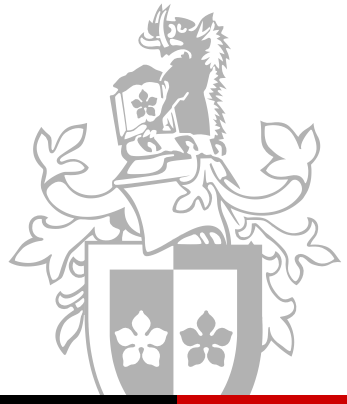
COS30041 Creating Secure and Scalable Software

Lecture 04c StateLess Session Bean, SLSB



SWINBURNE UNIVERSITY OF TECHNOLOGY

Learning Objectives

- After studying the lecture material, you will be able to
  - Understand and describe what session bean is
  - Explain why there is a need to have two different types of session bean
  - Describe what a stateless session bean is
  - Explain the life cycle of a stateless session bean
  - Understand the issues involved in programming stateless session beans
  - Program stateless session bean and the client application that calls the services provided by stateless session bean

Outline

- Session Bean
- Stateless Session Bean
- Programming Stateless Session Bean

Roadmap

- **Session Bean**
- Stateless Session Bean
- Programming Stateless Session Bean

Session Bean, SB

- An EJB that models the business processes
  - Business logic
  - Business rules
  - Algorithms
  - Workflow
- Example
  - accessing bank account
  - verifying credit card details
  - preparing an invoice

Lifetime of a Session Bean

- The lifetime of a session
- Or, the lifetime of the client code calling the session bean
- Examples
  - The time of a browser window is open
  - The time of your Java applet is running
  - A standalone client application is open
  - Another enterprise bean is using your session bean
- The EJB container will destroy session beans if clients time out

General Issues

- A session bean cannot be shared between clients
- Session beans do not represent data in a database
- Session beans are transaction aware

Different types of Session Bean

- StateLess session bean, SLSB
  - Does not hold conversation between the client and the bean
- StateFul session bean, SFSB
  - Hold conversation between the client and the bean
- Singleton session bean (new in Java EE 6)
  - Hold common information within the entire application
  - All clients share this information

Roadmap

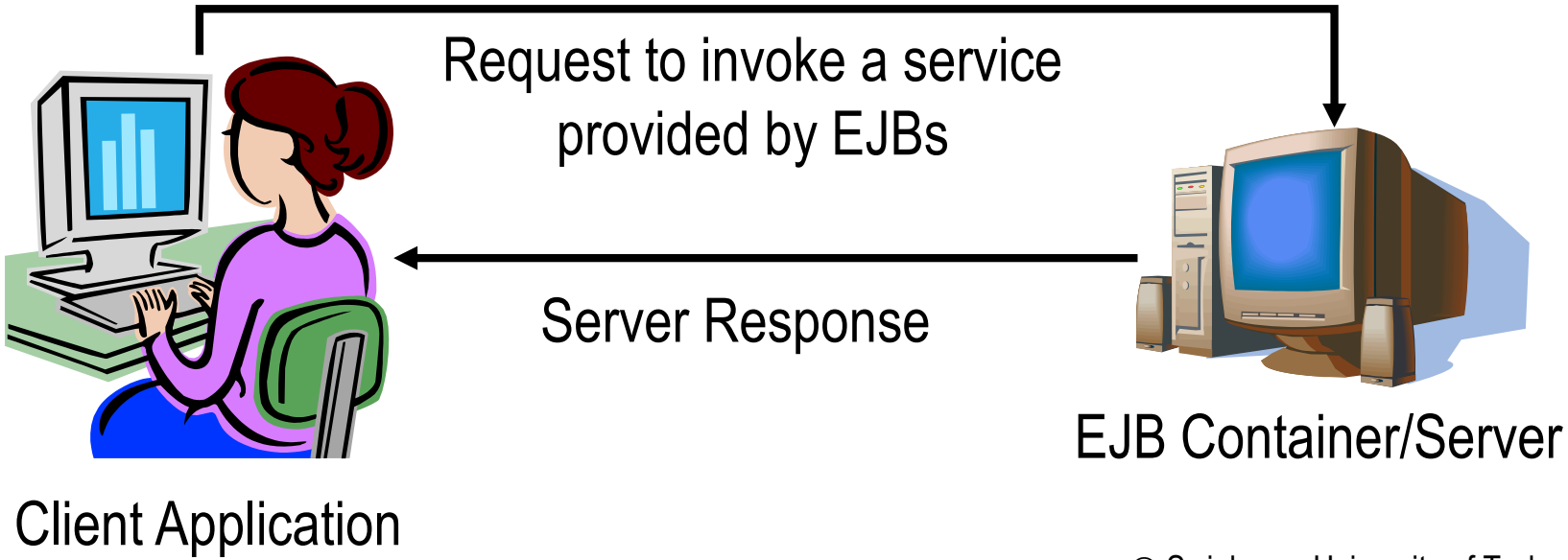
- Session Bean
- **Stateless Session Bean**
- Programming Stateless Session Bean

Stateless Session Bean

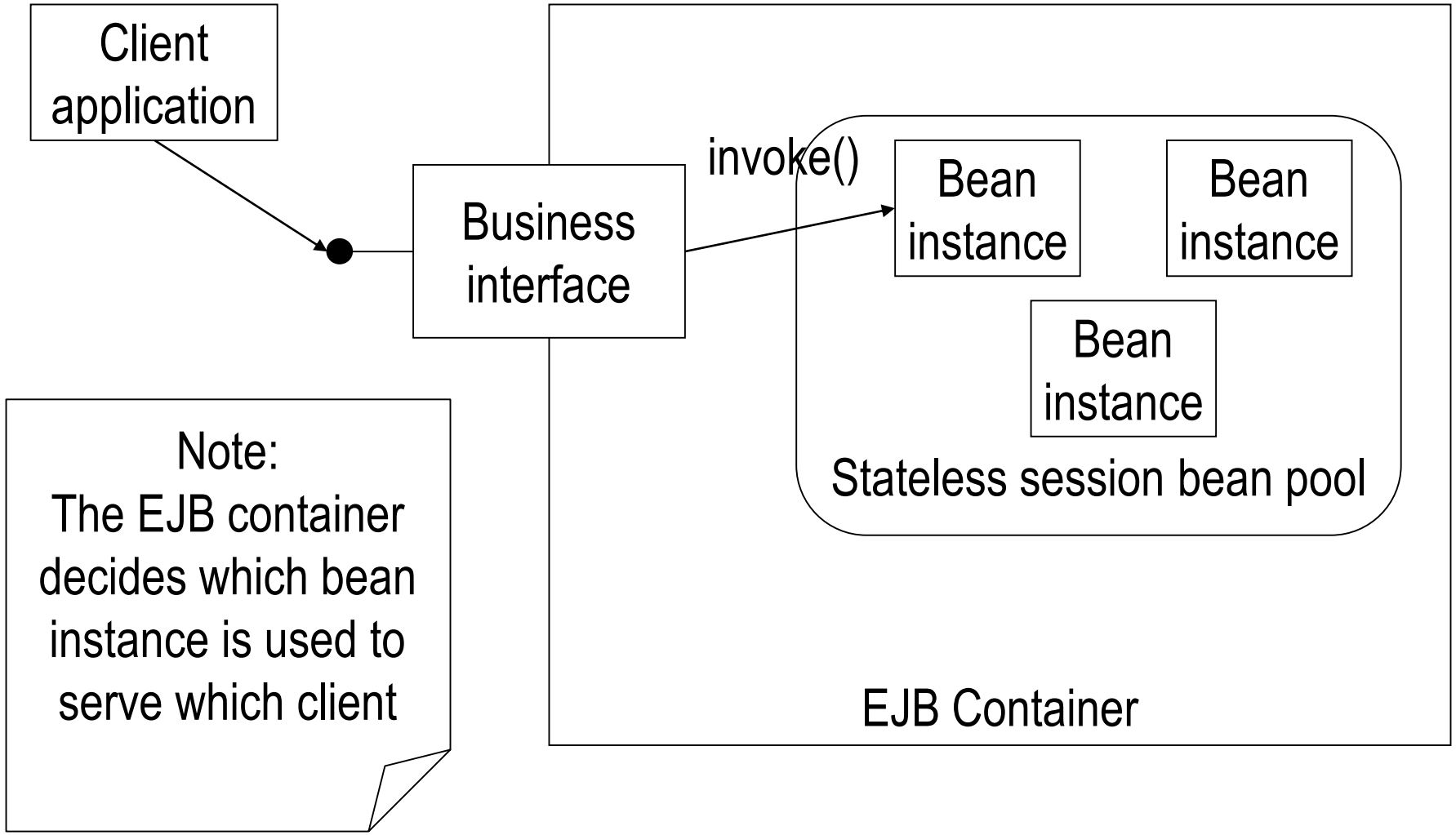
- Stateless = does not remember
- The conversation between client and the stateless session bean only spans a ***single*** method request
- Do not hold multi-method conversations with clients
- Example
  - A session bean for verifying credit card transactions

Stateless SB – Typical Invocation

- A client makes a request to invoke a service provided by a session bean
- The session bean resides in the EJB container/server
- The EJB container intercepts and processes the request
- The EJB container then sends the result back to the client



Stateless SB – Typical Invocation (cont'd)



Adapted from Fig. 4.1 of [MEJB,p.94]

Why Stateless Session Bean?

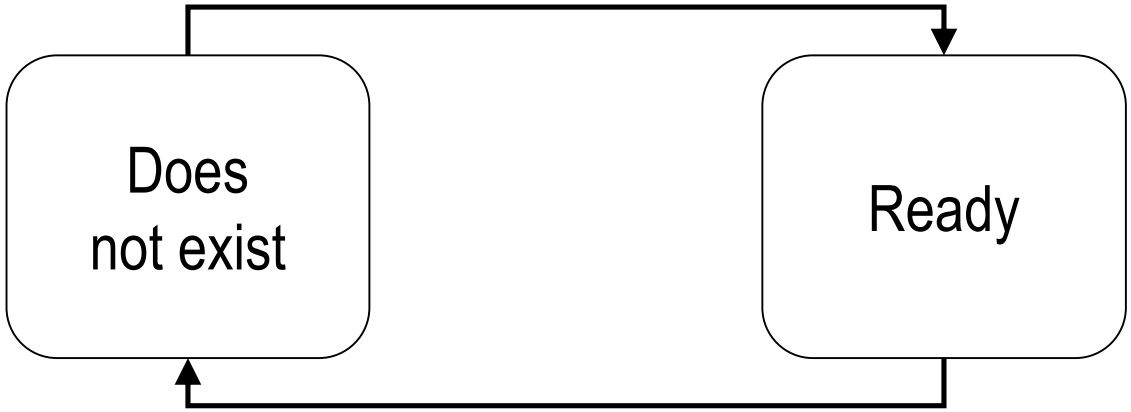
- Good for business needs where client dialogue is not needed
- Typically for requests where the request is made and the answer received without the need for intermediate stages of input and response
- Offer better scalability when compared with stateful session beans
  - Serve same number of clients with lesser stateless session beans

Life Cycle of Stateless Session Bean

- Two possible states
  - "Does not exist"
    the bean instance does not exist in the EJB container
  - "Ready for method call"
    the bean instance exists in the EJB container and it is ready to receive any method calls from client
- See Figure 32-3 in [JEE7T, p.32-12]

Life Cycle of Stateless Session Bean (cont'd)

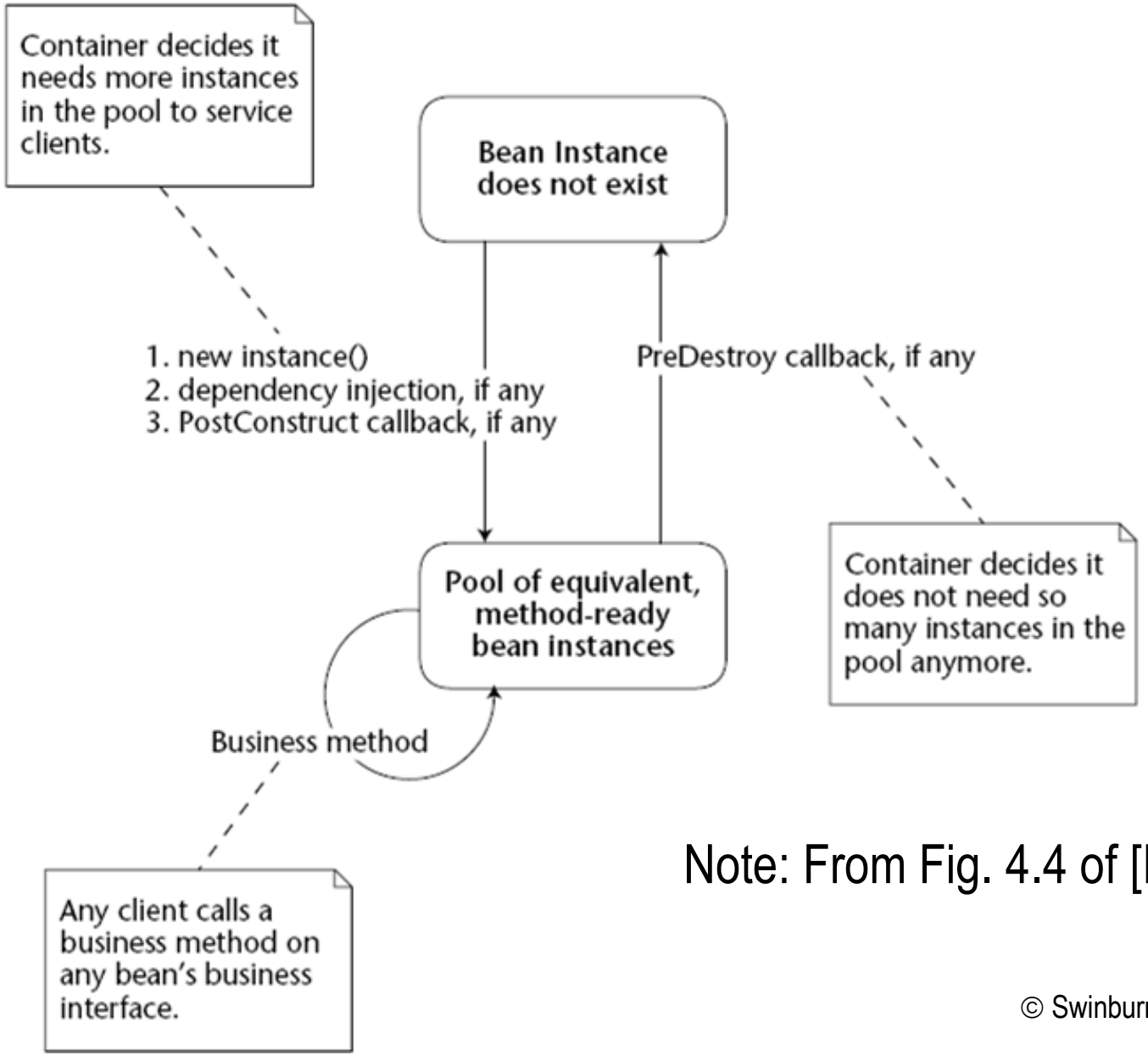
1. EJB container decides when to create a stateless session bean
2. Once created, the stateless session bean instances will be put in the "Ready" status – waiting to serve client's request



3. EJB container decides when to destroy a stateless session bean
4. Once destroyed, the stateless session bean instances does not exist

Note: Adapted from Fig. 32-3 of [JEE7T,p.32-12]

Life Cycle of Stateless Session Bean (cont'd)



Note: From Fig. 4.4 of [MEJB,p.111]

Pooling with Stateless Session Beans

- Achieving scale
- EJB container/server normally manages many instances of Stateless Session Beans for efficiency and scalability concerns
- Pooling with stateless session beans are simple because all instances of the same stateless session bean class are, in fact, equivalent and indistinguishable to a client

Roadmap

- Session Bean
- Stateless Session Bean
- **Programming Stateless Session Bean**

Programming Stateless Session Bean

The Client Side

Some programs that request the services of a stateless session bean

- A client application
- A web page (e.g. Servlets, JSPs)

The Server Side

- Stateless session bean (EJB)

Download the example – EX-SLSB-HelloWorld.zip

Prog. SLSB – Server side – Java EE 5/6/7 specific

- The EJB class (or, simply the Bean class)
- The Remote interface class
- The Local interface class
- The Deployment Descriptor
- The Vendor-Specific Files
- The EJB-JAR File

[NB7.1 or later] need to put this in a separate package outside the EAR / EJB-JAR

Done by Sun's Application Server or some IDE

Programming the Remote Interface – Java EE 5/6/7

- A simple plain old Java interface (POJI)
- Use the annotation "`@Remote`" to indicate the interface class is a remote interface for EJB
  - Example: ([NB6.9.1 or later])

```
@Remote
public interface HelloWorldBeanRemote { … }
```

- Expose every business method about the stateless session bean for remote client applications
  - Name the methods and their corresponding parameters
  - Example

```
public String getGreetings(String name);
```

Programming the Local Interface – Java EE 5/6/7

- A simple POJI
- Use the annotation "`@Local`" to indicate the interface class is a local interface for EJB
  - Example: ([NetBeans 6.9.1 or later])

```
@Local
public interface HelloWorldBeanLocal { … }
```

- Expose every business method about the stateless session bean for local client applications
  - Name the methods and their corresponding parameters
  - Example

```
public String getGreetingsLocal(String name);
```

Programming the Bean class – Java EE 5/6/7

- A simple plain old Java object (POJO) implementing the business methods
- Use "`@Stateless`" to indicate that it is a Stateless SB
- [Optional in NB7.1] Use "`@Remote(...) / @Local(...)`" to indicate the corresponding remote/local interfaces
- Example:

Optional in NB7.1

```
@Stateless
@Remote(HelloWorldBeanRemote.class)
@Local(HelloWorldBeanLocal.class)
public class HelloWorldBean { … }
```

Prog. the Bean class – Java EE 5/6/7 (cont'd)

- Program the business methods as defined in Remote and Local Interfaces
  - Example

```
public String getGreetings(String name) {
        return "Hello, " + name + "!";
}
```

- Should avoid same method in both Remote and Local interfaces

Prog. Stateless SB – Server – Example

- WANT: a Stateless Session EJB "`HelloWorldBean`" to greet user with `username` (default is "`World`")

Prog. SLSB – Server – Example (Java EE 5/6/7)

Example: ([NetBeans 6.9.1 or later])

- The Remote Interface class
  - See EJB-Session-Stateless demo, `HelloWorldBeanRemote.java`
- The Local Interface class
  - See EJB-Session-Stateless demo, `HelloWorldBeanLocal.java`
- The Bean class
  - See EJB-Session-Stateless demo, `HelloWorldBean.java`

Programming Stateless SB – Client

- The application client
  - An application that
    calls the business methods of an EJB object remotely and then displays the returned results locally
  - [NB6.9.1 or earlier] When compiling client application in a different machine, you need the EJB-JAR file from the EJB developed on the server
  - [NB7.1 onwards] Since the remote interface is now in a separate package, you need to include its JAR file in the application client development

Programming the Client for Stateless SB

- Naming Convention: Stateless Session EJB – SBean
  - The Client Application – SClient ([JEExT])
  - NetBeans uses “Main.java” but I renamed it to “SAppClient.java” for consistency purposes in the sample code
- Example: Stateless Session EJB – HelloWorldBean
  - The Client Application – HelloWorldAppClient

Prog. the Client for SLSB – Java EE 7

- [NetBeans 8.2] Use
  "`@EJB private static SBeanRemote sBean;`"
  to declare the required SLSB, "`SBean`" –
  as a variable whose name is "`sBean`" in the client app
- Example:
  ```
  @EJB private static HelloWorldBeanRemote helloWorldBean;
  ```
- Call the business methods of the EJB as usual
  Example:
  ```
  helloWorldBean.getGreetings(name);
  ```

Prog. the Client for Stateless SB – Example

Example: ([NetBeans 6.9.1 or later])

- WANT: a client to call the business methods provided by the "`HelloWorldBean`" stateless session EJB
- Client Application
  - See EX-SLSB-HW-AppClient demo, `HelloWorldAppClient.java`

Deploying the EJB

- This involves
  - Preparing the deployment descriptor,
  - Preparing any vendor-specific files, and
  - Packaging the EJB-JAR file
- NetBean IDE handles these steps automatically
  - Sun's Application Server can only prepare Sun-Specific files
  - Need to consult the corresponding vendors for their specific files

Running the Client Application

- This involves
  - deploying the EJB services on an EJB container/server, and
  - executing the Client Application

References

- [MEJB] R.P. Sriganesh, G. Brose, M. Silverman (2008) *Mastering Enterprise JavaBeans 3.0*, 4th ed., John Wiley & Sons
  - Chapters 3 and 4
- [JEE7T] E. Jendrock et al. (2014) *The Java EE 7 Tutorial*, Oracle, June 2014
  - Chapters 32 – 34
  - Earlier versions
    - [JEE6T] E. Jendrock et al. (2010) *The Java EE 6 Tutorial*, Oracle, June 2010; Chapters 14 – 16
    - [JEE5T] E. Jendrock et al. (2008) *The Java EE 5 Tutorial Update 5*, Sun Microsystems, Oct 2008; Chapters 20 and 21