31.8.–3.9.2015
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Das Aquarium lässt grüßen

Die Oracle GlassFish-Strategie und wie es mit Java EE 8 weitergeht

Wolfgang Weigend

ORACLE Deutschland B.V. & Co. KG

ORACLE

Das Aquarium lässt grüßen

Die Oracle GlassFish Strategie und wie es mit Java EE 8 weitergeht

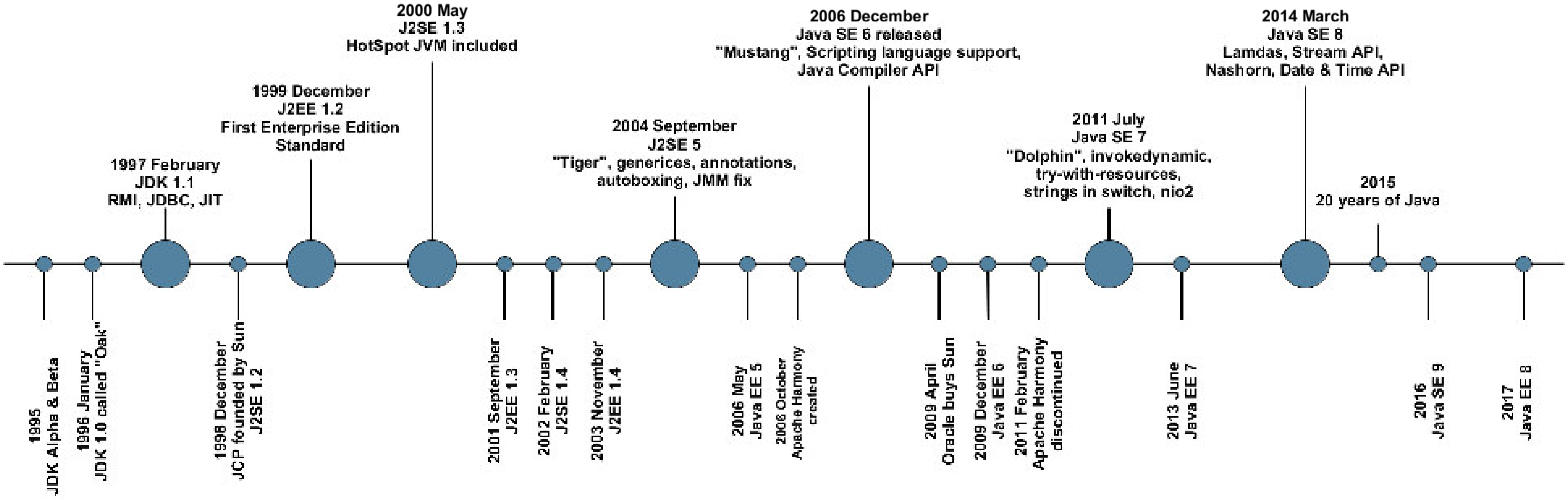
Wolfgang Weigend
Systemberater
Java Technologie & Architektur

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

- Wie sind wir hierher gekommen?
- Wohin gehen wir?
- Wie kann man sich beteiligen?



1995
JDK Alpha & Beta

1996 January
JDK 1.0 called "Oak"

1997 February
JDK 1.1
RMI, JDBC, JIT

1998 December
JCP founded by Sun
J2SE 1.2

1999 December
J2EE 1.2
First Enterprise Edition
Standard

2000 May
J2SE 1.3
HotSpot JVM included

2001 September
J2EE 1.3

2002 February
J2SE 1.4

2003 November
J2EE 1.4

2004 September
J2SE 5
"Tiger", generices, annotations,
autoboxing, JMM fix

2006 May
Java EE 5

2006 October
Apache Harmony
created

2006 December
Java SE 6 released
"Mustang", Scripting language support,
Java Compiler API

2009 April
Oracle buys Sun

2009 December
Java EE 6

2011 February
Apache Harmony
discontinued

2011 July
Java SE 7
"Dolphin", invokedynamic,
try-with-resources,
strings in switch, nio2

2013 June
Java EE 7

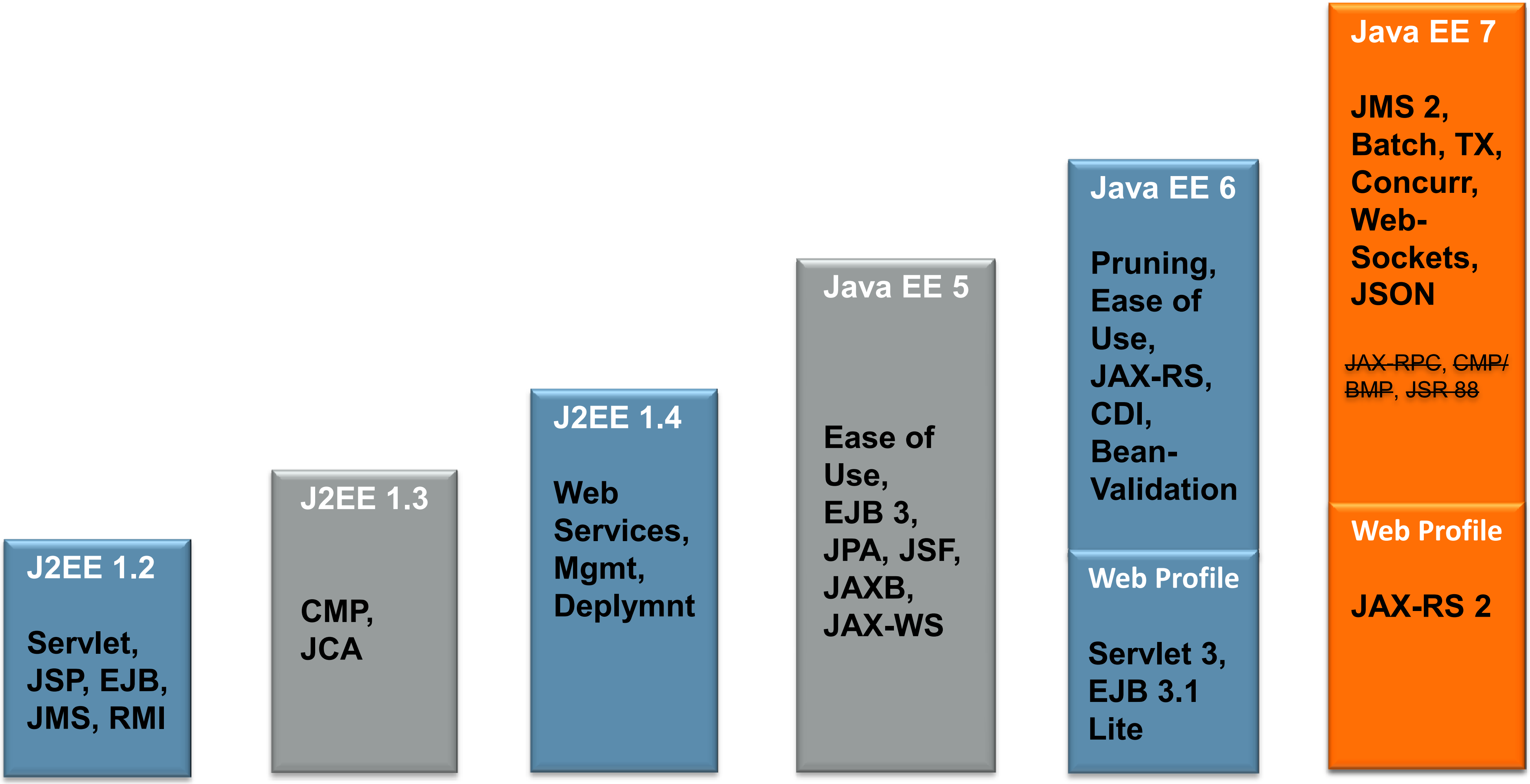
2014 March
Java SE 8
Lamdas, Stream API,
Nashorn, Date & Time API

2015
20 years of Java

2016
Java SE 9

2017
Java EE 8

Java EE Past, Present, & Future



J2EE 1.2

Servlet, JSP, EJB, JMS, RMI

J2EE 1.3

CMP, JCA

J2EE 1.4

Web Services, Mgmt, Deplymnt

Java EE 5

Ease of Use, EJB 3, JPA, JSF, JAXB, JAX-WS

Java EE 6

Pruning, Ease of Use, JAX-RS, CDI, Bean-Validation

Web Profile

Servlet 3, EJB 3.1 Lite

Java EE 7

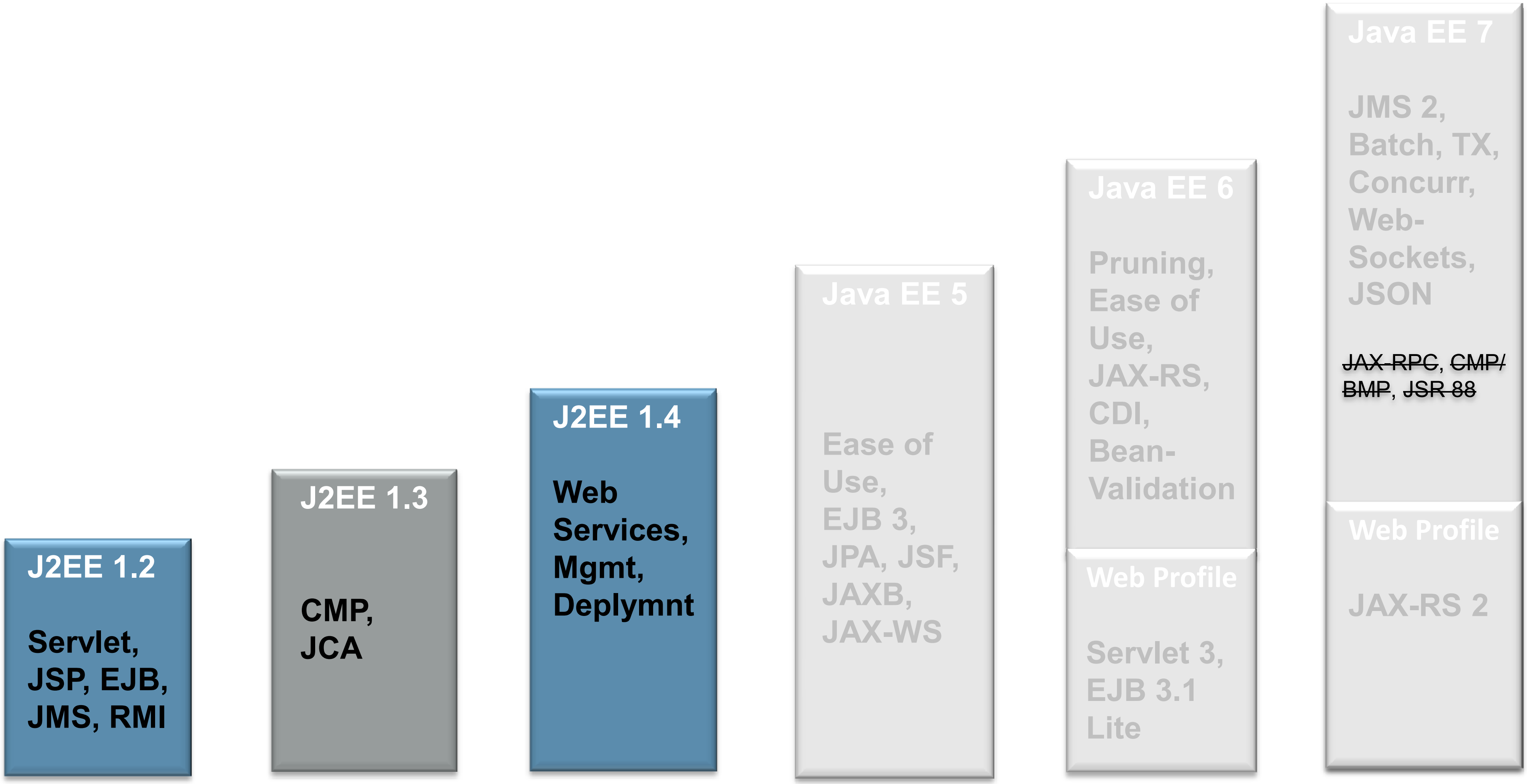
JMS 2, Batch, TX, Concurr, Web-Sockets, JSON

~~JAX-RPC~~, ~~CMP/ BMP~~, ~~JSR 88~~

Web Profile

JAX-RS 2

Java EE Past, Present, & Future



J2EE 1.2

Servlet, JSP, EJB, JMS, RMI

J2EE 1.3

CMP, JCA

J2EE 1.4

Web Services, Mgmt, Deplymnt

Java EE 5

Ease of Use, EJB 3, JPA, JSF, JAXB, JAX-WS

Java EE 6

Pruning, Ease of Use, JAX-RS, CDI, Bean-Validation

Web Profile

Servlet 3, EJB 3.1 Lite

Java EE 7

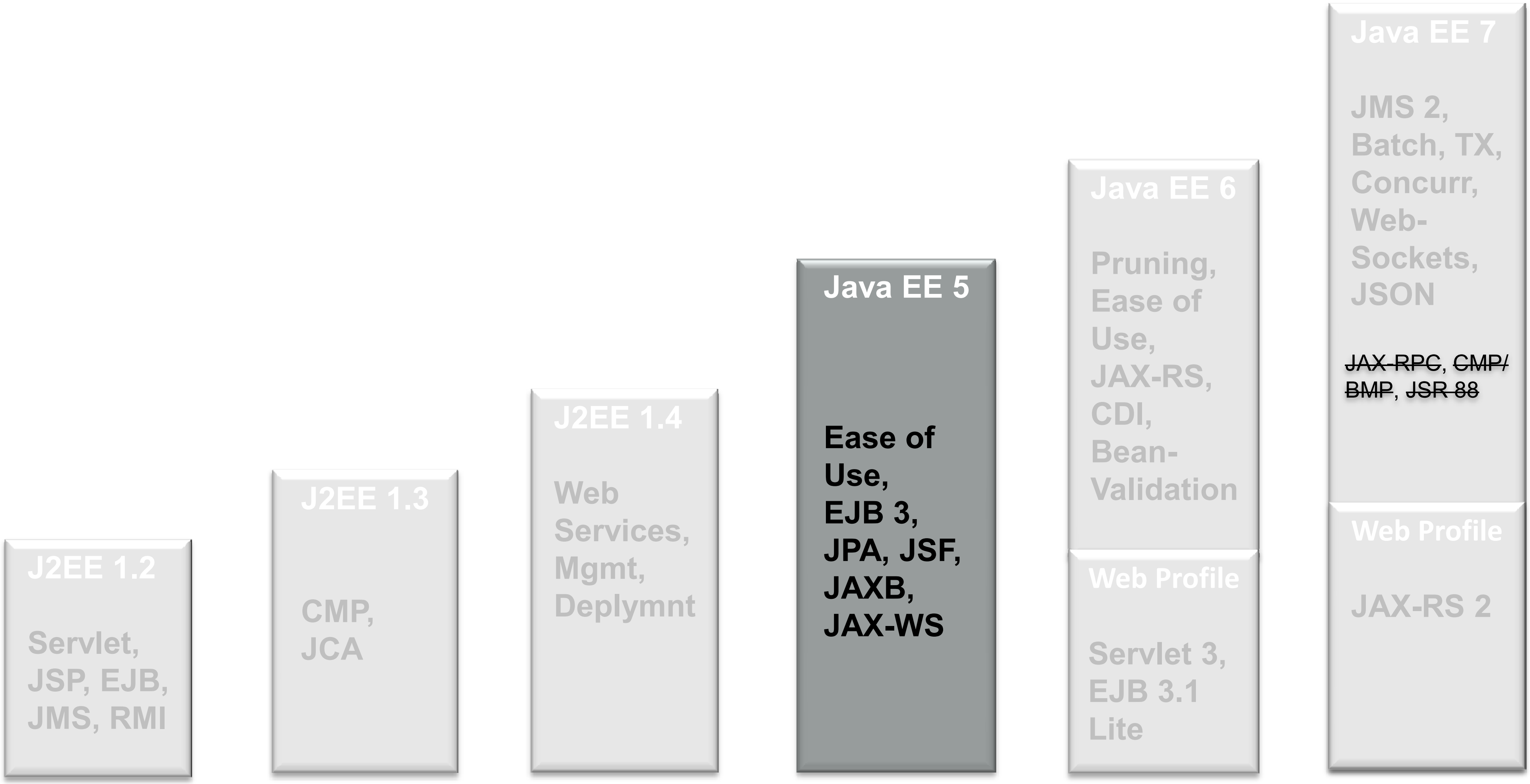
JMS 2, Batch, TX, Concurr, Web-Sockets, JSON

~~JAX-RPC~~, ~~CMP/BMP~~, ~~JSR 88~~

Web Profile

JAX-RS 2

Java EE Past, Present, & Future



J2EE 1.2

Servlet, JSP, EJB, JMS, RMI

J2EE 1.3

CMP, JCA

J2EE 1.4

Web Services, Mgmt, Deplymnt

Java EE 5

Ease of Use, EJB 3, JPA, JSF, JAXB, JAX-WS

Java EE 6

Pruning, Ease of Use, JAX-RS, CDI, Bean-Validation

Web Profile

Servlet 3, EJB 3.1 Lite

Java EE 7

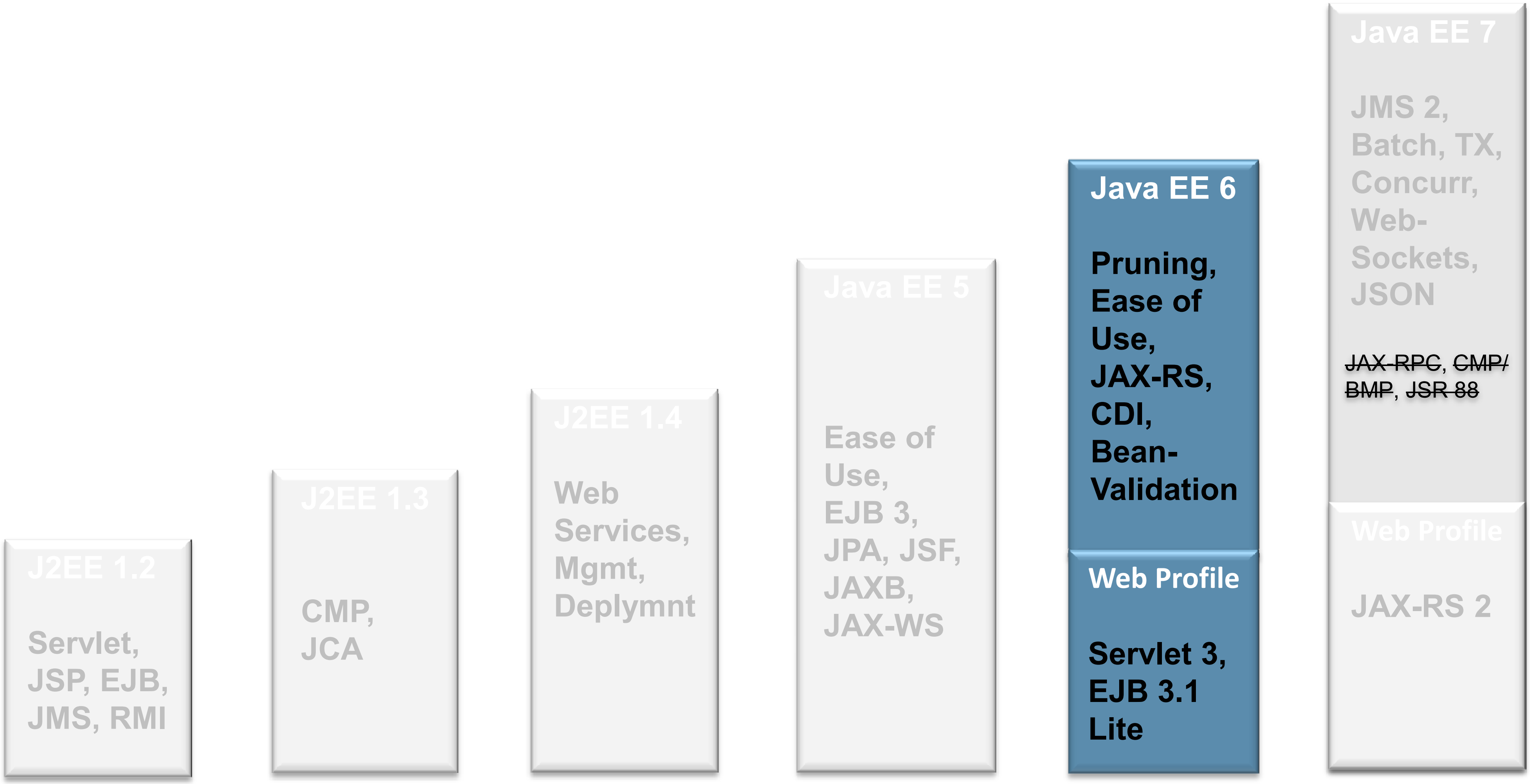
JMS 2, Batch, TX, Concurr, Web-Sockets, JSON

~~JAX-RPC~~, ~~CMP/BMP~~, ~~JSR 88~~

Web Profile

JAX-RS 2

Java EE Past, Present, & Future



J2EE 1.2

Servlet, JSP, EJB, JMS, RMI

J2EE 1.3

CMP, JCA

J2EE 1.4

Web Services, Mgmt, Deplymnt

Java EE 5

Ease of Use, EJB 3, JPA, JSF, JAXB, JAX-WS

Java EE 6

Pruning, Ease of Use, JAX-RS, CDI, Bean-Validation

Web Profile

Servlet 3, EJB 3.1 Lite

Java EE 7

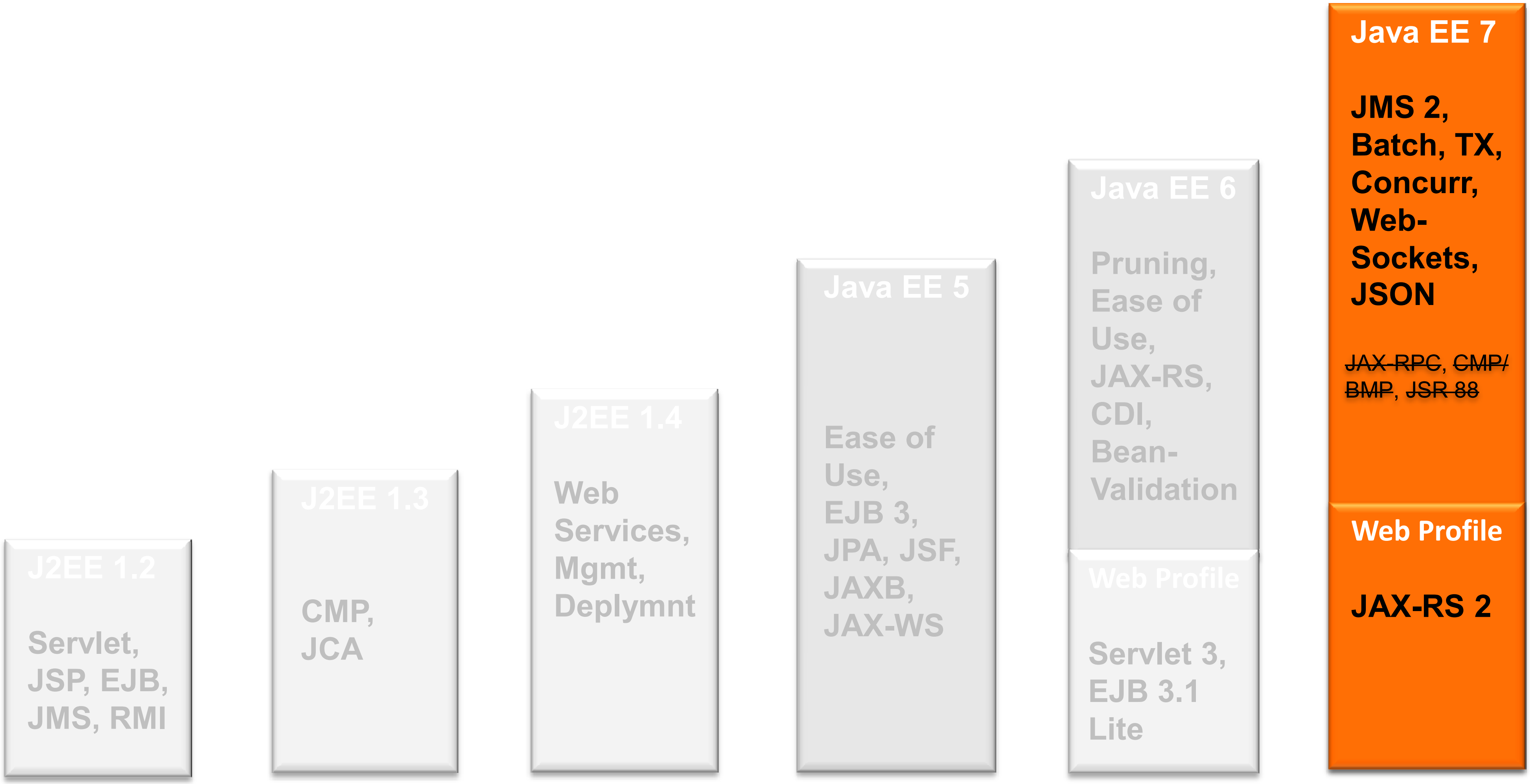
JMS 2, Batch, TX, Concurr, Web-Sockets, JSON

~~JAX-RPC~~, ~~CMP/ BMP~~, ~~JSR 88~~

Web Profile

JAX-RS 2

Java EE Past, Present, & Future



J2EE 1.2

Servlet, JSP, EJB, JMS, RMI

J2EE 1.3

CMP, JCA

J2EE 1.4

Web Services, Mgmt, Deplymnt

Java EE 5

Ease of Use, EJB 3, JPA, JSF, JAXB, JAX-WS

Java EE 6

Pruning, Ease of Use, JAX-RS, CDI, Bean-Validation

Web Profile

Servlet 3, EJB 3.1 Lite

Java EE 7

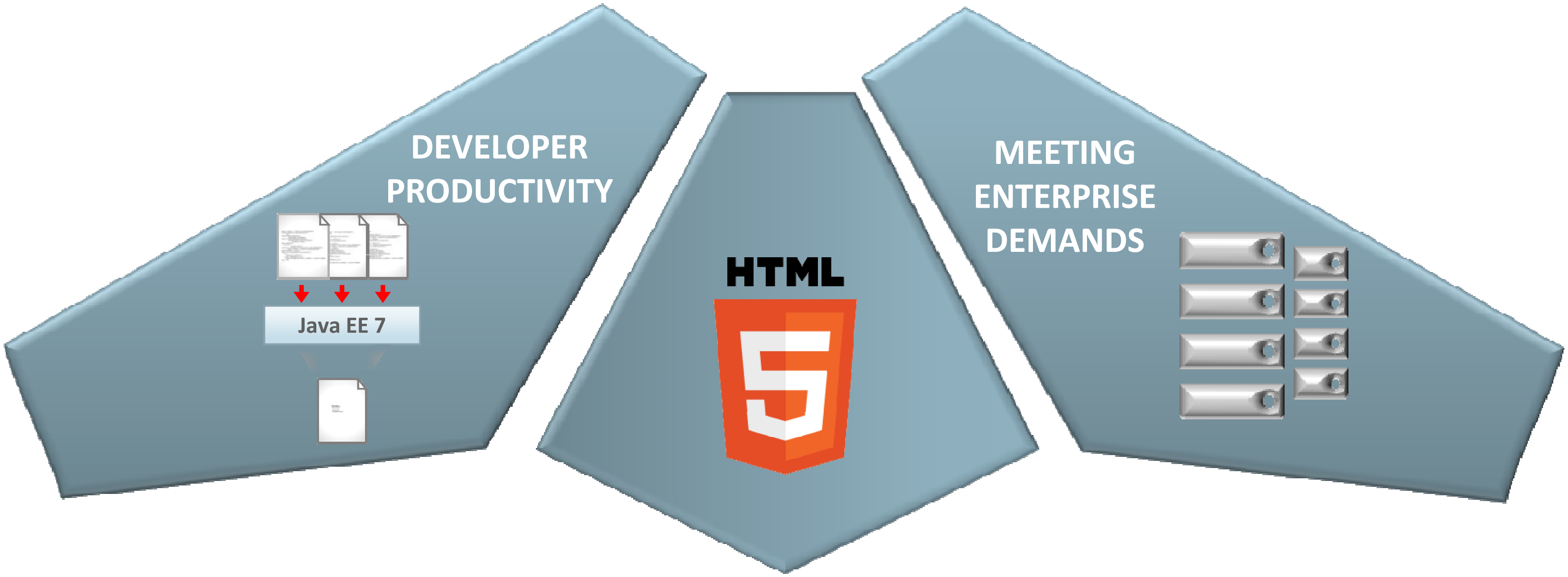
JMS 2, Batch, TX, Concurr, Web-Sockets, JSON

~~JAX-RPC~~, ~~CMP/BMP~~, ~~JSR 88~~

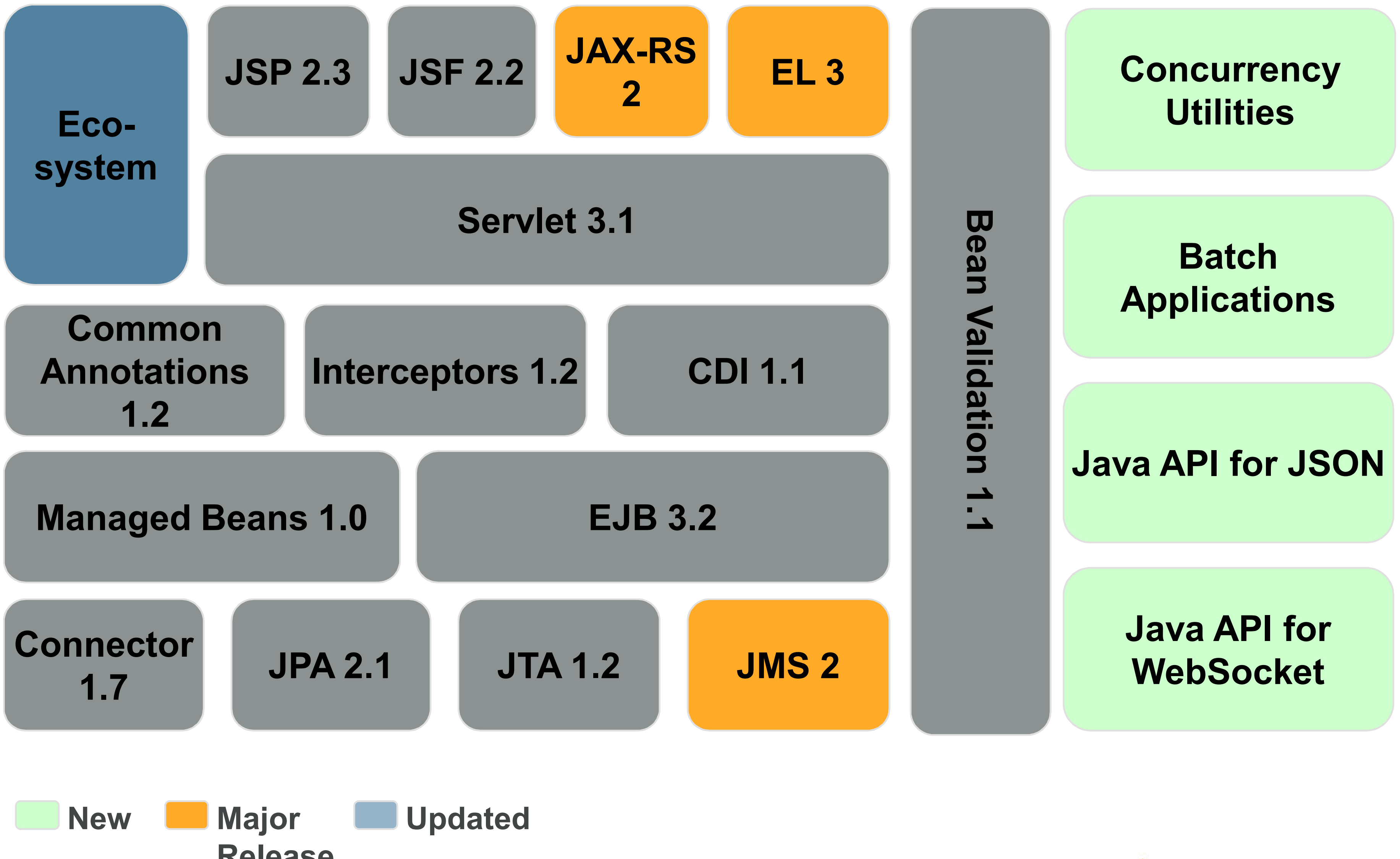
Web Profile

JAX-RS 2

Java EE 7 Themen



Java EE 7



Eco-system

JSP 2.3

JSF 2.2

JAX-RS 2

EL 3

Servlet 3.1

Bean Validation 1.1

Concurrency Utilities

Batch Applications

Common Annotations 1.2

Interceptors 1.2

CDI 1.1

Java API for JSON

Managed Beans 1.0

EJB 3.2

Connector 1.7

JPA 2.1

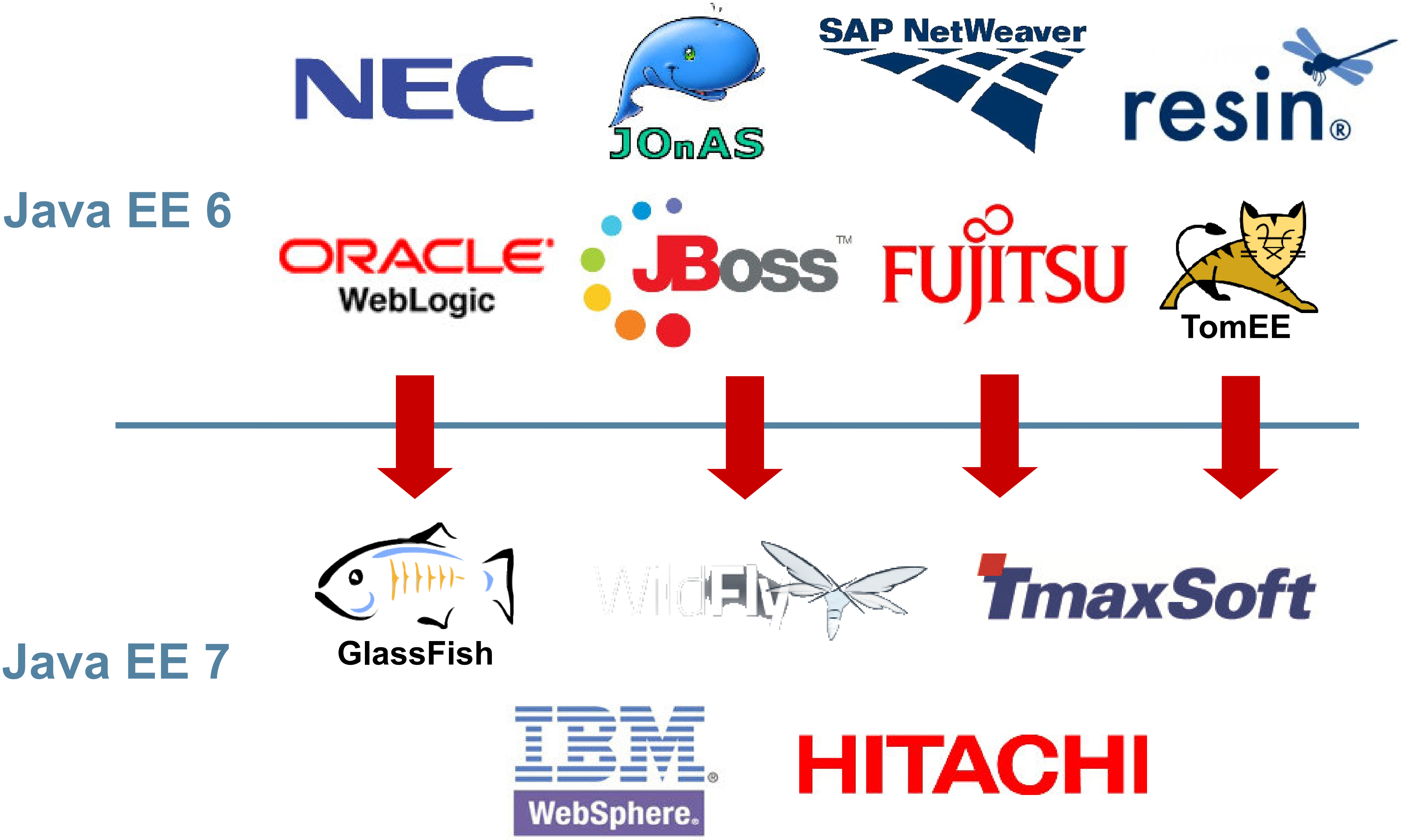
JTA 1.2

JMS 2

Java API for WebSocket

🟩 New 🟧 Major Release 🟦 Updated

Java EE 7 Plattformen



Java EE 6

Java EE 7

GlassFish Produkt-Distributionen

- Java EE Reference Implementation (RI)
  - Java Community Process requires specification lead to deliver a licensable implementation. Many licensees; 20+ Java EE implementations
  - Java EE RI is a substantial subset of GlassFish
- Java EE SDK
  - Tutorial, samples and documentation for developers learning Java EE
- GlassFish Server Open Source Edition
  - Free open source, unsupported server deployment of Java EE applications
- Oracle GlassFish Server
  - Commercially supported deployment platform for Java EE / GlassFish

GlassFish Server Produktstrategie

Java EE SDK und Java EE RI – Partner, Java EE Licensees primär

- Java EE RI
  - Java Community Process requires specification lead to deliver a licensable implementation
  - Released with each major update of the Java EE standard
  - Plan: Continue with future releases of Java EE
- Java EE SDK
  - Tutorials, samples and documentation for developers learning Java EE
  - Released with each major update of the Java EE standard
  - Plan: Continue with future releases of Java EE

GlassFish Server Produktstrategie

GlassFish Server Open Source Edition

- GlassFish Server Open Source Edition
  - Developed in open source
  - Delivered in open source
  - Not commercially supported
- GlassFish Server Open Source Edition ongoing release plans
  - **GlassFish Server Open Source Edition 4.1**
  - Additional updates as Java EE specification evolves
  - Plan: Regular patch updates will be delivered as needed
- GlassFish Server Open Source Edition Major Release
  - Delivered coincident with new versions of Java EE Platform

GlassFish 4.1 und WebLogic 12.1.3

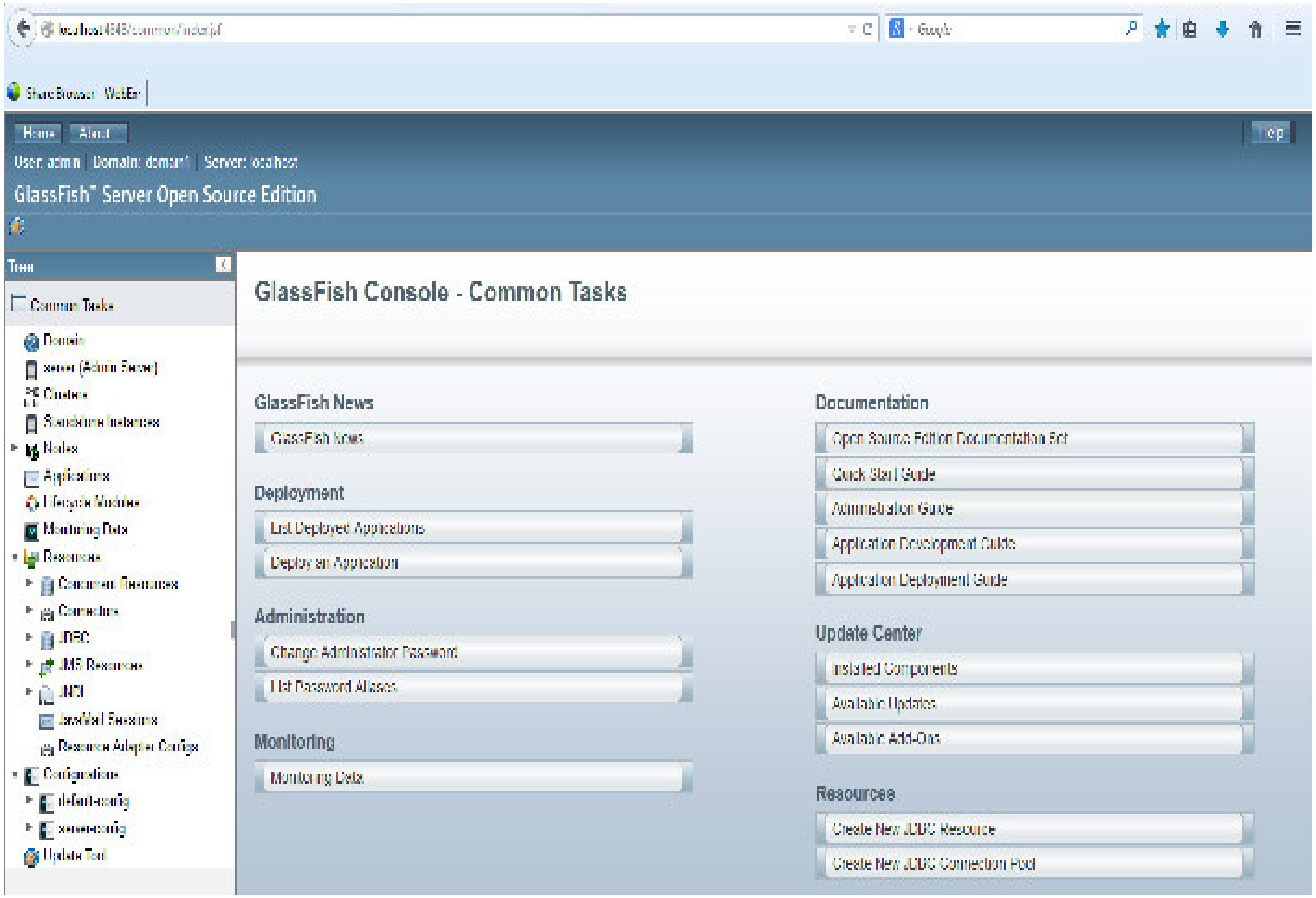
- Glassfish 4.1
  - Addressing security, must-fix and important bugs
  - Over 800+ bugs fixed!
  - JDK 8 Support
  - Updates to projects: Jersey, Mojarra, WebSocket 1.1, Tyrus, Grizzly, CDI 1.2, Weld 2.2
  - NetBeans includes GlassFish 4.1
- WebLogic 12.1.3
  - WebSocket, JSON-P, JAX-RS 2, JPA 2.1
  - JDK 8 support
  - Server-Sent Events (SSE)
  - WebSocket fallback
  - Improved Maven support
  - Official Docker images

GlassFish 4.1 Versionen

- Support for Java SE 8
- Incorporate bugs fixes addressed since GF 4.0, see the list of fixed bugs (recent fixes might not yet be incorporated into the Nightly Build)
- Update embedded sub-projects like Jersey, Tyrus, Weld, Mojarra, JavaDB,etc.
- NetBeans 8.x alignment
- **Download GlassFish 4.1**
  - Java EE 7 Web Profile ***`glassfish-4.1-web.zip`*** (57 MB)
    - http://download.java.net/glassfish/4.1/release/glassfish-4.1-web.zip
  - Full Java EE 7 Platform ***`glassfish-4.1.zip`*** (103 MB)
    - http://download.oracle.com/glassfish/4.1/release/glassfish-4.1.zip
  - **GlassFish 4.1 Nightly Build**
    - http://download.java.net/glassfish/4.1/
    - http://download.oracle.com/glassfish/4.1/nightly/index.html
- **The Aquarium - News from the GlassFish Community**
  - https://blogs.oracle.com/theaquarium/

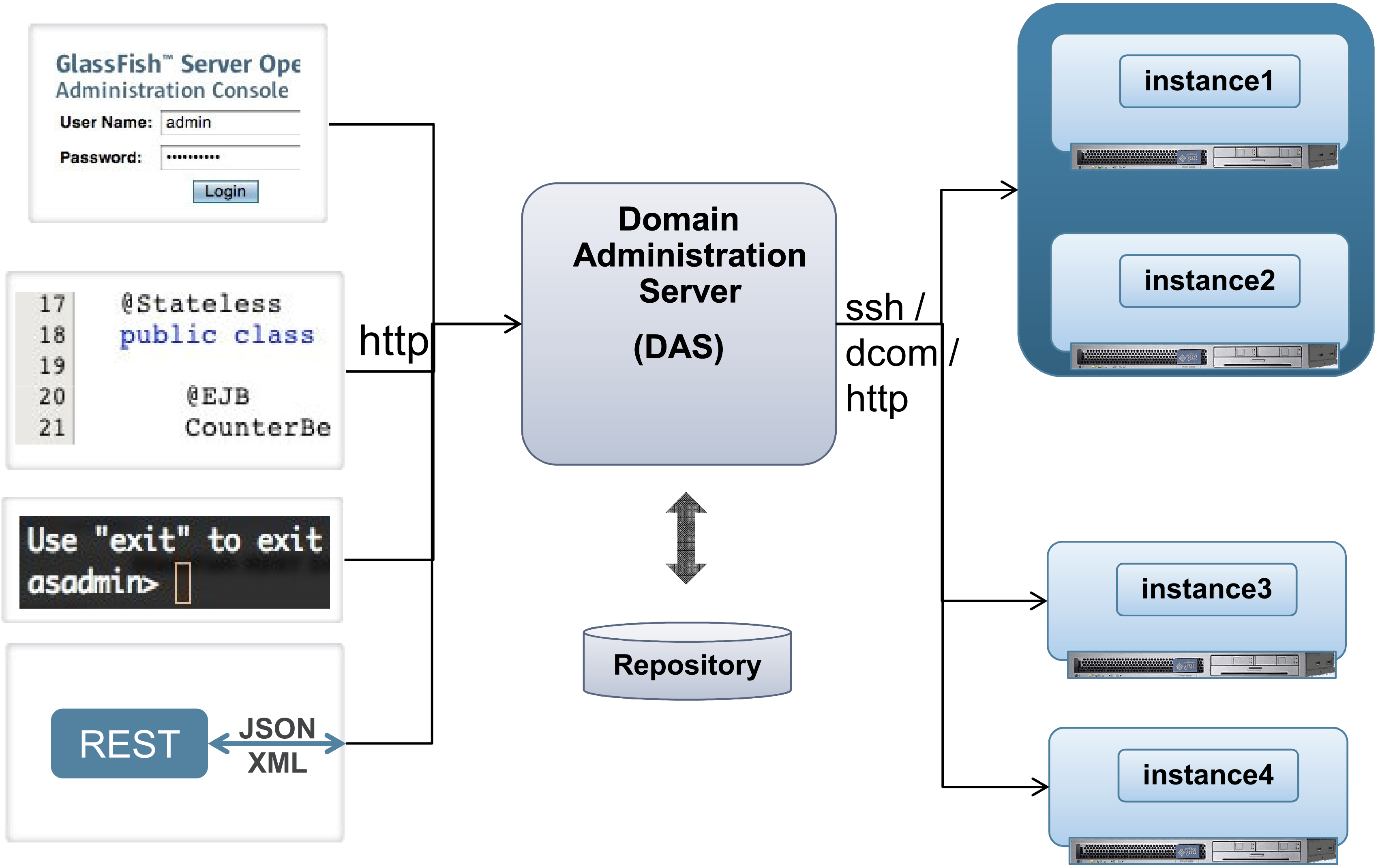
- **Clustering**
- **Improve modularization**
- **GlassFish Server Control**
- **Oracle GlassFish Server**
- **Bug fixes**

GlassFish Server Open Source Edition 4.1



- Built in open source
- Lightweight / modular / easy to use

GlassFish Server Administration Architecture



GlassFish™ Server Ope
Administration Console

User Name: admin

Password: ••••••••••

Login

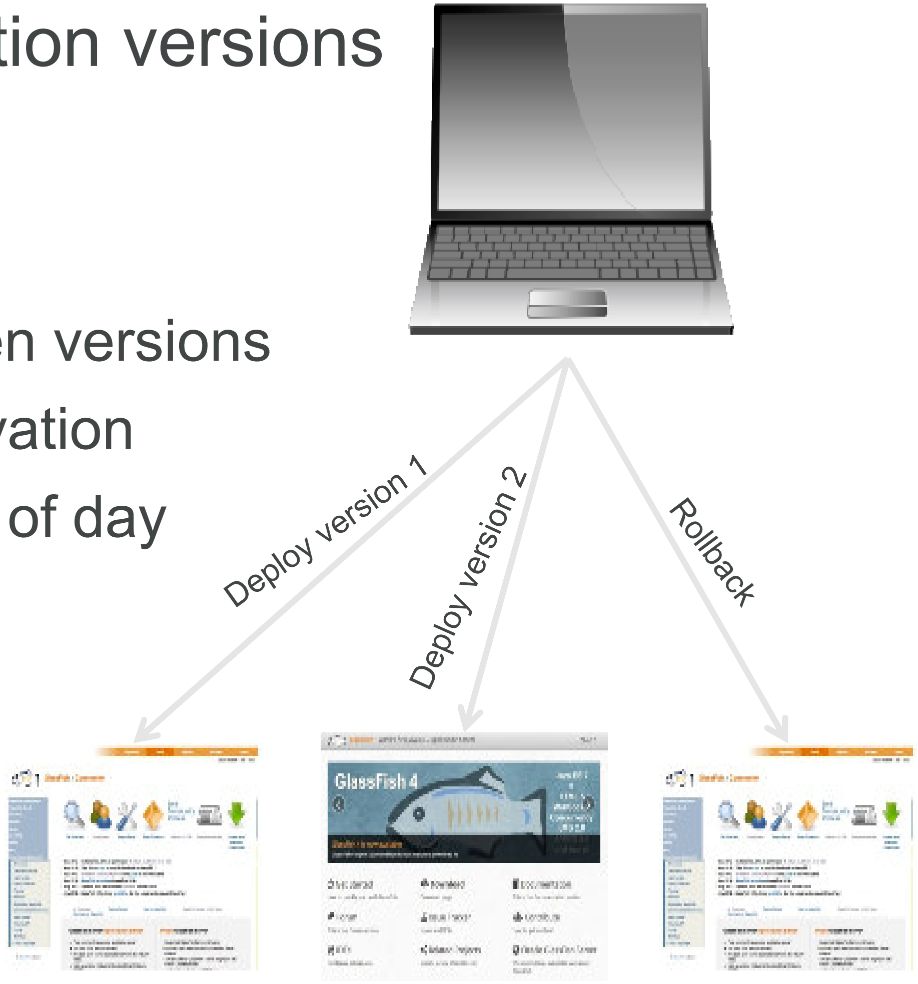
```
    @Stateless
    public class

        @EJB
        CounterBe
```

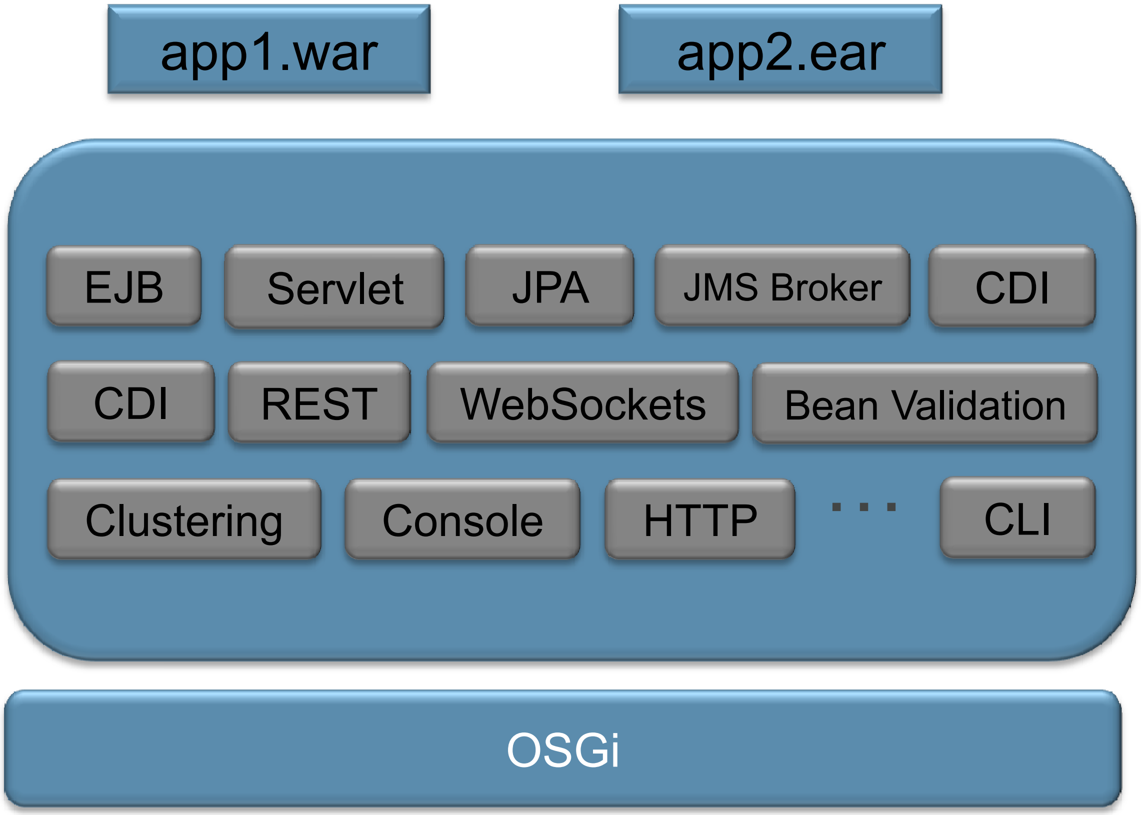
```
Use "exit" to exit
asadmin>
```

Application Versioning

- Deploy multiple versions application versions
- Activate any version
- For example:
  - Roll forward and backwards between versions
  - Pre-deploy application for later activation
  - Enable new version at specific time of day

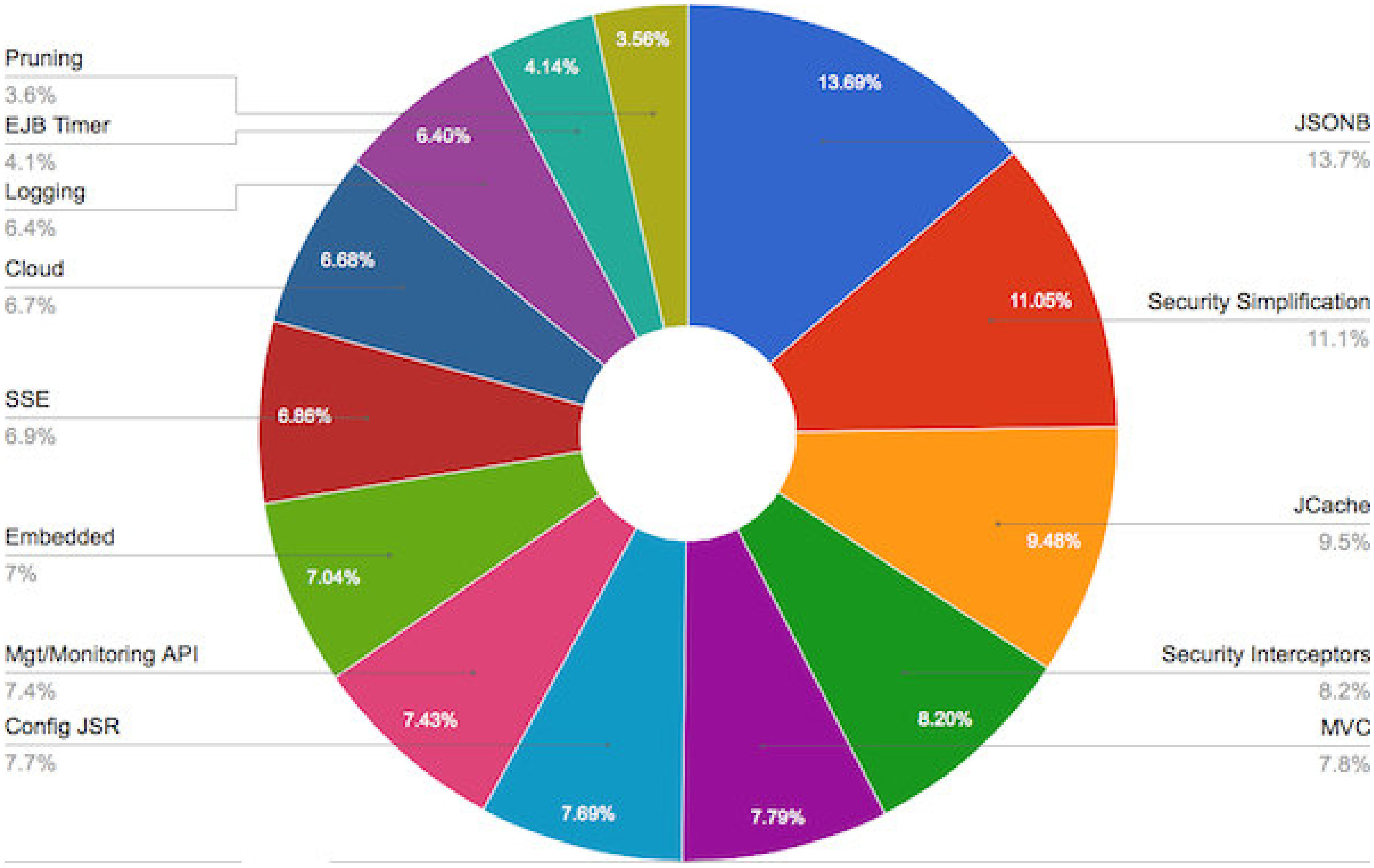


GlassFish Modularity



- Starts in seconds
- Only loads required modules
- Including “infrastructure” features

Java EE 8 Community Umfrage



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 - Möglichkeiten

- Web Standards/HTML5 Alignment
  - HTTP2, SSE, JSON-B, JSON-P, action-oriented web framework, hypermedia
- Cloud
  - Simple security providers, REST management/monitoring
- CDI Alignment
  - CDI 2, EJB services outside EJB, EJB pruning
- Enterprise
  - JCache, Configuration, JMS
- Java SE 8 alignment

Java EE 8 API Status

Been There

- Java EE 8 (JSR 366)
- CDI 2 (JSR 365)
- JSON-B (JSR 367)
- JMS 2.1 (JSR 368)
- Servlet 4 (JSR 369)
- JAX-RS 2.1 (JSR 370)
- MVC (JSR 371)
- JSF 2.3 (JSR 372)

Newer

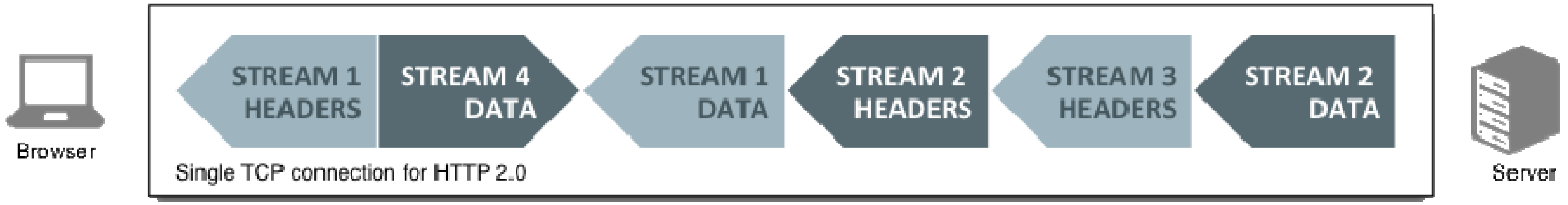
- Java EE Security (JSR 373)
- Java EE Management (JSR 375)
- JSON-P 1.1 (JSR 374)

Still to Come

- Concurrency Utilities
- WebSocket
- JPA
- And more…

Servlet 4.0

- Principal goal to support HTTP 2
  - Request/response multiplexing over single connection
  - Multiple streams
  - Stream Prioritization
  - Server Push
  - Binary Framing
  - Header Compression



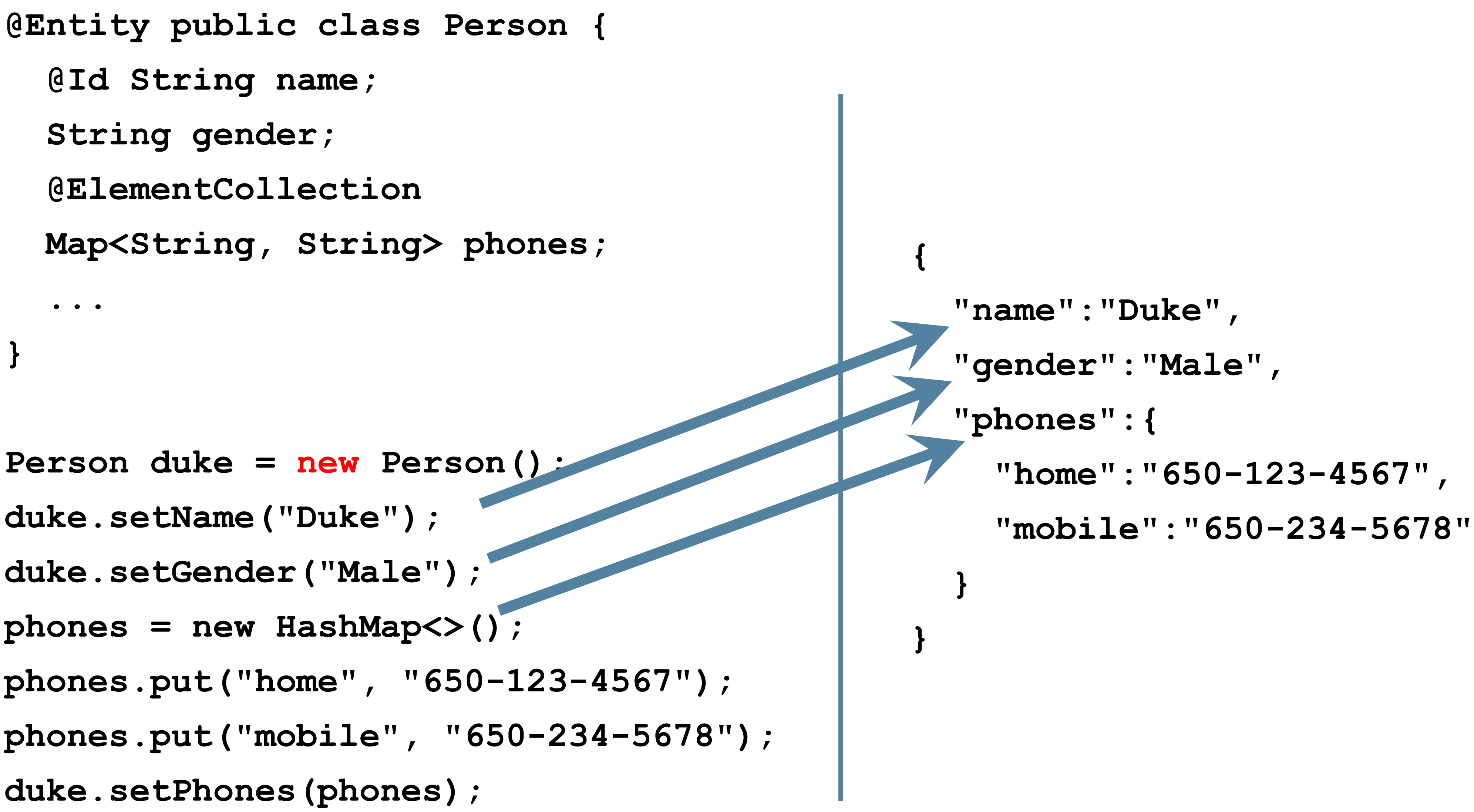
- Hopefully most of it can be done without major API changes

JSON-B

Java API for JSON Binding

- API to marshal/unmarshal POJOs to/from JSON
  - Very similar to JAXB in the XML world
- Default mapping of classes to JSON
  - Annotations to customize the default mappings
  - @JsonProperty, @JsonTransient, @JsonValue
- Draw from best of breed ideas in existing JSON binding solutions
  - MOXy, Jackson, GSON, Genson, Xstream, …
  - Allow switching providers
- Provide JAX-RS a standard way to support “application/json” for POJOs
  - JAX-RS currently supports JSON-P

JSON-B - Möglichkeiten



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

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

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

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 - Möglichkeiten

Search & reference parts of a JSON document using URL like paths

```
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

- 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 - Möglichkeiten

Replace as updated persons mobile record;
Remove entire person record from JSON document

```
[
  {
    "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());
```

Server-Sent Events (SSE)

- Lesser known 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
- SSE on the Client-Side

MVC

- Standard action-based web framework for Java EE
  - First class peer to JSF, JSF to continue on it’s evolution path
- Model
  - CDI, Bean Validation, JPA
- View
  - Facelets, JSP
- Controller
  - Majority of work here
  - Based on JAX-RS

Java EE Security

- Simplify security for Java EE and improve portability
- Simple security providers
- Simple pluggability and mappings
- Enabling existing security annotations (@RolesAllowed) for all beans
- EL enabled security annotations via interceptors

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="...", ...)
```

EL enabled Security Annotationen

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

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

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

Simple Security Pluggability

```
@SecurityProvider
public class MySecurityProvider {

  @Inject UserService userService;

  @OnAuthentication
  // The parameters should suit the credentials mechanism being used
  public Principal getPrincipal(
      String username, String password) {
    // Construct the principal using the user service
  }

  @OnAuthorization
  public String[] getRoles (Principal principal) {
    // Construct an array of roles using the principal and user service
  }
}
```

JMS 2.1

- Essentially continuation of JMS 2
- Declarative message listeners
  - Alternative to MDB
  - More powerful features
  - Available to all beans
- Improving JMS provider portability
- Minor features and corrections
  - Redelivery delay, redelivery limits, dead message queues

Declarative JMS Listeners

```
@ApplicationScoped
@MaxConcurrency(10)
public class HandlingEventRegistrationAttemptConsumer {
  @JmsListener(
     destinationLookup="jms/HandlingEventRegistrationAttemptQueue",
     selector="source = 'mobile'",
     batchSize=10, retry=5, retryDelay=7000,
     orderBy=TIMESTAMP)
  @Transactional
  public void onEventRegistrationAttempt(
      HandlingEventRegistrationAttempt... attempts) {
    ...
  }
}
```

CDI 2

- Java SE Bootstrap
- XML configuration
- Asynchronous events
- @Startup for CDI beans
- Portable Extension SPI simplification
- Small features and enhancements
- Asynchronous CDI Events?

Asynchronous CDI Events ?

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

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

EJB @Schedule annotation could be made available to all managed beans via CDI – outside EJB

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

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

Java EE Management

- Revamp of dated JSR 77 (J2EE Management)
- REST/SSE instead of EJB 2.x remoting
- Not just management/monitoring but deployment as well

Adopting Java SE 8

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

Weitere Merkmale zur Diskussion für Java EE 8 API’s

- Reactive programming with JAX-RS client API
- More hypermedia support in JAX-RS
- Non-blocking I/O in JAX-RS
- Improving CDI integration with JAX-RS
- Improving CDI integration with WebSocket
- Prune EJB 2 interfaces
- Prune CORBA interoperability

Java EE 8 Roadmap

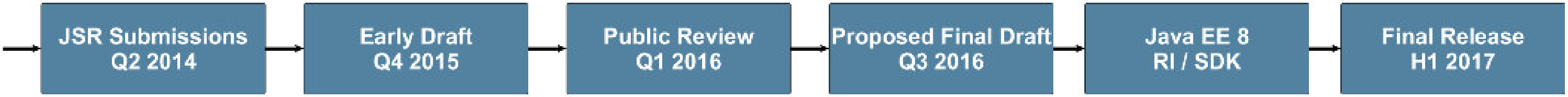
Updates to the original JSR 366

Q4 2015 Early Draft

Q1 2016 Public Review

Q3 2016 Proposed Final Draft

H1 2017 Final Release



Adopt-a-JSR für 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

Grupo de Usuários Java de Campinas e região
Egyptian Java User Group
SOUJava
sociedade de usuários java
Morocco JUG
www.BeJUG.org
LJC
CONGO JUG

http://glassfish.org/adoptajsr

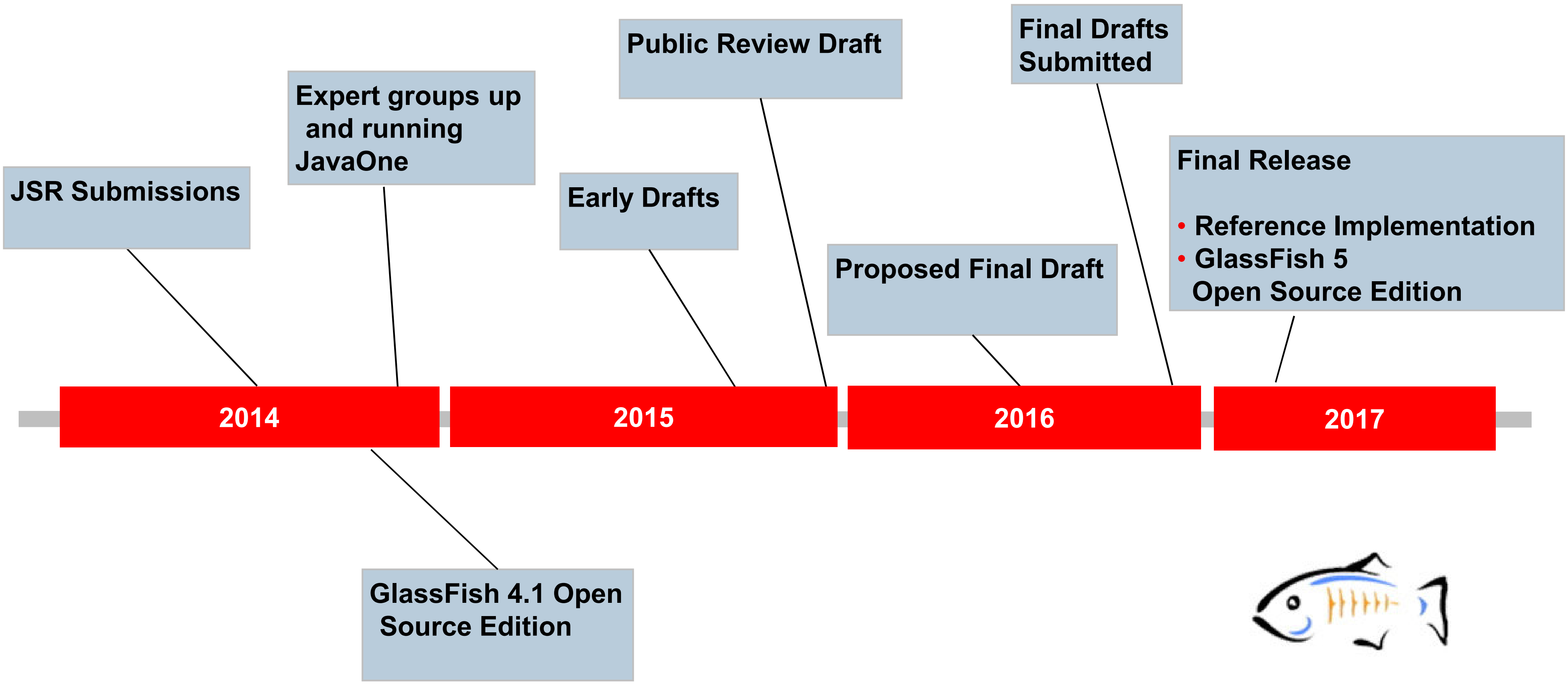
Java EE 8 JSR’s mit JUG Unterstützung

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							✓	

Java EE Commitment

- Oracle committed to the future of Java EE
  - Delivered Java EE 7 in June 2013
- GlassFish Server is the strategic reference implementation of Java EE
- GlassFish distributions for Java EE continue to be regularly updated with major releases of the Java EE specification
  - GlassFish SDK
  - GlassFish RI
  - GlassFish Open Source Edition
- Oracle provides one commercially supported strategic application server – Oracle WebLogic Server

Java EE / GlassFish Roadmap



JSR Submissions

Expert groups up and running JavaOne

GlassFish 4.1 Open Source Edition

Early Drafts

Public Review Draft

Proposed Final Draft

Final Drafts Submitted

Final Release

- Reference Implementation
- GlassFish 5 Open Source Edition

2014

2015

2016

2017

What is Payara?

Payara is 24/7 software support for GlassFish Server Open Source Edition.

Payara Server is a drop in replacement for GlassFish Server with guaranteed quarterly releases. We provide enhancements, bug fixes and patches to upstream GlassFish Server and dependent libraries including Tyrus, Eclipse Link, Jersey and others.

We want to optimise Payara Server, making it the **best server for production Java EE applications** with responsive 24/7 dedicated incident and software support from the best middleware engineers in the industry.

Payara's relationship to GlassFish

- Built from GlassFish 4.1
- Public Open Source Development
  - GitHub
- Quarterly Releases
  - incorporating revisions from upstream projects e.g. Metro, Tyrus, Jersey etc.
- Patches
- Fixes pushed to upstream GlassFish

Payara’s modules that have been upgraded since the last release 4.1.144 and new in Payara Server 4.1.153

- Jersey 2.15
- Tyrus 1.9
- JSTL-Impl 1.2.4
- Mojarra 2.2.9
- Weld 2.2.7
- Javax Batch API 1.0.1-b01
- JBatch Container 1.0.1-b04
- JBatch SPI 1.0.1-b04
- Grizzly 2.3.18
- HK2 2.4.0-b08
- HK2 Plugin 2.4.0-b08
- Jackson 2.5.0
- Jettison 1.3.7
- Shoal 1.6.22
- Metro – 2.3.2-b608
- JAXB-API – 2.2.13-b141020.1521
- JAXB – 2.2.12-b141219.1637

- HTTP and HTTPS Port Auto-Binding
- Payara Micro API
- JCache Injection
- Updated Start-Domain Command
- Payara Blue Server release for the IBM JDK
- Updated Modules and Bug Fixes

Payara’s GitHub issues marked as enhancements that have been implemented for this release

25 – Disable SSLv3 in default domain template
28 – Integrate Hazelcast as JSR107 provider
29 – Integrate Hazelcast as a clustered web session store
80 – Integrate MySQL persistence manager into JBatch
81 – Integrate Oracle persistence manager into JBatch
82 – Integrate PostgreSQL persistence manager into JBatch
83 – Integrate DB2 persistence manager into JBatch
86 – Integrate different persistence managers into GF-batch-connector
111 – Allow Blank Schema for JBatch
149 – Update underlying specification implementations Fixed Issues
45 – Glassfish 21148 - fixed bug in SSO clustered session management
46 – Glassfish 21219
47 – GLASSFISH-21146 Fixed NPE in log
53 – Merge Patched Grizzly jar into build
68 – Merge Fix for GLASSFISH-21251
70 – Merge Fix for GLASSFISH-21007
76 – Upgrade Weld to 2.2.4.Final or newer
79 – Merge fix for GLASSFISH-21261
84 – Fix XForwarded-Proto GLASSFISH-20842
85 - Create Fix for https://java.net/jira/browse/GLASSFISH-21249
101 – Force early creation of static transaction manager to fix GLASSFISH-21175
114 – Fix GLASSFISH-21265
120 – Fix GLASSFISH-20994
122 – JDBC Monitoring MBeans not working in JConsole
127 – Further MBeans are broken GLASSFISH-21276
131 – Java EE 7 Sample Chunk CSV Database Test Fails
141 – Fix GLASSFISH-21017
180 – Fix GLASSFISH-21125

Optimised for production

- Production Tuned Domain Configuration
- Secured by Default
- Focus on maintaining and enhancing production capabilities
- Remove all unused modules

Future production enhancements

- JDBC Enhancements
  - Oracle RAC support
- Extensive Instrumentation
  - New Monitoring Console / Cross Domain
- Performance Tuning
- New Cluster Storage
  - Replace Shoal with Hazelcast
- Enterprise Security Integration

Open Source infrastructure

- Payara is on GitHub
  - Github.com/Payara

- Payara Jenkins for nightly builds
  - Jenkins.payara.co

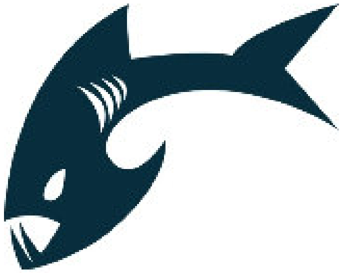
- GitHub for main development branch
  - o/s issues on GitHub
  - Customer cases on *ZenDesk* support

Git strategy

- GlassFish SVN trunk synched to GitHub upstream branch
- Pull Requests from GitHub contributions synched on Master branch
- Contributors must sign Contributor agreement

Why was Payara created?

- GlassFish is a great Java EE 7 Server
  - Modern OSGI microkernel created for 3.x
  - Great administration capabilities
- Oracle invest in Java EE X as the RI
  - Many modules now in/will be in WebLogic (tyrus, metro, jersey)
  - We can concentrate on prod/ops
- Java EE needs another o/s server



Zusammenfassung

- Java EE 8 Themes
  - Support for the latest web standards (HTTP 2.0)
  - Continue to work on ease of development
  - Improve the infrastructure for cloud support
  - Alignment with Java SE 8
- New JSR’s added to the Platform
  - JCache
  - Java API for JSON Binding
  - Java Configuration
- Updated JSR‘s
- Inside upcoming GlassFish Application Server

Danke!

Wolfgang.Weigend@oracle.com

Twitter: @wolflook

More Information

- Java EE Transparent Expert Groups
  - http://javaee-spec.java.net
- Java EE Reference Implementation
  - http://glassfish.org
- The Aquarium
  - http://blogs.oracle.com/theaquarium