31.8.–3.9.2015
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Bestandsaufnahme

Java Enterprise Edition 8

Peter Doschkinow

ORACLE Deutschland B.V. & Co. KG

Safe Harbor Statement

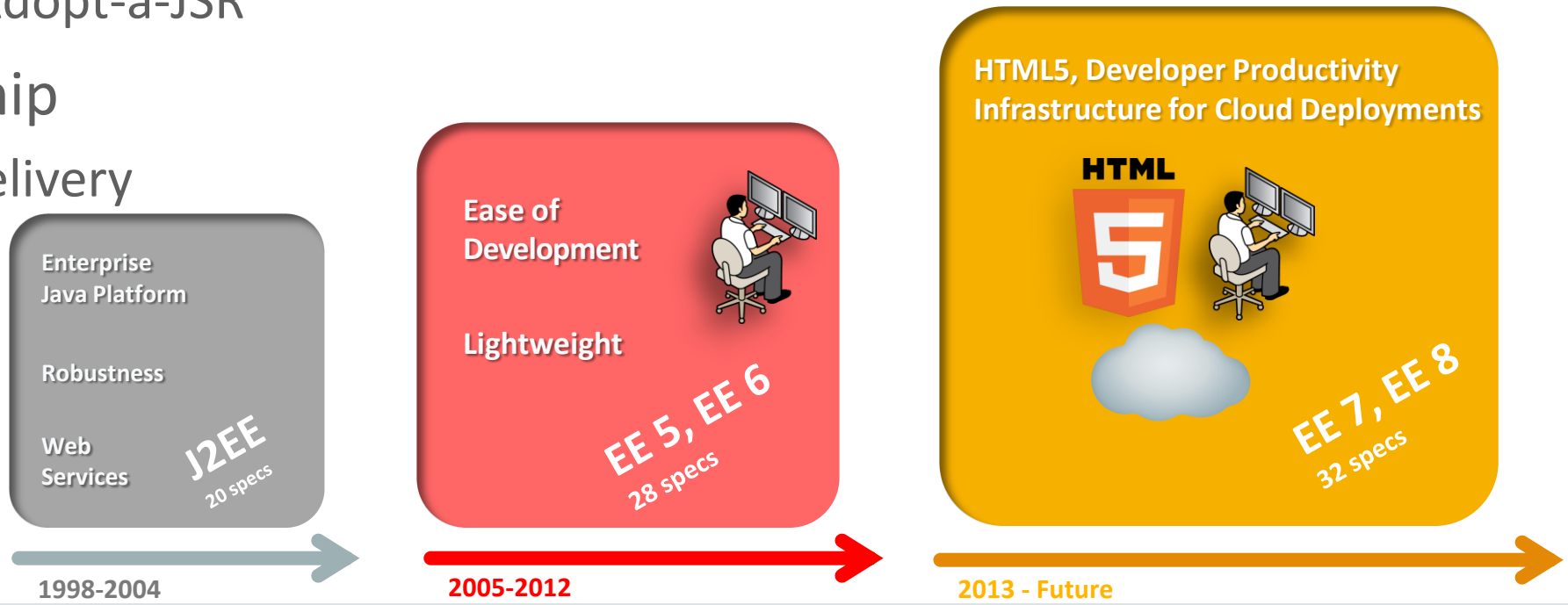
The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development,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

Agenda

1. Java EE Status
2. Java EE 8 Themes
3. Java EE 8 Roadmap

Java EE Evolution and Status

- Technology
  - Balance innovation, standardization, compatibility
- Community
  - Involvement: JCP, Adopt-a-JSR
- Oracle‘s stewardship
  - Investments and delivery



Java EE 7

- Full implementations
  - GlassFish 4.x
  - WildFly 8.x
  - TMAX JEUS 8
  - Hitachi Cosminexus 10.0
  - IBM WebSphere Liberty 8.5.5.6
- Partial commercial implementations
  - WebLogic 12.1.3

http://www.oracle.com/technetwork/java/javaee/overview/compatibility-jsp-136984.html

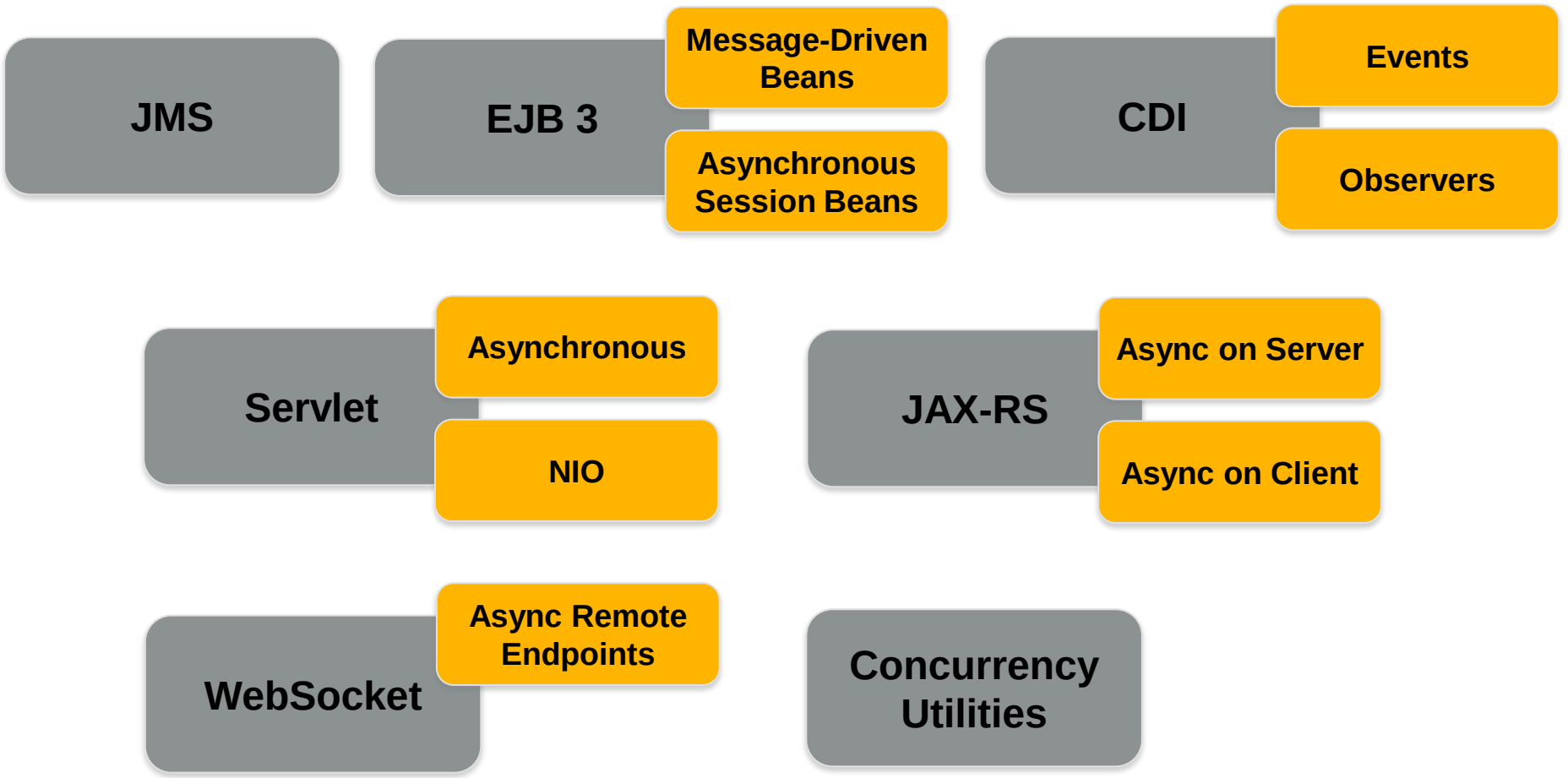
Industry Trends

HTTP/2

Microservices

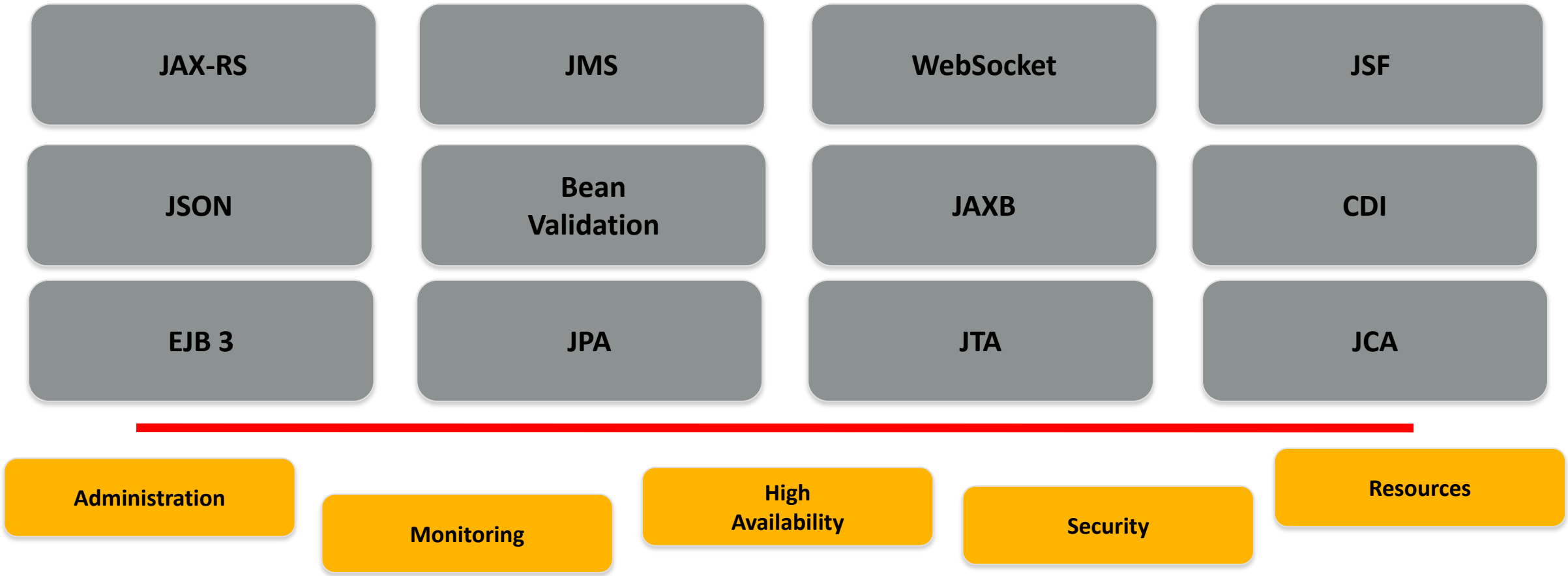
Reactive Programming

Java EE and Reactive Programming



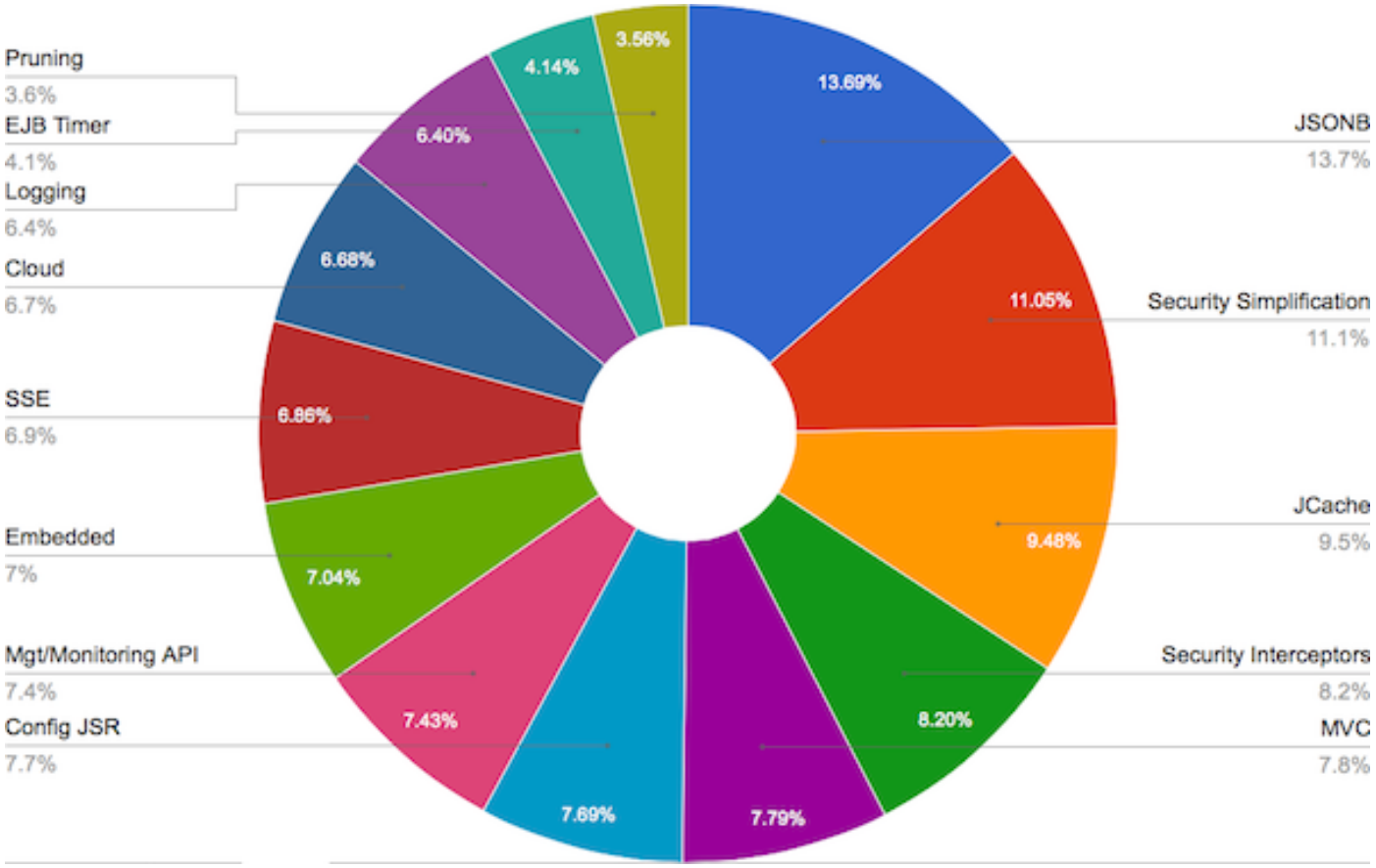
Java EE and Microservices

Building blocks for pragmatic microserveces



Java EE 8 Community Survey

4500+ respondents



https://blogs.oracle.com/ldemichiel/entry/results_from_the_java_ee

https://java.net/downloads/javaee-spec/JavaEE8_Community_Survey_Results.pdf

Java EE 8 Themes

- HTML5 / Web Tier Enhancements
- Ease of Development / CDI alignment
- Infrastructure for running in the Cloud
- Java SE 8 alignment
  - Utilizing Java SE 8 new features

HTML5 Support and Web Tier Enhancements

- HTTP/2 support
- Action-based MVC
- Server-sent events
  - Part of HTML5
  - By extending JAX-RS (already supported by Jersey)
- JSON Binding
  - Similar to JAXB (already supported by MOXy, included in Jersey)
- JSON Processing enhancements
  - Track new standards JSON-Pointer and JSON-Patch

HTTP/2

Address the Limitations of HTTP

- HTTP/2 IETF standard finalized in May (RFC 7540)
- Reduce latency
  - Address the Head-of-Line blocking problem
- Support parallelism
  - Without requiring multiple connections
- Retain semantics of HTTP 1.1
- Define interaction with HTTP 1.x

HTTP/2

- Request/Response multiplexing over single connection
  - Fully bidirectional
  - Multiple streams
- Stream prioritization
- Server push
- Binary framing
- Header compression (RFC 7541)
- Upgrade from HTTP 1.1

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: new class Priority
- Server push
- Upgrade from HTTP 1.1
- Binary framing
  - Hidden from API

JAX-RS 2.1

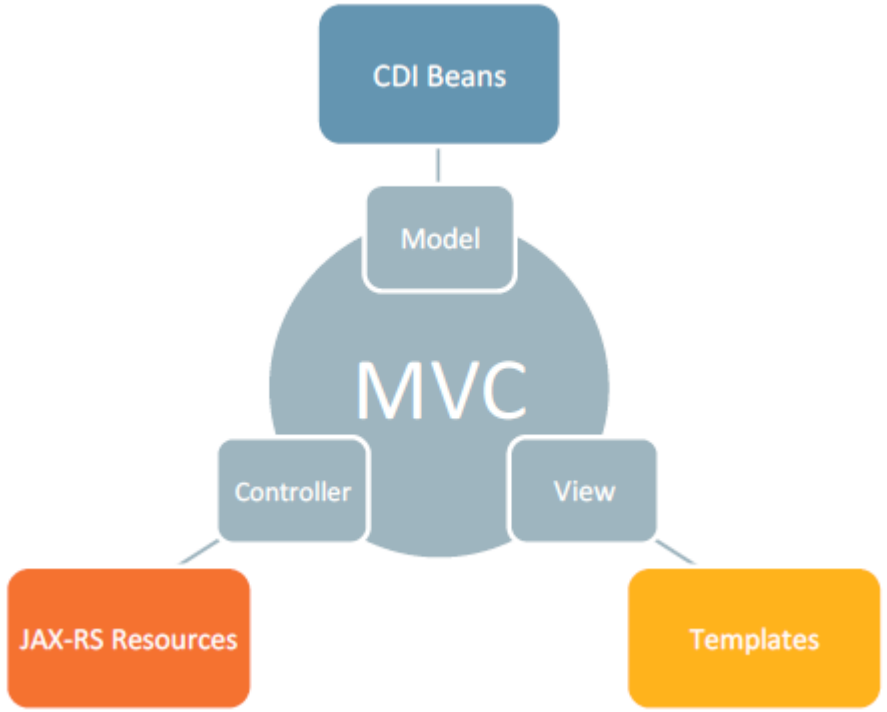
Covers many of the new Java EE 8 API

- Support for Server Sent Events
- JSON-B integration
- MVC support through JAX-RS ressources as controllers
- Non-blocking I/O in porviders
- Better CDI integration
- Hypermedia API improvements
- Reactive programming API
- Most of the above already implemented in Jersey (JAX-RS 2.0 RI)

MVC 1.0

JSR 371

- Action-based Model-View-Controller architecture
  - vs. component-based like JSF, Wicket, Tapestry
- Glues together key Java EE technologies:
  - Model
    - CDI, Bean Validation, JPA
  - View
    - Facelets, JSP
  - Controller
    - JAX-RS Ressources
- https://ozark.java.net (RI)



MVC 1.0

Example

```
@Path("hello")
public class HelloController
{
        @Inject private User user;

        @GET
        @Controller
        public String hello(@QueryParam("name") String name) {
        user.setName(name);
        return "hello.jsp";
        }
 }
```

Server-Sent Events (SSE)

- Part of HTML 5
  - Standard JavaScript API on the browser
- Server-to-client streaming
  - “Stock tickers”, monitoring applications
- Just plain long-lived HTTP
  - Between the extremes of vanilla request/response and WebSocket
  - Content-type ‘text/event-stream’
- Support via JAX-RS.next()
  - Already supported in Jersey JAX-RS reference implementation

SSE on the Server-Side

```
@Path("tickers")
public class StockTicker {
  @Resource ManagedExecutorService executor;

  @GET @Produces("text/event-stream")
  public EventOutput getQuotes() {
    EventOutput output = new EventOutput();

    executor.execute(() -> {
      ...
      output.send(new StockQuote(...));
      ...
    });

    return output;
  }
}
```

SSE on the Client-Side

```
WebTarget target = client.target("http://example.com/tickers");

EventSource eventSource = new EventSource(target) {
  @Override
  public void onEvent(InboundEvent inboundEvent) {
    StockQuote quote = inboundEvent.readData(StockQuote.class);
    ...
  }
};

eventSource.open();
```

JSON-P 1.1

- Updates to new API in Java EE 7
- Adapt to new JSON standards
  - JSON-Pointer (IETF RFC 6901)
  - JSON-Patch (IETF RFC 6902)
- Editing operations for JsonObject and JsonArray
- Helper classes and methods to better utilize SE 8’s stream operations

JSON-Pointer

Similar to XPATH in XML

```
JsonArray contacts = ...;
JsonPointer pointer =
    Json.createPointer(
        "/0/phones/mobile");
JsonValue value =
    pointer.getValue(contacts);
```

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

JSON-Patch

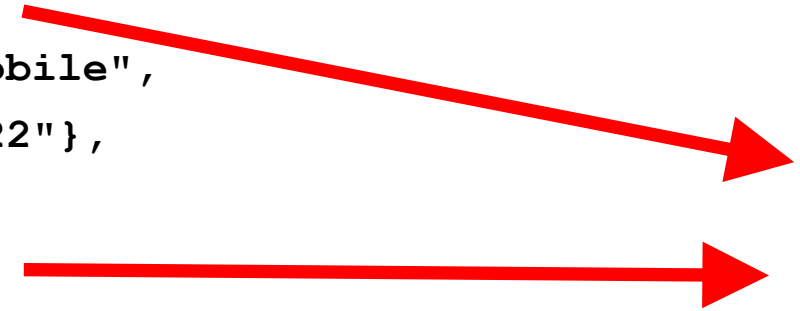
Similar to HTTP PATCH

- Modifying parts of a JSON document
- Patch itself a JSON document
  - add, replace, remove, move, copy, test operations
  - Must have "op" field and "path" field, may have “value” field
- JsonObject and JsonArray are immutable
  - Utilize Builder pattern for editing API?

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

JSON-Patch Possibilities

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



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

JSON Query using Lambda Operations

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

JSON Query Collecting Results in JsonArray

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

- Using specialized collectors we can retrieve JSON-P API objects like JsonArray that can be directly manipulated further

Ease of Development and CDI alignment

- EJB services outside EJB
- Simplified messaging through CDI-based “MDBs”
- CDI improvements
  - Events: async, ordering, range (war, ear, server, cluster)
  - WebSocket scopes
- Pruning of EJB 2.x client view and IIOP interoperability

EJB Services outside EJB

```
@ApplicationScoped
public class MyScheduledBean {
   ...
   @Schedule(...)
   public void myScheduledTask() { ... }
}

@ApplicationScoped
@Stereotype
@Retention(RUNTIME)
@Target(TYPE)
@Schedule(...)
public @interface MonthlyTask {}
```

Simplified messaging with CDI beans

JMS 2.1

- New API to receive messages asynchronously
- Alternative to message driven beans
- Simpler JMS annotations
  - Usable by any CDI bean
  - No need for MessageListener implementation

```
@RequestScoped
public class MyListenerComponent {

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

- Planning https://java.net/projects/jms-spec/pages/JMS21Planning

CDI 2.0

- Specification modularization
  - CDI full with Java EE support, CDI full, CDI light (no support for contexts and AOP)
- Observer ordering – priority annotation
- Asynchronous events
- Java SE bootstrap
- Improvements for interceptors and decorators
  - Repeatable annotations for qualifiers and interceptor bindings
- Extension SPI simplification
- EJB annotation usage in CDI beans

CDI 2.0

Asynchronous events – alternative implementations for sender and receiver?

```
@Inject @CargoInspected Event<Cargo> cargoInspected;
...
public void inspectCargo(TrackingId trackingId) {
   ...
   cargoInspected.fireAsync(cargo);
}


public void onCargoInspected(
     @Observes(async=true) @CargoInspected Cargo cargo) {
```

Infrastructure Modernization

For Cloud and on-premise

- Java EE Management 2.0
  - REST-based APIs for Management and Deployment
- Java EE Security 1.0: enhance portability, flexibility, ease-of-use
  - Password aliasing
  - Security interceptor annotations
  - User management and simple security providers
  - Role mapping
  - Authentication
  - REST authentication (Oauth/OpenID)

Java EE Management 2.0

- Update of JSR 77 (J2EE Management)
  - J2EE management is based on EJB2 remoting and CORBA
- Define RESTful API to augment and/or replace current Management EJB APIs
  - Currently used OBJECT_NAME to become URL
  - Define CRUD operations for individual managed object
  - Server-sent events used for event support
- Define RESTful API for application deployment
  - Focus on simple cases first

Java EE Security 1.0

Password aliasing

- Standardized syntax for password aliases
  - Avoids storing passwords in clear text in code, deployment descriptors and files

```
@DataSourceDefinition(
        name="java:app/MyDataSource",
        className="com.example.MyDataSource",
        ...
        user="duke",
        password="${ALIAS=dukePassword}")
```

- Standardized secure credentials archive for bundling alias and password with the application
  - Used by platform as credential store for resolving alias

CDI Interceptor for Authorization

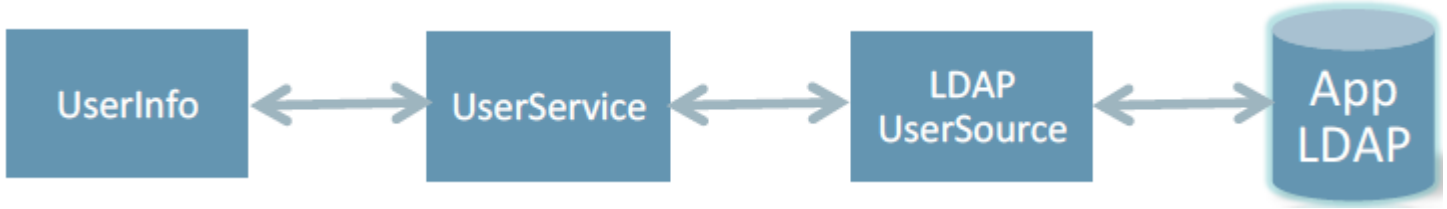
EL eanbled annotations

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

@IsAuthorized("hasRoles('Manager') && hasAttribute('directReports', employee.id)")
double getSalary(long employeeId);

@IsAuthorized(ruleSourceName="java:app/payrollAuthRules", rule="report")
void displayReport();
```

Java EE Security 1.0

User Management

- Allow application to manage its own user and groups
  - Without need to access server configuration
- Users stored in application-specified repository (e.g., LDAP)
  - Simple security providers
- User service manipulates users from user source



Java EE Security 1.0

Simple security providers

```
@EmbedddedSecurityProvider({
    @Credentials(username="reza", password="secret", roles="dad"),
    @Credentials(username="nicole", password="secret", roles="mom"),
    @Credentials(username="zehra", password="secret", roles="daughter"),
    @Credentials(username="xavier", password="secret", roles="son")})

@DatabaseSecurityProvider(
    lookup="java:global/MyDB",
    userQuery="SELECT password FROM principals WHERE username=?",
    rolesQuery="SELECT role FROM roles where username=?", ...)

@LdapSecurityProvider(url="...", dnPrefix="...", dnSuffix="...", ...)
```

Adopting Java SE 8

- Most of Java SE 8 can already be used with Java EE
  - GlassFish, WildFly and WebLogic support JDK 8
- Some APIs could adopt features
  - Repeatable Annotations
  - Date-Time API/JDBC 4.2
  - CompletableFuture
  - Lambda expressions, streams
  - Default methods

Java EE 8 JSR Status

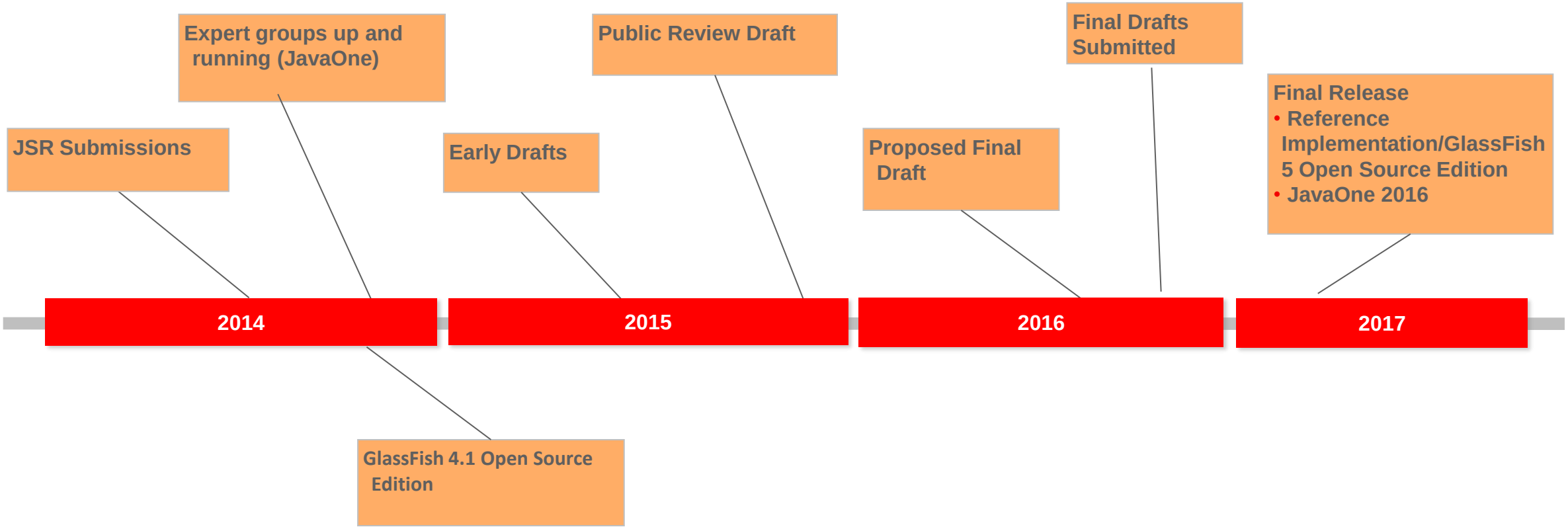
Submitted

- Java EE 8 Platform
- CDI 2.0
- JSON Binding 1.0
- JAX-RS 2.1
- MVC 1.0
- Java Servlet 4.0
- JSF 2.3
- JMS 2.1
- JSON-P 1.1
- Java EE Security 1.0
- Java EE Management 2.0

More To Come

- JCache
- Concurrency Utilities
- WebSockets
- JPA
- And more to follow…

Java EE 8 Roadmap



Continue to participate

Java EE 7

14 adopted JSRs

19 Java User Groups

Thank You!

Java EE 8

New JSRs

New Opportunities

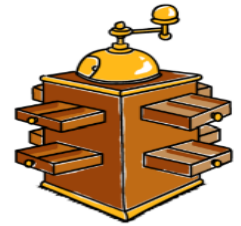
Get Involved!

Participate with us!

http://glassfish.org/contribute

Adopt-a-JSR for Java EE 8

- Grassroots participation to shape Java EE
- Launched in Java EE 7 time-frame, key community element for Java EE 8
  - 30 Java user groups participating!



https://java.net/projects/adoptajsr

Java EE 8 JSRs Already Adopted!

User Group	Java EE 8.0	CDI 2.0	JSON-B 1.0	JMS 2.1	Servlet 4.0	JAX-RS 2.1	MVC 1.0	JSF 2.3
London Java Community	✓	✓	✓	✓		✓	✓	
Morocco JUG		✓	✓		✓	✓	✓	
Egypt JUG	✓		✓	✓			✓	✓
Hellenic Java User Group	✓						✓	
Santa Catarina Java User Group		✓			✓	✓	✓	
Japan Java User Group							✓	

Links

- Java EE Tutorials
  - http://docs.oracle.com/javaee/7/tutorial/doc/home.htm
- Java EE Transparent Expert Groups
  - http://javaee-spec.java.net
- Adopt a JSR
  - https://java.net/projects/adoptajsr
- The Aquarium Blog
  - http://blogs.oracle.com/theaquarium
- Java EE 8 reference implementation
  - http://glassfish.org

Vielen Dank!

Peter Doschkinow
ORACLE Deutschland B.V. & Co. KG