

Getting Started

It is the intention that GLAST Java development should be IDE agnostic, and developers are encouraged to use the IDE of their choice.

- [Coding Guidelines](#)
- [CVS](#)
- [Directory Structure](#)
- [Projects](#)
 - [Data Catalog](#)
 - [Data Catalogue Folder Layout Discussions](#)
 - [Group Manager](#)
 - [org-glast-jobcontrol](#)
 - [LSF Notes](#)
 - [Profile System](#)
 - [Simple User Preferences System for Web Applications](#)
- [Toolset](#)
 - [2.2.2 XDoclet](#)
 - [2.2.4 Front End Libraries](#)
 - [3.2.0 Dreamweaver](#)
 - [3.2.1 Build Tools and Libraries](#)

Projects

This would be a great place to have some introductory documentation on each project. Currently we have

- [org-glast-jobcontrol](#)
- [Data Server Portal](#)
- [Group Manager](#)
- [Profile System](#)
- [Pipeline II](#)

Each project is built with Maven, and therefore has its own automatically generated web site. All of the web site are published here:

<http://glast-ground.slac.stanford.edu/docs/>

CVS

We currently use CVS for our revision control needs for all of our projects, although we have plans to move to [Subversion](#) once JIRA supports Subversion. [View CVS](#) is configured for viewing our CVS repository on the web, and we use [cvssspam](#) to send email notifications for all commits to our projects. We provide an email list you can subscribe to (go to <http://www-glast.stanford.edu/cgi-prot/subscribe.pl> and subscribe to the *Java CVS archive updates* list) to receive these commit notificaitons directly, or you can simply access the [email archive](#).

You must have a SLAC UNIX account to check out the source code from CVS and use SSH with your CVS client (we do not support anonymous pserver access). Here is an example of checking out the org-glast project:

```
$ export CVS_RSH=ssh
$ cvs -d :ext:glast-java.slac.stanford.edu:/cvs/java co org-glast
```

We use [CruiseControl](#) as our build and release manager, which does automatic builds of registered project from our CVS repository. Instructions for registering projects with cruise control can be found here: [Cruise Control](#).



org-glast

The org-glast CVS module is a suggested parent POM for GLAST Maven projects. For more information about how use use Maven, see out [Maven](#)

Toolset

The GRITS CVS module contains all of the third party jars required to build and run the GRITS web application. However, there are still some third party tools required to actually build and run GRITS, such as an installation of the JDK, the Ant build tool and a J2EE container to deploy the GRITS web application to.

The following table outlines the complete GRITS toolset. Each tool is typically composed of several jar files, which often times themselves depend on several other third-party jar files. These dependencies are described in the following sections.

Third Party Tool	Version	Home Page	Required	Included in CVS
Java 2 Platform, Standard Edition	1.4.2 (not 1.5 yet)	http://java.sun.com/j2se/1.4.2/	Yes	No

Apache Ant	1.6.1 or later	http://ant.apache.org/	Yes	No
2.2.2 XDoclet	1.2.1	http://xdoclet.sourceforge.net/	Yes	Yes
Hibernate 2	2.1.4 or later	http://www.hibernate.org/	Yes	Yes
WebWork 2	2.1 or later	http://www.opensymphony.com/webwork/	Yes	Yes
SiteMesh	2.0.1 or later	http://www.opensymphony.com/sitemesh/	Yes	Yes
displaytag	1.0-rc1 or later	http://displaytag.sourceforge.net/	Yes	Yes
JSP Standard Tag Library	1.0 (not 1.1)	http://jakarta.apache.org/taglibs/	Yes	Yes
Eclipse IDE with MyEclipse plugin	3.0	http://www.myeclipseide.com/	No	No
3.2.0 Dreamweaver	7.0.1	http://www.macromedia.com/	No	No

Unable to render {include} The included page could not be found.

XDoclet

XDoclet is used to maintain the many configuration files required by the tools used by GRITS (for example, [Hibernate](#)). The use of XDoclet is important because it keeps all of the information about an object (for example, its database mapping information) in one place where it belongs, namely the Java source file itself.

The XDoclet distribution is very modular and has its functionality distributed across several jar files, which allows a project to use only the subset of functionality it needs. GRITS currently uses the following parts of XDoclet 1.2.1:

```
lib/build/xdoclet-1.2.1.jar
lib/build/xdoclet-hibernate-module-1.2.1.jar
lib/build/xdoclet-web-module-1.2.1.jar
lib/build/xdoclet-webwork-module-1.2.1.jar
lib/build/xdoclet-xdoclet-module-1.2.1.jar
lib/build/xjavadoc-1.0.3.jar
```

Hibernate

GRITS uses Hibernate 2, an Object-Relational Mapping (ORM) framework, for all of its database persistence requirements. Hibernate requires several third party libraries, all of which are included in the GRITS CVS module.

```
lib/runtime/hibernate/cglib-full-2.0.1.jar
lib/runtime/hibernate/dom4j-1.4.jar
lib/runtime/hibernate/ehcache-0.7.jar
lib/runtime/hibernate/hibernate2.jar
lib/runtime/hibernate/odmg-3.0.jar

lib/runtime/commons-collections-2.1.jar
lib/runtime/commons-logging-1.0.3.jar
lib/runtime/log4j-1.2.8.jar
```

Front End Libraries

GRITS uses several third-party libraries to generate the web pages of the application. All of these are built upon the JSP 1.2 (not JSP 1.3) specification.

JSP stands for Java Server Pages, and refers to the fact that a .jsp page is compiled into a Java Servlet by the JSP container. The Java in JSP does not mean that Java code should be embedded in JSP pages. The fact that Java can be embedded in a JSP page according to older specifications of the JSP standard does not mean that you should, and as a matter of fact the ability to embed Java in JSP pages has been turned off by default in later specifications of the JSP standard.

There are much better ways to provide programability and scriptability to the front end. The following table lists the tools that provide this functionality.

Third Party Tool	Version	Home Page	Required	Included in CVS
WebWork2	2.1 or later	http://www.opensymphony.com/webwork/	Yes	Yes
SiteMesh	2.0.1 or later	http://www.opensymphony.com/sitemesh/	Yes	Yes
displaytag	1.0-rc1 or later	http://displaytag.sourceforge.net/	Yes	Yes
JSP Standard Tag Library	1.0 (not 1.1)	http://jakarta.apache.org/taglibs/	Yes	Yes

WebWork 2 is the workhorse of the GRITS front end. It is composed of the following jar files:

```
lib/runtime/webwork/cos-multipart.jar
lib/runtime/webwork/mail.jar
lib/runtime/webwork/ognl-2.6.5.jar
lib/runtime/webwork/oscore-2.2.4.jar
lib/runtime/webwork/pell-multipart.jar
lib/runtime/webwork/velocity-dep-1.3.1.jar
lib/runtime/webwork/webwork-2.1.jar
lib/runtime/webwork/xwork-1.0.1.jar
```

SiteMesh is the layout engine of the entire GRITS application, and allows for all of the web pages to be loosely coupled and created independently of other web pages. All of the SiteMesh functionality is contained in one jar:

```
lib/runtime/webwork/sitemesh-2.0.1.jar
```

The displaytag library is a JSP tag library whose only purpose it to ease the display of tables in JSP.

Unable to render {include} The included page could not be found.

Directory Structure

Upon initial checkout, the GRITS project has the following directory layout. A few hidden files related to the Eclipse IDE have been omitted, as they are irrelevant to the current discussion.

```
build.properties
build.xml
resin.xml
lib/
src/
```

The lib/ directory contains jar files needed for building and running grits. The lib/build/ directory contains all of the jar files required to compile the grits Java source code. It is this collection of jars you'll want to put on the CLASSPATH for whatever IDE you are using.

```
lib/build/*.jar
lib/runtime/hibernate/*.jar
lib/runtime/webwork/*.jar
```

The src/ directory contains all of the source code, including the Java source code, the JSP files, configuration files and deployment descriptors.

```
src/config/db/
src/config/middlegen/
src/java/
src/test/
src/webapp/
```

The Java source code lives in src/java/, while the JUnit tests for this source code lives in src/test/. The web application itself lives in src/webapp and has the standard directory structure for a J2EE container.