

J2EE Web Applications

Creating J2EE Web Applications with Tomcat

Instructor: Rick Palmer, SCWCD

rick@online-ettraining.com

Topics Covered

□ Java 2 Enterprise Edition (J2EE) Primer

- ⇒ Containers and Services
- ⇒ Components
- ⇒ APIs

□ Tomcat Web Server Overview

- ⇒ Installing and configuring a Tomcat web
- ⇒ Creating a Tomcat web application
- ⇒ Packaging and deploying a Tomcat web application

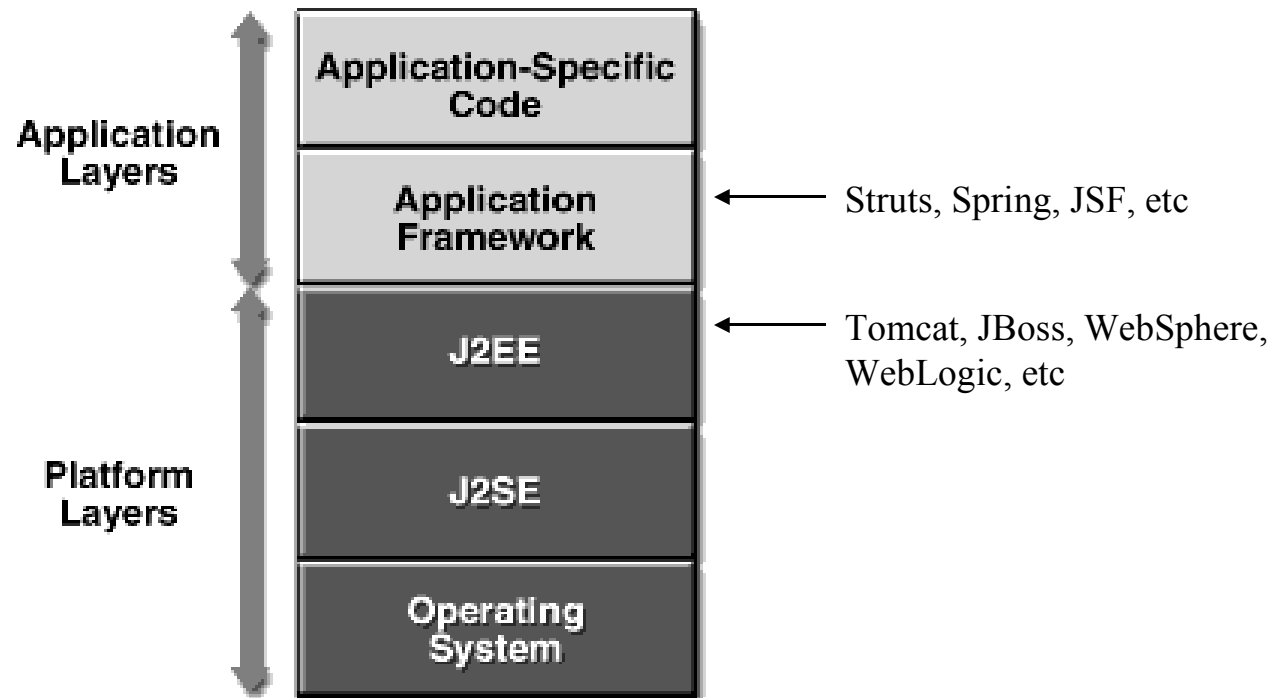


What is J2EE?

- ❑ J2EE is a specification, or blueprint, that defines the services required by distributed web applications: <http://java.sun.com/j2ee/1.4/docs/api>
 - ⇒ Internet communications (JSP, Servlets, JMS)
 - ⇒ Database connectivity and transactions (JDBC)
 - ⇒ Distributed business logic (EJB)
- ❑ Vendor software that implements the specification is said to be J2EE-compliant (e.g. JBoss).
- ❑ Applications developed for one J2EE server should also run on other J2EE servers.

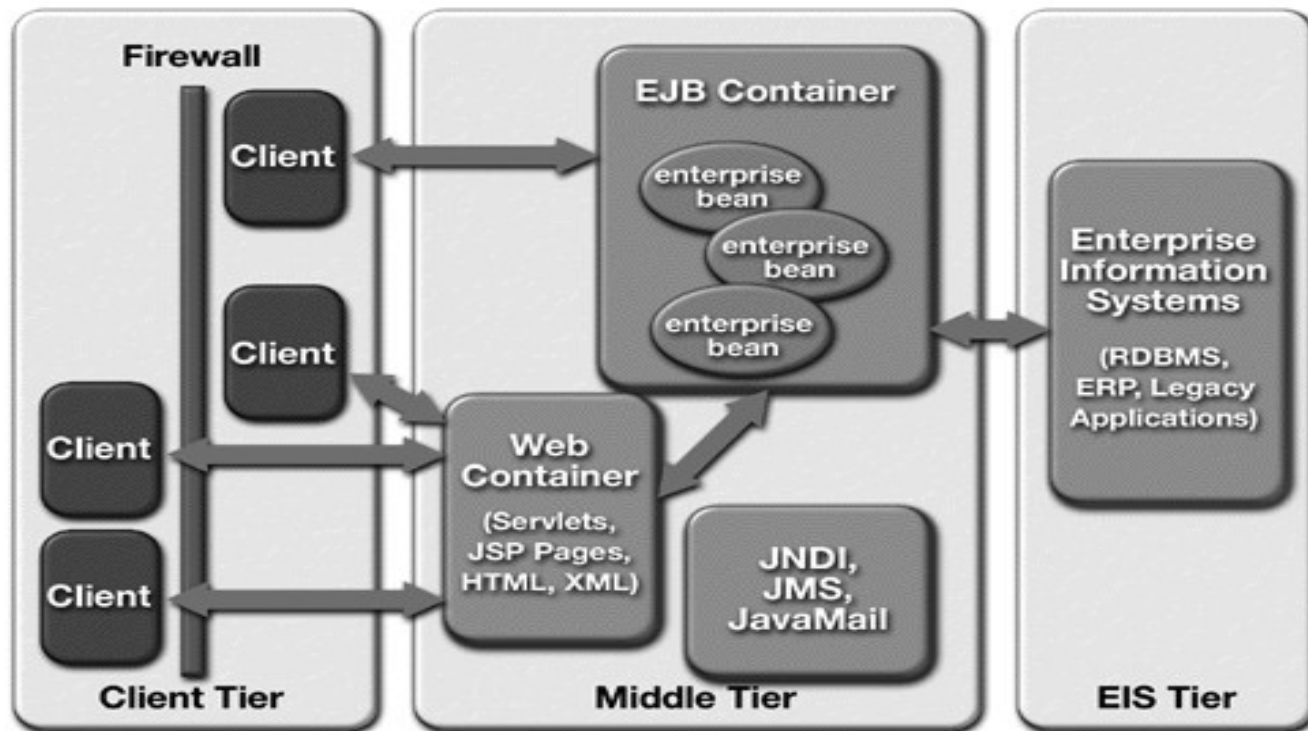
Where does J2EE fit in?

- ❑ Java 2 Enterprise Edition uses the Java 2 Standard Edition (e.g. collections, exception handling, and inheritance).
- ❑ J2EE comes bundled with web servers like Tomcat and JBoss that implement the J2EE specification.



J2EE Architecture

- ❑ Functionality is distributed across multiple logical and physical tiers
- ❑ J2EE containers provide services to middle tier components, which often run on multiple servers (clustering)



Why not just Client/Server?

- ❑ Client/server is a proven model for small networks, but does not scale well for a large number of users
 - ⇒ Faster performance for a handful of users
 - ⇒ Clients typically maintain their own network connection and database connection with the server, which quickly limits the number of users that can be serviced
- ❑ Presentation, business logic, and database connectivity are usually bundled in the client application (fat client)
 - ⇒ Difficult and costly to install, upgrade, or enhance
 - ⇒ Costs increase as size of user base increases

J2EE Objectives

□ Simple

⇒ Let developers focus on business logic, not infrastructure services like database connection pooling, object life cycle management, threading, etc.

□ Portable

⇒ Run on any J2EE-compliant server

□ Secure

⇒ Ensure authorized access only

□ Scalable

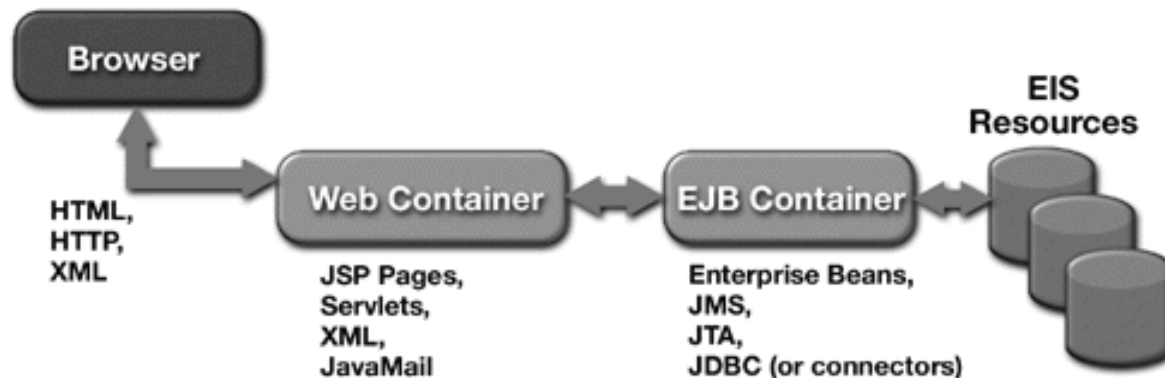
⇒ Perform well under heavier loads

⇒ Easily grow to meet increased demand

J2EE Objectives (2)

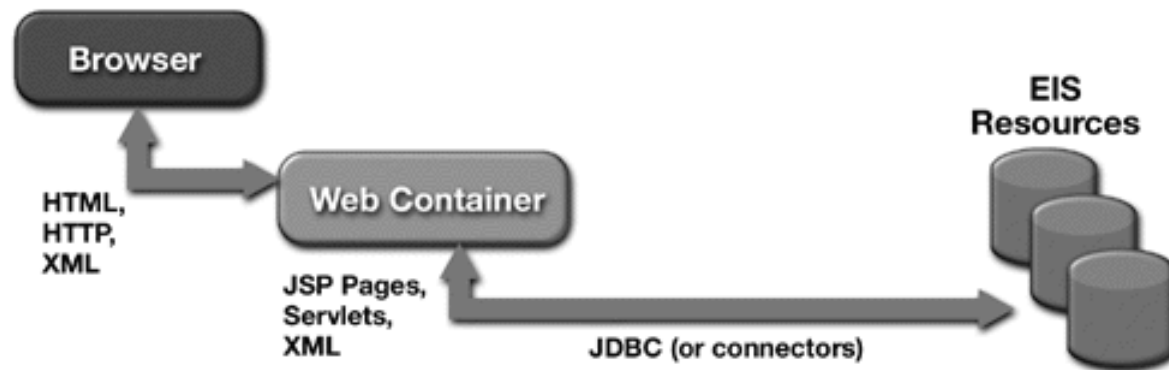
□ Component-based:

- ⇒ Components in one tier can be updated without impacting the rest of the application
- ⇒ Components map easily to application functionality, and promote reusability
- ⇒ Allows developers to focus on the component technologies that match their skill sets



Where does Tomcat fit in?

- ❑ Tomcat is a partially J2EE-compliant web container that implements the Servlet and JSP specification
- ❑ Handles Internet HTTP requests, typically from web browsers, but also from stand-alone clients that are connected to the Internet



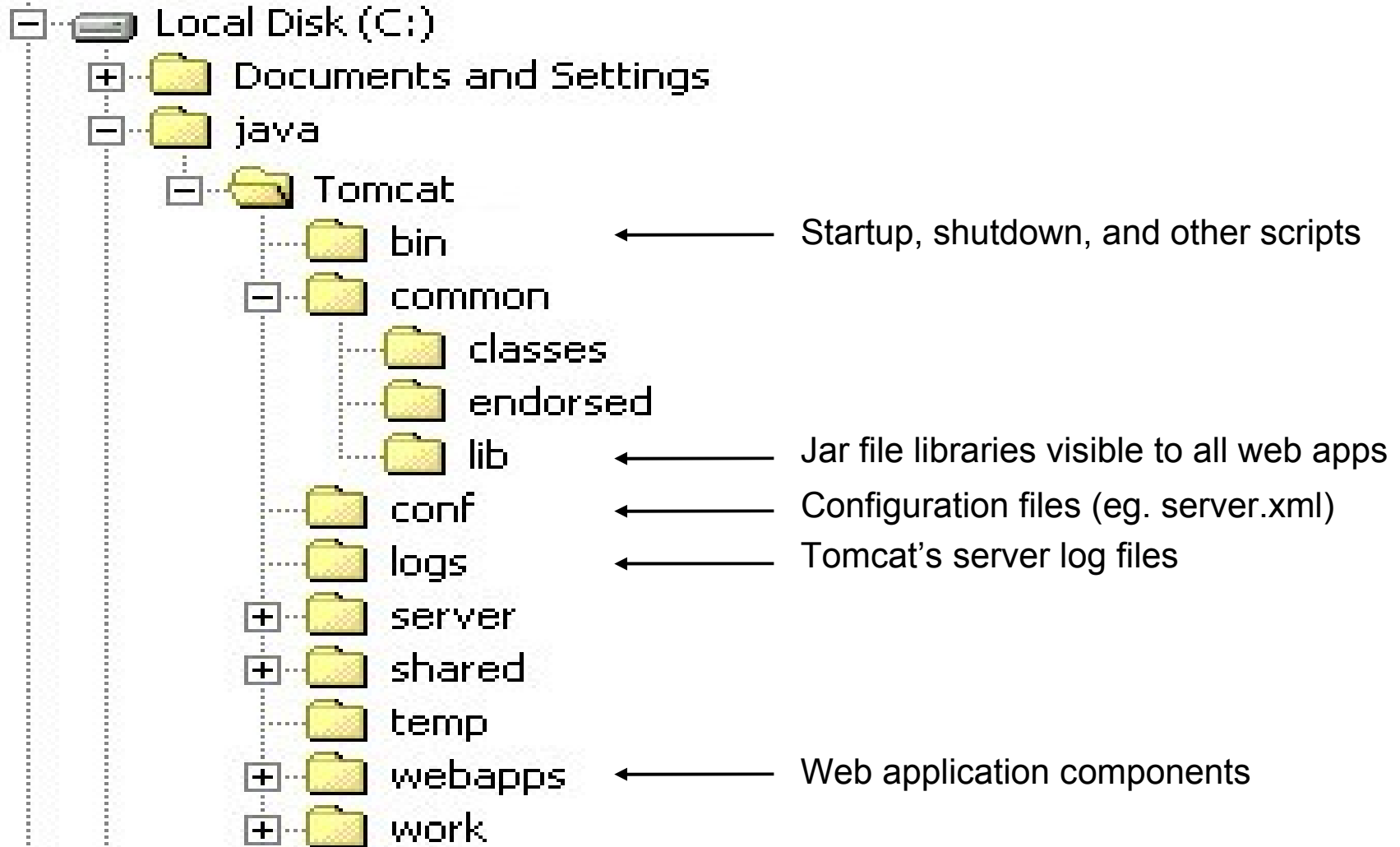
Installing a Tomcat Web Server

- ❑ Install JDK 1.4.2 SDK or most recent version
- ❑ Download Tomcat 5.0.28 from Apache
 - ⇒ <http://tomcat.apache.org/download-55.cgi#5.0.28> (Windows Installer)
- ❑ Run the executable
 - ⇒ Install in short folder location (eg. C:\Tomcat) - helps later when setting CLASSPATH values
- ❑ Start and stop Tomcat from the Start button - Programs - Apache Tomcat - Configure Tomcat menu
- ❑ Verify installation by pointing web browser to <http://localhost:8080> or <http://localhost>

Configuring Tomcat

- ❑ Tomcat Manager – <http://localhost/manager/html>
 - ⇒ List installed web apps
 - ⇒ Stop, start, reload web apps
 - ⇒ Deploy web apps
- ❑ Tomcat Administrator – <http://localhost/admin>
 - ⇒ Manage server settings
 - ❑ Connection Timeouts
 - ❑ Resources (Databases, Email, etc)
 - ❑ Users

Server Directory Layout



Deployment Descriptors

- *Web server* settings are stored in:

```
<tomcat_home>\conf\server.xml
```

- *Web application* settings are stored in:

```
<tomcat_home>\webapps\YourApp\WEB-INF\web.xml
```

⇒ Use a standard XML header

⇒ Use a root `<web-app>` element

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
```

```
  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/web-app_2_4.xsd"
```

```
  version="2.4">
```

```
<!-- Configuration elements for your web app go here.
```

```
All are optional, but MUST be in the correct order. -->
```

```
</web-app>
```

Changing the Default Server Port

- ❑ Tomcat runs on port 8080 by default:

- ⇒ <http://localhost:8080/>

- ❑ Running Tomcat on port 80 (default HTTP port):

- ⇒ Open `<tomcat_home>\conf\server.xml` and modify the port attribute for the non-SSL HTTP Connector element

```
<Connector port="80" maxThreads="150" minSpareThreads="25"
maxSpareThreads="75" enableLookups="false"
redirectPort="8443" acceptCount="100" debug="0"
connectionTimeout="20000" disableUploadTimeout="true" />
```

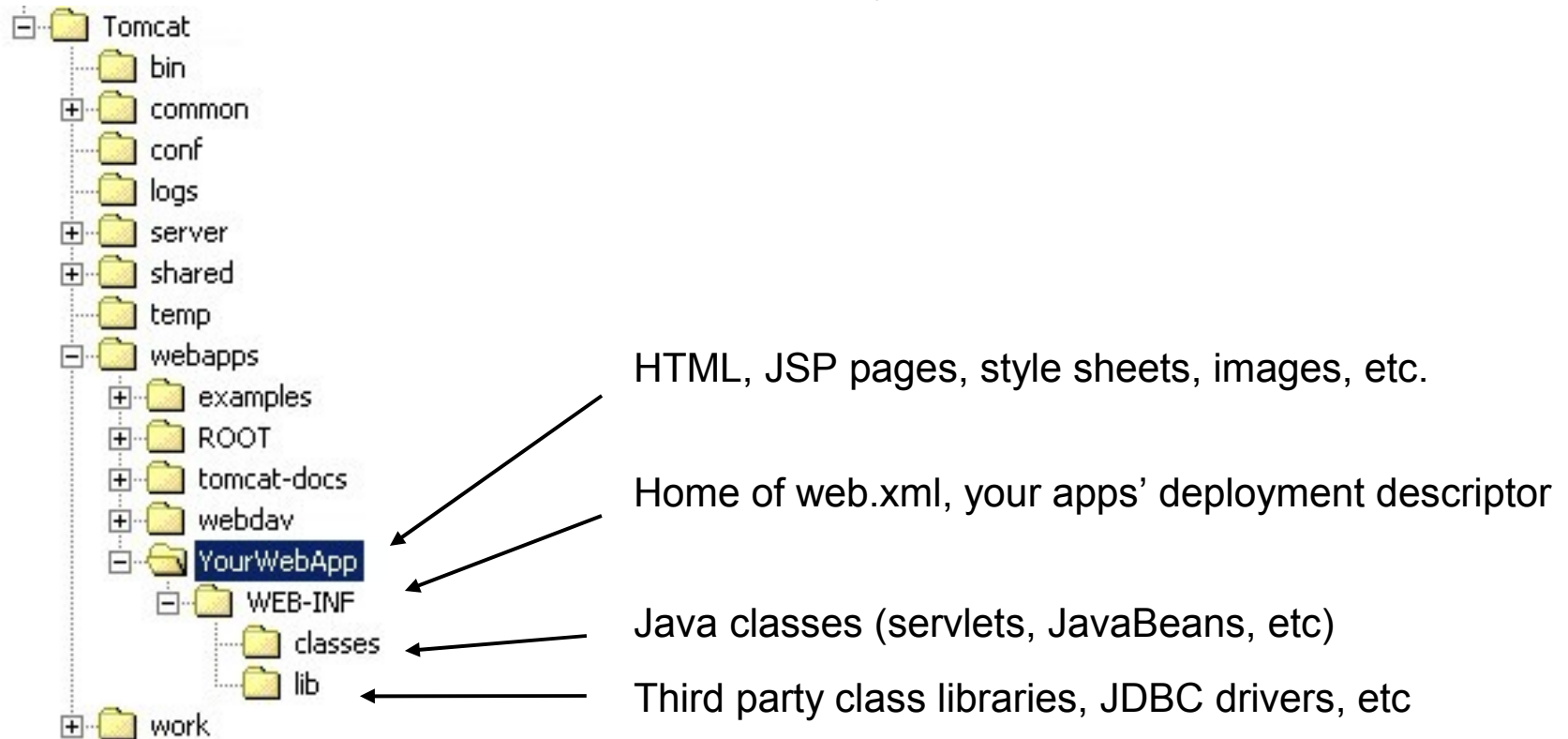
- ⇒ Save `server.xml` and restart Tomcat

- ⇒ Enter <http://localhost> in the browser (no port)

- ⇒ NOTE: Microsoft IIS must be disabled to avoid conflicts

Creating a Web Application

- ❑ First create web folders for your app



- ❑ Next create web components (HTML, JSP pages, servlets, etc...)

Specifying a Welcome Page

- ❑ For a given URL, the server looks for `index.jsp` or `index.html` to display as the welcome page
 - ⇒ If not found, then a list of available files is shown
- ❑ Use a `welcome-file-list` element in `web.xml` to specify an alternative welcome page:

```
<web-app>  
  <welcome-file-list>  
    <welcome-file>login.htm</welcome-file>  
  </welcome-file-list>  
</web-app>
```

Packaging and Deploying the App

- ❑ Build WAR file (Web Archive) from command line to package your web application:
 - ⇒ `jar cvf TestApp.war *`
 - ⇒ `cvf` = create new archive with verbose output and a specified file name
 - ⇒ `*` = bundle all files in current folder and in all subfolders
- ❑ Stop Tomcat
- ❑ Delete any existing folder and war file for your application
- ❑ Copy new war file to Tomcat's webapps folder
- ❑ Restart Tomcat
- ❑ Access your web app using <http://localhost/YourWebApp>

Resources

- ❑ The J2EE Tutorial: <http://java.sun.com/j2ee/1.4/docs/tutorial/doc/>