Example Maven Project

Example Maven Project

Overview

The SLAC CVS contains a sample Maven project called **ExampleMavenProject**.

Obtaining

To obtain the ExampleMavenProject module, execute this CVS command from a work directory.

```
cvs -d :pserver:anonymous@cvs.freehep.org:/cvs/lcd co ExampleMavenProject
```

This simple project can be used as a template for starting new project or as a platform for experimentation with package imports, scratch Java code, and JAS3 plugins.

Anatomy of the Project

Directories

The directory into which the project was checked-out will be called the "base" directory. By default, it will be called "Example Maven Project".

This directory should contain two subdirectories.

```
src
test
```

The **src** dir is the root area for Java source code.

> ✅ **Your Source Code**
>
> All the code for your project should be placed someplace under **src**.

The **test** dir contains unit tests. LCSim uses the JUnit test framework.

> ✅ **Unit Tests**
>
> Ideally, each of your Java classes should have a unit test.

These directories can be customized or additional ones can be added by modifying the **project.xml** file. (which is beyond the scope of this tutorial)

Maven Build Files

The **ExampleMavenProject** module contains the following Maven files, which should be included in a new Maven project.

```
maven.xml
project.properties
project.xml
```

The **project.xml** file is the primary Maven configuration file that lists all the project's meta-information and its dependencies.

The **project.properties** file is used to set named variables (properties) that determine certain project behavior, such as whether deprecation warnings are shown or the tests are skipped. You can put any custom
maven settings into this file.

For instance, to show Java deprecation messages when compiling, remove the "#" from in front of this line in the **project.properties** file.

```
#maven.compile.deprecation=on
```

maven.xml sets-up some project build defaults.

build.sh is a non-essential helper script demonstrating a project build with a set of maven commands.

Example Code

Two Java files are included within the sample project.

The **src/ExampleMavenProject.java** is a simple example of a Java source file.

The file **test/ExampleMavenProjectTest.java** is a very basic test case example that runs the main of a project class with some dummy arguments.

Building

This is the simplest way to build a Maven project.

```
cd ExampleMavenProject
maven
```

(It doesn't get much easier than that.)

The **maven.xml** file sets the default build target to create a jar in the **target** directory.

```
ExampleMavenProject-1.0.jar
```

The "1.0" version number comes from the **project.xml** file.

ExampleMavenProject also includes a bash build script.

```
cd ExampleMavenProject
./build.sh
```

This executes the following command.

```
maven -Drun.install=$(pwd) -Dmaven.test.skip=true clean jar:install jas:install run:install
```

This command builds the jar file, skipping tests, and installs the run script into the current directory. It also makes the jar available to JAS3.

This would be the step-by-step build procedure (leaving out the skipping of tests).

1. Build the jar file.
   ```
   maven
   ```

2. Create the project run script.
   ```
   maven -Drun.install=$(pwd) run:install
   ```

3. Install the jar into the **~/.JAS3/extensions** directory, so that JAS3 will automatically load a plugin when it starts up.

   ```
   maven jas:install
   ```

Testing

Now, test the run script.

```
./bin/ExampleMavenProject
```

This message should be printed to the screen.

```
ExampleMavenProject - Hello world!
```

> ℹ **Freehep Run Plugin**
>
> The cross-platform script creation is done using the Freehep Run Plugin, which is listed as a dependency in the **project.xml** file.

Java files can be added into **src** or **test**.

Classes placed here can access packages in the org.lcsim framework and all of its dependencies.

Simply rerun the build command given above to compile any classes that you add.

Using as a Template

First, create a directory for your project at the same level as the ExampleMavenProject directory.

```
mkdir MyProject
cd MyProject
```

Then you can copy all the files from **ExampleMavenProject** into your area, excluding CVS files, with a command like this.

```
bash
for f in $(find ../ExampleMavenProject -print | grep -v CVS); do cp $f .; done
```

> ✓ **CVS Repositories**
>
> Contact your system administrator about adding a project to your source code repository.
>
> Or mail Tony Johnson if you want to use the LCD CVS at SLAC.