Developer How-To Launch a Local Instance of the Reference Application

Outdated

As of March 2017, the resources mentioned on this page are outdated/ no longer maintained. The OpenMRS SDK is now the preferred method for launching a local instance of the Reference Application.

Introduction

Developers have the ability to deploy and test changes to any of the Reference Application's modules before pushing those changes to GitHub. The mechanism makes use of Vagrant and Puppet to provision test environments on the developer's workstation with one command.

Requirements

NOTE FOR WINDOWS USERS: Make sure you clone your repository with the git config option core.autocrlf=false to preserve Unix-style line breaks in scripts and properties files.

The following software will need to be installed on your system.

- Vagrant http://downloads.vagrantup.com/tags/v1.2.2
- VirtualBox https://www.virtualbox.org/wiki/Downloads
- The Reference Application's Distro repository needs to be cloned. https://github.com/openmrs/openmrs-distro-referenceapplication.git

Details on virtual machine

WIP

Process to launch an instance with latest code from HEAD

1. Clone / Pull the Reference Application's Distro repository. https://github.com/openmrs/openmrs-distro-referenceapplication.git
2. CD to the base directory of that clone.
3. Run the command `'mvn clean package'` to create a zip file of the war and omods for the Reference Application.
4. Run the command `'vagrant up'`
5. Browse to http://localhost:8080/openmrs to access the application
6. OPTIONAL: When done testing run `'vagrant halt'` to stop the virtual machine.

NOTES

- The first time you run vagrant up the command will attempt to download a virtual machine image that is ~270MB in size. This download will only happen once and the image will be preserved on your system afterwards. If your situation requires you to use a download manager, you can download the box at http://goo.gl/8kWkm. Once downloaded, you have to add the box to your vagrant setup using the following command before 'vagrant up' in the steps above:

  ```
  vagrant box add openMRSBase THE_LOCATION_OF_THE_DOWNLOADED_FILE
  ```

  For example, you could use the following location for Windows:

  ```
  vagrant box add openMRSBase C:/Users/admin/Downloads/UbuntuServer12.04amd64.box
  ```

  Note that we use forward slashes '/' instead of back slashes '/' in the file location.
- The first time you run vagrant up puppet will install necessary software (e.g. java, tomcat7, mysql) on the virtual machine. This process will take some time (~15 mins depending on your network speed), but is also a one time occurrence.
- If you already have tomcat running on your local workstation (or some other service using port 8080) the test application url above will not work, but port 2200 will. If possible shutdown the other process using port 8080.

Process to update an instance with work in progress (aka uncommitted) code changes

1. Run a `'mvn clean install'` under the module to which you want to test changes.
2. CD to the base directory of the Reference Application's Distro module.
3. Run the command `'mvn clean package'`
4. Run the command `'vagrant provision'` (if your test machine is already running) or `'vagrant up'` if your test machine is "powered off."

5. OPTIONAL: When done testing run `'vagrant halt'` to stop the virtual machine.

NOTES

- Assuming you previously ran through the initial setup of a test machine you will not be asked to update your database.
- If you already have tomcat running on your local workstation (or some other service using port 8080) the test application url above will not work, but port 2200 will. If possible shutdown the other process using port 8080.

An explanation of useful vagrant commands.

- `vagrant up` - used to create a new test environment or power up one that has been halted.
- `vagrant halt` - used to power off a test environment.
- `vagrant provision` - used to reapply puppet recipes and redeploy the Reference Application (or changes to it) on an already running test environment.
- `vagrant destroy` - used to remove a test environment entirely from your system including the virtual disks. This is safe to do but will require time for puppet to re-provision things the next time you "vagrant up" a machine.
- `vagrant ssh` - used to make an ssh connection to a running test environment.