



J-Fall

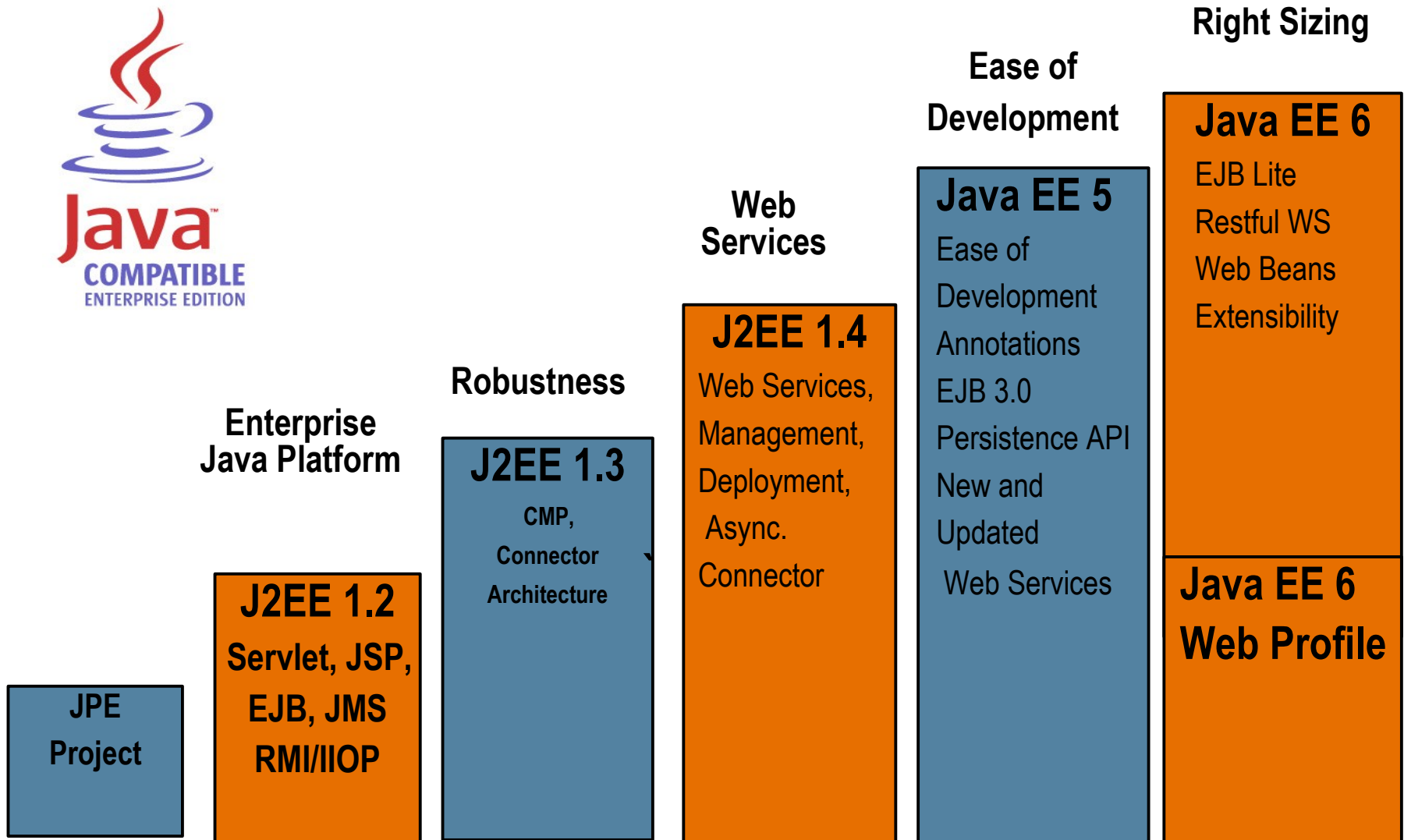
11 november 2009 Spant!



Java EE 6 and GlassFish v3: Paving the path for future

Arun Gupta, GlassFish Guy
Sun Microsystems, Inc.
blogs.sun.com/arungupta

Java EE: Past & Present



Compatible Java EE 5 Implementations



Goals for the Java EE 6 Platform

- Right Sizing the Platform
 - > Flexible, lighter weight
- Extensible
 - > Embrace Open Source Frameworks
- Easier to use, develop on
 - > Continue on path set by Java EE 5

Right Sizing the Platform: Profiles

- Make platform flexible
 - > Decouple specifications to allow more combinations
 - > Expands potential licensee ecosystem
 - > Profiles
 - > Targeted technology bundles
 - > Defined through the JCP
 - > First profile: Web Profile
 - Defined by the Java EE 6 Expert Group

Web Profile

- Fully functional mid-sized profile
 - > Actively discussed in Java EE Expert Group and outside it
 - > Technologies
 - > Servlet 3.0, EJB Lite 3.1, JPA 2.0, JSP 2.2, EL 1.2, JSTL 1.2, JSF 2.0, JTA 1.1, JSR-45, JSR-250

Right Sizing the Platform: Pruning

- Make platform lighter
 - > Makes some technologies optional
 - > Pruned today, means
 - > optional in the next release
 - > Deleted in the subsequent release
 - > Pruned Technologies will be marked in the javadocs
 - > Current pruning list
 - > JAX-RPC, EJB 2.X Entity Beans, JAXR, JSR-88

Extensibility

- Embrace open source libraries and frameworks
- Zero-configuration, drag-and-drop for web frameworks
 - > Servlets, servlet filters, context listeners for a framework get discovered and registered automatically
- Plugin library jars using *web fragments*



```
<web-fragment>
  <filter>
    <filter-name>wicket.helloworld</filter-name>
    <filter-class>org.apache.wicket.protocol.http.WicketFilter</filter-class>
    <init-param>
      <param-name>applicationClassName</param-name>
      <param-value>...</param-value>
    </init-param>
  </filter>

  <filter-mapping>
    <filter-name>wicket.helloworld</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
</web-fragment>
```



```
<web-fragment>
  <filter>
    <filter-name>LiftFilter</filter-name>
    <display-name>Lift Filter</display-name>
    <description>The Filter that intercepts lift calls</description>
    <filter-class>net.liftweb.http.LiftFilter</filter-class>
  </filter>

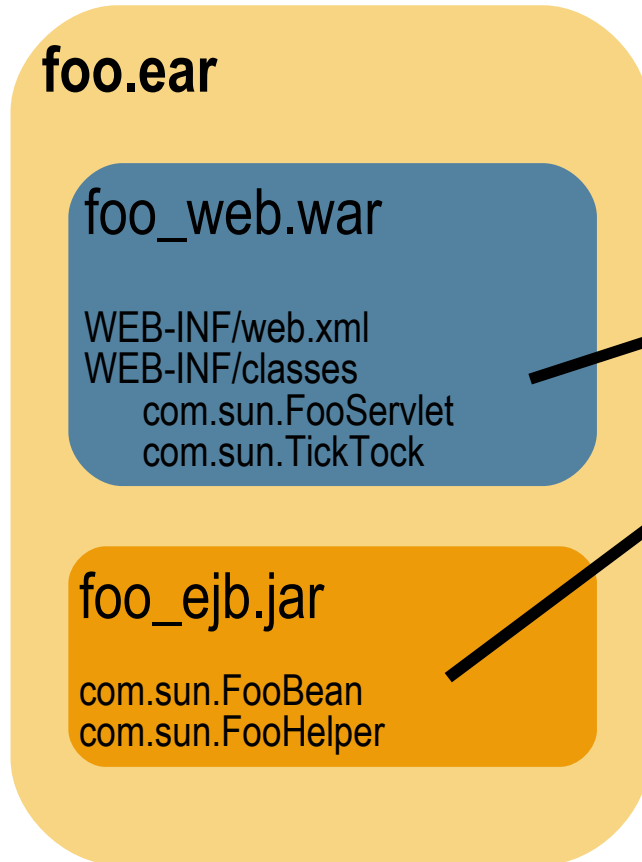
  <filter-mapping>
    <filter-name>LiftFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
</web-fragment>
```

Ease of Development

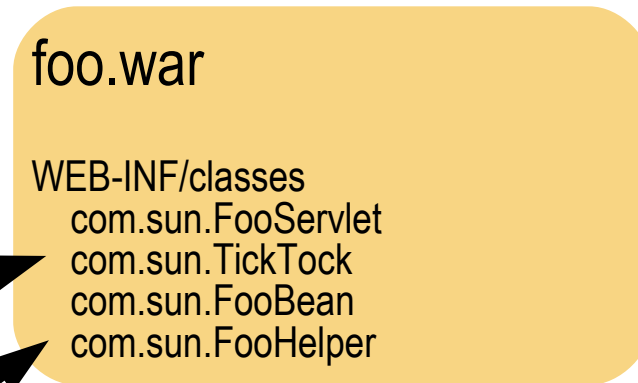
- Continue advancements of Java EE 5
- Primary focus: Web Tier
- Multiple Areas easier to use: EJB 3.1
- General principles
 - > Annotation-based programming model
 - > Reduce or eliminate need for deployment descriptors
 - > Traditional API for advanced users

Java EE 6: Ease of Development (EJB.Lite)

Java EE 5



Java EE 6



~~web.xml ?~~



EoD Example - Servlets

Servlet in Java EE 5: Create two source files

<pre> <!--Deployment descriptor web.xml --> <web-app> <servlet> <servlet-name>MyServlet </servlet-name> <servlet-class> com.foo.MyServlet </servlet-class> </servlet> <servlet-mapping> <servlet-name>MyServlet </servlet-name> <url-pattern>/myApp/* </url-pattern> </servlet-mapping> ... </web-app> </pre>	<pre> /* Code in Java Class */ package com.foo; public class MyServlet extends HttpServlet { public void doGet(HttpServletRequest req,HttpServletResponse res) { ... } ... } </pre>
--	--

EoD Example - Servlets

Servlet in Java EE 6: In many cases a single source file

```
package com.foo;
@WebServlet(name="MyServlet", urlPattern="/myApp/*")
public class MyServlet extends HttpServlet {
    public void doGet(HttpServletRequest req,
        HttpServletResponse res)
    {
        ...
    }
}
```

Java EE 6 Status

- Public reviews completed
- JSF 2.0 is final, majority of the specs are in Proposed Final Draft
- Reference Implementation is GlassFish V3
- Final release later this year – almost there!
- Download weekly builds
 - > <http://download.java.net/glassfish/v3/promoted/>

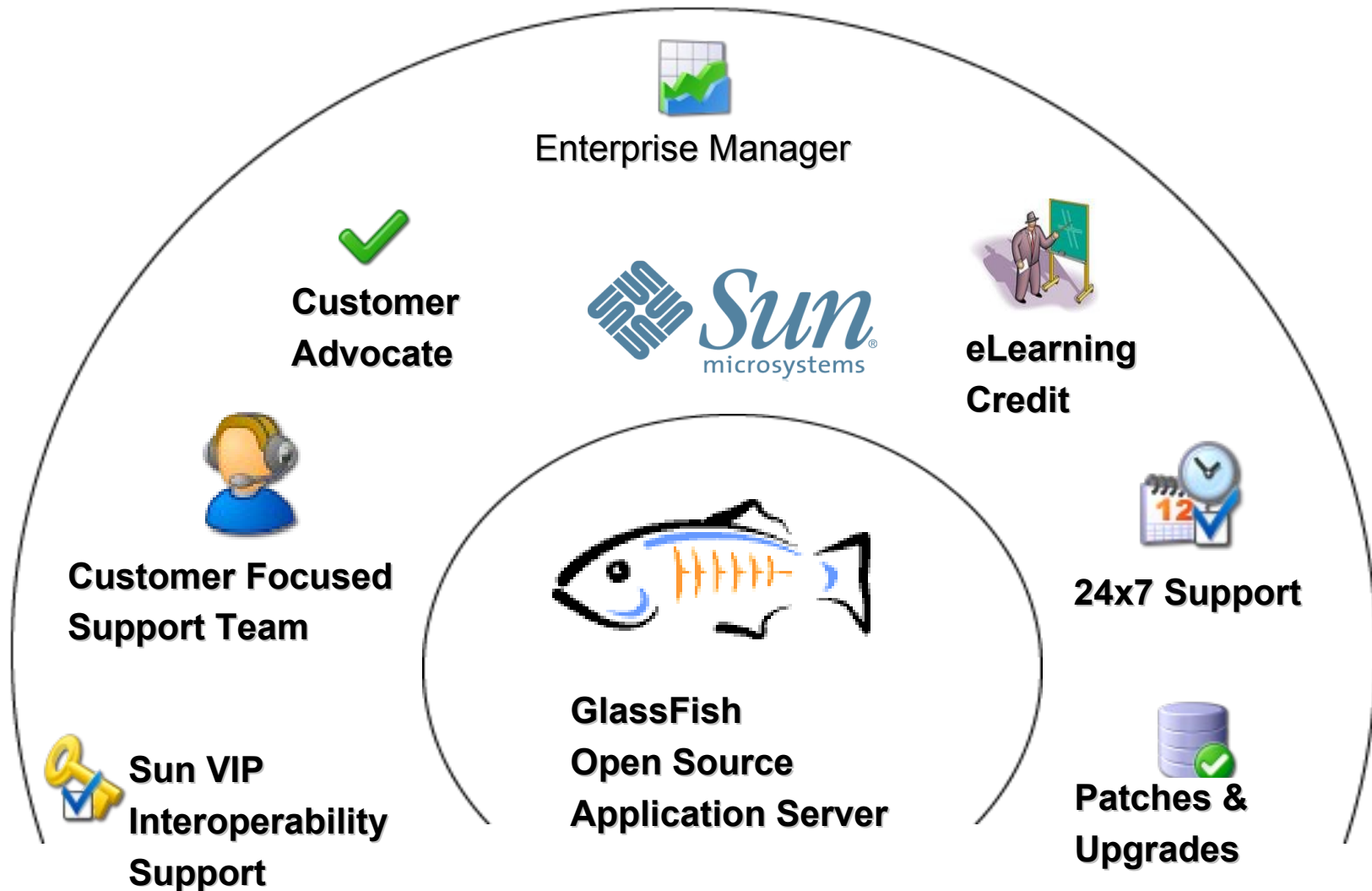
What is GlassFish ?



- A Community
 - > Users, Partners, Testers, Developers, ...
 - > Started in 2005 on java.net
- Application Server
 - > Enterprise Quality and Open Source (CDDL & GPL v2)
 - > Java EE 5/6 Reference Implementation
 - > Full Commercial Support from Sun
- Leverages Sun's experience in other Java, Middleware, SDK

<http://glassfish.org>

Sun GlassFish Enterprise Server



GlassFish v3

- Modular
 - > Maven 2 – Build & Module description
 - > Felix – OSGi runtime
 - > Allow any type of Container to be plugged
 - Start Container and Services on demand
- Embeddable: runs in-VM
- Extensible: pluggable containers
 - > Rails, Grails, Django, ...
- Java EE 6 Reference Implementation
- Support for upcoming Java EE 6 profiles

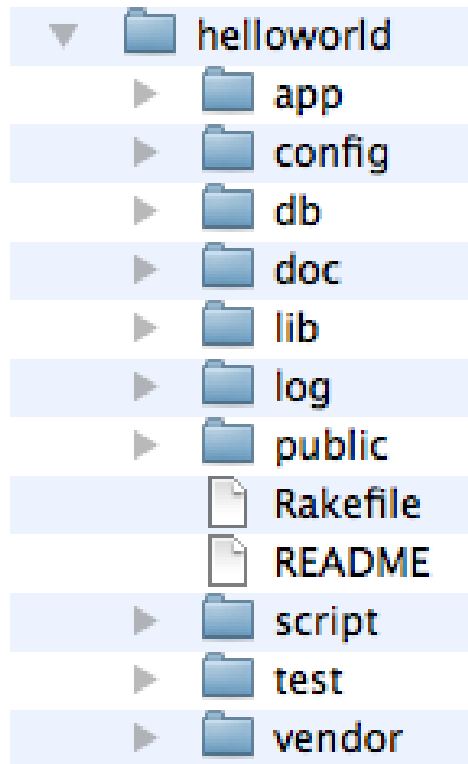
<http://glassfish.org/v3>

Dynamic Languages & Frameworks



<http://glassfish-scripting.dev.java.net>

Rails Deployment Choices



Credits: <http://birdwatchersdigest.com>

Demo

NetBeans/Eclipse and Java EE 6

http://blogs.sun.com/arungupta/entry/screencast_27_simple_web_application
http://blogs.sun.com/arungupta/entry/screencast_28_simple_web_application
http://blogs.sun.com/arungupta/entry/screencast_26_develop_run_debug/
http://blogs.sun.com/arungupta/entry/totd_93_getting_started_with/
http://blogs.sun.com/arungupta/entry/totd_94_a_simple_java
http://blogs.sun.com/arungupta/entry/totd_95_ejb_3_1
http://blogs.sun.com/arungupta/entry/totd_102_java_ee_6
http://blogs.sun.com/arungupta/entry/totd_99_creating_a_java
<http://blog.arungupta.me/2008/11/screencast-28-simple-web-application-using-eclipse-and-glassfish-v3-prelude/>

Embeddable GlassFish

```
public void testServlet() throws Exception {  
    int port = 9999;  
    GlassFish glassfish = newGlassFish(port);  
    URL url = new URL("http://localhost:" + port + "/" +  
NAME + "/SimpleServlet");  
    BufferedReader br = new BufferedReader(  
        new InputStreamReader(  
            url.openConnection().getInputStream()));  
    assertEquals("Wow, I'm embedded!", br.readLine());  
    glassfish.stop();  
}
```

... Embeddable GlassFish

```
private GlassFish newGlassFish(int port) throws Exception {  
    GlassFish glassfish = new GlassFish(port);  
    ScatteredWar war = new ScatteredWar(NAME,  
        new File("src/main/resources"),  
        new File("src/main/resources/WEB-INF/web.xml"),  
        Collections.singleton(new  
File("target/classes").toURI().toURL()));  
    glassfish.deploy(war);  
    System.out.println("Ready ...");  
    return glassfish;  
}
```

Extending GlassFish ... 1, 2, 3.

```
@Service(name="mycommand")
@Scoped(PerLookup.class)
public class CLIPluggabilityCommand implements AdminCommand {
    ...
}
```

```
...
// this value can be either runtime or os for our demo
@param(primary=true)
String inParam;
...
```

```
public void execute(AdminCommandContext context) {
    ...
}
```

Light-weight & On-demand Monitoring

- Event-driven light-weight and non-intrusive monitoring
- Modules provide domain specific probes (monitoring events)
 - > EJB, Web, Connector, JPA, Jersey, Orb, Ruby
- End-to-end monitoring on Solaris using DTrace
- 3rd party scripting clients
 - > JavaScript to begin with

Demo

GlassFish v3 Monitoring

REST Interface

- REST interface to management and monitoring data
 - > Configuration data, Commands invocation (start/stop instance, deploy, undeploy, ...), CRUD resources (JMS, JDBC, ...)
 - > localhost:4848/management/domain
 - > localhost:4848/monitoring/domain
- GET, POST, DELETE methods
- XML, JSON, HTML reps

Demo

GlassFish v3 REST Interface

http://blogs.sun.com/arungupta/entry/totd_113_javafx_front_end
http://blogs.sun.com/arungupta/entry/totd_96_glassfish_v3_rest



J-Fall

11 november 2009 Spant!



Java EE 6 and GlassFish v3: Paving the path for future

Arun Gupta, GlassFish Guy
Sun Microsystems, Inc.
blogs.sun.com/arungupta