From Java EE to Jakarta EE

A user perspective

@jefrajames
A few words about me

Speaker me = SpeakerOf.setLastName("James")
    .setFirstName("Jean-François")
    .setBackgroundColor(Period.ofYears(32))
    .setCompany("Worldline")
    .setLocation("France")
    .addSkill("Software Architecture", Year.parse("1990"))
    .addSkill("Java", Year.parse("1997"));
Agenda

- Retrospective from J2EE to Java EE 7
- Shifting to cloud-native applications
- The long road to Java EE 8
- Summer 2017: the turbulent zone
- Preparing the future
From J2EE to Java EE 7

J2EE Prehistoric times
- J2EE
  - Servlet
  - JSP
  - EJB
  - JMS

Java EE times
- Java EE 5
  - Ease of Dev
  - EJB 3
  - JPA
  - JSF
  - JAX-WS
- Java EE 6
  - CDI
  - JAX-RS
  - Bean Validation
  - Web Profile
- Java EE 7
  - JBatch
  - Web Socket
  - JSON-P
  - Concurrency

Innovation vs standardization

“Java EE is the place to standardize innovation in the Enterprise Java space”
Shifting to cloud-native applications
# Shifting to cloud-native applications

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Private on premise data centers</td>
<td>- Cloud platforms</td>
</tr>
<tr>
<td>- Bare metal infrastructure</td>
<td>- Infrastructure as code</td>
</tr>
<tr>
<td>- Mutualized full-blown application servers</td>
<td>- Just enough runtime</td>
</tr>
<tr>
<td>- Monolith</td>
<td>- Microservices</td>
</tr>
<tr>
<td>- SQL-only</td>
<td>- Polyglot persistency</td>
</tr>
<tr>
<td>- ACID transactions</td>
<td>- Eventual consistency</td>
</tr>
<tr>
<td>- Synchronous &amp; imperative programming</td>
<td>- Asynchronous &amp; functional programming</td>
</tr>
<tr>
<td>- Standalone applications</td>
<td>- API-first platforms</td>
</tr>
</tbody>
</table>

- SQL-only
- ACID transactions
- Synchronous & imperative programming
- Standalone applications
The long road to Java EE 8

2014, 2015: the empty years
The long road to Java EE 8
2016: the year of rebirth
The long road to Java EE 8

21 Sept. 2017: Welcome to Java EE 8!

Java EE 8
Java SE 8
Security 1.0
JSON-B 1.0
Servlet 4.0
CDI 2.0
JPA 2.2
Bean Val. 2.0
JSON-P 1.1
JAX-RS 2.1

Glassfish 5
Java EE 8: end of the story
2017: entering the turbulent zone

- Decision to open up Java EE
- Eclipse Foundation selected
- EE4J project

Timeline:
- 17/08: Decision to open up Java EE
- 07/09: New Java SE Cadence release
- 12/09: Eclipse Foundation selected
- 21/09: Java SE 9 & Java EE 8
- 29/09: EE4J project

Text:
- Java SE 9 & Java EE 8
2017: leaving the turbulent zone
Replacing the JCP?

The mechanism for developing standard technical specifications for Java technology

A global community of individuals and organizations with a mature, scalable, and commercially focused environment for collaboration and innovation
Eclipse foundation is more than Eclipse IDE!

- 350+ Projects
- 275+ Members
- 1550+ Committers
- 30 Professional Staff

- Internet of Things
- Java Runtimes
- GeoSpatial
- Automotive
- Model Based Engineering
- IDEs
- Emerging Technologies

Vendor neutrality
## Focus on 5 Eclipse projects

<table>
<thead>
<tr>
<th>Project</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE4J</td>
<td>Top-level project to host the future of Java EE</td>
</tr>
<tr>
<td>MICROPROFILE</td>
<td>Open Specification for Enterprise Java Microservices</td>
</tr>
<tr>
<td>OpenJ9</td>
<td>Optimized JVM to run Java applications cost-effectively in the cloud</td>
</tr>
<tr>
<td>VERT.X</td>
<td>Toolkit for building reactive applications on the JVM</td>
</tr>
<tr>
<td>JNOQL</td>
<td>Framework for NoSQL integration</td>
</tr>
</tbody>
</table>
Great job done by the JCP during 20 years
Each specification managed as a Java Specification Request
Under the responsibility of an Expert Group and a Spec Lead
3 expected deliverables:
1. A Specification Document
2. A Test Compatibility Kit
3. A Reference Implementation
Following a well-defined lifecycle:
Eclipse Foundation Specification Process

Jakarta EE Working Group

- Specifications and implementations run as standard open source projects
- No Expert Group, no Specification Leader, no exclusive ownership of the IP
- No Reference Implementation
- At least one compatible Open Source implementation
- Free access to the TCK, self certification
Being a good cloud citizen

Orchestration (Kubernetes)
- Routing
- Load balancing
- Service registry & discovery

Runtime (JVM, App. Server)
- Fast startup
- Low foot print

MicroProfile
- Configuration
- Built-in monitoring, health-check
- Built-in resiliency
- Distributed tracing
- Self-documented APIs
- Security
MicroProfile in motion


June 2018

Java EE 8
# My wishlist for 2019

## Platform improvements
- EJB and CDI convergence
- Modular APIs and implementations

## Modernization
- Microservices
- NoSQL
- GraphQL
- ST like DeltaSpike Data to simplify JPA

## Governance
- Proper specification process execution
- Keep MicroProfile and Jakarta EE projects separate (for the moment)
- Java SE 8 & Java SE 11 compliance (LTS)
- High-consistency and loose-coupling with Cloud platforms

## Dev Attention & Adoption
- Get developers attention: at big events but also locally
- Easy and intuitive adoption, project starters
- Ease of testing
- Lightweight (invisible?) App. Servers
- Seamless integration with K8S & Docker
Conclusion

- Jakarta EE is more than the future of Java EE
- Microprofile has paved the way
- Eclipse Foundation is the place-to-be
- 2019 will be decisive:
  - Execution of the specification process
  - Developers attention and adoption
- As users, we can be part of the story!
Ready for another 20 years?
Thank you!