Enhance Module Maven Archetype (Design Page)

The outcome of this project can be viewed/used here: Using the Module Maven Archetype

<table>
<thead>
<tr>
<th>Primary mentor</th>
<th>Matthew Blanchette</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup mentor</td>
<td>Ben Wolfe</td>
</tr>
<tr>
<td>Assigned to</td>
<td>Goutham Vasireddi</td>
</tr>
</tbody>
</table>

Background

OpenMRS is built on an extensible architecture. Anyone can create a module to add new functionality. A module is able to add new pages, add database tables, modify data, etc. The problem is that the startup time for developers is sometimes prohibitive. See the Creating Your First Module wiki page for the current process.

We recently switched from using ant to build projects to maven. With maven you can create "wizards" or what they call archetypes. The archetype should ask the developer a few simple questions (eg module name, module id, etc) then create the module folder and file project hierarchy with all the right names.

By the time GSOC starts, there will probably be a very basic maven archetype written. See the TRUNK-1733 ticket for more on that. (TODO: link to wiki page for basic archetype when it is completed)

A great example to look at as far as short startup time is the Atlassian SDK for some of their projects. Its as easy as downloading a zip file from them and running one maven command. This command runs a wizard to walk you through creating the the atlassian plugin + running the atlassian project with that plugin already installed and running. Note that this SDK is an end goal for all of OpenMRS, not for this archetype project. This archetype will potentially be one piece of the larger SDK goal.

Purpose

The purpose of this is to speed up developer startup time when creating modules. Creating a module is currently a laborious process that includes a lot of file renaming and copy/pasting from other module templates. A maven wizard (archetype) that walked a user through creating the different types of modules to create and naming things correctly would help to at least speed up the initial creation time.

Domain Expert(s) / User(s)

- Matthew Blanchette (Maven expert)
- user-a952d (Original author of basic openmrs module maven archetype)
- Jeremy Keiper (user with ideas about what should go in the wizard/archetype)

Required/Preferred Skills

- Java
- XML
- Maven
- Velocity

Design

The archetype should not be IDE dependent. Use the maven command line tools / execution of the wizard/archetype to do the different types of functions.

There is no need for extra OS dependent scripts. The wiki page about this archetype should tell the user the different options/flags/mavenmodules for how to do the different options.

A utopia would be that someone could simply run "mvn module:module" (or some better maven goal and target name) and the archetype wizard asks them what to name the module, their username, etc, etc, if they want a service, if they want an admin link, etc. The module id/user name/etc data goes into the config.xml file. See the current basic archetype for an example of that. The output is a module filestructure with all those things in it.

A bonus would be able to run them separately so that someone could run that on an already-built module to add in a service, eg.

Objectives

The archetype should allow for multiple

1. Wizard/archetype that asks developers for the basic module info:
   a. Module name, module id, module package, module description
   b. See the basicmodule for the barebones filestructure that a module needs.
2. Wizard should ask dev if they want to create a service/serviceimpl/dao/hibernatedao
   a. See the https://github.com/OpenMRS/openmrs-module-example-jsp-and-service for a module that has the service files added (warning: not mavenized)
3. Wizard should ask dev if they want an admin page link and what that link should be named
a. See the extension point in config.xml: https://github.com/OpenMRS/openmrs-module-htmlformentry/tree/master/omod/src/main/resources/config.xml
b. And the class that it points to: https://github.com/OpenMRS/openmrs-module-htmlformentry/tree/master/api/src/main/java/org/openmrs/module/formentry/extension/html/FormEntryAdminExt.java

4. Wizard should ask dev if they want to add some example annotation driven mvc pages
   a. See https://github.com/OpenMRS/openmrs-module-example-spring-mvc-annotation for web controllers that are annotated instead of xml-based (warning: not mavenized)

5. Wiki page describing all aspects of the archetype and how a module developer would use it

6. Wiki page describing how to update the maven archetype if/when the module structure changes or a new wizard option is needed.

Extra Credit

1. Turn this archetype into one part of an "OpenMRS Module SDK" (see paragraph in Background section about Atlassian SDK)
   a. Create a downloadable (or git check-out-able) package that allows the user to walk through the archetype/wizard and then jetty:run openmrs
   b. There will need to be a maven archetype or some other function that is able to install the openmrs module into the newly running openmrs automatically

Project Plan

- Creating a basic archetype.
  - Includes bare bone structure of basic module.
  - Asking basic parameters such as, module name, id, package, description.
- Discussion of above basic structure with mentors.
- Further improvement of the archetype.
- Implementation of archetype and including Add-on's such as Service / ServiceImpl
- DAO / Hibernate DAO
- Links to admin page
- Annotation-driven MVC pages
- Documentation
  - Technical documentation
  - How to update the maven archetype if/when the module structure changes or a new option is needed.
  - User documentation
  - Detailed steps in working with archetype wizard on wiki page.
- Implementation of this archetype to Open-Mrs developers, mentors and fellow interns for testing purpose.
- Gathering the suggestions and improvising the project and wiki description.
- Implementation of above features into an "OpenMRS Module SDK"

Timeline

<table>
<thead>
<tr>
<th>Week</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 23 - May 30</td>
<td>Creation of archetype, which creates a basic module with complete bare bone structure.</td>
</tr>
<tr>
<td>May 31 - June 3</td>
<td>Improvising the archetype on suggestions.</td>
</tr>
<tr>
<td>June 4 - June 10</td>
<td>Implementation of Service and ServiceImpl in archetype.</td>
</tr>
<tr>
<td>June 11 - June 17</td>
<td>Implementation of Dao/hibernateDao in archetype.</td>
</tr>
<tr>
<td>June 18 - June 24</td>
<td>Implementation of admin page link in archetype.</td>
</tr>
<tr>
<td>June 25 - July 1st</td>
<td>Documentation</td>
</tr>
<tr>
<td>July 2th - July 10th</td>
<td>Polishing the code for Midterm evaluations.</td>
</tr>
<tr>
<td>July 11 - July 16th</td>
<td>Implementation of Annotation driven mvc pages.</td>
</tr>
<tr>
<td>July 14</td>
<td>Presentation on the things completed and yet to be completed.</td>
</tr>
<tr>
<td>July 17th - July 24th</td>
<td>Code testing</td>
</tr>
<tr>
<td>July 25th - August 15th</td>
<td>Concentrating on extra credits.</td>
</tr>
<tr>
<td>August 15 - August 22th</td>
<td>Scrub code, write tests, improve documentation and submit code to mentor.</td>
</tr>
<tr>
<td>August 25</td>
<td>Final presentation of project</td>
</tr>
</tbody>
</table>

Resources

- Guide to creating archetypes
- TRUNK-1733
- Atlassian SDK