Building Server-Side Eclipse based Web applications

Part 1 – Basics

Technical Sessions – Short Talks, 2007-06-27
The presenters

Jochen Hiller
Business Operation Systems, Germany
jo.hiller@googlemail.com

Gunnar Wagenknecht
Truition, Germany
gunnar@wagenknecht.org
Eclipse Committer

Based on materials contributed by Jochen Hiller, Simon Kaegi, Martin Lippert, Gunnar Wagenknecht for EclipseCon 2007 tutorials
What is Server-Side Eclipse (SSE)?

Recognition... that many of the features that have made RCP successful are equally applicable in server-side contexts.

- Standardized component model (OSGi)
- Pervasive extensibility – Extension Registry
- Runtime provisioning

Integration... with existing server-side infrastructure and technologies

- J2EE Application Servers
- Servlets and JSPs
- Application Frameworks
- ...

Jeff McAffer stated at EclipseCon 2007, Equinox BOF (03/06/2007):
Server-Side Eclipse is a concept, a marketing name, to illustrate possible usage scenarios (like RCP).
What is SSE from an OSGi perspective?

• An OSGi implementation (Equinox)
• Different HttpService implementation(s) (Equinox)
  • Embedded Jetty
  • ServletBridge with JavaEE Web Container
• Enhanced Http Support (Equinox)
  • Declarative contribution to HttpService
  • JSP Support
  • Additional adapter helpers
• Extensive Toolset (Eclipse PDE)
  • Development tooling
  • Deployment support
What motivates developers to use SSE?

Web developer
- Java™ EE application
- + component model
- + use 3rd party plugins

RCP developer
- RCP application
- + server support
- + re-use plugins
- + distributed applications

Infrastructure developer
- application framework
- + component model
  - modular
  - flexible
  - dynamic

Server-Side Eclipse
What is the OSGi HttpService?

- HttpService is part of OSGi v4 component specification
- Provides HttpService based on Servlet 2.1 spec
- Key concepts:
  - HttpService
  - Servlets, Resources
  - HttpContext
- Unified URI space
- Focused on small, embedded devices

```java
/* © Copyright OSGi Alliance */
package org.osgi.service.http;

public interface HttpService {
    public HttpContext createDefaultHttpContext();
    public void registerResources(String alias, String name, HttpContext context)
        throws NamespaceException;
    public void registerServlet(String alias, Servlet servlet, Dictionary initparams, HttpContext context)
        throws ServletException, NamespaceException;
    public void unregister(String alias);
}

public interface HttpContext {
    public String getMimeType(String name);
    public URL getResource(String name);
    public boolean handleSecurity(HttpServletRequest request, HttpServletResponse response)
        throws IOException;
}
```
HttpRegistry extension point

- Provides a declarative alternative to using the HttpService directly in code.
  - [httpcontexts] – supports creation of a basic parameterized HttpContext or user-defined HttpContext
  - [servlets] – provides the equivalent symantics of the HttpService.registerServlet(...) call.
  - [resources] – provides the equivalent symantics of the HttpService.registerResource(...) call.

- One difference: registration lifecycle
  - Follows Eclipse extension point approach Æ not dynamic
  - resolve/unresolve vs. start/stop
HttpRegistry extension point – example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?eclipseversion="3.2"?>
<plugin>
  <extension point="org.eclipse.equinox.http.registry.httpcontexts">
    <httpcontext id="root-context">
      <resource-mapping
        path="/WebContent">
      </resource-mapping>
    </httpcontext>
  </extension>
  <extension point="org.eclipse.equinox.http.registry.resources">
    <resource
      alias="/sample"
      base-name="/"
      httpcontextId="root-context"/>
  </extension>
  <extension point="org.eclipse.equinox.http.registry.servlets">
    <servlet
      alias="/sample/hello"
      class="sample.HelloServlet"
      httpcontextId="root-context"/>
    <serviceSelector filter="(http.port=8080)"/>
  </extension>
</plugin>
```

HttpContext maps to resources
Register resources for URL
Register servlets for URL
Filter for specific HttpService
Deployment scenarios of Server-Side Eclipse

- Equinox embedding a HttpService
  - e.g. based on Jetty

- Application Server running an embedded Equinox
  - Bridging aspect is referred to as the Servletbridge
  - Isolation between multiple web applications/Equinox instances

Source: Jeff McAffer, Eclipse Summit Europe, Server-Side Symposium, Oct 12nd 2006
Demo

- Servlet Sample
- Http Service Tracker
- Stop / start bundles
- Service Tracker vs. Http Registry
Different approaches for Server-Side UI‘s

- Java EE standards
  - Based on Java EE technologies (JSP, Struts, …)
  - Proposal for RSP-UI from Infonoia and others
  - See: http://www.eclipse.org/proposals/rsp/

- Rich AJAX Platform
  - Accepted as Eclipse technology project
  - RCP styled development approach
  - See: http://www.eclipse.org/rap/

- Google Web Toolkit
  - Client-centric UI approach
  - See: http://code.google.com/webtoolkit/
Java EE vs. the OSGi HttpService (1)

- Dynamic Registration
  - Replaces web deployment descriptor (e.g. web.xml)
  - Dynamic registration supports register/unregister of resources/servlets
  - Code based: tied to START/STOP lifecycle
    - e.g. during BundleActivator.start()
  - Declaration based: tied to RESOLVED/UNRESOLVED lifecycle
    - e.g. using Equinox HttpRegistry
  - Other options: Declarative services, OSGi-Spring
- URL mapping differences
  - alias roughly equivalent to <url-mapping>
  - No more than one registration per alias (worth planning, use relative URLs)
  - No welcome-file support (Mapping from / to /index.html)
  - No support for extension mappings
Java EE vs. the OSGi HttpService (2)

- HttpContext
  - Maps 1:1 to a ServletContext
  - MIME type retrieval, resource retrieval, authentication
  - What's missing:
    - `getNamedDispatcher`
    - `getResourcePaths`
    - `getInitParameters (*)`
    - "Context Path" (*)

- Servlets and Filters: What's missing:
  - Filter (*), HttpSessionListener's,
    ServletContextListener's (*), ServletRequestListener's

(*) Workarounds available: Wrapper technique.
See org.eclipse.equinox.http.helper[s] in Equinox Incubator (not API yet)
Equinox JSP Support

- Use `{path}/*.jsp` style alias for JSPs.
- JSP lookup consistent with OSGi HttpService resource registration
- Provides JSP support for Help / UA in the Eclipse 3.3 SDK
- Work with RSP-UI to provide further framework integrations
- JSP Tag Library Discovery supported along bundle classpath
- Supported under the Servletbridge…
  - Allows for portable pre-compiled JSPs
- Provided by: `org.eclipse.equinox.jsp.jasper.JspServlet`
  - `public JspServlet(Bundle bundle,
      String bundleResourcePath, String alias)`
- JSP Extension Registry Support provided
What is Rich AJAX Platform (RAP)?

Brings Eclipse RCP based development style to Web 2.0 era

RAP is:
- An Eclipse technology project (proposed by Inoopract and others)
- Java based UI / AJAX development
  - Using RAP Widget Toolkit (RWT), subset of SWT API
  - Based on JavaScript client-side framework (qooxdoo)
- Eclipse Workbench concept extended for the web
  - incl. Session support
- API compliant with SWT, JFace, Workbench, extension point namespaces

See also:
- http://www.eclipse.org/rap/
- Active community, great interest
What are the RAP components?

- qooxdoo: underlying client-side JavaScript framework
- RWT: RAP Widget Toolkit (migrated from W4T), subset of SWT
- JFace, Workbench: RAP based implementations of Eclipse RCP
The Google Web Toolkit (GWT)

- Build AJAX apps in the Java language
- Embed components in existing HTML
- Full desktop app clones
- Apache 2.0 license
- Vibrant community
Deploying as Eclipse Server-Side Bundles

- One GWT module ‡ Two OSGi bundles

- Server-side bundle
  - RPC service implementation
  - Exposed using `org.eclipse.equinox.http.registry.servlets`

- Client-side bundle
  - Compiled GWT code (Java + JavaScript) together with HTML, style sheets, images, etc.
  - Exposed using `org.eclipse.equinox.http.registry.resources`
Demo

- PlanetEclipse – FeedReader demo
- RAP FeedReader
- GWT FeedReader
- Starting / stopping bundles using Knopflerfish console
- Install new bundles
For more information...

EclipseCon 2007 Short tutorials:

Project hub:
http://www.eclipse.org/equinox/server

Newsgroup:
news://news.eclipse.org/eclipse.technology.equinox

Dev Mailing List:
equinox-dev@eclipse.org

Thank-you
Legal Notices

• Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both

• Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

• Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

• Other company, product, or service names may be trademarks or service marks of others.