Java EE7 Web
Java Servlet 技術

鄭安翔

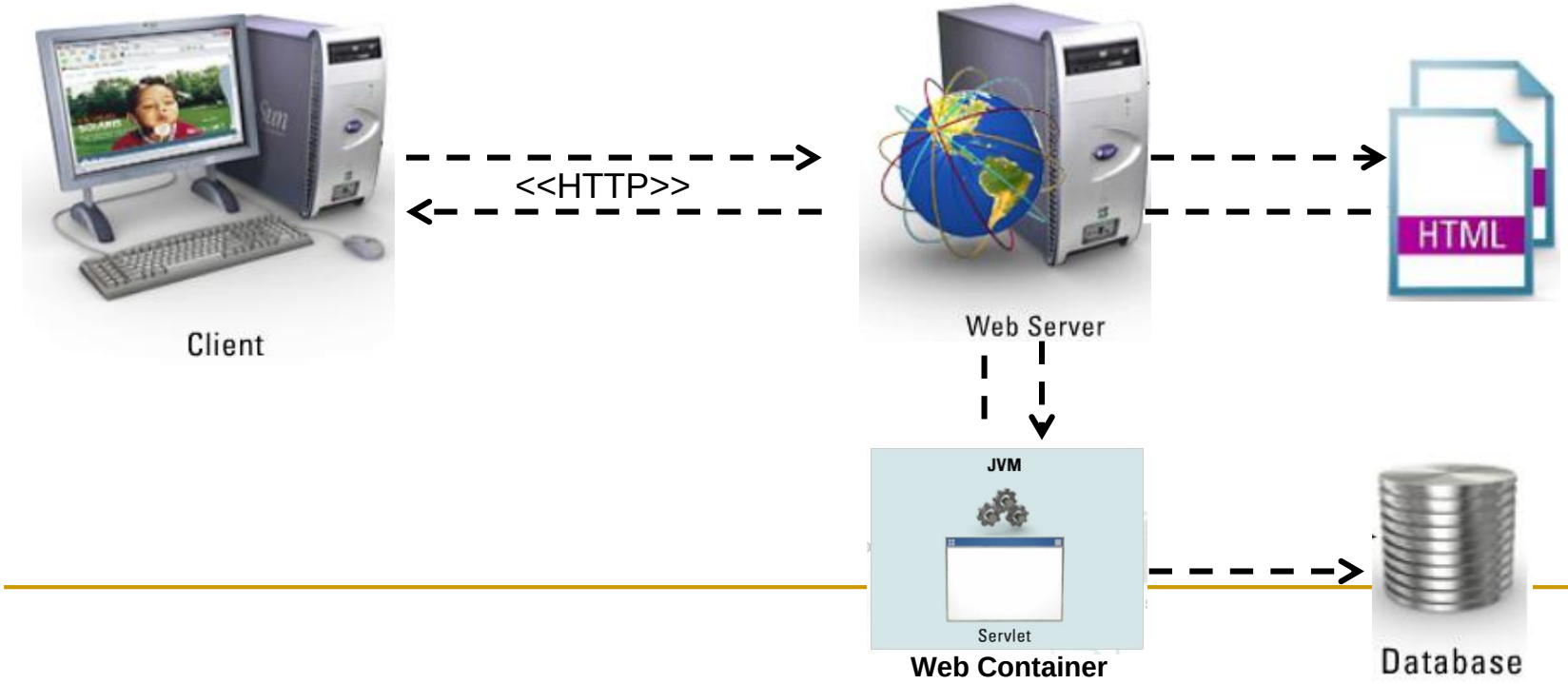
ansel_cheng@hotmail.com

課程大綱

1) **Java Servlet 技術**
   - **Servlet運作架構**
   - **Java Servlet 範例**
2) HTTP 通訊協定
3) Web Container 網路元件容器架構

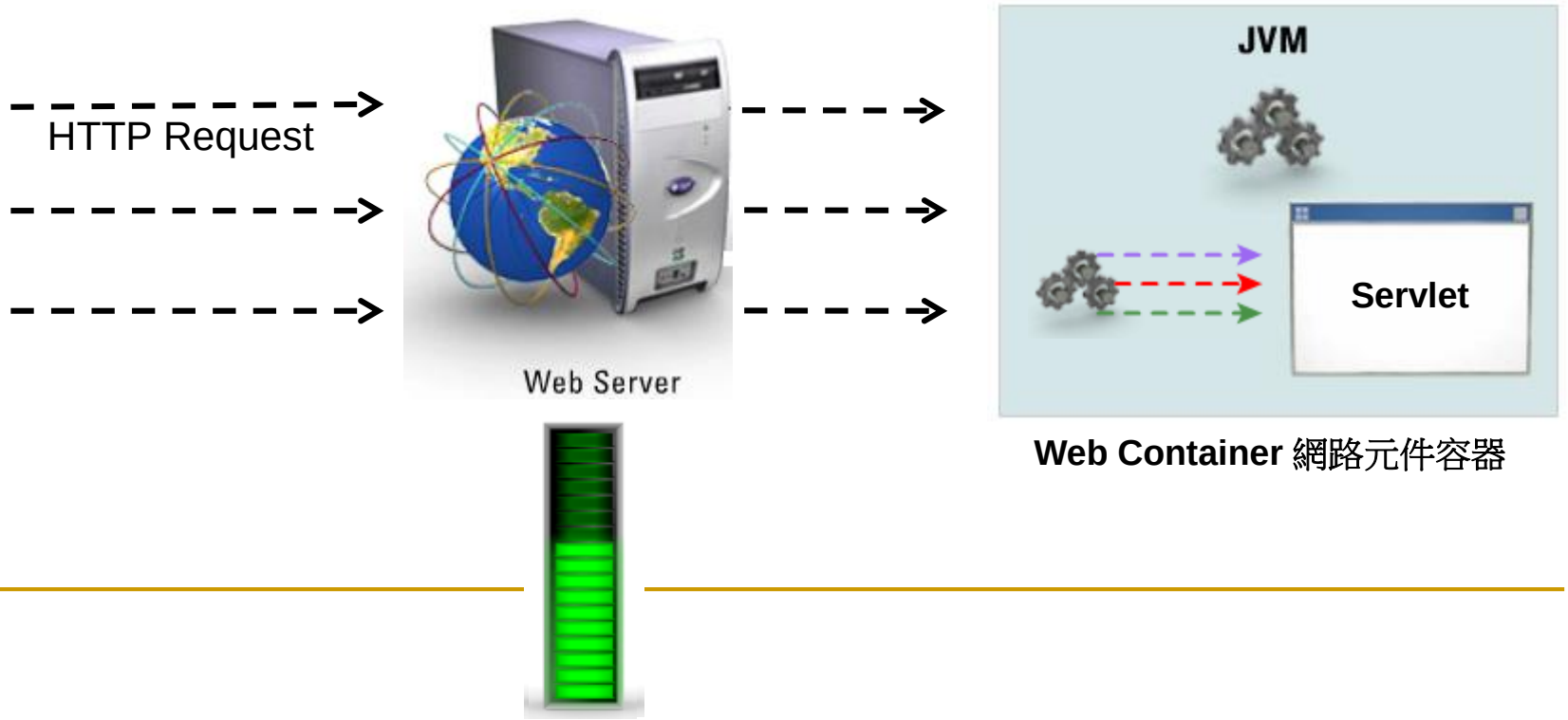
Java 語言網路應用程式

- Java Servlet 來支援動態網路應用程式開發
- 執行的步驟與CGI程式相似
- 底層執行架構不同
  - 單一程序(Process),多執行緒(Threads)



Java Servlet 執行架構

- Java Servlet 需在網路元件容器中執行
  - 網路元件容器是一個作業系統程序
  - 以服務方式執行,持續存在
  - Servlet在網路元件容器中以一個執行緒執行

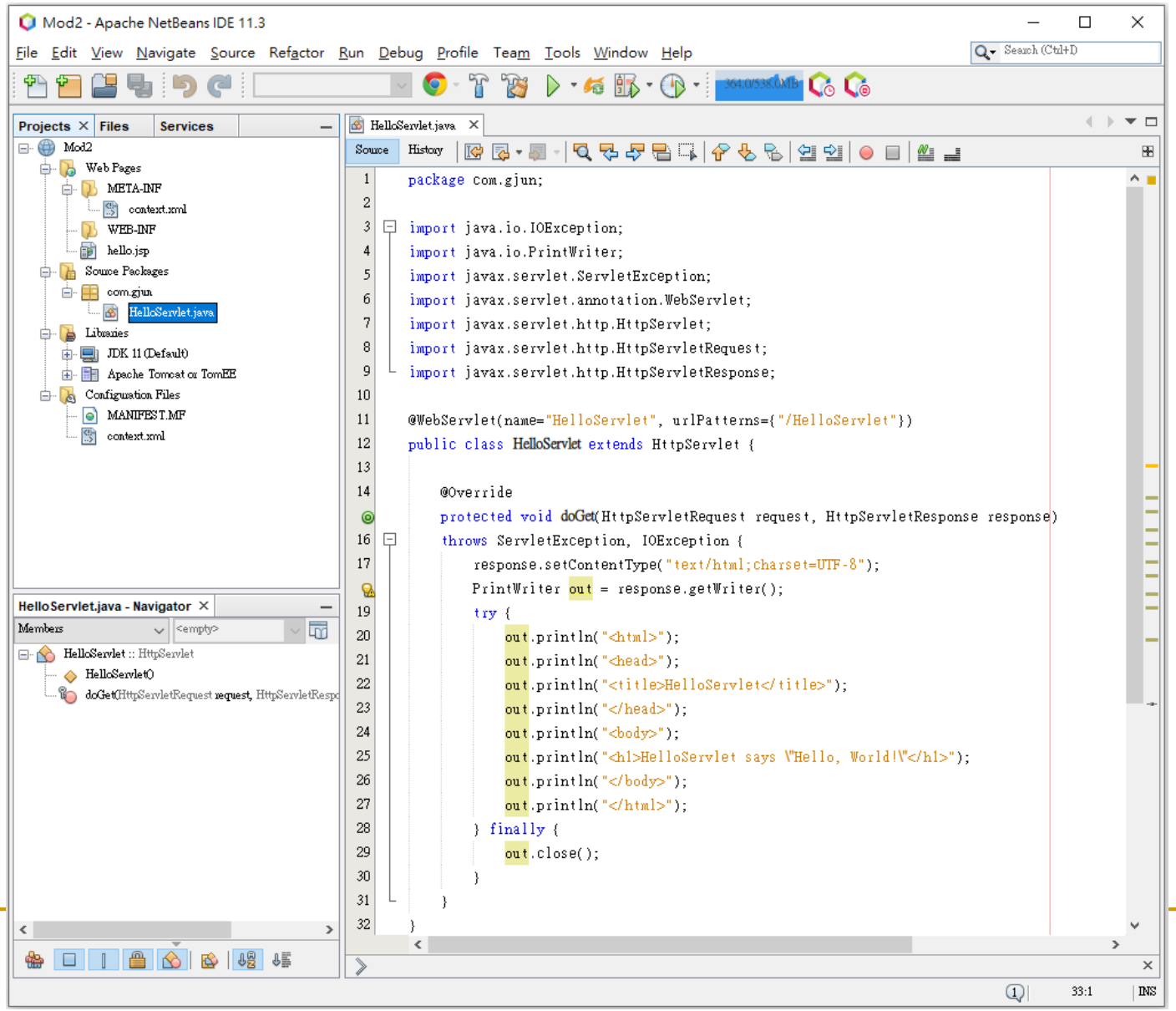


Web Container 網路元件容器

Java Servlet 技術優缺點

- Java Servlet 技術優點
  - 程式回應時間較快, 單一程序中多執行緒
  - 擴充性較佳, 不受限於作業系統程序的資源
  - 物件導向語言
    - 程式可重複使用
      - 基本的請求回應處理直接繼承, 不需撰寫
    - 亦不受限作業系統平台提供的功能
  - 網路容器提供額外的功能
    - 安全性、例外處理及日誌(logging)存取機制
- 缺點
  - 需自行處理並行及同步的問題
  - 同時處理商業邏輯與展示層,不易維護

第一支 Java Servlet 範例



第一支 Java Servlet 範例

HelloServlet

```
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
@WebServlet(name="HelloServlet", urlPatterns={"/HelloServlet"})
public class HelloServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
                                                    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            out.println("<html>");
            out.println("<head>");
            out.println("<title>HelloServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>HelloServlet says \"Hello, World!\"</h1>");
            out.println("</body>");
            out.println("</html>");
        } finally {
            out.close();
        }
    }
}
```

設定URL與Servlet元件的對應

- 設定請求URL對應的Servlet元件
  - 網路元件容器Servlet Container用以尋找並執行網路程式元件
  - Servlet 3.0可使用annotation標註

```
@WebServlet ( name = "ServletName",
                    urlPatterns = { "URL" },
                    loadOnStartup = 1     )
```

  - Servlet 3.0 之前需使用佈署描述檔 Deployment Descriptor(web.xml)

Servlet 運作流程

- 客戶端瀏覽器對HTTP伺服器發出HTTP請求。
  - HTTP伺服器收到HTTP請求，將請求轉送網路元件容器處理
- 網路元件容器
  - 剖析HTTP請求內容，建立HttpServletRequest、HttpServletResponse 物件
  - 網路元件容器由請求的URL轉送至適當Servlet來處理請求
    - 依據請求的HTTP方法呼叫Servlet對應的處理方法
- Servlet
  - 依據請求物件 (HttpServletRequest) 取得客戶請求資訊
  - 透過回應物件 (HttpServletResponse)來建立回應。
    - PrintWriter out = response.getWriter();
    - out.println("<HTML>…</HTML>");

HTTP Methods

常用HTTP請求方法	用途	Servlet 中對應方法 處理使用者HTTP請求
Get 方法	取得Server中指定位置的檔案	protected void doGet (HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException
Post 方法	請求Server中指定位置之程式處理	protected void doPost (HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException

- NetBeans 將doGet()及doPost()方法轉送至同一方法
  - 實務上,HTTP請求Get方法與Post方法處理邏輯經常是相同

```
protected void processRequest(HttpServletRequest request,
                              HttpServletResponse response)
    throws ServletException, IOException
```

常用HTML標籤

語法	說明	語法	說明
<html></html>	文件的開始與結束	<head></head>	標示文件標頭資訊
<title></title>	文件標題	<body></body>	顯示文件本文
 	換行		分段
<hr>	加上分隔線		置中對齊
<h1>~<h6>	設定文字標題大小	<pre></pre>	使用原有文字格式
<font color="rrggbb">	設定字體顏色	<font size="n">	設定字體大小
<u></u>	文字加上底線		粗體字效果
	非排序性列表		斜體字效果
	列表的項目		排序性列表
<img src="圖檔URL" width="寬" height="高">	插入圖片並預設圖形大小	說明文字	設定超鏈結
<table></table>	產生表格	<tr></tr>	表格的一行
<th></th>	表格的標頭欄位	<td></td>	表格的資料欄位
<form action="URL" method="get/post">	表單格式	<input type=text>	文字欄位
<input type= submit/ reset/button>	按鈕	<select> <option> </select>	下拉式選項
<input type=checkbox>	多選核取方塊	<input type=radio>	單選核取方塊
<textarea></teatarea>	文字輸入方塊	<!--註解文字-->	HTML文件中的註解

Lab 1

- 使用NetBeans IDE，建立Mod2lab1 網路應用專案
- 建立一個Servlet
  - 刪除index.html
  - 新增一個com.gjun.Lab1Servlet
- 檢視NetBean建立的Servlet程式
  - doGet() / doPost() / processRequest()
- 撰寫Servlet程式碼
  - 自行撰寫
- 執行Servlet
- 使用瀏覽器檢視動態網頁

課程大綱

1) Java Servlet 技術
2) **HTTP 通訊協定**
   - **傳送及取得HTTP請求參數**
   - **HTTP 方法**
3) Web Container 網路元件容器架構

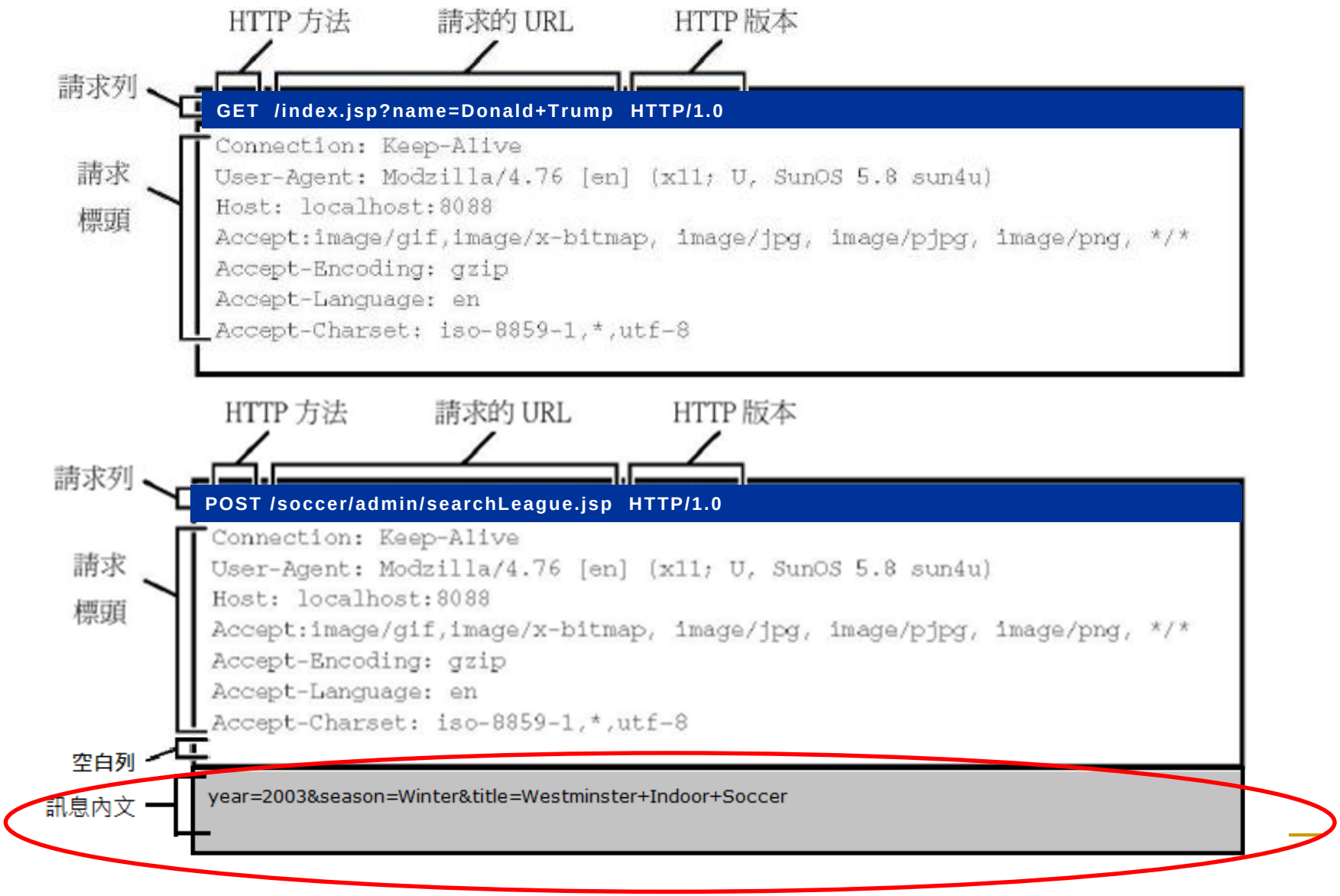
HTTP 方法

HTTP Method	用途
GET	取得Server中指定位置的檔案
POST	將資訊包含在Request中送給Server中指定位置之程式處理
PUT	將資訊儲存至Server指定位置
HEAD	與GET方法相似,但只取標頭(Header)訊息,不含內容
DELETE	刪除Server中指定位置的資源
TRACE	追蹤Request 傳送路徑
CONNECT	取得Server中指定位置可使用之溝通形式
OPTIONS	保留作為與代理伺服器通訊時使用

GET vs. POST

	GET	POST
啟動方式	在網址欄輸入 HTML 超連結<A HREF= …> Form 中指定Method為GET Form未指定Method	Form 中指定Method為POST
資料傳輸	附加在Request URL之後	置於Message Body之中
用途	取的Server上的資料	改變Server上的資料
使用時機	Form Data數量不多 Bookmark Request	Form Data數量很多 Data包含安全性資料

HTTP Request



取得HTTP Request 請求參數

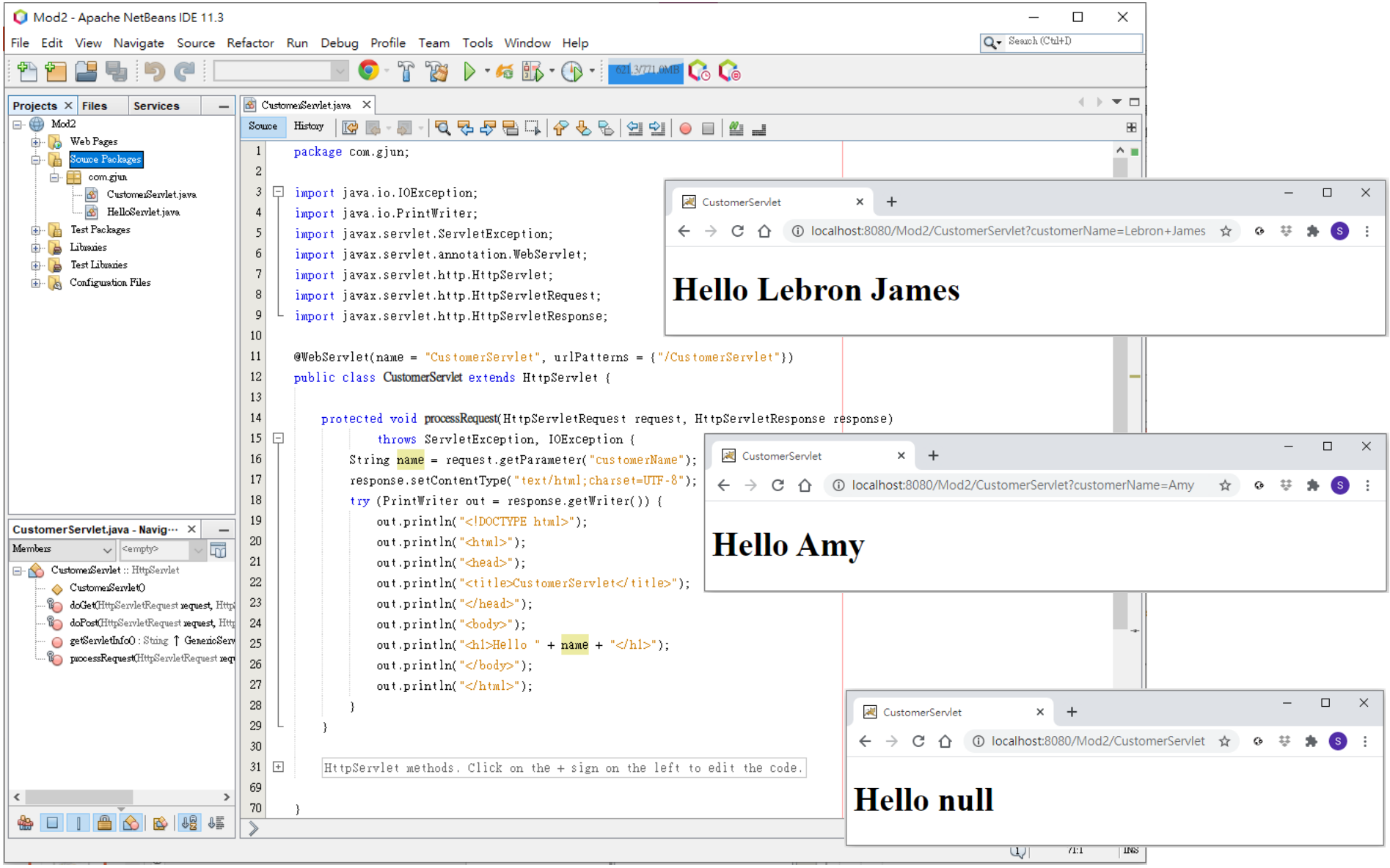
- HTTP GET request 請求參數

http://localhost:8080/Mod2/CustomerServlet?customerName=Sean

- request.getParameter("*paramName*");

```
String name = request.getParameter("customerName");
response.setContentType("text/html;charset=UTF-8");
try (PrintWriter out = response.getWriter()) {
    out.println("<!DOCTYPE html>");
    out.println("<html>");
    out.println("<head>");
    out.println("<title>CustomerServlet</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Hello " + name + "</h1>");
    out.println("</body>");
    out.println("</html>");
}
```

取得HTTP Request 請求參數



HTML FORM 表單

- 一個HTML檔案中可包含多個表單 FORM
  - 將多個的GUI元件集合成一組
- 一個表單中有多個input元件
  - 不同的input type代表不同外觀的元件
  - 不同表單的input元件不能共用
- 當請求送出時,Form Data會被包裝成鍵值對送至Web Server

HTML FORM 標籤

<FORM ACTION='*servlet-URL*' METHOD='{*GET*|*POST*}'>
  {HTML form input tags or other HTML content}*
</FORM>

- ACTION – 指定Form 送出後的目的地 (相對路徑)
- METHOD – 指定HTTP方法 (GET or POST)

https://www.w3schools.com/html/html_forms.asp

FORM Data：INPUT 標籤

< INPUT TYPE='*Type*'
NAME='*Name*' [VALUE='*Value*'] >

- TYPE
  - 文字型態：text, password, hidden
  - Button型態：submit, reset
  - 選擇型態：checkbox, radio
- Name
  - Input Tag中的屬性名稱
  - 請求中的參數鍵值
- VALUE
  - 傳送至伺服端的參數值
  - 內容依input tag 型態而有不同

Input 標籤元件型態 Component

Component	Tag	Description
Text Field	<Input Type='text' ...>	單行文字輸入
Password	<Input Type='password' ...>	單行文字輸入,文字不顯示
Hidden Field	<Input Type='hidden' ...>	表格中不顯示,但會隨請求送至Server
Checkbox	<Input Type='checkbox' ...>	選擇多個選項
Radio Button	<Input Type='radio' ...>	選擇單一選項
Submit Button	<Input Type='submit' ...>	按下後送出表格中資料
Reset Button	<Input Type='reset' ...>	按下後重設表格中資料

其他標籤

- SELECT
  - 下拉是選擇清單

```
<SELECT NAME='select name' [MULTIPLE]>
    <OPTION VALUE='option value' [SELECTED]>
    …
</SELECT>
```

- TEXTAREA
  - 多行文字輸入

```
<TEXTAREA NAME='text area name'
                ROWS='rows' COLUMNS='columns'>
    default content
</TEXTAREA>
```

標籤傳送值

- 標籤傳送值
  - 文字型態：Text, TextArea, Password, Hidden
    - value 內容為輸入欄位預設文字
    - 傳送值為傳送時輸入欄位的文字
    - 空白字元：用+符號代替

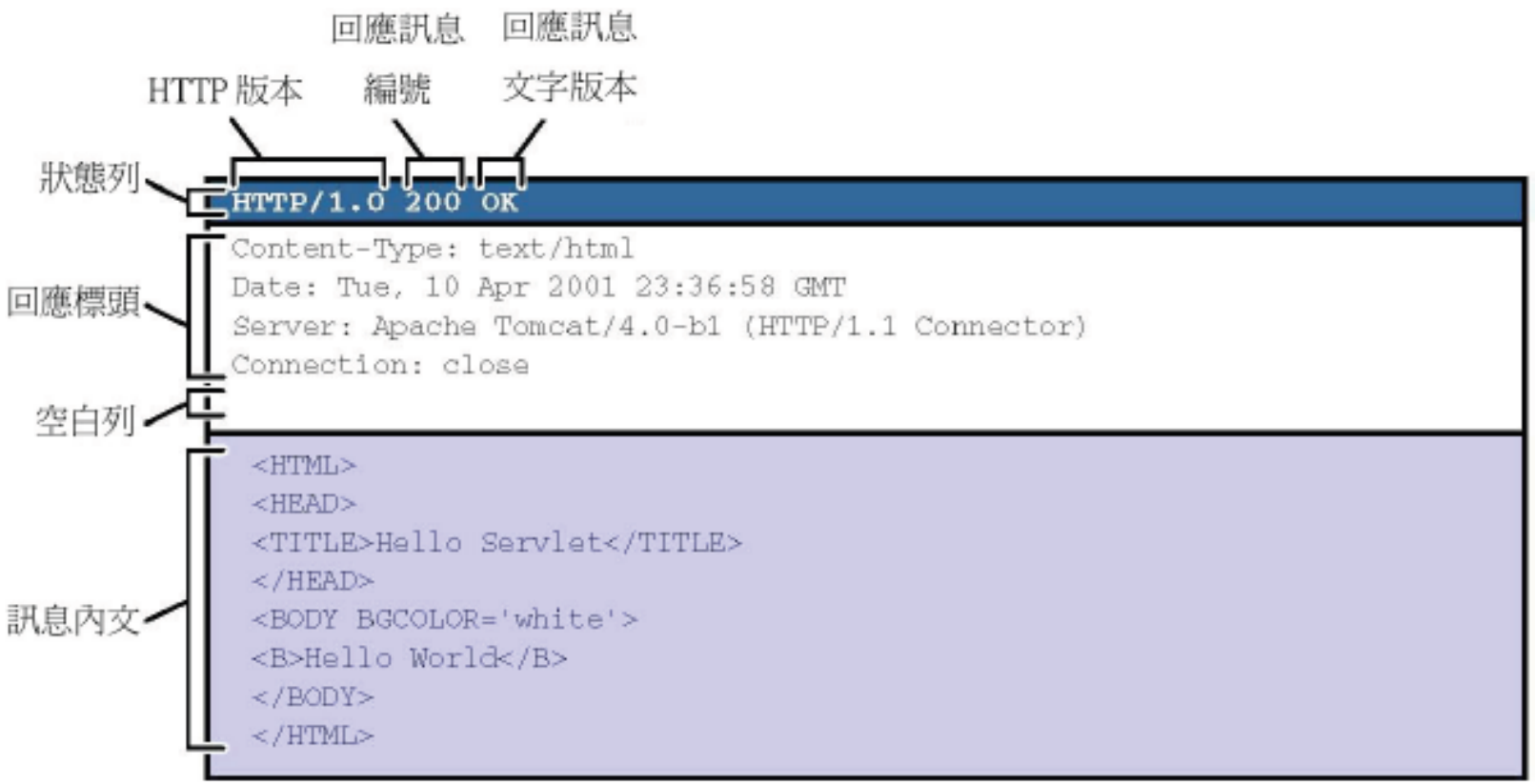
      name = Sean + Cheng
  - 選項型態：Checkbox, Radio, Select
    - 多個標籤,有相同的name屬性,對應不同的value屬性
    - 傳送值為傳送時被選取選項的value屬性內容
    - 多選選項：一個name被選取多個value, 用&符號連接

      fruit = apple & fruit = banana
  - Button按鈕型態
    - Submit, Reset, Button
    - value是按鈕上的文字,沒有資料送出

HTTP 請求標頭

標頭	使用
Accept	用戶端可以接受的 MIME 型態
Host	網際網路主機和被請求資源的通訊埠編號
Referer	從請求 URI 獲得的地址
User-Agent	關於用戶端產生的請求資訊

HTTP Response



常用 Response 訊息代碼

代碼	意義
200	OK
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not Found
405	Method Not Allowed
408	Request Time Out
500	Internal Server Error
503	Service Unavailable

HTTP 回應標頭

標頭	使用
Content-Type	在回應中的資料型態歸類的 MIME 型態 (像是 text/html)
Content-Length	回應的承載長度 (位元組)
Server	回應這個 HTTP 請求的伺服器有關的資訊字串
Cache-Control	網頁瀏覽器 (或代理伺服器) 的指令，用來指示回應內容是否應該要快取

Lab 2

- 使用NetBeans IDE，建立Mod2lab2 網路應用專案
- 建立index.html檔案
  - 修改 index.html檔案,檔案中包含下面的表單Form

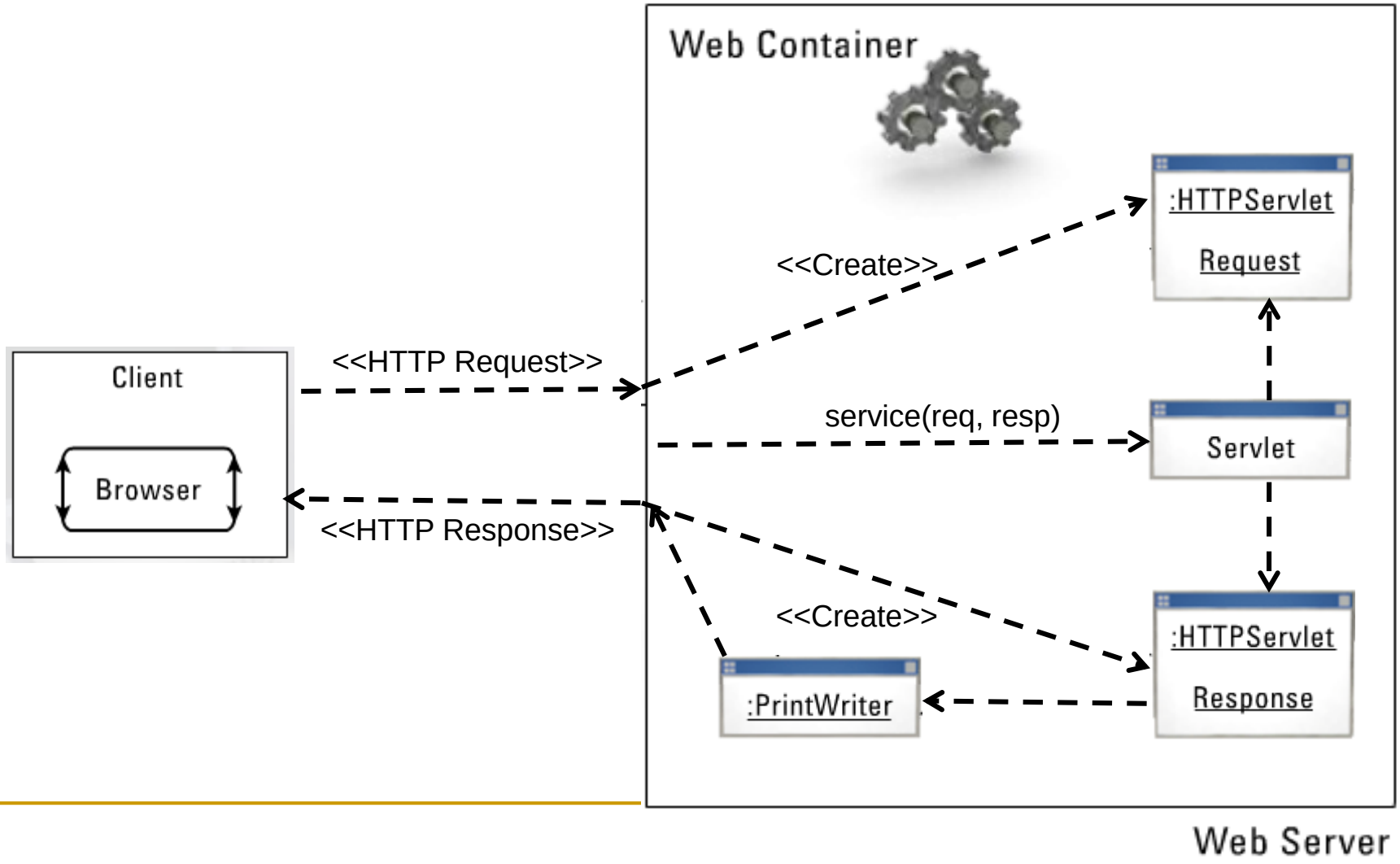
```
<form action="AnimalServlet">
    Please enter your favorite animal: 
    <input type="text" name="favoriteAnimal" value="Gnu" />
    <input type="submit" />
</form>
```

- 建立AnimalServlet.java程式
  - title 設為Animal Lover’s page
  - <h1>標籤中顯示 Hello Animal Lover
  - 使用內建request物件取得表格中傳來的favoriteAnimal參數,並顯示在頁面中
- 執行專案
  - 送出表單並檢視執行結果

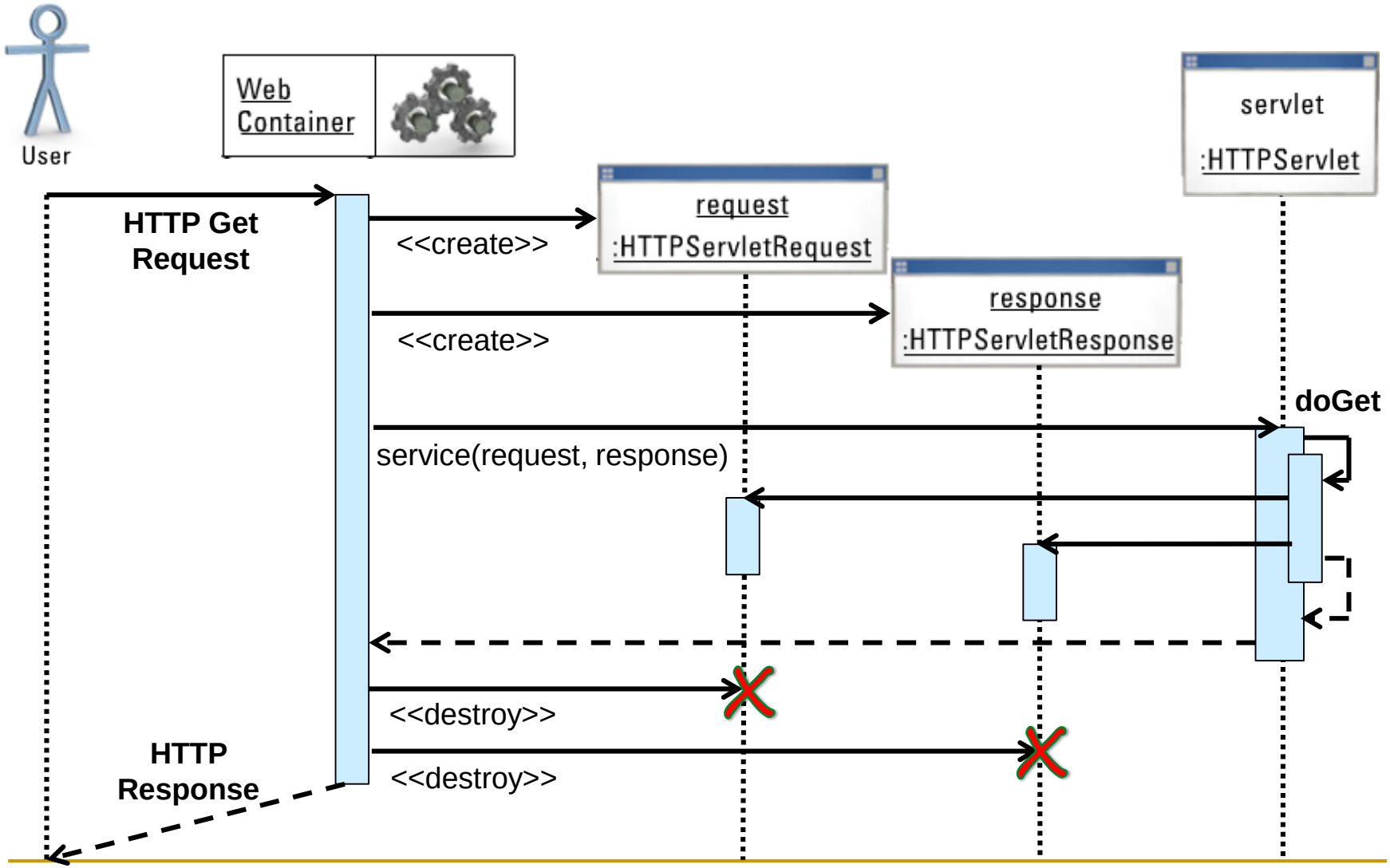
課程大綱

1) Java Servlet 技術
2) HTTP 通訊協定
3) **Web Container 網路元件容器架構**
   - **Servlet API**

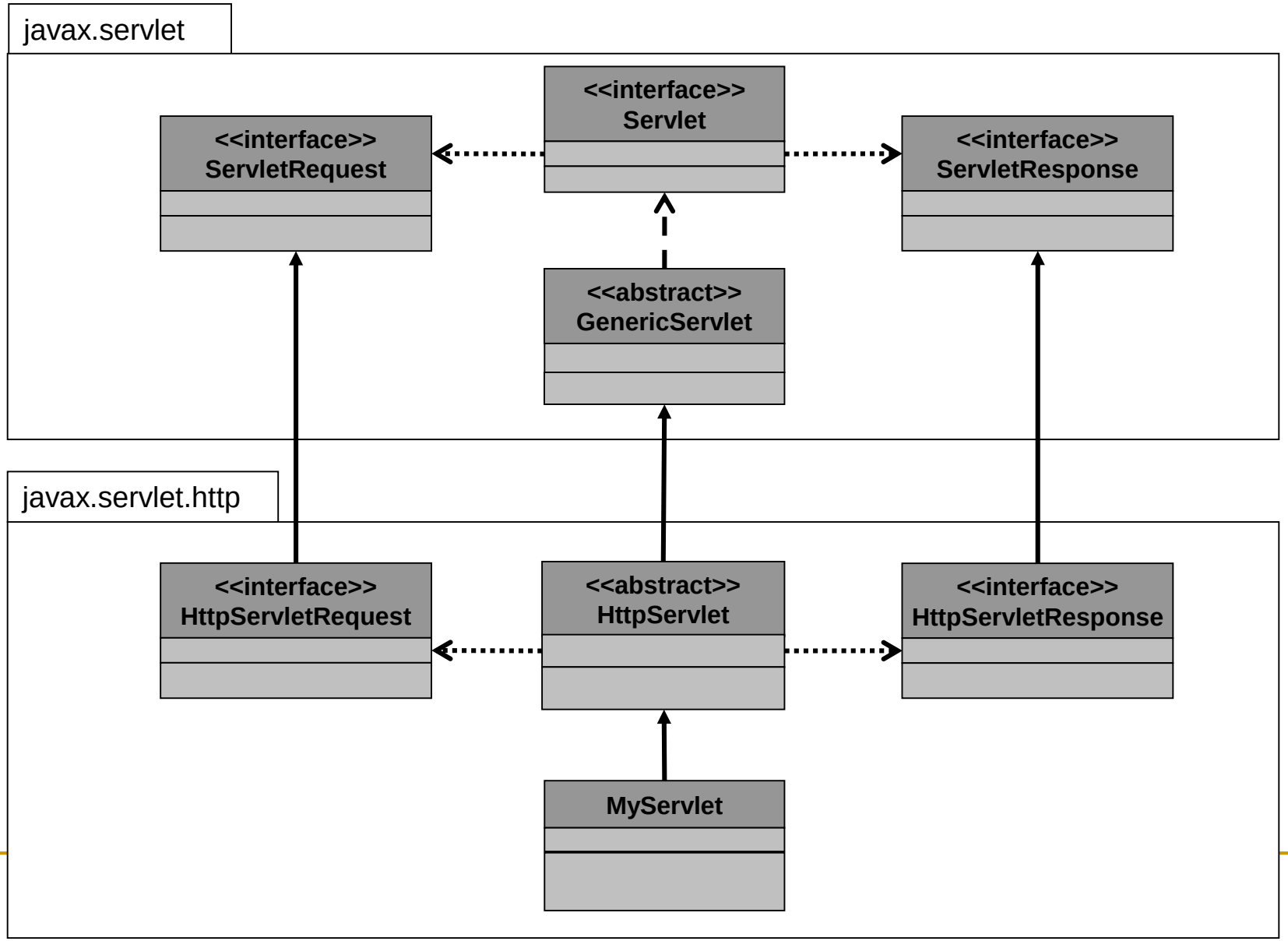
Web Container 網路元件容器架構



元件容器執行流程圖


User
Web Container
servlet
:HTTPServlet
request
:HTTPServletRequest
response
:HTTPServletResponse
HTTP Get Request
<<create>>
<<create>>
doGet
service(request, response)
<<destroy>>
<<destroy>>
HTTP Response

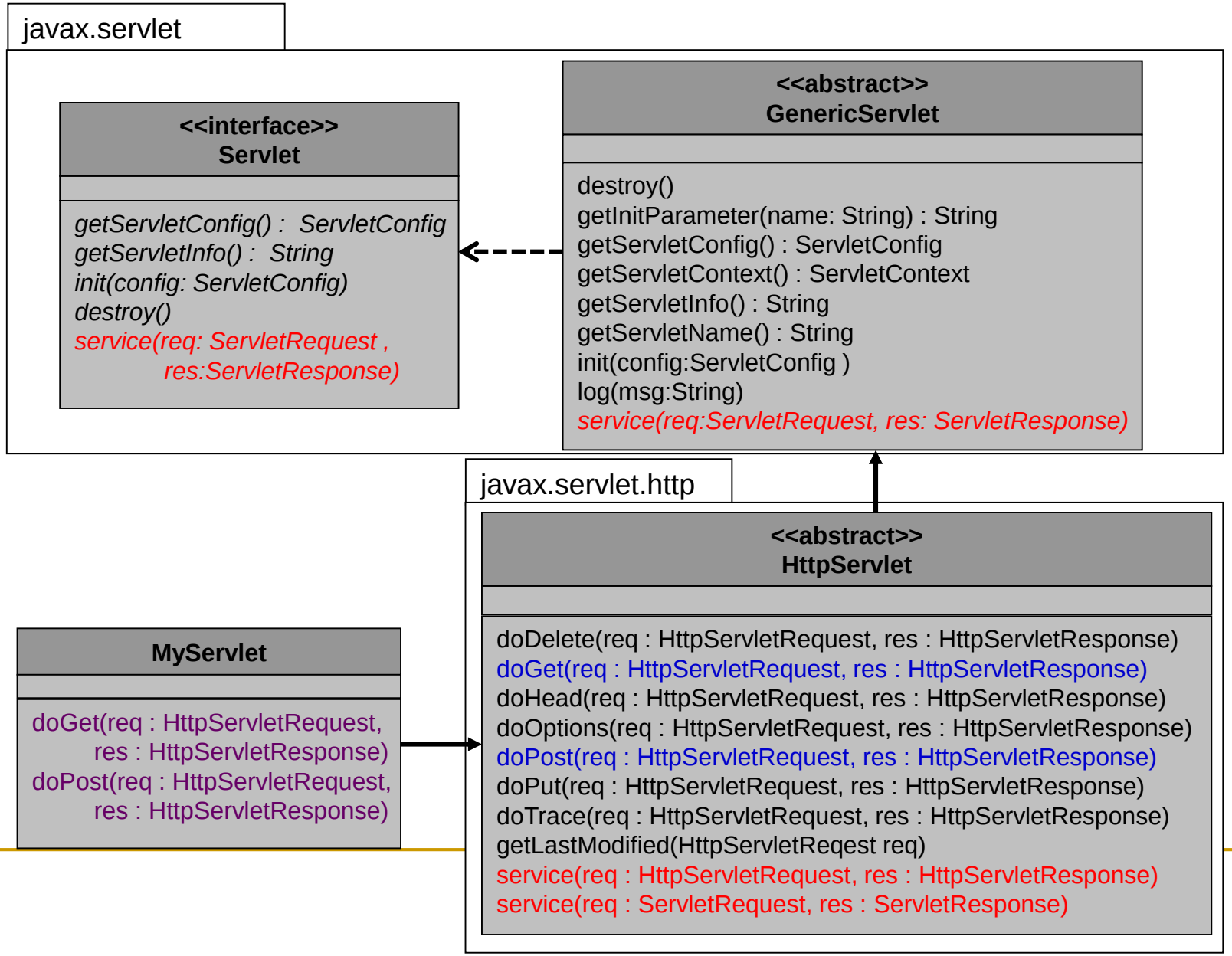
Servlet API


javax.servlet
<<interface>>
Servlet
<<interface>>
ServletRequest
<<interface>>
ServletResponse
<<abstract>>
GenericServlet
javax.servlet.http
<<interface>>
HttpServletRequest
<<abstract>>
HttpServlet
<<interface>>
HttpServletResponse
MyServlet

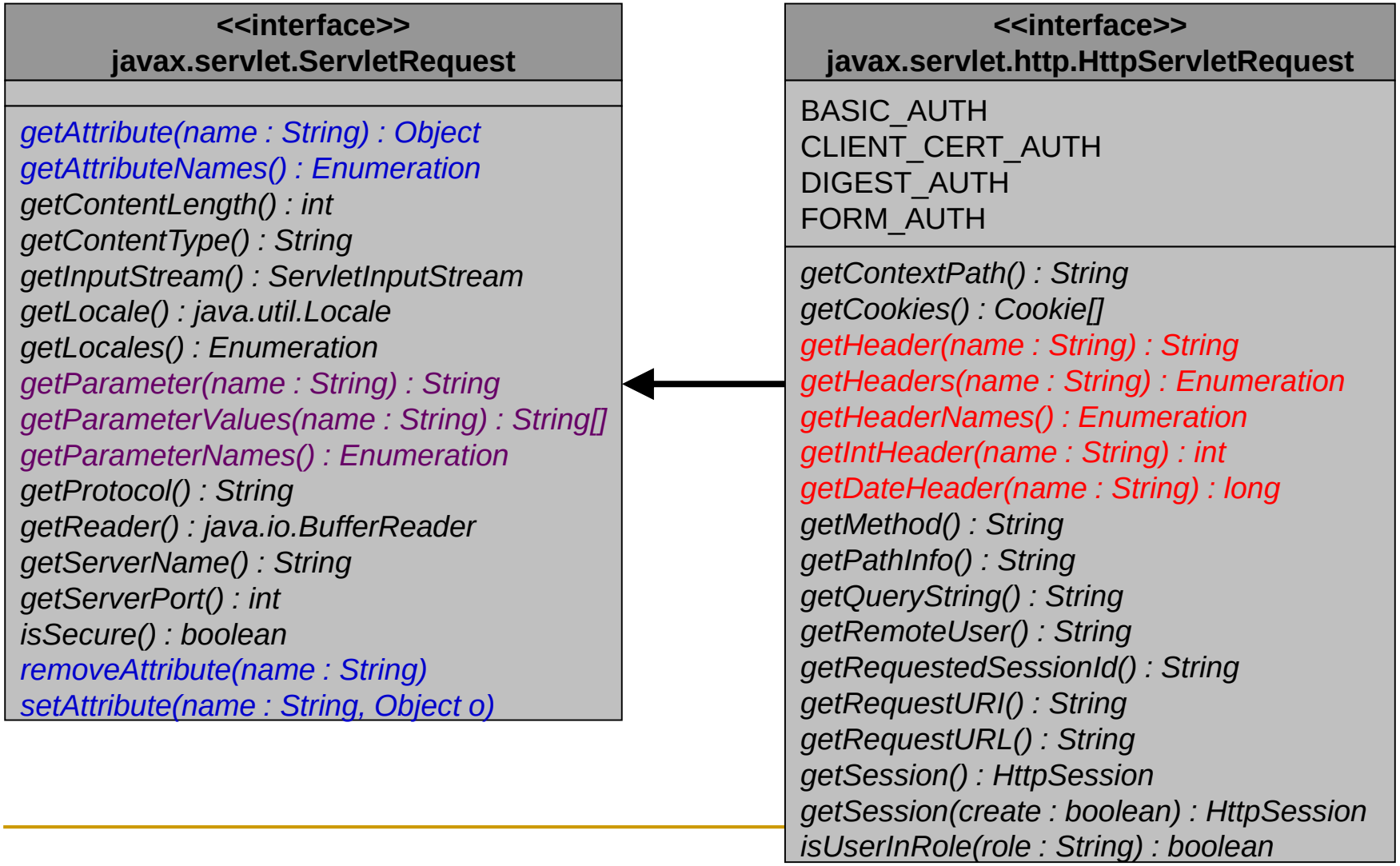
HttpServlet 方法的對應

HTTP 方法	對應的**HttpServlet 方法**
OPTIONS	doOptions(…)
GET	doGet(…)
HEAD	doHead(…)
POST	doPost(…)
PUT	doPut(…)
DELETE	doDelete(…)
TRACE	doTrace(…)
CONNECT	doConnect(…)

javax.servlet.Servlet



HttpServletRequest



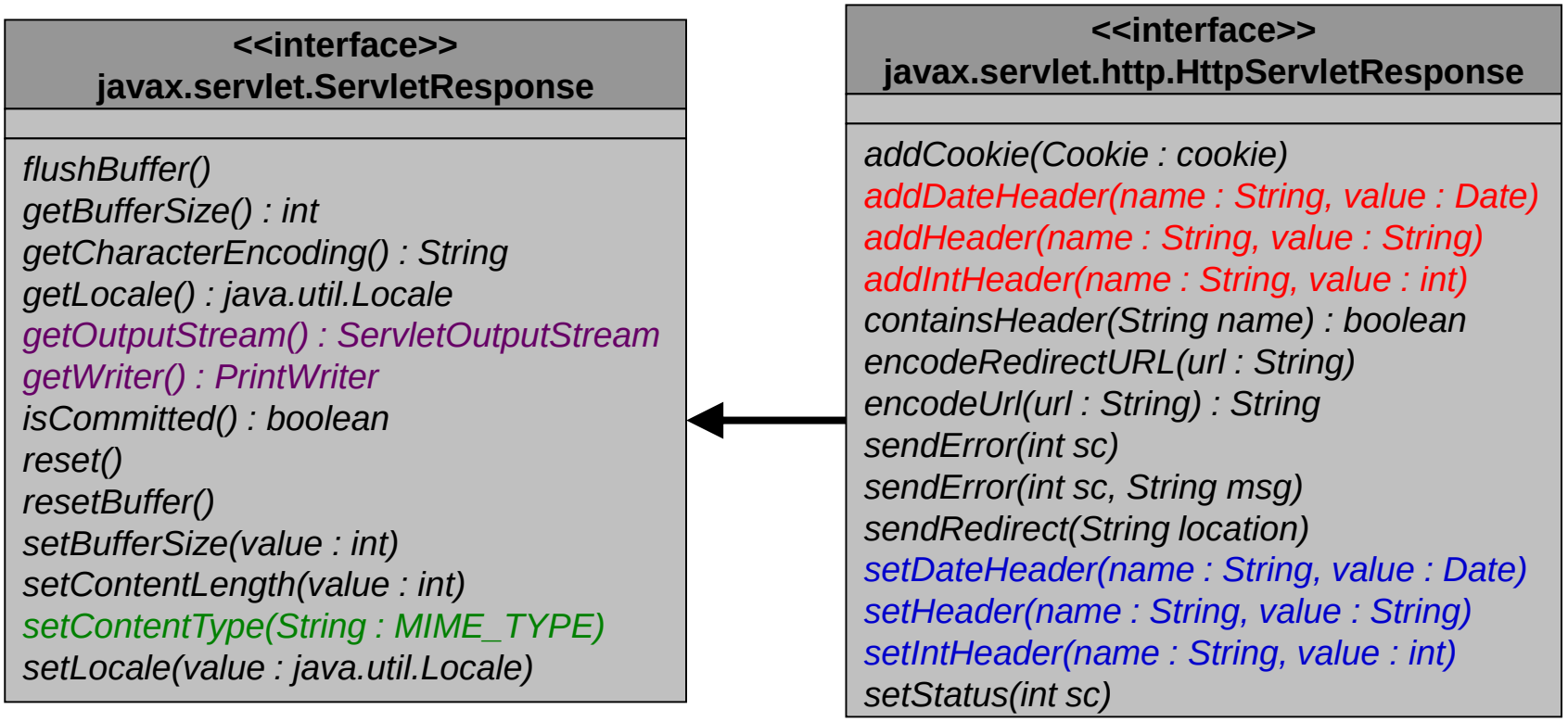
HttpServletRequest - Header

方法名稱	回傳型態	用途說明
getHeader(name: String)	String	取得特定標頭訊息
getHeaders(name: String)	Enumeration	取得特定標頭訊息,回傳值為一Enumeration物件
getHeaderNames()	Enumeration	取得請求中所有標頭訊息,回傳值為一Enumeration物件
getIntHeader(name: String)	int	取得整數型態之特定標頭訊息
getDateHeader(name: String)	long	取得日期型態之特定標頭訊息,回傳值為自1970/01/01 00:00:00至今之毫秒數

Request Header 範例

```
boolean displayXHTML = false;
String userAgent = request.getHeader(“User-Agent”);
if(userAgent.startsWith(“Mozilla/5.0”)) {
      // browser can handle XHTML content
      displayXHTML = true;
}
if(displayXHTML) {
      // XHTML content output here
} else {
      // regular HTML content output here
}
```

HttpServletResponse



ServletResponse

方法名稱	回傳型態	用途說明
setContentType(type : String)	void	設定http回應之內容型態
getWriter()	PrintWriter	取得產生http回應之字串資料流
getOutputStream()	ServletOutput Stream	取得產生http回應之二元資料流

HttpServletResponse

方法名稱	回傳型態	用途說明
addHeader(name:String, value:String)	void	設定http回應之標頭訊息
setHeader(name:String, value:String)	void	修改http回應之標頭訊息
addIntHeader(name:String, value:int)	void	設定http回應之整數型態標頭訊息
setIntHeader(name:String, value:int)	void	修改http回應之整數型態標頭訊息
addDateHeader(name:String, value:long)	void	設定http回應之日期型態標頭訊息
setDateHeader(name:String, value:long)	void	修改http回應之日期型態標頭訊息

Response Header 範例

- Response Header 範例

```
response.setContentType("text/html");
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Cache-Control", "private");
```

- 常用內容型態

常用內容型態
text/plain (default)
text/html
image/jpeg
image/png
audio/au

  - 若Servlet Response未指定內容型態,則型態為 text/plain
    - 大部分Browser會將HTML tag當作內文顯示
  - 網頁顯示中文
    - text/html; charset=Big5

Lab 3

- 使用NetBeans IDE，建立Mod2lab3 網路應用專案
- 編輯index.html
  - 提供一個連結,連結至ListHeaders
- 建立 com.gjun.HeaderServlet.java
  - Servlet URL 為 ListHeaders
  - 取得所有 headers 鍵值對，以下列格式顯示於網頁

```
<UL>
    <LI>key = value</LI>
    ....
    ....
</UL>
```

- 測試、執行