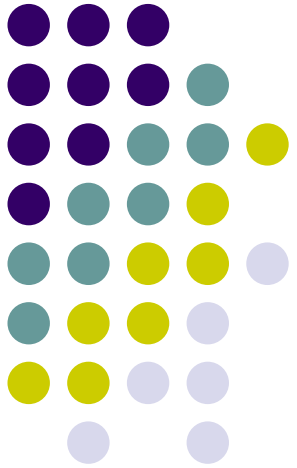
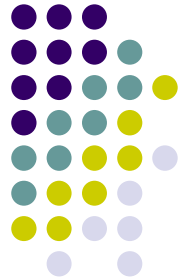
專題實驗(一)

http://mail.tku.edu.tw/125572/1002project2.ppt



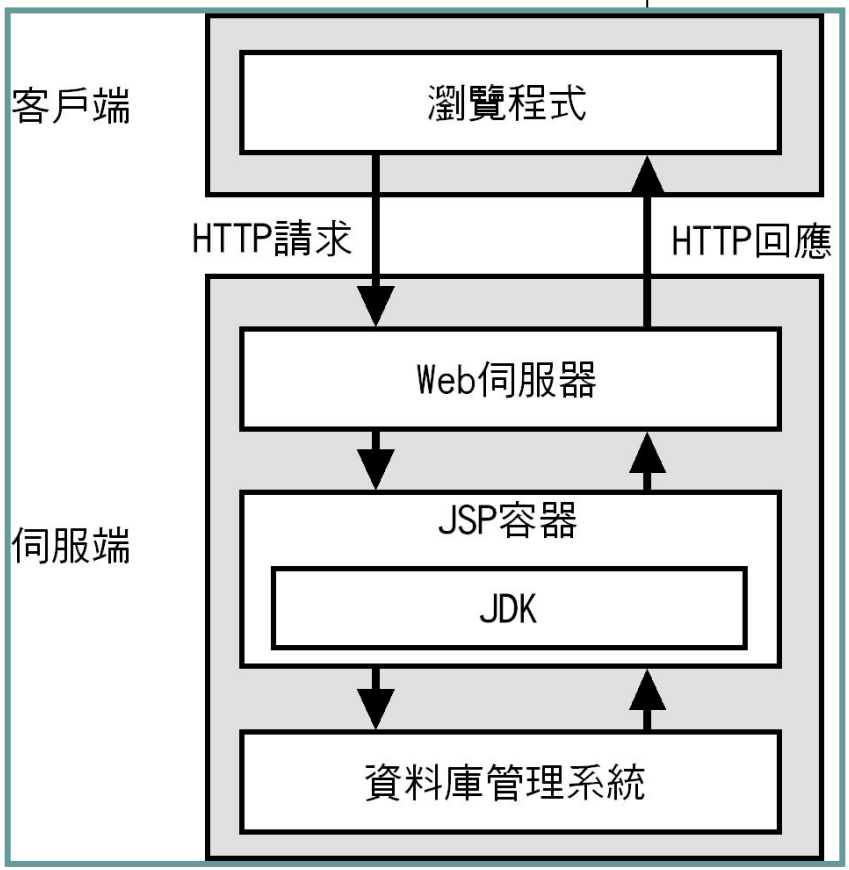
第1章 JSP的開發環境建立



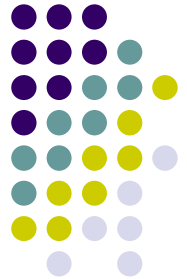
- 1-1 JDK的下載與安裝
- 1-2 Resin伺服器的下載和安裝
- 1-3 啓動與測試Resin伺服器
- 1-4 停止與關閉Resin伺服器

1 JSP的開發環境建立

- JSP技術
  - 一種伺服端網頁技術
  - 開發環境
    - 主從架構的開發環境，如右圖所示：



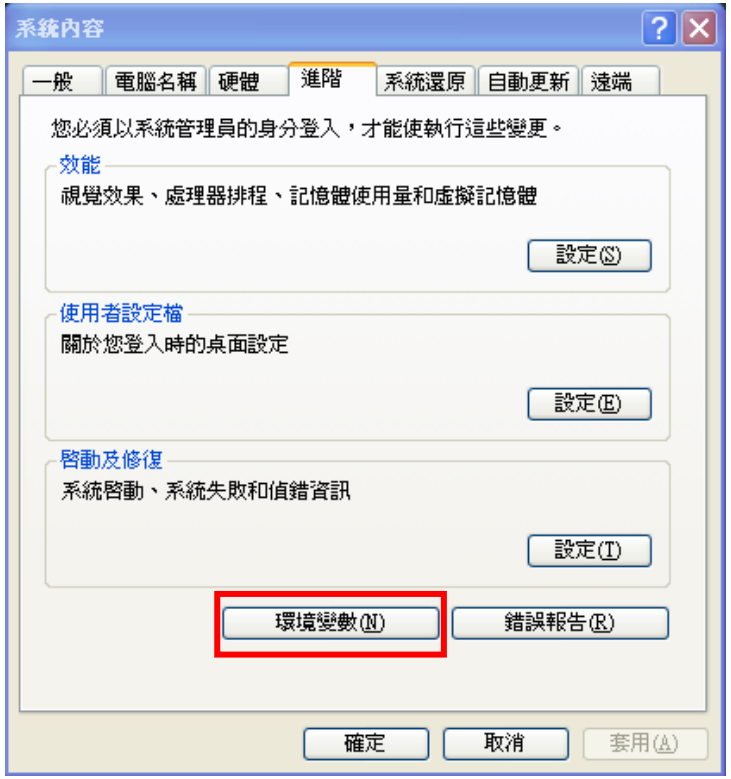
1-1 JDK的下載與安裝

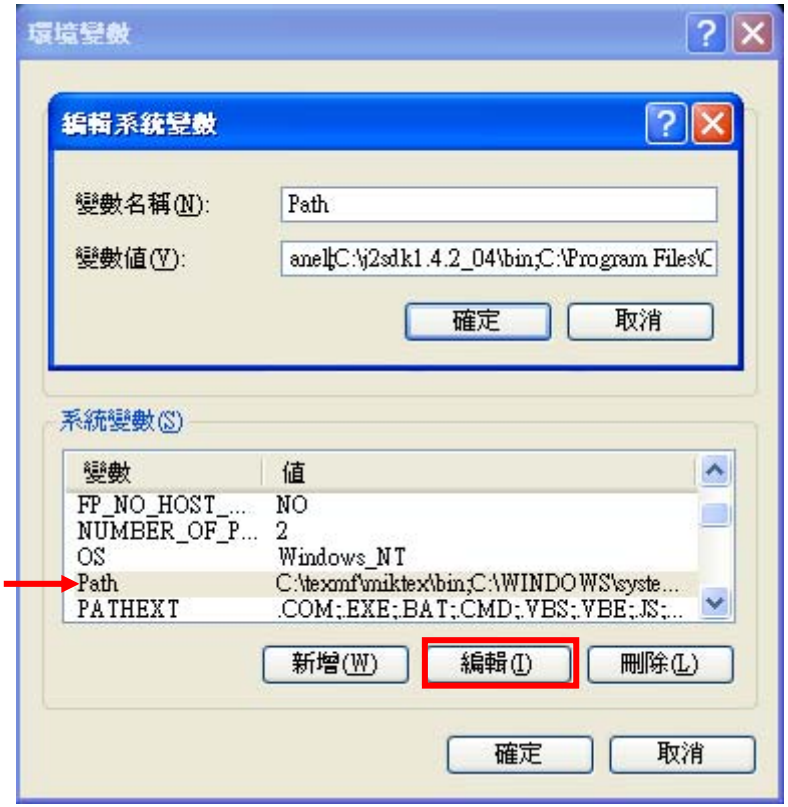


- JDK
  - 從昇陽網站免費下載，目前版本爲1.5.0版
  - 網址: http://java.sun.com/j2se/
- 在安裝好JDK後
  - 需要設定JDK執行環境
    - Windows XP新增環境變數Path的搜尋路徑
      「C:\Program Files\Java\jdk1.5.0\bin」
    - C:\Program Files\Java\jdk1.5.0\是JDK的安裝路徑。

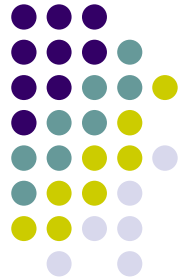
設定JDK執行環境

- 新增Windows XP環境變數
  - Path: 增加搜尋路徑
    - 「C:\Program Files\Java\jdk1.5.0\bin」





1-2 Resin伺服器的下載和安裝



- JSP容器（JSP Container）
  - 支援執行JSP程式的Web伺服器
- 目前市面上有多套應用程式或伺服器，如下所示：
  - Tomcat:
    - http://jakarta.apache.org/tomcat/index.html。
  - JRun
    - http://www.macromedia.com。
  - Resin (本課程使用的Web伺服器)
    - http://www.caucho.com。
  - ServletExec
    - http://www.newatlanta.com。

1-2 Resin伺服器的下載和安裝

- Resin伺服器
  - Caucho Technology開發的Servlet和JSP容器
  - 支援HTTP 1.1的快速Web伺服器和Servlet/JSP容器
  - 獨立運作來支援Java網頁技術的Web應用程式
  - 亦可以搭配其它Web伺服器
    - 例如：IIS，讓Web伺服器IIS也能夠開發和執行Servlet/JSP程式。
- Resin伺服器在3.0版
  - 支援Java Servlet 2.4和JSP 2.0版，提供負載平衡（Load Balancing）增加系統可靠度
  - 可以使用Servlet過濾（Servlet Filtering）功能來轉換XSL和XML，建立客戶端輸出的HTML、WAP或XML內容。

1-2 Resin伺服器的下載和安裝

- STEP1: 下載Resin伺服器
  - Resin伺服器目前的版本是 3.0版，在書附光碟提供resin-3.0.9.zip的ZIP格式壓縮檔，其下載網址，如下所示：
    - 下載Resin：
      http://www.caucho.com/download/index.xtp
- STEP2: 安裝Resin伺服器
  - 安裝Resin伺服器只需使用WinZIP或Windows XP的解壓縮精靈等工具即可解開ZIP格式的壓縮檔，以本書爲例是解壓縮到硬碟根目錄C:，預設的安裝路徑爲：「C:\resin-3.0.9\」。

1-3 啓動與測試Resin伺服器

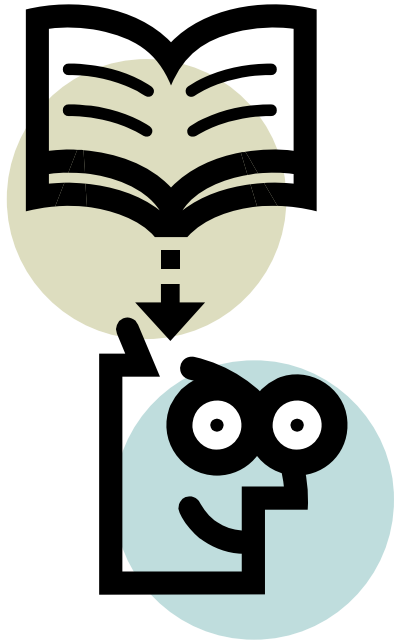
- Resin伺服器本
  - 可以在Windows作業系統下獨立執行的Web伺服器
  - 在解壓縮檔案安裝好Resin伺服器後，就可以啓動Resin伺服器，建立JSP技術的開發和執行環境。
- 測試伺服器是否成功啓動
  - STEP1: 開啓C:\resin-3.0.9\httpd.exe
  - STEP2: 啓動Resin伺服器後，我們就可以啓動Internet Explorer瀏覽程式測試伺服器是否成功啓動，請在瀏覽程式的【網址】欄輸入下列URL網址，如下所示：
    - http://localhost:8080/
  - 上述URL網址的localhost是指本機電腦，預設埠號爲8080，按Enter鍵，稍等一下，就可以看到網頁內容。

1-4 停止與關閉Resin伺服器

- 停止Resin的Web伺服器
  - 在「Resin」視窗選【Stop】選項，如此即可停止Resin的Web伺服器執行。
  - 按【Quit】鈕，就可以離開Resin伺服器。

第2章 JSP程式的開發工具

- 2-1 記事本
- 2-2 Gel中文介面開發工具



2-1 記事本

- Windows作業系統的記事本就可以編輯JSP原始程式碼
- 在Windows XP請執行「開始/所有程式/附屬應用程式/記事本」指令啓動記事本，如下圖所示：

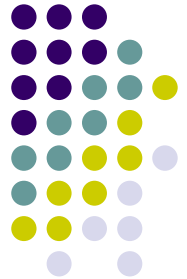
```
<!-- JSP程式：Ch1_3_2.jsp -->
<html>
<head>
<title>Ch1_3_2.jsp</title>
</head>
<body>
    測試JSP程式的執行.....<br>
    1 + 1 = <%= 1+1 %>
</body>
</html>
```

STEP1:
存成檔名爲**1.jsp**的檔案

STEP2:
將1.jsp檔移至**C:\resin-3.0.9\webapps\ROOT**

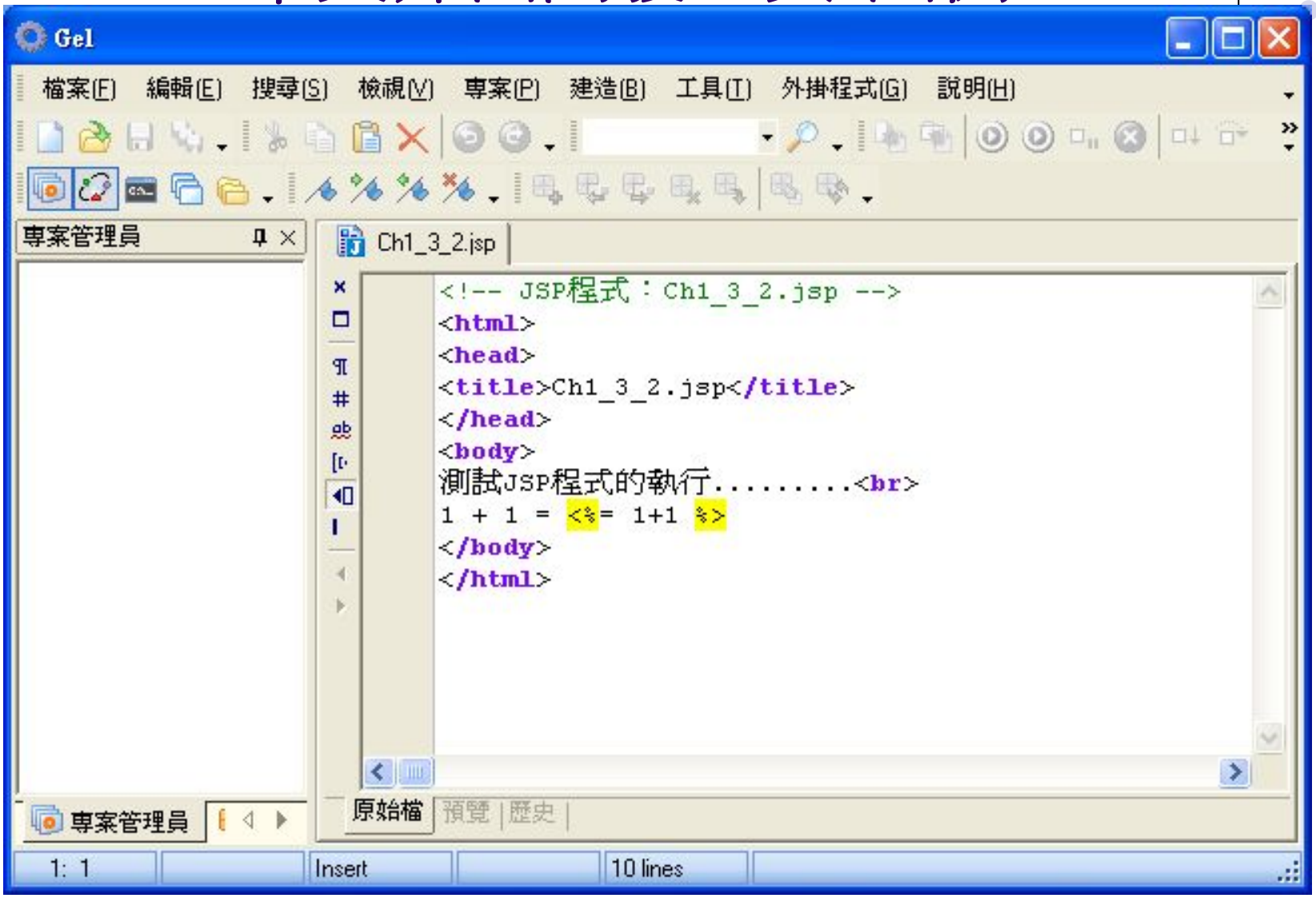
STEP3:
輸入網址:
http://localhost:8080/1.jsp

2-2 Gel中文介面開發工具-說明



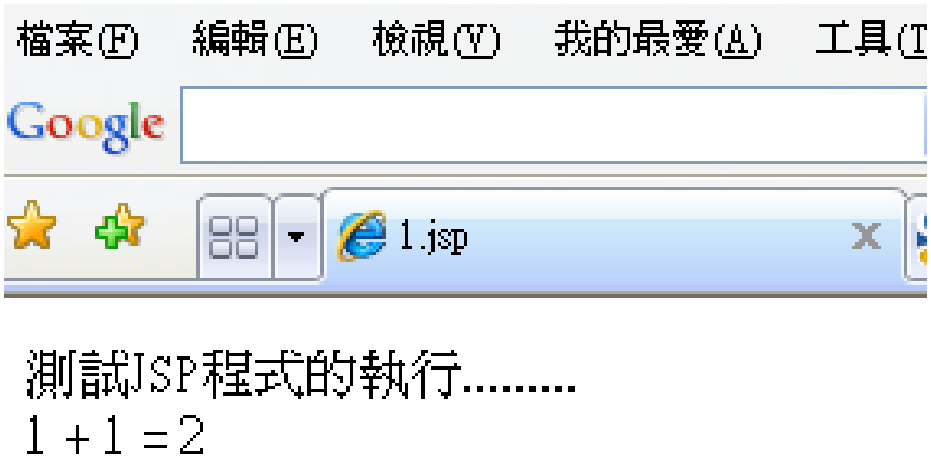
- Gel
  - 使用Delphi開發的Java/JSP IDE
    - 尺寸非常小、佔用記憶體少且執行速度快
  - Gel擁有強大功能
    - 提供中文使用介面
    - 語法標示
    - 程式碼自動完成
    - 參數提示和專案管理功能
  - 其下載網址
    - http://www.gexperts.com/download.html

2-2 Gel中文介面開發工具-圖例



隨堂練習

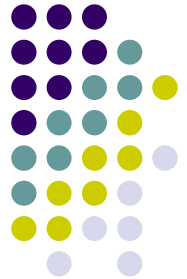
- 請執行右圖的jsp檔。



第3章 Java語法的JSP程式

- 3-1 Java語言的基礎
- 3-2 JSP程式的基本架構
- 3-3 Java的變數與資料型態
- 3-4 Java的運算子
- 3-5 Java的流程控制
- 3-6 Java的陣列與字串

3-1 Java語言的基礎-說明

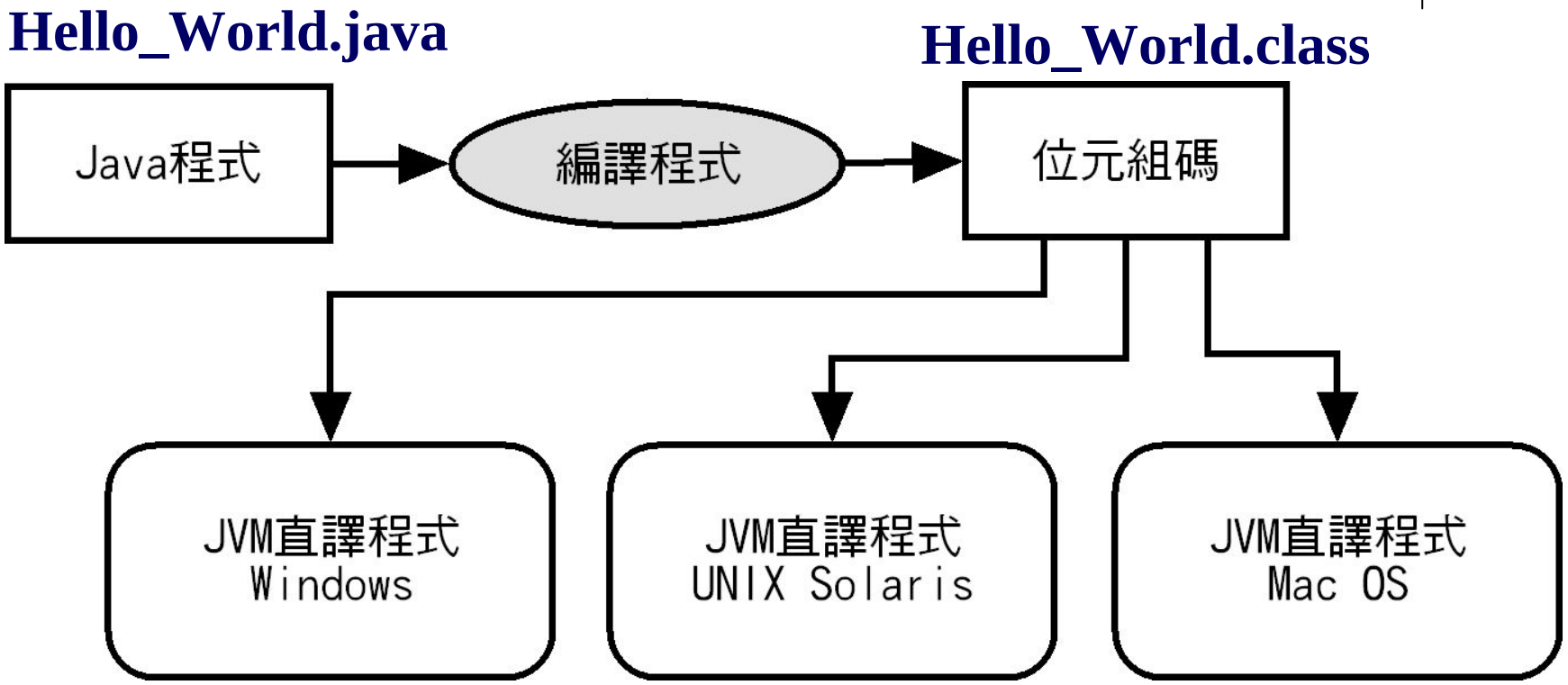


- Java語言
  - 是一種高階和物件導向程式語言
  - 類似C++語言的編譯式語言
  - 不過並不完全相同，因爲它是結合編譯和直譯優點的程式語言
- Java程式語言的「平台」（Platform）
  - 平台: 是一種結合硬體和軟體的執行環境，簡單的說，電腦程式是在平台上執行
  - 因爲Java屬於一種與硬體無關和跨平台的程式語言，所以Java平台是一種軟體平台
    - 主要是由JVM和Java API兩個元件所組成

3-1 Java語言的基礎-JVM(說明)

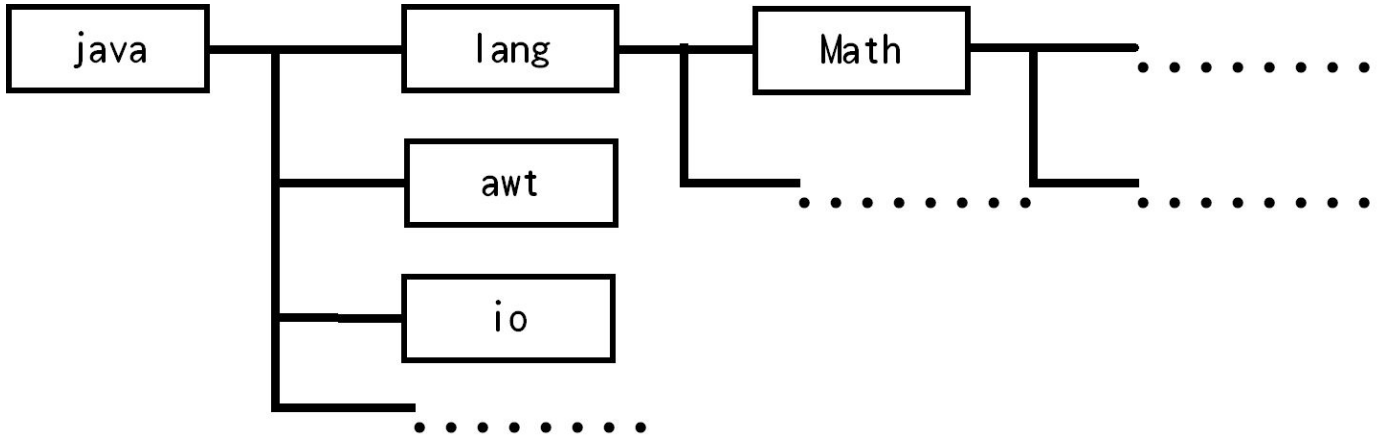
- Java編譯程式
  - 將Java原始程式碼編譯成「位元組碼」（Bytecode）
    - 這種程式碼是一種虛擬的機器語言，這台電腦稱爲「JVM」（Java Virtual Machine），換句話說，在作業系統需要安裝JVM的直譯程式，才能夠直譯和執行位元組碼。
- Java原始程式碼（副檔名.java）
  - 在編譯成位元組碼（副檔名.class）後
  - 可以在Windows、UNIX或Machintosh的Mac OS作業系統上執行
  - 只需作業系統安裝JVM直譯程式，同一個位元組碼檔案，就可以跨平台在不同作業系統上正確的執行。

3-1 Java語言的基礎-JVM(圖例)



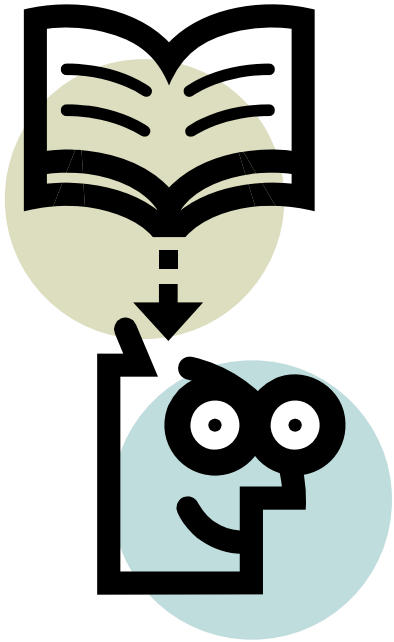
3-1 Java語言的基礎-Java API

- Java API (Application Programming Interface)
  - 軟體元件的集合
  - 也就是在C/C++語言所謂的函式庫，提供
    - 集合物件、GUI元件、檔案處理、資料庫存取和網路等相關的類別和介面，稱爲「套件」（Packages）
- Java標準API是一個名爲java的大型套件
  - 擁有多個子套件lang、awt和io等，每個子套件擁有許多類別，如下圖所示：



3-2 JSP程式的基本架構

- 3-2-1 第一個JSP程式
- 3-2-2 儲存和上傳JSP程式
- 3-2-3 執行JSP程式
- 3-2-4 JSP程式的執行過程
- 3-2-5 Java語言的命名與註解



3-2-1 第一個JSP程式-說明

- 爲了分辨HTML標籤與JSP程式
  - 所以使用特定符號的標籤來區分哪部分是HTML標籤，哪些部分是JSP程式碼
  - 稱爲JSP的「腳本元素」（Scripting Element），如下所示：
    - <%……%>
- 在標籤符號之間的是JSP程式碼
  - 伺服端支援JSP容器的Web伺服器
    - 將這部分程式碼轉換成Servlet程式
    - 在編譯和執行後，產生HTML文件送到客戶端瀏覽程式顯示。

3-2-1 第一個JSP程式-範例

```
<!-- JSP程式：Ch3_2_1.jsp -->
<%@ page contentType="text/html; charset=Big5"%>
<html>
<head>
<title>Ch3_2_1.jsp</title>
</head>
<body>
    <%! int from, to;  // 變數宣告 %>
    <% // 指定變數值
            from = 4;
            to = 6;
            // 設定字型尺寸由小變大
            for (int i = from; i <= to; i++ ) {
    %>
            <font size= <%= i %> >
             <p>第一個JSP程式</p>
    <% } %>
</body>
</html>
```

第一個JSP程式

第一個JSP程式

第一個JSP程式

3-2-1 第一個JSP程式-範例說明

- 程式開始的第2列是JSP指引元素的page指令，如下所示：

```
01: <%@ page contentType="text/html;
    charset=Big5"%>
```

  - page指令指定contentType屬性的網頁文件類型爲HTML文件，編碼爲繁體中文Big5
- 在<body>標籤區塊的HTML標籤擁有多個JSP程式碼區塊
  - 這些JSP腳本元素（Scripting Element）分爲3種

3-2-1 第一個JSP程式-Declarations元素(宣告元素)

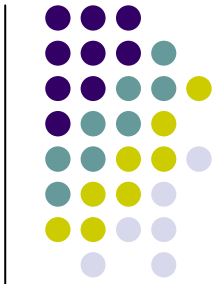
- Declarations元素：變數宣告
  - Declarations元素是位在下列符號之間的程式碼

    <%!　　%>

- 宣告元素
  - 不會產生任何輸出
  - Declarations元素宣告的變數或方法會翻譯（Translate）成Servlet類別的實例變數或方法宣告。
  - 例如：在第8列宣告整數int變數from和to的程式碼就是Declarations元素，如下所示：

```
08: <%! int from, to;  // 變數宣告 %>
```

3-2-1 第一個JSP程式-Scriptlets元素



- Scriptlets元素：指定敘述和for迴圈
  - 可以在JSP程式插入任何合法腳本語言的程式碼片斷，這是位在下列符號之間的Java程式碼：

    <%　　%>
- 例如
  - 使用Scriptlet元素指定變數值和for迴圈。迴圈結束前後都有HTML標籤，所以程式碼也需使用"<%"和"%>"符號標示出來。

3-2-1 第一個JSP程式-Expressions元素(表示式元素)

- Expressions元素：內含於HTML標籤的運算式
  - 一個插入腳本語言的運算式，在執行時會自動轉換成字串資料態，這是位在下列符號之間的Java程式碼，如下所示：

    <%＝　　%>
- Expression元素的運算式
  - 不需要Java語言的結束符號";“
  - 例如：指定<font>標籤的字型尺寸就是Expression元素，如下所示：

```
14: <font size=<%= i %>>
```

3-2-2 儲存和上傳JSP程式-儲存

- JSP程式的副檔名爲【.jsp】
  - 使用記事本或Gel儲存時，儲存成.jsp檔案
    - 例如：在【記事本】執行「檔案/儲存檔案」指令儲存檔案，可以看到「另存新檔」對話方塊。
- 在【存檔類型】選【所有檔案】
  - 【檔名】欄輸入檔案全名Ch3_2_1.jsp，包含副檔名【.jsp】，然後按【儲存】鈕即可儲存JSP程式。

3-2-2 儲存和上傳JSP程式-公佈

- 在建立好JSP檔案後
  - 需要將檔案公佈到Web伺服器的指定目錄
    - 以在第1章建立的開發環境爲例，如下所示：
    - **Resin的Web伺服器**
      - 複製到「C:\resin-3.0.9\webapps\ROOT\Ch??」資料夾，"??"是章節編號01、02、….16等，請自行建立Ch??子資料夾。

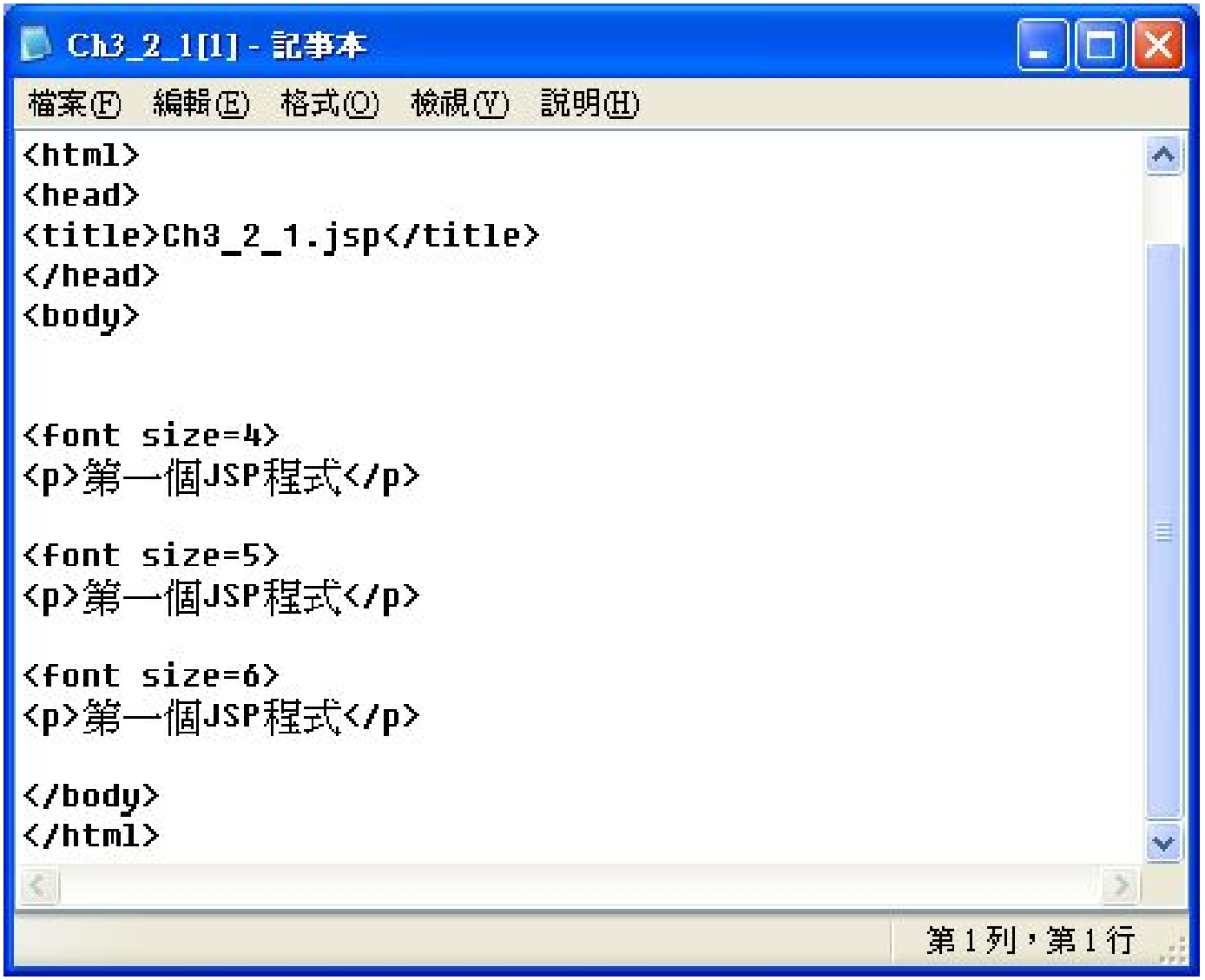
3-2-3 執行JSP程式-網址

- 在Internet Explorer瀏覽程式，在【網址】欄輸入URL網址，如下所示：
  - **Resin的Web伺服器**：URL網址的埠號爲8080，如下所示：

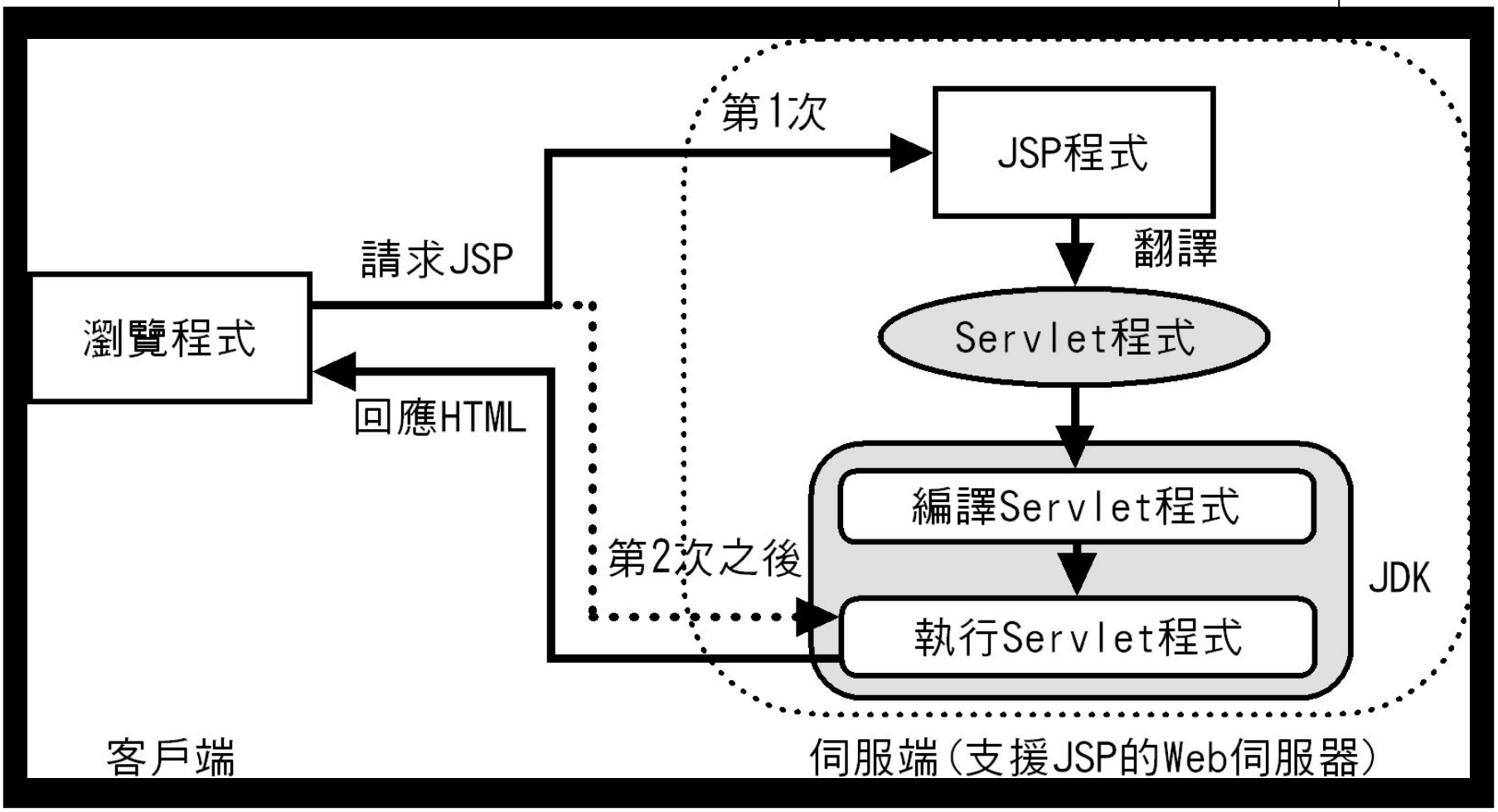
http://localhost:8080/Ch03/Ch3_2_1.jsp

3-2-3 執行JSP程式-圖例

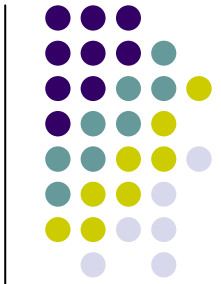
3-2-3 執行JSP程式-原始程式碼



3-2-4 JSP程式的執行過程-圖例



3-2-4 JSP程式的執行過程-檔案位置

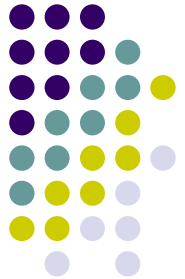


- 以Resin伺服器爲例
  - 轉換成的Servlet程式碼是儲存在「C:\resin-3.0.9\webapps\ROOT\WEB-INF\work_jsp」資料夾。
    - _ch3_02_01__jsp.java是JSP程式Ch3_2_1.jsp轉換的Servlet程式碼檔案。
    - _ch3_02_01__jsp.class是編譯後的Class類別檔，換句話說，我們真正執行的是_ch3_02_01__jsp.class類別檔案。

3-2-5 Java語言的命名與註解-命名

- Java的命名需要遵循程式語言的語法
  - 名稱是一個合法的「識別字」（Identifier）
    - 使用英文字母開頭
      - 不限長度的Unicode統一字碼字元的字串，包含字母、數字和底線”_”。
  - 名稱區分英文字母的大小寫
    - 例如：apple、Apple和APPLE屬於不同變數。
  - 名稱不能使用Java語法的「關鍵字」（Keyword）
    - 例如：保留字的布林值true或false和null。
  - 名稱在「範圍」（Scope）中必需是唯一的
    - 例如：在程式中可以使用相同的變數名稱，不過各變數名稱需要在不同的範圍。

3-2-5 Java語言的命名與註解-關鍵字



abstract	boolean	break	byte	case
catch	char	class	const	continue
default	do	double	else	extends
final	finally	float	for	goto
if	implements	import	instanceof	int
interface	long	native	new	package
private	protected	public	return	short
static	strictfp	super	switch	synchronized
this	throw	throws	transient	try
void	volatile	while		

3-2-5 Java語言的命名與註解-Java註解

- JSP程式的Scriptlets元素是Java語言的程式碼片斷，所以使用Java語言的註解
  - 以"//"符號開始的列，或放在程式列後的文字內容，如下所示：

```
<%
// 顯示訊息
out.print("目前的日期/時間: ");   // 顯示訊息
%>
```

- 如果註解文字不只一列，可以使用"/*"和"*/"符號標示註解文字，如下所示：

```
/* JSP程式：Ch3_2_5.jsp  */
```

3-2-5 Java語言的命名與註解-HTML註解

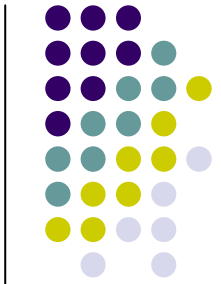
- "<!--"和"-->"是HTML標記語言的註解，如下所示：

```
<!-- JSP程式：Ch3_2_5.jsp -->
```

- 上述註解是位在JSP元素之外的HTML標籤，所以在HTML標記語言的註解也可以使用Expression元素，如下所示：

```
<! --目前時間：<%= new java.util.Date() %> -->
```

3-2-5 Java語言的命名與註解-JSP的基本輸出

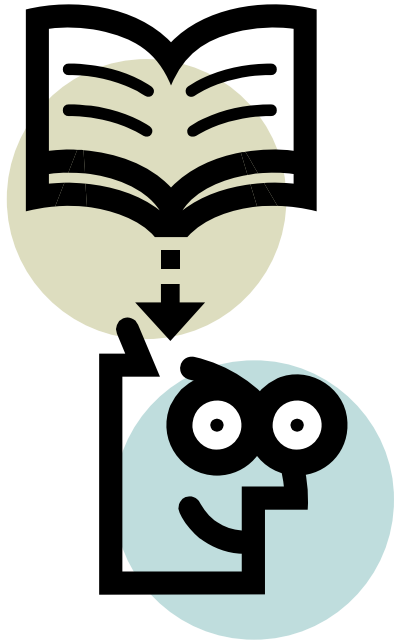


- JSP的基本輸出
  - 使用out隱含物件輸出網頁內容
  - 事實上，就是Java語言的System.out
- out物件:可以輸出字串
  - print()和println()方法，其差異在於是否在最後加上換行符號'\n'，如下所示：

```
out.print("目前的日期/時間: ");
out.println("目前的日期/時間: ");
```

3-3 Java的變數與資料型態

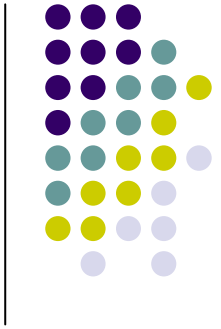
- 3-3-1 變數的資料型態
- 3-3-2 變數、常數宣告與指定敘述
- 3-3-3 Escape逸出字元



3-3-1 變數的資料型態-說明

- 分爲「基本」（Primitive）和「參考」（Reference）兩種資料型態
  - 基本資料型態
    - Java變數擁有byte、short、int、long、float、double、char和boolean一共8種資料型態。
  - 參考資料型態
    - Java的類別就是一種參考資料型態，其建立的物件變數值是記憶體位置，參考到物件儲存的位置
    - 例如：String、StringBuffer、Character和Number類別建立的物件就屬於參考資料型態。

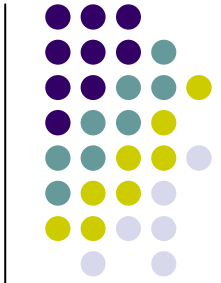
3-3-1 變數的資料型態-整數資料型態



- 「整數資料型態」（Integral Types）
  - 指變數的資料爲整數沒有小數點
  - 依照整數資料長度的不同（即變數佔用的記憶體位元數），分爲4種整數資料型態，如下表所示：

整數資料型態	位元數	範圍
byte	8	$-2^7 \sim 2^7-1$，即-128 ~ 127
short	16	$-2^{15} \sim 2^{15}-1$，即-32768 ~ 32767
int	32	$-2^{31} \sim 2^{31}-1$，即-2147483648 ~ 2147483647
long	64	$-2^{63} \sim 2^{63}-1$，即-9223372036854775808 ~ 9223372036854775807

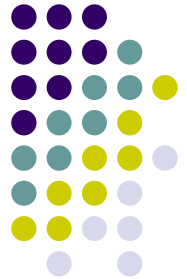
3-3-1 變數的資料型態-浮點數資料型態



- 「浮點數資料型態」（Floating Point Types）
  - 指整數加上小數
    - 例如：3.1415926、123.567等
  - 依照長度的不同（即變數佔用的記憶體位元數），分爲2種浮點數的資料型態，如下表所示：

浮點數資料型態	位元數	範圍
float	32	1.40239846e-45 ~ 3.40282347e38
double	64	4.94065645841246544e-324 ~ 1.79769313486231570e308

3-3-1 變數的資料型態-布林資料型態



- 「布林資料型態」（Boolean Type）
  - 只能有2個值true和false
  - 這不是變數名稱，而是Java的保留字
  - 主要是使用在條件和迴圈控制的條件判斷

3-3-1 變數的資料型態-字元資料型態

- 「字元資料型態」（Char Type）
  - 「無符號」（Unsigned）的16位元整數所表示的Unicode字元
  - Unicode字元是使用2個位元組表示字元，可以用來取代ASCII字元使用一個位元組的表示方式。

3-3-2 變數、常數宣告與指定敘述-說明

- Java語言是「強調型態」（Strongly Typed）程式語言，在變數儲存的值需要指定資料型態。
- 簡單的說
  - 資料型態的目的是告訴編譯程式宣告的變數準備儲存什麼樣的資料
  - 且不論如何存取變數值，都不能更改變數的資料型態。

3-3-2 變數、常數宣告與指定敘述-變數的宣告

- Java整數變數宣告的範例，如下所示：

```
int score;
```

  - 上述程式碼宣告1個整數變數，資料型態爲整數int，名稱爲score，如果需要同時宣告多個變數，請使用","號分隔，如下所示：

```
int i, j, k;
```

- 變數宣告單純只是表示保留記憶體空間，至於儲存的資料是什麼？可以在宣告時指定初始值。在Java宣告變數且指定變數初值，如下所示：

```
int score = 85;
int i = 20;
```

3-3-2 變數、常數宣告與指定敘述-常數的宣告

- 「常數」（Named Constants）
  - 是指一個變數在設定初始值後，就不會變更其值，簡單的說，就是在程式中使用一個名稱代表一個固定值。
- Java的常數宣告和指定初值的變數宣告相同，只需在前面使用final關鍵字，如下所示：

```
final double PI = 3.1415926;
```

- 上述程式碼宣告圓周率的常數PI。請注意！在宣告常數時一定要指定常數值。

3-3-2 變數、常數宣告與指定敘述-指定敘述

- 「指定敘述」（Assignment Statement）
  - 使用指定敘述即“=”等號，指定變數值或更改變數值。
  - 一個指定敘述的範例，如下所示：

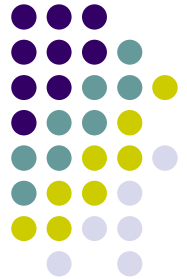
    balance = 6000;
  - 上述程式碼指定變數balance的內容，運算式是數值6000

3-3-3 Escape逸出字元-說明1

- Java提供Escape逸出字元，這是使用"\"符號開頭的字串，可以顯示一些無法使用鍵盤輸入的特殊字元，如下表所示：

Escape 逸出字元	Unicode 碼	說明
\b	\u0008	Backspace，Backspace 鍵
\f	\u000C	FF，Form feed 換頁符號
\n	\u000A	LF，Line feed 換行符號
\r	\u000D	CR，Enter 鍵
\t	\u0009	Tab 鍵，定位符號
\'	\u0027	「'」單引號
\"	\u0022	「"」雙引號
\\	\u005C	"\"符號

3-3-3 Escape逸出字元-說明2

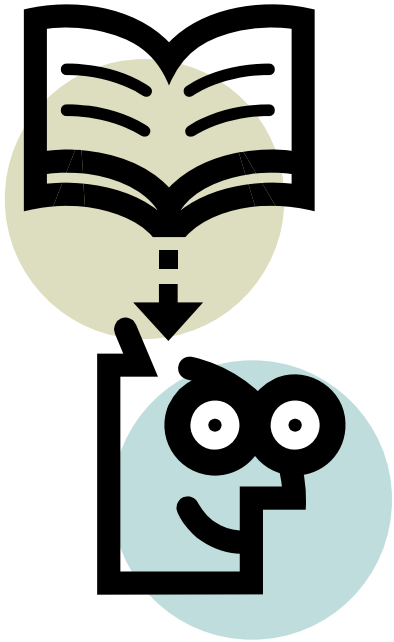


- 對於JSP標籤來說，如果在JSP程式需要顯示"<%"和"%>"標籤，我們需要使用替代字串或Escape逸出字元，如下表所示：

替代子串	JSP 標籤
<%	<%
%\>	%>

3-4 Java的運算子

- 3-4-1 運算子的優先順序
- 3-4-2 算術運算子
- 3-4-3 關係與條件運算子
- 3-4-4 位元運算子
- 3-4-5 指定運算子



3-4 Java的運算子

- Java指定敘述的左邊是「運算式」（Expressions），這是由「運算子」（Operator）和「運算元」（Operand）所組成，Java擁有完整的算術、指定、位元和邏輯運算子，一些運算式的範例，如下所示：

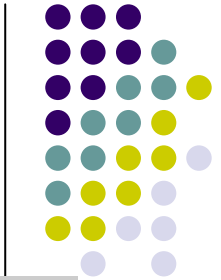
  A + B – 1

  A >= B

  A > B && A > 1

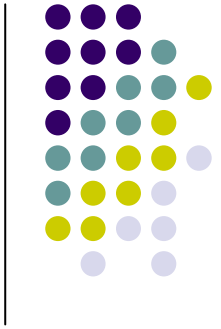
- 上述運算式變數A、B和數值1都屬於運算元，"+"、"-"、">="、">"和"&&"爲運算子，Java運算子是使用1到3個字元所組成的符號。

3-4-1 運算子的優先順序



運算子	說明
()	括號
!、-、++、--	條件運算子 NOT、算數運算子負號、遞增和遞減
*、/、%	算術運算子的乘、除法和餘數
+、-	算術運算子加和減法
<<、>>、>>>	位元運算子左移、右移和無符號右移
>、>=、<、<=	關係運算子大於、大於等於、小於和小於等於
==、!=	關係運算子等於和不等於
&	位元運算子 AND
^	位元運算子 XOR
|	位元運算子 OR
&&	條件運算子 AND
||	條件運算子 OR
?:	條件控制運算子
=、op=	指定運算子

3-4-2 算術運算子



- Java的「算術運算子」（Arithmetic Operators）
  - 是常用的數學運算子，大部分運算元都是數值，加法運算子還可以連結2個字串
  - 在下表爲各種運算子的範例，變數a的值爲10，如下表所示：

運算子	說明	運算式範例
-	負號	-7
++	遞增運算	a++ = 11
--	遞減運算	a-- = 9
*	乘法	5 * 6 = 30
/	除法	7.0 / 2.0 = 3.5
%	餘數	7 % 2 = 1
+	加法或字串連結	4 + 3 = 7
-	減法	4 – 3 = 1

3-4-3 關係與條件運算子-關係

- 「關係運算子」（Relational Operators）
  - 使用迴圈和條件敘述的判斷條件，可以比較2個運算元間的關係，例如："=="爲true表示運算元相等，關係運算子的範例，如下表所示：

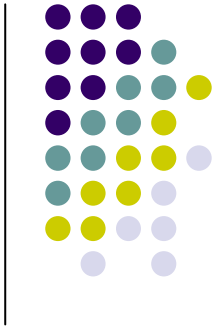
運算子	說明	運算式範例	結果
==	等於	7 ==5	false
!=	不等於	7 != 5	true
<	小於	7 < 5	false
>	大於	7 > 5	true
<=	小於等於	7 <= 5	false
>=	大於等於	7 >= 5	true

3-4-3 關係與條件運算子-條件

- 關係運算式如果比較複雜，不只一個關係運算式。
  - Java提供6種「條件運算子」（Conditional Operators）可以連結多個關係運算式建立更複雜的關係運算式，如下表所示：

運算子	範例	說明
!	! op	NOT 運算，傳回運算元相反的值，true 成 false，false 成 true
&&	op1 && op2	AND 運算，連結的 2 個運算元都爲 true，運算式爲 true
||	op1 || op2	OR 運算，連結的 2 個運算元，任一個爲 ture，運算式爲 true
&	op1 & op2	如果 op1 和 op2 爲布林資料型態的變數，此時的運算子如同 "&&"，如果運算元爲數字就是下一節的位元運算
|	op1 | op2	如果 op1 和 op2 爲布林資料型態的變數，此時的運算子如同 "||"，如果運算元爲數字就是下一節的位元運算
^	op1 ^ op2	XOR 運算，連結的 2 個運算元，只需任一個爲 true，結果爲 true，如果同爲 false 或 true 時結果爲 false

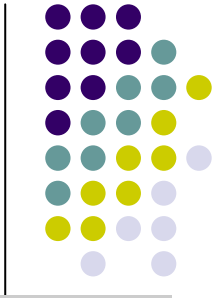
3-4-4 位元運算子



- Java的「位元運算子」（Shift and Bitwise Operators）
  - 進行二進位值的位元運算，提供向左移或右移幾個位元或NOT、AND、XOR和OR的位元運算，如下表所示：

運算子	A	B	C	D	範例	結果	說明
~	1(01)				~A	-2(10)	NOT 運算
<<			3(11)		C<< 2	12(1100)	左移運算
>>		2(10)			B >> 1	1(1)	右移運算
>>>				16(1000)	D >>> 1	8(0100)	無符號右移
&	1(01)		3(11)		A & C	1(01)	AND 運算
^	1(01)	2(10)			A ^ B	3(11)	XOR 運算
|	1(01)	2(10)			A | B	3(11)	OR 運算

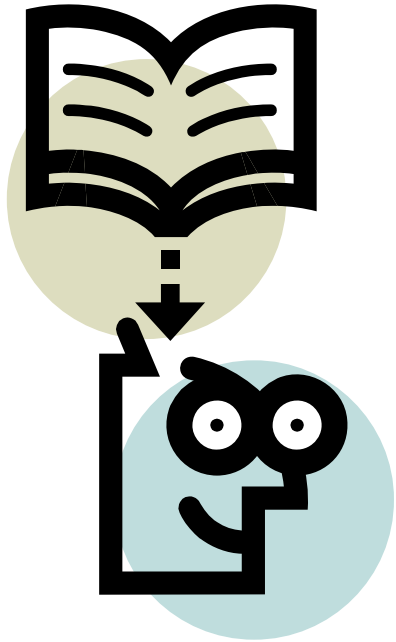
3-4-5 指定運算子



運算子	範例	相當的運算式	說明
=	x = y	N/A	指定敘述
+=	x+ = y	x = x + y	數字相加或字串連結
-=	x -= y	x = x - y	減法
*=	x *= y	x = x * y	乘法
/=	x /= y	x = x / y	除法
%=	x %= y	x = x % y	餘數
<<=	x <<= y	x = x << y	位元左移 y 位元
>>=	x >>= y	x = x >> y	位元右移 y 位元
>>>=	x >>>= y	x = x >>> y	無符號右移 y 位元
&=	x &= y	x = x & y	位元 AND 運算
|=	x |= y	x = x | y	位元 OR 運算
^=	x ^= y	x = x ^ y	位元 XOR 運算

3-5 Java的流程控制

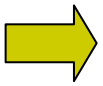
- 三種迴圈 (for)
  - 3-5-4 for迴圈敘述
  - 3-5-5 while與do/while迴圈敘述
    - while迴圈敘述
    - do/while迴圈敘述
  - 3-5-6 break與continue指令敘述
- 三種決策判斷 (if)
  - 3-5-1 是否選和二選一
  - 3-5-2 多選一的條件敘述
  - 3-5-3 ?:條件敘述運算子



3-5-4 for迴圈敘述

- Java的for迴圈稱爲「計數迴圈」（Counting Loop），這是一種簡化的while迴圈，可以重複執行固定次數的程式區塊。
  - for迴圈預設擁有一個計數器，計數器每次增加或減少一個值，直到for迴圈的結束條件成立爲止，
  - 例如：計算1加到10的總和，每次增加1，如下所示：

```
for ( i = 1; i <= 10; i++ ) {
   out.print("數值: " + i + “<br>”);
   total += i;
}
```



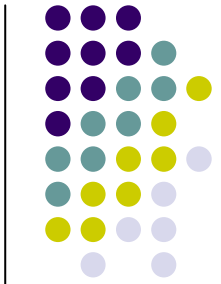
```
for ( 起始值;終止條件; 計數器 ) {
   內容敘述
}
```

An Example – For迴圈

- Example
  - 計算1加到10的總和，每次增加1，如下所示：

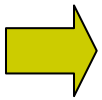
```
<html>
<body>
<%
    int total=0;
    for ( int i = 1; i <= 10; i++ ) {
            out.print("數值: " + i + “<br>”);
    total += i;}
    out.println(“total=”+ total);
%>
</body>
</html>
```

3-5-5 while與do/while迴圈敘述-while迴圈敘述



- while迴圈敘述
  - 不同於for迴圈需要自己處理計數器的增減
  - while迴圈是在進入程式區塊的開頭檢查結束條件，如果條件爲true才允許進入迴圈執行
  - 例如：使用while迴圈來計算1加到10的總和，如下所示：

```
int total = 0, counter=0;
while (counter <= 10 ) {
   total = total + counter;
   counter += 1;
}
```



```
int total = 0, counter=0;//起始值
while (終止條件) {
   total = total + counter;
   counter += 1;//計數器
}
```

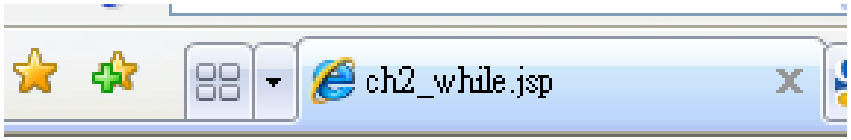
An Example - While迴圈

- Example
  - 計算1加到10的總和，每次增加1，如下所示：

```
<html>
<body>
<%
    int total=0, int counter=0;
    while (counter <= 10) {
            total = total + counter;
           counter += 1;
    }
    out.println(“total=”+ total);
%>
</body>
</html>
```

隨堂練習

- 請寫出右圖的jsp檔

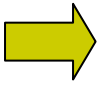


0 + 0 = 0
0 + 1 = 1
1 + 2 = 3
3 + 3 = 6
6 + 4 = 10
10 + 5 = 15
15 + 6 = 21
21 + 7 = 28
28 + 8 = 36
36 + 9 = 45
45 + 10 = 55

3-5-5 while與do/while迴圈敘述-do/while迴圈敘述

- do/while和while迴圈敘述的差異
  - 在迴圈的結尾檢查結束條件
  - 換句話說，do/while迴圈的程式區塊至少會執行一次
  - 例如：使用do/while迴圈來計算1加到10的總和，如下所示：

```
int total=0, counter = 0;
do {
    total += counter;
    counter+= 1;
} while ( counter <= 10 );
```



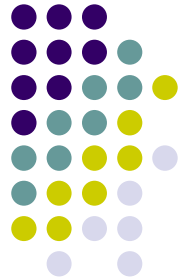
```
int total=0, counter = 0;//起始值
do {
    total += counter;
    counter+= 1;//計數器
} while (終止條件 );
```

An Example – do/while迴圈

- Example
  - 計算1加到10的總和，每次增加1，如下所示：

```
<html>
<body>
<%
    int total=0, counter = 0;
    do {
            total += counter;
            counter+= 1;
    } while ( counter <= 10 );
    out.println(“total=”+ total);
%>
</body>
</html>
```

3-5-6 break與continue指令敘述-break指令



- break指令
  - 可以在某些條件成立時，強迫終止迴圈執行
  - An Example:
    - 在do/while無窮迴圈配合break指令計算5! 階層函數值，也就是計算1*2*3*4*5的值，如下所示：

```
<%
    int result =0, i=0;
    do {
            out.print("數值: " + i + "<br>");
            result *= i;
            i++;
            if ( i > 5 ) break;
    } while ( true );
%>
```

3-5-6 break與continue指令敘述-continue指令

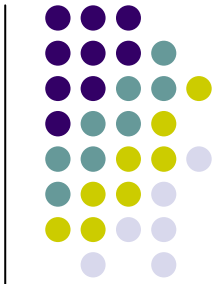
- continue指令敘述
  - 在迴圈執行過程中，可以馬上繼續執行下一次迴圈，不過它並不會執行程式區塊位在continue指令敘述後的程式碼
  - 如果使用在for迴圈，一樣會自動更新計數器變數。
- 例如：活用迴圈和continue指令敘述計算1~10的偶數和，如下所示：

```
<%
    int i, total=0;
    for ( i = 1; i <= 10; i++ ) {
            if ( (i % 2) != 0 ) continue;
            out.print("數值: " + i + "<br>");
            total += i;
            }
    out.println(“Total:”+ total);
%>
```

3-5 Java的流程控制

- Java的流程控制
  - For迴圈:重複執行指定區塊的程式碼
  - If判斷句: 配合條件判斷來執行不同區塊的程式碼
    - 條件控制是一個選擇題，可能是單一選擇或多選一，依照條件運算子的結果，決定執行哪一個區塊的程式碼。
    - 三種決策判斷 (if)
      - 3-5-1 是否選和二選一
      - 3-5-2 多選一的條件敘述
      - 3-5-3 ?:條件敘述運算子

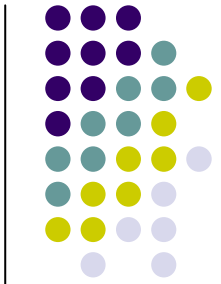
3-5-1 是否選和二選一-if是否選條件敘述



- if條件敘述
  - 一種是否執行的單選題，決定是否執行程式區塊內的程式碼
  - 如果關係和條件運算結果爲true，就執行括號間的程式區塊
- 例如：判斷成績是否及格的if條件敘述，如下所示：

```
if ( score >= 60 ) {
   out.print("JSP網頁製作");
   out.print("徹底研究-及格！<br>");
}
```

3-5-1 是否選和二選一-if/else二選一條件敘述



- If/else二選一條件敘述
  - 如果條件是擁有排它性的兩個執行區塊，需要二選一，只需加上else指令，if的關係/條件運算式爲true時執行到else前的程式碼，false執行else之後的程式碼
- 例如：判斷學生成績及格或不及格的if/else條件敘述，如下所示：

```
if ( score >= 60 ) {
   out.print("成績及格<br>");
}
else {
   out.print("成績不及格<br>");
}
```

3-5-2 多選一的條件敘述-if/else多選一條件敘述

- If/else多選一條件敘述
  - 在Java程式可以重複使用if/else條件建立多選一的條件敘述
- 例如：使用年齡判斷搭乘公車的乘客票價是學生、普通或敬老票，如下所示：

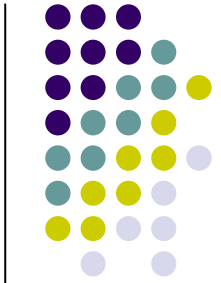
```
if ( age <= 18 ){
        out.print("學生票：12元<br>");}
else if ( age >= 65 ){
        out.print("敬老票：8元<br>");}
else{
        out.print("普通票：15元<br>");}
```

3-5-2 多選一的條件敘述-switch多選一條件敘述

- Switch多選一條件敘述
  - Java的switch多條件敘述是依照符合的條件執行不同程式區塊的程式碼
- 例如：學生成績是使用GPA的A、B、C、D來打成績，我們可以使用switch條件敘述來顯示轉換的成績範圍。

```
int grade = 1;
switch (grade) {
   case 1:
      out.print("學生成績超過80分<br>");
      break;
   case 2:
      out.print("學生成績爲70~79分<br>");
      break;
   default:
      out.println("學生成績低於60分<br>");
}
```

3-5-3 ?:條件敘述運算子



- Java的條件敘述運算子“?:”
  - 可以指定敘述以條件來指定變數值
- 例如：12小時與24小時制的時間轉換條件敘述運算子，如下所示：

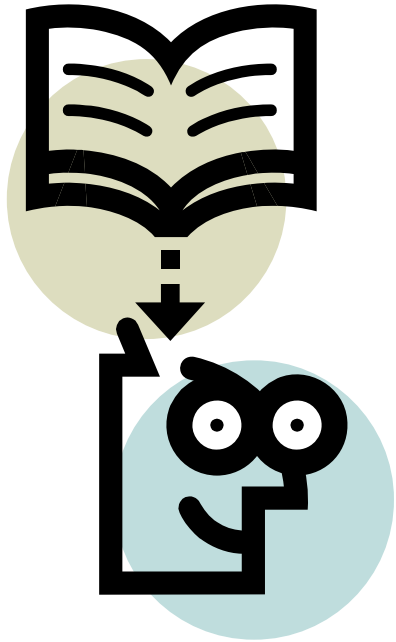
hour = (hour >= 12) ? hour-12 : hour;

如果”**(hour >= 12) ?**”成立 ➔ **hour = hour-12**

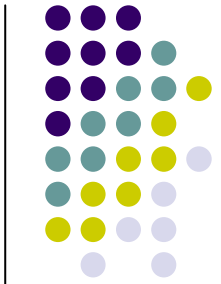
反之”**(hour >= 12) ?**”不成立 ➔ **hour = hour**

3-6 Java的陣列與字串

- 3-6-1 Java的陣列
- 3-6-2 字串處理



3-6-1 Java的陣列-一維陣列(宣告)



- 方式1: 宣告陣列變數時，同時指定陣列內容
  - E.g. 整數的一維陣列宣告，如下所示：

    int[] tips = {150, 300, 500};
  - 上述程式碼
    - 宣告int基本資料型態的陣列tips
    - int[]是一個類別，宣告陣列變數tips
    - 使用大括號指定陣列元素值所以並不用指定陣列大小，以此例一共有3個陣列元素
- 方式2: 使用new運算子建立Array物件
  - 例如：建立double資料型態的陣列儲存成績資料，如下所示：

    double[] scores = new double[6];

3-6-1 Java的陣列-一維陣列(存取)

- 取出陣列的值
  - E.g. 取出索引值爲1的陣列內容：

    scores[0] = 66.5;

- 走訪整個陣列內容
  - 將陣列索引設爲變數i，只需使用一個for迴圈，就可以顯示陣列的每一個元素，如下所示：

for (int i=0; i < scores.length; i++) { …… }

3-6-1 Java的陣列-多維陣列(宣告)

- 方式1: 宣告多維陣列變數時，同時指定陣列內容
  - E.g.

```
int[][] grades = { { 54, 68 },
                   { 67, 78 },
                   { 89, 93 } };
```

- 方式2: 使用new運算子建立二維陣列
  - E.g. 宣告3組String陣列, 每一陣列包含2個值

String[][] users = new String[3][2];

3-6-1 Java的陣列-多維陣列(存取)

- 指定二維陣列的元素值方法，如下所示：

```
users[0][0] = "Joe";
users[0][1] = "1234";
users[1][0] = "Jane";
users[1][1] = "5678";
users[2][0] = "Tony";
users[2][1] = "9012";
```

3-6-2 字串處理-建立字串

- Java字串
  - 就是String物件
  - 在Java程式使用String類別建立的字串物件屬於唯讀字串
  - 直接使用字串文字值（使用「"」雙引號括起）來建立字串物件，如下所示：

```
String str = "JSP網頁製作徹底研究";
```

- Java提供數種String物件的建構子，可以使用new運算子建立String物件，如下所示：

```
str1 = new String();     // 空字串
str2 = new String("Java 2程式設計範例教本2e");
str3 = new String(str2); // 使用字串物件
str4 = new String(charArr);    // 使用字元陣列
str5 = new String(charArr, 0, 4);
str6 = new String(byteArr);    // 使用byte陣列
```

3-6-2 字串處理-
字串長度與大小寫轉換

- String物件提供方法取得字串長度和進行英文字串的大小寫轉換，相關String物件的方法，如下表所示：

方法	說明
int length()	取得字串長度，傳回字串擁有多少個字元或中文字
String toLowerCase()	將字串的英文字母轉換成小寫字母
String toUpperCase()	將字串的英文字母轉換成大寫字母

An Example

- 字串長度與大小寫轉換

```
String st = "JSP Programming";
out.println(st.length()+"<br>");
out.println(st.toLowerCase()+"<br>");
out.println(st.toUpperCase()+"<br>");
```

st.length() **st.toLowerCase()** **st.toUpperCase()**



15 **“jsp programming”** **“JSP PROGRAMMING”**

3-6-2 字串處理-子字串搜尋

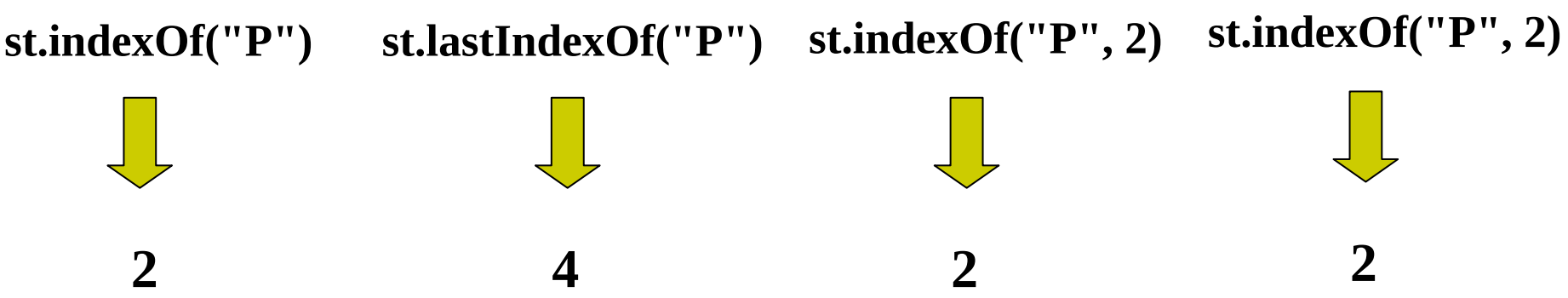
- String物件相關子字串搜尋方法，如下表：

方法	說明
int indexOf(String)	傳回第 1 次搜尋到字串的索引位置，如果沒有找到傳回-1
int lastIndexOf(String)	傳回反向從最後 1 個字元開始搜尋到字串的索引位置，如果沒有找到傳回-1
int indexOf(String, int)	傳回第 1 次搜尋到字串的索引位置，如果沒有找到傳回-1，傳入的參數 String 爲搜尋的字串，int 爲開始搜尋的索引位置
int lastIndexOf(String, int)	如同上一個 indexOf()方法，不過是從尾搜尋到頭的反向搜尋

An Example

- 子字串搜尋

```
String st = "JSP Programming";
out.println(st.indexOf("P")+"<br>");
out.println(st.lastIndexOf("P")+"<br>");
out.println(st.indexOf("P", 2)+"<br>");
out.println(st.lastIndexOf("P", 2)+"<br>");
```



3-6-2 字串處理-
子字串和字元處理

- String物件提供方法取代和取出字串中所需的字元和子字串，相關方法如下表所示：

方法	說明
char charAt(int)	取得參數 int 索引位置的字元
String substring(int)	從參數 int 開始取出剩下字元的字串
String substring(int, int)	取出第 1 個參數 int 到第 2 個參數 int 間的子字串
String replace(char, char)	將字串中找到的第 1 個參數 char 取代成爲第 2 個參數 char
String concat(String)	將參數的 String 字串新增到 String 物件的字串之後
String trim()	刪除字串前後的空白字元

An Example

- 子字串和字元處理

```
String st = "JSP Programming";
out.println(st.substring(0, 3)+"<br>");
out.println(st.replace('P', 'J')+"<br>");
out.println(st.trim()+"<br>");
```

st.substring(0, 3)

JSP

st.replace('P', 'J')

JSJ Jrogramming

st.trim()

JSPProgramming

3-6-2 字串處理-字串的比較

- String物件的字串可以一個字元一個字元比較字元的內碼值，直到分出大小為止。其相關方法如下表所示：

方法	說明
int compareTo(String)	比較 2 個字串內容，傳回值是整數，0 表示相等，<0 表示參數的字串比較大，>0 表示參數的字串比較小
int compareToIgnoreCase(String)	忽略大小寫，比較 2 個字串的內容
boolean equals(Object)	比較 2 個字串是否相等，傳回值 true 表示相等，flase 表示不相等，參數不一定是字串物件，也可以使用在其它物件
boolean equalsIgnorCase(String)	忽略大小寫，比較 2 個字串內容是否相等

An Example

- 字串的比較

```
String st1 = "ab", st2 = "ac";
out.println(st1.compareTo(st2)+"<br>");
out.println(st1.equals(st2)+"<br>");
```

st1.compareTo(st2)

-1: 表示st1比較大

st1.equals(st2)

false

隨堂練習

- 使用年齡判斷搭乘公車的乘客票價是學生、普通或敬老票，用for、if語法撰寫下列程式
- 18歲(含)：學生票
- 18-65歲(含)：普通票
- 65歲以上：敬老票
- 變數宣告 int[] age = {12,45,77,60,33};

12歲	45歲	77歲	60歲	33歲
學生票：12元	普通票：15元	敬老票：8元	普通票：15元	普通票：15元