# Creating a Project that Depends on org.lcsim

**THIS PAGE IS OUT OF DATE.**

The information on this page is no longer accurate, as lcsim-based projects have switched to using maven 2.

While there are currently no equivalent instructions for maven 2, please refer to the **lcsim-contrib** project in the SLAC CVS for an example of an m2 project that depends on lcsim.

## Creating a Maven Project that Depends on org.lcsim

### Overview

This tutorial shows how to create a Maven-based project that depends on org.lcsim, so that your code can seemlessly access this framework and all of its dependencies.

The Maven tool is used to build the org.lcsim software. It automates the low-level details involving dependencies and compiling. For instance, it can automatically download required jar files based on a repository URL. Additionally, it can build a complete project website including source code metrics, cross-reference, and JavaDocs.

**Source Control**

It is always a good idea to store your projects in a source control system, such as CVS. Contact your local system administrator for instructions on setting up a module for your project.

### Directory Setup

Create a directory for your new project and go into it.

```
mkdir ExampleProject
cd ExampleProject
```

Now, create a directory for your source files, including directories for the org.lcsim base package.

```
mkdir -p src/org/lcsim
```

A directory for test cases should also be created.

```
mkdir -p test/org/lcsim
```

### Build Files

The project's root directory should contain three Maven configuration files.

- **project.xml** - main configuration file, listing the project's core information and its dependencies
- **maven.xml** - Maven settings, such as the default build target
- **project.properties** - project properties file, including source repository locations, e.g. freehep.org

These can be obtained from org.lcsim's root directory.

```
cvs -d :pserver:anonymous@cvs.freehep.org:/cvs/lcd co lcsim
cd lcsim
cp project.xml maven.xml project.properties ..
cd ..
rm -rf lcsim
```

Customize the following information in **project.xml** for your project.

```
<artifactId>ExampleProject</artifactId>
<currentVersion>0.1</currentVersion>
<organization>
  <name>Example Organization</name>
  <url>http://www.example.org</url>
</organization>
<description>This is an example Maven project.</description>
<shortDescription>Example Maven project</shortDescription>
<url>http://www.example.org/ExampleProject</url>
<issueTrackingUrl>http://www.example.org/ExampleProject/bugs</issueTrackingUrl>
<repository>
  <connection>scm:cvs:pserver:anonymous@cvs.example.org:/cvs/example:ExampleProject</connection>
</repository>
<name>ExampleProject</name>
<inceptionYear>2005</inceptionYear>
```

After the project information, the following lines should be inserted into **project.xml** to make it depend on org.lcsim, itself.

```
<dependency>
  <groupId>lcsim</groupId>
  <artifactId>lcsim</artifactId>
  <version>0.9</version>
  <url>http://www.lcsim.org</url>
</dependency>
```

**org.lcsim version**

Maven requires a specific version tag for dependencies. This means that the version string needs to updated when a new org.lcsim release is made, or the older version will be used instead.

The org.lcsim JAR is **not** currently maintained as a downloadable dependency. Each project user needs to compile and build this program themselves in order to install it to the local repository.

## Basic Build Command

The project can be built from the command line with this simple command.

```
maven
```

This creates the default JAR file and installs it into the **~/.maven/repository** directory under **lcsim**.

Alternately, you can use the Netbeans IDE to build your Maven-based projects.

## Run Plugin

The FreeHep Run Plugin can generate a run script for your project.

To enable this functionality, insert the following into the **project.xml** file.

```
<dependency>
  <groupId>freehep</groupId>
  <artifactId>freehep-run-plugin</artifactId>
  <version>1.1.1</version>
  <url>http://java.freehep.org/maven/freehep/plugins</url>
  <type>plugin</type>
</dependency>
```

Maven needs to know which class the run script should execute. This goes into the **project.properties** file.

```
maven.jar.mainclass=org.lcsim.example.ExampleMain
```

To build the script, execute the following target.

```
maven -Drun.install=$(pwd) run:install
```

Two run scripts named after your project should now be found in the **bin** directory.

There is a Unix/Linux script

```
bin/ExampleProject
```

and also one for Windows.

```
bin/ExampleProject.bat
```

These scripts will setup the classpath and execute the main function of the specified class.

```
public static void main(String[] args)
```

On Linux, the script can be run from the current directory, as follows.

```
./bin/ExampleProject [args]
```

The command line syntax of the script is completely up to you.

## JAS

JAS3 can be configured to automatically load your project's classes on startup.

This target will copy the project's JAR into JAS3's extensions directory, located at **~/.JAS3/extensions**.

```
maven jas:install
```

The version number will be stripped out of the JAR name, and any existing JAR by the same name will be overwritten.

The class should now be available using the **File -> Load** command within JAS3.

## Build Script

A full build command for your project, incorporating all of the above features, would look something like this.

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

This will do a clean build, skipping tests, and installing the run script to the current directory. It also copies the JAR files into the **~/maven/repository/lcsim** and **~/.JAS3/extensions** directories.