
ORACLE®

ORACLE

The Java EE 7 Platform: Productivity++ and Embracing HTML5

Arun Gupta, Java EE & GlassFish Guy
@arungupta

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java EE 6 Platform
December 10, 2009

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java EE 6 – Key Statistics

- **50+ Million Java EE 6 Component Downloads**
- #1 Choice for Enterprise Developers
- #1 Application Development Platform
- Fastest implementation of a Java EE release

 Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

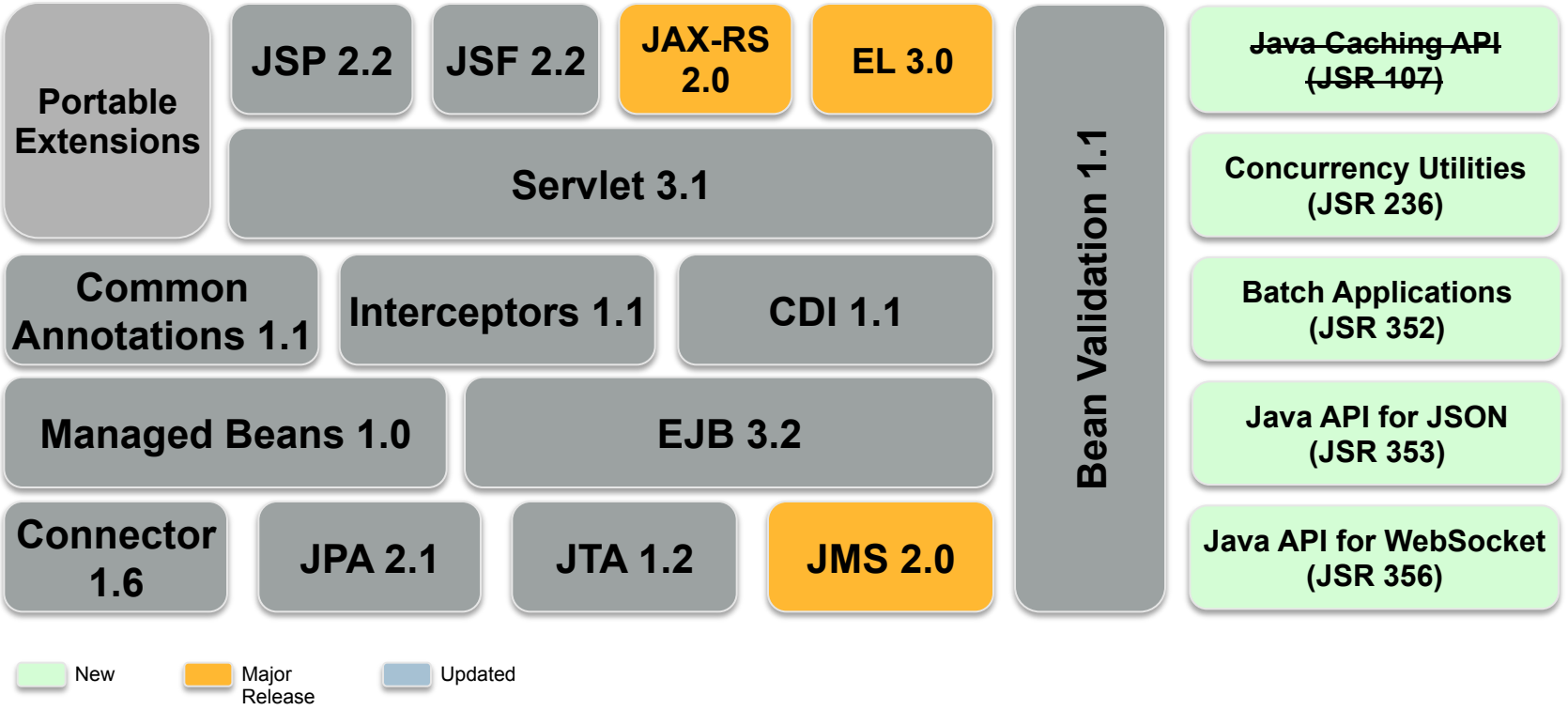
Java EE 7 Revised Scope

Productivity and HTML5

- Higher Productivity
  - Less Boilerplate
  - Richer Functionality
  - More Defaults
- HTML5 Support
  - WebSocket
  - JSON
  - HTML5 Forms

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java EE 7 – Candidate JSRs



Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for RESTful Web Services 2.0

- Client API
- Message Filters & Entity Interceptors
- Asynchronous Processing – Server & Client
- Hypermedia Support
- Common Configuration

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for RESTful Web Services 2.0

Client API - Now

```
// Get instance of Client
Client client = ClientFactory.newClient();

// Get customer name for the shipped products
String name = client.target("../orders/{orderId}/customer")
                    .resolveTemplate("orderId", "10")
                    .queryParam("shipped", "true")
                    .request()
                    .get(String.class);
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java Message Service 2.0

Simplify the existing API

- Less verbose
- Reduce boilerplate code
- Resource injection
- `Connection`, `Session`, and other objects are `AutoCloseable`
- Requires Resource Adapter for Java EE containers
- Simplified API in both Java SE and EE

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java Message Service 2.0

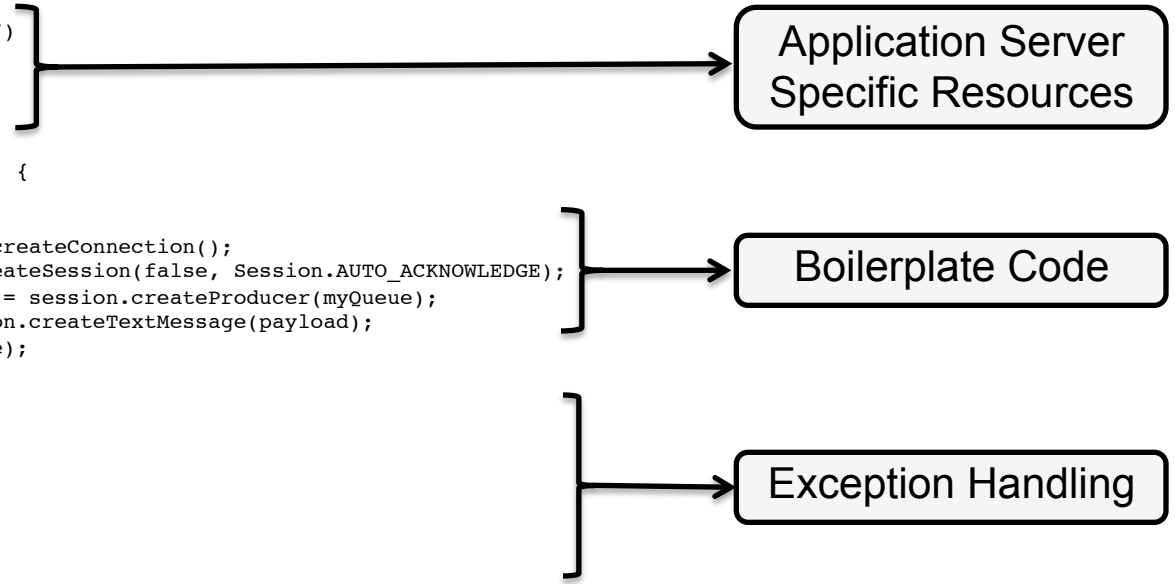
Sending a Message using JMS 1.1

```
@Resource(lookup = "myConnectionFactory")
ConnectionFactory connectionFactory;

@Resource(lookup = "myQueue")
Queue myQueue;

public void sendMessage (String payload) {
    Connection connection = null;
    try {
        connection = connectionFactory.createConnection();
        Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
        MessageProducer messageProducer = session.createProducer(myQueue);
        TextMessage textMessage = session.createTextMessage(payload);
        messageProducer.send(textMessage);
    } catch (JMSException ex) {
        //. . .
    } finally {
        if (connection != null) {
            try {
                connection.close();
            } catch (JMSException ex) {

                //. . .
            }
        }
    }
}
```



Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java Message Service 2.0

Sending message using JMS 2.0

```
@Inject
JMSContext context;

@Resource(lookup = "java:global/jms/demoQueue”)
Queue demoQueue;

public void sendMessage(String payload) {
    context.createProducer().send(demoQueue, payload);
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for JSON Processing 1.0

- API to parse and generate JSON
- Streaming API
  - Low-level, efficient way to parse/generate JSON
  - Provides pluggability for parsers/generators
- Object Model
  - Simple, easy-to-use high-level API
  - Implemented on top of Streaming API
- Binding JSON to Java objects forthcoming

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for JSON Processing 1.0

Streaming API – JsonParser

```
{
     "firstName": "John", "lastName": "Smith", "age": 25,
   "phoneNumber": [
      { "type": "home", "number": "212 555-1234" },
      { "type": "fax", "number": "646 555-4567" }
    ]
}
Iterator<Event> it = parser.iterator();
Event event = it.next();                    // START_OBJECT
event = it.next();                          // KEY_NAME
event = it.next();                          // VALUE_STRING
String name = parser.getString();           // "John"
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for WebSocket 1.0

- API for WebSocket Client/Endpoints
  - Annotation-driven (`@WebSocketEndpoint`)
  - Interface-driven (`Endpoint`)
  - Client (`@WebSocketClient`)
- SPI for data frames
  - WebSocket opening handshake negotiation
- Integration with Java EE Web container

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for WebSocket 1.0

Hello World – POJO/Annotation-driven

```
import javax.websocket.*;

@WebSocketEndpoint("/hello")
public class HelloBean {

    @WebSocketMessage
    public String sayHello(String name) {
        return “Hello “ + name;
    }
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for WebSocket 1.0

Chat Server

```
@WebSocketEndpoint("/chat")
public class ChatBean {
    Set<Session> peers = Collections.synchronizedSet(…);

    @WebSocketOpen
    public void onOpen(Session peer) {
        peers.add(peer);
    }

    @WebSocketClose
    public void onClose(Session peer) {
        peers.remove(peer);
    }

    . . .
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java API for WebSocket 1.0

Chat Server (contd.)

```
    . . .

    @WebSocketMessage
    public void message(String message, Session client) {
        for (Session peer : peers) {
            peer.getRemote().sendObject(message);
        }
    }
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

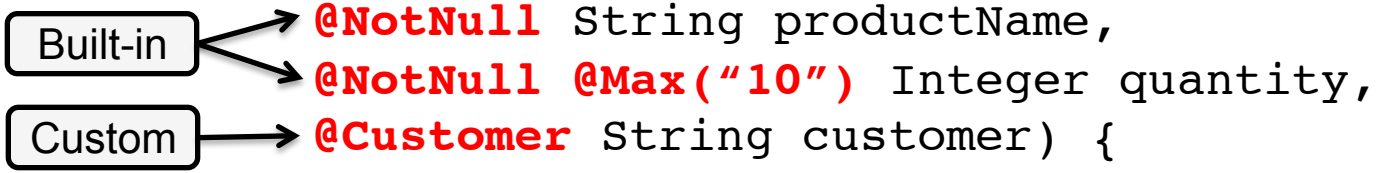
Bean Validation 1.1

- Open: Spec, Reference Implementation, TCK
- Alignment with Dependency Injection
- Method-level validation
  - Constraints on parameters and return values
  - Check pre-/post-conditions

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Bean Validation 1.1

Method Parameter and Result Validation




```
public void placeOrder(
        @NotNull String productName,
        @NotNull @Max("10") Integer quantity,
        @Customer String customer) {
            //. . .
}

@Future
public Date getAppointment() {
    //. . .
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Batch Applications for the Java Platform 1.0

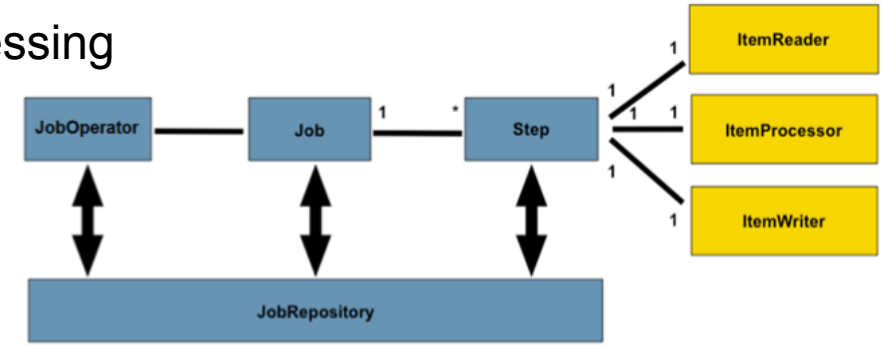
- Suited for non-interactive, bulk-oriented and long-running tasks
- Computationally intensive
- Can execute sequentially/parallel
- May be initiated
  - Adhoc
  - Scheduled
    - No scheduling APIs included

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Batch Applications for the Java Platform 1.0

Concepts

- **Job**: Entire batch process
  - Put together through a Job Specification Language (XML)
- **Step**: Independent, sequential phase of a job
  - **ItemReader**: Retrieval of input for a step, one at a time
  - **ItemProcessor**: Business processing of an item
  - **ItemWriter**: Output of an item, chunks of items at a time
- **JobOperator**: Manage batch processing
- **JobRepository**: Metadata for jobs



Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Batch Applications for the Java Platform 1.0

Job Specification Language – Chunked Step

```
<step id="sendStatements">
  <chunk reader="AccountReader"
     processor="AccountProcessor"
     writer="EmailWriter"
     chunk-size="10" />
</step>
```

```
@ReadItem
public Account readAccount() {
    // read account using JPA
}
```

```
@ProcessItem
public Account processAccount(Account account) {
    // calculate balance
}
```

```
@WriteItems
public void sendEmail(List<Account> accounts) {
    // use JavaMail to send email
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java Persistence API 2.1

- Schema Generation
- Unsynchronized Persistence Contexts
- Bulk update/delete using `Criteria`
- User-defined functions using `FUNCTION`
- Stored Procedure Query

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Servlet 3.1

- Non-blocking I/O
- Protocol Upgrade
- Security Enhancements

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Servlet 3.1

Non-blocking IO - Traditional

```
public class TestServlet extends HttpServlet
  protected void doGet(HttpServletRequest request,
                       HttpServletResponse response)
                  throws IOException, ServletException {
    ServletInputStream input = request.getInputStream();
    byte[] b = new byte[1024];
    int len = -1;
    while ((len = input.read(b)) != -1) {
      . . .
    }
  }
}
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Servlet 3.1

Non-blocking I/O: doGet Code Sample

```
AsyncContext context = request.startAsync();
ServletInputStream input = request.getInputStream();
input.setReadListener(
    new MyReadListener(input, context));
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Servlet 3.1

Non-blocking I/O: MyReadListener Code Sample

```
@Override
public void onDataAvailable() {
  try {
    StringBuilder sb = new StringBuilder();
    int len = -1;
    byte b[] = new byte[1024];
    while (input.isReady() && (len = input.read(b)) != -1) {
      String data = new String(b, 0, len);
      System.out.println("--> " + data);
    }
  } catch (IOException ex) {
    . . .
  }
}
. . .
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Concurrency Utilities for Java EE 1.0

Goals

- Provide concurrency capabilities to Java EE application components
  - Without compromising container integrity
- Support simple (common) and advanced concurrency patterns

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Concurrency Utilities for Java EE 1.0

Defining ManagedExecutorService using JNDI

- Recommended to bind in `java:comp/env/concurrent` subcontext

```
<resource-env-ref>
  <resource-env-ref-name>
    concurrent/BatchExecutor
  </resource-env-ref-name>
  <resource-env-ref-type>
    javax.enterprise.concurrent.ManagedExecutorService
  </resource-env-ref-type>
</resource-env-ref>
```


Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Concurrency Utilities for Java EE 1.0

Submit Tasks to ManagedExecutorService using JNDI

```
public class TestServlet extends HTTPServlet {
  @Resource(name="concurrent/BatchExecutor")
  ManagedExecutorService executor;

  Future future = executor.submit(new MyTask());

  class MyTask implements Runnable {
    public void run() {
      . . .  // task logic
    }
  }
}
```

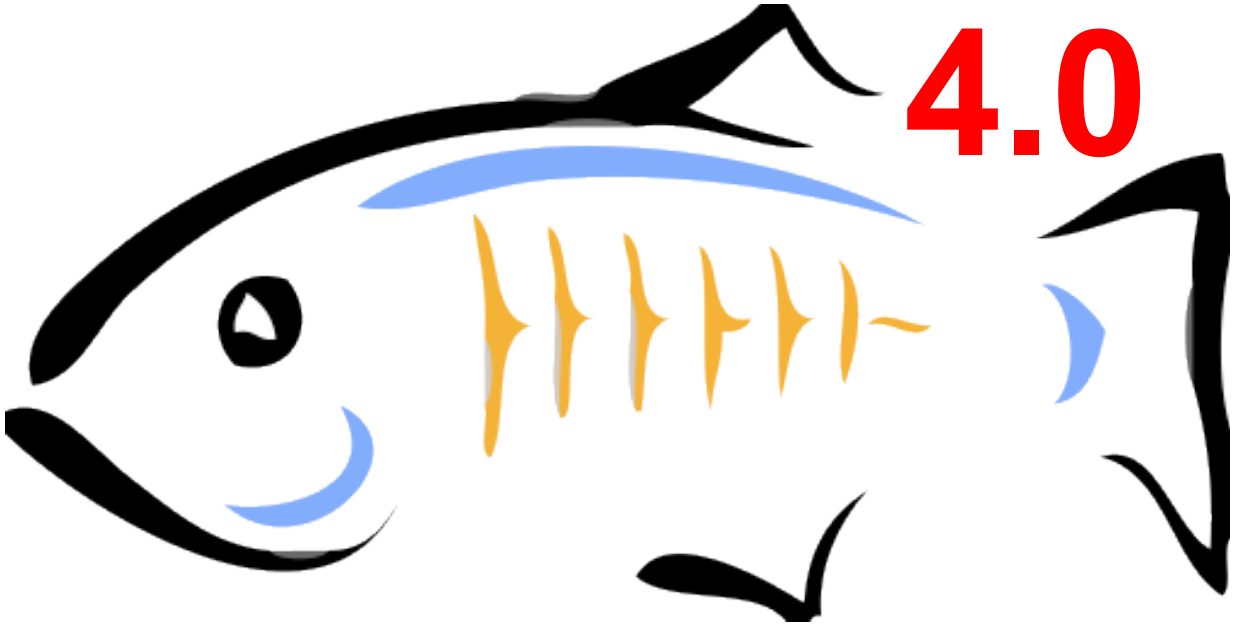

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

JavaServer Faces 2.2

- Flow Faces
- HTML5 Friendly Markup Support
  - Pass through attributes and elements
- Cross Site Request Forgery Protection
- Loading Facelets via `ResourceHandler`
- File Upload Component
- Multi-templating

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java EE 7 – Implementation Status



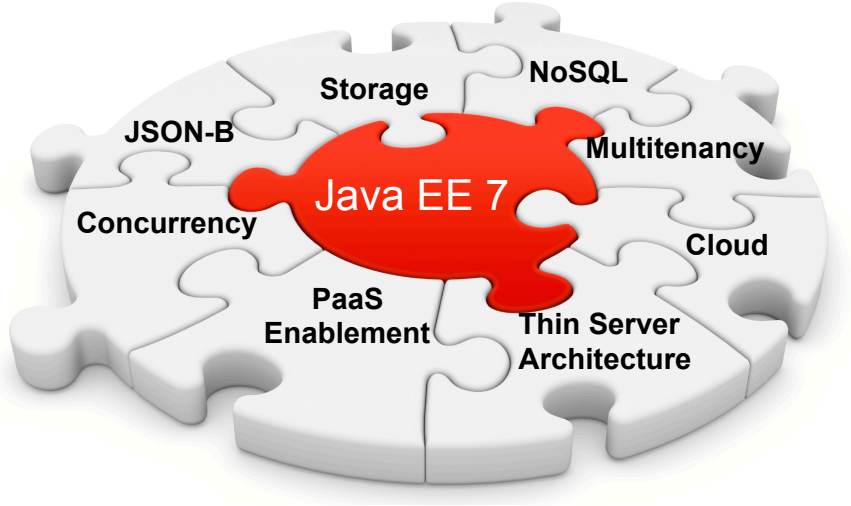
download.java.net/glassfish/4.0/promoted/

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Java EE 8 and Beyond

Standards-based cloud programming model

- Deliver cloud architecture
- Multi tenancy for SaaS applications
- Incremental delivery of JSRs
- Modularity based on Jigsaw



Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Adopt-a-JSR

How do I get started ? – glassfish.org/adoptajsr

- Java API for Temporary Caching 1.0 (JSR 107)
- Concurrency Utilities for Java EE 1.0 (JSR 236)
- Java Persistence API 2.1 (JSR 338)
- Java API for RESTful Web Services 2.0 (JSR 339)
- Servlet 3.1 (JSR 340)
- Expression Language 3.0 (JSR 341)
- Java Message Service 2.0 (JSR 343)
- JavaServer Faces 2.2 (JSR 344)
- Enterprise JavaBeans 3.2 (JSR 345)
- Contexts and Dependency Injection 1.1 (JSR 346)
- Bean Validation 1.1 (JSR 349)
- Batch Applications for the Java Platform 1.0 (JSR 352)
- Java API for JSON Processing 1.0 (JSR 353)
- Java API for WebSocket 1.0 (JSR 356)
- Java Transaction API 1.2 (JSR 907)

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Adopt-a-JSR

Participating JUGs

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Call to Action

- Specs: **javaee-spec.java.net**
- Implementation: **glassfish.org**
- The Aquarium: **blogs.oracle.com/theaquarium**
- Adopt a JSR: **glassfish.org/adoptajsr**
- NetBeans: **wiki.netbeans.org/JavaEE7**

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Q&A

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

Hardware and Software

ORACLE

Engineered to Work Together


ORACLE®