:entando-archetype-webapp-generic (Generic web app Archetype for Entando: an agile, modern and user-centric open source web app like platform.)
4. If prompted for a version, select 5.0.0.
5. Enter values for groupId, artifactId, version, and package. These values go into the Maven setup in the new project.
6. Open the directory created by the Maven archetype in Step 2. The directory will have the same name as the value you entered for artifactId in Step 5. This directory contains all of the structure necessary to start the Entando MApp-Engine and execute the application. See Launching the Application section below for instructions on starting the newly created project.

Launching the Entando Application

Quick Start Using Derby and Jetty

Once you have created an Entando Application, you can launch it using Jetty and an embedded Derby database.

To do so, run the following command inside your project:
`mvn clean jetty:run`

Once started the logs will print the following messages:
[INFO] Started SelectChannelConnector@0.0.0.0:8080
[INFO] Started Jetty Server
[INFO] Starting scanner at interval of 5 seconds

The logs will indicate that your application is running. Follow the steps in Launching the App Builder section to launch the App Builder.

Setting up a Database:

You can configure a newly created Entando application to connect to a database as its backing store. Derby is the default configuration in archetype-created applications, but you can change the default to a different value. Open the Filter Properties file in src/main/filters and enter the appropriate environment and database configuration.

To connect the MApp Engine to a database server:

1. In your database server, create a user for the application.
2. Create two databases. Give the user from Step 1 permission to create, read, and write.
3. Update the appropriate Filter Properties file in src/main/filters to use the configuration for the database properties. For example, on a macOS, you would update + *filter-development-unix.properties*.
4. Set the user, database, and password for the values created in Steps 1 and 2.
5. Launch the application.

NOTE	When launching with the mvn jetty:run command, Jetty Entando will automatically create the table structure required to run the application. This can be used to instantiate an empty database when the target deployment is an app server such as JBoss or Tomcat.

Deploying to Docker using s2i (Source to Image)

Prerequisites:

- Installed and running docker instance

- s2i command installed (https://github.com/openshift/source-to-image)
- An Entando App created using the steps in Creating a New Entando Project can be run directly in Docker using the Entando s2i images. Images are provided for Wildfly 11, JBoss EAP X, and Tomcat.

To use the Docker s2i images, you must have already set up a database. See the Setting up a Database section for details.

By default, the app deployed in Docker will connect to a Postgres database to persist resources created using the App Builder and by the MApp Engine. In the app created from the archetype, update the properties in: **<your application>/.s2i/environment** to point to the user and databases created in the Setting up a Database section.

After configuring the database:

1. Pull in the Docker image using the following command:
   `docker pull entando_wildfly_s2i_5.0.0`
2. Build the image using s2i using the command to build and deploy a Docker app in Docker:
   `s2i build <path or URL of your project> entando_wildfly_s2i_5.0.0 <your image name>`
   Where:
   - **<path or URL of your project>** is the path to your project or a URL to access the code. The build for this project will be invoked and the resulting war file deployed to the app server in the image
   - **Entando_wildfly_s2i_5.0.0** is the name of the base docker image provided by Entando
   - **<your image name>** is the name for this docker image

Using OpenShift

Prerequisite:

- Installed minishift environment

You can configure an Entando app to launch directly into OpenShift using a pre-configured archetype. Create an OpenShift-ready project using the following command:
`mvn archetype:generate -Dfilter=entando-archetype-webapp-openshift`

Launching the App Builder

Build from Source

Prerequisites:

- git
- npm

- node

Clone and set up

Enter the following commands in your command line:

1. `git clone https://github.com/entando/frontend-common-components.git`
2. `cd frontend-common-components.git`
3. `npm install`

NOTE	The npm install command installs npm dependencies

Deploy

Enter the following commands in your command line:

1. `npm run lint`
   Runs the linter. It fails if linting rules are not matched.
2. `npm run coverage`
   Runs unit tests. It fails if a unit test fails, or if the minimum coverage threshold is not met.
3. `npm run import-plugins`
   Compiles and imports Entando plugins.
4. `npm run build`
   Compiles the project and creates the build directory.
5. `npm run build-full`
   Runs npm run lint, npm run coverage, npm run import-plugins and npm run build

Development

`npm start`
Starts the application in dev mode (local web server).

Using Docker

1. Pull in the docker image:
   `docker pull entando/app-builder-5.0.0`
2. Run the image. Example docker command:
   `docker run -it --rm -d -p 5000:5000 -e DOMAIN=http://localhost:8000/my-app appbuilder-5.0.0`
   Where:
   - **DOMAIN**= is the url of a running instance of the MApp Engine. The App Builder uses the REST APIs in the engine to create and manage the application

Build the MApp Engine from Source

To download the latest source code:

1. Open your terminal and create an empty directory for your project:
   `mkdir ~/my_new_project`
2. Move to the new directory
   cd ~/my_new_project
3. Clone the following repositories IN ORDER: entando-core, entando-components, entando-archetypes, entando-ux-packages projects:
   a. Entando-core:
      `git clone https://github.com/entando/entando-core`
   b. Entando-components:
      `git clone https://github.com/entando/entando-components`
   c. Entando-archetypes:
      `git clone https://github.com/entando/entando-archetypes`
   d. (Optional) Entando-ux-packages:
      `git clone https://github.com/entando/entando-ux-packages`
      The Entando UX Packages repository contains samples of pre-made Entando-based applications.
4. Install, IN ORDER, the entando-core, entando-components, entando-archetypes projects:
   a. cd entando-core
      `mvn clean install -DskipTests`
   b. cd entando-components
      `mvn clean install -DskipTests`
   c. cd entando-archetypes
      `mvn clean install -DskipTests`
5. Complete the download by following the steps from the Creating a New Entando Project section.

NOTE

The command to use the artifacts you have installed locally with an additional switch on the archetype command is:

```
mvn archetype:generate -Dfilter=entando-archetype-webapp-generic
-DarchetypeCatalog=local
```