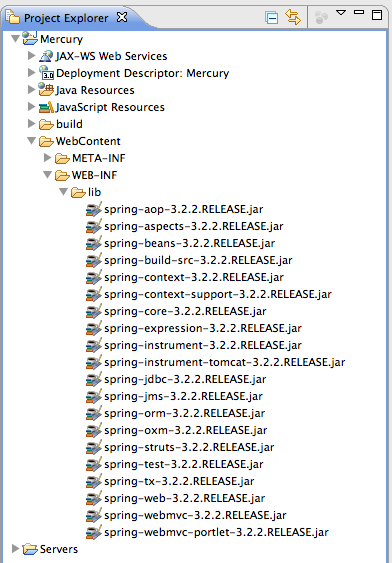
**Spring MVC 3 實作教學 (1) - 架構簡介與專案建立 ( Start Spring MVC 3 tutorial )**

因 Spring Framework 越來越完整，看了 Spring 官方介紹發現利用 Spring MVC 來實作 Web Application 實在是很方便。故之前花了些時間稍微瞭解，在此將實作過程紀錄下來。

提醒 : 本系列教學使用 Eclipse 與 Tomcat ，檔案編碼為 UTF-8。  
首先到 Spring [官方下載位置](http://www.springsource.org/download/community)取得專案建立所需相關 Library後，建立一個新的專案，滑鼠右鍵 新增 > 動態Web專案 ( Dynamic Web Project )，專案建立完成之後將 Spring Framework 的 Library 加入，如下圖 :

[](http://4.bp.blogspot.com/-2PzRl-qfg60/UdZqEnuuSeI/AAAAAAAAAiM/66tkgCYS8pM/s563/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%882.35.26.png)

雖然單純的 Spring MVC 實作不需要使用到全部的 jar 檔，但之後的文章會慢慢提到其他 Spring Framework 的功能，故在此先全部加入了。  
  
除了 Spring Framework 以外，我們還需要以下幾個 Library：  
(1.) [Common Logging](http://commons.apache.org/proper/commons-logging/download_logging.cgi" \t "_blank)   
(2.) [JSTL (搜尋 JSTL 並選擇最新)](http://search.maven.org/)  
(3.) [Jackson ( 取得 core、mapper )](http://jackson.codehaus.org/)  
(4.) [SiteMesh](http://wiki.sitemesh.org/display/sitemesh/Download)  
(5.) [log4j](http://logging.apache.org/log4j/1.2/)、[slf4j](http://www.slf4j.org/" \t "_blank)  
(6.) [H2 Data Base ( 本系列文章將會以H2 Data Base 作為範例 )](http://www.h2database.com/html/main.html)  
(7.) [Hibernate](http://www.hibernate.org/downloads)  
  
Spring MVC 顧名思義脫離不了 MVC 特點，當使用者發出 Request 時透過 Spring 提供的 Library 將 Request 導向我們的 Controller，而 Controller 將需求處理完後將執行結果( Data Model ) 透過指定的 View 將結果呈現給使用者。  
  
看完上面的敘述你不禁會懷疑，那我們如何設定、得知 Request 進來要導向哪個 Controller ? 或者 Controller 執行完的結果資料，這些資料呈現方式有可能是頁面或為 JSON 格式物件，到底該怎麼處理？這些問題將會在後面的文章得到答案。

**Spring MVC 3 實作教學 (2) - Hello Spring MVC tutorial !**

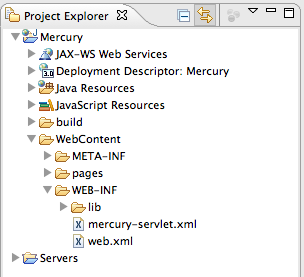
延續實作教學(1) 建立好專案後，如何讓我的 Spring MVC Project 動起來？在開始動手之前，可以先閱讀官方 [Spring MVC 教學文章](http://blog.springsource.com/2011/01/04/green-beans-getting-started-with-spring-mvc/)，透過清楚的 Spring MVC 流程圖可以瞭解整個運作的情況。接著開始實作吧！

**Config :**

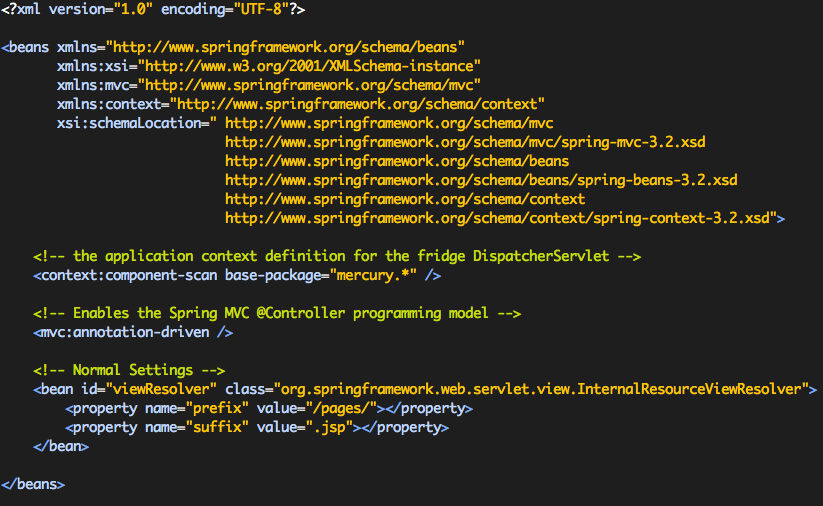
若詳細看過教學文章可以知道 Spring MVC 是透過名為 DispatcherServlet 的程式來將使用者的 Request 導向我們的 Controller ( 邏輯處理程式 )。而為了讓 Java EE Container 啓動時知道我們將會使用這個  DispatcherServlet  所以我們必須先在設定檔宣告，該設定檔即為   /WEB-INF/web.xml，設定如下：

[](http://3.bp.blogspot.com/-qoEz-Q6tmXE/UdbJ4Hb7K4I/AAAAAAAAAjI/lALLaRkBhfA/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%889.27.37.png)

Servlet :  
可以看到上述 web.xml 中定義了 Servlet 名稱 (別名) 為 mercury ，實作 Class 為剛剛提到的 DispatcherServlet 。而 load-on-startup 指定為 1 ，可以解讀為當你在 web.xml 裡面定義很多個 Servlet 的情況下， Jave EE Container 啓動時載入這些 Servlet 的順序。若不指定的話，它就會隨機選擇載入時間點。  
  
Servlet-mapping :  
這邊指定了剛剛定義的 Servlet mercury 來實作 mapping 動作，url-pattern 則是定義 Servlet mercury 何時該介入處理使用者的請求。例如：當使用者發出 /account/CalculateMoney 的 Request ，因符合我們定義 "/" 路徑底下，此時 Servlet mercury 就會介入執行 mapping 動作。若還不瞭解的話，於後面實作程式時應該可以有更進一步的認知。  
  
  
設定 web.xml 只是讓 Java EE Container 認識 Servlet，針對 Spring 功能還有許多參數需要去宣告，因此我們還需要一個 Spring 核心功能設定檔，這個檔案預設讀取位置是在 /WEB-INF 底下，設定檔預設名稱為  <servletname>-context.xml ，官方教學雖然是這樣寫但我實際測試名稱應該是 <servletname>-servlet.xml。若你沒新增設定檔或設定檔名稱錯誤，於啓動專案時會出現找不到該設定檔的錯誤訊息，因此到時後可以檢查錯誤訊息裡面顯示的設定檔名稱。另外要指定設定檔於特殊路徑底下，可以在 web.xml 裡面去設定 (請參考 [Spring MVC 教學文章](http://blog.springsource.com/2011/01/04/green-beans-getting-started-with-spring-mvc/)裡面的 web.xml )。  
範例如下圖：

[](http://2.bp.blogspot.com/-4scNgi-xvOM/UdbXnvoVg6I/AAAAAAAAAjo/d-TYKEfOuxY/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8810.26.16.png)

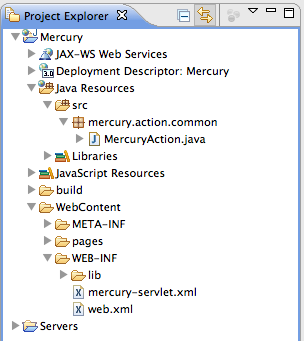
Spring 核心功能設定檔 ( /WEB-INF/mercury-servlet.xml )內容如下：

[](http://4.bp.blogspot.com/-rO-tFLdx5xY/UdbY5BHkvqI/AAAAAAAAAj8/-QEe2XKVTRw/s1600/%25E8%259E%25A2%25E5%25B9%2595%25E5%25BF%25AB%25E7%2585%25A7+2013-07-05+%25E4%25B8%258B%25E5%258D%258810.31.38.png)

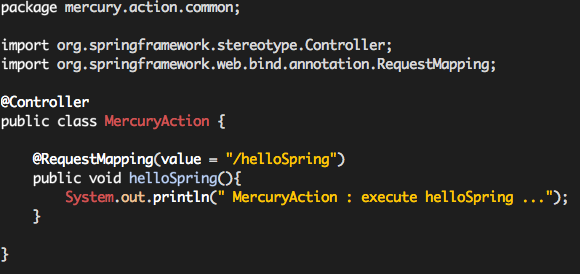
<context:component-scan base-package="mercury.\*" />：  
這邊定義了專案裡特定 package 底下的程式，確保其中若有使用到 Annotation ( 如 @Controller 、@Service... ) 的程式會被找到。範例設定符合 mercury.\* 的 package 內的程式都會被找到。  
  
<mvc:annotation-driven /> :  
此參數為將 Request 會自動導向對應 Controller 的功能開啓、宣告的動作。  
  
ViewResolver :  
這裡定義了當 Controller 執行完畢，若無特定指定呈現 View 的情況下，它會直接將 View 視為 /pages/ 路徑底下某個相對應名稱的 .jsp。 例如：當 Request 為 /Login 時，若無指定呈現的 View，則會自動以 /pages/Login.jsp 來作為預設的 View。當然 View 放置的路徑我自己習慣的模式，若你不喜歡這樣也可以照自己的設定來實作。

**Controller and View :**

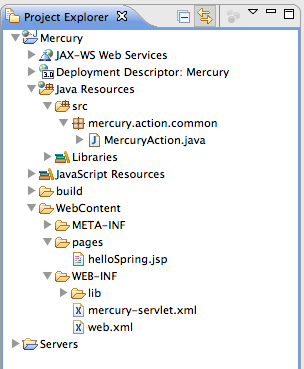
以上將兩個主要的設定檔都設定完後，還需要一個處理 Request 的 Controller 實作程式。因此根據 /WEB-INF/mercury-servlet.xml 裡面宣告的 <context:component-scan base-package="mercury.\*" /> ，我將會建立一個 package ，如下圖：

[](http://4.bp.blogspot.com/-zQFxvUPhBdo/UdbhaTB0C_I/AAAAAAAAAkI/M_oUlMo2sw4/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.08.09.png)

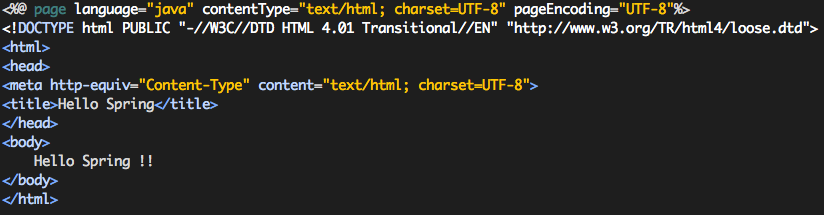
建立好名為 mercury.action.common 的 package，新增一個簡單的 Pojo，其內容如下：

[](http://3.bp.blogspot.com/-7uCsxwP4RKM/UdbjDnuRxYI/AAAAAAAAAkY/THM9CFfEAy0/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.15.05.png)

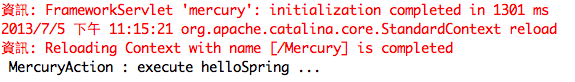
可以看到這支程式非常簡潔，首先使用 Annotation @Controller 將此程式標註為處理 Request 的 Controller，裡面有一個名為 hellSpring 的 Method，而這個 Method 帶有 @RequestMapping( value="/helloSpring" )，其意思為當使用者 Request 為 /helloSpring 時，Request 會對應到 MercuryAction 裡的 helloSpring  Method。而你可以看到 helloSpring 並沒有回傳任何型態，此時該 Method 執行完就會尋找對應的 View 來顯示，意即 /pages/helloSpring.jsp。所以我們還需要新增一個對應  JSP 檔案。此範例完整檔案如下：

[](http://4.bp.blogspot.com/-MT9RjJLSPoo/UdbnTpwicaI/AAAAAAAAAko/F_NSC8GMBqA/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.33.19.png)

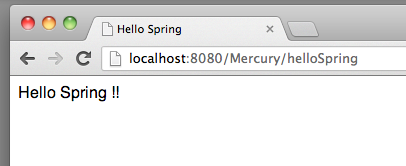
JSP 內容如下：

[](http://2.bp.blogspot.com/-3zuq5bUv36w/UdbpD0Uf6oI/AAAAAAAAAlA/ef-R627QtuI/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.40.17.png)

以上完成所有準備工作。  
當我們輸入 url : http://localhost:8080/Mercury/helloSpring 時，透過 Spring MVC 會將 Request 導向 MercuryAction 並執行 helloSpring 時 Console  就會出現我們印的訊息，

[](http://1.bp.blogspot.com/-qkbEO1KvIs8/UdboSVFIDRI/AAAAAAAAAk4/CNuvYV_GuC8/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.36.35.png)

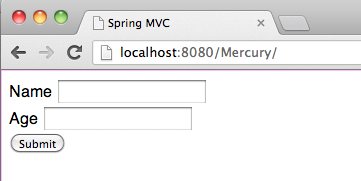
最後將由 /pages/helloSpring.jsp 將結果呈現。

[](http://4.bp.blogspot.com/-ubqiCZogYfQ/Udbqp5A0l5I/AAAAAAAAAlc/8e9pLtGGZsQ/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-05+%E4%B8%8B%E5%8D%8811.47.30.png)

**Spring MVC 3 實作教學 (3) - 基本參數傳遞與處理 ( Get and return parameters )**

本文章會繼續討論 Spring MVC 的基本功能實作，當然之前的文章範例 Hello Spring 內容提到的只是 Spring MVC 的基礎設定，接著我們要慢慢地來實作一個 Modern Web 該有功能。

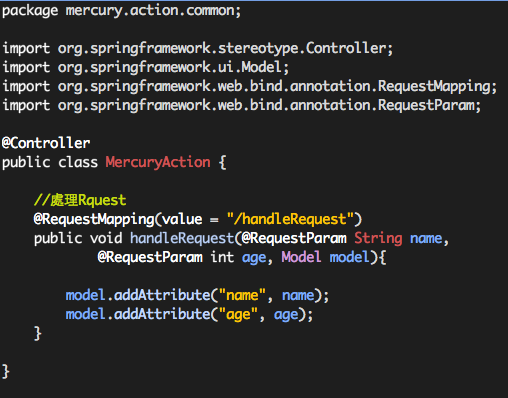
如何在 JSP 與 Controller 之間傳遞參數？  
假設需求為使用者會在畫面填寫資料，填寫完後點擊送出按鈕並導向另一個新的頁面，並在新的頁面將剛剛填入的資料顯示出來。首先來建立填寫資料的畫面，我習慣將頁面都放置於 /WebContent/pages/ 底下，因此我建立了一個頁面 /WebContent/pages/home.jsp，操作畫面看起來如下圖 :

[](http://1.bp.blogspot.com/-RU2Mml6v6Rk/UdpnYmbJkLI/AAAAAAAAAl0/8h9RsqZG964/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%883.15.53.png)

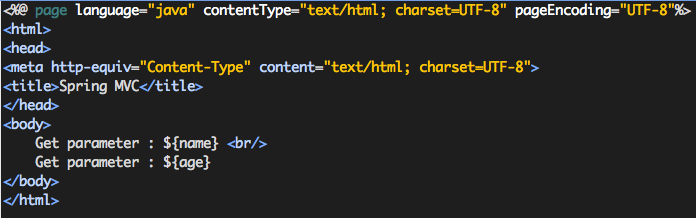
可以看到很簡單的名字與年齡輸入頁面，其頁面程式碼內容如下圖：

[](http://1.bp.blogspot.com/-jvP1kfS2_m0/Udpo4rWu4oI/AAAAAAAAAmE/xZpCHLKWxig/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%883.14.17.png)

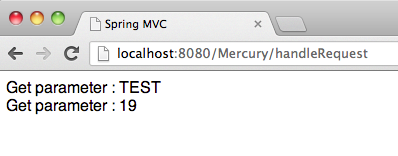
本次範例利用 Form 標籤為主， action 屬性直接指定 Request 的目的地，method 設定為 post。當使用者按下 Submit 按鈕後，Request 透過之前提到的 Dispatcher 將它導向對應的 Controller，且該 Controller 內必須有一個 @RequestMapping (value="/handleRequest") 的 Method，因此其 Controller 內容如下：

[](http://1.bp.blogspot.com/-y22q81yyuDw/UdpuY0bQt-I/AAAAAAAAAmU/yUGIhBGTCfg/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%883.46.22.png)

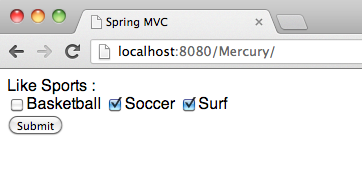
除了@RequestMapping 之外，還可以看到 @RequestParam、Model 這兩個新朋友。  
  
@RequestParam   
即宣告將會傳入的參數型態與名稱，傳入參數名稱與宣告名稱相同就會自動代入參數值，若你宣告了該參數卻並未傳入時會出現錯誤訊息，例如我將年齡欄位從 JSP 拿掉或參數名稱不同，其訊息為 Http Status 400 - Required int parameter 'age' is not present。  
  
Model  
Controller 中 Model 物件你可以簡單的想像成它是個 HashMap，Controller 執行完畢後 Model 物件將如同一個資料包裹被傳遞出去，在此範例中我們將要顯示的參數通通都放進去，如上面的程式碼內容。  
  
別忘了該 Controller 執行完畢，若未指定 View 會自動尋找相對應的 JSP，因此我們還缺少了 View，按照我的習慣建立 /WebContent/pages/handleRequest.jsp，其內容如下：

[](http://4.bp.blogspot.com/-BL0jixCtVLc/Udp7A36I8yI/AAAAAAAAAms/WfQHsbviHtM/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%884.40.03.png)

按照範例填入資料並值執行送出，結果顯示應該如下：

[](http://4.bp.blogspot.com/-ITLB6Rnb6f4/Udp9EntjgrI/AAAAAAAAAm8/e1TwE088U5I/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%884.47.49.png)

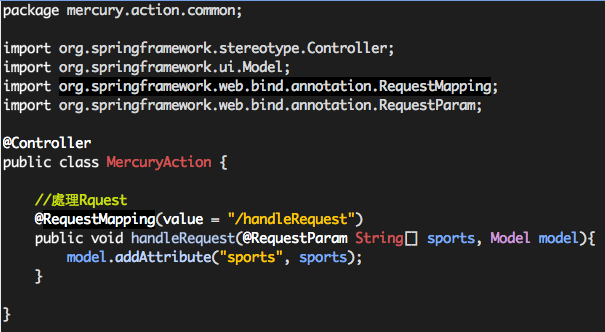
以上是最簡單狀況，但當參數是陣列時該怎麼處理？我另外做了一個簡單的範例，如下：

[](http://3.bp.blogspot.com/-j9sbl6MFb0k/UdqBhWfTWEI/AAAAAAAAAnM/juawQzbHaf4/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%885.07.42.png)

其頁面程式碼內容如下：

[](http://1.bp.blogspot.com/-1izDTLU0-7s/UdqByI666uI/AAAAAAAAAnU/eHDwwka9610/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%885.09.08.png)

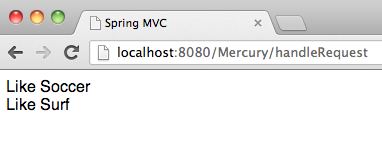
宣告 Controller 傳入參數為字串陣列 String [ ] ，內容如下：

[](http://4.bp.blogspot.com/-ShnV8jkRnUE/UdqCjl1jLMI/AAAAAAAAAnk/m8pMznrhD3Q/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%885.12.01.png)

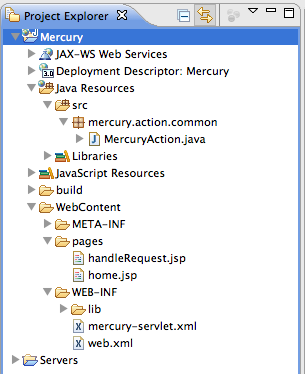
如果回傳資料為陣列且要顯示於畫面中，本次範例是用 JSP taglib 來作處理，因此會需要增加 JSTL 的 Library ( [於此搜尋 JSTL 使用最新版本](http://search.maven.org/) ) 與宣告在 JSP 中，，顯示頁面程式碼內容下：

[](http://3.bp.blogspot.com/-rFm0EYZ32Uo/UdqDXV3PtDI/AAAAAAAAAnw/8e8yxvRiTV4/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%885.12.22.png)

最後顯示結果如下：

[](http://3.bp.blogspot.com/-OWHxjoIcEJU/UdqEA6N1NcI/AAAAAAAAAn4/uw-qzsAnrbA/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-08+%E4%B8%8B%E5%8D%885.18.30.png)

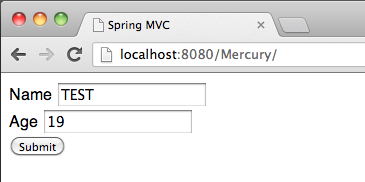
提供一下本範例所有檔案擺放位置：

[](http://1.bp.blogspot.com/-aPRdlw3B3RY/Udt2BgR_5CI/AAAAAAAAAoI/A8euc1QH9Rw/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8A%E5%8D%8810.19.02.png)

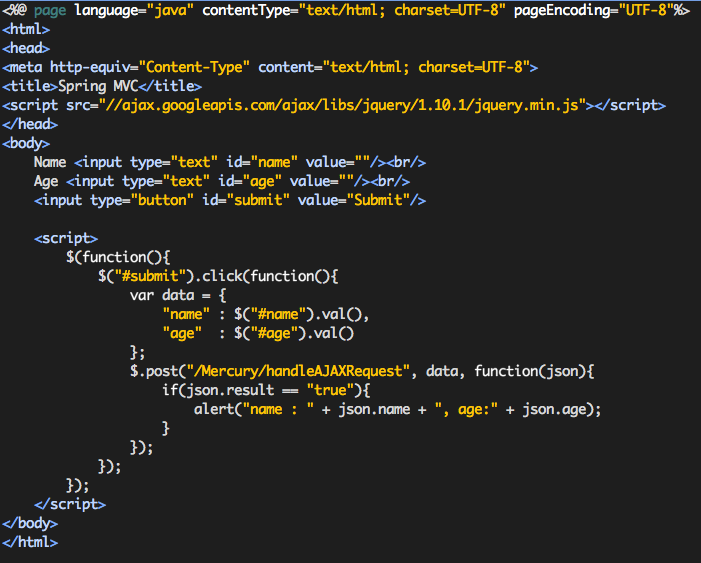
以上為 Spring MVC 基本的參數傳遞範例與介紹。

Spring MVC 3 實作教學 (4) - AJAX 請求處理與回傳 JSON 格式資料 ( Get and return JOSN )

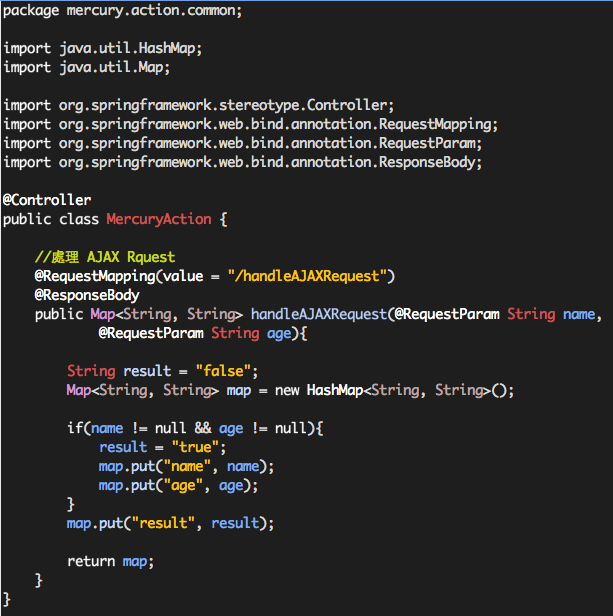
上一篇有討論了基本的參數傳遞與處理，但缺少了 AJAX 範例的應用，因此本篇要繼續討論 Controller 是如何處理 AJAX 請求以及如何回傳 JSON 格式資料。  
  
本篇繼續沿用 Spring MVC 實作教學 (3) 的範例，之前使用 Form 標籤來達成目的，接下來我們只需要將它改寫成使用 AJAX 的方法即可 ( AJAX 部份為使用 JQuery 提供的函式 ) 。一樣模擬使用者填寫資料的畫面 ( /WebContent/pages/home.jsp )，操作畫面如下：

[](http://2.bp.blogspot.com/-4-QaBgy1LRg/UdutS0OsmEI/AAAAAAAAAoc/BHUtYcowz-M/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8B%E5%8D%882.26.48.png)

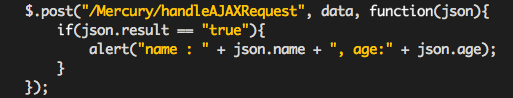
填寫資料頁面程式碼內容如下：

[](http://1.bp.blogspot.com/-L9A6KSFmkF8/UdvAcDuqrLI/AAAAAAAAAp0/0CqBHzdZ2F8/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8B%E5%8D%883.48.38.png)

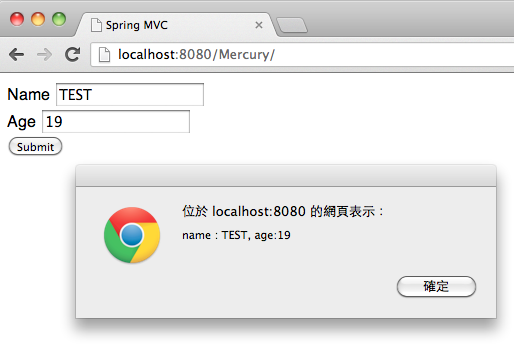
可以看到上述內容 Request 目的地為  /handleAJAXRequest ，傳入參數分別為名稱與年齡兩個欄位， AJAX 部份還包含一個 call back function 判斷回傳資料彈出訊息。故我們的 Controller 需要有一個對應的 Method，Controller 內容如下：

[](http://2.bp.blogspot.com/-ptbrO2vtLJc/UduyRzvLlDI/AAAAAAAAAo0/yWu1xUhWwr0/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8B%E5%8D%882.47.35.png)

比較之前範例 @RequestMapping、@RequestParam 這兩個部份寫法相同，這次多了 @ResponseBody 以及你是否發現這次的 Method 增加了回傳型態？  
  
@ResponseBody  
於此加入這個 @ResponseBody Annotation 意思在於告訴 Spring 該 Method 執行完畢是要將資料回傳到前端，並非回傳 View。故我們 handleAJAXRequest 這個範例回傳的是一個 Map，我們將需要回傳的資料丟進 Map。你當然也可以回傳其他類型的物件或者是你自行定義的物件。  
  
Controller 照上述修改完後若你直接執行程式，很不幸地的會得錯誤訊息：HTTP Status 406 - The resource identified by this request is only capable of generating responses with characteristics not acceptable according to the request "accept" headers.，這是因為你還沒有提供 Spring 處理 JSON 的 Library，Spring MVC 預設處理 JSON 的 Library 版本為 Jackson，所以可以到 [Jackson 官方網站](http://jackson.codehaus.org/)下載最新的檔案並加至 /WEB-INF/lib 底下 ( core 、mapper 都要 )。  
  
提醒：在下載 Jackson 的檔案可以看到它有分為 ASL 與 LGPL 兩個版本，這是關於軟體 License 問題並非程式面的問題，詳細可以參考此[討論](http://stackoverflow.com/questions/5215023/whats-the-difference-between-jackson-core-asl-and-jackson-core-lgpl" \t "_blank)。  
  
上述問題處理完成後，重新執行功能就會正常運作，我們在前面 JSP 部份已經將回傳資料如何取得的部份已經寫完了，如下圖：

[](http://3.bp.blogspot.com/-9jrm8QK3Onc/UdvAocsWIlI/AAAAAAAAAp8/I-RM1P4hT2Y/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8B%E5%8D%883.49.30.png)

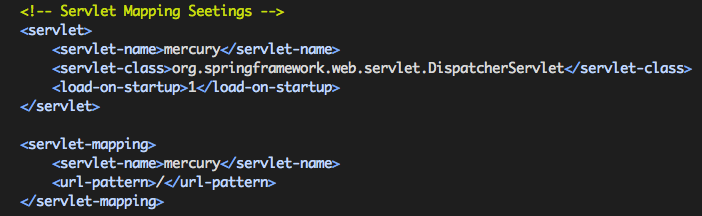
成功執行畫面，如下圖：

[](http://2.bp.blogspot.com/-Y__zhrk2Epo/Udu6zwuhmsI/AAAAAAAAApM/pQVYtXs5CLU/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-09+%E4%B8%8B%E5%8D%883.24.39.png)

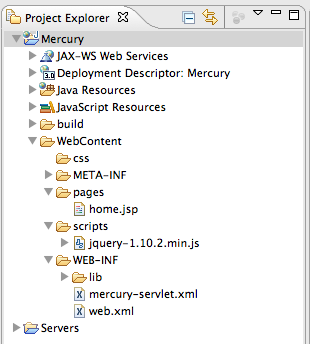
**Spring MVC 3 實作教學 (5) - 取得靜態資源 ( Get static resources )**

相信之前的教學文章已經可以讓你開始建構自己的 Web Project，不過 JSP 頁面總是需要用到 js、css、jpg、png 等等檔案。當你在撰寫頁面時將它們引入，但實際運行時卻發現這些檔案都出現該訊息： Http Status 404 - Not found.

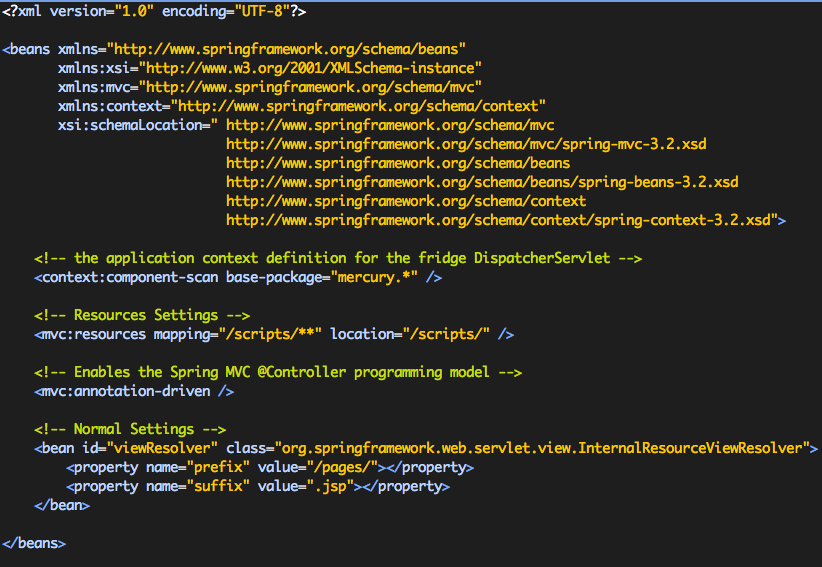
發生上述錯誤訊息是因為 web.xml 裡面的設定，還記我們在一開始設定了 DispatcherServlet 對應的 url-pattern 為 " / "，如下圖：

[](http://1.bp.blogspot.com/-l61zYCC_IG4/Udz75hwjN5I/AAAAAAAAAqM/e7E7zgcBu5I/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-10+%E4%B8%8B%E5%8D%882.14.34.png)

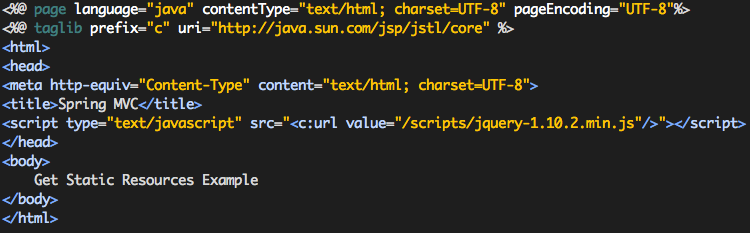
因為我們設定 url-pattern 為 " / "，故連我們在 JSP 裡引進的 js、css 檔案都會符合 " / " 的情況進而透過 DispatcherServlet  將 Request 導向對應的 Controller，在這樣的情況下當然無法找到對應的檔案，因此 Server 端回應 404 錯誤訊息。  
  
在這樣的情況下，首先必須思考我們 url-pattern 是否真的需要設定為 " / "？是否可以縮小 DispatcherServlet 處理 Request 的範圍。  
  
當然 Spring 也有考慮到這個問題，在 3.0.? 版之後增加了取靜態資源的設定方法。我們先將靜態檔案擺放到自行設定的路徑，例如我增加了/WebContent/scripts/jquery-1.10.2.min.js ，參考下圖：

[](http://1.bp.blogspot.com/-RhiSTr6nK7Y/Ud0Dlrp3MpI/AAAAAAAAAqc/OFk75WdY9BE/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-10+%E4%B8%8B%E5%8D%882.47.29.png)

靜態檔案加入後，接著修改我們的 Spring 設定檔 ( 延續之前範例設定檔為 /WebContent/WEB-INF/mercury-servlet.xml )，如下圖：

[](http://4.bp.blogspot.com/-LM9gRGh2Gp8/Ud0EBciedsI/AAAAAAAAAqk/WQ6uEnKSKTo/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-10+%E4%B8%8B%E5%8D%882.49.21.png)

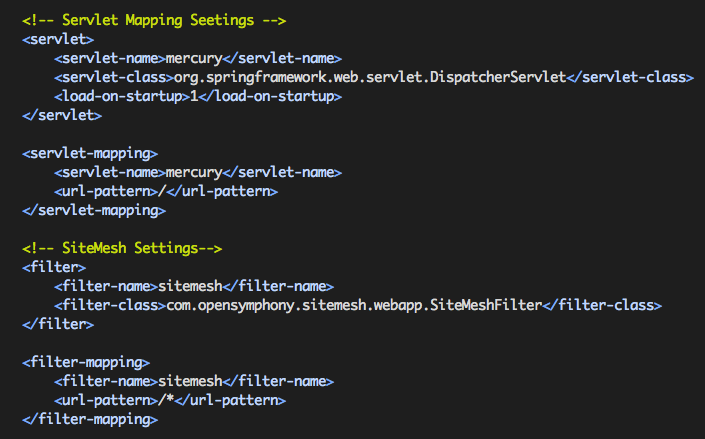
從上面設定檔的內容可以發現，新增加了一個 <mvc:resources /> 標籤，其屬性的意思為該靜態資源對應 ( mapping ) 的路徑 /scripts/\*\*，其靜態檔案實際位置 ( location ) 為 /scripts/，如此一來 Spring 就認得靜態資源的路徑，<mvc:resources />  可以設定多個，因此你若有其他路徑也可以增加該標籤並加入你需要的 mapping 路徑與 location。除修改 Spring 設定檔之外， JSP 引進靜態資源的寫法也需要跟著修改，修改方式如下圖：

[](http://1.bp.blogspot.com/-WG3-QFv65Ns/Ud0HGL6CkkI/AAAAAAAAAq0/33GjvvVfp2g/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-10+%E4%B8%8B%E5%8D%883.02.03.png)

我們在 script src 的屬性部份改用 taglib 取得該靜態資源 <c:url value="" />，如此一來就可以正常的取得我們需要的 js 檔案。

### Spring MVC 3 實作教學 (6) - Spring MVC 與 SiteMesh ( 頁面規劃框架 ) ( Use SiteMesh with Spring MVC )

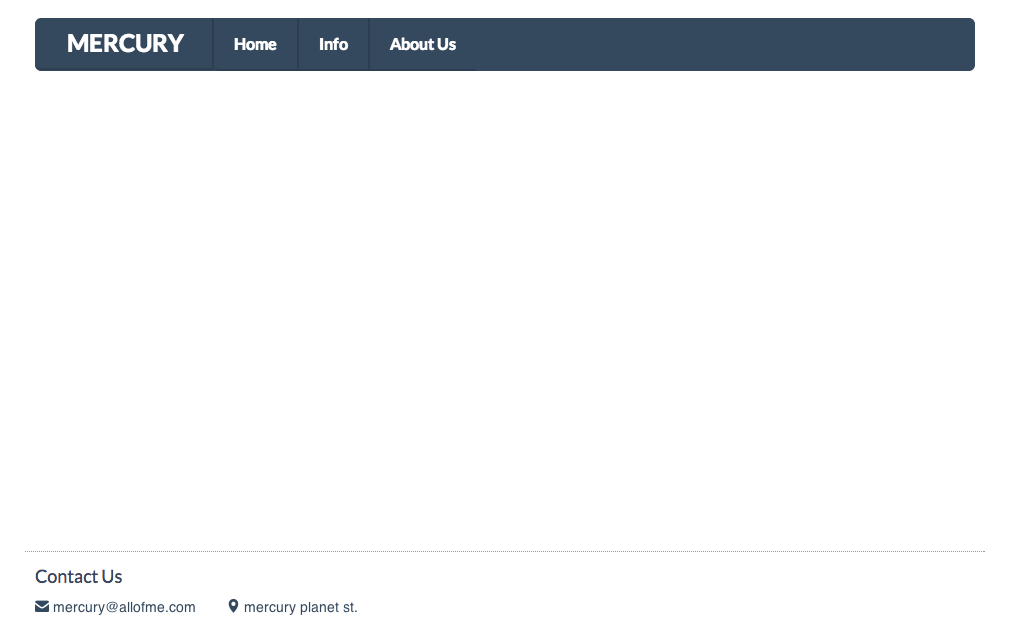
相信在開發 Web Project 時常常會遇到某些頁面內容重複，以往可以使用 JSP include 將會重複的內容引入，但若需要調整頁面版型、風格甚至內容，這些都要重新一一修改，因此基於這個問題衍生出許多頁面規劃的框架，當然今天要介紹的 SiteMesh 也是其中一個。  
  
SiteMesh 只是眾多框架選擇之一，其他的框架當然也可以跟 Spring MVC 作結合。但基於 SiteMesh 方便、簡單所以選擇它當作範例來討論。首先可以先到 [SiteMesh 官方網站](http://wiki.sitemesh.org/display/sitemesh/Download" \t "_blank)下載最新的 Library，一樣將 Library 放置在 /WebContent/WEB-INF/lib 底下。另外還需要設定 /WebContent/WEB-INF/web.xml 檔案，因我們將使用 SiteMesh 所以必須在 web.xml 裡面先宣告，設定如下：

[](http://4.bp.blogspot.com/-6hp3p1viYdM/Ud9tlTgXDWI/AAAAAAAAAsE/R3q5xQaUYnY/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8810.44.21.png)

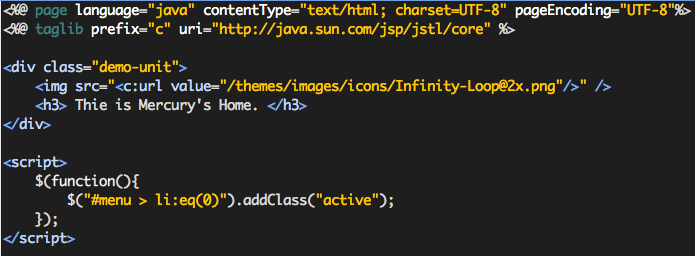
設定檔裡面將實作 filter 的部份定義出來，url-pattern 設定為 " /\* "，意即符合這個路徑底下的頁面都會進行判斷是否需要套用裝飾頁面的動作。  
  
實作之前我們先來假設需求，若畫面上方導覽列 ( Menu ) 與下方其他資訊、聯絡方式 ( Footer ) 是每頁都會出現的固定內容，只有中間部份會根據 Menu 切換而變動，也就是說我們只需要定義一個裝飾 ( 或者思考為通用的版型 ) 頁面，該頁面內容包含 Menu 與 Footer，接下來只要將其他內容頁面套用這個裝飾頁面即可。頁面擺放可以按照自己的習慣，如我的習慣會將所有頁面放置在 /WebContent/pages/ 底下。  
  
首先建立我們的裝飾頁面 /WebContent/pages/layouts/base.jsp，內容如下圖：

[](http://4.bp.blogspot.com/-AqkYwDFA5hk/Ud6CviEjDYI/AAAAAAAAArE/wnD1oPy8x9I/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-11+%E4%B8%8B%E5%8D%886.01.15.png)

檢視 JSP 內容可看到我依照剛剛的設計，本頁包含了 Menu 與 Footer 部份，除此之外 JSP 最上方還加入宣告使用 SiteMesh 的 taglib，另外於頁面中加入 <dec:body / > 標籤，透過標籤標註此處為內容加入處。也就是當其他頁面套用到 base.jsp 時，其頁面 <body></body> 標籤裡所有內容會被加入 <dec:body /> 目前所在位置。內容會變成 base.jsp 的一部份一起呈現。  
  
先來預覽一下腦海中 base.jsp 畫面：

[](http://3.bp.blogspot.com/-BFXuWgh74uA/Ud9pvzeps_I/AAAAAAAAArk/zQm_7Y0t12M/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8810.27.52.png)

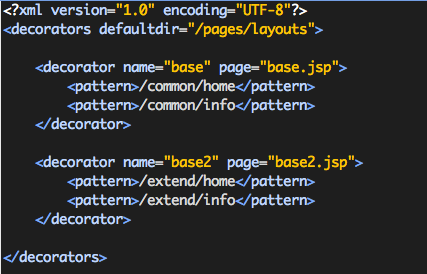
通用裝飾頁面完成後，我們接著來設計內容頁面 /WebContent/pages/common/home.jsp 剛剛說過當內容頁面是一個完整的 HTML 格式 ( 有 head、body 等標籤 )，透過 <dec:body /> 會將 <body></body> 的內容截取放置到 base.jsp 中。但我們也可以只寫 <body></body> 裡面的內容，如下圖就是只有內容部份的範例：

[](http://4.bp.blogspot.com/--CSUpHKjTSg/Ud6JS5FJbBI/AAAAAAAAArU/nXL1rpq5lGE/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-11+%E4%B8%8B%E5%8D%886.30.10.png)

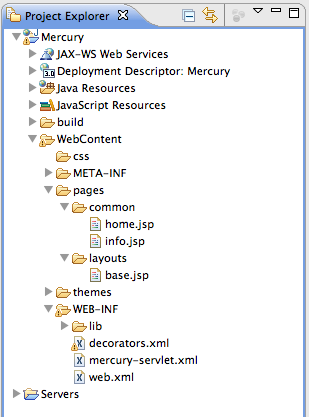
預覽一下腦海中 home.jsp 畫面：

[](http://4.bp.blogspot.com/-y2BxAmCzi8Y/Ud9qym-4TtI/AAAAAAAAAr0/2QUxQEjvgng/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8810.32.29.png)

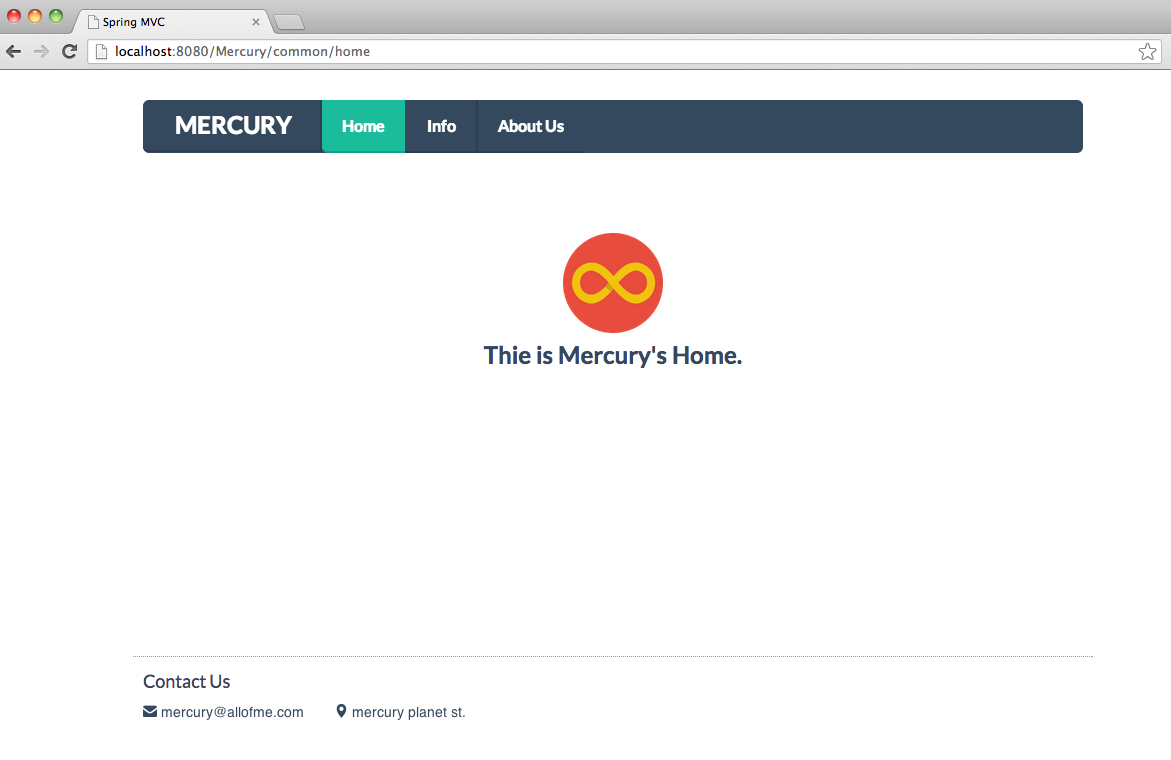
當然此時的 home.jsp 還沒有與 base.jsp 結合，因為我們還未告訴 SiteMesh 哪些頁面須要套用到哪一個裝飾頁面，所以還需要一個重要的 SiteMesh 設定檔，設定檔預設位置於 /WebContent/WEB-INF/ 底下，預設檔名為 decorators.xml。設定內容如下：

[](http://1.bp.blogspot.com/-Lwm8e8ngyXY/Ud9xzsDqDFI/AAAAAAAAAsg/KkYNaNvBoaU/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8811.02.16.png)

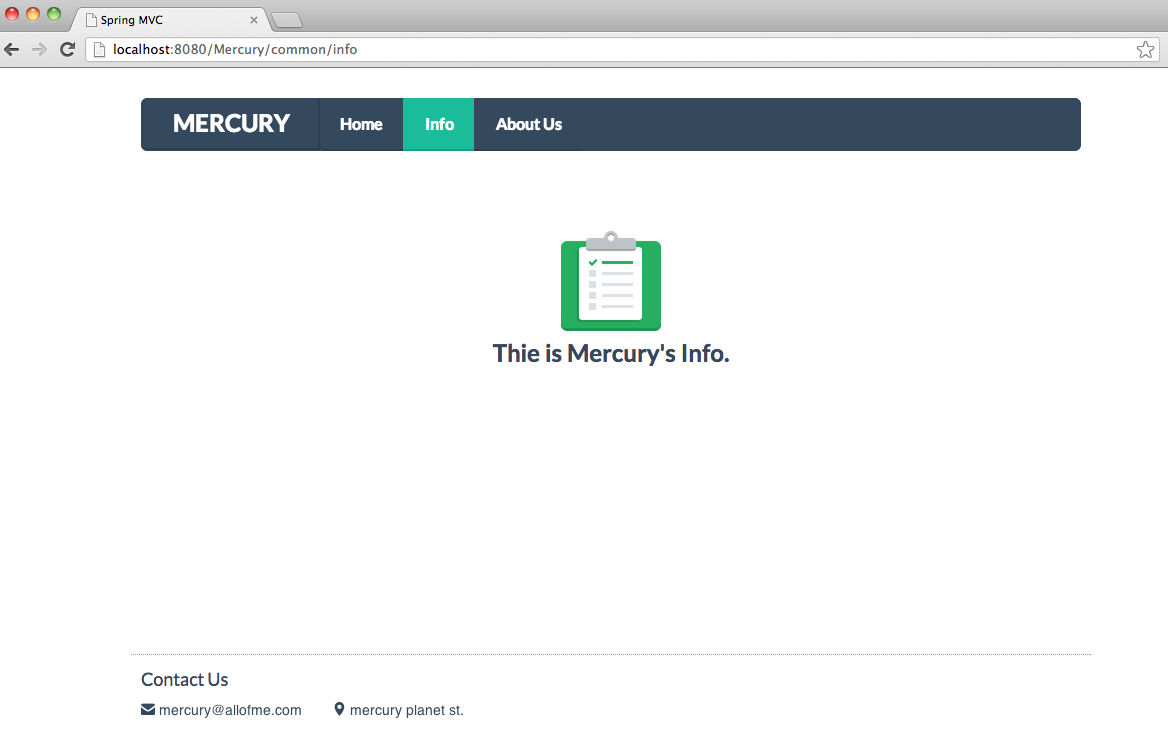
首先必須先將你的裝飾頁面的路徑指定，如範例中為 defaultdir="/pages/layouts"，因實際專案中會有多個裝飾頁面，故可以設定多組的 decorator ，每一個 decorator 都要擁有自己的名稱 name="base" ，以及這個 decorator 是使用哪一個裝飾頁面 page="base.jsp"，最後只需要將內容頁面的路徑，如範例  <pattern>/common/home</pattern> 加至你想要的 decorator 底下即可。  
  
我們專案檔案目前分布如下：

[](http://4.bp.blogspot.com/-ouF9gQvkuZY/Ud95bbhUuNI/AAAAAAAAAtQ/AkdumFGznJU/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8811.34.52.png)

以上幾個步驟就完成了 SiteMesh 的設定，接著來實際瀏覽我們的結果畫面，進入 /common/home 時畫面如下：

[](http://2.bp.blogspot.com/-HeQxkyEQycA/Ud9zrXzCreI/AAAAAAAAAsw/t6wdWR3C8iE/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8811.10.06.png)

可以看到之前的 base.jsp 與 home.jsp 結合在一起呈現，切換 Menu 選單至 /common/info 頁面：

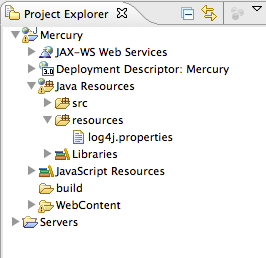
[](http://2.bp.blogspot.com/-PgrQ7vsSDgw/Ud902_EWcYI/AAAAAAAAAtA/EjQstmC0D8k/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-12+%E4%B8%8A%E5%8D%8811.15.18.png)

兩個頁面都順利地照 decorator.xml 設定的套用 base.jsp ，如此一來往後遇內容重複的部份都可以思考如何使用 SiteMesh 來幫助開發，SiteMesh 提供的功能當然不只這次所討論的部份，詳細可以參考官方文件。最後提醒，SiteMesh 雖然方便，但裝飾頁面與內容頁面 js 檔案引入的順序還是要小心處理。  
  
p.s. CSS Framework 使用 Bootstrap 延伸的模組。

**Spring MVC 3 實作教學 (7) - 使用 log4j ( 搭配 slf4j ) ( Use and configure log4j )**

完整的 Web Project 必定需要完善的 Log 紀錄，故將 Log 紀錄實作納入本次系列文章中。當然 Spring MVC 可以搭配的 Log Library 也有很多選擇， log4j 與 slf4j 的組合不過是其中的選擇之一。

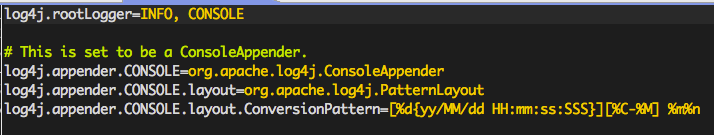
 首先於 [Apache log4j 網站](http://logging.apache.org/log4j/1.2/)下載我們需要的 Library， 下載完後將它放置到 /WebContent/WEB-INF/lib 底下。除了需要將 Library 放到指定路徑下，我們還須要一個 log4j 的設定檔，檔名為 log4j.properties，我習慣擺放路徑如下：

[](http://3.bp.blogspot.com/-QW1vyNOy_m8/UeT5BQO91nI/AAAAAAAAAts/8QZ94tay8r0/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%883.39.36.png)

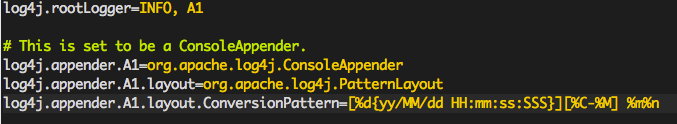
除了  log4j 之外，我們還需要到 [slf4j 網站](http://www.slf4j.org/)下載 Library，一樣將我們需要的兩個檔案放置到 lib 路徑底下。故我們總共需要三個 JAR 檔分別為 log4j-1.2.17.jar、slf4j-api-1.7.5.jar、slf4j-log4j12-1.7.5.jar，當然 JAR 檔的版本盡量是使用最新的版本較佳，如下圖：

[](http://3.bp.blogspot.com/-wjl0pa0Fbjo/UeT8a7ttKEI/AAAAAAAAAt8/P-XaF3Yi7fA/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%883.55.05.png)

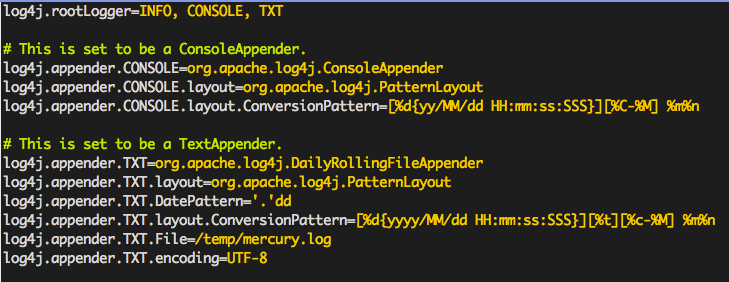
如此我們的準備工作完成，接著我們還要來定義 log4j.properties 內容，內容如下：

[](http://4.bp.blogspot.com/-fpmtULv27sU/UeT-Pknh4TI/AAAAAAAAAuM/y6TBYfzezYk/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%884.02.53.png)

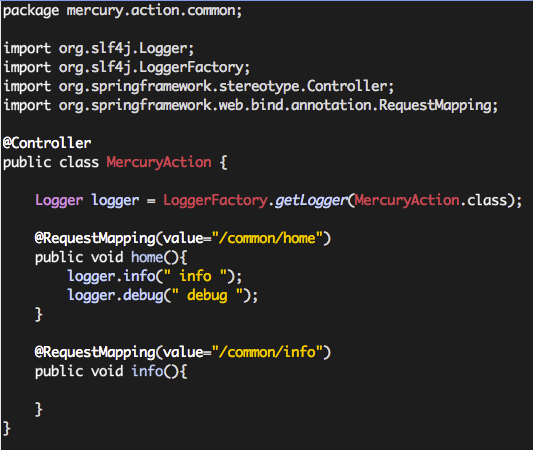
因 log4j  定義了幾個 log 輸出的層級分別為 TRACE、DEBUG 、INFO 、WARN、ERROR、FATAL，這幾個層級關係到我們 Log 輸出的資訊，照上面範例可看到  log4j.rootLogger 部份定義參數為 INFO 與 CONSOLE，代表 Log 輸出層級設定為 INFO，而後面的 CONSOLE 不代表層級，而是指定 log 輸出模式的代號 log4j.appender.CONSOLE。若將 CONSOLE 改為 A1 輸出模式則設定檔內容如下：

[](http://2.bp.blogspot.com/-9u6mcXX7Xuo/UeUCUoO4SnI/AAAAAAAAAuc/Ti1U-wT3nTs/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%884.20.15.png)

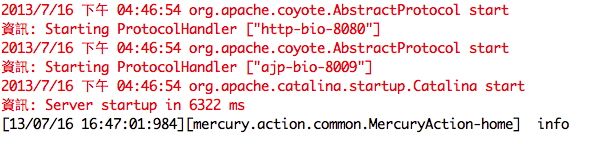
範例目前設定為輸出模式為顯示在 Console 畫面，而你也可以同時輸出多個模式，例如增加輸出文字檔案，如下圖：

[](http://2.bp.blogspot.com/-_wSqGgkgBsQ/UeUFC3md3NI/AAAAAAAAAus/YNsg_IIINO8/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%884.31.30.png)

設定檔完成之後，於 Controller 中做個測試，如下：

[](http://1.bp.blogspot.com/-0LZPFx-CghA/UeUFxeqsW4I/AAAAAAAAAu0/hxFy29wyr6Y/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%884.34.48.png)

可看到我們在 Controller 中增加了兩個輸出訊息方式，隨著 log4j 對應的層級也分別有它對應的 Method ，例如 INFO 對應為 logger.info( ) 而 DEBUG 對應 logger.debug( )。還記得我們在 log4j.properties 裡的 log4j.rootLogger 定義了輸出層級為 INFO 嗎，雖然在 home () 裡面執行了兩個 Log Method ，但根據層級設定為 INFO 時，實際輸出的資訊只會有 logger.info( ) 的部份，根據設定檔的的參數不同，於輸出 Log 更加有調整彈性。測試結果如下：

[](http://3.bp.blogspot.com/-K1YXFatECZw/UeUIvZav_LI/AAAAAAAAAvE/ABpNSGsgSO0/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-07-16+%E4%B8%8B%E5%8D%884.47.07.png)

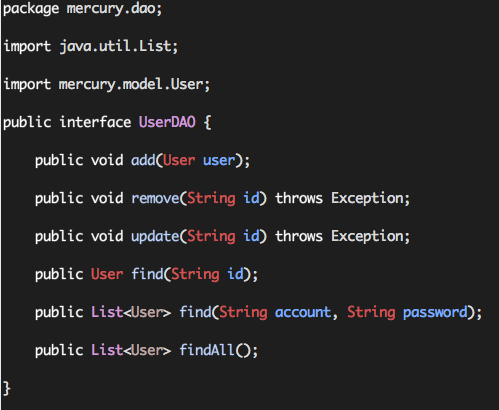
如此一來就完成了 lo4j 與 slf4j 的搭配使用。當然 log4j 還有其他更多的應用，詳細的部份可以參考官網文件。

**Spring MVC 3 實作教學 (10) - 資料存取 ( Transaction、DAO 設定 )**

前篇文章介紹了 Hibernate 與 Entity 的設定，建立了 Model 後還缺少了實際新增、修改、刪除的部份，因此本篇文章將會介紹 DAO、Service 部份如何實作。當然一旦有存取的發生勢必就需要 Transaction 控制，所以本篇也會一起介紹如何設定 Transaction Manager。

**DAO :**

這部份開始前可以先參照之前建立的 User 物件，通常在撰寫 User 物件的新增、修改、刪除功能會抽離為兩個部份。為了增加程式上的彈性通常會定義出功能的介面與實作兩個部份，讓應用程式只知道介面部份，而實作部份就可根據不同狀況下調整程式，現在針對它來建立 DAO 介面部份。首先將 DAO 會擁有的 Method 列出，如下圖：

[](http://2.bp.blogspot.com/-WAa9Y_JEadg/Uh-KkBdkbgI/AAAAAAAAA10/_e-4cdDwyLo/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-30+%E4%B8%8A%E5%8D%881.52.34.png)

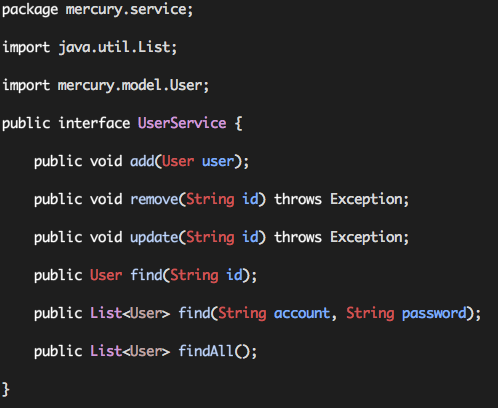
接著實作 UserDAO，針對每個 Method 定義出實際內容，如下圖：

[](http://3.bp.blogspot.com/-eU8VadQWLOg/Uh-T1R_jjUI/AAAAAAAAA2k/nuIDbWlB20M/s1600/2013-08-30_023108.jpg)

上面我們將每個 Method 都一一實作出來，這邊記得必須使用 sessionFactory.getCurrentSession( ) 來運作。另外須注意一點，範例中 find 方法為 sessionFactory.getCurrentSession( ).load( ) ，Hibernate 有提供另一種方法 sessionFactory.getCurrentSession( ).get( )，其中差異請至 Hibernate 官方文件參考。

**Service :**

DAO 部份設定好後，我們還需要 Service 來提供服務。Service 定義的部份也是抽離為介面與實作兩個部份，Service 介面如下：

[](http://1.bp.blogspot.com/-E5CzeZgqZUE/Uh-MU7YH86I/AAAAAAAAA2A/YqJasQvl1sw/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-30+%E4%B8%8A%E5%8D%882.00.22.png)

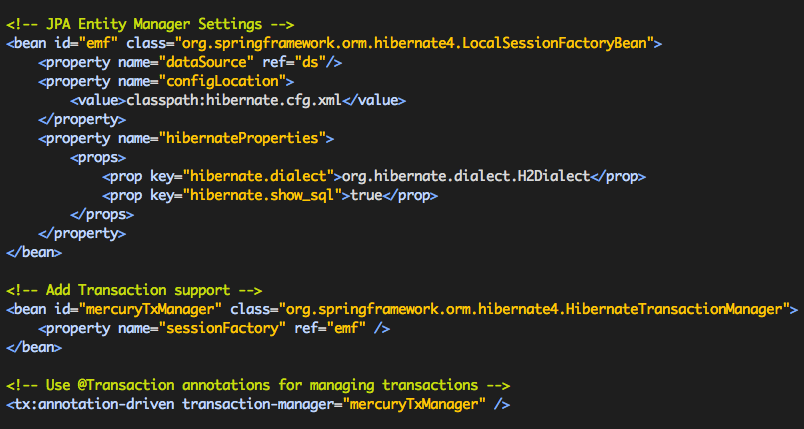
接著實作 UserService，內容如下：

[](http://1.bp.blogspot.com/-NlJO5QaEVZA/Uh-R7orXs_I/AAAAAAAAA2c/utdaJo3111k/s1600/2013-08-30_021837.jpg)

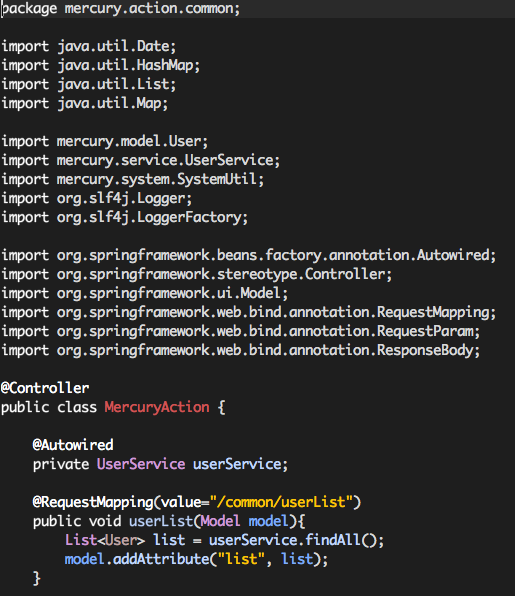
這邊使用剛剛定義的  UserDAO，並且透過 @Service 標註此 Class  為 Service 元件。@Transactional 可針對此 Class 或 Method 附加上交易的管理。

**Transaction Manager :**

剛剛看到 Service 實作裡面有 @Transactional 的 Annotation，因此我們要在 Spring 設定檔裡宣告 Transaction Manager 以及其屬性。不過在設定 Transaction Manager 之前，我們還需要 [AOP 的 Library](http://ebr.springsource.com/repository/app/bundle/version/detail?name=com.springsource.org.aopalliance&version=1.0.0) 以及 [aspectjweaver](http://search.maven.org/) JAR檔 ( 搜尋 aspectjweaver 取得最新版本 )，放置於 /WebContent/WEB-INF/lib 底下。  
Transaction Manager 於 Spring 設定檔 ( 如範例 mercury-servlet.xml ) 內容如下：

[](http://3.bp.blogspot.com/-bCCI8YhXDN4/Uh-Unp3FrMI/AAAAAAAAA2s/wX_s_xGyF4E/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-30+%E4%B8%8A%E5%8D%882.34.59.png)

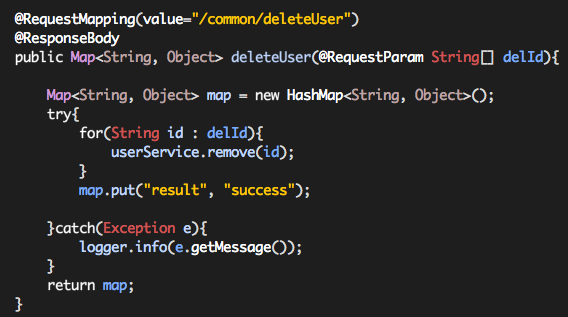
如此一來 Transaction Manager 基本上設定好了，Transaction 預設為發生 RuntimeException 時會 Roll Back，但如果想要指定某些特定的 Exception 發生時 Roll Back 或不 Roll Back。因此這自訂動作就需要運用到我們剛剛加入的 AOP 部份，詳細可以參考[官方說明文件](http://static.springsource.org/spring/docs/3.2.x/spring-framework-reference/html/transaction.html)。  
  
接著我們在 Controller 建立基本資料存取的範例，首先查詢資料範例內容如下：

[](http://2.bp.blogspot.com/-0Ecw6vT6SYc/UiDmlqzhpsI/AAAAAAAAA28/OztbzI15zEc/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-31+%E4%B8%8A%E5%8D%882.21.47.png)

新增資料範例如下：

[](http://3.bp.blogspot.com/-THKUr5kMUnk/UiDpjvDhz0I/AAAAAAAAA3I/Q9rJpsEUyFA/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-31+%E4%B8%8A%E5%8D%882.50.12.png)

刪除資料範例如下：

[](http://4.bp.blogspot.com/-nQhx3fxXsrI/UiDrkZdoVKI/AAAAAAAAA3U/OcMmvf4sFbo/s1600/%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7+2013-08-31+%E4%B8%8A%E5%8D%882.58.32.png)

如此一來 Spring MVC CRUD 的基本功能都建立完成了。