
Java EE7 Web JSP 網頁文稿元素

鄭安翔

ansel_cheng@hotmail.com

課程大綱

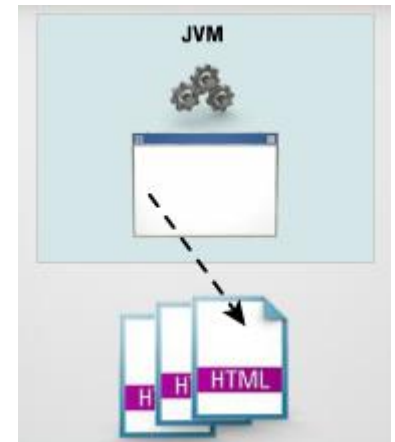
1) **Java Sever Page 網頁技術**

- **JSP 運作機制**
- **JSP 範例**

2) **JSP scripting文稿元素**

Java Servlet 技術編輯缺點

- Servlet中編輯HTML程式碼很困難
 - 表示層與商業邏輯部份混在一起
 - 無法使用”所見即所得”型態編輯器
 - 使用標籤編輯HTML很繁瑣且容易出錯



- Template Page 技術
 - 在HTML頁面中嵌入的具運算功能的標籤碼
 - 動態生成網頁內容及數據
 - Java 的 Template Page 技術
 - Java Server Page



常用 Template Page 技術

Table of numbers squared:

member	squared
0	0
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81

■ PHP (PHP Hypertext Preprocessor)



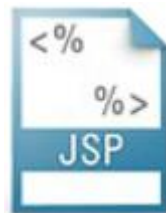
```
<? for ( $i=0; $i<10; $i++ ) { ?>
    <TR><TD><? echo $i ?></TD><TD><? echo ($i * $i) ?></TD></TR>
<? } ?>
```

■ ASP (Active Server Pages)



```
<% FOR I = 0 TO 10 %>
    <TR><TD><%= I %></TD><TD><%= (I * I) %></TD></TR>
<% NEXT %>
```

■ JSP (Java Server Pages)



```
<% for ( int i=0; i<10; i++ ) { %>
    <TR><TD><%= i %></TD><TD><%= (i * i) %></TD></TR>
<% } %>
```

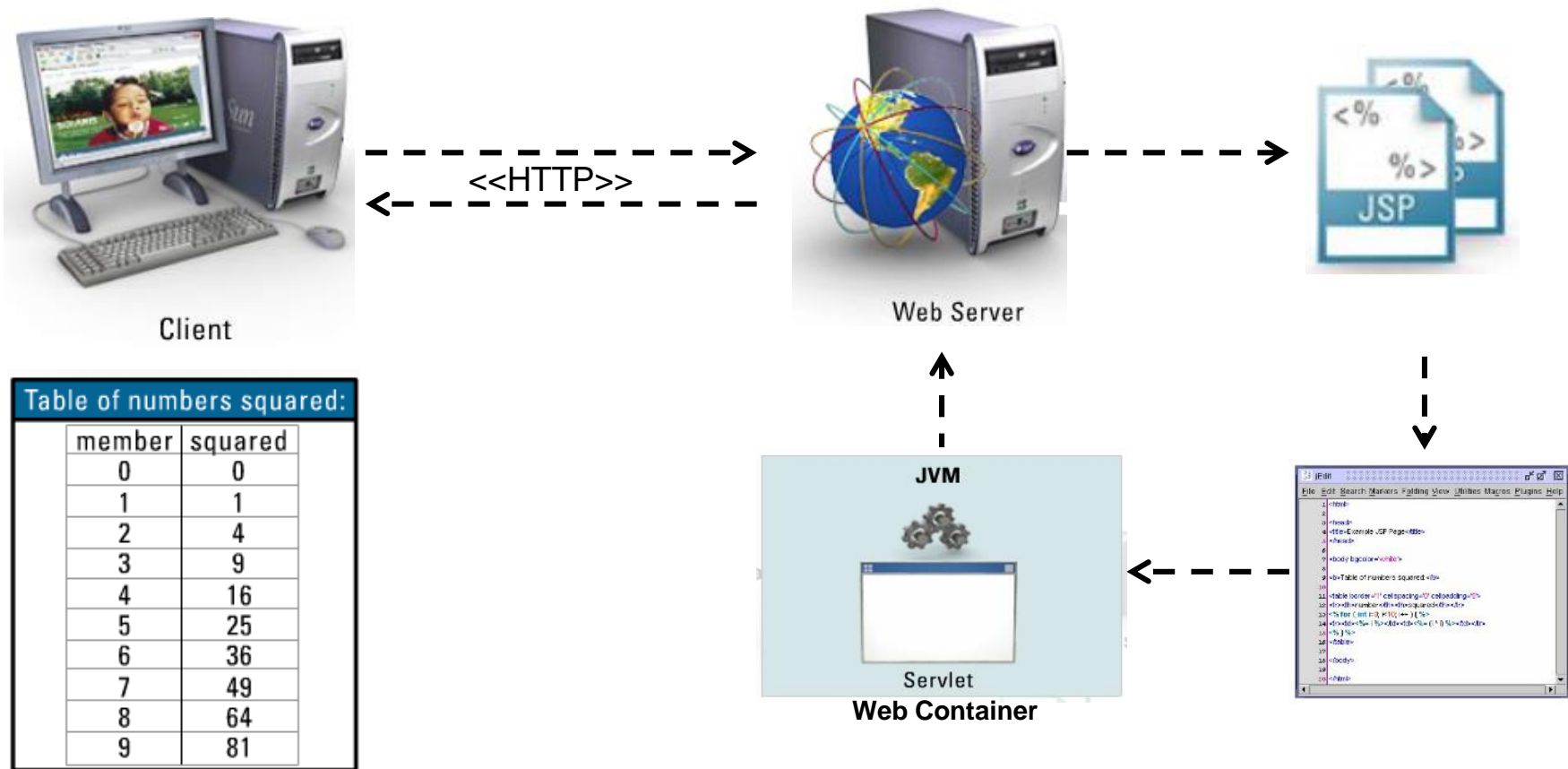
Java Server Pages 技術

■ Java Server Pages

- JSP頁面由網路元件容器編譯成為Java Servlet類別
 - 與一般直譯式的Template Page 技術不同
 - 但與直譯式Template Page 技術一樣,不需深度程式設計概念,讓網頁設計者能夠輕鬆建立動態網頁內容
- 執行時載入Servlet類別
 - Servlet能作到的功能, JSP技術幾乎都做得到
 - 可透過標籤函式庫的方式擴充JSP的功能

JSP 網頁應用程式執行架構

■ Java Server Page動態網路應用程式架構



JSP技術優缺點

■ JSP技術優勢：

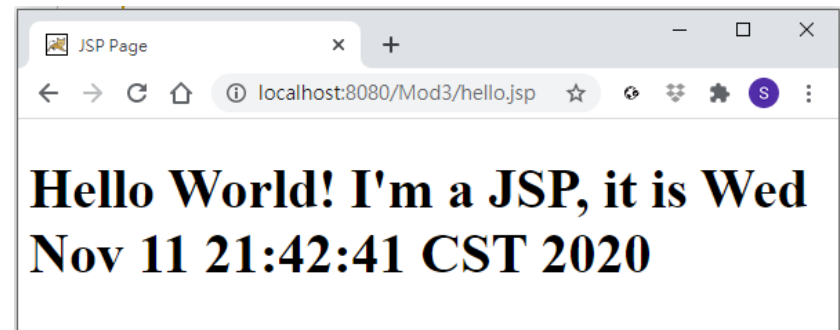
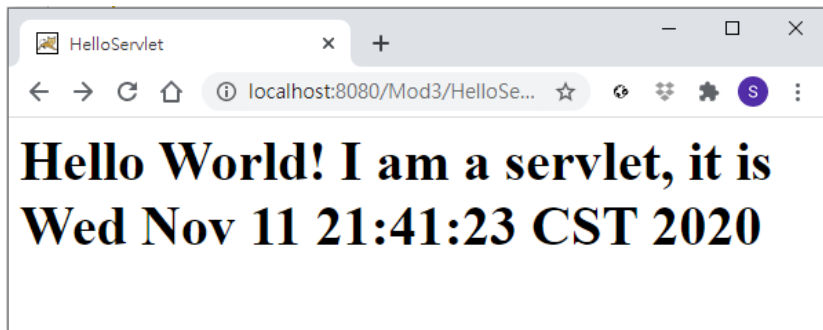
- JSP技術具有servlet的所有的優點技術：
 - 高效能
 - 高可擴展性
 - 跨平台
- 可使用Java語言作為文稿(scripting)元素,享受物件導向的好處

■ JSP技術的缺點：

- 單獨使用JSP網頁技術時,需同時處理商業邏輯與展示層
- 並行及同步的問題
- JSP頁面轉成Servlet檔再編譯、執行,不容易Debug

Java Servlet 技術 vs. JSP 技術

- **Java Servlet** 專注重點在程式邏輯
 - 在程式語言中產生**HTML**顯示頁面輸出
 - 適合程式設計人員製作**Controller**元件(商業邏輯)
- **JSP** 專注重點在產生**HTML**網頁
 - 在**HTML**中嵌入部份程式邏輯標籤
 - 適合網頁設計人員製作**view**元件(網頁顯示)
- **Servlet + JSP**技術
 - 網路應用程式區分為表示層和商業邏輯部份
 - 網頁設計人員不需具備深度**Java**程式語言技術
 - **Java** 技術程式人員不需具備網頁設計美工能力



```
@WebServlet( urlPatterns = {"/HelloServlet"})
public class HelloServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest request,
        HttpServletResponse response) throws IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<HTML>");
        out.println("<HEAD>");
        out.println("<TITLE>Hello Servlet</TITLE>");
        out.println("</HEAD>");
        out.println("<BODY>");
        out.println("<H1>Hello World! I am a servlet, it is " +
            new java.util.Date() + "</H1>");
        out.println("</BODY>");
        out.println("</HTML>");
        out.close();
    }
}
```

```
<%@page contentType="text/html"
    pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC
    "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type"
        ontent="text/html; charset=UTF-8">
    <title> JSP Page</title>
  </head>
  <body>
    <h1>Hello World! I'm a JSP, it is
        <%= new java.util.Date() %></h1>
  </body>
</html>
```

JSP 網頁內容

■ 靜態內容

- 標準HTML 標籤：內容原封不動輸出

■ 動態內容

- Scripting元素：需經編譯及解譯後，動態產生輸出內容

Scripting元素	範例	說明
Expression 表示式標籤	<code><%= expression %></code>	定義任何有效的Java運算式 當網頁被請求時運算,運算結果包含在回應之中

■ 隱含變數Implicit Variables

變數名稱	型別	用途	Scope
request	javax.servlet. HttpServletRequest	包含客戶端送出的 HTTP 請求訊息	request

取得HTTP Request 請求參數

■ HTTP GET request 請求參數

http://localhost:8080/Mod3/index.jsp?customerName
=Donald+Trump

■ request.getParameter("paramName")

```
<html>
  <head>
    <meta http-equiv="Content-Type"
      content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Hello
      <%= request.getParameter("customerName") %>
    </h1>
  </body>
```

取得HTTP Request 請求參數

The screenshot displays the Apache NetBeans IDE 11.3 interface. The main editor shows the source code of `index.jsp`. The code defines a JSP page that outputs "Hello" followed by the value of the `customerName` request parameter. If the parameter is null, it outputs "Hello null".

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<html>
<head>
    <meta http-equiv="Content-Type"
        content="text/html; charset=UTF-8"%>
    <title>JSP Page</title>
</head>
<body>
    <h1>Hello
        <%= request.getParameter("customerName") %>
    </h1>
</body>
</html>
```

Three browser windows are shown, demonstrating the output of the JSP page:

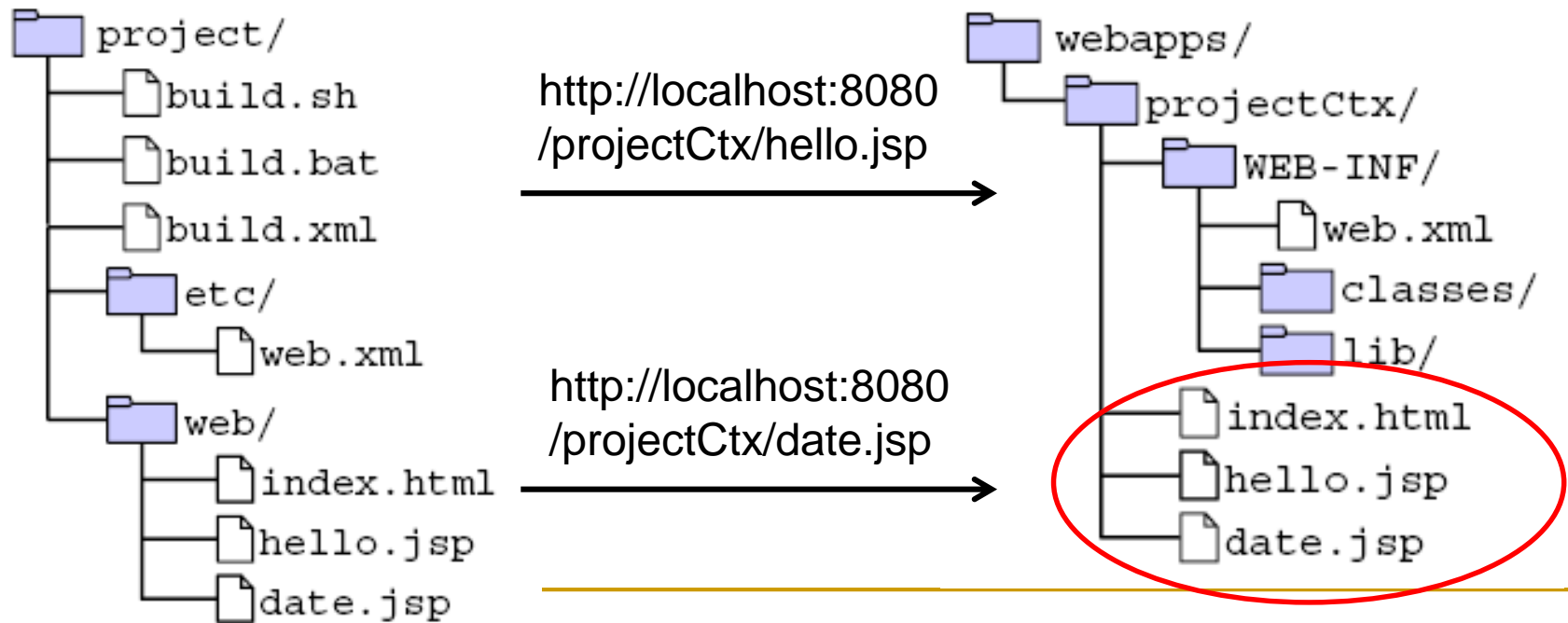
- Window 1: `localhost:8080/Mod3/` displays "Hello null".
- Window 2: `localhost:8080/Mod3/index.jsp?customerName=Amy` displays "Hello Amy".
- Window 3: `localhost:8080/Mod3/index.jsp?customerName=Lebron+James` displays "Hello Lebron James".

The IDE's Project and Navigator views are also visible, showing the project structure and the current file being edited.

JSP 網頁開發與佈署

■ JSP網頁佈署

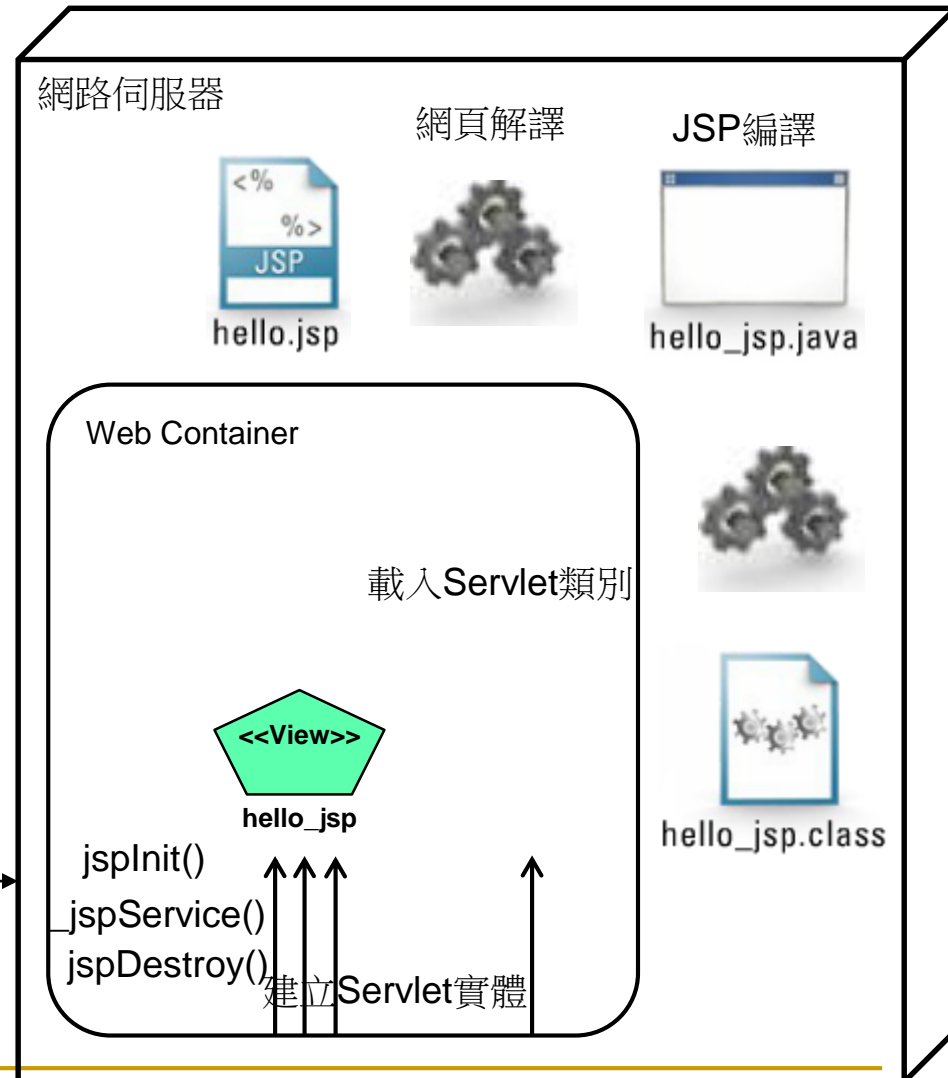
- 與靜態網頁置於相同的位置
- 不需特定宣告及設定
- 容器可察覺JSP網頁更新, 自動重新解譯及編譯



JSP Page Life Cycle

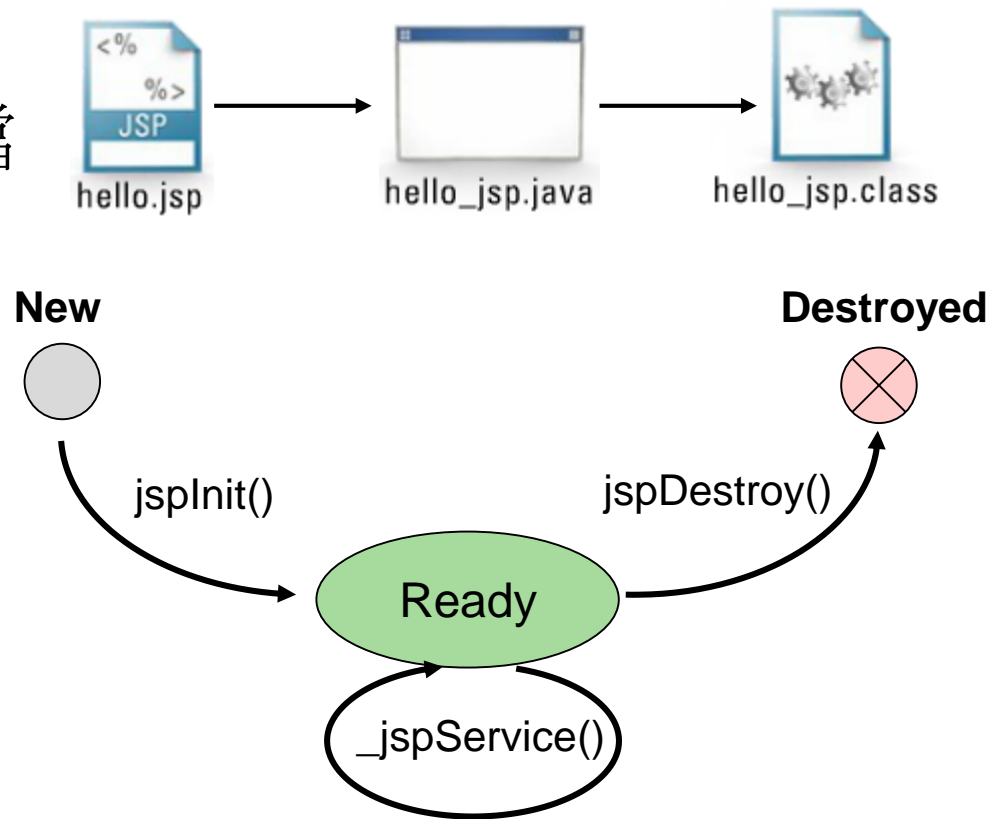
1. JSP解譯為Servlet
2. Servlet編譯為類別檔
3. 載入Servlet類別
4. 建立Servlet實體
5. 呼叫 `jspInit()`
6. 呼叫 `_jspService()`
7. 呼叫 `jspDestroy()`

<http://localhost:8080/hello.jsp>

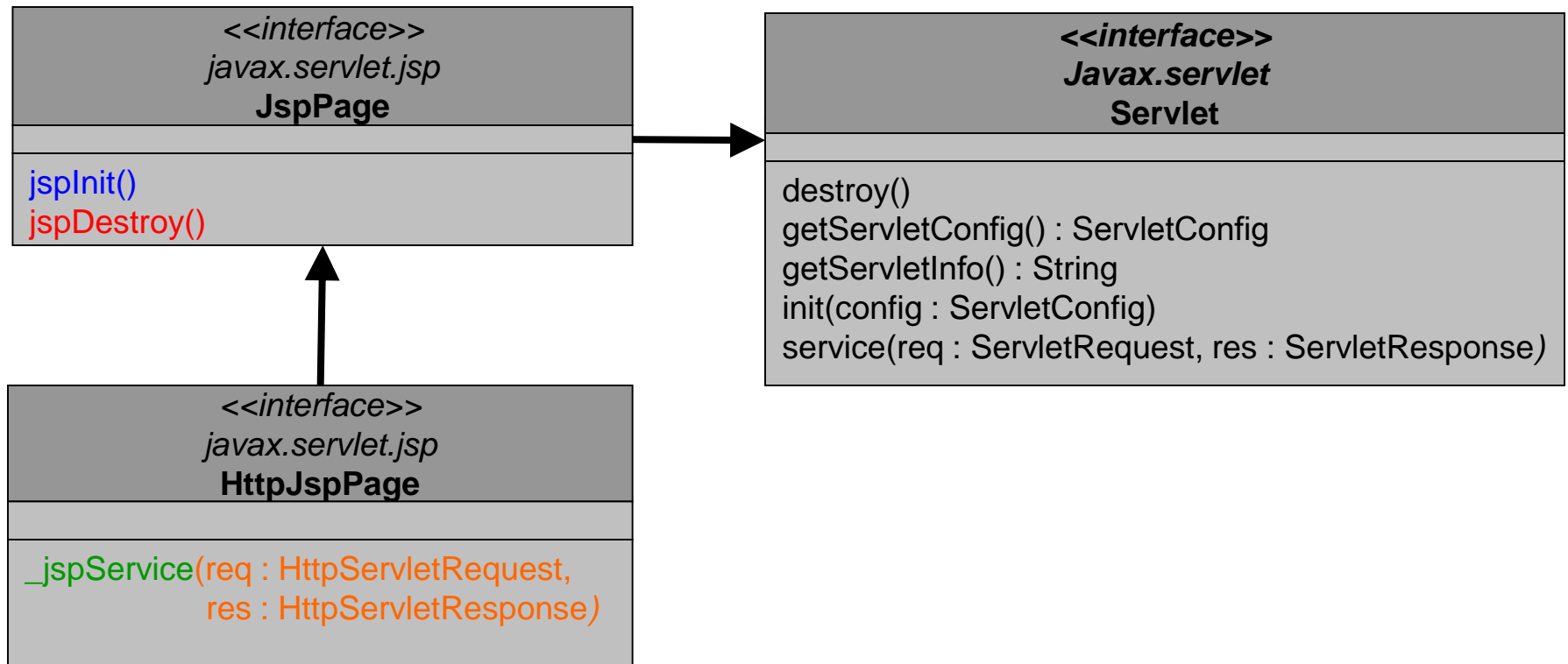


JSP Page Life Cycle

1. JSP解譯為Servlet
2. Servlet編譯為類別檔
3. 載入Servlet類別
4. 建立Servlet實體
5. 呼叫 `jspInit()`
6. 呼叫 `_jspService()`
7. 呼叫 `jspDestroy()`



JspPage – Lift Cycle Methods



課程大綱

1) Java Server Page 網頁技術

2) **JSP scripting**文稿元素

- **Scriptlet**標籤

- 表示式標籤

- 宣告標籤

- 指令標籤

- 註解標籤

JSP 網頁內容

- 靜態內容
 - 標準HTML 標籤：內容原封不動輸出
- 動態內容
 - Scripting元素：需經編譯及解譯後，動態產生輸出內容

Scripting元素	範例	說明
Scriptlet 標籤	<code><% code %></code>	定義 _jspService()中的程式碼
Expression 表示式標籤	<code><%= expression %></code>	定義任何有效的Java運算式 當網頁被請求時運算,運算結果包含在回應之中
Declaration 宣告標籤	<code><%! declaration %></code>	宣告解譯後之servlet 類別的成員(屬性,方法,巢狀類別)
Directive 指令標籤	<code><%@ directive %></code>	定義解譯階段所需的JSP組態訊息
Comment 註解標籤	<code><%-- comment --%></code>	JSP 網頁註解 只在JSP 網頁檔案裏面可見。JSP解譯為Servlet時，標籤中內容會被忽略

Scriptlet Tag

■ `<% JavaCode %>`

- ❑ 撰寫 `_jspService()` 中動態內容程式碼
- ❑ 可定義
 - 區域變數 local variable
 - 分支結構(if-else, switch)
 - 迴圈結構(for, while, do)
 - 呼叫其他方法

```
<% if ( i > 10 ) { %>  
    I am a big number.  
<% } else { %>  
    I am a small number  
<% } %>
```


Table of numbers squared:	
member	squared
0	0
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81

```
<TABLE BORDER='1' CELLPADDING='5'>  
<TR><TH>number</TH><TH>squared</TH></TR>  
<% for ( int i=0; i<10; i++ ) { %>  
<TR><TD><%= i %></TD><TD><%= ( i * i ) %></TD></TR>  
<% } %>  
</TABLE>
```

Expression Tag 表示式標籤

■ `<%= JavaExpression %>`

- 可定義有傳回值的Java運算式
- 當網頁被請求時運算,運算結果包含在回應之中

■ `<%=name %>`  `<% out.print(name); %>`

- 運算結果會被轉型為字串,包含在回應之中
- 基本型別運算及轉換
- String類別
- 物件：呼叫toString()

```
<B>Ten is <%= (2 * 5) %></B>
```

```
Thank you, <|><%= name %></|> !
```

```
The current day and time is: <%= new java.util.Date() %>
```

- 運算標籤中不可以使用分號(Semicolons)

Declaration Tag 宣告標籤

■ `<%! Attribute / Method Declaration %>`

□ 宣告解譯後之 **servlet** 類別的成員(屬性,方法,巢狀類別)

□ 宣告

- instance variables
- instance methods
- class variables
- class methods
- inner classes

```
<%! public static final String DEFAULT_NAME = "World";%>  
<%!  
    public String getName(HttpServletRequest request) {  
        return request.getParameter("name");  
    }  
%>  
<%! int counter = 0; %>
```

□ 可用來覆寫 **life cycle methods**

- `jspInit()`
- `jspDestroy()`
- ~~`_jspService()`~~

```
<%!  
    public void jspInit() {  
        /* 初始化程式碼 */  
    }  
    public void jspDestroy() {  
        /* 清除資源的程式碼 */  
    }  
%>
```

指令標籤 Directive Tag

- `<% @ DirectiveName [attr="value"]* %>`
 - 提供JSP解譯器某些影響JSP頁面轉換為Servlet程式碼時所需資訊
 - 定義解譯階段所需的JSP頁面組態訊息
 - JSP中包含其他JSP
 - 自訂標籤函式庫資訊

JSP 標準指令	用途	範例
page	設定JSP頁面組態資訊	<code><% @ page import = "java.util.*" %></code>
include	在JSP中包含其他JSP的內容	<code><% @ include file = "left.html" %></code>
taglib	在JSP中使用自訂標籤函式庫	<code><% @ taglib prefix = "abc" uri = "taglib.tld" %></code>

Page 指令屬性

屬性	說明	範例
language	JSP中使用語言種類	<%@ page language="java" %>
extends	解譯後之servlet 繼承之父類別	<%@ page extends="myServlet" %>
import	匯入之套件及類別	<%@ page import="java.util.*" %>
session	是否包含在連線階段中	<%@ page session="true" %>
buffer	輸出緩衝區大小 (none 或 n kb)	<%@ page buffer="none/64kb" %>
autoFlush	緩衝區滿了,是否自動輸出(true) 或丟出例外(false)	<%@ page autoFlush="false" %>
isThreadSafe	頁面是否執行緒安全 false→SingleThreadMode	<%@ page isThreadSafe="true" %>
info	網頁摘要說明	
isErrorPage	是否為錯誤頁面	<%@ page isErrorPage="true" %>
errorPage	當例外發生時所導向之錯誤頁面	<%@ page errorPage="error.jsp" %>
contentType	字元集及MIME型態	<%@ page contentType="text/html"%>
pageEncoding	JSP編碼方式	<%@ page pageEncoding="big5" %>
isELIgnored	是否忽略Expression Language	<%@ page isELIgnored="false" %>

JSP中註解

■ JSP註解

- JSP解譯為Servlet時,此標籤中內容會被忽略

`<%-- JSP comment --%>`

■ Java註解

- 解譯為Servlet時,此標籤中內容會被置於_jspService()中

`<% // Java comment %>`

`<% /* Java comment */ %>`

`<% /** Javadoc comment */ %>`

■ HTML註解

- 解譯為Servlet時,此標籤中內容被置於回應之中

`<!-- HTML comment. -->`

`out.print("<!-- HTML comment. -->");`

JSP中註解

■ JSP註解標籤

`<%--` 這是JSP 註解，它只會在JSP 程式碼中被看到，
並不會隨著 HTTP 回應輸出 `--%>`

■ Java註解

`<% /*` 這是Java 註解，會出現在servlet 程式碼中，
但不會出現在HTML 回應中 `*/ %>`

■ HTML註解

`<!--` 這是HTML 註解，會在隨著HTTP 回應出現 `-->`

註解型態	出現在JSP page	出現在Servlet 程式原始檔	出現在HTTP response
JSP註解	Yes	No	No
Java註解	Yes	Yes	No
HTML註解	Yes	Yes	Yes

隱含變數Implicit Variables

變數名稱	型別	用途	Scope
request	javax.servlet. HttpServletRequest	包含客戶端送出的 HTTP 請求訊息	Request
response	javax.servlet. HttpServletResponse	傳給客戶端的HTTP 回應	Page
out	javax.servlet.jsp. JspWriter	輸出資料流物件,繼承 java.io.Writer	Page
session	javax.servlet.http. HttpSession	HTTP 連線階段內容	Session
application	javax.servlet. ServletContext	網路應用程式的組態訊息	application

隱含變數Implicit Variables

變數名稱	型別	用途	Scope
config	javax.servlet. ServletConfig	代表解譯後Servlet之組態 訊息	Page
pageContext	javax.servlet.jsp. PageContext	JSP網頁資訊	Page
page	java.lang.Object	Servlet之物件參考,相當於 this	Page
exception	java.lang.Throwable	其他JSP所拋出之例外事件, 只在錯誤頁面中存在	Page

Lab - 觀察JSP解譯過程

- 使用NetBeans IDE，建立Mod3lab 網路應用專案
 - 刪除 index.html
- 撰寫一些簡單的JSP程式
 - index.jsp
- 測試、執行
- 檢視 JSP 解譯後的 Java程式碼
 - 選取JSP檔案,按滑鼠右鍵,選擇 View Servlet

Lab - 觀察JSP解譯過程

- 對JSP作下列修改,檢視解譯後Java程式碼的變化
 - 使用註解標籤(Comment Tag)
 - 使用指令標籤(Directive Tag), `import java.util.*`
 - 使用宣告標籤(Declaration Tag),宣告一個`private int x`
 - 使用宣告標籤(Declaration Tag),覆寫 `jspInit()`方法
 - 使用表示式標籤(Expression Tag),輸出 `x` 的值
 - 使用scriptlet標籤, 列印1到x的值
 - 試著移除 `<% } %>`, 使程式有誤