# ► HTTP Whiteboard - OSGI Compendium 6.0

# How webapps should have been!

Raymond Augé <raymond.auge@liferay.com>

### **E** Outline

- Http Whiteboard
- Design Goals
- ServletContextHelper
- Servlets
- **©** Resources
- **ø** Filters

- **© Event Listeners**
- Error Pages
- **o** DTOs
- **o In Action**
- **o** Verdict

# Http Whiteboard

- Started as RFC 189 Http Service Updates
- Evolved into OSGi Enterprise Section 140 Http Whiteboard
   Specification
  - .. an addition which left the original Http Service untouched
- Added whiteboard style programming model

# Design Goals

- Bring support for Servlet 3.0 (minimally)
- Enhancements which were long missing
  - multiple url patterns
  - servlet filters
  - event listeners
  - error pages
- Clarify relationship between contexts in OSGi and Servlet spec
- Introspection of current state of http services
- Define a capability namespace

# ServletContextHelper

#### SampleServletContextHelper.java

```
@Component(
    property = {
         HttpWhiteboardConstants.HTTP_WHITEBOARD_CONTEXT_NAME +
"=default",
         HttpWhiteboardConstants.HTTP_WHITEBOARD_CONTEXT_PATH +
"=/"
     }
)
public class SampleServletContextHelper extends ServletContextHelper {
}
```

## Servlets

#### SampleServlet.java

```
@Component(
    property = {
         HttpWhiteboardConstants.HTTP_WHITEBOARD_SERVLET_PATTERN +
"=/"
    service = Servlet.class
public class SampleServlet extends HttpServlet {
    @Override
    protected void service(
             HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
         PrintWriter writer = response.getWriter();
         writer.write("Hello World!");
```

## Resources

#### SampleResources.java

```
@Component(
    property = {
         HttpWhiteboardConstants.HTTP_WHITEBOARD_RESOURCE_PREFIX +
"=/META-INF/resources",
         HttpWhiteboardConstants.HTTP_WHITEBOARD_RESOURCE_PATTERN
+ "=/resources/*"
     }
)
public class SampleResources {
}
```

## Filters

#### SampleFilter.java

```
@Component(
    property = {
         HttpWhiteboardConstants.HTTP_WHITEBOARD_FILTER_PATTERN +
"=/*"
public class SampleFilter implements Filter {
    @Override
    public void doFilter(
              ServletRequest request, ServletResponse response,
             FilterChain filterChain)
         throws IOException, ServletException {
         filterChain.doFilter(request, response);
```

## **Event Listeners**

SampleServletContextListener.java

```
@Component
public class SampleServletContextListener implements
ServletContextListener {
    @Override
    public void contextDestroyed(ServletContextEvent arg0) {
    @Override
    public void contextInitialized(ServletContextEvent arg0) {
```

# **Error Pages**

SampleErrorPage.java

```
@Component(
    property={
HttpWhiteboardConstants.HTTP WHITEBOARD SERVLET ERROR PAGE +
"=java.io.IOException",
HttpWhiteboardConstants.HTTP_WHITEBOARD_SERVLET_ERROR_PAGE +
=5xx
    service = Servlet.class
public class SampleErrorPage extends HttpServlet {
    @Override
    protected void service(
             HttpServletRequest request, HttpServletResponse response)
        throws IOException, ServletException {
```

```
//
}
}
```

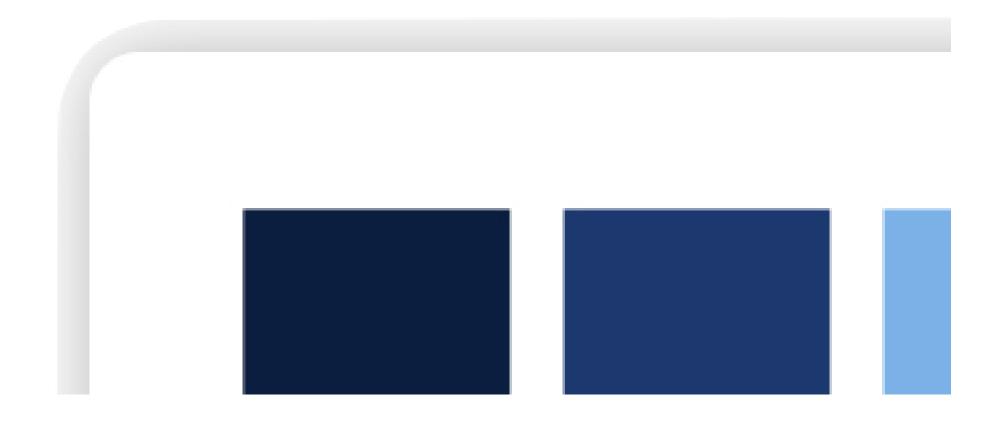
## **O** DTOs

#### SampleDTOReportServlet.java

```
@Component(
    immediate = true,
    property = {
         HttpWhiteboardConstants.HTTP WHITEBOARD SERVLET PATTERN +
"=/dto"
    },
    service = Servlet.class
public class SampleDTOReportServlet extends HttpServlet {
    @Override
    protected void service(
             HttpServletRequest request, HttpServletResponse response)
         throws IOException, ServletException {
         final HttpServiceRuntime httpServiceRuntime = httpServiceRuntime;
         if (httpServiceRuntime == null) {
```

```
response.sendError(
              HttpServletResponse.SC INTERNAL SERVER ERROR,
              "Something went terribly wrong...");
         return;
    }
    response.setContentType("application/json");
    response.setCharacterEncoding("UTF-8");
    PrintWriter writer = response.getWriter();
    writer.write(httpServiceRuntime.getRuntimeDTO().toString());
    writer.close();
@Reference(unbind = "-")
protected void setHttpServiceRuntime(
    HttpServiceRuntime httpServiceRuntime) {
    httpServiceRuntime = httpServiceRuntime;
```

private volatile HttpServiceRuntime \_httpServiceRuntime;





· DEMO TIME!

## Verdict

- I'm excited with the result of the Http Whiteboard specification
- OSGi R6 Early Draft
- Eclipse Project 4.5 M3 New and Noteworthy Friday
- There's very little time left to get your feedback in. But, if you have any, please contact myself or any member of the EEG

## Thank you!