UNIT III

Java Server Pages

JSP and Servlet

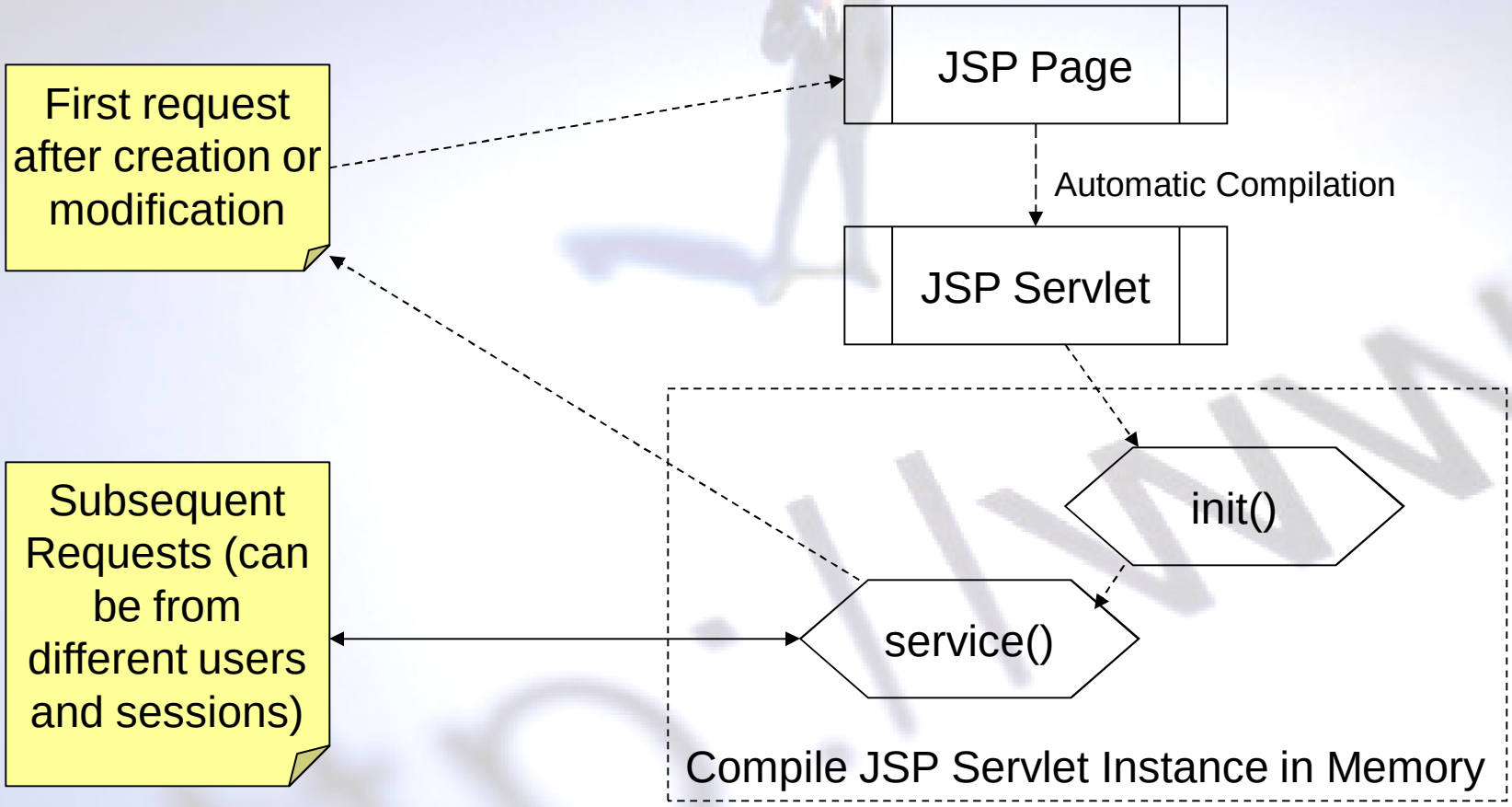
- Limitations of servlet
  - It is difficult to write HTML
  - It's ineffective to *design* webpages
  - It's inaccessible to non-programmers

- JSP is a complement to servlet
  - JSP focuses on user interface and presentation
  - JSP enhances the design capability of servlet
  - JSP pages can be written with any text editor, including HTML editor
  - JSP is a server side technology

JSP Pages

- JSP page file ends with ".jsp" by default
- JSP pages are organized like any other HTML files using the normal directory/file structure
- A JSP page is usually composed of regular HTML tags and JSP scripting elements
- JSP page is implicitly compiled to servlet class and loaded into memory
  - when the page is requested the first time after creation, or
  - when the page is requested the first time after modification
  - Refer to table 10.1 in the textbook and the next slide

JSP Compilation and Execution



Servlet and JSP

	Servlet	JSP
Development	java classes (.java)	scripting file (.jsp)
Deployment	Manually compiled; Specifically mapped	Directly mapped: copy JSP files to intended directories
Execution	No need of source files	Automatic compilation; automatic reloaded; source files (.jsp) are necessary

JSP Elements

- Scripting elements
  - Scriptlet
    - Regular Java code
  - Expression
    - Shortcut for output
  - Declaration
    - Declaring variables and methods at the class level
- Directive
- JSP action
- Comments (<%-- … --%>)

Scriptlets

- Wraps regular Java statements which are usually written *within* a method

```
<%
    … (Java statements)
    // may include comments, variable declaration and assignment, loops,
    conditional statements, object initialization, method call, etc…
%>
```

- Using the *implicit object* “out” as the standard output

```
out.println( … ) or out.print( … )
```

Expression

- A shortcut to print out a value or an expression

  <%= [expression]%>

  - Expression can be a variable, formula, object property, string concatenation, method with return value, or anything that returns a value

JSP Output Practices

- Ways to treat static HTML content
  - Regular/block output (servlet way)
    - Uses “out.println()” or “out.print()” method to generate all content, including static content
  - “Spaghetti”/mixed output (scripting way)
    - Uses JSP scriptlets or expressions for dynamic content only
    - Mixes scripting elements and static content

Regular Output

- Using “out.print()” or “out.println()” method to generate HTML as a block, even the whole page – Servlet way

- StringBuffer is often used to construct HTML content first, and then printed out at one time

Spaghetti Output

- Expression elements are often used where dynamic content is needed
- Use regular HTML for static content; don't include them in JSP scripting elements
- How mixed should it be?
  - Depends on your own style
  - Coding should be most convenient and clear
  - Depends on development requirement

Declarations

- Declaration element is used to define *member variables* and methods

  <%! … %>

  - Variables not defined in declaration element are local / method level variables
  - Methods can only be defined in the declaration element

- Like regular class variables and methods, the location where you define these variables and methods is not important

JSP Page Directive

- Directives affects the overall structure of the servlet generated

  <%@ … %>

- Use page directive to import classes

  <%@ page import="…, …, …"%>

  - This is equivalent to the "import" statement in regular Java classes

JSP Include Directive

- How to reuse code?
- Use include directive to include source code from another file

  <%@ include file="…" %>

  - Inclusion happens at the **compilation** time
  - What is included is the source, not generated result
  - Often used to include method definitions

JSP Include Action

- Use “jsp:include” action to dynamically include content from other files
  - The statement is placed where the actual content will be inserted

  <jsp:include page=“…” />

- “page” points to a ***local*** text file (.html, .htm, .jsp, .txt)
  - Relative path

  <jsp:include page=“menu.jsp” />

  - Absolute path
    - Note: absolute path starts from the current **application context**

  <jsp:include page=“/menu.jsp” />

Include Action Usage

- “jsp:include” is often used to include the contents that are *consistent* on many pages, e.g., menus, titles, page headers, footnotes, …
  - http://www.delta.com
  - See example “ssi.jsp” and “WEB-INF/menu.jsp”
- Or, it is often used to include contents that are different (dynamic inclusion)
  - http://www.cardmemberservices.com/
  - http://jackzheng.net/cis3270summer2006/
  - See example “home.jsp” and “WEB-INF/course.htm”
- Or a hybrid model (templating)

Include Action and Directive Comparison

	Include Action	Include Directive
When does inclusion occur?	At request/run time	At compilation time
What's included?	Final output of the included page	Source code/content
Main page maintenance	Updates of the included page is automatically reflected	Updates of the included page is NOT automatically reflected

- See table 13.1 one page 380 for a complete comparison of include directive and include action

Redirection, Refreshing and Forwarding

- Redirection
  - response.sendRedirect()
- Refreshing
  - response.setHeader(“Refresh”, “10; url=…”)
- Forwarding <jsp:forward page=“…” />
  - The “page” attribute follows the same rule as that of <jsp:include/>
  - Forwarding does not invoke network traffic
  - The destination URL is hidden; original requested URL does not change in browser address bar after forwarding
- Compare redirecting and forwarding

Request Processing

- Using implicit object “request”
- Processing HTTP request headers
  - The same way as servlet
- Reading URL parameter

  http://localhost/appcontext/request.jsp?choice=yes

  - Parameter processing is the same way as servlet, using request.getParameter(“…”), request.getPameterValues(“…”)

Form Processing with JSP

- The same way as servlet

  request.getParameter("…")

  request.getParameterValues("…")

  - Note: the action attribute of the form should be a JSP file that processes data

    <form method="post" action="studentprofile.jsp">…</form>

Database Processing with JSP

- The same way as servlet
  - Don’t forget the directive

    <%@ page import="java.sql.*" %>

  - See the example “product.jsp”

JSP Implicit Objects Summary

- Some system objects are initialize automatically ready to use in the JSP environment
  - out: standard output object
  - request: represents request information and behavior
  - response: represents response information and behavior
  - [session]: represents a typical time period of communication between a client and a server
  - [application]: represents context of a web application