目次

一、PHP 標籤（<?php ... ?>）

請先安裝 Visual Studio Code，包括以下擴充功能（extensions）：

- HTML Boilerplate
- HTML Snippets
- PHP IntelliSense

專案與程式

- Windows 環境裡，放置於 C:\xampp\htdocs\
- http://localhost/ 或 http://127.0.0.1/

學習、檢索 PHP 的網站

- https://www.php.net/
- https://www.runoob.com/php/php-tutorial.html
- https://www.w3schools.com/php/default.asp

說明
起始標籤：<?php 結束標籤：?> 這是 PHP 的起始和結束標籤（tags），標籤裡面放置我們所撰寫的 PHP 的程式，讓 PHP 解析程式的起始和結束，方便讓我們在 .php 檔案內任意嵌入 PHP 程式碼。
若是 .php 檔案只有撰寫 PHP 程式（程式檔案純粹撰寫 PHP 程式），那麼可以**不用加上結束標籤（?>）**。這可以避免 PHP 程式結束後，意外在結束標籤外（後）加了一個空白或文字，導致 PHP 程式額外輸出這些空白或文字，但我們並沒有打算要輸出。
原則上 PHP 程式碼要用分號（;）結尾，若是單行程式碼，且「PHP 標籤沒有換行」，可以不用分號結尾。
PHP 可以與 HTML、JavaScript、CSS 標籤與內容整合，一起寫在 .php 的檔案當中。有人戲稱這樣的程式碼排列模式為義大利麵，其開發方式被稱為義大利麵開發法。

範例 1-1.php
`<?php` `phpinfo();`

範例 1-2.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
<!--「echo」是輸出資料的語法，不用加括號，類似 print 的效果。-->
<?php echo "每天都被自己帥醒，壓力好大" ?>

<?php
//單行註解

/**
 * 多行註解
 * 1.
 * 2.
 * 3.
 */
?>

<?php
//PHP 標籤換行，則程式碼需要分號結尾
echo "Hello World!";
echo "Hello PHP!";
?>
</body>
</html>
```

二、變數

變數是用來儲存資料（值）的一個容器，在命名時，有一些原則：

- 在 PHP 中，變數的命名必須以「$」符號作為開頭，後面再加上變數名稱。

- PHP 的變數是由英文字母或底線開始，接著是任意長短的字元、數字或底線。
- 第一個字元不可以使用數字，也不建議用中文來命名。
- 變數名稱有區分大小寫之分，例如 $myVar 與 $myvar 就會被視為不同的變數。
- 變數需要初始化（要有初始值）。
- 變數不能使用保留字作為變數名稱。

PHP 程式為了敘述與陳述式的需求，定義了一些具有特定意義及功能的文字，來執行或表達程式內容，只能留給 PHP 程式使用，這些文字稱為「保留字」。以下是常見的保留字列表：

PHP 常用保留字				
__halt_compiler()	abstract	and	array()	as
break	callable	case	catch	class
clone	const	continue	declare	default
die()	do	echo	else	elseif
empty()	enddeclare	endfor	endforeach	endif
endswitch	endwhile	eval()	exit()	extends
final	finally	for	foreach	function
global	goto	if	implements	include
include_once	instanceof	insteadof	interface	isset()
list()	namespace	new	or	print
private	protected	public	require	reuiqre_once
return	static	switch	throw	trait
try	unset()	use	var	while
xor	yield			

範例

```php
<?php
//以下是正確的變數命名方法
$myvar
$_myVar
$myVar1
$my_var

//以下是錯誤的變數命名方法
$1myVar
```

```
$我的變數
```

在 PHP 中，定義變數不需要設定資料型別（Data Type），只需要指定變數的值：

```
<?php
$myVar = 12;            // 指定變數 $myVar 為 12
$myVar = 6 + 12 * 12;   // 指定變數 $myVar 為 四則運算後的結果: 150
$myVar = "PHP";         // 指定變數 $myVar 為 字串 "PHP"
$myVar = TRUE;          // 指定變數 $myVar 為 布林值 TRUE
```

常數定義的方式如下：

```
define(常數名稱, 常數值 [, 大小寫是否區分]);
define("MATH_PI", 3.1415926, TRUE);
```

常數預設是區分大小寫的，定義時可不加第 3 個參數

```
define("MATH_PI", 3.1415926);
```

PHP 5.3 之後，可以使用關鍵字 const 進行常數的宣告：

```
const MATH_PI = 3.1415926;
```

PHP 程式中的變數，有 8 種資料型別：

- 基本型別
  - 布林值（boolean）
  - 整數（integer）
  - 浮點數（float, double）
  - 字串（string）
- 複合型別
  - 物件（object）
  - 陣列（array）
- 特別型別
  - 資源（resource）
  - 空值（null）

布林值（boolean）

- 一種邏輯型別，只有 true 或 false 這兩種值，通常用於判斷某些條件是否成立。在 PHP 裡，布林值不區分大小寫，true、True、TRUE 都代表是（真），而 false、False、FALSE 都代表否（假）。

- 布林值可以轉換成其它的資料型別。若為數值時，true 值為 1、false 值為 0；若為字串時，true 值為 "1"、false 值為 "0"。

範例

```php
<?php
$isBool = true;
$isBool = True;
$isBool = TRUE;
$isBool = false;
$isBool = False;
$isBool = FALSE;
```

整數（integer）與浮點數（float, double）

整數是不包含小數的數值，範圍可以是正整數、零和負整數。

範例

```php
<?php
$intVar = 10;   //10 進位的整數
$intVar = 010;  // 8 進位的整數
$intVar = 0x10; //16 進位的整數
```

浮點數為包含小數的數值、雙精數或實數。

範例

```php
<?php
$myVar = 1.234; //小數
$myVar = 1.2e3; //有科學記號的浮點數
$myVar = 7e-4; //有科學記號的浮點數
```

字串（string）

字串是由字母、數字、文字、符號所組合而成，經常使用下列 2 種方法來表示：

單引號（'）

範例

```php
<?php
echo '我是一個字串';
```

雙引號（"）

範例
<?php echo "我是一個字串";

物件（object）

- 可以看作一個容器，用「鍵 => 值」(key-value）的方式儲存。

範例
<?php $obj = ["name" => "Alex", "age" => 17];

陣列（array）

程式中的資料，通常是以變數來儲存，如果有大量的同類型資料需要儲存，必須宣告大量的變數，不但耗費程式碼，執行效率也不佳。陣列能夠改善大量變數宣告所造成的效能損失，並且能將相同類型的資料放置在同一個儲存位置，便於資料的操作。

範例
<?php $arr = array('Alex', 'Bill', 'Carl', 'Darren'); $arr = ['Alex', 'Bill', 'Carl', 'Darren'];

資源（resource）

通常由特殊的函式所傳回的值，例如檔案處理、資料庫處理、繪圖處理等操作之後所得到的內容，無法由其它資料型別轉換而來。

空值（null）

想要將變數的內容歸零或是清除時，可以賦予一個空值給變數。

範例 2-1.php

<?php
//字串變數
$strName = "Alex";
 	
//整數變數
$intStores = 7;
 	
//透過「.」來串接變數或是英數、特殊字元混合的資料

```php
echo $strName." 開了 ".$intStores." 間店。";

//可以整合 HTML 標籤
echo "<br /><br />";

//浮點數變數
$floatNumber = 3.1415926;

//「.」與變數、資料之間，也可以用空白區隔或排版
echo $floatNumber . " 是浮點數";

//可以整合 HTML 標籤
echo "<br /><br />";

//布林變數
$isActivated = true;
if( $isActivated ){
    echo "已開通。";
} else {
    echo "未開通";
}
```

範例 2-2.php

```php
<?php
//陣列變數
$arrName = ["Alex", "Bill", "Carl", "Darren"];
echo $arrName[0]."<br />";
echo $arrName[1]."<br />";
echo $arrName[2]."<br />";
echo $arrName[3];

//可以整合 HTML 標籤
echo "<br /><br />";

//物件變數
$obj = ["name" => "Alex", "age" => 17];
echo "姓名: " . $obj["name"] . "，年齡: " . $obj["age"];
```

範例 2-3.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
    <style>
    table { border: 1px solid; }
    table th {border: 1px dashed; }
    table td { border: 1px dotted; }
    </style>
</head>
<body>
    <table>
        <thead>
            <th>姓名</th>
        </thead>
        <tbody>
            <?php
            //學生姓名陣列初始化
            $arrStudent = ["Alex", "Bill", "Carl", "Darren"];

            //count() 函式幫助我們計算陣列的長度
            for($i = 0; $i < count($arrStudent); $i++){
                echo "<tr><td>".$arrStudent[$i]."</td></tr>";
            }
            ?>
        </tbody>
    </table>
</body>
</html>
```

三、字串

變數嵌入在字串當中的標示方式，可以有以下幾種方法：

範例 3-1.php

```php
<?php
//使用雙引號，變數可以嵌入字串來顯示
$myVar = "Darren";
echo "Hi, $myVar <br />";
echo "Hi, {$myVar} <br />";
echo "Hi, ${myVar} <br />";

echo "<br /><br />";

//使用單引號括住字串，變數會變成一般字串
echo 'Hi, $myVar <br />';
echo 'Hi, {$myVar} <br />';
echo 'Hi, ${myVar} <br />';
```

在程式裡，有些特殊字元（符號），例如「'」、「"」、「+」、「*」、「\」、「&」、「^」、「>」、「<」、「[」、「]」、「?」、「.」、「\\」、「(」、「)」、「$」等，會造成字串輸出或處理上的錯誤，我們需要使用跳脫字元（\），讓特殊字元（符號）失效，讓程式視為一般的字元。

範例 3-2.php

```php
<?php
echo "小明說:"妳一定很孤單吧?" 小美:"為什麼這麼說?" 小明:"因為我的心裡只住著妳一個人。"";

/**
 * 正確作法
 * echo "小明說:\"妳一定很孤單吧?\" 小美:\"為什麼這麼說?\" 小明:\"因為我的心裡只住著妳一個人。\"";
 */
```

字串常用函式如下：

函式名稱	說明
nl2br(字串)	以 HTML 的 取代分行字元（\n）
trim(字串[, 列表])	去除字串「起始處」與「結束處」的空白
ltrim(字串[, 列表])	去除字串「起始處」的空白
rtrim(字串[, 列表])	去除字串「結束處」的空白
explode(分隔字元, 字串[, 元素數])	指定分隔字元，將字串分割成另一個字串列（分隔字元可以使用正規表示式，並區分大小寫）
implode(分隔字元, 陣列)	將陣列的元素連結起來，成為字串
join(分隔字元, 陣列)	與 implode 相同
print_r(陣列)	直接將陣列內容輸出，不需要使用 echo
unset(變數[, 變數 2, 變數 3, …])	銷毀、釋放變數
strlen(字串)	查詢字串中的字元長度（個數）；一個 utf-8 編碼的中文字，佔 3 個字元
mb_strlen(字串[, 編碼])	查詢字串中的字元長度（個數），可直接用於 utf-8 的文字
strpos(字串, 查詢字元[, 起始位置])	查詢字元在字串中第一次出現的位置，字元的索引值，從 0 開始；一個 utf-8 編碼的中文字，佔 3 個字元；若是沒有找到查詢字元，則回傳 FALSE
mb_strpos(字元[, 編碼])	查詢中文字元在字串中第一次出現的位置，字元的索引值，從 0 開始，可直接用於 utf-8 的文字；若是沒有找到查詢字元，則回傳 FALSE
substr(字串, 起始位置[, 字數])	擷取字串中，指定開始位置，擷取字數的部分字串；不設定字數，字串會由開始位置取到最後；一個 utf-8 編碼的中文字，佔 3 個字元
mb_substr(字串, 起始位置[, 字數][, 編碼])	擷取字串中，指定開始位置，擷取字數的部分字串；不設定字數，字串會由開始位置取到最後；可直接用於 utf-8 的文字
str_replace(查詢字串, 取代字串, 字串)	在字串中，將查詢字串比對相等的所有文字部分，置換為取代字串
str_pad(字串, 字串總長度, 填入字元[, 類型])	指定填入字元，將原字串填滿到指定的字串總長度；若沒有填寫類型時，預設是向右填滿；一個 utf-8 編碼的中文字，佔 3 個字元

	類型： STR_PAD_RIGHT：向右填滿 STR_PAD_LEFT：向左填滿 STR_PAD_BOTH：向兩側填滿
str_repeat(字串, 次數)	指定字串和次數，來重複字串
strtolower(字串)	將字串中的英文轉成大寫
strtoupper(字串)	將字串中的英文轉成小寫
md5(字串[, 布林值])	使用 MD5 計算字串雜湊值，並返回。預設 32 個字元長度的字串。
sha1(字串[, 布林值])	使用 MD5 計算字串雜湊值，並返回。預設 40 個字元長度的字串。

範例 3-3.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
<?php
//nl2br()
$str01 = "小明:「我重要嗎？」\n 小美:「再重都要。」";
echo nl2br($str01);
?>

<hr>

<?php
//trim()、ltrim()、rtrim()
$str02 = "      我想妳一定很忙，所以只要看前三個字就好…      ";
echo trim($str02)."<br>";
echo ltrim($str02)."<br>";
echo rtrim($str02);
?>
```

```
<hr>

<?php
//explode()、print_r()
$str03 = "人,帥,得,體";
$arr03 = explode("," , $str03);
echo $arr03[0] . $arr03[1] . $arr03[2] . $arr03[3] . "<br>";
print_r($arr03);
?>

<hr>

<?php
//implode()、join()
$str04_1 = implode("~", $arr03);
echo $str04_1 . "<br>";
$str04_2 = join("...", $arr03);
echo $str04_2 . "<br>";
?>

<hr>

<?php
//strlen()、mb_strlen()、strpos()、mb_strpos()、substr()、mb_substr()
$str05_1 = "abcdefg";
$str05_2 = "懷疑人生";
echo strlen($str05_1) . "<br>";
echo mb_strlen($str05_2) . "<br>";
echo strpos($str05_1, "c") . "<br>";
echo mb_strpos($str05_2, "人") . "<br>";
echo substr($str05_1, 3, 5) . "<br>";
echo mb_substr($str05_2, 2, 3);
?>

<hr>

<?php
```

```php
//str_replace()、str_pad()、str_repeat()
$str06 = "正規表達式";
echo str_replace("達", "示", $str06) . "<br>";
echo str_pad("不要", 30, "啊") . "<br>";
echo "y" . str_repeat("e", 5);
?>

<hr>

<?php
//strtolower()、strtoupper()
$str07_1 = "HELLO ";
$str07_2 = "world!";
echo strtolower($str07_1) . strtoupper($str07_2);
?>

<hr>

<?php
//md5()
$strOrigin = "T1st@localhost";
echo "原始資料: " . $strOrigin . "<br>";
echo "md5() 加密後: " . md5($strOrigin) . "<br>";

$strOrigin = "test@localhost";
echo "修改後資料: " . $strOrigin . "<br>";
echo "md5() 加密後: " . md5($strOrigin) . "<br>";

$strOrigin = "T1st@localhost";
echo "回復原資料: " . $strOrigin . "<br>";
echo "md5() 加密後: " . md5($strOrigin) . "<br>";
?>

<hr>

<?php
//md5()
$strOrigin = "T1st@localhost";
```

```php
echo "原始資料: " . $strOrigin . "<br>";
echo "sha1() 加密後: " . sha1($strOrigin) . "<br>";

$strOrigin = "test@localhost";
echo "竄改後資料: " . $strOrigin . "<br>";
echo "sha1() 加密後: " . sha1($strOrigin) . "<br>";

$strOrigin = "T1st@localhost";
echo "回復原資料: " . $strOrigin . "<br>";
echo "sha1() 加密後: " . sha1($strOrigin) . "<br>";
?>

<hr>

<?php
//銷毀、釋放變數
unset($str01, $str02, $str03, $arr03, $str04_1, $str04_2);
?>

</body>
</html>
```

四、運算子

在程式設計中，必須使用變數或常數來儲存或是代表一些資料，再將這些資料，經過邏輯判斷與演算，去得到所需的結果，建構整個流程的內容即是運算式。運算式是由「運算元」與「運算子」所組合而成，其中「運算子」是指運算的方法（以符號來代表），運算元是用來運算的資料。

範例
`$a + $b`

加號（+）是運算子，代表運算的方式；$a 和 $b 是運算元，用來運算的資料。

在開始前，我們先補充一個觀念，就是三元運算子，其格式如下：

條件運算式 ? 成立時傳回運算式 : 不成立時傳回運算式;

範例

```php
$a = 2;
$b = ($a > 0) ? '正數' : '負數';
echo $b; //印出 正數
```

指派運算子

符號	說明	範例
=	符號兩側，**由右往左**賦值	$numVar = 3; $strVar = "May you have a nice day."; $myVar = 6 + 12 * 12;

算數運算子，執行程式時，進行加減乘除的動作。

符號	說明	範例	運算結果
+	加法	6 + 4	10
-	減法	6 – 4	2
*	乘法	5 * 5	25
/	除法	12 / 4	3
%	取餘數	5 % 3	2

關係運算子，或稱為「比較運算子」，會將運算兩邊的運算元加以比較，再將結果以布林值（true 或 false）回傳。

符號	說明	範例	運算結果
==	相等	$a == $b	當兩者相等時，成立
===	全等	$a === $b	當兩者相等，且型別一樣時，成立
!=	不等於	$a != $b	當兩者不等時，成立
!===	不全等	$a !== $b	當兩者不相等，或型別不一樣時，成立
<	小於	$a < $b	前者小於後者時，成立
>	大於	$a > $b	當前者大於後者時成立
<=	小於或等於	$a <= $b	當前者比後者小，或兩者一樣時，成立
>=	大於或等於	$a >= $b	當前者比後者大。或兩者一樣時，成立

範例 4-1.php

```php
<?php
//指派運算子
$myVar01 = 6;
echo $myVar01;
```

```php
echo "<hr>";

//算數運算子
$myVar02 = 3 + 7;
$myVar03 = 6 * 5;
$myVar04 = 7 % 3;
echo "myVar02 = {$myVar02}, myVar03 = {$myVar03}, myVar04 = {$myVar04}";

echo "<hr>";

//關係運算子
echo (1 == "1") ? '1 == "1" 為真' : '1 == "1" 為假';
echo "<br>";
echo (1 === "1") ? '1 === "1" 為真' : '1 === "1" 為假';
echo "<br>";
echo (5 > 3) ? "5 > 3 為真" : "5 > 3 為假";
echo "<br>";
echo (2 < 6) ? "2 < 6 為真" : "2 < 6 為假";
```

邏輯運算子，將運算式兩邊的運算元之**布林值**進行邏輯比較，再將結果以布林值回傳：

符號	說明	範例	運算結果
and	兩邊同為「真」值	$a and $b	當 $a、$b 都是 true 時成立
&&	兩邊同為「真」值	$a && $b	當 $a、$b 都是 true 時成立
or	兩邊任一為「真」值	$a or $b	當 $a、$b 任一是 true 時成立
||	兩邊任一為「真」值	$a || $b	當 $a、$b 任一是 true 時成立
xor	異值符號	$a xor $b	當 $a、$b 都是 true 或都是 false 時，不成立；一個為 true、一個為 false 時，成立。 口訣：相同為 0，相異為 1
!	反值符號	!$a	$a 的相反值

複合運算子

可以結合字串、指派運算子、算術運算子，是簡化運算式的一種方法。以往在程式中，常會將舊的變數值，加上另一個值，再帶回原變數。

符號	說明	範例	運算結果
+=	加法指派	$a += $b	$a = $a + $b;

-=	減法指派	$a -= $b	$a = $a - $b;
*=	乘法指派	$a *= $b	$a = $a * $b;
/=	除法指派	$a /= $b	$a = $a / $b;
%=	餘數指派	$a %= $b	$a = $a % $b;
.=	字串連結指派	$a .= $b	$a = $a . $b;

遞增/遞減運算子

開發的過程中，常常需要讓變數的值遞增或遞減 1。

符號	說明	作用	範例	運算結果
++	遞增	將變數值加 1	$a = 5; $a++;	6
--	遞減	將變數值減 1	$a = 5; $a--;	4

範例 4-2.php

```php
<?php
//邏輯運算子
$a = true;
$b = true;
echo ($a && $b) ? '$a && $b 為真' : '$a && $b 為假';

echo "<hr>";

$a = true;
$b = false;
echo ($a || $b) ? '$a && $b 為真' : '$a && $b 為假';

echo "<hr>";

$c = true;
echo (!$c) ? '$c 為真' : '$c 為假';

echo "<hr>";

//複合運算子
$x1 = 10;
$y1 = 5;
$x1 = $x1 + $y1; //可以改寫成 $x1 += $y1;
```

```php
echo $x1;

echo "<hr>";

$x2 = 10;
$y2 = 5;
$x2 = $x2 / $y2; //可以改寫成 $x2 /= $y2;
echo $x2;

echo "<hr>";

//遞增、遞減運算子
$x3 = 1;
$x3++;
echo $x3;

echo "<hr>";

$x4 = 7;
$x4--;
echo $x4;
```

五、流程控制

程式的執行，基本上是循序漸進、由上而下一行一行地執行，但是有時①內容會因為判斷的情況不同，而去執行不同的程式區塊，或②設定條件執行某些重複的內容。

在 PHP 中，流程控制的指令，分為兩類：條件控制、迴圈。

①條件控制的指令包括：	②迴圈指令包括：
● if ● if … else ● if … elseif … else ● switch	● while ● do … while ● for ● foreach
在流程控制中，還有指令是控制由判斷式或迴圈中跳出的動作，包括： ● break	

● continue

單向選擇 if…

格式	說明
if (條件式) 執行程式的內容;	單行程式

if (條件式) {
　　執行的程式內容;
　　……;
}
	若程式內容不是單行，就必須使用左右大括號 {} ，將程式區塊包含起來

雙向選擇 if … else

格式

if (條件式) {
　　條件成立時執行的程式內容;
　　……;
} else {
　　條件不成立時執行的程式內容;
　　……;
}

多向選擇 if … elseif … else

格式

if(條件式 1) {
　　條件 1 成立時，所執行的程式內容;
　　……;
} else if(條件 2) {
　　條件 2 成立時，所執行的程式內容;
　　……;
} else {
　　所有條件都不成立時，所執行的程式內容;
　　……;
}

switch 條件控制

格式
switch (自訂變數) { 　　case 條件值 1: 　　　　自訂變數等於條件值 1 所執行的程式內容;

```
    break;

    case 條件值 2:
        自訂變數等於條件值 2 所執行的程式內容;
    break;

    case 條件值 3:
        自訂變數等於條件值 3 所執行的程式內容;
    break;
    ……
    default:
        當自訂變數與所有條件值都不相等時，預設執行的程式內容;
}
```

範例 5-1.php

```php
<?php
//單向選擇 if
$a = 5;
if($a > 0) echo '$a 變數的值是正數';
echo "<br>";
if($a > 0) {
    echo '$a 變數的值大於 0';
}

echo "<hr />";

//雙向選擇 if … else
$a = -5;
if($a > 0) {
    echo '$a 變數的值是正數';
} else {
    echo '$a 變數的值是負數';
}

echo "<hr />";

//多向選擇 if … else if … else
$score = 85;
```

```php
if($score >= 60 && $score < 70) {
    echo '丙等';
} elseif($score >= 70 && $score < 80) {
    echo '乙等';
} elseif($score >= 80 && $score < 90) {
    echo '甲等';
} elseif($score >= 90 && $score <= 100) {
    echo '優等';
} else {
    echo '不及格';
}

echo "<hr />";

//switch 條件控制
$direction = "南";
switch($direction) {
    case '東':
        echo "我要往東走";
    break;

    case '南':
        echo "我要往南走";
    break;

    case '西':
        echo "我要往西走";
    break;

    case '北':
        echo "我要往北走";
    break;

    default:
        echo "我不知道要往哪裡走";
}
```

在程式流程控制中，另一個相當重要的結構，就是迴圈。在程式某些區塊，會

因為條件判斷或是設定次數的關係，重複執行，一直到不符合條件，或達到設定次數後，才往下執行，這就是所謂的迴圈。

while 迴圈

格式

while (條件式) {
　　執行的程式內容;
　　……;
}

do…while 迴圈

格式
do { 　　執行的程式內容; 　　……; } while (條件式);

do … while 迴圈與 while 迴圈幾乎是一樣的，只是 do … while 迴圈是先執行迴圈中的程式，在最後才進行條件判斷。

for 計次迴圈

先設定一個變數的初值，再設定該變數執行計次的條件，最後設定變數的計次方式。當符合條件，即執行指定的程式區塊後計次，一直到不符合條件，才跳出迴圈，結束程式或往下執行。

格式

for(①設定變數初值; ②條件式; ④變數計次方式) {
　　③執行的程式內容;
　　……;
}

補充：HTML 實體參照

可用於 echo 時的符號輸出。

HTML 原始碼	顯示結果	描述
&lt;	<	小於號或顯示標記
&gt;	>	大於號或顯示標記
&amp;	&	可用於顯示其它特殊字元

&quot;	“	引號
&reg;	®	已註冊
&copy;	©	版權
&trade;	™	商標
		半個空白位
		一個空白位
		不斷行的空白

範例 5-2.php

```php
<?php
//while 迴圈
$i = 0;
while ($i < 10) {
    echo $i . " "; //  為空白字元
    $i++;
}

echo "<hr>";

//do…while 迴圈
$i = 0;
do {
    echo $i . " "; //  為空白字元
    $i++;
} while ($i < 10);


echo "<hr>";

//for 迴圈
for($i = 0; $i < 10; $i++) {
    echo $i . " "; //  為空白字元
```

```
}
```

程式設計時，有時需要將目前的執行動作直接跳出去流程控制區塊或是迴圈，執行下一輪的迴圈動作，或向下執行程式。此時就必須使用跳躍指令。break 跟 continue 是流程控制中的跳躍指令，它們都能停止目前的程式動作，不同的是 break 指令會跳出流程控制區塊，而 continue 指令會結束目前的迴圈程式執行內容，在迴圈內，進行下一輪迴圈的執行。

範例 5-3.php

```php
<?php
//只要 $i 不是偶數(即為奇數)，就會執行 break，直接跳出 for 迴圈
for($i = 1; $i <= 10; $i++) {
    if($i % 2 != 0) {
        echo $i." ";
    } else {
        break;
    }
}

echo "<hr>";

/**
 * 只要 $i 不是偶數(即為奇數)，不會立刻結束迴圈，
 * 而是 $i++ 後，繼續執行迴圈內的程式內容
 */
for($i = 1; $i <= 10; $i++) {
    if($i % 2 != 0) {
        echo $i." ";
    } else {
        continue;
    }
}
```

六、陣列

在沒有使用陣列的情況下，我們需要這樣記錄資料：

```
let name1 = "Alex";
let name2 = "Bill";
let name3 = "Cook";
let name4 = "Darren";
.
.
let name9999 = 'Somebody';
```

上面這種列表會變得很不好用，假設每一個人的名字都需要一張紙來記錄，這要浪費多少紙張？於是我們可以使用類似「清單」概念的陣列，將所有名字都記錄在同一張列表上，這樣就簡單多了。

語法格式：

- 初始化陣列：
  - 陣列名稱 = array(元素 1, 元素 2, 元素 3, …);
  - 陣列名稱 = [元素 1, 元素 2, 元素 3, …];
- 陣列賦值：陣列名稱[索引鍵] = 元素值;
- 通常索引鍵從「0」開始（整數索引值）。
- 有維度的概念（一維陣列、二維陣列、…）
- 索引鍵可以用「字串」的格式來取代整數索引值。
- 可以使用$arr[] = 變數或值，達到 append 或 push 的效果。

一維陣列

範例 6-1.php

```php
<?php
//陣列初始化
$arr = array('Alex', 'Bill', 'Carl', 'Darren');
$arr = ['Alex', 'Bill', 'Carl', 'Darren'];

//指定索引，印出相對應的值
echo "印出 {$arr[3]}"; //印出 Darren

echo "<hr>";

//新增兩個值，在陣列的尾端
$arr[] = "Eric";
$arr[] = "Fox";
```

```php
//指定索引，印出相對應的值
echo "新增了 {$arr[4]} 和 {$arr[5]}";

echo "<hr>";

//使用 for 迴圈，逐一輸出陣列中的值
for($i = 0; $i < count($arr); $i++){
    echo $arr[$i] . " ";
}
```

二維陣列

範例 6-2.php

```php
<?php
$arr[0][0] = '00 同學';
$arr[0][1] = '01 同學';
$arr[0][2] = '02 同學';
$arr[0][3] = '03 同學';
$arr[0][4] = '04 同學';
$arr[1][0] = '10 同學';
$arr[1][1] = '11 同學';
$arr[1][2] = '12 同學';
$arr[1][3] = '13 同學';
$arr[1][4] = '14 同學';
$arr[2][0] = '20 同學';
$arr[2][1] = '21 同學';
$arr[2][2] = '22 同學';
$arr[2][3] = '23 同學';
$arr[2][4] = '24 同學';

for($i = 0; $i < count($arr); $i++){
    for($j = 0; $j < count($arr[$i]); $j++){
        echo $arr[$i][$j] . "<br />";
    }
}
```

關聯式陣列，又稱結合式陣列（associative array），或稱關聯式陣列。陣列除了使用整數當作索引鍵，也可以使用字串當作索引鍵的值，使用方法類似「物件變數」（Object）。

範例 6-3.php

```php
<?php
$myArray = [];
$myArray['myName'] = 'Alex';
$myArray['myHeight'] = 160;
$myArray['myWeight'] = 90;
echo "大家好，我的名字是 ".$myArray['myName']."，";
echo "我的身高是 ".$myArray['myHeight']."，";
echo "我的體重是 ".$myArray['myWeight']."。";
```

有一個很重要的觀念，就是整數索引鍵與字串索引鍵的綜合應用。

假設我們有一個通訊錄，看起來像這樣：

學號	姓名	性別	生日	手機號碼
101	阿土伯	男	2000/3/14	0910123456
102	錢夫人	女	2000/6/6	0978222333
103	孫小美	女	2000/7/15	0939666999
104	約翰喬	男	2000/8/7	0910765432

我們希望建立 101 到 104 的學生資料，可以這麼做：

範例

```php
$arrStudents = [];

$arrStudents[] = array(
    '學號' => '101',
    '姓名' => '阿土伯',
    '性別' => '男',
    '生日' => '2000/3/14',
    '手機號碼' => '0910123456'
);

$arrStudents[] = array(
    '學號' => '102',
    '姓名' => '錢夫人',
    '性別' => '女',
    '生日' => '2000/6/6',
    '手機號碼' => '0978222333'
```

```php
$arrStudents = [];

$arrStudents[] = [
    '學號' => '101',
    '姓名' => '阿土伯',
    '性別' => '男',
    '生日' => '2000/3/14',
    '手機號碼' => '0910123456'
];

$arrStudents[] = [
    '學號' => '102',
    '姓名' => '錢夫人',
    '性別' => '女',
    '生日' => '2000/6/6',
    '手機號碼' => '0978222333'
```

```
);

$arrStudents[] = array(
    '學號' => '103',
    '姓名' => '孫小美',
    '性別' => '女',
    '生日' => '2000/7/15',
    '手機號碼' => '0939666999'
);

$arrStudents[] = array(
    '學號' => '104',
    '姓名' => '約翰喬',
    '性別' => '男',
    '生日' => '2000/8/7',
    '手機號碼' => '0910765432'
);
```

```
];

$arrStudents[] = [
    '學號' => '103',
    '姓名' => '孫小美',
    '性別' => '女',
    '生日' => '2000/7/15',
    '手機號碼' => '0939666999'
];

$arrStudents[] = [
    '學號' => '104',
    '姓名' => '約翰喬',
    '性別' => '男',
    '生日' => '2000/8/7',
    '手機號碼' => '0910765432'
];
```

透過 print_r($arrStudents) 函式的執行，可以看到陣列的全貌：

```
Array
(
    [0] => Array
        (
            [學號] => 101
            [姓名] => 阿土伯
            [性別] => 男
            [生日] => 2000/3/14
            [手機號碼] => 0910123456
        )

    [1] => Array
        (
            [學號] => 102
            [姓名] => 錢夫人
            [性別] => 女
            [生日] => 2000/6/6
            [手機號碼] => 0978222333
        )

    [2] => Array
        (
            [學號] => 103
            [姓名] => 孫小美
            [性別] => 女
            [生日] => 2000/7/15
            [手機號碼] => 0939666999
        )

    [3] => Array
        (
            [學號] => 104
            [姓名] => 約翰衛
            [性別] => 男
            [生日] => 2000/8/7
            [手機號碼] => 0910765432
        )

)
```

圖: $arrStudents 陣列的全貌

當然，你也可以直接對陣列賦值：

範例

```php
$arrStudents[0]['學號'] = '101';
$arrStudents[0]['姓名'] = '阿土伯';
$arrStudents[0]['性別'] = '男';
$arrStudents[0]['生日'] = '2000/3/14';
$arrStudents[0]['手機號碼'] = '0910123456';

$arrStudents[1]['學號'] = '102';
```

```php
$arrStudents[1]['姓名'] = '錢夫人';
$arrStudents[1]['性別'] = '女';
$arrStudents[1]['生日'] = '2000/6/6';
$arrStudents[1]['手機號碼'] = '0978222333';

$arrStudents[2]['學號'] = '103';
$arrStudents[2]['姓名'] = '孫小美';
$arrStudents[2]['性別'] = '女';
$arrStudents[2]['生日'] = '2000/7/15';
$arrStudents[2]['手機號碼'] = '0939666999';

$arrStudents[3]['學號'] = '104';
$arrStudents[3]['姓名'] = '約翰喬';
$arrStudents[3]['性別'] = '男';
$arrStudents[3]['生日'] = '2000/8/7';
$arrStudents[3]['手機號碼'] = '0910765432';
```

由於二維陣列中，第一維的索引鍵是整數，我們可以透過迴圈的執行，逐一顯示第二維字串索引的值。

範例 6-4.php

```php
<?php
//初始化陣列變數
$arrStudents = [];

//各別 append 或 push 關聯式陣列 (類似物件)
$arrStudents[] = [
    '學號' => '101',
    '姓名' => '阿土伯',
    '性別' => '男',
    '生日' => '2000/3/14',
    '手機號碼' => '0910123456'
];

$arrStudents[] = [
    '學號' => '102',
    '姓名' => '錢夫人',
    '性別' => '女',
```

```
    '生日' => '2000/6/6',
    '手機號碼' => '0978222333'
];

$arrStudents[] = [
    '學號' => '103',
    '姓名' => '孫小美',
    '性別' => '女',
    '生日' => '2000/7/15',
    '手機號碼' => '0939666999'
];

$arrStudents[] = [
    '學號' => '104',
    '姓名' => '約翰喬',
    '性別' => '男',
    '生日' => '2000/8/7',
    '手機號碼' => '0910765432'
];

//將陣列內容印出來檢視
echo "<pre>";
print_r($arrStudents);
echo "</pre>";

//這個案例的整數索引，從 0 開始；count() 函式可以算出陣列的長度：4
for($i = 0; $i < count($arrStudents); $i++){
    echo '學號: '.$arrStudents[$i]['學號']."<br />";
    echo '姓名: '.$arrStudents[$i]['姓名']."<br />";
    echo '性別: '.$arrStudents[$i]['性別']."<br />";
    echo '生日: '.$arrStudents[$i]['生日']."<br />";
    echo '手機號碼: '.$arrStudents[$i]['手機號碼']."<br />";
}
```

若需要對於陣列中每一個元素進行處理時，可以使用 foreach 迴圈取出元素的內容。foreach 迴圈是專門設計給陣列使用的迴圈，每重複一次即可將陣列移動到下一個元素中。

語法：不使用陣列索引鍵

格式

foreach (陣列名稱 as 值 value) {
　　執行的程式內容;
}

語法：使用陣列索引鍵

格式

foreach (陣列名稱 as 索引值變數 key => 變數值 value) {
　　執行的程式內容;
}

範例 6-5.php

```php
<?php
//不使用陣列索引鍵
$arrSeasons = ['春', '夏', '秋', '冬'];
echo "每年的四季分別為： ";
foreach($arrSeasons as $value) {
    echo $value." ";
}

echo "<hr>";

//各別輸出陣列的索引鍵 key，同時輸對應的值 value
$arrPerson = [
    '學號' => '103',
    '姓名' => '孫小美',
    '性別' => '女',
    '生日' => '2000/7/15',
    '手機號碼' => '0939666999'
];

foreach($arrPerson as $key => $value) {
    echo $key.": ".$value."<br />";
}

echo "<hr>";
```

```php
//輸出二維陣列的結果，其中第二維放置關聯式陣列（物件）
$arrStudents = [];
$arrStudents[] = ["name" => "Alex", "age" => 18];
$arrStudents[] = ["name" => "Bill", "age" => 21];
$arrStudents[] = ["name" => "Carl", "age" => 13];
$arrStudents[] = ["name" => "Darren", "age" => 19];

foreach($arrStudents as $key => $obj){
    echo "{$obj['name']} 今年 {$obj['age']}<br>";
}
```

判斷陣列變數：

- 若是要判斷陣列指定索引的值「是否為空」，可以使用 empty() 函式，為空則回傳 true；不為空，則回傳 false
  - "" (空字串)
  - 0 (代表整數 0)
  - 0.0 (代表浮點數 0)
  - "0" (代表字串符號的 0)
  - NULL
  - FALSE
  - array() (一個空陣列)
  - $var; (變數已經宣告，卻沒有提供初始值)
- 若是要判斷「變數」或是「陣列對應的索引值」是否已經初始化，可以用 isset() 函式，已初始化，則回傳 true；不為空，則回傳 false。

範例 6-6.php

```php
<?php
//建立一個空陣列
$arr = [];

$arr[0] = "";
$arr[1] = 0;
$arr[2] = false;
$arr[3] = 10;
$arr[4] = NULL;

echo '$a[0] = "" ... 是否為空? ';
echo empty($arr[0]) ? '為空' : '不為空';
```

```php
echo "<hr />";

echo '$a[1] = 0 ... 是否為空? ';
echo empty($arr[1]) ? '為空' : '不為空';

echo "<hr />";

echo '$a[2] = false ... 是否為空? ';
echo empty($arr[2]) ? '為空' : '不為空';

echo "<hr />";

echo '$a[3] = 10 ... 是否為空? ';
echo empty($arr[3]) ? '為空' : '不為空';

echo "<hr />";

echo '$a[4] = NULL ... 是否為空? ';
echo empty($arr[4]) ? '為空' : '不為空';
```

範例 6-7.php

```php
<?php
//使用 isset() 判斷變數是否初始化
$myVar01 = 10;
if( isset($myVar01) ){
    echo "myVar01 已初始化";
} else {
    echo "myVar01 尚未初始化";
}

echo "<hr />";

//拿一個尚未宣告的變數來判斷
if( isset($myVar02) ){
    echo "myVar02 已初始化";
} else {
    echo "myVar02 尚未初始化";
```

```php
}

echo "<hr />";

//判斷陣列元素是否初始化
$arr = ['春', '夏', '秋', '冬'];
if( isset($arr[0]) ){
    echo "arr[0] 的值 {$arr[0]} 已初始化";
} else {
    echo "arr[0] 尚未初始化";
}

echo "<hr />";

if( isset($arr[4]) ){
    echo "arr[4] 的值 {$arr[4]} 已初始化";
} else {
    echo "arr[4] 尚未初始化";
}
```

七、GET 與 POST

補充說明：
以「$_」開頭的 PHP 變數，原則上屬於全域變數，例如 $_GET、$_POST、$_FILES、$_SERVER、$_SESSION、$_COOKIE 等。

網頁資料將以字串方式附加在網址（URI）的後面傳送，在網址尾端，會以「?」符號，開啟跟著表單中的資料，每個欄位間的值，以「&」連接起來。

一般來說，GET 參數的格式類似下列文字：
https://www.104.com.tw/jobs/search/?ro=0&jobcat=2007001006&kwop=7&keyword=php&expansionType=area,spec,com,job,wf,wktm&area=6001001000&order=12&asc=0&page=2&mode=s&jobsource=2018indexpoc

key（鍵）	value（值）
ro	0

jobcat	2007001006
kwop	7
keyword	php
expansionType	area,spec,com,job,wf,wktm
area	6001001000
order	12
asc	0
page	2
mode	s
jobsource	2018indexpoc

我們先建立一個超連結，將 GET 參數以字串「?ro=0&jobcat=2007001006&kwop=7&keyword=php&expansionType=area,spec,com,job,wf,wktm&area=6001001000&order=12&asc=0&page=2&mode=s&jobsource=2018indexpoc」放在網址尾端，在點選超連結時，一併送到另一個 PHP 頁面來接收。接收的變數為 `$_GET`，其中 `$_GET['ro']` 代表值 0、`$_GET['jobcat']` 代表值 2007001006，依此類推。

範例 7-1.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

<a href="./7-1-
1.php?ro=0&jobcat=2007001006&kwop=7&keyword=php&expansionType=area,sp
ec,com,job,wf,wktm&area=6001001000&order=12&asc=0&page=2&mode=s&jobso
urce=2018indexpoc">請按我<a/>

</body>
</html>
```

範例 7-1-1.php

```php
<?php
//列出所有 GET 變數的資訊
echo $_GET['ro'];
echo "<hr>";
echo $_GET['jobcat'];
echo "<hr>";
echo $_GET['kwop'];
echo "<hr>";
echo $_GET['keyword'];
echo "<hr>";
echo $_GET['expansionType'];
echo "<hr>";
echo $_GET['area'];
echo "<hr>";
echo $_GET['order'];
echo "<hr>";
echo $_GET['asc'];
echo "<hr>";
echo $_GET['page'];
echo "<hr>";
echo $_GET['mode'];
echo "<hr>";
echo $_GET['jobsource'];

echo "<hr>";

//判斷某個 key 是否存在於 GET 變數當中
if(isset($_GET['keyword'])) {
    echo '有 $_GET["keyword"]';
} else {
    echo '沒有 $_GET["keyword"]';
}
```

在課堂中，我們將會使用 HTML 當中的 form 表單，來進行 POST 資料傳遞。以下是網頁表單的格式與說明：

格式

```
<form name = "myForm"
action = "資料處理程式的 URI"
method = "GET / POST"
enctype = "application/x-www-form-urlencoded / multipart/form-data")
>
……
</form>
```

說明

- name 是指 form 的名稱，例如：myForm。
- action 是指該 form 被使用者送出之後，負責接收與處理資料的程式之 URI（Uniform Resource Identifiers）。如果省略不寫的話，會以當前所在的 URI 來取代。
- method 用來規範該 form 被送出時，所採用的 HTTP method，預設值是 GET。
  - POST 方法是將資料包裝在 HTTP 標頭內（headers）傳送給 web server。
  - 而 GET 方法則是將資料直接加在 URI（?key01=value01&key02=value02&…）之後。
  - 使用 GET method 所能傳遞的資料有限（連同 URI 共 255 字元），在需要上傳大量資料或檔案時，請使用 POST method。
- enctype 用以規範該 form 被送出時，所採用的 content type。可用的值有兩種：application/x-www-form-urlencoded（預設值）與 multipart/form-data。
  - 當您打算透過表單來**上傳檔案**時，請務必**將 enctype 設為 multipart/form-data**，同時 **method 也要設為 POST** 才行。

範例 7-2.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
```

```
<form name="myForm"
        action="./7-2-1.php"
        method="POST"
        enctype="application/x-www-form-urlencoded">
    <label>我的姓名: </label>
    <input type="text" name="myName" /> <br />
    <label>我的年紀: </label>
    <input type="text" name="myAge" /> <br />
    <label>我的身高: </label>
    <input type="text" name="myHeight" /> <br />
    <label>我的體重: </label>
    <input type="text" name="myWeight" /> <br />
    <input type="submit" name="smb" value="送出" />
</form>

</body>
</html>
```

範例 7-2-1.php

```
<?php
echo "我的姓名: ".$_POST['myName']."<br />";
echo "我的年紀: ".$_POST['myAge']."<br />";
echo "我的身高: ".$_POST['myHeight']."<br />";
echo "我的體重: ".$_POST['myWeight'];

echo "<hr>";

//判斷某個 key 是否存在於 POST 變數當中
if(isset($_POST['myName'])) {
    echo '有 $_GET["myName"]';
} else {
    echo '沒有 $_GET["myName"]';
}
```

當使用 HTML 表單的 input 標籤時，其 type 為何時，可以有多個標籤有相同的 name 屬性值，常用的有 radio 和 checkbox。

radio

適用於數個項目擇其一的情況，例如選擇是男生還是女生。

範例 7-3.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

<form name="myForm"
        action="./7-3-1.php"
        method="POST"
        enctype="application/x-www-form-urlencoded">
    <label>男: </label>
    <input type="radio" name="myGender" checked="checked" value="男" /> <br />
    <label>女: </label>
    <input type="radio" name="myGender" value="女" /> <br />
    <input type="submit" name="smb" value="送出" />
</form>

</body>
</html>
```

範例 7-3-1.php

```
<?php
echo "您所選擇的性別為: ".$_POST['myGender'];
```

checkbox

適用於多個項目複選的情況，例如學校老師上課時，需要用到白板筆，白板筆有黑色、藍色、紅色、綠色，我們可以一次複選幾種顏色的筆。名稱與 radio 不同的是，我們需要將名稱設定為：name= "myColor[]"，讓其在送出時，被 PHP 視為一個陣列，可以透過陣列的處理方式，取得傳送過來的值。

範例 7-4.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

<form name="myForm"
        action="./7-4-1.php"
        method="POST"
        enctype="application/x-www-form-urlencoded">
    <h3>上課所需要用到的白板筆顏色</h3>
    <label>黑色: </label>
    <input type="checkbox" name="myColor[]" value="黑色" /> <br />
    <label>藍色: </label>
    <input type="checkbox" name="myColor[]" value="藍色" /> <br />
    <label>紅色: </label>
    <input type="checkbox" name="myColor[]" value="紅色" /> <br />
    <label>綠色: </label>
    <input type="checkbox" name="myColor[]" value="綠色" /> <br />
    <input type="submit" name="smb" value="送出" />
</form>

</body>
</html>
```

範例 7-4-1.php

```
<?php
for($i = 0; $i < count($_POST['myColor']); $i++) {
    echo "您所選擇的顏色為: " . $_POST['myColor'][$i] . "<br />";
};
```

檔案上傳

即是將檔案由客戶端的主機，藉由瀏覽器傳送到伺服器的資料夾上。整個檔案

上傳的流程為：

- 在表單檔案欄位選取要上傳的檔案
- 表單送出，將檔案傳送到伺服器
- 伺服器在接收的過程中，先將接收到的檔案，放置到暫存資料夾中
- 傳送完畢後，將完整的檔案搬移到指定的網頁資料夾中

在進行檔案上傳之前，我們需要進行檔案上傳功能的調整（在 php.ini）；若是設定上傳的路徑或資料夾，則需要有可以編輯的權限：

- file_uploads = On
- upload_tmp_dir = "C:\xampp\tmp"
- upload_max_filesize = 40M
  - phpMyAdmin 匯入 .sql 檔案的大小限制，來自這裡；若需要自行開發上傳功能，其上傳限制可以依需求調整。
- date.timezone = Asia/Taipei
  - 一般來說，在 php.ini 當中，可以搜尋「date.timezone」，若前面有「;」，則將前面的「;」拿掉，同時加入「Asia/Taipei」。
  - XAMPP 在 php.ini 最底下，有額外設定 [Date] 的時區，所以要滾動、移動到檔案最下方，尋找 [Date] 標籤，XAMPP 預設為「Europe/Berlin」，我們要改成「Asia/Taipei」，之後重新啟動 Apache 即可（使用 XAMPP control panel，按下 apache 的 stop，再重新 start）。

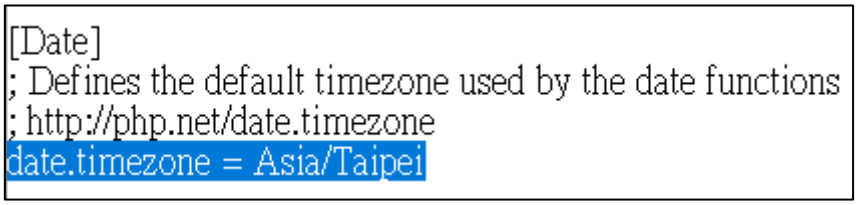

圖: 在一般設定下，只要改變預設 date.timezone 值即可

圖: 在 XAMPP 裡，要到 php.ini 檔案最底部來設定 date.timezone 的值

上傳檔案的表單，有幾個重要的注意事項：

- 在 form 標籤中，action 屬性必須設定為接收檔案的 PHP 程式檔。
- 在 form 標籤中，method 屬性必須設定為 POST (大小寫皆可)。
- 在 form 標籤中，因為有檔案上傳，所以要設定資料的編碼方式，這時候要加上 enctype= "multipart/form-data"，才能讓檔案正確地送出。
- 上傳檔案的 input 標籤，type 屬性必須設定為 type= "file"，才能在使用時，出現瀏覽鈕，讓使用者選擇上傳的檔案。

接收上傳的檔案時，可以透過 $_FILES 取得暫存檔的資訊。以下列出相關資訊：

函式名稱	說明與範例
$_FILES["欄位名稱"]["tmp_name"]	取得上傳檔案暫存檔名稱
$_FILES["欄位名稱"]["name"]	取得上傳檔案原始檔名
$_FILES["欄位名稱"]["type"]	取得上傳檔案類型，例如 text/plain（文字檔）、images/jpeg（JPEG 圖片檔）
$_FILES["欄位名稱"]["size"]	取得上傳檔案大小
$_FILES["欄位名稱"]["error"]	錯誤碼

錯誤碼說明：

編號	錯誤碼	說明
0	UPLOAD_ERR_OK	上傳正常，且完成上傳
1	UPLOAD_ERR_INI_SIZE	檔案 size 超過 php.ini 裡所設定的 [可上傳檔案的大小] upload_max_filesize

2	UPLOAD_ERR_FORM_SIZE	檔案 size 超過 php.ini 裡所設定的[可接受的檔案大小] MAX_FILE_SIZE
3	UPLOAD_ERR_PARTIAL	僅有部份檔案被上傳
4	UPLOAD_ERR_NO_FILE	檔案沒有被上傳
6	UPLOAD_ERR_NO_TMP_DIR	上傳檔案所需的 tmp 資料找不到
7	UPLOAD_ERR_CANT_WRITE	將檔案寫入磁碟時發生錯誤
8	UPLOAD_ERR_EXTENSION	所用版本 php 的某項延伸功能阻檔了檔案上傳

那麼，讓我們開始進行簡單的檔案上傳吧。

補充說明
若是使用 MacOs，請將 tmp 資料夾權限改為 755。 sudo chmod 755 tmp

單一檔案上傳

範例 7-5.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

<form name="myForm"
        action="./7-5-1.php"
        method="POST"
        enctype="multipart/form-data">
    <h3>請選擇所要上傳的檔案</h3>
    <input type="file" name="fileUpload" /><br />
    <input type="submit" value="送出資料" />
</form>
```

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

範例 7-5-1.php

```php
<?php
//判斷上傳是否成功 (error === 0)
if( $_FILES["fileUpload"]["error"] === 0 ) {
    //若上傳成功，則將上傳檔案從暫存資料夾，移動到指定的資料夾或路徑
    $isSuccess = move_uploaded_file(
        $_FILES["fileUpload"]["tmp_name"], //上傳暫存路徑
        "./tmp/".$_FILES["fileUpload"]["name"] //實際存放路徑與自訂檔名
    );

    //判斷上傳是否成功
    if( $isSuccess ) {
        echo "上傳成功!!<br />";
        echo "檔案名稱: ".$_FILES["fileUpload"]["name"]."<br />";
        echo "檔案類型: ".$_FILES["fileUpload"]["type"]."<br />";
        echo "檔案大小: ".$_FILES["fileUpload"]["size"]."<br />";
    } else { //檔案移動失敗，則顯示錯誤訊息
        echo "上傳失敗…<br />";
        echo "<a href='javascript:windows.history.back();'>回上一頁
</a>";
    }
}
```

如果希望檔案是自動建立，例如以「西元年月日時分秒」作為主要檔名，可以整合時間函式（請修改 7-5.php 的 form action 值為 7-5-2.php）：

範例 7-5-2.php

```php
<?php
//判斷上傳是否成功 (error === 0)
if( $_FILES["fileUpload"]["error"] === 0 ) {
    /**
     * 上傳檔案有副檔名的情況下，可以用簡單的方法取得副檔名
     */
    //1. 先取得完整的檔案名稱
    $fileName = $_FILES["fileUpload"]["name"];
```

```
    //2. 將完整檔案名稱依特定字元切割，轉為陣列
    $arr = explode(".", $fileName);
    //3. 特定字元切割後的陣列最後一個元素，即為副檔名
    $extension = $arr[ count($arr) - 1 ];

    /**
     * 使用時間函式，定義上傳檔案名稱
     * 用法 -> date("格式化字串")
     * 例如 -> date("Y-m-d H:i:s") -> 2021-12-31 13:14:00
     * Y: 西元年
     * m: 兩位數的月份
     * d: 兩位數的日
     * H: 兩位數(24 小時制)的時
     * i: 兩位數(24 小時制)的分
     * s: 兩位數(24 小時制)的秒
     */
    //1.建立時間字串
    $fileName = date("YmdHis");
    //將時間字串結合副檔名
    $fileName = $fileName . "." . $extension;

    //若上傳成功，則將上傳檔案從暫存資料夾，移動到指定的資料夾或路徑
    $isSuccess = move_uploaded_file(
        $_FILES["fileUpload"]["tmp_name"], //上傳暫存路徑
        "./tmp/".$fileName //實際存放路徑與自訂檔名
    );

    //判斷上傳是否成功
    if( $isSuccess ) {
        echo "上傳成功!!<br />";
        echo "檔案名稱: ".$fileName."<br />";
        echo "檔案類型: ".$_FILES["fileUpload"]["type"]."<br />";
        echo "檔案大小: ".$_FILES["fileUpload"]["size"]."<br />";
    } else { //檔案移動失敗，則顯示錯誤訊息
        echo "上傳失敗…<br />";
        echo "<a href='javascript:windows.history.back();'>回上一頁
</a>";
    }
```

```
}
```

八、函式

將程式碼中重複使用的部分，單獨寫成函式，在應用時 只要呼叫即可使用，不必重複撰寫。

格式

```
function 函式名稱 ([參數 1, 參數 2, 參數 3, ... ,參數 n]) {
  執行的程式內容;
  .....;
  [return 傳回值;]
}
```

呼叫方式

```
函式名稱([引數 1, 引數 2, 引數 3,..., 引數 n]);
```

函式的使用，有以下好處：

- 函式具有重複使用性，程式可以在任何地方進行呼叫即可使用，不必重複撰寫相同的程式碼造成困擾，提升程式效率。
- 函式的加入，會讓程式碼更為精簡、結構更為清楚，在閱讀或是維護上，會更加輕鬆。
- 若是函式中的程式產生錯誤，在修正時，只要針對函式內容進行修改，所有程式中呼叫的地方，即可正確執行。

函式基本用法

範例 8-1.php

```php
<?php
//定義一個未設定參數、沒有回傳結果，直接輸出結果的函式
function sayHelloWorld(){
    echo "Hello World!";
}

//執行 sayHelloWorld()
sayHelloWorld();
```

```php
echo "<hr>";

//定義一個有設定參數、有回傳結果的函式
function getGreeting($strMyName){
    $str = "你好，{$strMyName}。";
    return $str;

    //也可以寫成 return "你好，{$strMyName}。";
}

//透過引數的設定，回傳函式的執行結果
$strName = "Darren";
$strResult = getGreeting($strName);
echo $strResult;

echo "<hr>";

//定義一個變數相乘的函式
function get_X_times_Y($a, $b){
    return $a * $b;
}

//計算變數相乘後的結果
$x = 3;
$y = 4;
echo get_X_times_Y($x, $y);
```

區域變數和全域變數

- 一般來說，變數在指定值後，在整個程式中都能讀取並使用變數的值，我們稱為「全域變數」(global variable)；若是應用在函式（function）中的變數，僅能在函式的區域中使用，我們稱為「區域變數」(local variable)。
- 全域變數的有效範圍，僅限於主要程式中，不會影響到函式中的**同名變數**，也就是全域變數與區域變數互不侵犯；若要變數能由主要程式的範圍通透到函式中，就要將變數進行全域的宣告。

範例 8-2.php

```php
<?php
//這裡的全域變數 $msg，與區域變數 $msg 互不影響
```

```php
$msg = "這是全域變數<br />";

function showMsg(){
    //同樣的名稱，這裡的 $msg，不會影響到全域變數
    $msg = "這是區域變數<br />";
    echo $msg;
}

echo $msg; //先輸出全域變數的值
showMsg(); //透過函式輸出區域變數的值
echo $msg; //最後再輸出全域變數的值
```

如果在函式當中，想要使用相同名稱的全域變數，可以在函式的同名變數前面，加上「global」關鍵字。

範例 8-3.php

```php
<?php
$msg = "這是全域變數<br />";

function showMsg(){
    /**
     * 使用 global 關鍵字，
     * 此時函式內的 $msg，就代表全域(外部)變數的 $msg
     */
    global $msg;

    //修改全域變數的值
    $msg = "這是區域變數<br />";
    echo $msg;
}

//先輸出全域變數的值
echo $msg;

//函式中，使用 global 指令，將區域變數 $msg，變成全域變數
showMsg();

//最後輸出 $msg，發現全域 $msg 的值，在函式中被改變了
```

```
echo $msg;
```

九、COOKIE 與 SESSION

使用者在瀏覽網頁時，並不會一直與伺服器保持在連線的狀態下，當需要新資料，或是更新顯示內容時，都必須重新載入頁面，或是重新發出請求，但遇到在網站運作上有些需要「維持記憶」的狀況時，例如記住當前登入使用者的資訊，或是保持在購物車裡未結帳的商品，以供下次繼續使用。

Cookie

指某些網站為了辨別使用者身分而儲存在用戶端（Client Side）上的資料（通常經過加密），是一個小型的文字檔案。當瀏覽者開啟網站時，即可在程式的設定下，將指定的檔案資料儲存在用戶端電腦中，並可設定該資料的有效時間、存放路徑與有效網域。

Cookie 的特性有：

- 以檔案型態，儲存在客戶端（瀏覽者的電腦）中
- 瀏覽器只能儲存使用 300 個 cookie
- 每個 cookie 最多僅有 4k Bytes 的容量
- 瀏覽器可以關掉 cookie 功能，只是可能造成 cookie 無法使用
- 藉由 request headers 帶給伺服器
- 使用 setcookie() 函式前，不能有 echo 或 print 任何字串

儲存 cookie 資料

格式
setcookie(名稱[, 值] [, 有效時間] [, 儲存路徑] [, 網域] [, 安全性]);

參數	說明
名稱	cookie 儲存時的名稱。
值	cookie 儲存值，並以 URL 編碼，由於是以明碼的方式儲存，所以不建議儲存重要資料。
有效時間	cookie 的有效時間，格式為時間戳記。若指定的時間為過去，會刪除 cookie。
儲存路徑	cookie 的有效路徑，不設定時，預設為 <php.ini> 的 session.cookie_path 設定值。
網域	cookie 的有效網域。

安全性	1: 只有 HTTPS 才能傳送 cookie。 0: 能使用 HTTP 傳送 cookie ，也是忽略時的預設值。

用法

```php
<?php
//基本儲存方式，結束時間預設為瀏覽階段結束
setcookie('TestCookie_01', 'Hello World 01');

//設定 cookie 存活時間為 1 個小時
setcookie('TestCookie_02', 'Hello World 02', time() + 3600);

//設定名稱、值、有效時間、暫存路徑、使用網域、安全性
setcookie('TestCookie_03', 'Hello World 03', time() + 3600, "/", "localhost", 0);

//設定到期時間(透過 mktime() 設定 2019 年 12 月 31 日 23 點 59 分 59 秒過期)
//格式: mktime(時, 分, 秒, 月, 日, 年);
setcookie('TestCookie_04', 'Hello World 04', mktime(23, 59, 59, 12, 31, 2020));

//設定到期時間(透過 strtotime() 設定 時間字串 '2020-12-31-23-59-59')
setcookie('TestCookie_05', 'Hello World 05', strtotime('2020-12-31-23-59-59'));

//可以使用 $_COOKIE 來存取 cookie 的資料：
echo $_COOKIE['TestCookie_01']; //輸出 Hello World 01

//可以使用陣列的格式來儲存 cookie
setcookie("students[name]", "Darren");
setcookie("students[age]", 18);
setcookie("students[height]", 150);
setcookie("students[weight]", 100);

//輸出以陣列格式儲存的 cookie
echo $_COOKIE['students']['name']; //輸出 Darren
```

建立 Cookie

範例 9-1.php

```php
<?php
$cookie_name = "username";
$cookie_value = "darren";
```

```php
// 86400 = 1 天， 86400 * 15 = 15 天
setcookie($cookie_name, $cookie_value, time() + (86400 * 15), "/");
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
<?php
//如果 cookie 不存在，則顯示尚未設定
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie '{$cookie_name}' 還沒有設定…";
} else { //若是設定，則顯示 cookie 內容
    echo "Cookie '{$cookie_name}' 已經設定。<br>";
    echo "Cookie '{$cookie_name}' 的值是: {$_COOKIE[$cookie_name]}";
}
?>
</body>
</html>
```

刪除 Cookie

範例 9-1-1.php

```php
<?php
//刪除 cookie (透過指定過去時間，例如 3600 秒前)
setcookie('username', '', time() - 3600);
```

Session

一個小型文字的檔額，儲存瀏覽者與伺服器連線的工作期間，所保持的狀態和資訊。它的使用時間是在開啟瀏覽者後進入啟動 session 機制的網站開始，只要瀏覽器沒有關閉，回到原網站時，會發現原來的 session 仍然有效。

補充說明

session 檔案在 xampp 中的預設存放路徑： session.save_path = "C:\xampp\tmp"

當使用者使用瀏覽器連線到伺服器網站時，網站伺服器會自動派發一個 Session ID 給這次的連線動作，網站程式即可依照這個 Session ID 分辨使用者來處理所儲存的狀態。事實上，在預設的狀態下，Session ID 會將伺服器所派發的 Session ID，以 Cookie 的方式，儲存在用戶端來記錄狀態，同一個網站的不同網頁，可以藉由這個 Cookie 來維持同一個 Session ID 的狀態。這個 Cookie 並沒有到期時間，所以預設會在瀏覽器關閉時同時消失。

圖: 儲存在 Cookie 中的 Session ID

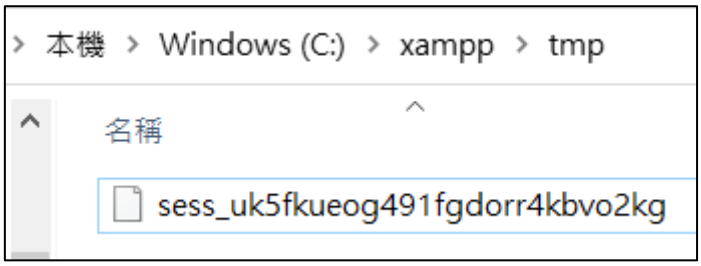

圖: session 的檔案名稱

用法
`<?php` `//啟動 session`

```php
session_start();

//儲存 session 資料
$_SESSION['myName'] = 'Darren';

//取得 session 資料
$myName = $_SESSION['myName'];

//輸出 session 資料
echo $_SESSION['myName']; //輸出 Darren

//刪除 session 資料
unset( $_SESSION['myName'] );

//除了刪除所有 session 資料，同時刪除 server 上的 session 檔案
session_destroy();
```

使用 session

範例 9-2.php

```php
<?php
//啟動 session
session_start();
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
<?php
//儲存 session 資料
$_SESSION['myName'] = 'Darren';

//取得 session 資料
```

```php
$myName = $_SESSION['myName'];

echo "我的名字: " . $_SESSION['myName'] . " [從 session] <br />";
echo "我的名字: " . $myName . " [從變數]";

//設定 session 陣列
$_SESSION['student']['name'] = 'Darren Yang';
$_SESSION['student']['age'] = 18;
$_SESSION['student']['height'] = 171;
$_SESSION['student']['weight'] = 80;

echo "全名: ".$_SESSION['student']['name']."<br />";
echo "年齡: ".$_SESSION['student']['age']."<br />";
echo "身高: ".$_SESSION['student']['height']."<br />";
echo "體重: ".$_SESSION['student']['weight']."<br />";
?>
</body>
</html>
```

關閉 session

範例 9-2-1.php

```php
<?php
//開啟當前的 session，在關閉 session 前，要先開啟
session_start();

//關閉、刪除 session
session_destroy();
```

接下來模擬一個基本的登入功能。

簡易登入功能

範例 9-3.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

<form name="myForm" method="post" action="./9-3-1-login.php">
    帳號: <input type="text" name="username" value="" /><br />
    密碼: <input type="password" name="pwd" value="" /><br />
    <input type="submit" value="登入" />
</form>


</body>
</html>
```

確認是否正確登入

範例 9-3-1-login.php

```
<?php
//啟動 session
session_start();

//預設帳號密碼，測試 session 登入用
$username = 'test';
$pwd = sha1('test'); //sha1 雜湊後的字串

//判斷 username 與 pwd 是否同時存在
if( isset($_POST['username']) && isset($_POST['pwd']) ){
    //判斷帳號、密碼是否為正確
    if( $_POST['username'] === $username && sha1($_POST['pwd']) === $
pwd){
        //3 秒後跳頁
        header("Refresh: 3; url=./9-3-2-admin.php");

        //將傳送過來的 post 變數資料，放到 session，
        $_SESSION['username'] = $_POST['username'];

        echo "登入成功!!! 3 秒後自動進入後端頁面";
```

```
    } else {
        //關閉 session
        session_destroy();

        //3 秒後跳頁
        header("Refresh: 3; url=./9-3.php");

        echo "登入失敗…3 秒後自動回登入頁";
    }
} else {
    //關閉 session
    session_destroy();

    //3 秒後跳頁
    header("Refresh: 3; url=./9-3.php");

    echo "請確實登入…3 秒後自動回登入頁";
}
```

登入後的管理/主要頁面

範例 9-3-2-admin.php

```
<?php
//啟動 session
session_start();

//判斷是否登入 (確認先前指派的 session 索引是否存在)
if( !isset($_SESSION['username']) ) {
    //關閉 session
    session_destroy();

    //3 秒後跳頁
    header("Refresh: 3; url=./9-3.php");
    echo "請確實登入…3 秒後自動回登入頁";
    exit();
}
?>
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

這裡是後端管理頁面 - <a href="./9-3-3-logout.php?logout=1">登出</a>

<hr>

<img src="https://www.dora-
world.com.tw/upload/2020_02_12_1744480.jpg">

</body>
</html>
```

登出功能

範例 9-3-3-logout.php

```
<?php
//開啟 session
session_start();

//判斷是否要登出
if(isset($_GET['logout']) && $_GET['logout'] == '1'){
    //關閉 session
    session_destroy();

    //3 秒後跳頁
    header("Refresh: 3; url=./9-3.php");
    echo "您已登出…3 秒後自動回登入頁";
} else {
    //關閉 session
    session_destroy();
}
```