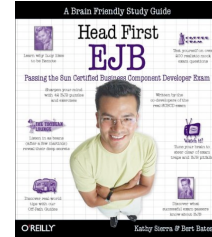


EJB

William Pugh

1

Recommended books



2

What are EJB?

- Some RMI
- Some software component technology/terminology
- Ability to intercept calls and add system-level functionality
- Functionality that can be added
 - transactions, security, activation/passivation
 - ...

3

Kinds of beans

- Entity beans
 - represents thing in persistent store
- Message-driven beans
 - listeners
- Session beans
 - stateless
 - stateful

4

Entity Beans

- Represents a thing in the persistent store
 - a row in a table
 - A noun
- Cached in memory
 - not all persistent entities are materialized as entity beans
- You can search for entity beans
- Deleting a bean deletes the entity from the persistent store

5

Message-driven

- Clients can't get references to message driven beans
- Attached to messaging service
- Listen for certain kinds of messages
- Provides asynchronous message invocation

6

Stateless Session Beans

- Handles one request from a client
- Then handles another request from a different client
- Client can't come back to the same stateless session bean
- Typically, represent verbs
 - verify credit card
 - perform purchase

7

Stateful Session Beans

- A client gets a persistent reference to a stateful session bean
- Can retain state across several method calls
- Can't search for a stateful session bean
- Not durable: after timeout, or machine crash, they just go away

8

Bean clients

- Clients can only get (indirect) references to session and entity beans
- Can create messages that are processed by message driven beans

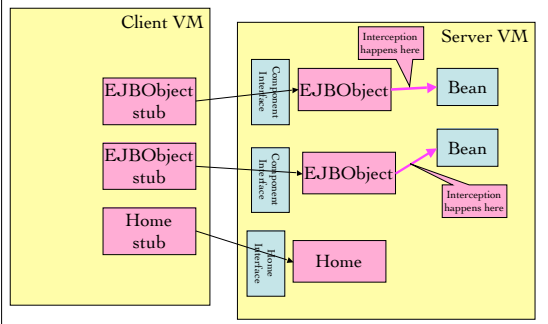
9

Home sweet home

- Beans have a home interface
- Home interface allows beans to be created, located and removed
- Factory design pattern
 - I would have preferred factory to home as a naming convention
 - But we are stuck with home
- User specifies interface only, no code

10

EJB Architecture



Assumptions

- Initially, we'll assume that all calls to EJB's are from remote machines
 - at least, via RMI
 - assumption relaxed in EJB 2.0, we'll talk about it later
- Clients never get to talk directly to a bean
 - would allow circumvention of security and transaction checks

12

Component interface

- Called remote interface in a lot of the literature
 - made more sense when all client references were remote
- User specifies interface only, no code

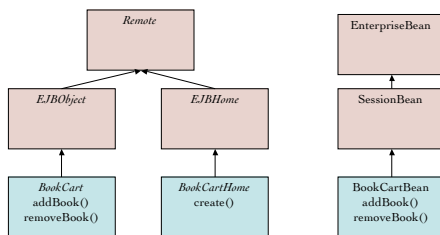
15

Responsibilities

- You provide
 - Component interface
 - Home interface
 - Bean implementation class
- Container generates
 - Home implementation and stubs
 - EJBObject implementation and stubs

14

Creating a Stateful Session Bean



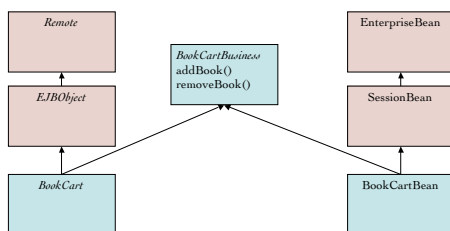
15

Bean doesn't implement component interface

- Typically, the bean doesn't implement the component interface
- A bean is not an instance of a something that can be invoked by a client
- How do you know that a bean can handle all the methods supported by the interface?
 - tools

16

If this worries/bothers you



17

XML Deployment Descriptions

- How does the app server know which class is which?
- All specified in an XML file
- J2EE loves XML
- writing it by hand is possible, but tedious and rarely done

18

Hello world as an EJB

- Stateless Session Bean that gives Advice

19

Component Interface

```
import javax.ejb.*;
import java.rmi.RemoteException;

public interface Advice extends EJBObject {

    public String getMessage()
        throws RemoteException;

}
```

20

Home Interface

```
import javax.ejb.*;
import java.rmi.RemoteException;

public interface AdviceHome extends EJBHome {

    public Advice create()
        throws CreateException, RemoteException;

}
```

21

Bean Implementation

```
public class AdviceBean implements SessionBean {

    private String[] adviceStrings = {"test", "test1", "test2", "test3"};

    public String getMessage() {
        System.out.println("in get advice");
        int random = (int) (Math.random() * adviceStrings.length);
        return adviceStrings[random];
    }

    public void ejbCreate() {
        System.out.println("in ejb create");
    }

}
```

22

XML deployment descriptor

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar xmlns="http://java.sun.com/xml/ns/j2ee" version="2.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
  http://java.sun.com/xml/ns/j2ee/.ejb-jar_2.1.xsd">
  <display-name>Ejb1</display-name>
  <enterprise-beans><session>
    <ejb-name>AdviceBean</ejb-name>
    <home>headfirst.AdviceHome</home>
    <remote>headfirst.Advice</remote>
    <ejb-class>headfirst.AdviceBean</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Bean</transaction-type>
    <security-identity><use-caller-identity/></security-identity>
  </session></enterprise-beans>
</ejb-jar>
```

23

EJBObject interface

EJBHome	getEJBHome()
Handle	getHandle()
Object	getPrimaryKey()
boolean	isIdentical(EJBObject obj)
void	remove()

24

EJBHome interface

```
EJBMetaData getEJBMetaData()
HomeHandle  getHomeHandle()
void        remove(Handle handle)
void        remove(Object primaryKey)
```

25

Handles

- Handles are persistent references to EJBObjects
- Can be serialized, passed between machines
 - stored in Servlet sessions

26

isIdentical

- Used to determine whether two EJBObject stubs refer to the same bean
- Stateless session beans
 - true if they come from the same home
- Stateful session beans
 - false for any two distinct stubs
- Entity beans
 - true if entities have same primary key

27

Session/Entity Bean Creation

- Home interface must have create(...) methods
- EJB must have matching ejbCreate(...) interfaces
- Stateless session beans should have only no-argument create methods

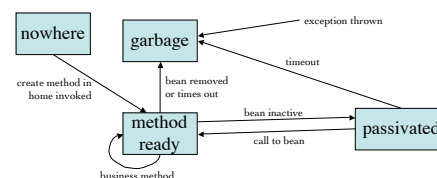
28

Removal

- Removing an entity bean deletes the corresponding info from persistent store
- Removing a stateful session bean says that you are done with it
 - removing a stateless session bean is a no-op (or an error?)

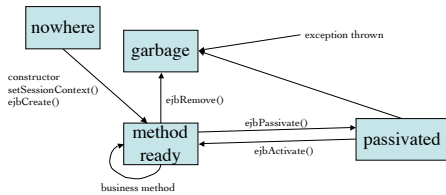
29

Stateful Session Bean Lifecycle



30

Lifecycle notications



31

Stateless Session Beans

- No state that lasts longer than a single call from a client
 - Only one call at a time on a stateless session bean
 - Can have instance variables / state that persist for the duration of one call
 - Can keep resources, debugging info, caches
- Different invocations from a client can be routed to different beans

32

Local interfaces

- EJB2.0 adds local interfaces for components and homes
 - extend `EJBLocalHome` and `EJBLocalObject`
- Local objects don't have handles
- Call by reference rather than deep copy
- Methods don't throw remote exception

33

Mix and Match

- No need for one bean to support both Local and Remote interfaces?
 - rather different
- Client of bean must know what it is getting

34

Container Managed Persistence

- You can write all of the code yourself to move data between entity beans and the database
 - but you don't want to
- Container managed persistence allows the EJB container to write all the code

35

Container managed relations

- Simple case as things like `String` and integer values
- CMR allows mapping of 1-1, 1-many, many-1 and many-many relationships
- Treated as references, sets, etc.

36

Transactions

- You can use container managed transactions or bean managed transactions
 - container managed transactions are specified declaratively
 - bmt are done by starting and ending transactions

37