MVC模型

段維瀚 老師

綱　要

- 簡述只使用Servlet的重大缺點
- 撰寫簡單的JSP(Java Server Pages)
- 簡述JSP如何轉譯為Servlet
- 了解MVC的基本目的

Servlet的缺點

- 編輯HTML較困難
- 無法使用所見即所得(WYSIWYG)工具進行設計
  - What You See Is What You Get
- 手動編輯較冗長且容易導致錯誤

使用JSP解決問題

- JSP在HTM中嵌入Java程式

```
<%--
Simple Hello World JSP example
--%>
<%@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>
charset=UTF-8">
    </head>
    <body>
<h1><%= "Hello World! I'm a JSP, it is " + new java.util.Date() %></h1>
</body>
</html>
```

JSP 標籤樣式

- <%@ 指令設定
- <%! 宣告實作區
- <% 程式邏輯片段
- <%= 顯示資料，相當於 out.print();
- <%-- 註解
- ${ } EL

如何處理 JSP

- JSP is Servlet
  - 轉換為相對應的Servlet
  - 將JSP視為「原始碼」
- JSP能存取原本Servlet可使用的變數
  - 包含request與response變數
    - <%=request.getParameter("x") %>
      - 資料來源：http://.../foo?x=100
    - <%=request.getAttribute("y") %>
      - 資料來源：request.setAttribute("y", 200);

如何處理 JSP

- 透過GET請求傳遞的參數含有「空白」
  - http://.../foo.jsp?name=Tuan YU
  - http://.../foo.jsp?name=Tuan+YU
    - 以加號（+）取代空白

JSP方式仍存在問題

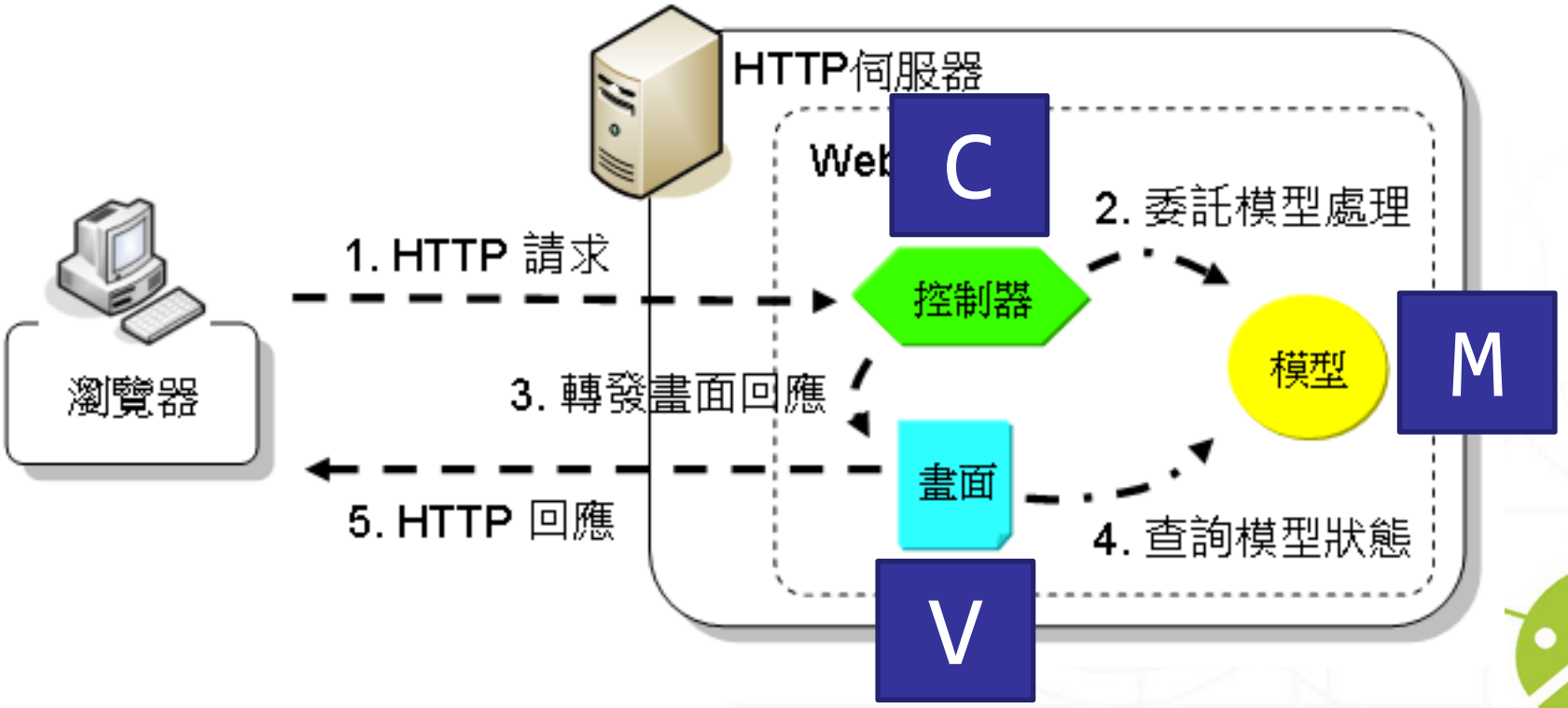
- 於HTML檔中嵌入Java程式碼仍然容易受影響與被混淆

```
<%--
Simple Hello World JSP example
--%>
<%@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>
charset=UTF-8">
    </head>
    <body>
<h1><%= "Hello World! I'm a JSP, it is " + new java.util.Date() %></h1>
</body>
</html>
```

MVC 簡介

- 模型(Model)
  - 指的是需要被解決的重要問題的計算模型
- 顯示(View)
  - 負責將模型計算的結果呈現給使用者
- 控制器(Controller)
  - 負責取得輸入資訊做前置處理

MVC

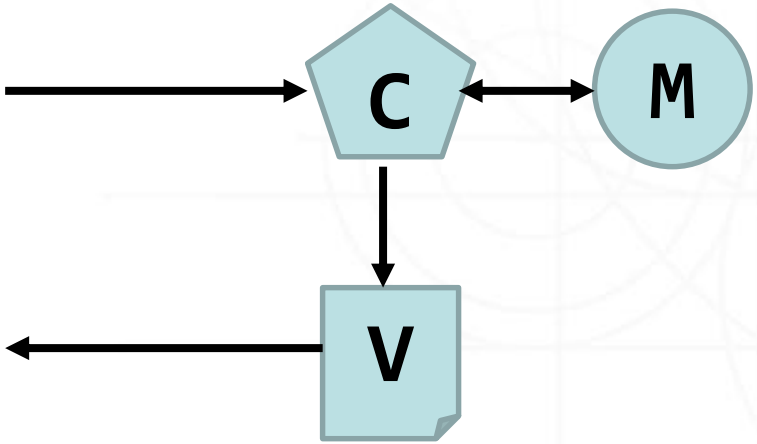

HTTP伺服器
Web
C
2. 委託模型處理
1. HTTP 請求
控制器
模型
M
瀏覽器
3. 轉發畫面回應
畫面
5. HTTP 回應
4. 查詢模型狀態
V

MVC 簡介

- 結合模型、顯示、控制器元件，實作可運行的MVC解決方案
  - 使用POJO實作模型(Model)設計元素
  - 使用JSP/HTML實作顯示(View)設計元素
  - 使用Servlet實作控制器(Controller)設計元素

如何實作 MVC

- 如何實作控制器元件?
  - 如何實作模型元件?
  - 如何實作顯示元件?



控制器任務分派

- 控制器:
  1. Web容器的連結/進入點
  2. 分析參數資料
  3. 連接模型(Model)與顯示(View)
     - 透過Java的方法呼叫將請求資料傳遞給模型
     - 轉送給顯示(JSP/HTML)
       - » RequestDispatcher rd = request.getRequestDispatcher("view.jsp"); rd.forward(request, response);

由 Servlet傳送資料到 JSP

- Servlet 端
  - req.setAttribute("x", 100);
- JSP 端
  - JSP-JavaCode
    - request.getAttribute("x");
  - JSP-EL
    - ${requestScope.x}

由 Servlet傳送資料到 JSP

- Servlet 端
  - User user = new User();
    user.setName("Mary");
    req.setAttribute("user", user);
- JSP 端
  - JSP-JavaCode
    - User user = (User)request.getAttribute("user");
      out.print(user.getName());
  - JSP-EL
    - ${requestScope.user.name}

```
String username = req.getParameter(“username”);
User user = new User(username);
boolean check = user.isMember();
req.setAttribute(“check”, check);
req.setAttribute(“username”, username);
RequestDispatcher rd = request.getRequestDispatcher("view.jsp");
rd.forward(request, response);
```

C

```
public class User {
    private String username;
    public User(String username) {
        this.username = username;
    }
    public Boolean isMember() {
        // …驗證此人是否是會員？
        // return chack;
    }
}
```

M

```
<%@ page …%>
<%
    String username = (String)request.getParameter(“username”);
    boolean check = (Boolean)request.getParameter(“check”);
%>
<HTML>
    <Head> … </Head>
    <Body>
        …
        <%=username %> 是否是會員？ <%=check %>
    </Body>
</HTML>
```

V

MVC 範例程式

BMI MVC

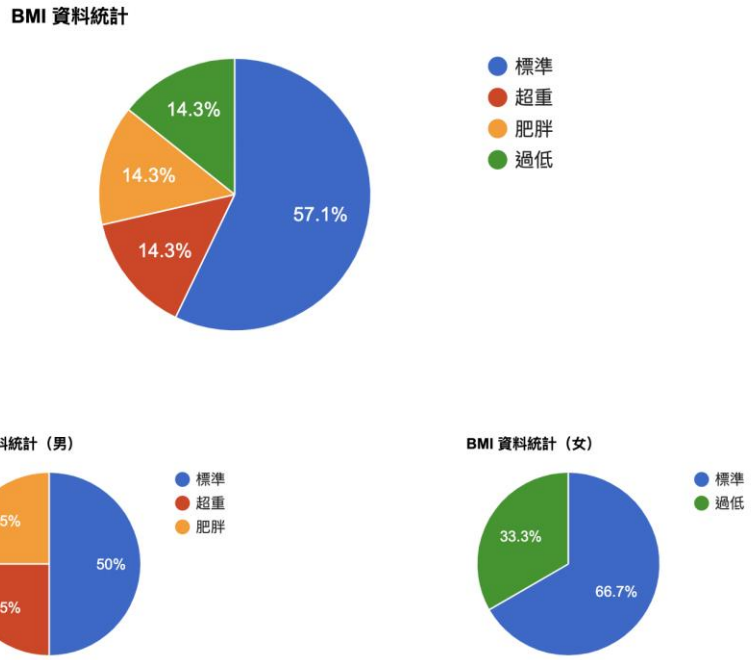
性別： ○男 ○女

請輸入身高

請輸入體重

save reset

id	sex	height	weight	bmi	delete
1	男	170.0	60.0	20.76	刪除
2	女	162.0	45.5	17.34	刪除
3	女	168.5	63.0	22.19	刪除
4	男	177.0	85.0	27.13	刪除
5	男	180.5	92.5	28.39	刪除
6	女	150.0	45.0	20	刪除



一個設計的問題

- 顯示是否應該直接從控制器中取得請求資料，或由模型中間接取得
- 良好的OO設計會分離不相關的事物，使得這 些事物改變時彼此不受影響
- 直接從顯示中存取請求資料，造成元件間無謂的關連

總結

- 使用Servlet實作控制器設計元件
- 使用POJO實作模型設計元件
- 使用JSP與表示式語言實作顯示設計元件
- 結合模型、顯示、控制器元件來實作可運行 的MVC解決方案


JUnit
JS
ORACLE
FUSION MIDDLEWARE
WEBLOGIC SERVER
JBoss
by Red Hat
HIBERNATE
HTML
ORACLE
SPATIAL
PostGIS
SoapUI
by SMARTBEAR
spring
by Pivotal
GeoServer
CSS
eclipse link
OpenLayers
SUBVERSION
jQuery
write less, do more.
PRIME FACES
Java™
Microsoft
SQL Server
mongoDB
NetBeans
maven
git
MySQL
ORACLE
DATABASE
ANGULARJS
by Google
LIFERAY
eclipse
PostgreSQL
sonarqube
Visual Studio