第 2 章 Spring 入門

Spring 是個全方位的應用程式框架（Application framework），要瞭解所有 Spring 的功能需要相當的時間，不過要完成您第一個 Spring 程式卻是相當簡單的，在這個章節中，將實際完成您第一個 Spring 程式，以增強您對使用 Spring 的信心。

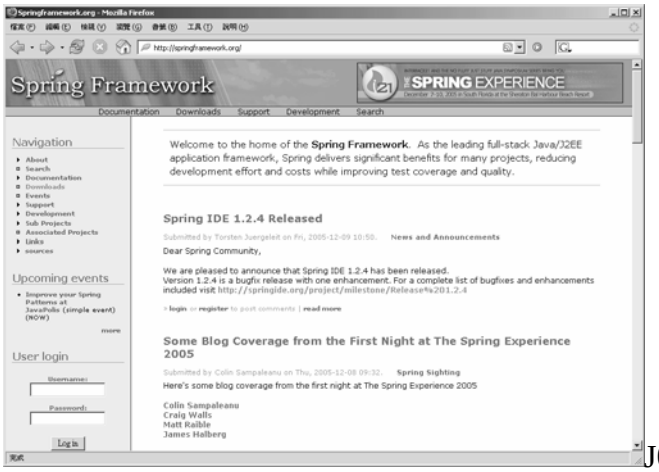
在本書中所有的 Spring 程式將使用 Eclipse 來作爲整合開發環境（Integrated Development Environment, IDE），這是多數使用 Spring 開發程式的開發人員之選擇，而這個章節中也將介紹如何在 Eclipse 中安裝 Spring IDE，並認識一下 Spring IDE 的一些基本功能。

2.1 第一個 Spring 程式

作爲一個應用程式框架，入手第一個 Spring 程式是很簡單的，這一個小節中將進行 Spring 的相關檔案下載，並使用 Eclipse 進行設定、撰寫、執行，完成您第一個 Spring 程式。

2.1.1 下載、設定 Spring

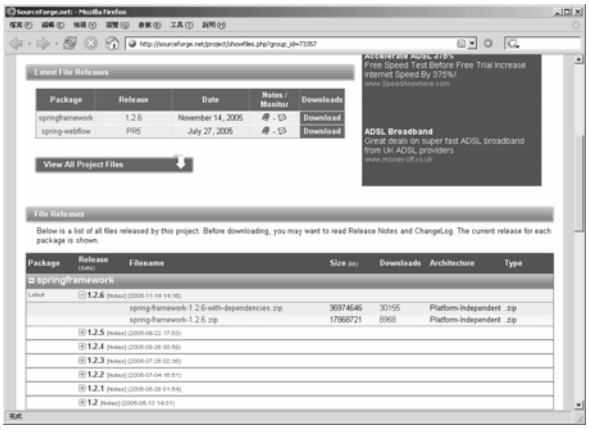
撰寫此書的時候，Spring 最新的版本是 1.2.6，請連接至 Spring 的官方網站進行檔案的下載，網址是在 **http://springframework.org/**：



J003

圖 2.1 Spring 官方網址

點選左邊的「Downloads」鏈結，可以進入至下載頁面，下載頁面中會有個「Download it from Sourceforge.」鏈結連接至 **Sourceforge**（http://sourceforge.net/project/showfiles.php?group_id=73357）：



J004

圖 2.2 從 Sourceforge 下載 Spring 相關檔案

在 Sourceforge 上有兩個可供下載的檔案，一個是 spring-framework-1.2.6-with-dependencies.zip，一個是 spring-framework-1.2.6.zip。with-dependencies 的下載檔案包括了一些 ant、jakarta-commons、struts、velocity 等等其它開源 Java 專案的相依 library 檔案，如果您也需要這些相關檔案，可以下載這個版本，如此就不用再分別至各個專屬網站下載這些檔案，如果您已經有這些相關檔案，則只需要下載 spring-framework-1.2.5.zip 這個檔案。

建議您直接下載 spring-framework-1.2.6-with-dependencies.zip，選擇檔案的下載鏈結之後，會需要您選擇鏡射（mirror）網站，您可以選擇一個最接近您下載所在區域的鏡射網站，下載時會比較快一些。

下載 zip 檔案並解壓縮之後，在 dist 目錄下就是使用 Spring 所需要的相關檔案，如果下載的是 with-dependencies 版本，則在 lib 目錄中的是您可能會用到的相依檔案，如果您需要查詢 Spring 的 API 文件，或者需要閱讀 Spring 參考手冊，則可以 docs 目錄中找到相關文件。

J005

圖 2.3 spring-framework-1.2.6-with- dependencies 的內容

在 dist 目錄下，spring-core.jar 是 Spring 的核心，如果日後需要使用到 Spring 其它的子框架支援，再將其它的 jar 檔案加入至 Classpath 的設定即可，例如 spring-aop.jar、spring-webmvc.jar 等。您也可以直接使用 spring.jar 這個檔案，它包括了所有 Spring 支援的功能所需要的類別，而不再需要加入個別的 jar 檔案。

注意在 Spring 1.2 之後，原先於 spring-core.jar 中與 Bean 相關的一些套件，現在已移至 spring-beans.jar 中。

良葛格的話匣子

爲了統一操作的環境，假我跟您的 spring 相關檔案都是放在 **C:\workspace\library\spring-framework-1.2.6-with-dependencies\spring-framework-1.2.6**，請參考圖 2.3 的說明，另外我所安裝的是 **JDK 5.0**，而所使用的整合開發環境是 **Eclipse 3.1**，您可以在 Eclipse 官方網站 **http://www.eclipse.org/**進行下載，使用 Eclipse 的原因是它很純粹，可以讓您接觸到撰寫 Spring 時所要注意的一些細節，當然，很多開發人員選擇使用 Eclipse 來開發 Spring 相關應用程式也是原因之一。

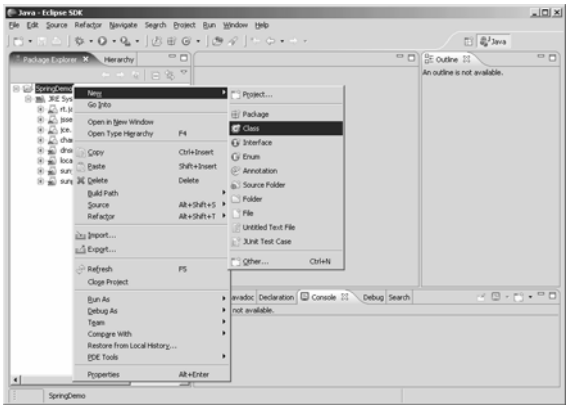
2.1.2 準備 Spring 設計環境（使用 Eclipse）

在使用應用程式框架撰寫程式時，最好是搭配使用整合開發環境（Integrated Development Environment, IDE）以方便進行各種資源（如 Classpath）的管理，這邊假設您已經下載了 Eclipse 3.1，請執行 Eclipse 之後先進行專案的新增，執行 Eclipse 選單上的「File/New/Project...」項目之後，在「New Project」對話方塊中選擇「Java Project」，按下「Next」按鈕，並爲您的專案取名稱，這邊假設是 SpringDemo 名稱，而專案是儲存在 **C:\workspace** 目錄下，如下圖所示：

J006

圖 2.4 在 Eclipse 中新建一個 SpringDemo 專案

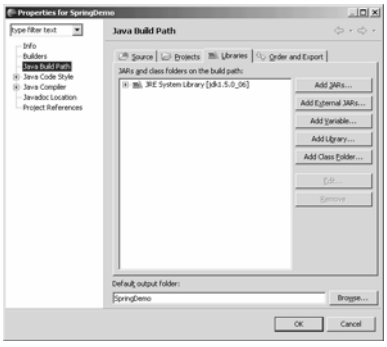
在專案新建之後，可以在專案上直接按滑鼠右鍵新增各種檔案，以後各章節中若提到新建某檔案，都是這樣進行操作，如下圖所示：



J007

圖 2.5 在 Eclipse 中新建檔案的方式

接著您必須將 Spring 相關的 library 加入至專案管理之中（具體而言就是將.jar 檔案設定至 Classpath 中），請執行選單上「Project/Properties」項目，選擇「Properties for SpringDemo」對話方塊上的「Java Build Path」，並切換頁面標籤至「libraries」，如下圖所示：



J008

圖 2.6 設定 library

您可以在這個頁面中加入相依的 jar 檔案，例如按下「Add External JARS...」即可選擇.jar 檔案的來源，接下來要練習的第一個 Spring 程式，要將 spring-core.jar、spring-beans.jar，以及相依的 commons-logging.jar 加入，您可以在 dist 目錄及 lib 目錄的 jakarta-commons 目錄中找到這些檔案，加入完成後按下「OK」按鈕即可。

2.1.3 撰寫第一個 Spring 程式

來撰寫您的第一個組件（Component），它只是一個簡單的 JavaBean，用來向使用者打聲招呼：

SpringDemo HelloBean.java

```
package onlyfun.caterpillar;

public class HelloBean {
    private String helloWord;

    public void setHelloWord(String helloWord) {
        this.helloWord = helloWord;
    }

    public String getHelloWord() {
        return helloWord;
    }
}
```

稍後您可以透過 setHelloWord()這個 Setter 來設定新的招呼語，不過並不是親自撰寫程式來作這些事，而是在組態檔案加以定義，由 Spring 來爲您作設定的動作，接著可以撰寫 Bean 的定義檔案，定義檔案會告訴 Spring 容器，如何完成相依物件的關係注入等動作，Bean 定義檔的檔案名稱可以自由定義，例如這邊取名爲 beans-config.xml：

SpringDemo beans-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING/DTD BEAN/EN"
  "http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <bean id="helloBean"
          class="onlyfun.caterpillar.HelloBean">
        <property name="helloWord">
            <value>Hello!Justin!</value>
```

```
        </property>
    </bean>
</beans>
```

在 Bean 定義檔中是以**<beans>**作爲根節點，而使用**<bean>**來爲每一個 Bean 進行設定，**"id"**屬性用以設定 Bean 的實例別名，稍後可以使用 id 來取得 Bean 的實例，class 屬性用來指定 Bean 的類別名稱，**<property>**標籤的 helloWord 設定了 Setter 的名稱（setHelloWorld），並在**<value>**標籤上設定了將注入的字串值。

接著來撰寫一個簡單的示範程式：

SpringDemo SpringDemo.java

```
package onlyfun.caterpillar;

import org.springframework.core.io.FileSystemResource;
import org.springframework.core.io.Resource;
import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;

public class SpringDemo {
    public static void main(String[] args) {
        Resource rs =
            new FileSystemResource("beans-config.xml");
        BeanFactory factory = new XmlBeanFactory(rs);

        HelloBean hello =
            (HelloBean) factory.getBean("helloBean");
        System.out.println(hello.getHelloWord());
    }
}
```

org.springframework.core.io.FileSystemResource 類別實作了 org.springframework.core.io.Resource.Resource 介面，可指定絕對路徑或相對路徑來指定 Bean 定義檔的位置，由於這邊使用的是 XML 定義檔，所以使用 org.springframework.beans.factory.xml.XmlBeanFactory 類別，XmlBeanFactory 實作

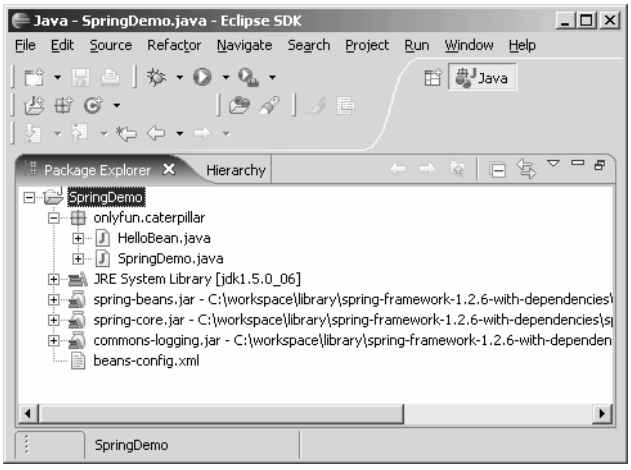
了 **org.springframework.beans.factory.xml.BeanFactory** 介面，用來讀取定義並建立 BeanFactory 實例。

BeanFactory 是 Factory 模式的一個實作例子，但用途更爲一般，可以建立、管理不同型態的物件，在 Spring 1.2 之後，XmlBeanFactory 只接受實作 Resource 介面的物件，像是 ClassPathResource、FileSystemResource、InputStreamResource、ServletContextResource、UrlResource 等，如果您的 Bean 定義檔是位於 Classpath 路徑中，您也可以使用 ClasspPathResource 來取得定義檔。

這是從比較低層次的角度來使用 Spring 的 IoC 容器功能，藉由 BeanFactory 來讀取組態檔案並完成依賴的關係注入，這邊的依賴是什麼？指的是 HelloBean 的實例相依於 String 物件，透過 Setter 所保留的介面，使用 Setter injection 來完成這個依賴注入，而不是將招呼語寫死在 HelloBean 中，BeanFactory 是整個 Spring 的重點所在，整個 Spring 的核心都圍繞著它。

BeanFactory 讀取 Bean 的組態設定並完成關係維護之後，可以藉由 getBean()方法並指定 Bean 的別名來取得 Bean 實例，如果使用 BeanFactory 的話，只有在真正需要 Bean 物件時，才會實際建立 Bean 實例，而不是在一開始建立 BeanFactory 時就建立 Bean 實例。

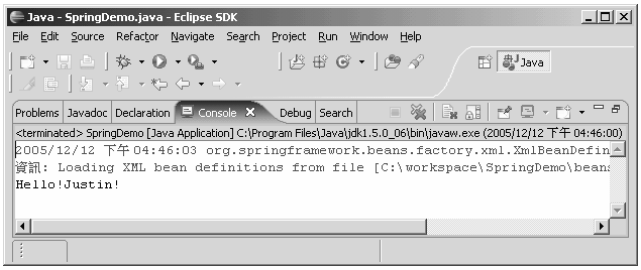
在 Eclipse 中，以上所撰寫的各檔案之相對位置如下圖所示：



J009

圖 2.7 第一個 Spring 程式

可以在開啓 SpringDemo.java 檔案後，執行選單上的「Run/Run As/Java Application」來看看實際運行之後的效果：



J010

圖 2.8 第一個 Spring 程式執行結果

如果今天您要想改變招呼語，則只要更改 beans-config.xml 就可以了，不用修改 HelloBean.java，從比較一般的角度來看，就意味著如果您想要改變一些物件之間的依賴關係，則只要修改組態檔即可，而不用修改組件的任何一行程式。

良葛格的話匣子

您可以在光碟的 examples 目錄下找到相對應的 SpringDemo 專案，您可以將之複製至您的 workspace 目錄，解除唯讀屬性，然後執行 Eclipse 選單上的「File/Import...」，選擇「Existing Projects into Workspace」，然後選擇 examples 目錄將專案匯入您的 workspace 目錄中，以後的章節中若有完整的範例示範，也都可以依這個方式找到對應的專案。

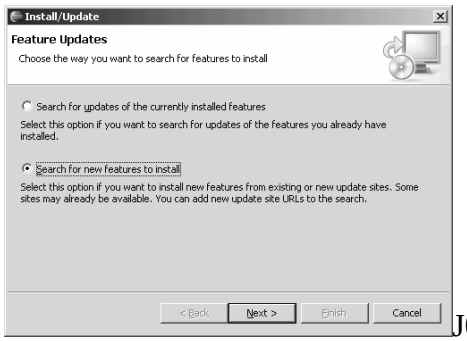
2.2 安裝、使用 Spring IDE

不免仍是老話一話：「工欲善其事、必先利其器」。使用 Eclipse 並結合 Spring 來開發應用程式是個不錯的組合，但可以更好，您可以在 Eclipse 中安裝 Spring IDE 的 plug-in，讓您在使用 Spring 開發應用程式時有更高的效率，這個小節中將來介紹如何安裝與使用 Spring IDE。

2.2.1 安裝 Spring IDE

Spring IDE 是 Spring 官方網站所推薦的 Eclipse plug-in，可提供您在開發 Spring 時對 Bean 定義檔進行驗證、以可視化的方式觀看 Bean 與 Bean 之間的依賴關係等功能，在這邊來介紹一下如何於 Eclipse 中安裝 Spring IDE。

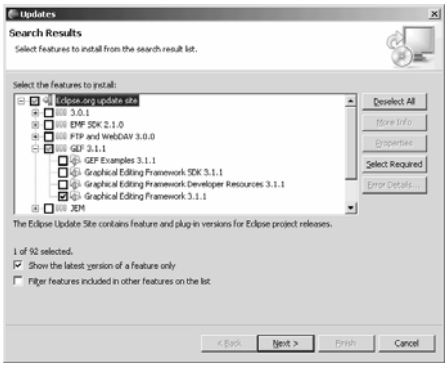
在進行 Spring IDE 的安裝之前，您必須先安裝 **Graphical Editing Framework**，因爲 Spring IDE 使用它作爲基礎以顯示 Bean Graph，也就是以圖形化的方式顯示 Bean 與 Bean 之間的依賴關係，首先請您執行 Eclipse 選單上的「Help/Software updates/Find and Install」，然後選擇「Search for new features to install」，再按下「Next」按鈕，如下圖所示：



J011

圖 2.9 搜尋可用的 Plug-in

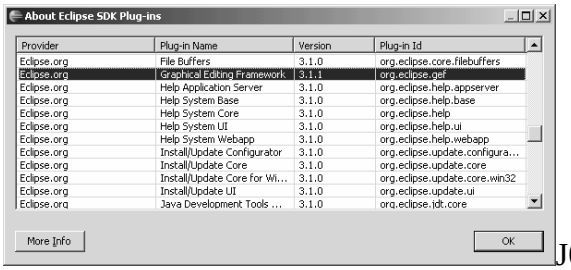
在下一個畫面中，您可以選擇 Update sites 或是其 Mirror 網站以進行 plug-in 的搜尋，搜尋完成之後，會列出所有可安裝的 plug-in，請選擇「Graphical Editing Framework」（以下簡稱 GEF），在撰寫這個章節時的 GEF 最新版本爲 3.1.1，由於這邊只是要使用 GEF，其它的 SDK 等相關檔案您不用安裝，一個示範畫面如下：



J012

圖 2.10 安裝 Graphical Editing Framework

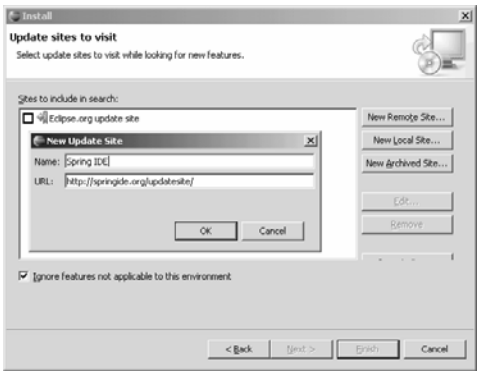
接下來就請依安裝精靈的指引，一步一步完成 GEF 的下載與安裝，安裝完成後會要求您重新啓動 Eclipse，重新啓動之後，您可以執行選單上的「Help/About Eclipse SDK」指令，並按下「Plug-in Details」來看看 GEF 是否安裝完成。



J013

圖 2.11 檢視 GEF 是否安裝完成

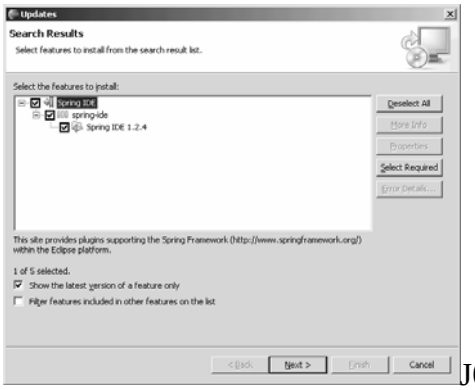
接下來您可以進行 Spring IDE 的安裝，請執行 Eclipse 選單上的「Help/Software updates/Find and Install」，然後選擇「Search for new features to install」，再按下「Next」按鈕，在接下來的「Install」視窗中，按下「New Remote Site...」按鈕，新增 Spring IDE 的 Update Site，網址是：**http://springide.org/updatesite/**：



J014

圖 2.12 新增 Spring IDE 的 Update Site

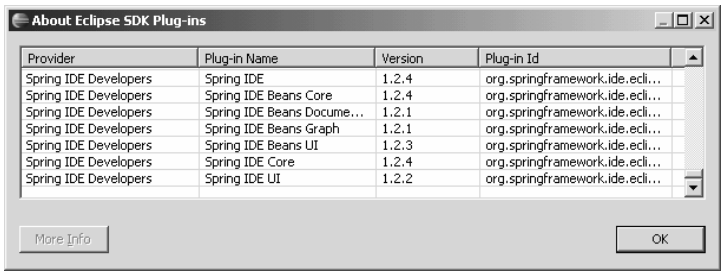
在新增 Spring IDE 的 Update Site 之後，請選擇新增的「Spring IDE」項目，並按下「Finish」按鈕以進行 plug-in 的搜尋，在撰寫這個章節時，Spring IDE 最新的版本爲 1.2.4，請如下圖選擇項目以進行安裝：



J015

圖 2.13 安裝 Spring IDE

接下來同樣的，請依安裝精靈的指引，一步一步完成 Spring IDE 的下載與安裝，安裝完成後會要求您重新啓動 Eclipse，重新啓動之後，您可以執行選單上的「Help/About Eclipse SDK」指令，並按下「Plug-in Details」來看看 Spring IDE 是否安裝完成。

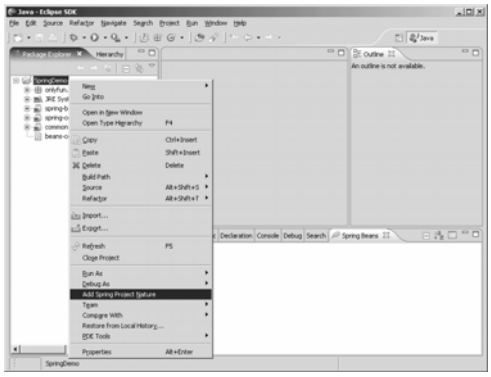


J016

圖 2.14 檢視 Spring IDE 是否安裝完成

2.2.2 使用 Spring IDE

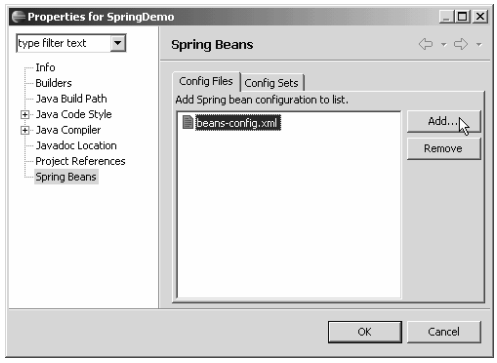
以先前您所開發的 SpringDemo 來作爲示範，要加入 Spring IDE 的功能支援，請在 SpringDemo 專案上按滑鼠右鍵，執行「Add Spring Project Nature」指令，如下圖所示：



J017

圖 2.15 在專案上加入 Spring IDE 支援

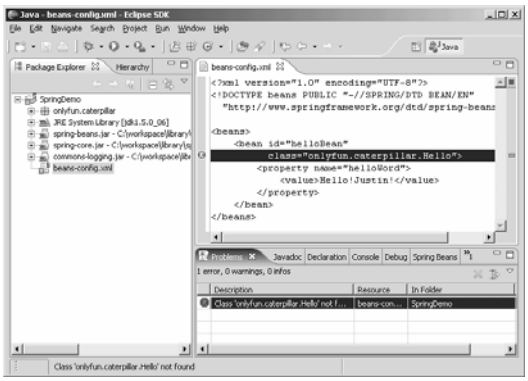
Spring IDE 可以協助您檢查 Bean 定義檔的內容是否設定正確，爲了擁有這個功能，您必須執行選單列上的「Project/Properties」指令，然後選擇「Spring Beans」項目，在「Config Files」頁籤中按「Add...」按鈕加入要進行驗證管理的 Bean 定義檔：



J018

圖 2.16 將 Bean 定義檔納入驗證管理

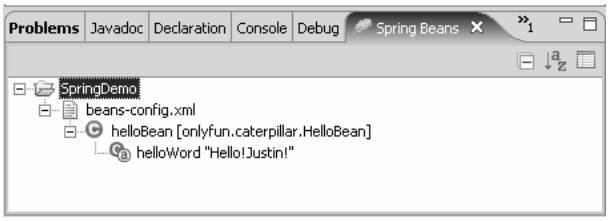
Spring IDE 除了可以驗證 XML 檔案中的標籤如<beans>等是否設定正確之外，它還可以協助您驗證所設定的類別名稱是否正確，例如若您設定時鍵入錯別的類別名稱，則會出現錯誤標示與說明：



J019

圖 2.17 Spring IDE 可檢驗出 Bean 定義檔上的錯誤設定

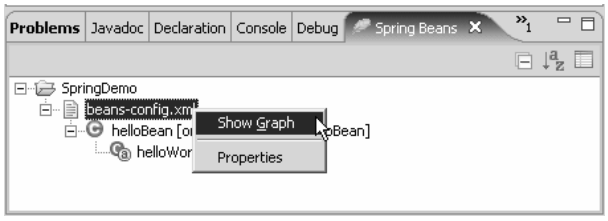
您可以切換 Eclipse 右下方的頁籤至「Spring Beans」，Spring IDE 可以在這個頁籤中以可視化的方式顯示 Bean 定義檔中的節點摘要：



J020

圖 2.18 Spring IDE 的 Beans 節點摘要顯示

您可以在 Bean 定義檔上按滑鼠右鍵執行「Show Graph」指令，Spring IDE 會以圖形化的方式顯示 Bean 定義檔中的 Bean 定義，以及 Bean 與 Bean 之間的依賴關係：



J021

圖 2.19 顯示 Bean Graph

爲了突顯 Spring IDE 可視化的 Bean 管理方式，我隨意的加入了一些 Bean 的定義，並擷取了 Bean Graph 的圖片供作參考：

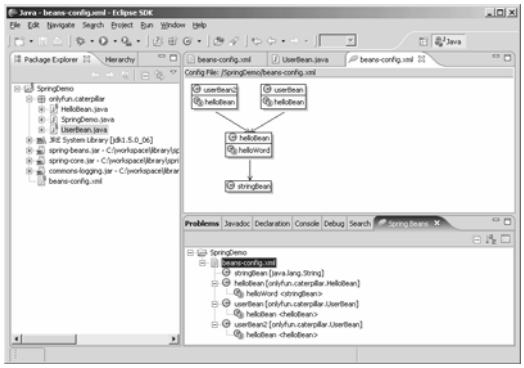

J022

圖 2.20 Spring IDE 的可視化 Bean 管理

2.3 接下來的主題

這個小節中以一個簡單的 Spring 程式，讓您了解如何在 Eclipse 中開發 Spring 應用程式，並也介紹了 Spring IDE 的入門使用，入門 Spring 是簡單的，但接下來就要一步一步的探索 Spring 的每一個功能，首先在下一個章節開始，將接續這個章節的第一個 Spring 程式，深入了解 Spring 的 IoC 容器之功能與設定方式。

J029

圖 2.21 您的第一個 Spring