

# Enterprise Java Beans 3.0

## *Ease of Use*


**AJUG**  
**10/19/04**

Goals of EJB 3.0

JDK 5.0 Annotations

Session Beans

Entity Beans



## Goals of EJB 3.0


Professional Open Source

EJB 2.1 is too “noisy”

- Too many interfaces to implement
- “XML Hell” too many complex deployment descriptors
- API is too verbose
- API too complicated in general

Simplify the EJB programming model  
Focus on ease of use  
Facilitate Test Driven Development  
Make it simpler for average developer  
Increase developer base

© JBoss Group, 2003. 3



## JDK 5.0 Annotations

Professional Open Source

XDoclet like metadata, C#-like metatags


- Typesafe, compiler-checked

Syntax you can add to the Java language  
Metadata you can attach to class, method, field, constructor, or parameter

- Metadata is compiled into class file
- Available at compile time and/or runtime

EJB 3.0 makes extensive use of annotations to replace XML

© JBoss Group, 2003. 4


 **JDK 5.0 Annotations**

Professional Open Source

```
public @interface MyMetadata {
    String value() default "hello";
}
-----
import org.acme.MyMetadata;

@MyMetadata("stuff") public class MyClass {
    @MyMetadata public void someMethod() {...}
}
```

© JBoss Group, 2003. **5**

 **JBoss Group**

Professional Open Source

## Session Beans

Compare EJB 2.1 vs. EJB 3.0

© JBoss Group, 2003. 20 October 2004. **6**

## EJB 2.1: Requirements

Professional Open Source

- Home interface
- Remote/Local interface
- Bean class must implement javax.ejb.SessionBean
- XML Deployment descriptor

© JBoss Group, 2003. 7

## EJB 2.1: Required Interfaces

Professional Open Source

- Homes for stateless beans unnecessary
- Remote interface must inherit from EJBObject
- Remote methods must throw RemoteException
  - Dependency on RMI

```
public interface CalculatorHome extends javax.ejb.EJBHome {
    public Calculator create() throws CreateException;
}

public interface Calculator extends EJBObject {
    Public int add(int x, int y) throws RemoteException;
    Public int subtract(int x, int y) throws RemoteException;
}
```

© JBoss Group, 2003. 8

## EJB 2.1: Bean class requirements

Professional Open Source

Must extend verbose javax.ejb.SessionBean  
Unnecessary and verbose callback methods

```
public class CalculatorBean implements javax.ejb.Sessionbean {
    private SessionContext ctx;
    public void setSessionContext(SessionContext ctx) { this.ctx = ctx; }

    public void ejbCreate() {}
    public void ejbRemove() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}

    public int add(int x, int y) {
        return x + y;
    }

    public int subtract(int x, int y) {
        return x - y;
    }
}
```

© JBoss Group, 2003. 9

## EJB 2.1: XML Deployment Descriptor

Professional Open Source

```
<session>
  <ejb-name>CalculatorBean</ejb-name>
  <home>org.acme.CalculatorHome</home>
  <bean>org.acme.CalculatorBean</bean>
  <remote>org.acme.CalculatorRemote</remote>
  <session-type>Stateless</session-type>
  <transaction-type>Container</transaction-type>
</session>

....
```

© JBoss Group, 2003. 10

## EJB 3.0: Goals

Professional Open Source


- Remove unnecessary interfaces
- Remove unnecessary callbacks
- Make deployment descriptors optional
- Make beans pojo-like
- Use default values where they make sense

## Required Interfaces

Professional Open Source


- Homeless
- Methods don't throw RemoteException
- No verbose interface implementations

```
@Remote public interface Calculator {  
    public int add(int x, int y);  
    public int subtract(int x, int y);  
}  
  
@Stateless Public class CalculatorBean implements Calculator {  
    public int add(int x, int y) {  
        return x + y;  
    }  
    public int subtract(int x, int y) {  
        Return x - y;  
    }  
}
```

 **EJB 3.0: XML Deployment Descriptor**

Professional Open Source

© JBoss Group, 2003. 13

 **Transactions and Security**

Professional Open Source

```
@Stateless public class AccountDAOBean implements Calculator {  
  
    @Tx(TxType.REQUIRED)  
    @MethodPermission({"teller"})  
    public void withdraw(double amount) {  
        ...  
    }  
}
```

© JBoss Group, 2003. 14

## EJB 3.0: Dependency Injection

Professional Open Source

- Bean class specifies dependencies instead of lookup
- Facilitates Test Driven Development
- Possible to test EJBs outside of container

```
@Stateful public class ShoppingCartBean implements ShoppingCart {
    @Inject private SessionContext ctx;

    private DataSource jdbc;

    @Resource(jndiName="java:DefaultDS")
    public void setDataSource(DataSource db) { this.jdbc = db; }

    public void buy(Product product) {
        ...
    }
}
```

© JBoss Group, 2003.

15

## EJB 3.0: Callbacks on demand

Professional Open Source

- Callback methods still available
- Annotate on an as-needed basis

```
@Stateless public class FacadeBean implements Facade {
    @EjbTimeout void licenseExpired(Timer t) {...} // committee is leaning towards this

    public void ejbCreate() {} // currently in public draft
}
```

© JBoss Group, 2003.

16

## Entity Beans

POJO based persistence

- Same goals as session beans
  - Fewer interfaces, optional XML DDs, etc.
- No required interfaces or subclassing
- Plain Java based
  - Allow new()
- Solve/remove value object anti-pattern
- Provide full Object/Relational mapping
- Supports Inheritance
- Expand EJBQL
  - Fully featured
  - Parallel SQL
  - Polymorphic Queries

### O/R Mapping Metadata as annotations

- Table mappings, @Table, @SecondaryTable
- Column mappings, @Column, @JoinColumn
- Associations, @ManyToOne, @OneToOne, @OneToMany, @ManyToMany
- Inheritance, @Inheritance, @DiscriminatorColumn
- Identifier + Version properties, @Id, @Version

```
@Entity
@Table(name="AUCTION_ITEM", schema="AUCTION")
public class Item {
    private Long id;
    private int version;
    private String description;
    private Set<Bid> bids = new HashSet();
    private User seller;

    @Column(name="DESC", nullable=false, length=500)
    public String getDescription() {
        return description;
    }
    public void setDescription(String desc) {
        this.description = desc;
    }
}
....
```

Entity Annotations

Professional Open Source

```
...  
  
@Id @Column(name="ITEM_ID")  
public Long getId() {  
    return id;  
}  
public void setId(Long id) {  
    this.id = id;  
}  
  
@Version  
public int getVersion() {  
    return version;  
}  
protected void setVersion(int version) {  
    this.version = version;  
}  
  
...
```

© JBoss Group, 2003. 21

Entity Annotations

Professional Open Source

```
@ManyToOne(fetch=LAZY)  
@JoinColumn(name="USER_ID", nullable=false, updatable=false)  
public User getUser() {  
    return user;  
}  
  
protected void setUser(User user) {  
    this.user = user;  
}  
  
@OneToMany(cascade=ALL)  
@JoinColumn(name="ITEM_ID", nullable=false, updatable=false)  
protected Set<Bid> getBids() {  
    return bids;  
}  
protected void setBids(Set<Bid> bids) {  
    this.bids = bids;  
}
```

© JBoss Group, 2003. 22

Detyped Home interface

- Entities have no homes

Analogous to Hibernate Session

create(), remove(), and merge()

createQuery(), find()

Factory for query objects

All access to Entities through this service

```
@Session public class ItemDAOImpl {  
  
    @Inject private EntityManager em;  
  
    public Item findById(Long itemId) {  
        return (Item) em.find("Item", itemId);  
    }  
  
    public Long create(Item item) {  
        em.create(item); //item is now managed  
        return item.getId();  
    }  
  
    public void merge(Item item) {  
        em.merge(item);  
    }  
}
```



## Entities as Value Objects

Professional Open Source

- Entities are plain java objects
  - Solve value object anti-pattern
  - Entities can become value objects
  - Entities can be detached and reattached to persistence storage
    - Can be serialized to remote client
    - Remote client can perform updates on local copy
    - Copy can be sent back to server and merged back in
- EntityManager.merge()



## Query API

Professional Open Source

- Queries may be expressed as EJBQL strings
  - Embedded in code
  - Externalized to metadata (named queries)
- Invoke via `Query` interface
  - Named parameter binding
  - Pagination control

```
@Session public class ItemDAOImpl {  
    ...  
    public List findByDescription(String description, int page) {  
        return em.createQuery("from Item i where i.description like :d")  
            .setParameter("d", description)  
            .setMaxResults(50)  
            .setFirstResult(page)  
            .listResults();  
    }  
    ...  
}
```

EJBQL 3.0 is very similar to HQL (Hibernate Query Language)

Aggregation, projection

- `select max(b.amount) from Bid b where b.item = :id`
- `select new Name(c.first, c.last) from Customer c`

Fetching

- `from Item i left join fetch i.bids`

Subselects

- `from Item i join i.bids bid where bid.amount = (select max(b.amount) from i.bids b)`

Group By, Having, Joins

Persistence mapping supports inheritance

- Single table per hierarchy – `SINGLE_TABLE`
- Join table per subclass – `JOINED`
- Distinct table per subclass – `UNION`

Queries on class hierarchy are polymorphic

## Inheritance – SINGLE\_TABLE

Professional Open Source

```
@Entity
@Table(name="Animal")
@Inheritance(strategy=InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name="TYPE")
public class Animal {
    @Id private int id;
    @Column(name="AVG_WEIGHT")
    private int averageWeight;
    ...
}

@Entity
public class Dog extends Animal{
    @Column(name="BREED")
    private String breed;
    ...
}
```

© JBoss Group, 2003. 29

## Inheritance – SINGLE\_TABLE

Professional Open Source

```
create table Animal
(
    ID Number,
    TYPE varchar(255),
    AVG_WEIGHT Number,
    BREED varchar(255)
);
```

© JBoss Group, 2003. 30

**Inheritance – JOINED**

Professional Open Source

```
@Entity
@Inheritance(strategy=InheritanceType.JOINED)
@DiscriminatorColumn(name="TYPE")
@Table(name="Animal")
public class Animal{
    @Id private int id;
    @Column(name="AVG_WEIGHT")
    private int averageWeight;
    ...
}

@Entity
@InheritanceJoinTable(name="DOGGY_ID")
@Table(name="Doggy")
public class Dog extends Animal{
    @Column(name="BREED")
    private String breed;
    ...
}
```

© JBoss Group, 2003. 31

**Inheritance – JOINED**

Professional Open Source

```
create table Animal
(
    ID Number,
    TYPE varchar(255),
    AVG_WEIGHT Number
);

create table Doggy
(
    DOGGY_ID Number,
    BREED varchar(255)
);
```

© JBoss Group, 2003. 32

## Inheritance – UNION

Professional Open Source

```
create table Kitty
(
  ID Number,
  TYPE varchar(255),
  AVG_WEIGHT Number
  BREED varchar(255),
  LIVES Number
);

create table Doggy
(
  DOGGY_ID Number,
  TYPE varchar(255),
  AVG_WEIGHT Number
  BREED varchar(255)
);
```

© JBoss Group, 2003. 33

## EJB3 Demo

Professional Open Source

Demo from JBoss EJB3 tutorial

- Stateful Session Bean
- Entity Bean
  - Using Join Inheritance

© JBoss Group, 2003. 34



### EJB3 = Easy Enterprise Java Beans

#### Powerful

- Same feature set as EJB 2.1
- Standards based (JSR-220)

#### Easy

- To develop
- To test
- To deploy

#### References

- EJB 3.0 Specification (early draft)
  - [http://docs.jboss.org/ejb3/ejb-3\\_0-edr-spec.pdf](http://docs.jboss.org/ejb3/ejb-3_0-edr-spec.pdf)
- JBoss EJB 3.0 Preview Release
  - <http://www.jboss.org/ejb3>