第 3 章

PHP 基本語法

3-1 在瀏覽器畫面顯示訊息

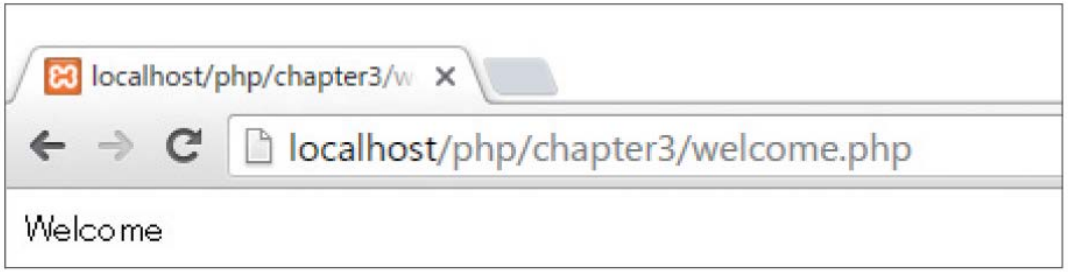
- 在瀏覽器上顯示文字訊息
  - 請利用 XAMPP 控制面板啟動 Apache，並與 Chapter2 的說明一樣，在瀏覽器開啟下列 URL。
  - 執行 http://localhost/php/chapter3/welcome.php

List chapter3 \ welcome.php PHP

```
<?php
echo 'Welcome';
?>
```

- 若程式正確執行，則瀏覽器將顯示「Welcome」

瀏覽器上顯示的訊息



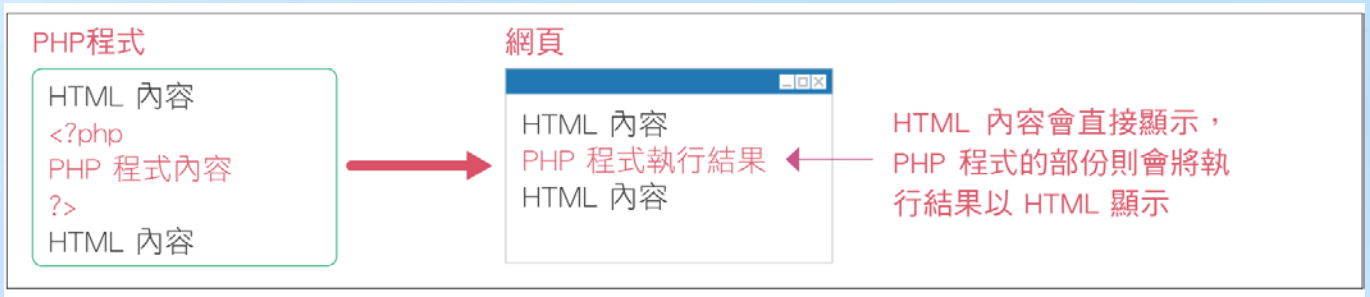
- 「<?php」與「?>」
  - PHP 程式必須以**「<?php」開始，以「?>」結束**

語法 PHP 標籤與程式碼

```
<?php
PHP程式內容
?>
```

–以「<?php」與「?>」將程式碼特別標示出來的方式，就能起很大的作用

–下列是 HTML 與 PHP 混合在同一檔案時的例子



- echo
  - echo 是 PHP 中用來顯示文字訊息的指令

```
echo 'Welcome';
```

  - echo 的使用方法如下

語法 echo

```
echo '文字訊息';
```

- 字串與單引號（'）
  - 「'」稱為單引號（single quote 或 single quotation），用於顯示文字訊息的程式中
  - 可用「"」雙引號代替「'」，以 "Welcome" 的形式表示字串

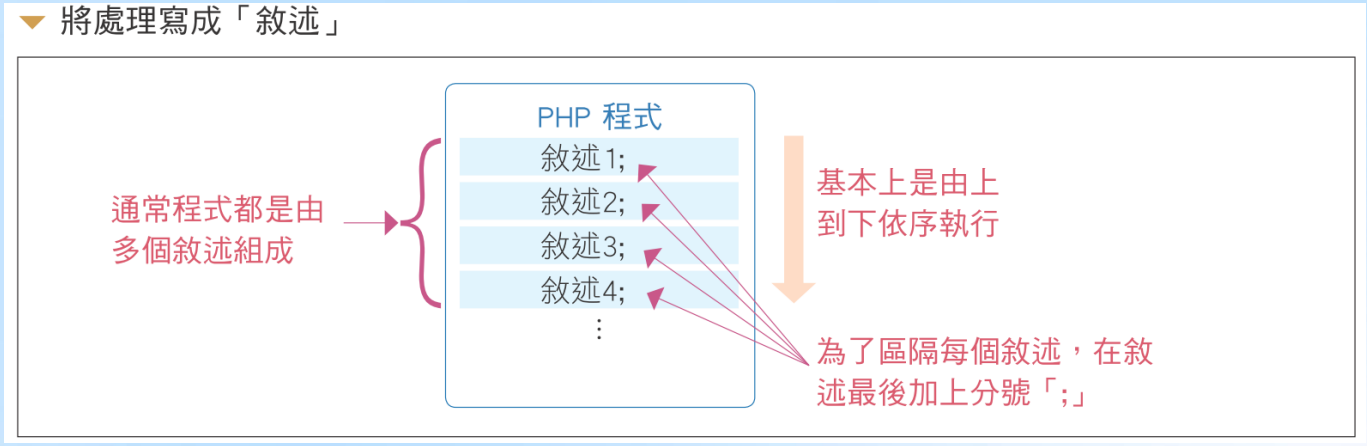
List welcome2.php PHP

```php
<?php
echo "Welcome";
?>
```

- 敘述與分號「;」
  - –在每行程式的最後，都會有一個分號「;」

```
echo 'Welcome';
```

  - –分號是用來表示一個敘述結束的符號，echo 'Welcome'; 就算是一個敘述 (Statement)。



- 出現錯誤時
  - 錯誤（Error）是指程式的語法有誤
  - 錯誤訊息（Error Message）則是系統為了告知有錯誤發生而顯示的訊息

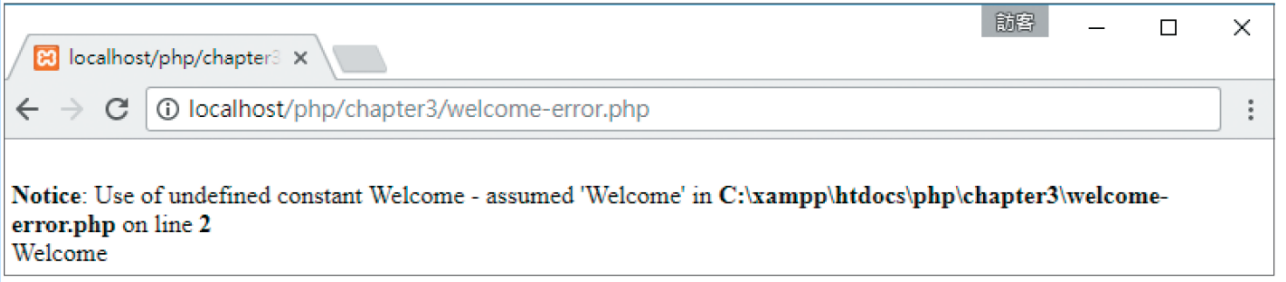
List welcome-error.php

PHP

```php
<?php
echo Welcome;
?>
```

前後刻意不加' '

錯誤訊息



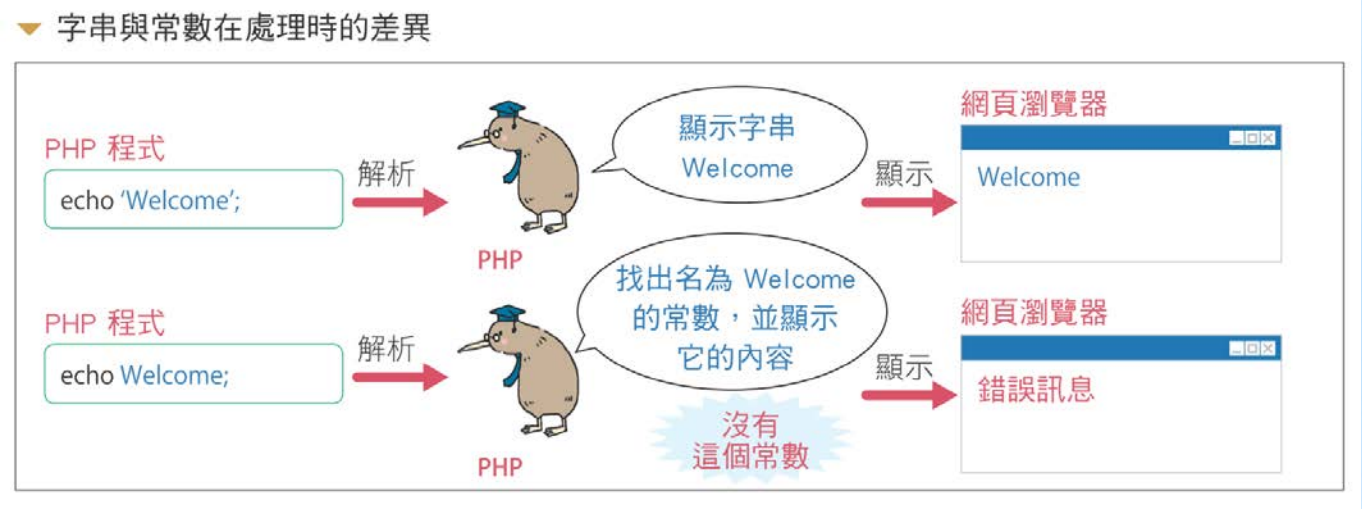
- 解讀錯誤訊息
  - –本例的錯誤訊息如下

```
Notice: Use of undefined constant Welcome - assumed 'Welcome' in C:\xampp\
htdocs\php\chapter3\welcome-error.php on line 2

Welcome
```

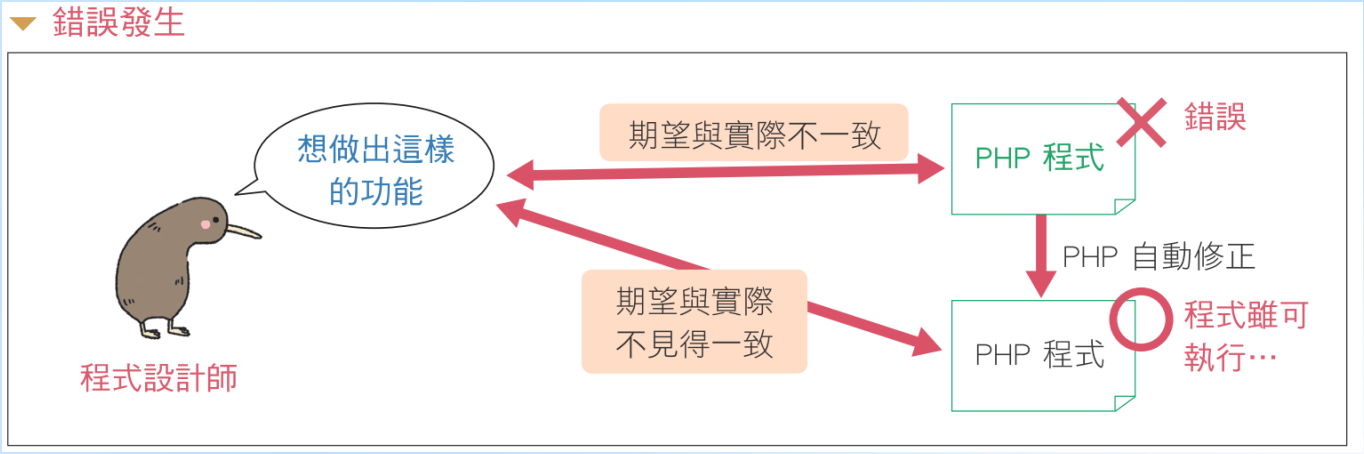
  - –這段訊息的意思如下

```
提示：使用未定義常數 Welcome - 推測應為字串 'Welcome'

錯誤發生於 C:\xampp\htdocs\php\chapter3\welcome-error.php 第 2 行

Welcome
```

–常數 (Constant) 後面會介紹到，是程式中來方便處理字串或數值的機制



- 依提示修正程式
  - –在錯誤訊息出現時，應好好利用其中所提到的下列三點資訊
    - 錯誤發生位置（在檔案中的哪一行）
    - 錯誤發生原因
    - 錯誤的解決方法



- 改用 print 顯示文字訊息

List welcome3.php PHP

```php
<?php
print 'Welcome';
?>
```

- Print
  - –print 的使用方法如下

語法 print

```
print '文字訊息';
```

- echo 與 print 都可用來顯示文字訊息，本書主要使用 echo。原因如下：
  - –echo 的處理速度較快，不返回任何值。
  - –echo 具有可將多個字串、數值連接起來顯示的功能
  - –print 只能輸出一個字串，並始終返回 1

PHP 5 echo 和 print 語句

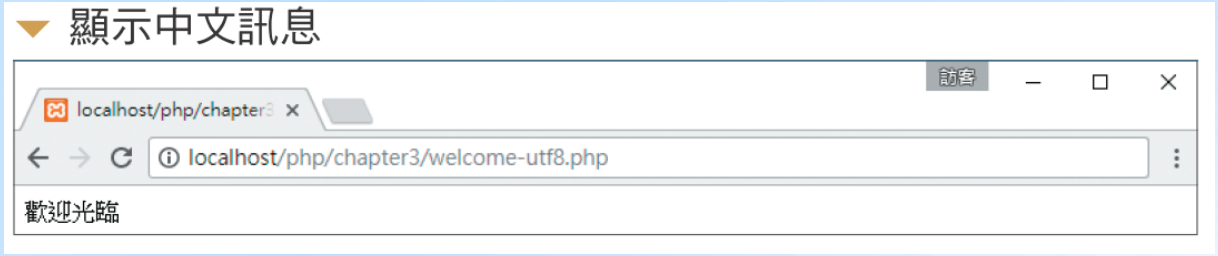
3-2 顯示中文訊息

- 撰寫顯示中文訊息的程式
  - 在儲存檔案時，必須將文字編碼設定為 UTF-8

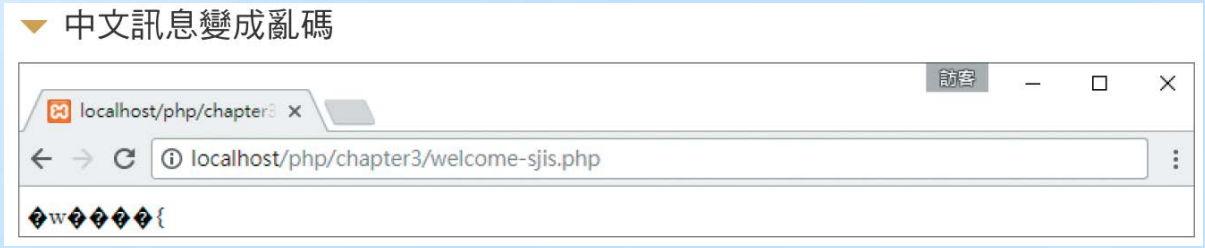
List welcome-utf8.php PHP

```php
<?php
echo '歡迎光臨';
?>
```

  - 程式若正確執行，則會顯示出「歡迎光臨」字樣



- 解決可能顯示亂碼的問題
  - 存檔時將文字編碼刻意設定 UTF8 以外的項目，例如 ANSI
  - 原本應該顯示「歡迎光臨」的地方變成了亂碼



解說

- 文字編碼
  - 顯示結果有誤的原因，就在於文字編碼。文字編碼其實就是電腦內部用來決定處理文字的方式
  - 以 ANSI 編碼的範例程式，就會因為用了錯誤的編碼去解析文字，使得文字變成亂碼
  - 無論是哪一個程式，都有可能因為瀏覽器的設定導致文字無法正確顯示

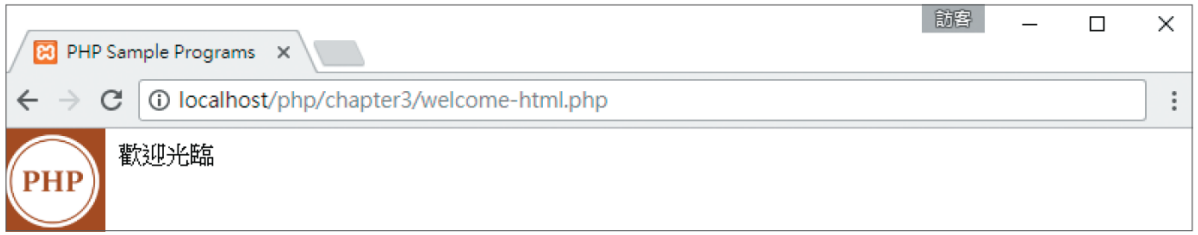
- 以 HTML 顯示訊息
  - 撰寫以 HTML 形式顯示訊息的 PHP 程式，並將檔案儲存為
  - chapter3\welcome-html.php
  - 除非有特別指定其它編碼，否則所有範例程式都以 UTF-8 的編碼方式儲存

List welcome-html.php PHP

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>PHP Sample Programs</title>
<link rel="stylesheet" href="../style.css">
<link rel="stylesheet" href="style.css">
</head>
<body>
<?php
echo '歡迎光臨';
?>
</body>
</html>
```

–在 HTML 的 <body> 標籤之間加入 PHP 程式的執行結果

在 HTML 指定文字編碼後顯示的訊息



解說

- HTML 部份的涵義

敘述	意義
<!DOCTYPE html>	表示檔案類型為 HTML5
<html>	HTML 程式開始
<head>	定義網頁的標題或其它連結資訊的區塊，從這裡開始
<meta charset="UTF-8">	指定文字編碼為 UTF-8
<title>PHP Sample Programs</title>	指定網頁標題
<link rel="stylesheet" href="../style.css">	載入樣式表，讓網頁更美觀
<link rel="stylesheet" href="style.css">	各 Chapter 若要加載額外的樣式表，則利用這行載入
</head>	定義網頁的標題或其它連結資訊的區塊，在這裡結束
<body>	瀏覽器要顯示的內容區塊從這裡開始

header.php

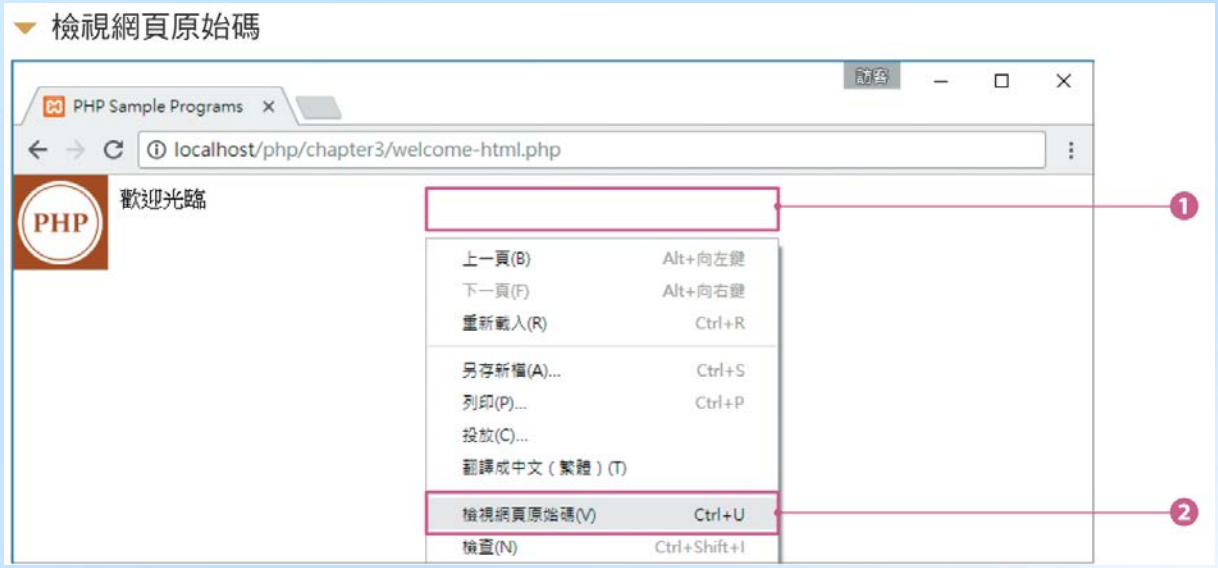
敘述	意義
</body>	瀏覽器要顯示的內容區塊在這裡結束
</html>	HTML程式結束

footer.php

–上列敘述中，最重要的是，將文字編碼指定為 UTF-8

```
<meta charset="UTF-8">
```

- 關於顯示出來的 HTML 程式
  - –在程式中，只有「歡迎光臨」字樣是 PHP 程式處理顯示的結果
  - –其它的 HTML 程式內容，都在 PHP 標籤之外



–開啟原始碼內容如下

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>PHP Sample Programs</title>
<link rel="stylesheet" href="../style.css">
<link rel="stylesheet" href="style.css">
</head>
<body>
歡迎光臨</body>
</html>
```

HTML 與 PHP 的用途區分

- HTML 區塊 → 用於顯示固定不變的內容
- PHP 區塊 → 用於顯示會因情況而變動的內容

echo 標籤

- 在程式中，像「**<?php** echo ' 歡迎光臨 '; **?>** 」這樣只有 1 個 echo 指令的PHP 區塊，可將 echo 拿掉，簡化為**<?=** '歡迎光臨'; **?>**

3-3在畫面上顯示使用者輸入的資料

- 製作輸入用的文字欄位
  - –參照下列程式，製作讓使用者輸入姓名的網頁，並將檔案儲存為 chapter3\user-input.php

List user-input.php PHP

```php
<?php require '../header.php'; ?>
<p>請輸入姓名：</p>
<form action="user-output.php" method="post">
<input type="text" name="user">
<input type="submit" value="確定">
</form>
<?php require '../footer.php'; ?>
```

使用者輸入資料的頁面

解說

- **重複使用**的部份統整在單獨的檔案(require 敘述)
  - 節省反複輸入重複內容的時間，**讓程式看起來更簡潔**。
  - 要修改共通使用的部份時，不需在多支程式裡分別修改，**只要修改單一檔案的內容即可**。
  - 在 PHP 中，要載入並執行放在其它檔案中的程式，必須使用 require 敘述。

語法 require

```
require '檔案名稱';
```

```
<?php require '../header.php'; ?>
```

include 和 require 語句用於在執行中插入寫在其他檔中的有用的代碼。
它們除了處理錯誤的方式不同之外，在其他方面都是相同的：

- **require 生成一個致命錯誤（E_COMPILE_ERROR），在錯誤發生後程式會停止執行。**
- **include 生成一個警告（E_WARNING），在錯誤發生後程式會繼續執行。**

–header.php，它儲存了 HTML 程式的開頭部份，檔案內容如下所示

List header.php PHP

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>PHP Sample Programs</title>
<link rel="stylesheet" href="../style.css">
<link rel="stylesheet" href="style.css">
</head>
<body>
```

–footer.php，它儲存了 HTML 程式的結尾部份，檔案內容如下所示

List footer.php PHP

```
</body>
</html>
```

顯示輸入用表單

–輸入資料到顯示結果的流程

–在程式中，下列區塊的內容即是用來製作表單

```
<form action="user-output.php" method="post">
...
</form>
```

– <form> 標籤用來表示輸入用表單由此開始，</form> 則為表單結束

利用 <form> 標籤製作輸入用表單

```
<form action="user-output.php" method="post">
<input type="text" name="user">          ← 文字欄位
<input type="submit" value="確定">       ← 按鈕
</form>
```

–文字欄位

- 利用 <input> 標籤，建立輸入文字用的控制元件「文字欄位（Text Box）
- 將 type 屬性指定為 text，就可產生文字欄位

```
<input type="text" name="user">
```

至於這個的意義，後面 3-26 頁會有說明

–按鈕

- type 屬性需指定為 submit，就可產生將表單內容傳送給網站伺服器的確定 (或送出) 鈕
- 在 value 屬性所設定的值，**會顯示在按鈕上**

```
<input type="submit" value="確定">
```

- 從文字欄位取得資料
  - 可從文字欄位取得字串後，將字串內容顯示在瀏覽器的程式

List user-output.php PHP

```php
<?php require '../header.php'; ?>
<?php
echo '午安，', $_REQUEST['user'], '您好。';
?>
<?php require '../footer.php'; ?>
```

取得 user 物件的內容

  - 將輸入的名字與文字訊息一起顯示

解說

- 送出表單時進行的處理
  - –在表單上按下按鈕，就可將控制元件的值傳送到處理輸出的程式
  - –以程式中的<from> 標籤為例說明如下

```
<form action="user-output.php" method="post">
```

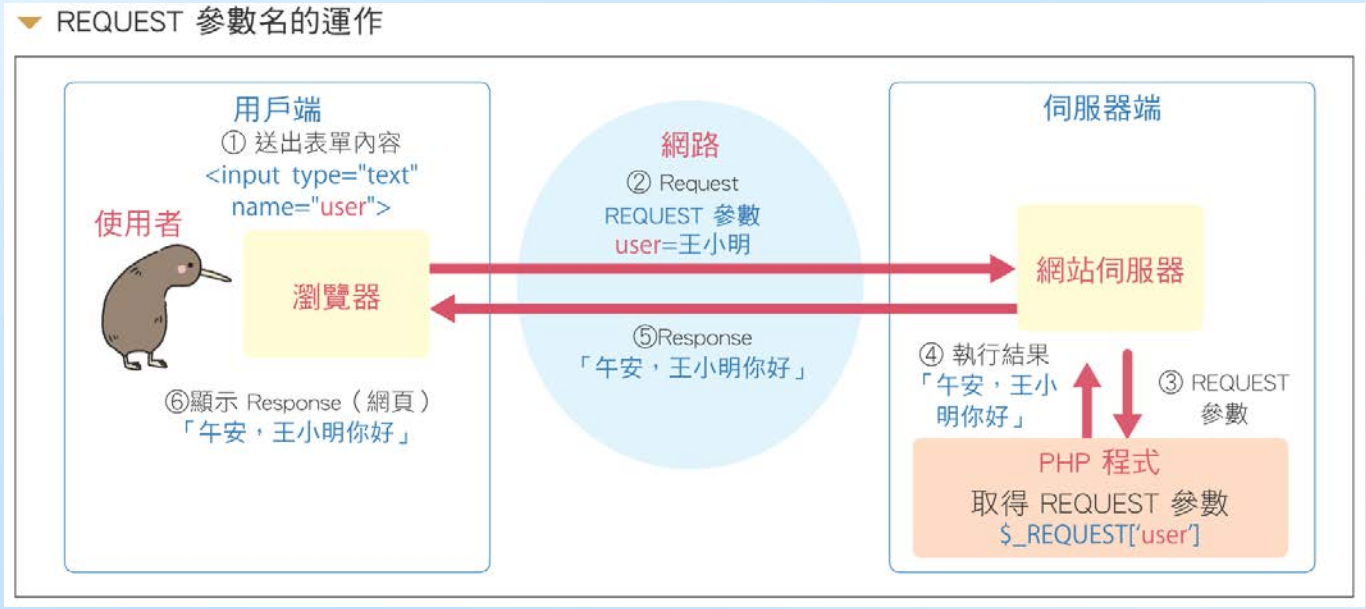
- REQUEST 參數
  - –以下是程式中，關於文字欄位的 <input> 標籤

```
<input type="text" name="user">
```

  - –它的 name 屬性指定為「user」**(或視為物件的名稱)**

```
echo '午安、', $_REQUEST['user'], '您好。';
```

–一個 Request 有可能伴隨著多個 REQUEST 參數，例如一個表單中除了輸入姓名，可能還有住址、身分證字號等資料。



- 取得及顯示 REQUEST 參數值
  - 要在程式中取得 REQUEST 參數，語法如下

語法 取得 REQUEST 參數

```
$_REQUEST['REQUEST參數名']
```

  - 如此即可取得指定名稱的 REQUEST 參數值

```
$_REQUEST['user']
```

  - 加上 echo 指令撰寫成如下的程式，就可將取得的字串內容「王小明」

```
echo $_REQUEST['user'];
```

– 串連顯示值

值	類型
'午安，'	字串
$_REQUEST['user']	REQUEST 參數（字串）
'您好。'	字串

```
echo '午安，', $_REQUEST['user'], '您好。';
```

- $_REQUEST
  - $_REQUEST 是用來取得 REQUEST 參數的工具，REQUEST 參數值會暫存於 $_REQUEST 內。

如果執行 **user-output.php** 時，當 **<input type="text" name="user">** 未被定義時，直接按 **[確定]**，會產生甚麼結果？要如何便免錯誤發生時，能夠提示程式設計者呢？

先檢查**"user"**是否有被定義，如果沒有，顯示提示訊息！

優化程式的方法


```php
<?php require '../header.php'; ?>
<?php
if (isset($_REQUEST['user'])) {
    echo '午安，', $_REQUEST['user'], '您好。';
}
?>
<?php require '../footer.php'; ?>
```

if 是一種控制流程分歧的條件式，當條件成立時才會執行**{}** 中所有的程式，會在 **Chapter 4** 詳細介紹。

isset() 函式用來檢查變數是否已定義。**'user'** 的輸入物件若有填寫資料表示已被定義，否則表示沒有被定義。

3-4 以單價及個數計算總金額

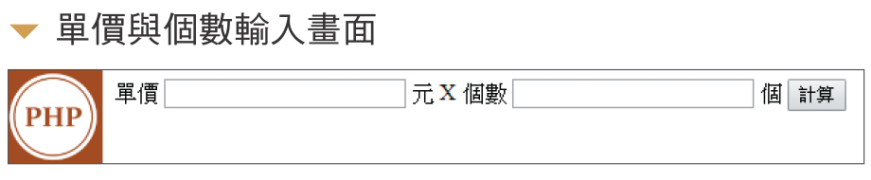
- 製作輸入單價與個數的畫面
  - 參照下列程式，製作輸入單價及個數的表單畫面，並將檔案儲存為chapter3\price-input.php

List price-input.php

PHP

```php
<?php require '../header.php';?>
<form action="price-output.php" method="post">
單價 <input type="text" name="price"> 元
×
個數 <input type="text" name="count"> 個
<input type="submit" value="計算">
</form>
<?php require '../footer.php';?>
```

–程式正常執行時，將會顯示出單價、個數的文字欄位以及「計算」按鈕



–程式中利用 <input> 標籤，製作 2 個文字欄位。程式碼如下

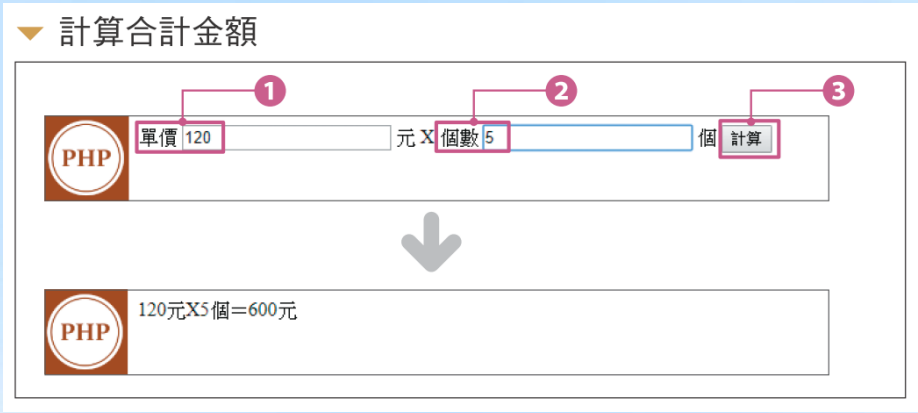
```
<input type="text" name="price">
<input type="text" name="count">
```

運用算符計算合計金額

–參照下列程式，製作從輸入畫面取得單價與個數並計算合計金額的程式

List price-output.php PHP

```php
<?php require '../header.php';?>
<?php
echo $_REQUEST['price'], '元X';
echo $_REQUEST['count'], '個=';
echo $_REQUEST['price'] * $_REQUEST['count'], '元';
?>
<?php require '../footer.php';?>
```



解說

- 取得 REQUEST 參數值與計算
  - 輸入單價與個數的文字欄位，REQUEST 參數名分別為 price 與 count

```
$_REQUEST['price']
$_REQUEST['count']
```

- 算符 (或稱運算子)
  - 範例將單價與個數相乘計算出合計金額。要在程式中進行乘除等運算時必須利用算符 (Operator)。

```
$_REQUEST['price'] * $_REQUEST['count']
```

  - 利用 echo 在計算出來的合計金額後面加上「元」字後，顯示在畫面上

```
echo $_REQUEST['price']*$_REQUEST['count'], '元';
```

–常用的算符

算符	作用
**	平方
++ --	加1、減1
!	邏輯（反值）
* / %	乘法、除法、餘數
+ - .	加法、減法、字串相連
< <= > >=	比較（小於、小於等於、大於、大於等於）
== !=	比較（等於、不等於）
&&	邏輯（AND）
||	邏輯（OR）
=	指派

- 利用變數存放數值，取出再計算
  - 變數是用來存放資料的機制

List price-output2.php

```php
<?php require '../header.php';?>
<?php
$price=$_REQUEST['price'];
$count=$_REQUEST['count'];
echo $price, '元X';
echo $count, '個=';
echo $price*$count, '元';
?>
<?php require '../footer.php';?>
```

–接著修改輸入畫面的程式，將表單送出後執行的程式改為 price-output2.php

List price-input2.php

PHP

```php
<?php require '../header.php';?>
<form action="price-output2.php" method="post">
單價 <input type="text" name="price"> 元
X
個數 <input type="text" name="count"> 個
<input type="submit" value="計算">
</form>
<?php require '../footer.php';?>
```

解說

- 變數命名規則
  - –①變數名稱的前面必須加上錢字號（$）。
  - –②開頭第 1 個字必須為英文字母或底線（_）。
  - –③除了第 1 個字之外，其它可用英文字母、數字、底線隨意組成。
  - –④英文字母的大小寫視為不同文字。

正確的變數命名：
$price、$price123、$price_tag

錯誤的變數命名：
$123price

不同的變數：
$price 與 $Price 不相同

PHP 使用變數前不需宣告變數型別 (整數、浮點數)，系統會自行設定。

PHP 預先定義好的變數

- PHP 已預先定義的預定義變數中，摘錄與本書內容相關的變數如下

變數名稱	功能
$_REQUEST	HTTP 的所有 REQUEST 參數（無論是以 GET 或 POST 方式傳送）
$_GET	HTTP 以 GET 方式傳送的 REQUEST 參數
$_POST	HTTP 以 POST 方式傳送的 REQUEST 參數
$_FILES	上傳檔案的資料
$_SESSION	Session
$_COOKIE	Cookie

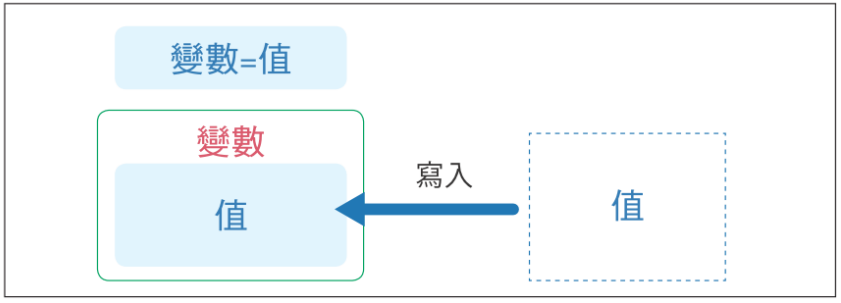
- 指派
  - 指派是將值放入變數之中

語法 指派

```
變數=值
```

  - 「＝」被稱為指派算符，可將它右邊的值寫入左邊的變數

將值指派給變數

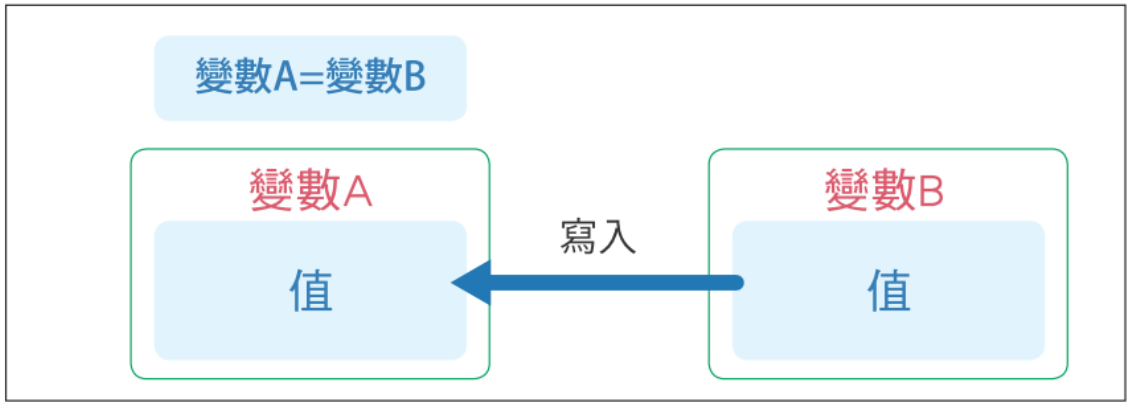


–變數也可指派給另一變數

語法 變數之間的指派

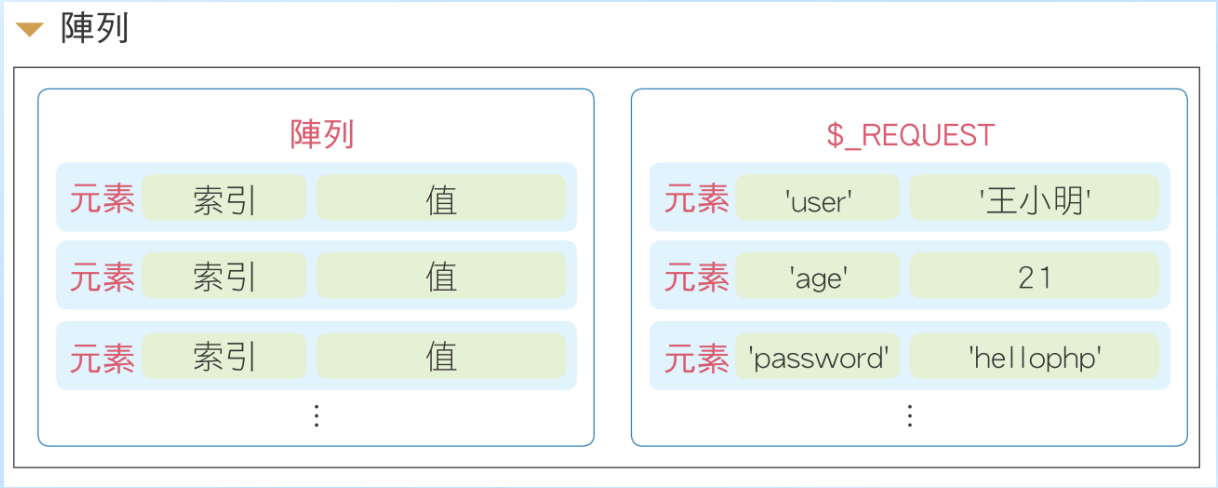
```
變數A=變數B
```

將變數指派給變數



陣列 (Array)

–一般變數只能儲存一個值，但陣列則可儲存多個值



- 陣列變數命名規則
  - ① 變數名稱的前面必須加上錢字號（$）。
  - ② 開頭第 1 個字必須為英文字母或底線（_）。
  - ③ 除了第 1 個字之外，其它可用英文字母、數字、底線隨意組成。
  - ④ 英文字母的大小寫視為不同文字。

```
$stock['apple']=30;
```

PHP 沒有創建變數的命令，變數會在首次為其賦值時被創建。

- 變數的使用
  - 單價及個數顯示在畫面上（與字串連結後顯示）

```
echo $price, '元X';
echo $count, '個=';
```

  - 要利用變數將單價與個數相乘，計算合計金額時

```
$price*$count;
```

  - 在計算出來的合計金額後面加上「元」後顯示

```
echo $price*$count, '元';
```

設置 PHP 常量

- 使用 define() 函數 - 它使用三個參數：
  - –首個參數定義常量的名稱
  - –第二個參數定義常量的值
  - –可選的第三個參數規定常量名是否對大小寫不敏感。默認是 false。

```
define("GREETING", "Welcome to W3School.com!", true);
```

PHP 變數作用域

- 變數的作用域指的是變數能夠被引用/使用的區域。
- PHP 有三種不同的變數作用域：
  - local (局部)
  - global (全域)
  - static (靜態)

- Local 和 Global 作用域
  - –函數之外聲明的變數擁有 Global 作用域，只能在函數以外進行訪問。
  - –函數內部聲明的變數擁有 LOCAL 作用域，只能在函數內部進行訪問。
  - –右邊的例子測試了帶有局部和全域作用域的變數：

Local 和 Global 作用域

```php
<?php
$x=5; // 全域作用域

function myTest() {
  $y=10; // 局部作用域
  echo "<p>測試函數內部的變數：</p>";
  echo "變數 x 是：$x";
  echo "<br>";
  echo "變數 y 是：$y";
}

myTest();

echo "<p>測試函數之外的變數：</p>";
echo "變數 x 是：$x";
echo "<br>";
echo "變數 y 是：$y";
?>
```

- PHP global 關鍵字
  - global 關鍵字用於在函數內訪問全域變數。
  - 要做到這一點，可在(函數內部)變數前面使用 global 關鍵字：

```php
<?php
$x=5;
$y=10;

function myTest() {
  global $x,$y;
  $y=$x+$y;
}

myTest();
echo $y; // 輸出 15
?>
```

  - PHP 同時在名為 $GLOBALS[index] 的陣列中存儲了所有的全域變數。下標存有變數名。這個陣列在函數內也可以訪問，並能夠用於直接更新全域變數。
  - 上面的例子可以這樣重寫：

```php
<?php
$x=5;
$y=10;

function myTest() {
  $GLOBALS['y']=$GLOBALS['x']+$GLOBALS['y'];
}

myTest();
echo $y; // 輸出 15
?>
```

- PHP static 關鍵字
  - 通常，當函數完成/執行後，會刪除所有變數。不過，有時，需要不刪除某個區域變數。
  - 要完成這一點，請在您首次聲明變數時使用 static 關鍵字：

```php
<?php

function myTest() {
  static $x=0;
  echo $x;
  $x++;
}

myTest();
myTest();
myTest();

?>
```

PHP 資料類型

- PHP 字串：$x = "Hello world!";
- PHP 整數：$x = 5985;
- PHP 浮點數：$x = 10.365;
- PHP 邏輯：$x=true;
- PHP 陣列：$cars=array("Volvo","BMW","SAAB");
- PHP NULL 值：$x=null;
- PHP 類別：

```php
<?php
class Car
{
  var $color;
  function Car($color="green") {
    $this->color = $color;
  }
  function what_color() {
    return $this->color;
  }
}
?>
```

PHP var_dump() 會返回變數的資料類型和值

PHP 字串函數

- strlen()：函數返回字串的長度，以字元計。
- str_word_count()：函數對字串中的單詞進行計數
- strrev()：函數反轉字串。
- strpos()：函數用於檢索字串內指定的字元或文本。
- str_replace()：函數用一些字串替換字串中的另一些字元。

PHP 運算子

运算符	名称	例子	结果
+	加法	$x + $y	$x 与 $y 求和
-	减法	$x - $y	$x 与 $y 的差数
*	乘法	$x * $y	$x 与 $y 的乘积
/	除法	$x / $y	$x 与 $y 的商数
%	取模	$x % $y	$x 除 $y 的余数

运算符	名称	例子	结果
.	串接	$txt1 = "Hello" $txt2 = $txt1 . " world!"	现在 $txt2 包含 "Hello world!"
.=	串接赋值	$txt1 = "Hello" $txt1 .= " world!"	现在 $txt1 包含 "Hello world!"

运算符	名称	例子	结果
==	等于	$x == $y	如果 $x 等于 $y，则返回 true。
===	全等（完全相同）	$x === $y	如果 $x 等于 $y，且它们类型相同，则返回 true。
!=	不等于	$x != $y	如果 $x 不等于 $y，则返回 true。
<>	不等于	$x <> $y	如果 $x 不等于 $y，则返回 true。
!==	不全等（完全不同）	$x !== $y	如果 $x 不等于 $y，或它们类型不相同，则返回 true。
>	大于	$x > $y	如果 $x 大于 $y，则返回 true。
<	小于	$x < $y	如果 $x 小于 $y，则返回 true。
>=	大于或等于	$x >= $y	如果 $x 大于或者等于 $y，则返回 true.
<=	小于或等于	$x <= $y	如果 $x 小于或者等于 $y，则返回 true。

```php
<?php
$aFew = 4;
$some = "4";

if($some == $aFew ){
  echo "相同";
}
?>
```

```php
<?php
$aFew = 4;
$some = "4";

if($some === $aFew ){
  echo "完全相同";
}
?>
```

Chapter3　小結

- 本章介紹了 PHP 的基本程式寫法。從訊息顯示開始，到製作動態網頁應用程式時最重要的 REQUEST 參數的使用方式，以及運用變數與算符等進行計算處理。
- 並為了能在下一章帶出更複雜的程式，在本章也稍微帶到了 if 判斷式等程式控制機制。