巨匠線上真人

Java Web OCE JWCD元件系統開發認證

巨匠線上真人

Java Web OCE JWCD元件系統開發認證

第四堂 Servlet組態與網站系統資源控制

本堂教學重點

- Servlet 生命週期與初始化組態配置
- Servlet存取網站系統資源ServletContext應用
- ServletContext系統生命週期探討與實作
- 請求配送(Dispatcher)的運行流程與狀態持續架構

Servlet 生命週期與初始化組態配置

- Servlet產生一個Instance進行初始化參數注入
  - 如同建構一個物件的建構子初始化一般。規劃Servlet時，必須保留空參數架構子(預設架構子)，建構不能進行注入資訊，又該如何進行初始化設定?
  - Servlet產生Instance(個體物件)，是經由Web Container設定隨網站系統啟動產生，或配合前端的第一次請求。因此需要保留空參數架構子。
- 我們可以使用ServletConfig介面與Servlet init()方法，操作初始化參數內容。
- 使用Servlet佈署web.xml，或者 @WebServlet Annotation進行參數初始化。
- Servlet container 會引發Servlet init(ServletConfig config) Method， 並將組態 ServletConfig 物件注入。

ServletConfig 介面說明

- ServletConfig 介面所提供的 4 個方法。
- ServletConfig 僅提供 getInitParameter() 方法(唯獨屬性)，只能取出參數內容。

方法	說明
String getInitParameter(String name)	取得指定的參數名的內容
Enumeration getinitParameterNames()	取得所有參數名稱(回應一個列舉)
ServletContext getServletContext()	參照出 ServletContext 物件(應用系統物件)
String getServletName()	取得 Servlet Instance 的名稱

ServletConfig 介面應用-Servlet參數設計

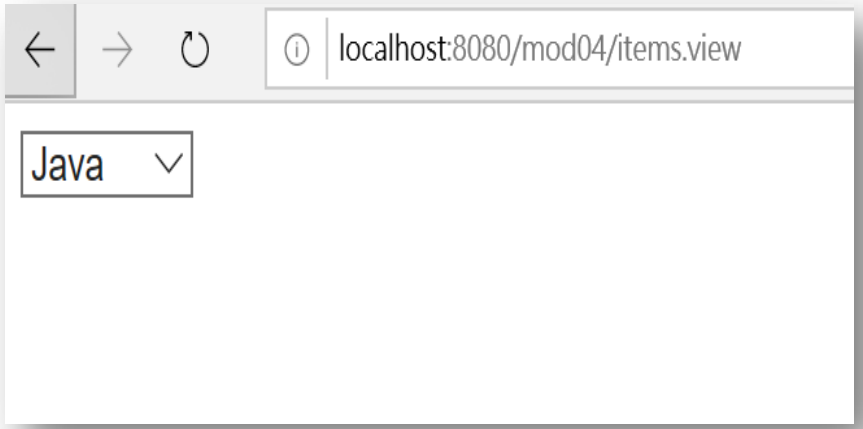
ItemsServlet.java

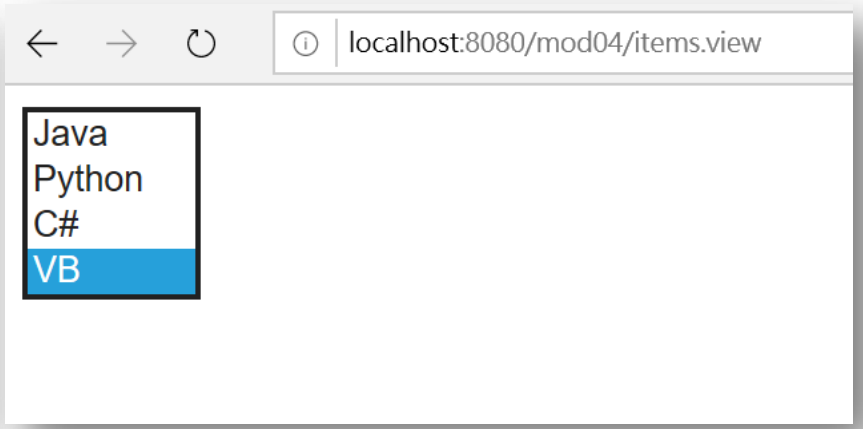
```java
public class ItemsServlet extends HttpServlet {
    private String[] items;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //設定回應編碼
        response.setContentType("text/html;charset=UTF-8");
        //參考出PrintWriter
        PrintWriter out=response.getWriter();
        out.println("<select>");
        //走訪初始化參數轉換的字串陣列
        for(String item : items) {
            out.println(String.format("<option value='%s'>%s</option>",item,item));
        }
        out.println("</select>");
    }

    @Override
    public void init(ServletConfig config) throws ServletException {
        super.init(config);
        //使用ServletConfig獲取初始化參數設定內容
        items=config.getInitParameter("items").split(",");
    }
```

初始化參數名稱

```xml
<!-- 佈署具有初始化參數的Servlet -->
<servlet>
  <servlet-name>items</servlet-name>
  <servlet-class>com.gjun.view.ItemsServlet</servlet-class>
  <init-param>
      <param-name>items</param-name>
      <param-value>Java,Python,C#,VB</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>items</servlet-name>
  <url-pattern>/items.view</url-pattern>
</servlet-mapping>
```

ServletConfig 範例執行結果





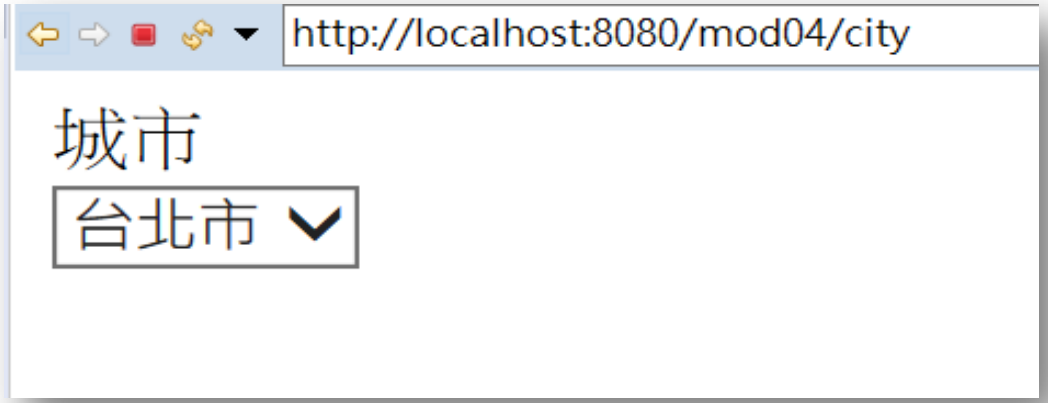
採用@WebServlet Annotation設定初始化參數

- 使用@WebInitParam Annotation
- 使用@WebServlet initParams Options。

```
@WebServlet(name="city",urlPatterns="/city",
initParams=@WebInitParam(name="cities",value="台北市,新北市,桃園市,台中市,台南市,高雄市"))
public class CityServlet extends HttpServlet {
    private String[] cities;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //設定回應編碼
        response.setContentType("text/html;charset=UTF-8");
        //參考出PrintWriter
        PrintWriter out=response.getWriter();
        out.println("<div>城市</div>");
        out.println("<select>");
        //走訪初始化參數轉換的字串陣列
        for(String item : cities) {
            out.println(String.format("<option value='%s'>%s</option>",item,item));
        }
        out.println("</select>");
    }
```

@WebInitParam annotation

採用@WebServlet設定初始化參數執行結果



Servlet存取網站系統資源-ServletContext應用

- ServletContext介面為存取Web Application(網站應用系統)資源的一個介面。
- javax.servlet.ServletContext
  - ServletContext Interface 定義了與Web Container 溝通的方法，藉此獲取網站應用系統的環境資訊。
  - 每一個網站應用系統都一個，且只有一個(just one/Only One) ServletContext物件對應。
- 如何參照出ServletContext
  - 透過GenericServlet.getServletContext()
  - ServletRequest.getServletContext()進行網站應用系統界接物件參考。

ServletContext Interface
提供方法進行參照物件的操作(應用系統範圍)

方法	說明
Object getAttribute(String name)	參照出特定屬性名稱對應的物件(Object)。
Enumeration getAttributeNames()	取出應用系統範圍參照的所有物件的屬性名稱。
void setAttribute(String name, Object value)	新增或修改特定屬性參照的物件內容。

ServletContext 介面實作-資源存取應用

- 使用getRealPath()方法，將虛擬目錄對應實際目錄，用來進行檔案的讀取。
- 使用getResourceAsStream(String Path)，獲取指定的網站目錄下的資源。
- 讀取網站目錄下的檔案，進行下載Demo。

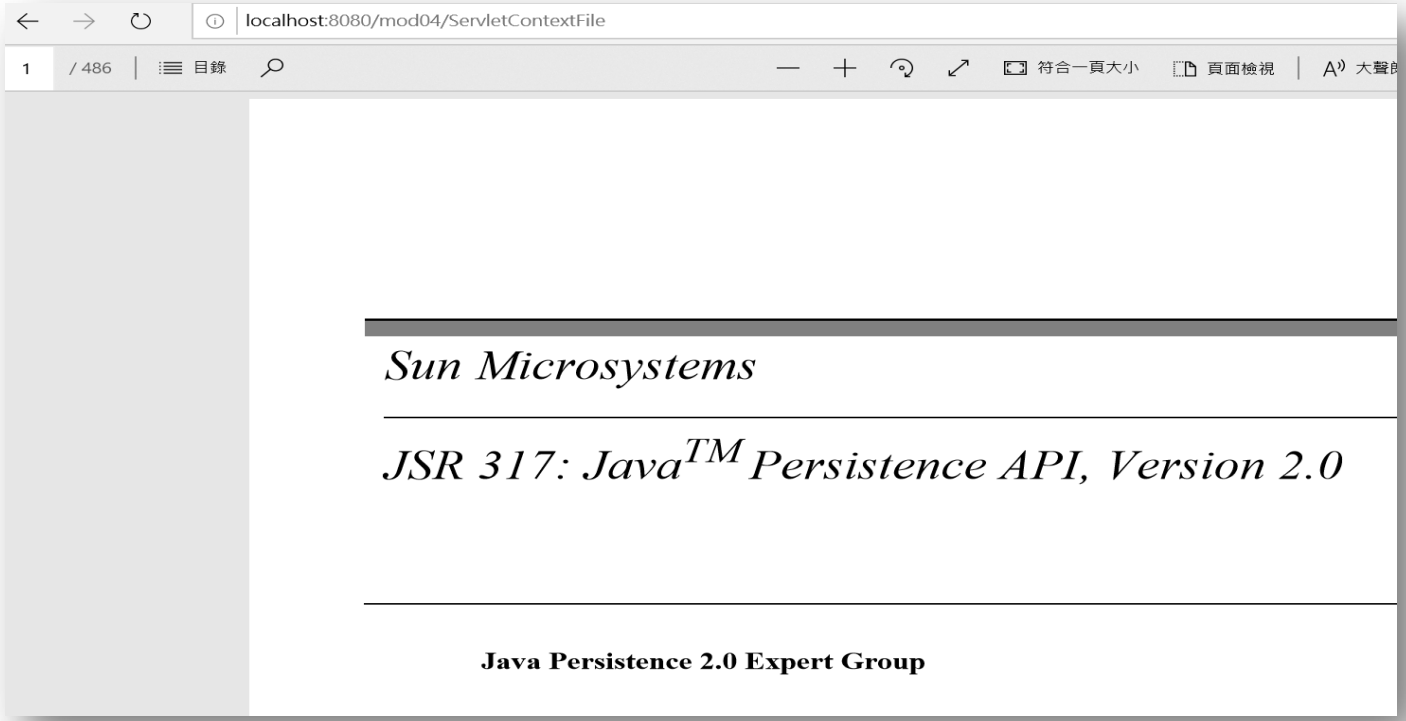
ServletContextFileServlet.java

```
@WebServlet("/ServletContextFile")
public class ServletContextFileServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //設定回應的Content Type
        response.setContentType("application/pdf");
        //參照出ServletContext物件
        ServletContext application=this.getServletContext();
        //直接指定虛擬目錄下檔案開啟讀取串流物件
        InputStream is=application.getResourceAsStream("/files/Persistence JPA.pdf");
        //建構緩存取
        byte[] buffer=new byte[is.available()];
        //讀取檔案
        is.read(buffer, 0, buffer.length);
        //透過Response OutputStream將檔案下載至前端
        ServletOutputStream out=response.getOutputStream();
        out.write(buffer, 0, buffer.length);
        out.flush();
        is.close();

    }
```

透過GenericServlet getServletContext()參照出ServletContext物件

透過getResourceAsStream()指向網站檔案進行開啟

ServletContext 介面實作結果



PDF檔案讀取

ServletContext 網站應用系統生命週期探討與實作

- ServletContext生命週期，代表對應一個應用系統的生命週期。
  - 生與死兩個重要的階段相關事件程序引發。
- 使用ServletContextListener介面，可 設計聆聽特定的網站系統啟動時，進行初始化或者停止服務引發的事件程序。
- ServletContextListener介面提供的5重要的Method(事件)
  - **contextInitialized**(ServletContextEvent sce)-聆聽網站應用系統啟動時進行應用系統初始化設定。
  - **contextDestroyed**(ServletContextEvent sce)-聆聽網站應用系統停止時。引發的回收資源等程序。

ServletContext生命週期聆聽(Listener)示範

- 網站應用系統啟動之後，設定網站應用系統共用資源。
- 網站系統停止服務時，寫出停止服務Log File，或整回收系統資源。
- 設計實作ServletContextListener介面的類別。
- 實作聆聽網站系統啟動與停止服務時引發的事件程序。
- 部署實作ServletContextListener介面的類別。

web.xml 應用系統初始化參數設定

```
<!-- 網站系統初始化參數 -->
<context-param>
   <param-name>driverClassName</param-name>
<param-value>com.mysql.cj.jdbc.Driver</param-value>
</context-param>
<context-param>
   <param-name>url</param-name>
   <param-value>jdbc:mysql://localhost:3306/test?useUnicode=true&amp;characterEncoding=utf8&amp;u
</context-param>
<context-param>
   <param-name>username</param-name>
   <param-value>root</param-value>
</context-param>
<context-param>
   <param-name>password</param-name>
   <param-value>1111</param-value>
</context-param>
<!-- 佈署具有初始化參數的Servlet -->
```

實作ServletContextListener初始化參數

- 設計一個封裝網站系統啟動時，初始化參數資訊與進行封裝資訊的Entity Class。
- 實作介面ServletContextListener進行應用系統初始化與停止服務引發的事件程序類別。

```
public class ApplicationHandler implements ServletContextListener{
```

```
@Override
public void contextInitialized(ServletContextEvent sce) {
    ServletContextListener.super.contextInitialized(sce);

    //取出介接聆聽的應用系統介面ServletContext
    ServletContext application=sce.getServletContext();
    //取出初始化參數
    String url=application.getInitParameter("url");
    String driverClassName=application.getInitParameter("driverClassName");
    String username=application.getInitParameter("username");
    String password=application.getInitParameter("password");
    //封裝成Entity物件 納入應用系統狀態管理
    DBConfig config=new DBConfig();
    config.setUrl(url);
    config.setDriverClassName(driverClassName);
    config.setUsername(username);
    config.setPassword(password);
    //應用系統生命週期參考物件設定
    application.setAttribute("dbconfig", config);

}
```

DBConfig.java

部署ServletContextListener

- 採用web.xml佈署ServletContextListener
- 透過<listener><listener-class></listener-class></listener>元素進行佈署。

```
<!-- 部署Listener -->
<listener>
  <listener-class>com.gjun.listener.ApplicationHandler</listener-class>
</listener>
```

ServletContextListener Demo

- 撰寫一個JSP網頁，動態參照出網站應用系統參照下的DBConfig物件 。
- 驗證ServletContextListener介面實作的結果(網站應用系統共用的資源設定)

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
import="com.gjun.entity.*"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<%=((DBConfig)application.getAttribute("dbconfig")).getDriverClassName()%>
</body>
</html>
```

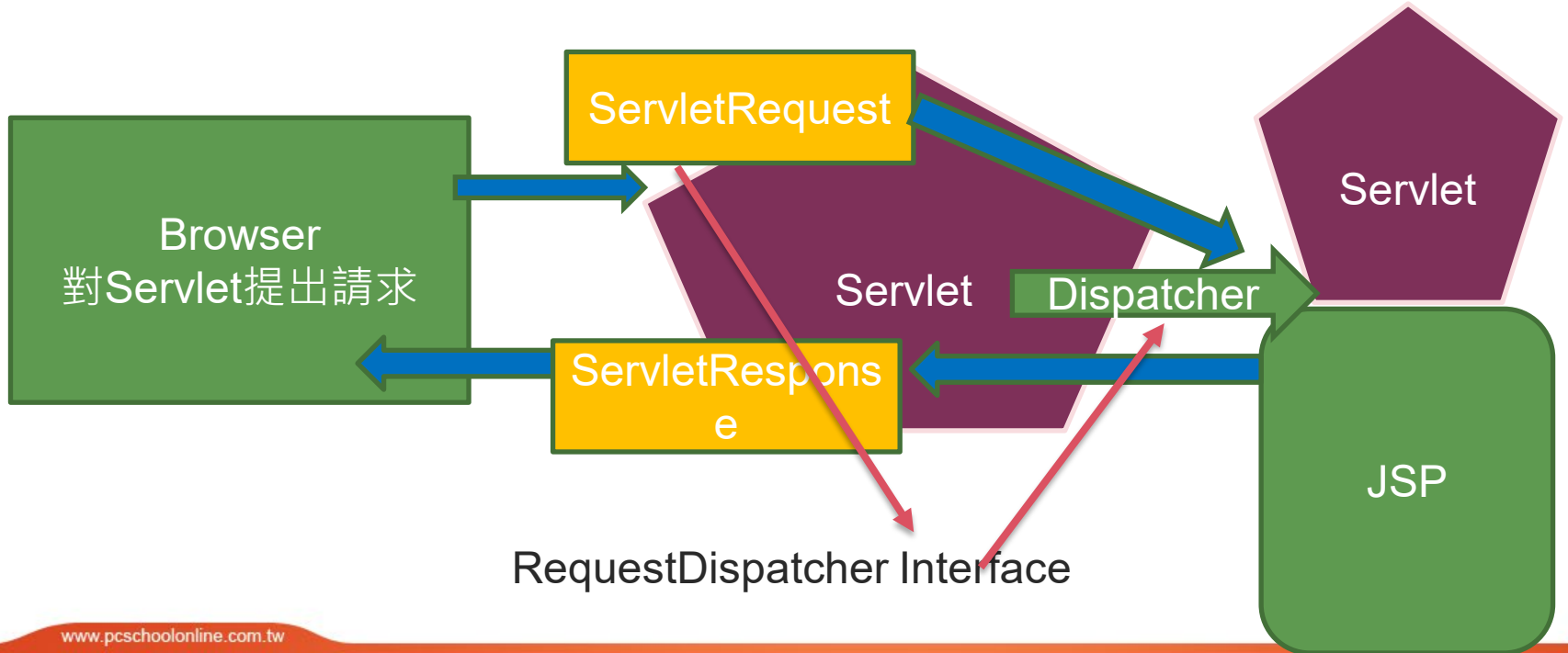
採用@WebListener進行配置

- 可以使用@WebListener Annotation進行ServletContextListener, ServletContextAttributeListener, ServletRequestListener, ServletRequestAttributeListener, HttpSessionListener, or HttpSessionAttributeListener, or HttpSessionIdListener佈署。

```
@WebListener
public class MyListener implements ServletContextListener {
        @Override
        public void contextInitialized(ServletContextEvent sce) {
```

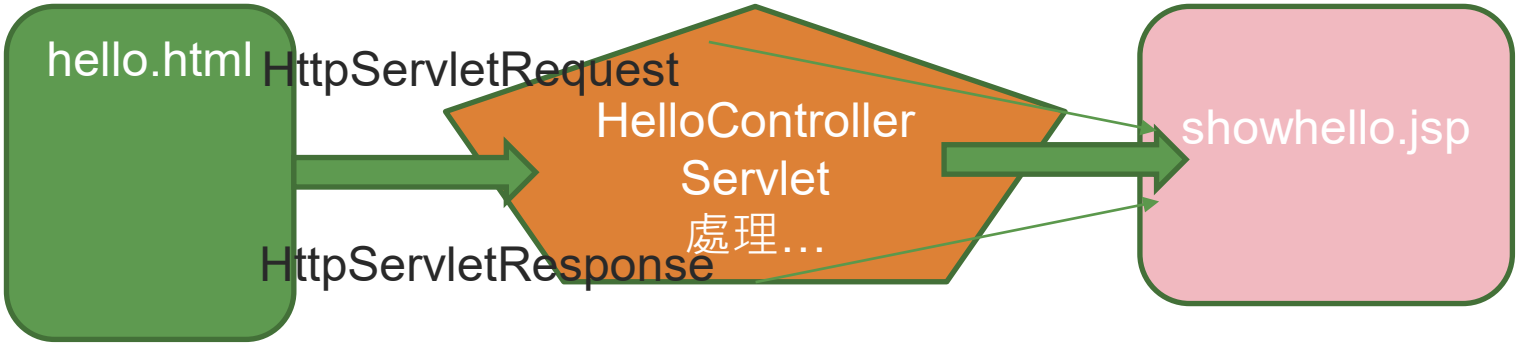
請求配送(Dispatcher)的運行流程與狀態持續架構

- 協調/分派 Servlet中的ServletRequest與ServletResponse生命週期



RequestDispatcher介面應用

- 在 Servlet 中我們可以利用 RequestDispatcher 介面，將目前所接收到的ServletRequest與ServletResponse兩個物件，分派到另一個所指定的資源(JSP 或 servlet 等…)中，用以持續ServletRequest與ServletResponse兩個物件生命週期。
- 首先我們可以可利用 ServletRequest.getRequestDispatcher() 來參考出RequestDispatcher 物件，再透過RequestDispatcher進行forward()或者是include()



表單頁面處理後的訊息持續到另一個showhello.jsp呈現

Form表單頁面設計

- <form>採用Http Request Method為POST。
- <form action="">指向Servlet url-pattern Name。
- 表單欄位(Form Field)設定name attribute，進行資料傳遞的參照用。

hello.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>打招呼</title>
</head>
<body>
    <form method="post" action="HelloController">
        <div>您的訊息</div>
        <input type="text" name="message"/>
        <input type="submit" value="傳送"/>
    </form>
</body>
</html>
```

處理Form請求資訊的處理Servlet

- 透過ServletRequest.getRequestDispatcher()參照出RequestDispatcher物件。
- 透過ServletRequest.setAttribute()參照處理好的狀態(State)，持續這些狀態到目標去。
- 使用RequestDispatcher.forward() Method分派到目標，同時延續原先的ServletRequest與ServletResponse物件生命週期。

HelloController.java

```java
@WebServlet("/HelloController")
public class HelloController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //擷取編碼設定
        request.setCharacterEncoding("UTF-8");
        //擷取表單欄位內容 打招呼訊息
        String message=request.getParameter("message");
        //處理
        String result=String.format("您說:%s 謝謝您的祝福!!!",message);
        //透過Request參考出RequestDispatcher介面
        RequestDispatcher dispatcher=request.getRequestDispatcher("showhello.jsp");
        //透過ServletRequest參照處理結過
        request.setAttribute("result",result);
        //分派到目標去
        dispatcher.forward(request, response);
        //後面不要撰寫了 因為採用forward控制權已交付到目標去
        return;
```

呈現派送進來的狀態JSP

- 使用EL運算語言${} 指令，取出ServletRequest.setAttribute()的屬性名稱參照的物件。並且進行內容呈現(Render UI)。

showhello.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>處理結果呈現</title>
</head>
<body>
${requestScope.result}
</body>
</html>
```

Lab

- 如何設計一個Servlet具有初始化參數設定與應用
- 如何採用@WebServlet進行初始化參數設定
- 設計存取網站資源的ServletContext應用
- 如何在一個網站應用系統啟動時，進行聆聽引發事件程序，配置一個應用系統共用的組態物件。