

# Debugging

## Debugging

### Quick Start

Jetty has its own builtin logging facade that can log to stderr or slf4j (which in turn can log to commons logging, log4j, nlog4j and java logging).

Jetty logging looks for a slf4j jar on the classpath. If found, slf4j is used to control logging otherwise stderr is used. The org.mortbay.log.Log class is used to coordinate logging and the following system parameters may be used to control logging:

org.mortbay.log.class	Specify an implementation of org.mortbay.log.Logger to use
DEBUG	If set, debug logs will be produced, else only INFO and WARN logs will be generated
VERBOSE	If set, verbose logging is produced, including ignored exceptions
IGNORED	If set (jetty 6.1.10 and later), ignored exceptions are logged (independent of DEBUG and VERBOSE settings)

### With Jetty Standalone

The core of Jetty has no hard dependencies on an external logging infrastructure. The only logging dependency is introduced by [Jasper](#), the JSP engine.

Jasper has traditionally had a dependency on [commons-logging](#), which proves problematic in container-based architectures. To overcome this, Jetty uses 2 different solutions, depending on the version of JSP standard.

**Note** that the start.jar mechanism will *automatically select* a JSP version for you at runtime based on the *jdk version*. As JSP2.1 mandates at least jdk1.5, if you are running with a 1.5 jvm then this will be the version selected. However, if you are running with a lower version jvm, the start.jar mechanism will place the JSP2.0 jars onto the classpath.

### JSP2.0

Jetty uses the [SLF4J](#) logging infrastructure to [bridge](#) to commons-logging for JSP2.0. This means that commons-log messages are sent to the SLF4J interface.

We ship the [Simple](#) log implementation, which will only output INFO level and above messages to stderr.

However, you can replace the Simple log with any other SLF4J log implementation by removing the `lib/jsp-2.0/slf4j-simple-1.0-rc5.jar` and copying in the SLF4J impl of your choice. The core Jetty code has a soft dependency on SLF4J, meaning that if an SLF4J impl is found on the classpath at startup Jetty will direct all logging messages to it.

Alternatively, you can remove the SLF4J jars altogether and use commons-logging instead by copying in the commons-logging jar and a commons-logging compliant log impl, such as log4j, to the `lib/` directory. However, if you do that, be aware that as the core Jetty code does not use commons-logging, it will log messages to `stderr` instead. Read on to learn how to get the `stderr` log mechanism to print DEBUG level messages.

## JSP2.1

With the newest version of the JSP specification, Jetty has removed all logging dependencies from the Jasper code. This means that no external logger is needed and Jetty uses its own `stderr` logging mechanism. By default, that will only output INFO level and above messages.

To enable logging of DEBUG level messages, invoke Jetty with the `-DDEBUG` flag:

```
java -DDEBUG -jar start.jar
```

## With Jetty embedded

As mentioned above, the core of Jetty has no hard external logging dependency. Therefore, if you only include the `lib/jetty-util.jar` and `lib/jetty.jar` on your classpath, Jetty will direct all log messages to `stderr`. To see DEBUG level messages, start your application with the `-DDEBUG` flag:

```
java -DDEBUG .....
```

Or alternatively call `SystemProperty.set("DEBUG", "true")` before calling `new org.mortbay.jetty.Server()`.

If your embedded application includes the jars from either of the JSP versions in `lib/`, then read the previous section for instructions on how to configure logging.

## With the Jetty Maven2 Plugin

As with all Maven plugins, log messages produced by the plugin itself will be sent to `stderr` and only at the INFO level and above. To see DEBUG level messages from the plugin, invoke the plugin with the `-X` flag:

```
mvn -X jetty:run
```

Log messages from the Jetty instance embedded in the plugin are controlled in much the same way as Jetty standalone. The plugin picks an appropriate JSP version based on the version of the JVM executing the plugin. Please see the [Jetty Maven2 Plugin pages](#) for an explanation of how to configure logging.