# JSR 365 (CDI 2.0) Review

June 16 2015

Antoine Sabot-Durand

# Agenda

- History & Background
- Goals
- CDI survey
- Expert Group and working method
- CDI 2.0 Early Draft 1
- Work done on RI and TCK
- Next steps
- Q&A

# History & background

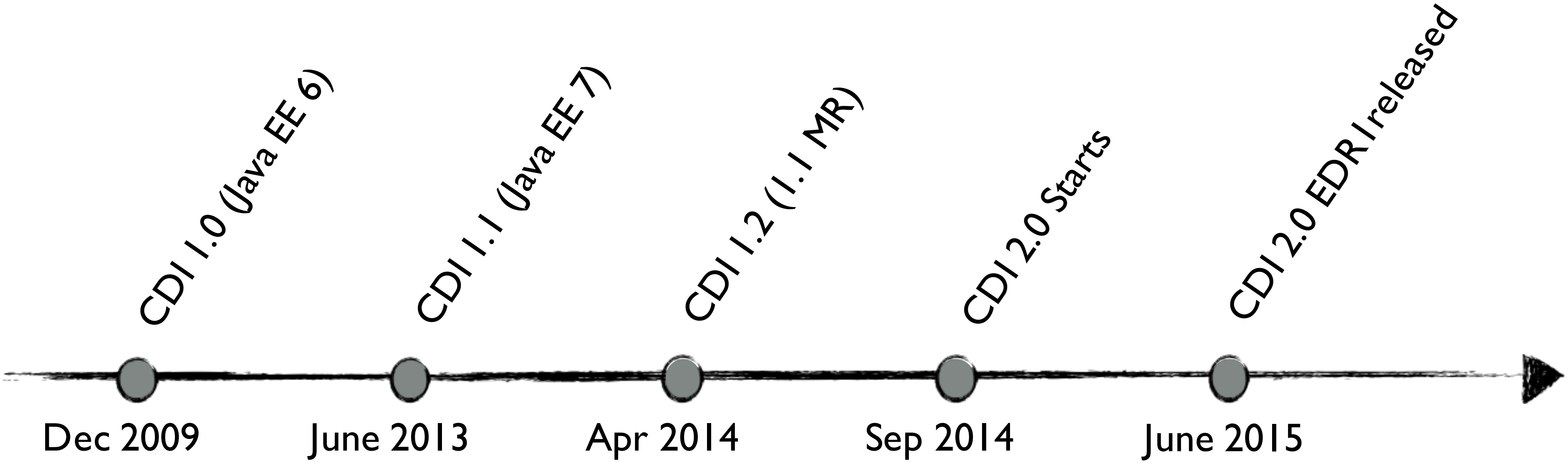
# What is CDI?

- **CDI stands for Contexts and Dependency Injection**

- **At the origin CDI was thought to fill the gap between Front spec and EJB**
  - All EJB session beans are also CDI beans
  - CDI integrates Servlet scope as its core built-in contexts

- **CDI propose a programming model which provides:**
  - A well-defined lifecycle for stateful objects bound to lifecycle contexts, where the set of contexts is extensible
  - A typesafe dependency injection mechanism
  - The ability to decorate or to associate interceptors to objects with a typesafe approach
  - An event notification model
  - An SPI allowing portable extensions to integrate cleanly with the container

# CDI Timeline



CDI 1.0 (Java EE 6)

Dec 2009

CDI 1.1 (Java EE 7)

June 2013

CDI 1.2 (1.1 MR)

Apr 2014

CDI 2.0 Starts

Sep 2014

CDI 2.0 EDR1 released

June 2015

# Background of CDI

- CDI 1.0 (JSR 299) was strongly focused on Java EE
- It was first delivered as part of Java EE 6.
- CDI 1.1 (JSR 346) was a minor update to 1.0, and focused on resolving issues and add features for advanced developers.
- CDI 1.1 was delivered with Java EE 7
- CDI has become increasingly popular, and is now seen as the core programming model in Java EE

# Goals

# Goals

- We had strong feedback from the community:
  - Users feedback:
    - Need CDI to be more widely available in other Java EE specifications as a transverse integration solution
    - Be able to run CDI outside of a Java EE container.
  - 3rd parties spec and framework feedback:
    - Need to have an enhance SPI to ease integration
    - Need to have a lighter footprint of CDI API / Impl

# CDI 2.0 main goals

- Java SE support (the spec was renamed Contexts and Dependency Injection for Java):
  - Use in desktop or alternative containers

- Modularity allowing us to work on a CDI light “part”:
  - Removing parts of spec requiring proxies in implementation
  - Ease 3rd parties to develop CDI integration
  - This could make CDI compatible with embedded platform…

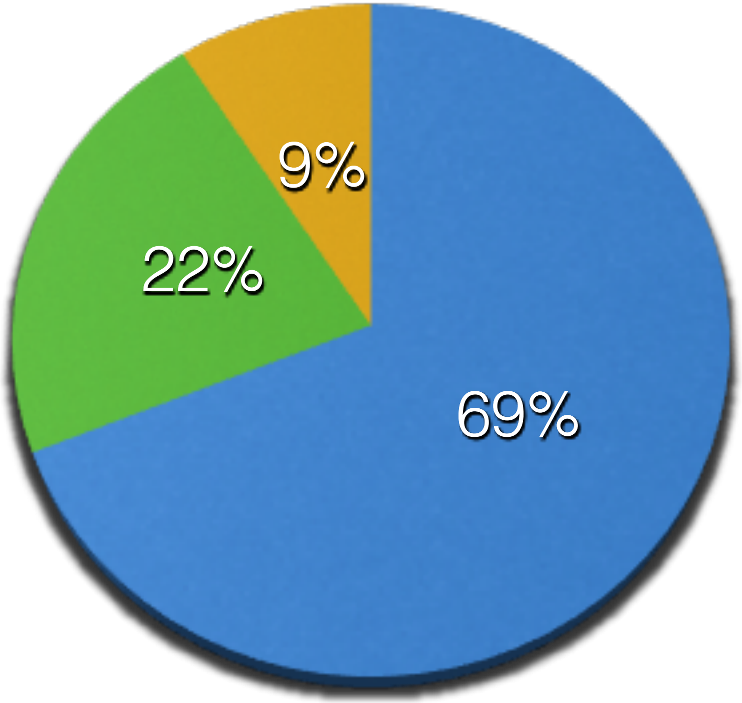
# CDI 2.0 survey

# •About this survey…

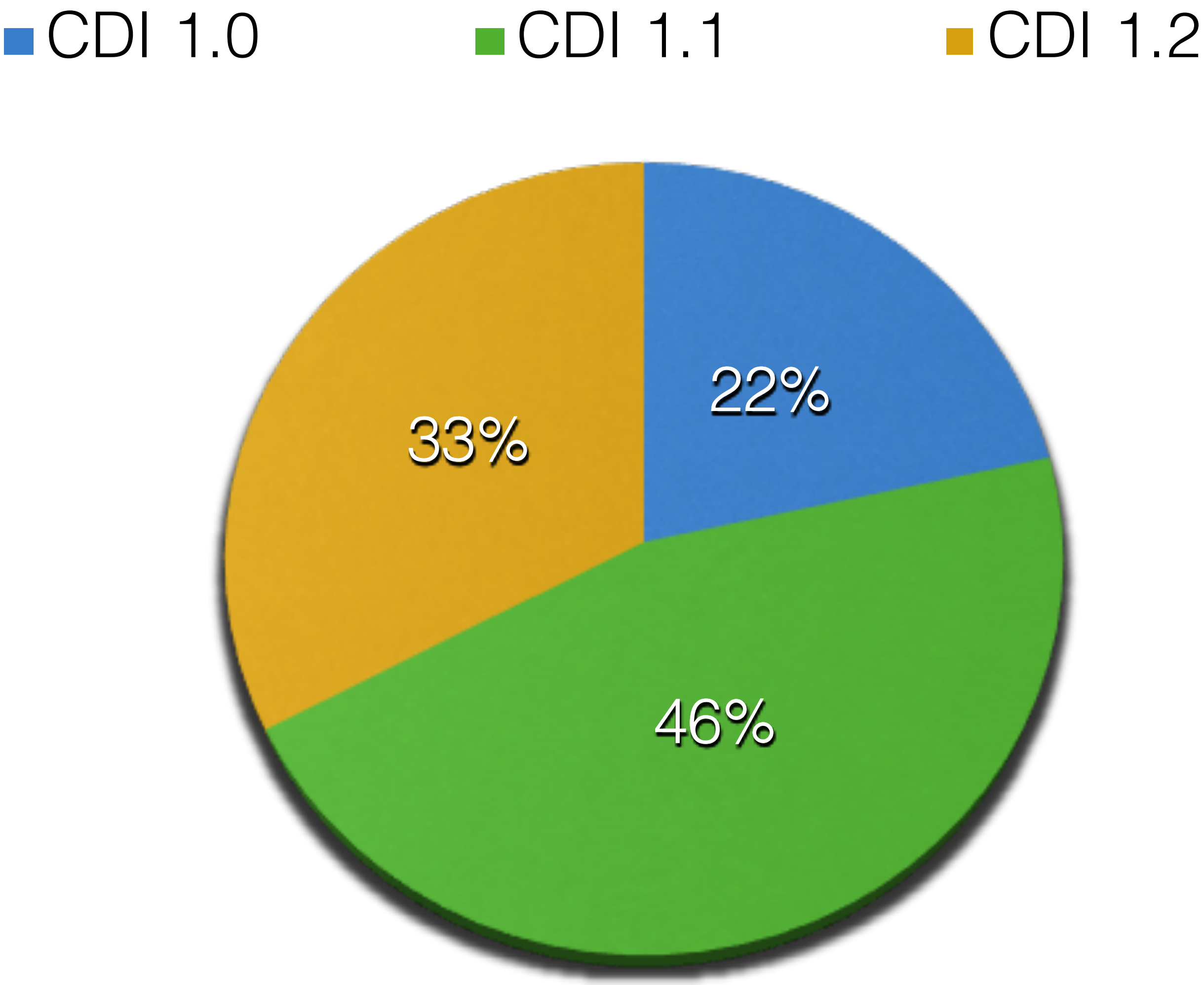
- There was a survey before the survey to gather wanted features
- Between May 28th and June 30th 2015
- 260 participants
- 3 « demography » question
- 20 new features to rate (from 1 to 5)

# •Who answered?

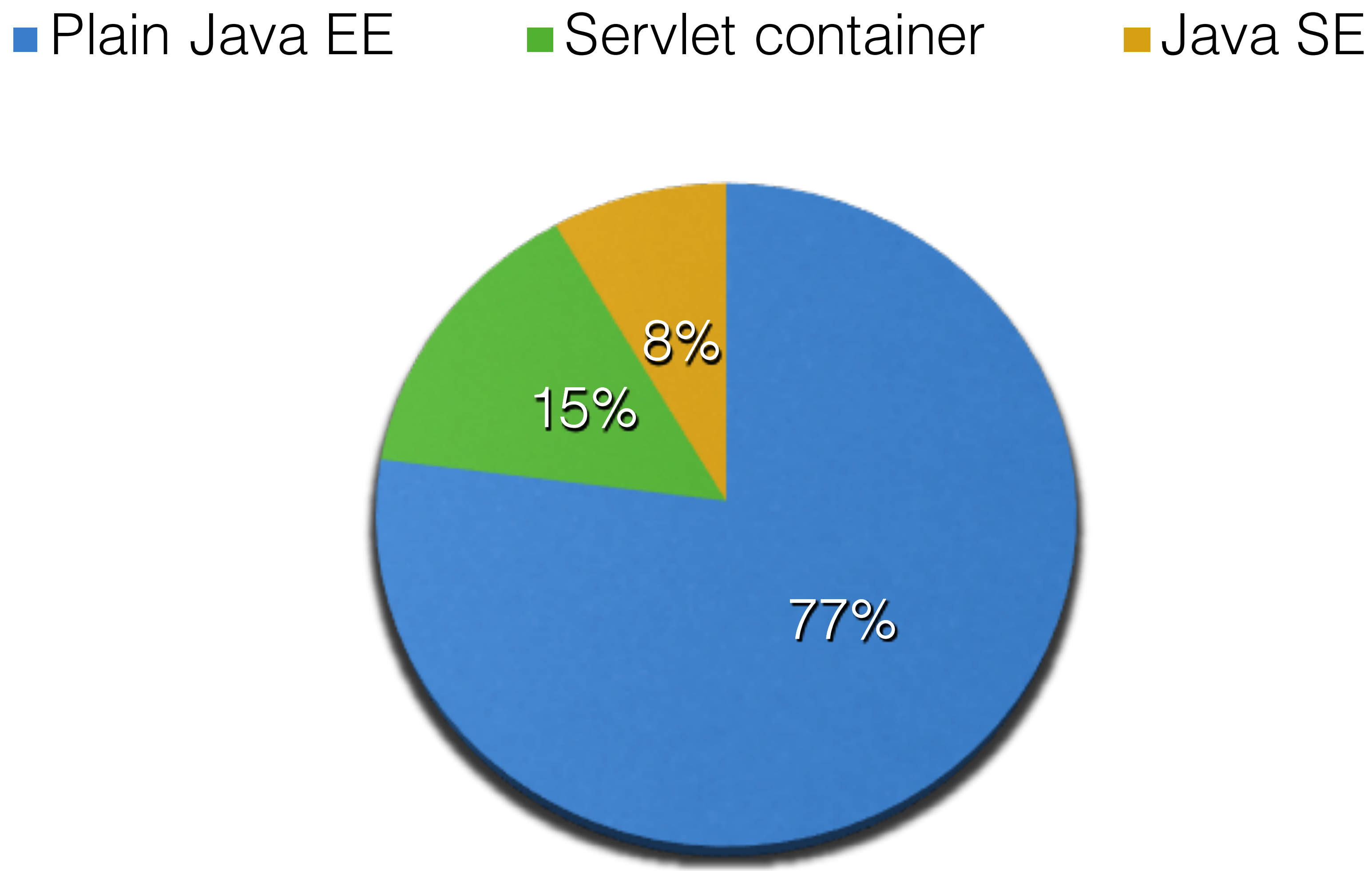
- 🟦 I'm a developer using CDI on my projects
- 🟩 I'm an advanced developer extending CDI on my projects
- 🟨 I'm developing a framework based on CDI



# •CDI version they're using



# •What are their usage?



# •Results 1/2

| Features from most to less wanted | av. rating |
|---|---|
| Asynchronous method invocation | 4,04 |
| Add asynchronous event support * | 4,00 |
| @Startup for CDI | 3,92 |
| Bootstrapping the container outside Java EE * | 3,90 |
| AOP for produced or custom beans * | 3,65 |
| Mutable container at runtime * | 3,62 |
| Security support | 3,61 |
| Observers ordering * | 3,53 |
| Better event control | 3,50 |
| Enhance SPI to give better access to all metadata | 3,41 |

# •Results 2/2

| Feature from most to less wanted | av. rating |
|---|---:|
| Better EAR support | 3,32 |
| Helpers to manipulate and build CDI metadata | 3,32 |
| An easier way to create AnnotationLiteral (and TypeLiteral) | 3,27 |
| Context SPI easily pluggable by the container | 3,24 |
| Configuration file | 3,17 |
| CDI parts * | 3,12 |
| Enhance SPI to retrieve Bean from its instances | 3,1 |
| JMX support | 3,07 |
| Introduce CDI Lite. * | 2,78 |
| Aligment with portlet 3.0 | 2,02 |

# Expert Group and working method

# EG members

- **Pete Muir (*Red Hat*)**
- **Antoine Sabot-Durand (*Red Hat*)**
- John Ament
- David Currie (*IBM*)
- Anatole Tresch (*Credit Suisse*)
- Antonio Goncalves
- Thorben Janssen
- Rajmahendra Hegde (*JUG Chennai*)
- Werner Keil
- Joseph Snyder (*Oracle*)
- Mark Paluch
- José Paumard
- Florent BENOIT (*SERLI*)
- Mark Struberg
- David Blevins (*Tomitribe*)
- George Gastaldi (*Red Hat*)
- Otavio Santana

# Work organization

- We use JCP version 2.9

- Our main communication tool is the mailing list:
  - Non EG members can also join the conversion
  - Archive of ML are accessible on our website

- Doing meeting each week on IRC:
  - Meeting archive are accessible on the

- Tasks are organized with a Jira server:
  - Every contribution is welcome there too

# CDI 2.0 Early Draft 1

# Early Draft

- Early Draft 1 is about to be released
- We are a bit late on our schedule but the work was bigger than expected
- Some very old and important request were included in it
- Features are in the 10 first requested ones by the survey

# Early Draft content, Java SE support

- **We specified API to boot CDI in Java SE:**

```
public static void main(String... args) {
     CDIProvider provider = CDI.getCDIProvider();
     CDI<Object> cdi = provider.initialize();
     // retrieve a bean and do work with it
     MyBean myBean = cdi.select(MyBean.class).get();
     myBean.doWork();
     // when done
     cdi.shutdown();
}
```

- **Desktop and non Java EE application can now use a standard way to boot CDI**

# Early Draft content, spec split

- **The specification is now split in 3 parts**
  - CDI core: the common specification regardless of the platform
  - CDI for Java SE: specific features for Java SE
    - Mainly SE boot
  - CDI for Java EE: specific feature for Java
    - EJB, servlet, EL, JSF integration
- **This split was necessary for Java SE support**
- **It’s also the first step for a future CDI light**

# Early Draft content, Event Ordering

- One of the oldest user request :

```
void afterLogin(@Observes @Priority(APPLICATION)
                LoggedInEvent event) { ... }
```

- We decided to use the @Priority from commons annotation (JSR 250)

- JSR 250 will need a Maintenance Release to have @Priority targeting a parameter

# Early Draft content, Asynchronous Event

- **The most popular feature in the survey:**

```
public interface Event<T> {
 …
 public <U extends T> CompletionStage<U> fireAsync(U event);
 public <U extends T> CompletionStage<U> fireAsync(U event, Executor executor);
 …
}
```

- **Based on Java 8 new Async API**

- **Introducing a new kind of observes for backward compatibility with @ObservesAsync**

# Work done on RI and TCK

# Work done on RI

- **JBoss Weld 3.x will the Reference Implementation of CDI 2.0**
- **Weld Team release alpha version of Weld 3.0 on regular basis to test ideas**
- **Weld 3.0 footprint was divided by 2 (in comparison with 2.x):**
  - Switching to Java 8 collections and streams help remove extra libraries
- **Weld team will release an alpha for EDR1 shortly after draft release:**
  - Very important to have community feedback

# Work done on TCK

- **Thanks to integrators and other implementation developers we have a very active community around TCK**
- **We are working on using TCK to teach CDI:**
  - Link the TCK and the Spec to see what are the test behind a rule in the spec
  - We'll release 2 HTML format of the specification document
    - One without TCK link
    - One with links to TCK challenges source code.
- **A side effect would be to increase contribution and enhance TCK quality**

# Next steps

# Our next steps

- **Complete new features:**
  - Contexts & Bean discovery in Java SE
  - Multi container support in Java SE
  - Discuss about CDI light
  - Enhance SPI for developers and users
- **We are aiming to start public review in Q4 2015/Q1 2016**
- **Release of CDI 2.0 in Q2 2016**

# Q&A

Java Community Process

Thank you!
http://jcp.org