

Java EE 8

JavaCro

David Delabasse - @delabasse
Oracle



Agenda

Preview of Java EE 8

- 1 ➤ How did we get here?
- 2 ➤ What do we want to do?
- 3 ➤ How can you get involved?

Safe Harbor Statement

The following 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.

Java EE 7



- More annotated POJOs
- Less boilerplate code
- Cohesive integrated platform

- WebSockets
- JSON
- Servlet 3.1 NIO
- REST

- Batch
- Concurrency
- Simplified JMS

Industry Trends We're Seeing



Cloud



HTTP/2



User Experience

Mobile



Reactive Programming



SECURITY

Feedback from the Community

- Many sources
 - Users lists of java.net projects
 - JIRAs
 - JavaOne 2013 Java EE BOF and Java EE EG meeting
 - Outreach by evangelists
- Consolidated into Community Survey

Java EE 8 Community Survey

- 3 parts over 3½ months
 - 47 questions
 - 15 fill-ins
 - 1000's of comments
- 4500+ respondents
- Prioritization of most-popular features

Java EE 8 Community Survey - Features Prioritization

[Submit my priorities!](#)

How this works

We give you 100 "points" to spend across a variety of technologies. From how you allocate your points, we'll infer the importance you attach to including these technologies in Java EE 8. You are free to assign 100 points to a single technology and zero to the rest, or to spread your budget across as many options as you see fit.

How much would you spend to see the following included in Java EE 8?

JCACHE (JSR 107)

[more info](#)

Java API for JSON Binding

[more info](#)

Java EE Configuration

[more info](#)

Java API for server-sent events (SSE)

[more info](#)

An additional web MVC framework

[more info](#)

Support for CDI-based Security Interceptors for use in authorization decisions

[more info](#)

Generalization of the EJB Timer Service and integration of timer notifications with the CDI event/observer facility

[more info](#)

Features to support use in the cloud -- in particular, multi-instance multitenancy

[more info](#)

Improvements in logging

[more info](#)

Improvements and simplifications for security

[more info](#)

Definition of an embedded web or Java EE container for improved testability, use in production, or both

[more info](#)

New APIs (e.g., REST APIs) for Deployment, Management, Monitoring

[more info](#)

Pruning of the EJB 2.x remote and local client view and CORBA/IIOP interoperability

[more info](#)

Java EE 8

Driven by Community Feedback



You asked for it, you got it!

Java EE 8 Themes

- HTML5 / Web Tier Enhancements
- Ease of Development
- Infrastructure for running in the Cloud

Java EE 8 Themes

- **HTML5 / Web Tier Enhancements**
- Ease of Development
- Infrastructure for running in the Cloud

HTML5 Support / Web Tier Enhancements

- JSON Binding
- JSON Processing enhancements
- Server-sent Events
- Action-based MVC
- HTTP/2 support

JSON-B

Java API for JSON Binding

- API to marshal/unmarshal Java objects to/from JSON
 - Similar to JAXB runtime API in XML world
- Default mapping of classes to JSON
 - Annotations to customize the default mappings
 - JsonProperty, JsonTransient, JsonNillable, JsonValue, ...

JSON-B

Standard API

- Draw from best practices of existing JSON binding implementations
 - Jackson, Genson, EclipseLink MOXy, Fleece, JSON-lib, Gson, Flexjson, Json-io, JSONiJ, Johnzon, Xstream, etc.
- Switch JSON binding providers
- Implementations compete on common ground

JSON-B

```
@Entity public class Person {  
    @Id String name;  
    String gender;  
    @ElementCollection Map<String,String> phones;  
    ... // getters and setters  
}
```

```
Person duke = new Person();  
duke.setName("Duke");  
duke.setGender("M");  
phones = new HashMap<String,String>();  
phones.put("home", "650-123-4567");  
phones.put("mobile", "650-234-5678");  
duke.setPhones(phones);
```

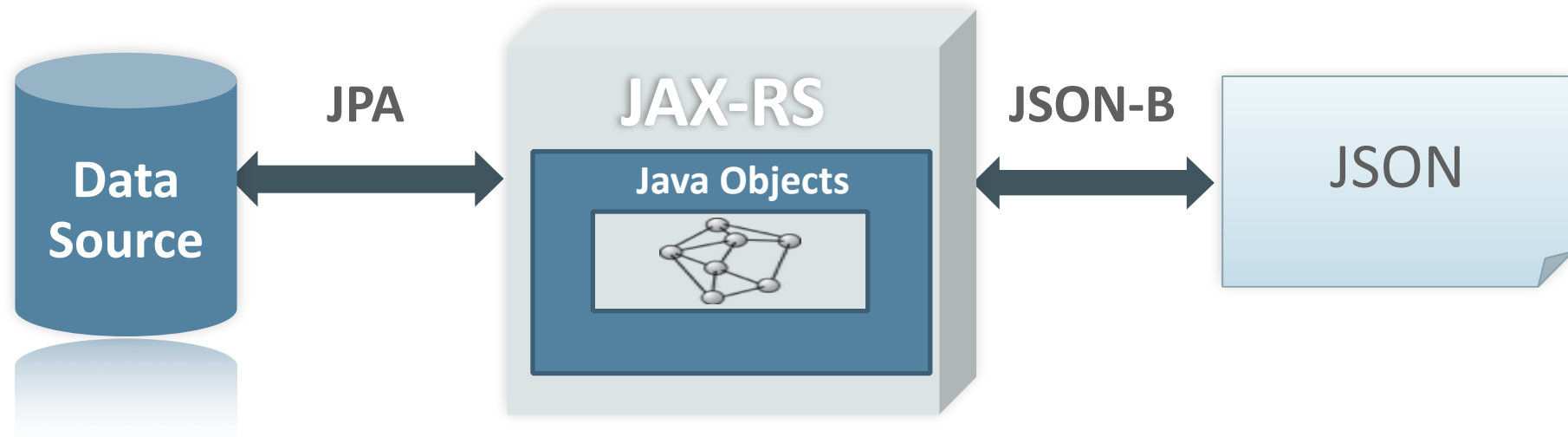
```
Marshaller marshaller = new JsonContext().createMarshaller().setPrettyPrinting(true);  
marshaller.marshal(duke, System.out);
```

```
{  
  "name": "Duke",  
  "gender": "M",  
  "phones": {  
    "home": "650-123-4567",  
    "mobile": "650-234-5678"}  
}
```


JSON-B

JSR 367

- All the way from client to database
 - JSON-B will provide JAX-RS a standard way to support “application/json” media type



JSON-P 1.1

Java API for JSON Processing

- Keep JSON-P spec up-to-date
- Track new standards
- Add editing operations to JsonObject and JsonArray
- Java SE 8
- JSON Big Data

JSON-P: Java API for JSON Processing 1.1

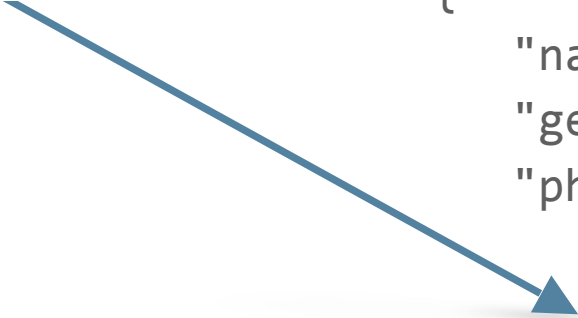
JSON-Pointer – IETF RFC 6901

- String syntax for referencing a JSON value
"/0/phone/mobile"

JSON-P 1.1

JSONArray contacts = ...

```
[
  {
    "name": "Duke",
    "gender": "M",
    "phones": {
      "home": "650-123-4567",
      "mobile": "650-234-5678"
    }
  },
  {
    "name": "Jane",
    "gender": "F",
    "phones": {
      "mobile": "707-555-9999"
    }
  }
]
```



JSON-P: Java API for JSON Processing 1.1

JSON-Pointer – IETF RFC 6901

- String syntax for referencing a JSON value
 - `"/0/phone/mobile"`
- Methods
 - `getValue()`
 - JSON Patch operations: `add()`, `replace()`, `remove()`, ~~`move()`~~, ~~`copy()`~~, ~~`test()`~~

JSON-P 1.1

JSON-Patch – IETF RFC 6902

- Patch is a JSON document
 - Array of objects / operations for modifying a JSON document
 - add, replace, remove, move, copy, test

```
[  
  {"op": "replace", "path": "/0/phones/mobile", "value": "650-111-222"},  
  {"op": "remove", "path": "/1"}  
]
```

- apply Vs. diff

JSON-P 1.1

```
[
  {
    "op": "replace",
    "path": "/0/phones/mobile",
    "value": "650-111-2222"},
  {
    "op": "remove",
    "path": "/1"}
]
```

```
[
  {
    "name": "Duke",
    "gender": "M",
    "phones": {
      "home": "650-123-4567",
      "mobile": "650-234-5678"}},
  {
    "name": "Jane",
    "gender": "F",
    "phones": {
      "mobile": "707-555-9999"}}
]
```

JSON-P 1.1

```
[
  {
    "op": "replace",
    "path": "/0/phones/mobile",
    "value": "650-111-2222"},
  {
    "op": "remove",
    "path": "/1"}
]
```

```
[
  {
    "name": "Duke",
    "gender": "M",
    "phones": {
      "home": "650-123-4567",
      "mobile": "650-111-2222"}},
  {
    "name": "Jane",
    "gender": "F",
    "phones": {
      "mobile": "707-555-9999"}}
]
```


JSON-P 1.1

```
[
  {
    "op": "replace",
    "path": "/0/phones/mobile",
    "value": "650-111-2222"},
  {
    "op": "remove",
    "path": "/1"}
]
```

```
[
  {
    "name": "Duke",
    "gender": "M",
    "phones": {
      "home": "650-123-4567",
      "mobile": "650-111-2222"}}
]
```

JSON-P 1.1

JSON Query using Lambda Operations

```
JsonArray contacts = ...;  
List<String> femaleNames = contacts.getValuesAs(JsonObject.class).stream()  
    .filter(x->"F".equals(x.getString("gender")))  
    .map(x->(x.getString("name"))  
    .collect(Collectors.toList());
```

JSON-P 1.1

JSON Query collecting results in JsonArray

```
JsonArray contacts = ...;  
JsonArray femaleNames = contacts.getValuesAs(JsonObject.class).stream()  
    .filter(x->"F".equals(x.getString("gender")))  
    .map(x->(x.getString("name"))  
    .collect(JsonCollectors.toJsonArray());
```

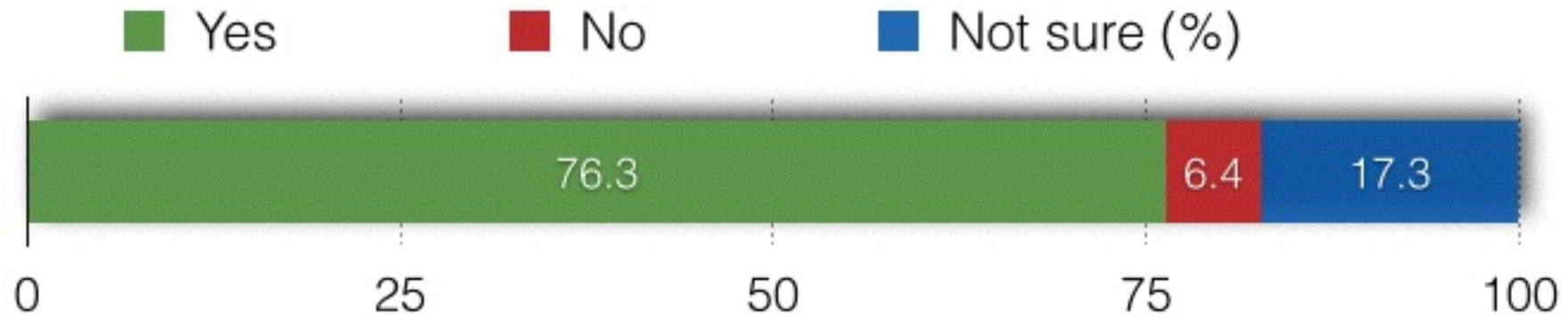
JSON-P 1.1

JSR 374

- Keep JSON-P spec up-to-date
- Track new standards
- Add editing operations to JsonObject and JsonArray
- Java SE 8
- JSON Big Data

Server-sent Events

Should we also standardize a Java API for server-sent events?



Server-sent Events

- Part of HTML5 standardization
- Server-to-client streaming of text data
- Media type: “text/event-stream”
- Long-lived HTTP connection
 - Client establishes connection
 - Server pushes update notifications to client

Server-sent Events

- Servlet, WebSocket, JAX-RS or standalone API?
- JAX-RS deemed most natural fit
 - Streaming HTTP resources already supported
 - Small extension
 - Server API: new media type; EventOutput
 - Client API: new handler for server side events
 - Convenience of mixing with other HTTP operations; new media type
 - Jersey already supports SSE

Server-sent Events

JAX-RS resource class

```
@Path("tickers")
public class StockTicker {
    ...
    @Get
    @Produces("text/event-stream")
    public EventOutput getQuotes() {
        EventOutput eo = new EventOutput();
        new StockThread(eo).start();
        return eo;
    }
}
```


Server-sent Events

JAX-RS StockThread class

```
class StockThread extends Thread {
    private EventOutput eo;
    ...
    @Override
    public void run() {
        try {
            ...
            eo.send(new StockQuote("..."));
        } catch (IOException e) { ... }
    }
}
```

Server-sent Events

JAX-RS Client

```
WebTarget target = client.target("http://example.com/tickers");
EventSource eventSource = new EventSource(target) {
    @Override
    public void onEvent(InboundEvent inboundEvent) {
        StockQuote sq = inboundEvent.readData(StockQuote.class);
        // ...
    }
};
eventSource.open();
```

MVC 1.0

Action-based Model-View-Controller architecture

- Why?
 - Java EE 8 Community Survey
 - UI landscape is not one size fits all

MVC 1.0

Glues together key Java EE technologies

- Model
 - CDI, Bean Validation, JPA
- View
 - Facelets, JSP, SPI?
- Controller
 - Invent new technology Vs. Leverage existing technologies?

Controller

```
@Path("hello")
@Controller
public class HelloController {

    @GET
    @View("view1.xhtml")
    public void hello() {
        ...
    }
}
```

Controller

```
@Path("hello")
public class HelloController {

    @GET
    @Controller
    public String hello() {
        ...
        return "viewA.jsp";
    }
}
```

Model

```
@Named("greeting")
@RequestScoped
public class Greeting {
    private String message;
    public String getMessage() {
        return message;
    }
    public void setMessage(String message) {
        this.message = message;
    }
}
```

View

```
<%@ page contentType="text/html; charset=UTF-8"
        language="java" %>

<html>
<head>
    <title>Hello</title>
</head>
<body>
    <p>Hello ${greeting.message}</p>
</body>
</html>
```

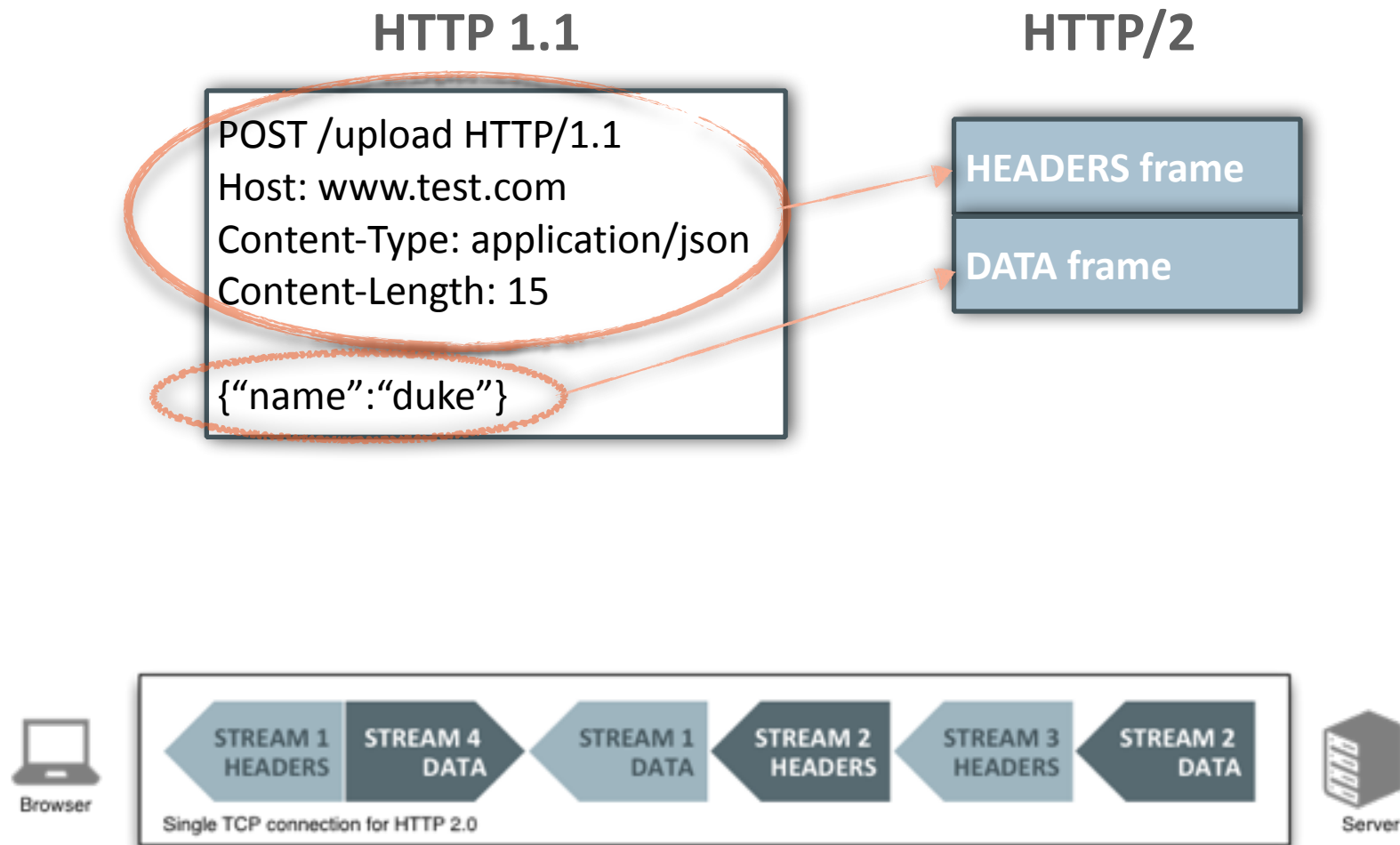

JSR 371

- View Engine
 - JSP & Facelets
 - FreeMarker, Velocity, Thymeleaf, Mustache, Handlebars & Pebble
- Validation
- Exception Mapping Providers
- Bootstrap via `javax.ws.rs.core.Application`
- `@Produces`
- CDI Events

HTTP/2

Multiplexed Binary Frames

- One TCP Connection
- Request -> Stream
 - Multiplexed
 - Prioritised
- Binary Framing Layer
 - Prioritisation
 - Flow Control
 - Server Push
- Header Compression



Servlet 4.0

HTTP/2 Features in Servlet API

- Request/response multiplexing
 - Servlet Request as HTTP/2 message
- Stream prioritization
 - Add stream priority to `HttpServletRequest`
- Server push
- Binary framing
- Upgrade from HTTP 1.1

JSF 2.3

- CDI Alignment
 - @Inject FacesContext, ExternalContext, etc.
 - Rely on CDI for EL resolving
 - CDI managed versions of Validator and Converter
 - Invoking CDI managed bean methods directly from Ajax, etc.
- “Adjustments” for MVC
 - Facelets, JSF scopes, etc.
- Misc.
 - Multi-field validation, etc.

Java EE 8 Themes

- HTML5 / Web Tier Enhancements
- **Ease of Development**
- Infrastructure for running in the Cloud

Ease of Development

- CDI alignment
- JAX-RS injection alignment
- Simplified messaging through CDI-based “MDBs”
- WebSocket scopes
- Pruning of EJB 2.x client view and IIOP interoperability
- Security interceptors
- ...

JMS 2.1

New API to receive messages asynchronously

- Alternative to EJB message-driven beans
- Simpler JMS-specific annotations
- Usable by any CDI bean
- No need for MessageListener implementation

JMS

JMS MDBs Today

```
@MessageDriven(activationConfig = {
    @ActivationConfigProperty(propertyName="connectionFactoryLookup", propertyValue="jms/myCF"),
    @ActivationConfigProperty(propertyName="destinationLookup", propertyValue="jms/myQueue"),
    @ActivationConfigProperty(propertyName="destinationType", propertyValue="javax.jms.queue")})

public class MyMDB implements MessageListener {
    public void onMessage(Message message) {
        // extract message body
        String body = message.getBody(String.class);
        // process message body
    }
}
```


JMS 2.1

Allow any Java EE bean to be a listener

```
public class MyListenerBean {  
    @JMSListener(destinationLookup="jms/myQueue")  
    @Transactional  
    public void myCallback(Message message) {  
        ...  
    }  
}
```

CDI 2.0

- Java SE support
- Modularity
- Enhanced Events
- Misc.
 - AOP
 - SPI
 - Interceptors and Decorators enhancements
 - Cleaning

CDI 2.0

Modularity

- Add new features to CDI without bloating the specification
- “Sub specification”(aka “parts”) that can be used independently
- Will help CDI adoption
- Parts
 - SE
 - EE
 - more?

CDI 2.0

Asynchronous Events

```
public class AsynchProducerClass {  
    @Inject Event<Payload> someEvent;  
    public void anotherMethod() {  
        CompletionStage<...> completionStage someEvent.fireAsync(...);  
    }  
}
```

```
public class AnotherClass {  
    public void someObserver(@Observes SomeEvent someEvent) {  
        ...  
    }  
}
```

CDI 2.0

Events Ordering

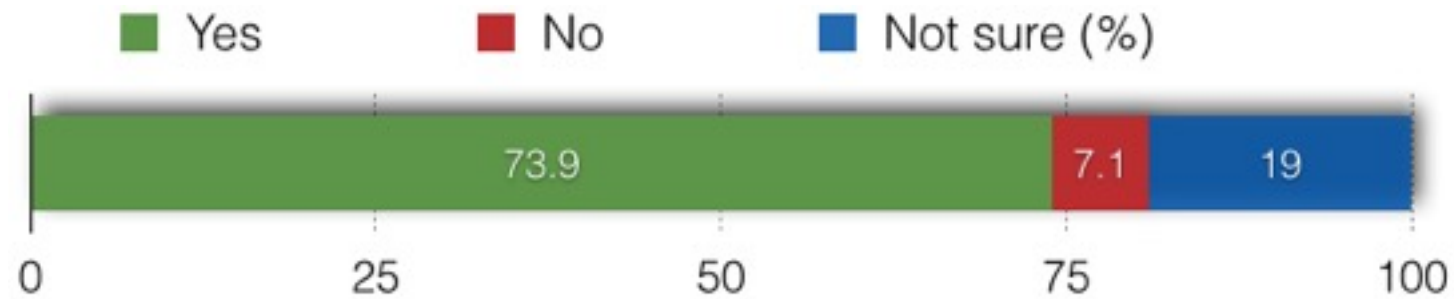
```
public void importantObserver(@Observes @Priority(1) MyEvent evt)
{
    ...
}

public void anotherObserver(@Observes @Priority(10) MyEvent evt)
{
    ...
}
```

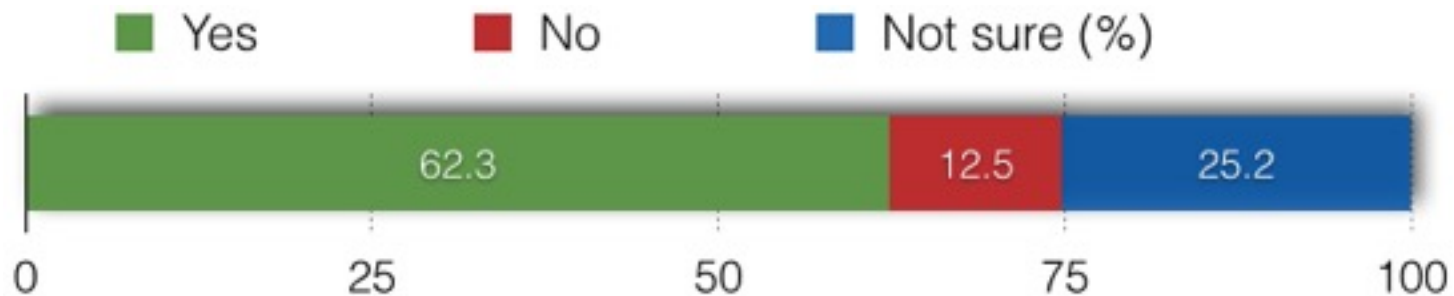
Pruning

Candidates for Proposed Optional status

Should we prune EJB 2.x remote and local client view (EJBObject, EJBLocalObject, EJBHome, and EJBLocalHome interfaces)?



Should we prune CORBA, including support for interoperability by means of IIOP?



Java EE 8 Themes

- HTML5 / Web Tier Enhancements
- Ease of Development
- **Infrastructure for running in the Cloud**

Modernize the Infrastructure

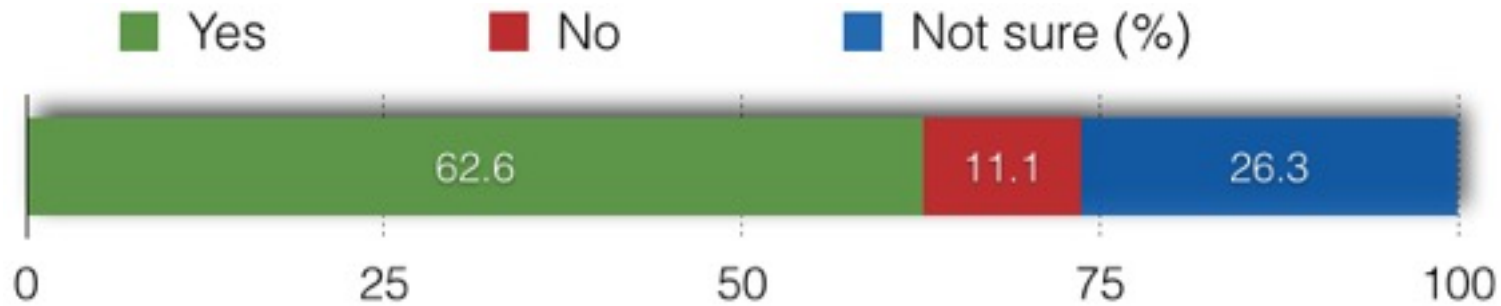
For On-Premise and for in the Cloud

- Java EE Management 2.0
 - REST-based APIs for Management and Deployment
- Java EE Security 1.0
 - Authorization
 - Password Aliasing
 - User Management
 - Role Mapping
 - Authentication
 - REST Authentication



Management and Deployment APIs

Should we define new APIs to deploy and manage applications?



Should such new Deployment and Management APIs be REST APIs or JMX APIs?

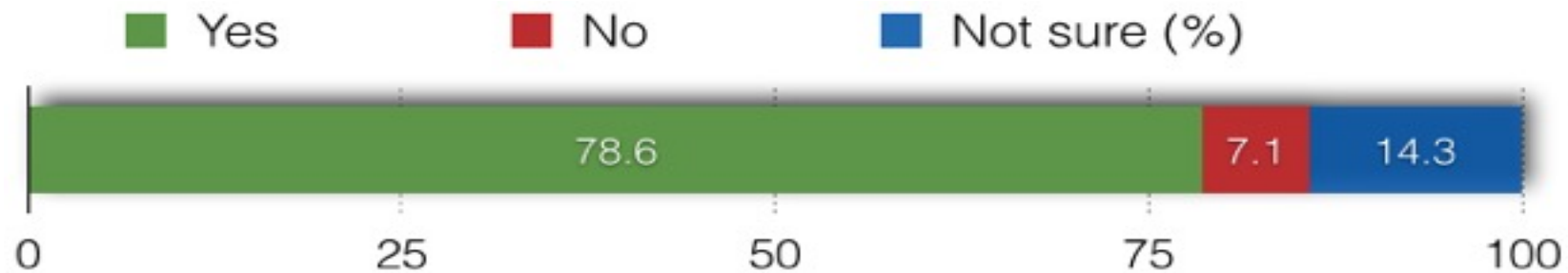


Java EE Management 2.0

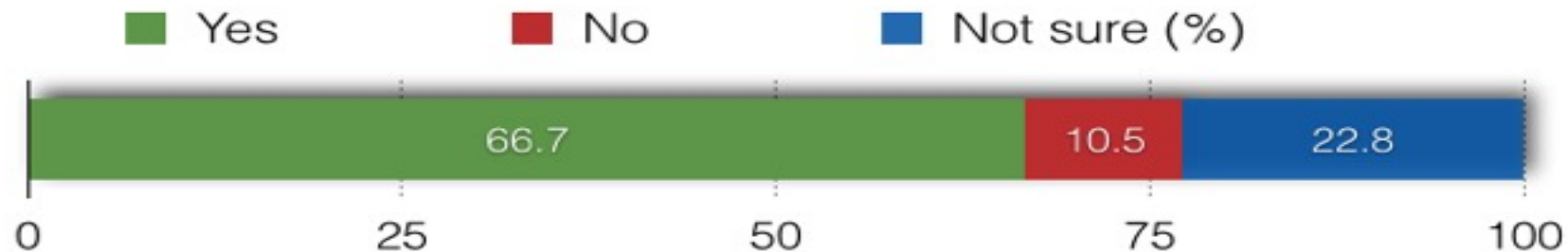
- Update to JSR 77 (“J2EE Management”)
- REST-based interfaces to augment (or replace) current Management EJB APIs
 - Currently used OBJECT_NAME to become URL
 - Define CRUD operations over individual managed objects
 - Server-sent events used for event support
- Simple deployment interfaces also to be considered as part of management API

Security Simplifications

Should we consider adding Security Interceptors in Java EE 8?



Should we simplify authorization by introducing an EL-enabled authorization annotation?



Java EE Security 1.0

Authorization via CDI Interceptors

```
@IsAuthorized("hasRoles('Manager') && schedule.officeHrs")  
void transferFunds()
```

Java EE Security 1.0

Candidate Areas to Enhance Portability, Flexibility, Ease-of-Use

- Authorization Interceptors
- Password Aliasing
- User Management
- Role Mapping
- Authentication
- REST Authentication

Java EE 8 Themes

- HTML5 / Web Tier Enhancements
- Ease of Development
- Infrastructure for running in the Cloud

Java EE 8 JSRs

So far.....

- Java EE 8 Platform (JSR 366)
- CDI 2.0 (JSR 365)
- JSON Binding 1.0 (JSR 367)
- JMS 2.1 (JSR 368)
- Java Servlet 4.0 (JSR 369)
- JAX-RS 2.1 (JSR 370)
- MVC 1.0 (JSR 371) *
- JSF 2.3 (JSR 372)
- Java EE Management 2.0 (JSR 373)
- JSON-P 1.1 (JSR 374)
- Java EE Security 1.0 (JSR 375)

And More to Follow...

- Bean Validation
- EL
- Concurrency Utilities
- Connector Architecture
- WebSocket
- Interceptors
- JPA
- EJB
- JTA
- JCache
- Batch
- JavaMail
- ...

Roadmap

- Tentative Delivery Schedule
 - Q3 2014: JSR 369 Expert Group formed
 - Q1 2015: early draft
 - Q3 2015: public review
 - Q4 2015: proposed final draft
 - Q3 2016: final release

How to Get Involved

- Join an Expert Group
 - <http://javaee-spec.java.net>
- Adopt a JSR
 - <http://glassfish.org/adoptajsr>
- The Aquarium
 - <http://blogs.oracle.com/theaquarium>
- Java EE Reference Implementation
 - <http://glassfish.org>



Thanks

CREATE THE FUTURE

