What's new in Http Service

Felix Meschberger, Adobe
Setting the Scope

Http Service Specification

Web Applications
OSGi-style

Web Applications
Specification

Web Applications
Support
Http Service 1.2

- More or less unmodified since OSGi R1 (May, 2000)
  - Resource Registration
  - Custom Authentication

- Based on Servlet API 2.1 (November, 1998)
  - Error Pages (Servlet API 2.2)
  - Filters (Servlet API 2.3)
  - Application Events (Servlet API 2.3)

- Limited Servlet Registration (path prefix only)

- Complex registration:
  - Requires HttpService to register
  - Requires keeping state of successful registration
  - Requires Http Service to unregister
Where will Http Service Go?

- Servlet API Version
- Web Application Descriptor Elements
- Simpler Registration
- Clarify Relationship between HttpSession and ServletContext
- Diagnostic API (based on RFC 185 / Data Transfer Objects)
Requirements 1: Http Service Updates

HS-1 The solution MUST define the relationship between the Http Service and Web Application specifications.

HS-2 The solution MUST update the Http Service specification to refer to the latest Servlet API specification and define to what extent the Http Service provides support.

HS-3 The solution MUST extend the HttpService service API to support Servlet registration with patterns as defined by the Servlet API specification (Section 12.2, Specification of Mappings, in the Servlet API 3.0 specification). This requirement aligns servlet registration to functionality provided by the Servlet API web application descriptor (web.xml).

HS-4 The solution MUST extend the HttpService service API to support registration of Servlet API filters with patterns as defined by the Servlet API specification (Section 12.2, Specification of Mappings, in the Servlet API 3.0 specification) or referring to servlets by their names. This requirement aligns mapping filters to requests to functionality provided by the Servlet API web application descriptor (web.xml).

HS-5 The solution MUST extend the HttpService service API to support registration of Servlet API listeners.

HS-6 The solution MUST add support for error page configuration.

HS-7 The solution MUST define how registered Servlets and Filters are named.

HS-8 The solution MUST clarify ServletContext implementation in the HttpService for both standalone and bridged Http Service implementations.

HS-9 The solution MUST clarify the ServletContext scope of Servlet API listeners registered through the HttpService.

HS-10 The solution MAY specify support for scripted request processing. For example supporting JSP with Tag Libraries.

HS-11 The solution MAY define how HttpService instances can be dynamically configured.

HS-12 The solution MUST define service registration properties for the HttpService to reflect configuration of the service.
Requirements 2: Whiteboard Registration

HS-13 The solution MUST define whiteboard registration of servlet services with the HttpSession.

HS-14 The solution MUST define whiteboard registration of filter services with the HttpSession.

HS-15 The solution MUST define whiteboard registration of servlet listener services with the HttpSession.

HS-16 The solution MUST define registration of OSGi HttpSession services used for Servlet and Filter registration.

HS-17 The solution MUST define how servlets, filters, and servlet listener services are matched with HttpSession services for registration.

HS-18 The solution MUST support registration of static resources according to the extender pattern.

HS-19 The solution MUST support registration of error pages according to the extender pattern.

HS-20 The solution MUST define a capability for the osgi.extender namespace. Bundles providing resources and/or error pages can then require this capability.

HS-21 The solution MUST define a capability for the whiteboard pattern registration in one of the standard namespaces (or a new namespace to be defined in the Chapter 135, Common Namespaces Specification). Bundles registering servlet, filter, and/or servlet listener services can then require this capability.
Which Servlet API to support/require?
- Servlet API 3.0 for the latest and greatest?
- Must not alienate Embedded Systems!

Servlet API and the Platform
- 2.4 requires Java 1.3
- 2.5 requires Java 5
- 3.0 requires Java 6
Web Application Descriptor Elements

Replacements

- EJB: Use Services
- MIME mapping: Use HttpServletRequest
public interface HttpService {

    public void registerServlet(String alias, Servlet servlet, Dictionary initparams, HttpContext context) throws ServletException, NamespaceException;

    public void unregister(String alias);

}
```java
@Component(service = {}, property = "alias=/")
public class SampleServletOld extends HttpServlet {

    private HttpService httpService;
    private String servletAlias;
    private boolean servletRegistered;

    @Activate
    private void activate(Map<String, Object> config) throws ServletException, NamespaceException {
        this.servletAlias = (String) config.get("alias");
        this.httpService.registerServlet(this.servletAlias, this, null, null);
        this.servletRegistered = true;
    }

    @Deactivate
    private void deactivate() {
        if (this.servletRegistered) {
            this.httpService.unregister(servletAlias);
        }
    }

    @Reference(unbind = "unbindHttpService")
    private void bindHttpService(final HttpService httpService) {
        this.httpService = httpService;
    }

    private void unbindHttpService(final HttpService httpService) {
        if (this.httpService == httpService) {
            this.httpService = null;
        }
    }
}
```
@Component(
    service = Servlet.class,
    property = "osgi.http.whiteboard.pattern=/")
public class SampleServletWhiteboard extends HttpServlet {
}

- No Http Service reference
- No State Management
Http Context and Servlet Context

Servlet Context 1
- Attributes
- Resources
- HTTP Sessions

Http Context 1
- Authentication
- MIME Types

Servlet Context n
- Attributes
- Resources
- HTTP Sessions

Http Context n
- Authentication
- MIME Types

Common Servlet Context (e.g. from Servlet Container)
- Context Path
- Logging
- Version
Whiteboard Pattern Support Explained

- Uses Servlet, Filter, HttpContext services
- Registers with Http Service
- Matches Servlet to HttpContext with context.name property
## Http Service Whiteboard Properties

<table>
<thead>
<tr>
<th>osgi.http.whiteboard.</th>
<th>Servlet</th>
<th>HttpContext</th>
<th>Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>.pattern</td>
<td>Registration Pattern</td>
<td>n/a</td>
<td>Registration Pattern</td>
</tr>
<tr>
<td>.context.name</td>
<td>Refers to HttpContext</td>
<td>Identifies</td>
<td>Refers to HttpContext</td>
</tr>
<tr>
<td>.context.shared</td>
<td>n/a</td>
<td>Shared across Bundles ?</td>
<td>n/a</td>
</tr>
<tr>
<td>.errorPage</td>
<td>Status Code Exception Type</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>.service.target</td>
<td>HttpService Selection</td>
<td>n/a</td>
<td>HttpService Selection</td>
</tr>
<tr>
<td>.prefix</td>
<td>Resource Prefix</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>
### Alias vs. Pattern

<table>
<thead>
<tr>
<th>Http Service</th>
<th>Servlet API</th>
<th>Match Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>/alias</td>
<td>/prefix/*</td>
<td>Prefix Match</td>
</tr>
<tr>
<td>*.ext</td>
<td></td>
<td>Extension Match</td>
</tr>
<tr>
<td>&quot;&quot; (empty)</td>
<td></td>
<td>Context Root Match</td>
</tr>
<tr>
<td>/</td>
<td></td>
<td>Default Servlet</td>
</tr>
<tr>
<td>/exact/match</td>
<td></td>
<td>Exact Match</td>
</tr>
</tbody>
</table>

- osgi.http.whiteboard.pattern Service Registration Property
- `String+: String, String[], Collection<String>`
Application Events

- Listeners are Services
  - Listener interface is Service Name
  - Supported Listeners depending on Servlet API Support
- Whiteboard Registration
- All Events from Http Service
- Exception: ServletContextListener
Diagnostic API

- ServletDTO[] getServlets();
- Map<String, String> getResources();
- FilterDTO[] getFilters();
- Map<Object, String> getErrorLocations();
What else?

- **Resources**
  - Replacement for `HttpService.registerResources`
  - `osgi.http.whiteboard.pattern` - String+ of URL patterns
  - `osgi.http.whiteboard.prefix` - String of name prefix

- **Http Service Registration Property**
  - `osgi.http.service.endpoints` - String+ of endpoint URIs, e.g. `http://127.0.0.1:4502`

- **Capability / Namespace**
  - New namespace `osgi.whiteboard`
  - Whiteboard Capability: `osgi.http`

  ```
  Provide-Capability: osgi.whiteboard;
  osgi.whiteboard="osgi.http";
  uses:="javax.servlet";
  version:Version="1.3"
  ```

  ```
  Require-Capability: osgi.whiteboard;
  filter:="(&(osgi.whiteboard=osgi.http)(version>=1.3))"
  ```
Links

- OSGi Early Access Draft 2013-03
- Discuss
  - mailto:osgi-dev@mail.osgi.org
- OSGi Service Gateway Specification Releases 1.0
  - http://www.osgi.org/Download/Release1
- OSGi Enterprise Release 5

Dienstag, 26. März 13
Give Feedback on the Session

1. Sign In: www.eclipsecon.org

2. Select Session Evaluate

3. Vote
   - +1
   - 0
   - -1