11

Deploying Java EE Applications

Objectives

After completing this lesson, you should be able to:

- Describe Java EE Web applications
- Describe Enterprise Deployment Architectures
- Package Web applications in several forms
- Define Web application structure and Web application archive
- Explain why WebLogic augments standard Java EE deployment descriptors with `weblogic*.xml` files
- Look at deployment descriptors `web.xml` and `weblogic.xml`
- Describe URLs and Web applications

Road Map

- Web applications
  - Web applications
  - Directory structure and deployment descriptors
  - Using the Console to deploy Web applications
  - Monitoring Web applications
- EJB applications
- Enterprise applications

Java EE Web Applications

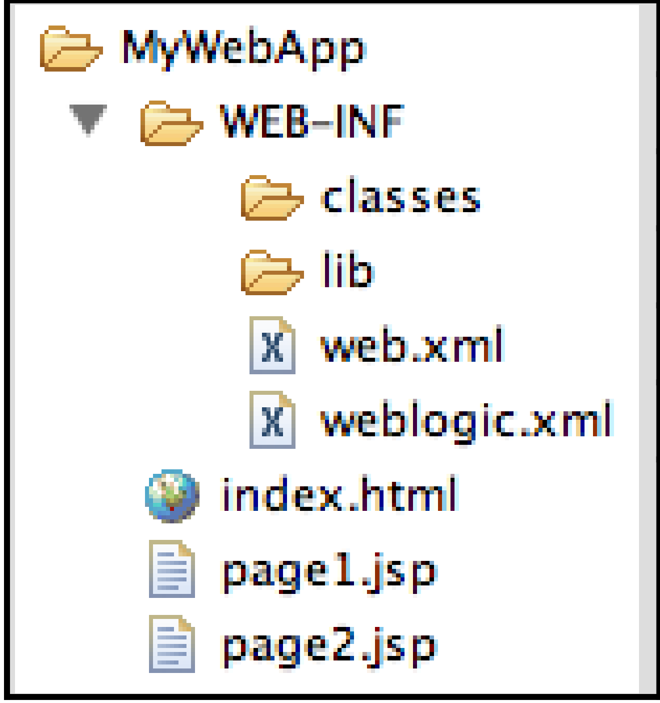
- Web application:
  - Responds to client requests using the HTTP protocol
  - Typically implements an interactive Web site
- The contents of a Web application can include:
  - Java servlets
  - JavaServer Pages (JSPs) for dynamic content
  - Static content (HTML, CSS, images, and so on)
  - Java classes and libraries
  - Client-side libraries (JavaScript, Java Applets, and so on)
  - XML deployment descriptors:
    - Standard (`web.xml`)
    - WebLogic specific (`weblogic.xml`)

Packaging Web Applications

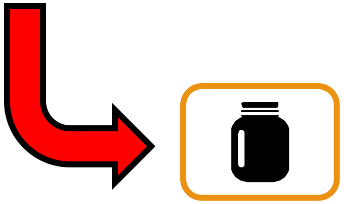
You should package an application before it can be deployed to Oracle WebLogic Server. To appropriately package a Web application, perform the following steps:

1. Arrange the resources in a prescribed directory structure.
2. Develop or copy the `web.xml` deployment descriptor (optional).
3. Develop or copy the `weblogic.xml` deployment descriptor (optional and WLS specific).
4. Archive the Web application into a `.war` file using Java Archive (JAR).
5. Deploy the Web application onto Oracle WebLogic Server.
6. Configure the Web application with the Oracle WebLogic Server Administration Console.

Web Application Structure



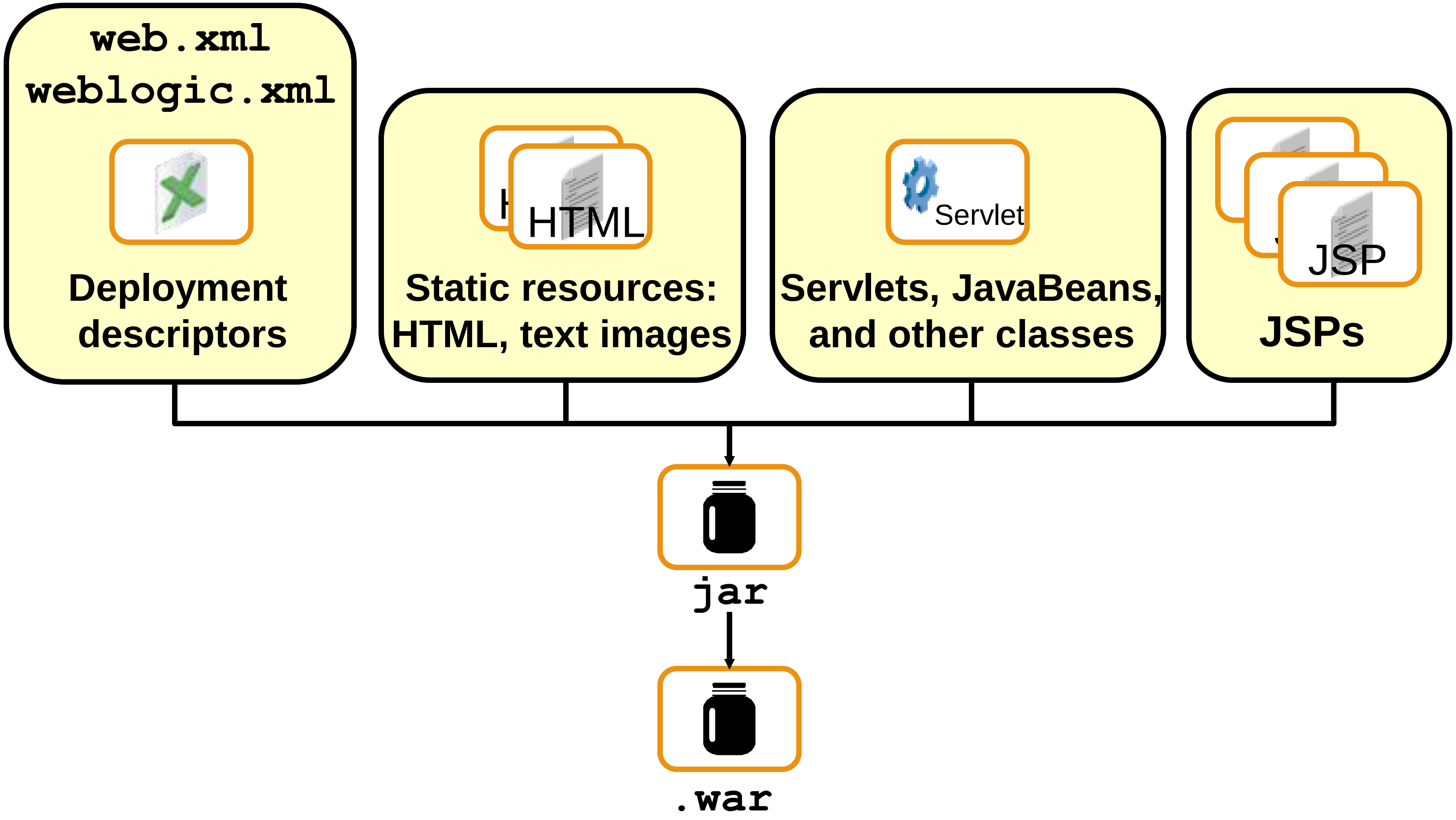
Directory or File	Description
MyWebApp	Public document root of Web application
WEB-INF	Private resources not served directly to clients
classes	Classes, such as servlets, filters, listeners
lib	Java libraries (JAR files)
web.xml	Optional Java EE deployment descriptor
weblogic.xml	Optional WebLogic deployment descriptor
index.html page1.jsp page2.jsp	Static and dynamic Web content



MyWebApp.war

Web Application Archive

Web archives are created using the `jar` utility:



web.xml
weblogic.xml
Deployment descriptors
HTML
HTML
Static resources: HTML, text images
Servlet
Servlets, JavaBeans, and other classes
JSP
JSPs
jar
.war

Optional Configuration of Web Applications

Web applications can be specified in `web.xml` and `weblogic.xml` deployment descriptors. The configurations include:

- Defining the run-time environment
- Mapping URLs to servlets and JSPs
- Defining application defaults such as `welcome` and `error` pages
- Specifying Java EE security constraints
- Defining work managers for applications
- Setting the context root for the application

web.xml

The `web.xml` file is used to configure the following:

- Servlets and JSP registration
- Servlet initialization parameters
- JSP tag libraries
- MIME type mappings
- Welcome file list
- Error pages
- Security constraints and roles
- Resources
- EJB references

`weblogic.xml`

Using `weblogic.xml`, you can configure the following:

- The application's root context path
- Application logging
- Security role mappings
- Advanced session settings
- Session clustering
- References to shared libraries
- References to server resources (data sources, EJBs, and so on)
- Work managers and threading
- Virtual directories
- JSP compiler options

weblogic.xml Deployment Descriptor

Example of the weblogic.xml deployment descriptor:

```
<?xml version='1.0' encoding='utf-8'?>
<weblogic-web-app
   xmlns="http://xmlns.oracle.com/weblogic/weblogi
   c-web-app"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema
-instance">
</weblogic-web-app>
```

URLs and Web Applications

The URL that is used to reference a resource in a Web application must include the name of the Web application.

Accessing a resource in a Web application:

`http://`*`hostname`*`:`*`port`*`/`*`MyWebApplication`*`/`*`resource`*

Where:

`Hostname`	Host name mapped to virtual host or *`hostname:port`*
`MyWebApplication`	Name of the Web application; not necessary if this is the default Web application
`resource`	Static page, servlet mapping, or JSP

Web Service Applications

A Web service application:

- Responds to HTTP client requests using the Simple Object Access Protocol (SOAP)
- Uses the same structure as a Java EE Web application
- Supports two additional deployment descriptors:
  - `webservices.xml`
  - `weblogic-webservices.xml`

Virtual Directory Mappings

Virtual directories:

- Can be used to refer to physical directories
- Enable you to avoid the need to hard-code paths to physical directories
- Allow multiple Web applications to share common physical directories for specific requests such as images
- Decrease duplication of files across applications
- Are configured in `weblogic.xml`

Virtual Directory Mapping: Example

```
<virtual-directory-mapping>
    <local-path>c:/usr/gifs</local-path>
    <url-pattern>/images/*</url-pattern>
    <url-pattern>*.jpg</url-pattern>
</virtual-directory-mapping>
<virtual-directory-mapping>
    <local-path>c:/usr/common_jsps.jar</local-path>
    <url-pattern>*.jsp</url-pattern>
</virtual-directory-mapping>
```

Road Map

- Web applications
- EJB applications
  - Major EJB types and their purpose
  - EJB deployment descriptor files
- Enterprise applications

EJB Applications

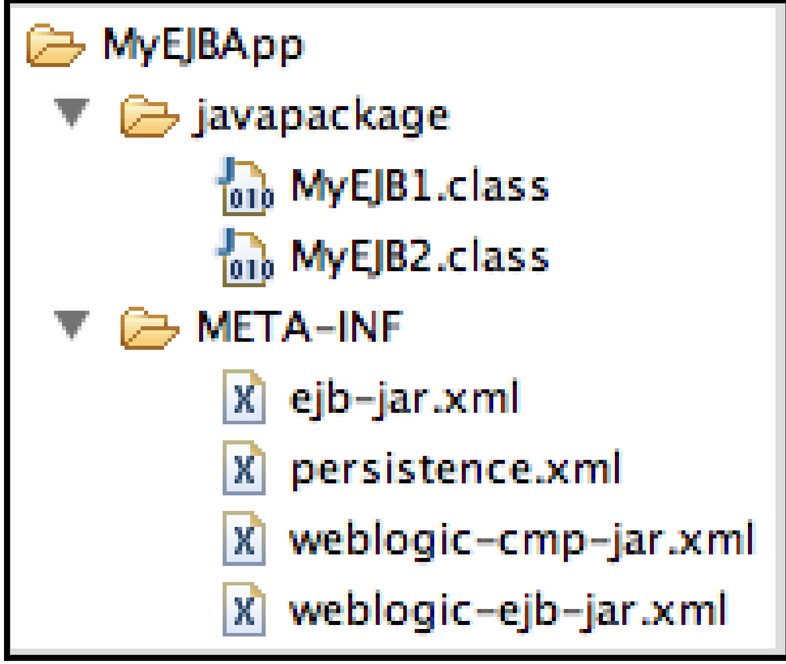
Enterprise JavaBeans (EJBs):

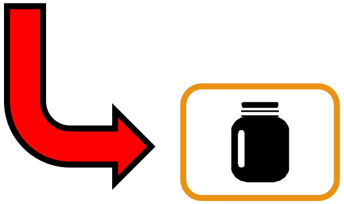
- Standardize the development and deployment of server-side distributed components
- Are annotated Java classes
- Are packaged with XML deployment descriptors
- Support the following capabilities:
  - Remote access over a network
  - Object-relational mapping via WLS or the Java Persistence API (JPA)
  - Transactions
  - Messaging integration
  - Dependency injection

Types of EJBs

EJB Type	Description	Example
Stateless Session	• Do not maintain state • Are synchronous • Are maintained in memory	• Check validity of stock symbol • Calculate billing of a phone call
Stateful Session	• Offer conversational interaction • Maintain state for client • Are synchronous	• Book a flight & car rental for travel • Manage a shopping cart
Entity	• Represent persisted data • Are synchronous	• Represent a player’s statistics • Represent a stock’s history
Message-Driven	• Are asynchronous & stateless • Consume JMS messages	• Store logging messages

EJB Application Structure

Directory or File	Description
 MyEJBApp	Application root folder
javapackage	Java classes organized into packages
MyEJB1.class MyEJB2.class	EJB and other Java class files
META-INF	Meta-information folder
ejb-jar.xml persistence.xml	Optional Java EE and JPA deployment descriptors
weblogic-cmp-jar.xml weblogic-ejb-jar.xml	Optional WebLogic deployment descriptors



`MyEJBApp.jar`

weblogic-ejb-jar.xml

Using `weblogic-ejb-jar.xml`, you can configure:

- Security role mappings
- Advanced security settings
- EJB clustering
- EJB pooling and caching
- Work managers and threading

Administrator Tasks with EJBs

The administrator tasks for EJBs include:

- Configuring and deploying
- Resolving JNDI and other infrastructure issues
- Monitoring EJB caches and pools

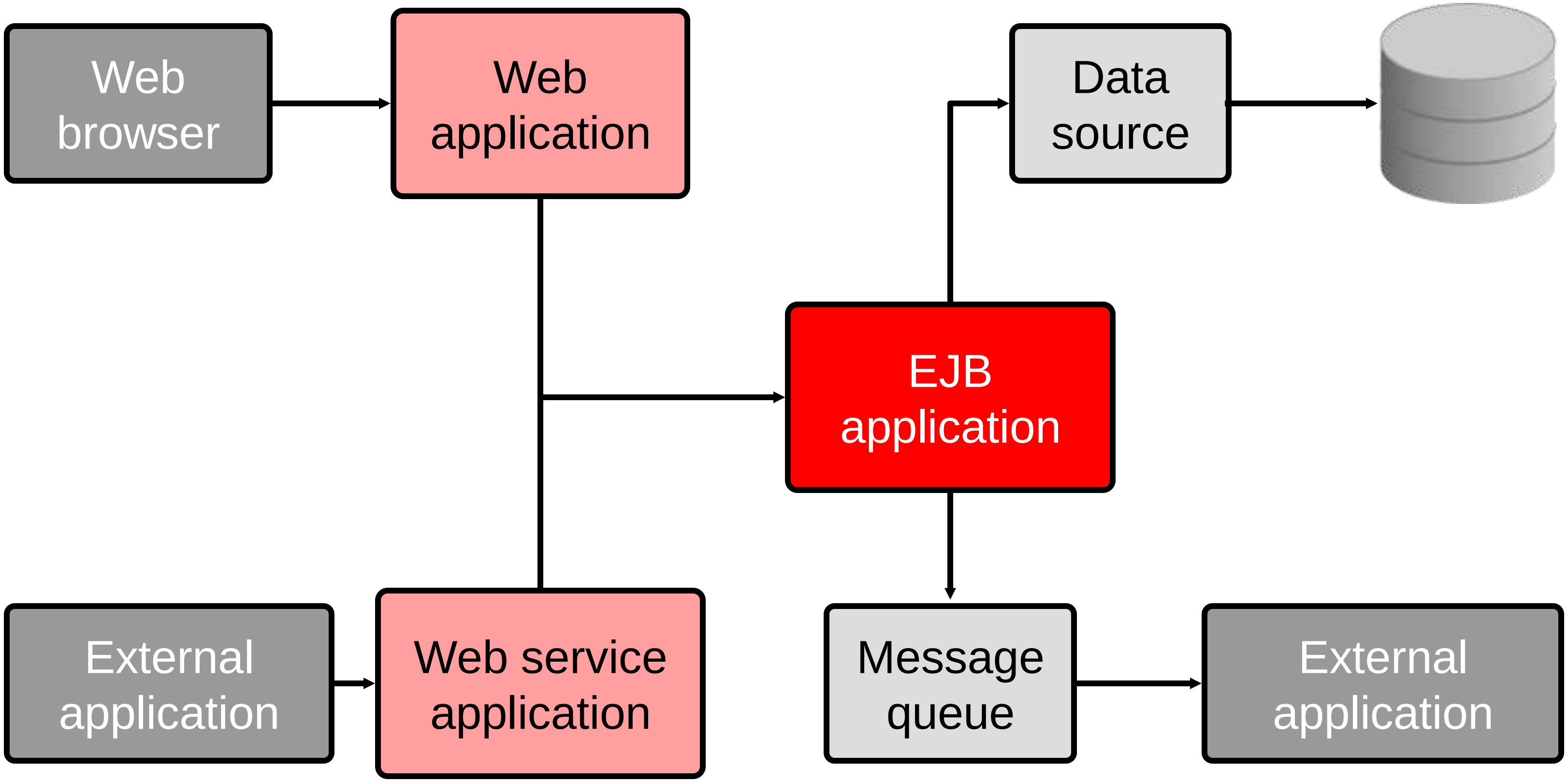
Road Map

- Web applications
- EJB applications
- Enterprise applications
  - Enterprise application concepts
  - Enterprise Archive (`.ear`) file structure
  - Enterprise application configuration

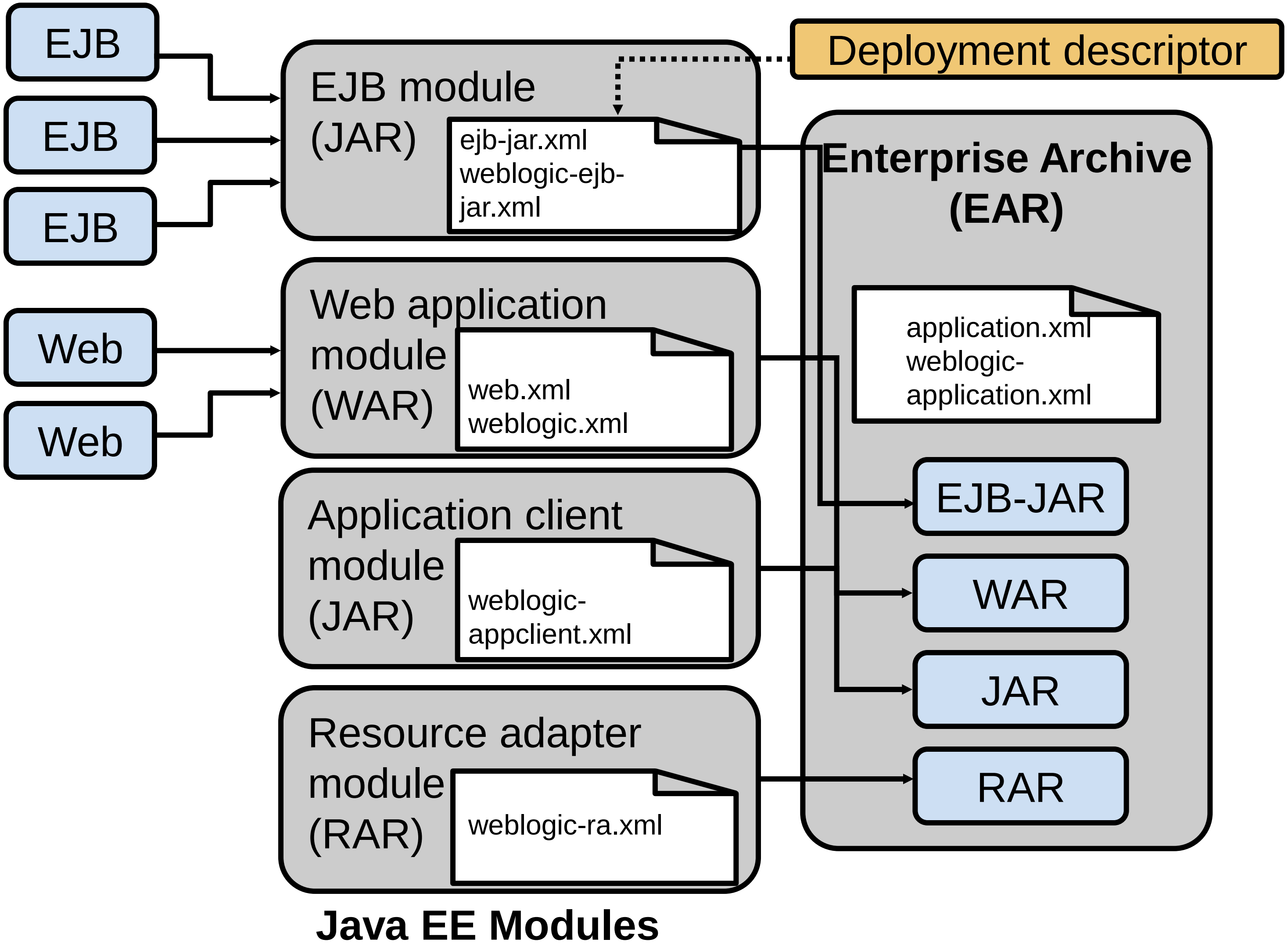
What Is an Enterprise Application?

- An enterprise application is a grouping of several resources into one deployable unit that is packaged in an `.ear` file.
- These resources include:
  - Web applications (`.war`)
  - EJB applications (`.jar`)
  - Java applications (`.jar`)
  - Resource adapters (`.rar`)

A Typical Java EE System



Java EE Enterprise Application



Java EE Enterprise Application

An enterprise application:

- Comprises one or more Java EE application modules:
  - Web applications
  - EJB applications
  - Other Java libraries (JARs)
- Allows related applications to be deployed as a unit
- Can include application-specific JDBC and JMS resources

Why Enterprise Applications?

Use enterprise applications to:

- Avoid namespace clashes
- Declare applicationwide security roles
- Deploy an application as one unit
- Share applicationwide EJB resources
- Configure local JDBC data sources
- Configure local JMS resources
- Configure local XML resources

Enterprise Application Structure

Directory or File	Description
📂 MyApp	Application root folder
▼ 📂 lib	Replaces /APP-INF/lib. May contain:
📂 mylib1	- Common Java class files
📂 mylib2	- Common Java libraries (JARs)
▼ 📂 META-INF	
📄 application.xml	Optional JEE deployment descriptor
📄 mydatasource-jdbc.xml	JDBC and JMS modules
📄 myqueue-jms.xml	
📄 weblogic-application.xml	Optional WebLogic descriptor
🗐 EJBApp.jar	Web and EJB application modules
🗐 WebApp1.war	
🗐 WebApp2.war	

MyApp.ear

`weblogic-application.xml`

Using `weblogic-application.xml`, you can configure:

- References to shared libraries
- Work managers and threading
- Default EJB and Web application parameter values

Application Scoping

Configure enterprisewide WLS-specific features with the `weblogic-application.xml` deployment descriptor:

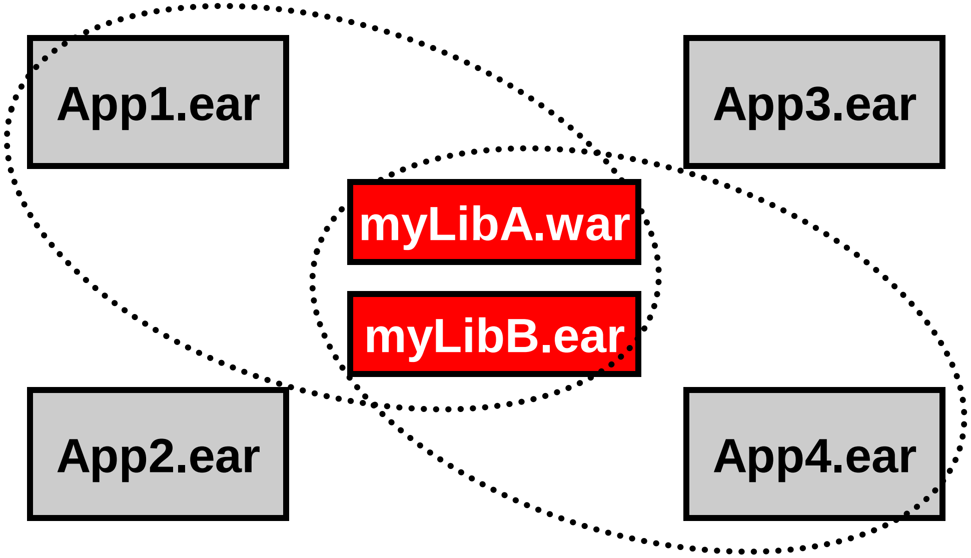
- XML parsers
- XML entity mappings
- JDBC data sources
- JMS connection factories and destinations
- Security realms

EAR Class Libraries

- Extending the Java 2 EE 1.4 specification, Oracle added `APP-INF/lib` and `APP-INF/classes` to the standard Java EE EAR file structure. For Java EE 5, it is preferable to use the `/lib` directory.
- When an application is initialized, the paths extracted are prefixed to the application’s classpath.
- Classes are added to the root classloader of the application.

Java EE Library Support

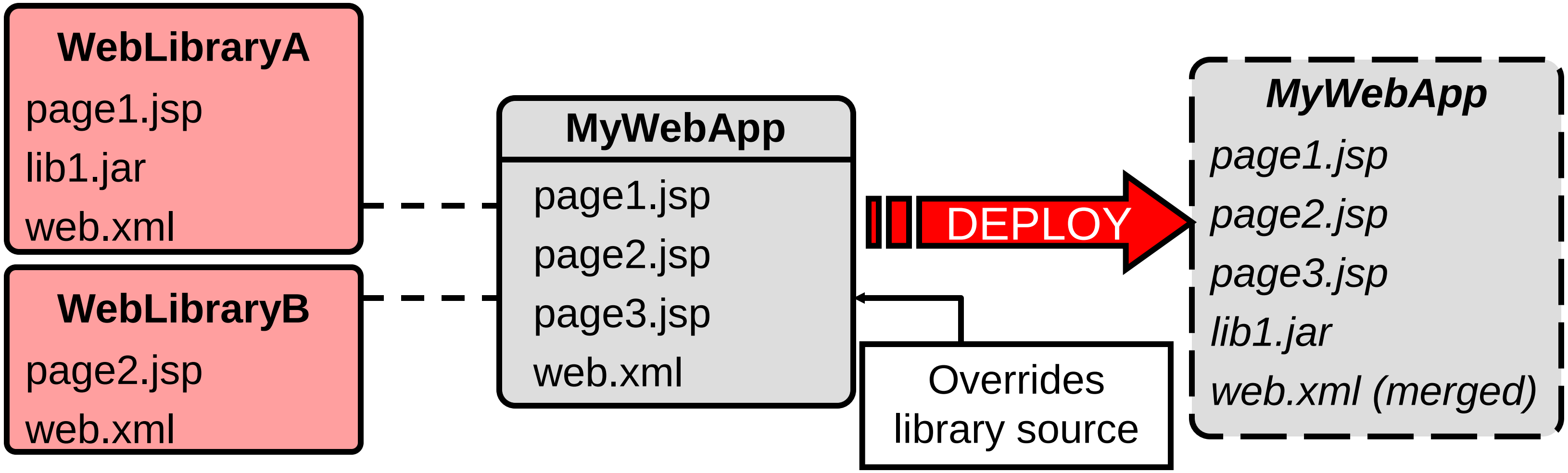
- Create a library of Java EE modules, package the modules into an EAR, a WAR, or an EJB file, and then deploy and register the module with the application container.
- Other applications can later use the modules as if they were packaged in their own EAR, WAR, or EJB files.
- This allows for more reusability between the applications.



WebLogic Java EE Shared Libraries

A Java EE shared library:

- Is a reusable portion of a Web or enterprise application
- Is referenced by other deployed applications
- Avoids duplicating source files among Java EE projects
- Can contain deployment descriptors that are merged with the application's descriptors



Shared Library References

- For Web applications, list the required shared libraries in `weblogic.xml`.
- For enterprise applications, list the required shared libraries in `weblogic-application.xml`.
- Excerpts from `weblogic.xml`:

Shared library name and version

```
:
<library-ref>
    <library-name>ajax-tools-lib</library-name>
    <specification-version>1.5.0</specification-version>
    <implementation-version>2.0.0</implementation-version>
</library-ref>

<library-ref>
    <library-name>help-web-lib</library-name>
    <specification-version>1.5.0</specification-version>
    <implementation-version>1.1.0</implementation-version>
</library-ref>
:
```

Quiz

A ____________ is a reusable Oracle WebLogic Server application that can be referenced by other deployed applications.

1. Java library
2. Shared library
3. Web library
4. Composite library
5. Reference library

Quiz

Which of the following is NOT a supported type of application in Oracle WebLogic Server?

1. Enterprise application
2. EJB application
3. Process application
4. Web service application
5. Web application

Quiz

Which of the following are valid Oracle WebLogic Server deployment descriptor files for configuring applications?

1. `weblogic-webapp.xml`
2. `weblogic-ejb-jar.xml`
3. `weblogic.xml`
4. `weblogic-application.xml`
5. `weblogic-library.xml`

Summary

In this lesson, you should have learned how to:

- Package and deploy Web applications
- Describe deployment descriptors
- Explain Enterprise JavaBeans concepts
- Configure and deploy EJBs

Practice 11 Overview: Web Application Deployment Concepts

This practice covers the following topics:

- Deploying (installing) prerequisite libraries
- Deploying (installing) applications
- Starting and stopping applications
- Testing applications
- Redeploying (updating) applications
- Undeploying (deleting) applications
- Front-ending applications with a Web server—for example, Oracle HTTP Server