Adding a Web Service Step by Step Guide for Module Developers

This guide walks a module developer through adding a new set of web service methods for a module object. There is a separate guide for core developers.

Conventions

The first thing to do is to read through the conventions section on the REST Web Services API For Clients page. Your module should follow similar conventions.

**Note:** When creating a REST resource and sub-resource URI and name respectively, the absolute and relative path names should be written in all-lowercase ASCII letters. Avoid capital letters, camel caps to mention a few. The norm is to use lower case letters.

Examples:
- /ws/rest/v1/conceptclass
- /ws/rest/v1/yourmoduleid/yourresourcename

Getting the REST Web Services Jar into your project

See Requiring another module in your module

You will need to check out the webservices.rest project and mvn clean install it to make it available to your module project. You will also need to add a few dependencies in your module's pom.xml files:

**main pom.xml file additions**

```xml
<dependency>
  <groupId>org.openmrs.module</groupId>
  <artifactId>webservices.rest-omod</artifactId>
  <version>2.0</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>org.openmrs.module</groupId>
  <artifactId>webservices.rest-omod-common</artifactId>
  <version>2.0</version>
  <classifier>tests</classifier>
  <scope>test</scope>
</dependency>
```

Adding new Web Service URLs

The documentation on how to create methods is the same as for the core developers. See that page for more information.

The only difference is that you want to expose your objects under /ws/rest/v1/yourmoduleid/yourresourcename and you need to add a controller in:

org.openmrs.module.yourmoduleid.web.controller package

:
@Controller
@RequestMapping("/rest/" + RestConstants.VERSION_1 + "/yourmoduleid")
public class YourModuleIdResourceController extends MainResourceController {

/**
 * @see org.openmrs.module.webservices.rest.web.v1_0.controller.BaseRestController#getNamespace()
 */
@override
public String getNamespace() {
    return "v1/yourmoduleid";
}
}

and name your resources accordingly @Resource(name=RestConstants.VERSION_1 + "/yourmoduleid/yourresourcename"
...

Then you can proceed and add your rest module's resources with org.openmrs.module.yourmoduleid.web.resource package

Examples
To be provided.