A Quick Tour of the CDI Landscape

Reza Rahman
Expert Group Member, Java EE 6 and EJB 3.1
Resin EJB 3.1 Lite Container Developer
Author, EJB 3 in Action
reza@caucho.com
Contexts and Dependency Injection for Java EE (CDI)

- Next-generation dependency injection for Java EE
- Synthesizes best ideas from Seam 2, Guice and Spring
- Many innovative features on its own right
- Focus on loose-coupling, Java-centric type-safety, annotations, expressiveness and ease-of-use
- Makes Java EE much more flexible, testable, pluggable and extensible
- Large ecosystem rapidly evolving around CDI
CDI and Java EE
CDI Features

- Basic dependency injection
  - `@Inject`, `@Qualifier`, `@Stereotype`, `@Alternative`, `@Instance`, `@All`, `@Any`

- Component naming
  - `@Named`

- Context management
  - `@Dependent`, `@RequestScoped`, `@SessionScoped`, `@ConversationScoped`, `@ApplicationScoped`, `@Scope`
CDI Features

- **Custom Object Factories**
  - `@Produces`, `@Disposes`, `InjectionPoint`

- **Lightweight Events**
  - `Event`, `@Observes`

- **Interceptors/Decorators**
  - `@Interceptor`, `@InterceptorBinding`, `@AroundInvoke`, `InvocationContext`, `@Decorator`, `@Delegate`

- **Portable extensions SPI**
CDI Implementations
Weld

- CDI reference implementation and compatibility test kit from JBoss
- Weld reference excellent documentation for CDI
- Evolved from Seam 2 core
- Included in GlassFish and JBoss AS
- Can be used with Tomcat, Jetty
- Can be used in standalone Java SE
CanDI

- Cauchio implementation included in the Resin application server
- CanDI forms the core of Resin itself
- Compact, schema-driven, type-safe XML configuration
- CDI XML used to configure Resin itself
- Use of annotations like @TransactionAttribute, @Schedule, @Asynchronous, @RolesAllowed, @RunAs, @Lock, @Startup and @Remote outside EJB in plain CDI managed beans
CanDI

- @Pooled, @TransactionScoped, @ThreadScoped custom scopes
- Embedded container based testing framework integrating CDI, EJB, JPA with JUnit
- Integrating CDI with PHP
- Portable extensions in progress for JDBC, iBATIS, Struts 2, Quartz
- Focus on performance
CanDI and Resin
OpenWebBeans

- Apache implementation of CDI included in Geronimo
- Closely related to Apache OpenEJB
- Great integration with Tomcat along with OpenEJB
- Extended injection of JMS objects (sessions, connection factories)
- Runs in standalone Java SE
- Portable extensions for major Apache projects
Java EE 6 Implementations
Java EE 6 Application Servers

- Weld
- JBoss
- resin®
- GlassFish
- OpenWebBeans
- Geronimo
- TmaxSoft
- OnAS
- WebSphere
- Jetty
GlassFish v3

- Open source Java EE 6 reference implementation
- Fully backed by Oracle
- Earliest Java EE 6 runtime
- Web Profile as well as Full Platform supported
- OSGi based modular architecture
- Excellent pooling, clustering and administration features
- Embedded container for testing
JBoss AS 6

- Most widely used open source Java application server
- Recently passed Java EE 6 Web Profile TCK
- Java EE 6 Full Profile implementation on the way
- Track record for innovation and thought leadership
- Pooling, caching, clustering, admin console
- HornetQ excellent JMS implementation
Resin 4

- Focus on fast, lightweight, reliable open source Java EE 6 Web Profile server
- Scheduling, asynchronous processing, Hessian remoting, message-driven beans and lightweight JMS also included
- HTTP server, proxy caching, load-balancing, pooling, clustering, caching, cloud support, security framework, SSL, administration
- Running PHP on Java
Geronimo 3

- Apache licensed open source application server
- Basis for IBM WebSphere Community Edition
- Geronimo 3 aimed to pass Java EE 6 Web Profile TCK shortly with a Full Profile implementation to follow
- OSGi, clustering, administrative console
- Includes Tomcat/Jetty, MyFaces, OpenWebBeans, OpenEJB, OpenJPA, ActiveMQ, CXF/Axis, Derby
- TomEE Tomcat centric Java EE 6 Web Profile offering
## More on the Way…

<table>
<thead>
<tr>
<th><strong>TmaxSoft JEUS</strong></th>
<th>Korean application server; JEUS 7 one of the earliest to get Java EE 6 certified; still in beta stage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WebLogic</strong></td>
<td>Next release, WebLogic Server 11g R2, will support both Java EE 6 Full and Web Profile; planned for calendar year 2011</td>
</tr>
<tr>
<td><strong>WebSphere</strong></td>
<td>WebSphere 8 beta previews Java EE 6</td>
</tr>
<tr>
<td><strong>JOnAS</strong></td>
<td>Open source application server from OW2 consortium; JOnAS 5.2 implementing some Java EE 6 features; JOnAS 6 will support Java EE 6</td>
</tr>
<tr>
<td><strong>SIwpas</strong></td>
<td>Tomcat based open source Java EE 6 Web Profile offering leveraging Apache projects from Turkish company MechSoft; not certified yet</td>
</tr>
</tbody>
</table>
Portable Extensions
Seam 3

- Set of CDI portable extensions independent of Weld/JBoss AS
- Each module developed separately
- There are Seam “umbrella” releases
# Seam Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solder</td>
<td>General enhancements to CDI</td>
</tr>
<tr>
<td>XML Configuration</td>
<td>XML configuration for managed beans</td>
</tr>
<tr>
<td><strong>Persistence</strong></td>
<td>Transactions and persistence for non-EJB managed beans</td>
</tr>
<tr>
<td>Faces</td>
<td>JSF enhancements</td>
</tr>
<tr>
<td>Servlet</td>
<td>Servlet enhancements</td>
</tr>
<tr>
<td><strong>JMS</strong></td>
<td>JMS integration</td>
</tr>
<tr>
<td>REST</td>
<td>REST enhancements</td>
</tr>
<tr>
<td>Remoting</td>
<td>JavaScript remoting of managed beans</td>
</tr>
</tbody>
</table>
# Seam Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catch</td>
<td>Exception handling framework</td>
</tr>
<tr>
<td>International</td>
<td>Internationalized/localized locale, time, messages</td>
</tr>
<tr>
<td>Security</td>
<td>Higher level security API for Java EE</td>
</tr>
<tr>
<td>Mail</td>
<td>JavaMail integration</td>
</tr>
<tr>
<td>Cron</td>
<td>CDI based scheduling</td>
</tr>
<tr>
<td>Document</td>
<td>Document generation (PDF, Excel)</td>
</tr>
<tr>
<td>Seam 2</td>
<td>Seam 2 backwards compatibility</td>
</tr>
<tr>
<td>Spring</td>
<td>Spring interoperability</td>
</tr>
</tbody>
</table>
## Seam Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wicket</td>
<td>Wicket Integration</td>
</tr>
<tr>
<td>GWT</td>
<td>GWT integration</td>
</tr>
<tr>
<td>Drools</td>
<td>Drools integration</td>
</tr>
<tr>
<td>jBPM</td>
<td>jBPM integration</td>
</tr>
<tr>
<td>JBoss ESB</td>
<td>JBoss ESB integration</td>
</tr>
<tr>
<td>Clouds</td>
<td>Cloud frameworks integration (JClouds, Infinispan)</td>
</tr>
</tbody>
</table>
Apache MyFaces CODI

- Set of CDI portable extensions from Apache
- Somewhat MyFaces/JSF centric, aims to provide a friendly “toolbox” for development with Java EE 6
- Modular, configurable and extensible
- Closely related to Apache OpenWebBeans
Using JSF Project Stage

```java
<context-param>
  <param-name>javax.faces.PROJECT_STAGE</param-name>
  <param-value>Development</param-value>
</context-param>

@Inject private ProjectStage projectStage;
...
boolean isDevProjectStage =
  ProjectStage.Development.equals(projectStage);

@Alternative
@ProjectStageActivated(ProjectStage.UnitTest.class)
public class TestServiceMockImpl implements Service {
  ...
}
```
Injecting Loggers

```java
public class DefaultAccountDao implements AccountDao {
    @Inject
    private Logger logger;

    @PersistenceContext
    private EntityManager entityManager;

    public void addAccount(Account account) {
        ...
    }
    ...
}
```
CODI Conversations

@Named @ConversationScoped
// @ConversationGroup(Wizard1.class)
public class CodiConversationBean {
   @Inject private Conversation conversation;
   ...
   public void finishWorkflow() {
      ...
      conversation.close();
      ...
   }
}
Other CODI Scopes

@Named @ViewScoped
public class MyViewScopedBean {
    ...
}

@Named @WindowScoped
public class MyWindowScopedBean {
    @Inject private Conversation conversation;
    ...
}
Enhanced JSF Injection

@Advanced
@FacesValidator(...) public class DependencyInjectionAwareValidator implements Validator {
    @Inject FacesContext facesContext;
    @Inject ValidationService validationService;
    ...

    public void validate(FacesContext facesContext,
                         UIComponent component, Object value)
        throws ValidatorException {
        Violation violation = 
          validationService.validate(value);
        ...
    }
}
JSF Life-Cycle Listeners

```java
public void onPreRender(@Observes @BeforePhase(RENDER_RESPONSE)
    PhaseEvent phaseEvent) {
    ...
}

public void onFacesRequestStart(@Observes @BeforeFacesRequest FacesContext facesContext) {
    ...
}
```
Injecting Bean Validation Objects

```java
@Inject  @Advanced
private Validator validator;

@Inject  @Advanced
private ValidatorFactory validatorFactory;
```
Injecting into Validators

public class UserNameValidator implements ConstraintValidator<UserName, String> {
    @Inject private UserService userService;

    public boolean isValid(String userName,
                             ConstraintValidatorContext constraintValidatorContext) {
        return userService.validateUserName(userName);
    }
}
CODI Transactions

```java
public class DefaultAccountService implements AccountService {
    @Inject
    private AccountDao accountDao;

    @Transactional
    public void addAccount(Account account) {
        accountDao.addAccount(account);
    }

    ...
}
```
Scripting (JSR-223) Integration

@Inject @ScriptLanguage(JavaScript.class)
private ScriptExecutor scriptExecutor;
...
return scriptExecutor.eval("10 + 4", Double.class);

@Inject @ScriptLanguage(JavaScript.class)
private ScriptBuilder scriptBuilder;
...
return scriptBuilder.script("x + y").namedArgument("x", a).namedArgument("y", b).eval(Double.class);

<h:outputText value="#{sExec.js['[a:#{bean1.result}, b:#{bean2.result}]['a + b']}}"/>
Cargo JUnit Test

@RunWith(JUnit4WithCargo.class)
public class DemoTestCase extends AbstractCargoTest {
  @Test
  public void doSomeTest() throws Exception {
    SimplePageInteraction interaction = new SimplePageInteraction(getTestConfiguration())
      .with(Pages.Page1.class).with(Pages.Page1.class).start(Pages.Page1.class);
    interaction.useForm("form1").set_value("form1:input1", "1").click("form1:button1");
    interaction.checkState(Pages.Page2.class)
      .checkTextValue("output1", "1");
  }
}
Plain JUnit Test

```java
@RunWith(JUnit4.class)
public class BidServiceTest extends AbstractTest {
    @Inject private BidService bidService;
    ...
    @Test
    public void testGetBid() {
        Bid bid = bidService.getBid(1L);
        assertNotNull(bid);
    }
    ...
}
```
ZK Framework CDI Integration

- ZK is Ajax-based event-driven RIA engine with a rich set of XHTML and XUL components
- ZUML mark-up language for creating UIs
- ZK Provides powerful CDI integration, including remote contexts (+ custom scopes)
ZUML Markup

<?page title="HelloWorld"
    contentType="text/html;charset=UTF-8"?>
<?variable-resolver class=
    "org.zkoss.zkplus.cdi.DelegatingVariableResolver"?>
<zk>
    <window id="helloWindow" title="HelloWorld"
        border="normal" width="250px"
        apply="${helloWorld}"
        ><textbox id="guestName" width="150px" />
    <button id="sayHelloBtn" label="Say hello"/>
    </window>
</zk>
ZK CDI Managed Bean

@Named
@SessionScoped
public class HelloWorld extends GenericComposer
    implements Serializable {

    @Inject @ComponentId("guestName") Textbox guestName;
    @Inject @ComponentId("sayHelloBtn") Button sayHelloBtn;
    @Inject @ComponentId("helloWindow") Window helloWindow;

    public void sayHello(@Observes
        @Events("sayHelloBtn.onClick") MouseEvent evt) {
        helloWindow.setTitle("Hello "
            + guestName.getValue());
    }
}
Tools
Arquillian

- Embedded or remote (or managed) container integration with JUnit/TestNG
- Container life-cycle management, micro-deployments, injection into unit tests
- Supports multiple containers and aims to be container-independent
- Ideal for cross-vendor unit testing
- Integrates with JSFUnit
## Arquillian Supported Containers

<table>
<thead>
<tr>
<th>Container</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>JBoss AS 5</td>
<td>Remote, managed</td>
</tr>
<tr>
<td>JBoss AS 6</td>
<td>Remote, managed, embedded</td>
</tr>
<tr>
<td>GlassFish 3</td>
<td>Remote, embedded</td>
</tr>
<tr>
<td>Tomcat 6</td>
<td>Embedded</td>
</tr>
<tr>
<td>Jetty 7</td>
<td>Embedded</td>
</tr>
<tr>
<td>Weld SE</td>
<td>Embedded</td>
</tr>
<tr>
<td>OpenWebBeans</td>
<td>Embedded</td>
</tr>
<tr>
<td>OpenEJB</td>
<td>Embedded</td>
</tr>
</tbody>
</table>
@RunWith(Arquillian.class)
public class BidServiceTest {
    @Deployment
    public static WebArchive createDeployment() {
        return ShrinkWrap.create(WebArchive.class, "test.war")
            .addClasses(BidService.class, DefaultBidService.class,
                        BidDao.class, DefaultBidDao.class, Bid.class)
            .addWebResource(EmptyAsset.INSTANCE, "beans.xml")
            .addWebResource("test-persistence.xml",
                            "classes/META-INF/persistence.xml");
    }

    @Inject private BidService bidService;

    @Test
    public void testGetBid() {
        Bid bid = bidService.getBid(1L);
        assertNotNull(bid);
    }
}
Forge

- Command-line RAD code-generation tool for Java EE
- Based on Maven
- Aimed at fast bootstrap and refactoring
- Highly pluggable and extensible
- Complements JBoss Tools CDI/Java EE 6 support
Forge Screenshot

```
[forge-distribution] dist $ cd ~/Desktop/
[no project] Desktop $ new-project --named example --topLevelPackage com.example
Use [/home/lb3/Desktop/example] as project directory? [Y/n]
Wrote /home/lb3/Desktop/example/src/main/java/com/example/HelloWorld.java
Wrote /home/lb3/Desktop/example/src/main/resources/META-INF/forge.xml
***SUCCESS*** Created project [example] in new working directory [example]
[example] example $ 
[example] example $ echo "that was easy..."
that was easy... 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
[example] example $ 
```
public class ExamplePlugin implements Plugin {
    @Command(name="perform")
    public void exampleCommand(
        @Option(name="one") String one,
        @Option(name="two") String two,
        @Option String three,
        @Option String four,
        PipeOut out) {
        out.println(">> first option equals: " + one);
        out.println(">> second option equals: " + two);
        out.println(">> third option equals: " + three);
        out.println(">> fourth option equals: " + four);
    }
}
JBoss Tools

- A set of Eclipse plug-ins for JBoss technologies like CDI, Seam, JSF, Facelets, Hibernate, JBoss AS, Drools, jBPM and JBoss ESB

- Provides excellent support for CDI (which Eclipse currently does not)

- CDI wizards, validation, quick-fix, auto-completion, navigation
CDI Wizards
CDI Validation
CDI Code Navigation
Eclipse

- Some Java EE 6 support (EJB 3.1, Servlet 3, Facelets, JPA 2) but not complete
- CDI not supported at all
- GlassFish plug-in includes a little better support for Java EE 6, but not much
- Using JBoss Tools with Eclipse good choice for CDI developers for the time-being
NetBeans

- Excellent support for CDI and Java EE 6
- CDI wizards, validation, quick-fix, auto-completion, navigation, dependency graph
CDI Validation

```
@Default
public class FastProducer extends Object implements Producer

Javadoc not found. Either javadoc documentation for this item does not exist or you have not added specified javadoc in the java Platform Manager or the Library Manager.
```

```
@Inject
public Producer myProducer;
```
CDI Auto-Completion

```java
package newpackage;

@NewQualifier
class NewQualifier {
}
```

```java
package exercise3;

import exercise2.Item;
import javax.validation.Validateable.

@Author
class ItemValidator {
    public
}
```

```java
javax.enterprise.

@Target(value=(TYPE, METHOD, FIELD))
@Retention(value=RUNTIME)
public @interface Alternative

An alternative is not available for injection, lookup or EL resolution to
classes or JSP/JSF pages in a module unless the module is a bean archive
and the alternative is explicitly selected in that bean archive. An
alternative is never available for injection, lookup or EL resolution in a
module that is not a bean Archive.
```
package newpackage;

import javax.enterprise.event.Event;
import javax.enterprise.inject.Any;
import javax.inject.Inject;

public class NewClass {
    @Inject @Any Event<EventObject> even
}

Observers for "event" event field.
Event Type: EventObject
Event Qualifier: @Any

newpackage.ObserverClass
void observer10Observers
@Inject
EventObject event

Qualifiers: @Named
Filters: empty,
IntelliJ IDEA

- CDI and Java EE 6 strongly supported
- Dependency diagrams, refactoring, navigation, code completion
- One of the earliest to support CDI
CDI Tool Window
CDI Validation

```java
public class InjectInMe {
    @Inject
    Credo credos;
    @Inject
    Credo2 credo2;
}
```

Ambiguous dependency: there are multiple beans that match the injection point more...

Unsatisfied dependency: no bean matches the injection point more...(Ctrl+F1)

```java
@SessionScoped @Named
public class Login implements Serializable {
    @Inject
    Credentials credentials;
    Injected normal scoped bean is not proxyable more...(Ctrl+F1)
    Unproxyable bean types inconsistency.
    Certain legal bean types cannot be proxied by the container: classes which don't have a
    non-private constructor with no parameters, classes which are declared final or have final
    methods, primitive types, and array types.
}
```

```java
org.jboss.weld.examples.login
@RequestScoped
@Named
@Default
public final class Credentials extends Object
    implements Serializable
```
Summary

- CDI next generation dependency injection for Java EE
- Rich ecosystem of implementations, supported runtime platforms, portable extensions and tools rapidly evolving around CDI
- Portable extensions for both ease-of-use, Java EE enhancements as well as open source third-party tools
- Tools for RAD/code generation, testing as well as traditional IDE based development
- More on the way!
References

CDI


Implementations

- Weld, the JBoss reference implementation for JSR 299, http://seamframework.org/Weld
- CanDI, the JSR 299 implementation for Caucho Resin, http://caucho.com/projects/candi/
References

- OpenWebBeans, Apache implementation of JSR 299, [http://openwebbeans.apache.org](http://openwebbeans.apache.org)

Application Servers

- GlassFish, [http://glassfish.java.net](http://glassfish.java.net)
- JBoss, [http://www.jboss.org/jbossas](http://www.jboss.org/jbossas)
- OpenEJB, [http://openejb.apache.org](http://openejb.apache.org)
References

Portable Extensions

- Seam 3, http://seamframework.org

Tools

References

- Seam Forge,
  http://seamframework.org/Documentation/SeamForge

- JBoss Tools,
  http://community.jboss.org/en/tools/blog/2011/01/31/
  overview-of-context-dependency-injection-tools
Thanks!

Email: reza@caucho.com

Twitter: cauchoresin

Facebook:
http://www.facebook.com/pages/Caucho-Technology/8671323790

Presentation materials: http://www.caucho.com/articles/