Intercept

Intercepting Connections and Requests.

Interception is a technique of Aspect Oriented Programming that allows an invocation to be enhanced with pre and post processing. Jetty 6 has been architected to support interception in many ways, plus there are many ways in which behaviour can be extended without interception. This page descibes some of the places in which extended behaviour can be added to Jetty 6.

Connector

Behaviour may be attached to a particular type of connector (eg SelectChanelConnector) by extending the class and adding implementations of one or more of:

- configure(Socket) - Configure a socket after it is accepted.
- customize(Endpoint,Request) - Customize a request for this connector / endpoint.

Server

All connections and requests are channeled via the Server instance, so it may be extended to add interception style behaviour on one or more of the following methods:

- handle(HttpConnection) - called when a thread is allocated to a HttpConnection.
- handle(target,request,response,dispatch)
- called to dispatch a request to the server.

Handler

The Jetty server maintains a tree of Handlers that are mostly implemented as interceptors: Context, Session, Security and RequestLogare all implemented as interceptors based on WrappedHandler. These handlers add their behaviour as an aspect before passing the request down the chain of handlers. Additional behaviour may be added by:

- extending the handle method of existing handlers
- writing new handlers that can be linked into the chain of handlers where required.

Filter

The Filter API is the standard in-context interceptor defined by the servlet specification. These allow portable interception of requests within the servlet spec.