<embed/it>

Enterprise Java Beans

PA165
30. 10. 2012
Petr Adámek

Content

- **EJB Introduction and history**
- **EJB Components**
  - Stateless Session Bean
  - Statefull Session Bean
  - Singleton Session Bean
  - Message Driven Bean
- **AOP / Interceptors**
- **Transaction Management**
- **Security Management**
- **EJB Time Service**
- **CDI**
- **EJB versus Spring**

EJB Introduction

- Enterprise JavaBeans (EJB) is a managed, server-side component architecture for modular construction of enterprise applications.

EJB History

- **EJB 1.0 (1998)**
- **EJB 1.1 (1999), J2EE 1.2**
  - XML descriptors, role driven security, Entity EJB
- **EJB 2.0 (2001), JSR 19, J2EE 1.3**
  - Message-Driven Beans, local interfaces, RMI-IIOP
- **EJB 2.1 (2003), JSR 153, J2EE 1.4**
  - Web services support, Timer Service, aggregation support in EJB-QL
- **EJB 3.0 (2006), JSR 220, Java EE 5**
  - Simpler development, Annotations, POJO components, convention-over-configuration, JPA, Entity Beans and home interfaces dropped.
- **EJB 3.1 (2009), JSR 318, Java EE 6**
  - Singletons, Local view, war packaging, EJB Lite, Portable JNDI names, App init and shutdown events, Time Service enhancements, @Asynchronous, embeddable EJB

EJB Container

- **Provides services to components**
  - Lifecycle control
  - Dependency injection
  - Transaction management
  - Remote access
  - Access control

- **You need application server with EJB Container**
  - Java EE 6 Full Profile
  - Java EE 6 Web Profile (only EJB Lite)

EJB Components

- **Entity Beans**
  - Deprecated in EJB 3.0, replaced with JPA
- **Session Beans**
  - Stateless
  - Statefull
  - Singleton
- **Message-Driven Beans**

AOP / Interceptors

```
public class MyInterceptor {
  @AroundInvoke
  public Object methodName(InvocationContext
    invocationContext)throws Exception {

    // Do, what we need…
    Object result =  invocationContext.proceed();
    // Do, what we need…
    return result;
}

@Interceptors({MyInterceptor.class})
// This annotation can be used on whole class or
// on only some methods
```

Transaction Management

- **Transactions are controlled by JTA**
  - Global transactions
  - Distributed transactions
- **Container Managed Transactions**
  - Declarative approach
  - Controlled with annotations
- **Bean Managed Transactions**
  - Imperative approach
  - Controlled with code

Transaction Management

Transaction attributes

TransactionAttributeType	Transaction already in progress	No transaction in progress
MANDATORY	Current transaction is used.	Exception is thrown.
NEVER	Exception is thrown.	No transaction is used.
NOT_SUPPORTED	Current transaction is suspended	No transaction is used.
REQUIRED	Current transaction is used.	New transaction is created.
REQUIRES_NEW	Current transaction is suspended, new transaction is created.	New transaction is created.
SUPPORTS	Current transaction is used.	No transaction is used.

Security Management

- **javax.annotation.security**
  - @DeclareRoles
  - @DenyAll
  - @PermitAll
  - @RolesAllowed
  - @RunAs

- **http://www.packtpub.com/article/hands-on-tutorial-ejb-security**

Timer Service

- @Schedule

CDI

- **Commons and Dependency Injection**
  - More flexible than standard Dependency Injection on Web, EJB, or JSF components (Qualifiers, Stereotypes, etc.)
  - Simplifies integration

EJB 3.1 versus Spring: General

- **Java EE 6 / EJB 3.1**
  - Platform
  - JCP Standard
  - Both free and commercial implementation available
  - Standard approaches for common problems
  - More convention-over-configuration principle, simpler configuration
  - EJB Container is requires
- **Spring**
  - Framework
  - Proprietary
  - Free, Open Source
  - Less invasive, you can choose your favorite approach for each problem
  - More flexible, but more complicated configuration
  - Just Spring IoC container required (part of framework)