Application development with CDI and Arquillian

Aleš Justin, JBoss by RedHat
Marko Štrukelj, JBoss by RedHat

About us

- Marko Štrukelj
  - JBoss GateIn Core Developer
  - OpenBlend Founding Member
- Aleš Justin
  - JBoss Weld Project Lead
  - Microcontainer Co-Author
  - OpenBlend Founding Member / President

Agenda

- Requirements and Tools
- CDI Introduction
  - Step-By-Step CDI
- Arquillian Introduction
  - Deep Dive Into Containers
- Extensions and Seam3

Requirements and Tools

- IDE; Eclipse, IntelliJ, NetBeans, ...
- JBossAS7, GlassFish 3.1, ...
- Maven3
- Git Client
- ...?

CDI introduction

- Context and Dependency Injection
- JSR-299 → JBoss Weld RI
- “Type-Safe” Programming
- “Contexts” Concept
- Easy to Extend: “Extensions”
- Note: beans.xml Marker File
- Part of JEE6 → Part of Application Servers

@Inject

- Actually Comes From JSR-330
- Type Matching, *not* Scope Matching
- Multiple Matching Types?
  - @Qualifier
  - @Alternative

@Qualifier

- @Default Added by Default
- Narrows Down Choices
  - Annotation Values also *Limit* Matching
- @NonBinding
- @Named(“openblend”)

@Scope

- Built-in Scopes
  - @Dependent
  - @RequestScoped
  - @ConversationScoped *
    - @Inject Conversation
  - @SessionScoped
  - @ApplicationScoped
- Add Your Own

@Produces

- @Disposes
- @Alternative
- @Stereotype
- @New
- @Typed

@Observes

- Simple “JMS”
- javax.enterprise.event.Event::fire
  - Runtime Sub-Selection
- Tx Supported
- Reception

@InterceptorBinding

- Markup Annotation
  - e.g. @Transactional
- Interceptor
  - Can also be CDI Bean
  - Any POJO
  - @AroundInvoke method
  - Single InvocationContext parameter

@Decorator

- Similar to Interceptors
  - But with Exact Type and Methods
- @Delegate
  - Replaces Original Down to Qualifiers

@Specializes

- Replaces Extended Bean
- Inherits All Qualifiers, Stereotypes, ...
- e.g. External Bean Replacement

Direct API

- Instance
  - Runtime Bean Selection
- InjectionPoint
- BeanManager
- TypeLiteral
- AnnotationLiteral
- Provider

How it all works?

- Entry Point / Integration
- Everything is a Proxy
- Runtime Integration SPI
- ... ?

Arquillian introduction

- Environment / Runtime abstraction
- JUnit & TestNG support
- Write Once Run Everywhere
- No Ant, Maven, etc - Pure Java Code
- ShrinkWrap Project Dependency
- Embedded, Managed, Remote
- Infinity of Implemented Containers

@Test

- @Deployment
- @OperateOnDeployment
- @TargetsContainer
- @RunAsClient
- @ShouldThrowException
- @OverProtocol

New Container

- DeployableContainer *
- ContainerConfiguration *
- LoadableExtension *
- AuxiliaryArchiveAppender
- TestEnricher

CDI Extensions

- javax.enterprise.inject.spi.Extension Marker
- Container Lifecycle Events
  - BeforeBeanDiscovery
  - ProcessAnnotatedType
  - ProcessInjectionTarget, ProcessProducer
  - ProcessBean, ProcessObserverMethod
  - AfterBeanDiscovery
  - AfterDeploymentValidation

Seam3

- Pure CDI Dependency; *no* Weld
- Plugs-in as CDI Extension
- Growing List of Modules
  - Solder
  - Catch
  - Security
  - Persistence
  - ...

Q&A

- http:/github.com/alesj/cdi-arq-workshop