

PHP 入門

參考專案

<https://github.com/shinder/mmmh57-php>

<https://bitbucket.org/lsd0125/mfee23-php>

開發執行環境

包含 Apache + MySQL + PHP (Apache module)

Mac 平台可以使用：

XAMPP (https://www.apachefriends.org/zh_tw)，使用 ~160M 不是 VM 的版本

MAMP (<https://www.mamp.info>)

Windows 平台可以使用：

XAMPP (https://www.apachefriends.org/zh_tw)

WAMP (<http://www.wampserver.com/en/>)

編輯環境（可選擇習慣的編輯軟體）

Visual Studio Code (免費軟體)

PhpStorm (付費軟體)

第一步測試

用文字編輯器將下列兩行存成 XAMPP/htdocs/phpinfo.php

```
<?php  
phpinfo();
```

用瀏覽器查看 <http://localhost/phpinfo.php>。找出以下的設定值：

Loaded Configuration File

Server Root

MYSQL_HOME

DOCUMENT_ROOT

網路基本

埠號 (port number)：虛擬的對外窗口編號，一個埠號只能讓一個應用程式（服務）偵聽。

區域網路網段：192.168.xxx.xxx，10.xxx.xxx.xxx

查詢 ip 使用：

- ipconfig (windows console)
- ifconfig (linux, mac os X)

Apache 的設定

主要設定檔：/Applications/MAMP/conf/apache/httpd.conf

附屬的設定檔：/Applications/MAMP/conf/apache/extra

PHP 的設定

設定檔位置，為 phpinfo 所描述的位置

例如：/Applications/MAMP/bin/php/php5.5.26/conf/php.ini

```
error_reporting
max_execution_time = 30
max_input_time = 60
memory_limit = 128M
file_uploads = On
upload_tmp_dir = /Applications/MAMP/tmp/php
upload_max_filesize = 32M
post_max_size = 32M
```

PHP 官網及說明文件

<http://php.net/>

建立專案

一般專案的建立方式

在 Document Root 開立一個專案的資料夾，一般命名不使用中文或奇怪的符號。通常使用英文字母數字，dash 或 underscore。

PHP 程式碼的標示

以 **<?php** 開頭，**?>** 結束。當 php 程式之後沒有其它 HTML 內容時，php 程式的結束標記 **?>** 可以省略。

PHP 的運算

PHP 一開始就是 WWW 伺服端的技術，用來撰寫動態網頁，其善長處理「文字」資料。

HTML 內容本來就是文字，許多資料在輸出到頁面時也會自動轉換成文字。

PHP 是 C-like 程式語言，程式裡放置空白（換行）相當有彈性，應確實標上分號（程式敘述結束符號）及遵守縮排規則，以利程式撰寫及除錯。

常數

- 所謂常數是不可變更的值。PHP 的常數可以分成三類：一般常數（例：17）、系統常數（例：PHP_VERSION）及自訂常數。
- 一般常數就是一些數值：0、-9、2.5、'abc' 等。
- 系統常數為 PHP 預設的常數，常用的有：PHP_VERSION（查看 PHP 版本）、__FILE__（PHP 檔的實體路徑）、TRUE 和 FALSE（布林值的 true 和 false）。系統常數大部份可以由 get_defined_constants() 函式取得（取得的資料是陣列類型）。
- 自訂常數是程式開發者所定義的常數，可以使用 define() 函式定義，語法為：
-

`define(常數名稱, 常數內容, 常數名稱是否不區分大小寫);`

- 常數名稱通常使用全部大寫的英文字，複合字時中間使用 underscore（_）分開。
- 常數內容則可以是數值或字串。
- 常數名稱大都使用全大寫，「是否不區分大小寫」一般是不使用的。

```
<?php
define("MY_CONSTANT", "常數 - 值在設定後不可變更");
echo MY_CONSTANT;
```

- PHP 程式的每一句敘述請標上分號（;），分號表示程式敘述的結束。只有在最後一行的程式敘述（?> 之前）可以不用標分號，其餘都應該要標示。

變數

- PHP 的變數在使用前可以不用宣告，其為自動型別，不需要限定變數只能使用某個型別。PHP 變數有個特色，就是在變數前必須要有個美金符號（\$），而且變數名稱是有大小寫的區別的。
- 自訂的常數前不使用 \$，而且通常所有字母為大寫。變數前必須要有 \$，有區分字母大小寫。
- PHP 變數的命名規則和一般程式語言一樣，第一個字母不可以是數字。

```
<?php
$my_var = 66;
$b = "22";
$c = "abc";
echo $my_var + $b; // 88
echo "<br />";
echo $my_var + $c; // 66
```

- PHP 的變數有個特點就是可以使用「以變數值為名的變數」(variable variables)。

```
<?php
$n = "name";
$$n = "shinder";
echo $name;
```

- 變數在設值之後，若不想再使用可用 unset() 刪除。

字串

- PHP 的字串標示方式使用一對雙引號（"）或一對單引號（'），不過兩者在使用上有所不同，雙引號標示者裡面若包含變數，會顯示變數值；單引號標示者則否。

```
<?php
$a = "Victor";
echo "Hello, $a";
echo '<br />';
echo 'Hello, $a';
```

- 雙引號標示中，有時輸出變數內容情況會比較複雜，此時可以使用大括號解決。

```
<?php
$a = "Victor";
echo "Hello, $a123 <br/>";
echo "Hello, {$a}123 <br/>";
echo "Hello, ${a}123 <br/>";
echo $a . '<br/>';
```

- 點符號（.）在 PHP 用來做字串串接，不是使用加號（+），這點要特別注意。

- 雙引號和單引號在使用脫逸字元時，亦不太一樣(執行後請查看頁面原始內容)。

```
<?php
$a = "xyz\nabc\"def\'ghi\\";
$b = 'xyz\nabc\"def\'ghi\\"';
echo $a;
echo $b;
```

- PHP 的字串還有個好處，就是可以換行標示，所以方便定義較長且包含換行的字串。若文字內容也包含雙引號時，可以使用「即為文件」(Heredoc)表示法。Heredoc 表示法是以 <<< 為開頭，緊接「標示名」(可自訂，通常為大寫字母)，加換行。Heredoc 結束時，在新的一行使用「標示名」即可。

```
<?php
$a = "PHP - MySQL
是好朋友!";
$b = <<<HDOC
<br>
<h1>PHP - </h1>
<div style="color:#F00;">MySQL</div>
HDOC;
echo $a;
echo $b;
```

運算子

- PHP 的運算子和其它 C-like 的程式語言大部份是相同的，在此我們來複習一下。算術運算子除了加 + 減 - 乘 * 除 / 四則運算外，還包括了求餘數 %。這裡的四則運算和數學的四則運算是相同的。

運算子	語法	說明	\$a=7, \$b=2 的運算結果
+	\$a + \$b	\$a 加 \$b	9
-	\$a - \$b	\$a 減 \$b	5
*	\$a * \$b	\$a 乘 \$b	14
/	\$a / \$b	\$a 除以 \$b	3.5
%	\$a % \$b	\$a 除以 \$b 的餘數	1

- 「\$a = \$a + 1」有遞增運算子可以簡化為「\$a++」。「\$a = \$a + 3」也有替代運算子，稱為算術指定運算子。

算術指定運算子	一般算術並指定的運算式	簡化後的運算式
+=	\$a = \$a + \$b	\$a += \$b
-=	\$a = \$a - \$b	\$a -= \$b
*=	\$a = \$a * \$b	\$a *= \$b
/=	\$a = \$a / \$b	\$a /= \$b
%=	\$a = \$a % \$b	\$a %= \$b

- 「關係運算子」(relational operator)或稱為比較運算子。

關係運算子	說明	範例	範例結果
>	是否大於	5 > 2	true
>=	是否大於等於	5 >= 2	true
<	是否小於	5 < 2	false
<=	是否小於等於	5 <= 2	false
==	是否等於	5 == 2	false
!=	是否不等於	5 != 2	true
===	是否等於 (型別必須相同)	5 === "5"	false
!==	是否不等於	5 !== "5"	true

- 條件運算子「?:」是唯一的三元運算子，其語法如下：

判斷值？真時選擇值：假時選擇值

第一個運算元(判斷值)應該為布林值，或是結果為布林值的運算式。當判斷值為 true 時，整個運算式的結果取真時選擇值;反之，判斷值為 false 時，取假時選擇值。通常我們會將整個條件運算結果，存放到某個變數內，所以會如下式：

變數 = 判斷值？真時選擇值：假時選擇值;

- 邏輯運算式和關係運算式有一個共同點，兩者的運算結果都是布林值，不是 true 就是 false。然而，邏輯運算式中的運算對象(運算元)，也是布林值。

運算子	意義	語法
&&	且，and	\$a && \$b
and	且，and	\$a and \$b
	或，or	\$a \$b
or	或，or	\$a or \$b
!	非，not	! \$a

- 運算子依性質不同分類，優先權高的類別會被先列出來，越往下優先權越低：
 - 1 單元運算子
 - 2 算術運算子
 - 3 關係運算子
 - 4 邏輯運算子
 - 5 指定運算子
- 除了上述的運算子，分隔符號中的括號「()」、逗號「,」也和運算式有關係。括號可以提升運算子和運算元的運算關係，以括號包起來的運算式會優先運算。逗號可用於分隔宣告變數(或變數宣告的指定式)，並不會影響任何值。

URL 變數 (GET 參數)

- 網址內容裡？之後為 GET 參數，其格式稱為 URL-encoded。PHP 可以使用 `$_GET` 預設變數取得其內容。

```
<?php  
echo $_GET['a'] + $_GET['b'];
```

- 測試網頁之後，在網址列輸入下式？之後的內容。
`http://localhost/my_test/untitled1.php?a=12&b=24`

PHP 的流程控制

流程控制通常可以分成兩類：選擇敘述和迴圈敘述。選擇敘述包含 if/else 和 switch，迴圈敘述包含 for、while、do/while、及 foreach/as。

if/else 敘述基本用法

簡單地講，if 敘述是選擇執行或不執行，if/else 敘述則是二選一執行。以下是 if/else 選擇結構的語法：

```
if(條件式) {  
    //區塊敘述一  
} else {  
    //區塊敘述二  
}
```

if/else 語法意義為：若條件式為 true 時，執行區塊敘述一，否則（條件式為 false 時）執行區塊敘述二。視需要也可以省略 else 區塊。

```
<?php
    $a = 0;
    if( isset($_GET['a']) ){
        $a = $_GET['a'];
    }
    echo $a;
```

一般的想法是 PHP 的程式碼是嵌在 HTML 的，其實剛好是相反的，我們可以把 HTML 的內容看成是透過 echo 輸出的。

```
<body>
<?php
    if(isset($_GET['name']) and
        $_GET['name'] == 'shinder') {
?>
    <p>您好! Shinder<br />
    歡迎光臨~
    </p>
<?php
    } else {
?>
    <p>您好! 陌生人<br />
    您無權使用管理系統哦!
    </p>
<?php
    }
?>
</body>
```

if/else 多層巢狀結構

if 和 if/else 的基本使用方式是，選或不選，及二選一。如果現在需要一個轉換成績的程式，要把 0~100 分別轉換成 A~F 級。90 分及其以上為 A，80~89 分為 B，70~79 分為 C，60~69 分為 D，59 分及其以下為 F。現在可不是選不選、或二選一的問題，而是要以某個數值成績的範圍來決定成績的等級，這也是個多選一的問題。

```
<?php
    $n = $_GET['n'];
    if ($n >= 90) {
        $g = 'A';
    } else {
        if ($n >= 80) {
            $g = 'B';
```



```
        } else {
            if ($n>=70) {
                $g = 'C';
            } else {
                if ($n>=60) {
                    $g = 'D';
                } else {
                    $g = 'F';
                }
            }
        }
    }
    echo $g;
```

省去 else 後 block 的大括弧

```
<?php
$n = $_GET['n'];
if ($n>=90) {
    $g = 'A';
} else if ($n>=80) {
    $g = 'B';
} else if ($n>=70) {
    $g = 'C';
} else if ($n>=60) {
    $g = 'D';
} else {
    $g = 'F';
}
echo $g;
```

else if 也可以黏在一起，寫成 elseif。

switch 選擇敘述

if/else 的巢狀結構可以解決多重選擇的問題，在這則是介紹專門設計給多選一使用的 switch 敘述。switch 並不使用條件式來決定執行的區段敘述，而是使用鍵值（通常為變數）。

```
switch (鍵值){
    case 條件值 1:
        // 區段 1敘述
        break;
    case 條件值 2:
        // 區段 2敘述
        break;
    .....
    case 條件值 N:
        // 區段 N敘述
```

```

        break;
    default:
        // default區段敘述
}

```

執行 switch 敘述時，會先求得鍵值，接著將鍵值和條件值 1 做比對。如果兩值相等，則執行區段 1 敘述，接著遇到 break 之後跳離 switch 敘述；如果兩值不相等，則再往下比對條件值 2，以此類推。如果鍵值和所有的條件值都不相等時，則執行「default:」下的 default 區段敘述。default 部份在 switch 敘述中是選擇性的，可有可無，不一定要放在最後面。

```

<?php
    $page = '';
    if( isset($_GET['page']) )
        $page = $_GET['page'];

    switch( $page ) {
        default :
        case 'main':
            echo 'main page';
            break;
        case 'content':
            echo 'content page';
            break;
        case 'about':
            echo 'about page';
            break;
    }

```

for 迴圈

for 迴圈的語法如下：

```

for (起始式; 條件式; 步進式) {
    //區塊內敘述
}

```

起始式—進入迴圈時，一開始執行的程式運算式，只執行一次。通常起始式，多為設定控制迴圈執行變數的起始值。

條件式—判斷是否執行迴圈內區塊的依據。如果條件式的結果為 true，則繼續執行；如果為 false，則跳離迴圈。

步進式—每經過一次迴圈，就會執行一次的運算式。通常步進式，多為設定控制迴圈執行變數的遞增或遞減運算式。

```

<table border="1" width="400px" cellpadding="0" cellspacing="0">

```

```

        <tr>
        <?php
        for($i=0; $i<10; $i++) {
            echo '<td>' . $i . '</td>';
        }
        ?>
        </tr>
    </table>

```

while 迴圈

while 迴圈和 for 迴圈很像，也可以說是 for 迴圈的簡化版，基本上 while 迴圈標頭只有用以判斷的條件式，而沒有起始式和步進式。

```

//起始式
while(條件式){
    // 區塊內敘述
    // 步進式
}

```

條件式若為 true，則執行大括弧內的區塊內敘述，執行完之後，回到前面再判斷條件式，若還是為 true 則再執行一次區段敘述，直到條件式為 false 時才跳離迴圈。

使用迴圈的時候都必須注意無窮迴圈，尤其是 while 迴圈。while 的控制變數並不像 for 可以直接放在敘述的標頭，所以可能會忘了加步進式，這個時候就會形成無窮迴圈。

條件式直接使用 true 是在「不知道要跑幾次迴圈」時使用，不過還是可以搭配跳離迴圈的關鍵字 break 來使用，讓它不至於無窮的執行下去。

```

while(true){
    // 區塊內敘述
}

```

從 mySQL 的資料集（Recordset）取出資料就常使用 while 迴圈，資料集取完後會得到 NULL（空值），所以不需要控制變數就可以讓迴圈正常停止。

do/while 迴圈

do/while 迴圈結構和之前介紹的兩種不太相同。

```

do{
    // 區塊內敘述
} while (條件式);

```

do/while 結構以 do 關鍵字為開頭，接著是大括弧包起來的區塊內敘述，最後是 while 關鍵字和小括弧包著的條件式。

請注意，for 和 while 迴圈都是以右大括弧結尾，所以不需要再加分號(;)標示敘述的結束。但是，do/while 結尾是右小括弧，並不能代表一個區塊的結束，因此要加上分號，以標示迴圈結尾。

若拿 while 迴圈和 do/while 迴圈做比較，最明顯的不同就是條件式的位置。do/while 的條件式擺在結構的最後面，也就表示先執行區塊敘述一次，再做條件式判斷，這也可以說是 do/while 的特色。

break 與 continue

關鍵字 break 和 continue，可以改變迴圈的流程。break 曾經在使用 switch 敘述時看到，不過在這將介紹 break 和迴圈的關係。

在 switch 選擇敘述中，我們使用過 break，讓流程跳離 switch 敘述區塊。而在 for、while、do/while 迴圈中，break 的作用也是相同的，都是跳離敘述區塊（跳離迴圈的大括號範圍）。一般使用 break 是為了在特殊情況下提早跳離重複結構。

continue 的功能是跳到迴圈的起始處。

```
<?php
for ($j = 1; $j<7; $j++) {
    if ($j == 4)
        continue;
    echo $j;
}
```

PHP 的陣列

何謂陣列呢？簡單地講，陣列就是「多個擁有相同名稱的變數集合」。PHP 的陣列又可細分成「索引式陣列」（Indexed Array）和「關聯式陣列」（Associated Array）。在 PHP 索引式和關聯式是可以混用的，不過依它們的特性我們先分開來討論。

索引式陣列

元素都有相同的名稱，為了區別每個陣列元素，它們都有各自的位置編號。使用陣列時，一般用 array() 函式建立函式的實體方式如下：

```
$ar = array(3, 2, 2, 0, 4, 1);
```

為了指出是哪個陣列元素時，必須以陣列名稱，後接一對中括弧[]，中括弧內放入位置編號。位置編號一般也稱為索引，索引由 0 開始，接著是 1、2、3...至元素個數減一。陣列的索引應該為大於或等於 0 的整數，或者結果為整數的運算式。

```
echo $ar[1];           // 2
```

PHP 的陣列是動態的，可以動態加入元素或移除元素，而且元素沒有限定類型。用 [] 直接「建立陣列並放入第一個元素值」或「推入一個元素值」：

```
<?php
    $br[] = 3;
    $br[] = 2;
    $br[] = 2;
    $br[] = 0;
    $br[] = 4;
    $br[] = 1;
    print_r($br);
```

`print_r()` 和 `var_dump()` 常用來查看資料內容和除錯。

`count()` 函式能取得陣列裡元素的個數。

```
<?php
    $ar = array(9,5,2,'abc', true);
    for($i=0; $i<count($ar); $i++) {
        echo " {$ar[$i]} <br />";
    }

?>
```

PHP 的內建函式名稱，雖然大部份會用有意思的名稱，有些則用縮寫容易讓人混淆。而且由於發展的歷史，大部份的函式是全域的，而不是透過物件導向的方式歸類在某些類別下。建議初學者常查看 PHP 手冊，或在 php.net 的「search for」進行查詢。

若要不理會索引值是否連續，依序取得陣列的元素值，可以使用 `foreach/as` 迴圈。

```
foreach(陣列變數 as 列舉變數) {
    //迴圈主體內容
}
```

列舉變數是自訂變數，由開發者自行決定。列舉變數會依迴圈執行的圈數，依序取得元素值。

```
<?php
    $ar = array(1,2,3);
    $ar[9] = 7;
```

```
foreach($ar as $v) {
    echo "$v <br />";
}
```

關聯式陣列

PHP 的關聯式陣列是由一個字串 key 對應到一個 value。

關聯式陣列可以看成是 hash table（雜湊表）的實作，只要給一個 key（屬性名稱）就可以得到對應的 value（屬性值）。

PHP 的關聯式陣列同樣用 array() 函式建立，key 和 value 之間用胖箭頭「=>」來設定。「=>」由一個等號和大於符號組成，注意中間沒有空白。

```
<?php
    $ar = array(
        3      => 'abc',
        "3"    => 'def',
        'name' => 'shinder',
        "pw"   => "pass" );

    echo $ar[3] . "<br/>";
    echo $ar["3"] . "<br/>";
    echo $ar['name'] . "<br/>";
```

注意，使用關聯式陣列時，通常 key 不使用數值，使用數值會變成使用索引式陣列。key 若可以轉換成數值，即使用雙引號標示，同樣視為相同，所以「3」和「" 3"」視為相同的 key，key 是唯一的。

關聯式陣列也可以用 foreach/as 迴圈來取值，其語法如下：

```
foreach(陣列變數 as 鍵變數 => 值變數) {
    //迴圈主體內容
}
```

「鍵變數」和「值變數」同樣使用「=>」指明其關係。

```
<?php
    $ar = array(
        3 => 'abc',
        "3" => 'def',
        'name' => 'shinder',
        "pw" => "pass" );

    foreach($ar as $k=>$v) {
        echo "$k : $v<br />";
    }
```

陣列的「=」設定

PHP 陣列在使用「=」做設定運算時，並不是設定參照（reference），而是做實體的複製。若要設定參照，必須使用 & 符號。

```
<pre>
<?php
    $ar = array(
        "num" => 17,
        'name' => 'shinder',
    );

    $br = $ar;
    $cr = &$ar;
    $ar['num'] = 26;

    print_r($ar);
    print_r($br);
    print_r($cr);

?>
</pre>
```

\$_GET 和 \$_POST

PHP 用來取出客戶端傳來資料的 \$_GET 和 \$_POST 是以陣列的形式存在。以下的例子在執行時，試著在網址後接著輸入「?a=123&bb=yes」，a 和 bb 會是 \$_GET 的 key，而 123 和 yes 分別是 \$_GET 的 value。

```
<pre>
<?php
    print_r($_GET);

?>
</pre>
```

測試登入表單。

```
<!DOCTYPE html>
<html>
<head>
    <title>example</title>
    <meta http-equiv="content-type" content="text/html; charset=utf-8">
</head>
<style type="text/css">.titleCol { float:left; width:100px; }</style>
<body>
<?php
if( isset($_POST['email']) ) {
    echo '<pre>' . print_r($_POST, true) . '</pre>';
} else {
```

```
?>
<form name="form1" method="post" action="">
  <div class="titleCol">Email:</div>
  <input type="text" id="email" name="email" />
  <br />
  <div class="titleCol">Password:</div>
  <input type="password" id="password" name="password" />
  <br />
  <div class="titleCol">&nbsp;</div>
  <input type="submit" value="登入" />
</form>
<?php
}
?>
</body>
```

\$_REQUEST 的功能是？

如何用 Chrome 觀察表單的 get 和 post 方法？

PHP 的函數

函式（function）就是有某種功能的小程式，可以看成是一個神奇的盒子，輸進一些資料後，可以得到處理過後的結果。一個程式是為了解決某個大問題，要一下子解決一個大問題並不容易著手，把大問題分開成許多小問題，再個個擊破，就比較容易。使用函式就是把問題分開個別解決。

之前使用過的 `phpinfo()`、`array()`和 `count()`都是 PHP 內訂的函式。

自訂函式

自訂函式的語法以 `function` 關鍵字為開頭：

```
function 函式名(參數 1, 參數 2, ...) {
    // 函式的內容敘述
    return 回傳值;
}
```

```
<?php
    function multi($a, $b=10) {
        return $a * $b;
    }
    echo multi(6, 7) . '<br />';      // 42
    echo multi(8) . '<br />';        // 80
```


以參照為參數。

```
<?php
    function swap(&$a, &$b) {
        $c = $a;
        $a = $b;
        $b = $c;
    }
    $m = 100;
    $n = 'abc';
    swap($m, $n);
    echo "$m $n";
```

函式外的變數

在 PHP，函式內不能直接使用函式外設定的變數，若要使用必須在函式內宣告變數為**全域變數**，才能使用。這樣的做法是，儘量避免在函式內使用外面的變數，可以增加函式的獨立性和再用性。

```
<?php
    $g1 = 1000;
    $g2 = 2000;

    function fun() {
        global $g2;
        echo ">{$g1}< >{$g2}<";
    }
    fun();
```

PHP 的類別

類別可以看成是設計圖，依照設計圖實作出來的就是物件。類別也可以想成是使用者定義的資料類型，而物件是以類別宣告出來的變數。

簡單類別

若先不考慮繼承，類別的定義語法如下：

```
class 類別名稱 //類別定義的標頭
{
    // 屬性宣告
    // 建構函式定義
    // 方法定義
}
```

屬性就是物件的變數。屬性的宣告可以用修飾字 `var`、`public` 和 `private`。 `var` 和 `public` 的功能是相同的，都是定義可以公開使用的屬性；`private` 則是定義私有屬性，只能在類別裡使用。下式用來說明 `public` 和 `private` 的區別。

```
<?php
class Person {
    var $name;
    public $mobile;
    private $sno = 'secret';
}

$p = new Person;
$p->name = "Shinder Lin";
$p->mobile = "0918123456";
// $p->sno = "123"; // 取用私有屬性將發生錯誤
print_r($p);
echo $p->name;
```

一個 PHP 檔可以定義許多類別。

PHP 5.3 以前的類別沒有 `package` 之分，需要時 `include` 或 `require` 引入即可。

方法

「方法」其實就是函式，主要區別是：方法屬於物件，函式則是全域的。在修飾字方面，使用 `private` 表示私有方法，否則為公開方法。類別定義中，除了「宣告並指定屬性的敘述」可以直接放在類別內，其餘敘述都應該放在方法內。

```
<?php
class Person {
    private $name;
    function setName($n) {
        $this->name = $n;
    }
    function getName() {
        return $this->name;
    }
}
$p = new Person;
$p->setName("Victor");
echo $p->getName();
```

建構函式及解構函式

建構函式是物件在使用 `new` 建立之後，會自動被呼叫的函式，用來做物件的初始設定之用。只要定義一個名稱和類別相同的 `function` 就是建構函式，然而在 PHP5 建議使用

「__construct」。PHP 5 在物件被刪除前會呼叫解構函式「__destruct」，有時資料庫的操作在物件刪除後就要關閉，即可在解構函式中進行關閉動作。

```
<?php
class Person {
    var $name;
    function __construct($n) {
        $this->name = $n;
        echo $this->name . '建立<br />';
    }

    function __destruct() {
        echo $this->name . '解構<br />';
    }
}
$p = new Person('p');
unset($p);
$q = new Person('q');
```

unset() 刪除物件前，解構函式會被呼叫；程式結束前（刪除所有物件前）所有物件的解構函式都會被呼叫。

類別方法

有些方法沒有用到 \$this，而且其目的是用來直接呼叫的，稱為類別方法。類別方法定義方式和一般方法一樣，不同處在於呼叫方式，使用「::」。

```
<?php
class MyMath {
    function distance($x1, $y1, $x2, $y2) {
        $dx = $x1 - $x2;
        $dy = $y1 - $y2;
        return sqrt($dx * $dx + $dy * $dy);
    }
}
echo MyMath::distance(1, 1, 11, 11);
```

Cookie

Cookie 是網站透過瀏覽器將資料存放在硬碟裡的機制。基於安全性，Cookie 有幾個特性：

1. 以網域為單位：例如 pchome.com.tw 不可讀取 yahoo.com.tw 所設定的 Cookie。
2. 大小受限：不同的瀏覽器有不同的大小限制，一般為 4096 bytes。
3. 客戶端可能可以看得到：IE 的 Cookie 會放在「C:\Documents and Settings\使用者\Local Settings\Temporary Internet Files」。

4. **客戶端可以設定**：瀏覽器可以設定是否使用 Cookie，一般設定為使用，筆者也建議使用。
5. **過期**：Cookie 是有期限的，過期就會失效。
6. **自動傳給主機**：瀏覽器每次做 request 時，都會將有效的 Cookie 放在 HTTP Header 裡，傳給 server。

Cookie 的內容應該要簡短，而且不應存放敏感性資料（例如：密碼）。Cookie 是好的，可以幫助網站做客製化的設定，大部份的 Session 機制也是建立在 Cookie 之上。

PHP 設定 Cookie 使用 `setcookie()` 函式：

```
setcookie("Cookie變數名稱","Cookie數值","期限","路徑","網域","安全");
```

通常我們只會用到前三個參數：「Cookie 名稱」、「Cookie 值」和「期限」。「期限」用的是 Unix 系統上的時間戳記（Unix timestamp），單位為秒，用 `time()` 函式可以取得當下的時間戳記。若不使用「期限」參數，瀏覽器的所有視窗關閉時為過期時間。

PHP 讀取 Cookie 使用 `$_COOKIE` 預設的陣列變數。

```
<?php
    if(! isset( $_COOKIE['mycookie'] ) ){
        setcookie("mycookie", "10", time()+30);
    } else {
        $c = $_COOKIE['mycookie'];
        setcookie("mycookie", $c+1, time()+30);
    }
?>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>Cookie</title>
</head>
<body>
    <?php
        if(! isset( $_COOKIE['mycookie'] ) ){
            echo '第一次設定 Cookie, 或重新設定';
        } else {
            echo $_COOKIE['mycookie'];
        }
    ?>
</body>
</html>
```

第一次用 `setcookie()` 設值時，用 `$_COOKIE` 是讀不到的，因為 server 是要告訴 client 設定 Cookie，並未從 client 端得到 Cookie。

Cookie 的設定（使用 `setcookie` 函式）是放在 HTTP 的 Header（表頭）裡，應該在 HTML 內容還沒有出現時設定。若 PHP 的設定「`output_buffering`」為「On」時，表示 HTTP 內容有做緩衝，或許不需要放在最前面。不過避免「`output_buffering`」為「Off」，或者緩衝區已滿的情況下，`setcookie` 函式應該在 HTML 內容還沒有出現時設定。

Session

一般講「會談」（session）是指客戶還在瀏覽該站，或者還在使用該站的服務。Session 是伺服器為了知道客戶端是否還在瀏覽該站的機制。為了不佔用連線資源，HTTP 是個溝通完就斷線的協定，所以伺服器無法得知客戶端是否還在該站瀏覽。因此，就必須訂定一些規則來實現這個機制。

首先，server 必須先分辨 client，一般有兩種做法。第一種，在網站的所有連結中都置入一個 GET 參數「客戶識別碼」（Session ID），這種方式實做起來較為麻煩，而且一不小心就會置換「客戶識別碼」。

第二種，是利用 Cookie 存放「Session ID」，只要在 client 第一次拜訪時將 Session ID 存入 Cookie 即可。這種是常用也是比較方便的做法。此時 Session 的存活就會依賴 Cookie 是否有效而定。

若 client 的瀏覽器停在某個網頁，使用者可能某些原因（例如：去洗澡）久久未再拜訪該站，或者根本就已離開該站。此時會依 Session 的存活時間，決定 Session 是否有效。當然，server 是以 client 最後一次拜訪開始計時的；若 client 在 Session 存活時間內，持續訪問該站，Session 就會一直有效。一般 Session 的存活時間會設定為 5~20 分鐘，流量越大的網站，Session 存活時間設得越短越節省主機資源。

有了 Session ID 之後，server 會在主機記憶體為每個 Session ID 建立一個對應的 Session 物件，資料就存在 Session 物件裡。有效的 Session 越多，使用的記憶體就會越多。

常用操作 Session 的函式有：`session_start()` 啟用 Session、`session_destroy()` 清除 Session 物件。讀取和設定 Session 使用 `$_SESSION` 預設變數。`session_name` 函式可以讀取或設定 Session ID 名稱，預設為 `PHPSESSID`。

```
<?php
session_start();
if(isset($_POST['name']) and isset($_POST['password'])) {
    if($_POST['name']=='shinder' and $_POST['password']=='12345') {
        $_SESSION['login'] = $_POST['name'];
    }
}
?>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>login test</title>
</head>
<body>
```

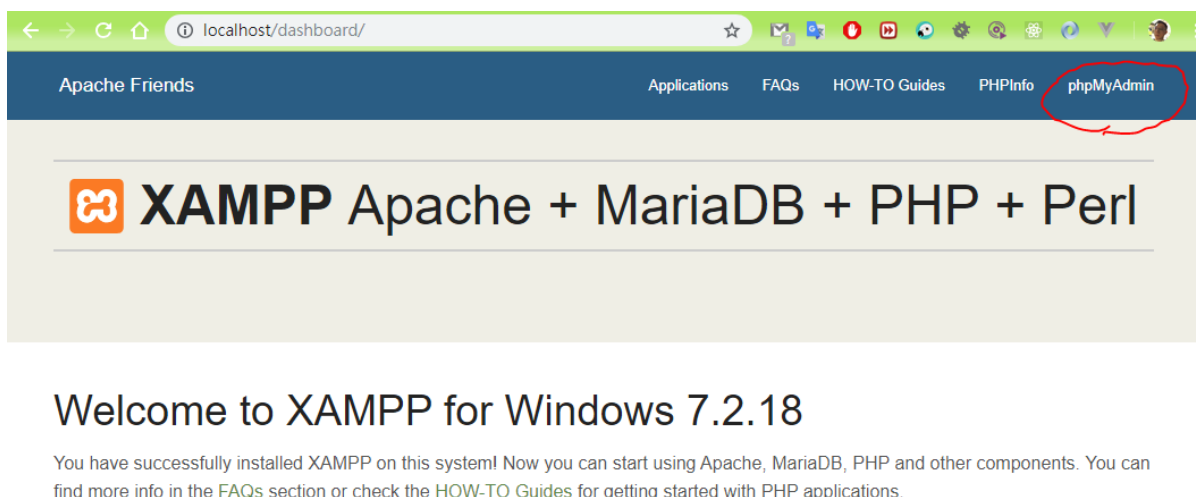
```
<?php
if( isset($_SESSION['login']) ) {
    echo $_SESSION['login'] . ', 您好!';
    echo '<br /><a href="session_logout.php">登出</a>';
} else {
    ?>
    <form id="form1" name="form1" method="post" action="">
        <label style="display:block">
            名稱<input type="text" name="name" />
        </label>
        <label style="display:block">
            密碼<input type="password" name="password" />
        </label>
        <label>
            <input type="submit" name="button" value="Submit" />
        </label>
    </form>
    <?php
}
?>
</body>
</html>
```

登出時，只要用 `session_destroy()` 函式即可。

```
<?php
session_start();
session_destroy();
header("Location: session_login.php");
```

用 phpMyAdmin 建立資料庫

MySQL 的管理工具，最基本的是 mysql client（在 Windows 上是 mysql.exe），不過由於它是文字指令介面，不是圖形介面。使用時要熟悉指令，這對一開始學習 MySQL 的程式設計師而言增加了門檻。



phpMyAdmin 是 PHP 撰寫而成的管理工具、圖形網頁介面、開放軟體、費免使用。可以執行 PHP 的環境就可以使用，而且大部份 PHP/MySQL 虛擬主機商都提供 phpMyAdmin 介面，讓使用者自行管理。

建立資料庫

透過 <http://localhost/phpmyadmin/> 裡的工具，打開 phpMyAdmin 主頁。然後在「建立新資料庫」的欄位填入資料庫的名稱，「校對」欄選「utf_general_ci」，接著按「建立」。



- ① 點選首頁（主目錄）。
- ② 點選「資料庫」分頁。
- ③ 填入資料庫名稱「test_db」。
- ④ 點選「utf8_general_ci」。
- ⑤ 點按「建立」。

按下「建立」之後，頁面會顯示建立成功，有些版本會秀出建立資料庫的 SQL 語法：

```
CREATE DATABASE `test_db` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;
```

建立資料表

◎ 建立資料表之前，先規劃好用途及 ER mapping（Entity-relation mapping）。ER mapping 是所謂實體關係對應。以建立通訊錄為例，通訊錄（簡化後）預計存放四項資料：姓名、email、手機號碼和地址，這個資料表至少要有 4 欄，外加一個流水號當主鍵，應該要有 5 欄。選定資料庫，並建立新資料表，輸入資料表名稱及欄位數目。



- ① 點選剛剛建立的資料庫。
- ② 輸入所要建立的資料表名稱「address_book」。
- ③ 填入欄位數為 5。
- ④ 點按「執行」。

第一個欄位為流水號名稱設定為「sid」，型態為 INT 整數，在「索引鍵 INDEX」欄設定為 PRIMARY (primary key, 主鍵)，並勾選 A_I 的核選盒。



- ① 第一個欄位名稱設定為「sid」。
- ② 型態為 INT 整數。
- ③ 設定為 PRIMARY。
- ④ 勾選 A_I 的核選盒。

主鍵的意思是該欄位為識別資料的主要欄位，所以該欄位的值不會重複。主鍵會編列索引以加快資料的查詢，一張資料表只會有一欄為主鍵。若其它欄也要有索引的功能，可設定為索引鍵。設定為索引鍵可加快查詢，同時也會較耗資源。

勾選 A_I 核選盒的意思是該欄為 Auto Increment (自動加號)。新增資料時，sid 欄不用設定，該值會自動累加。

資料型態可以使用的有許多種，常用的有，INT (整數)、FLOAT (浮點數)、VARCHAR (不定長度的字串)、TEXT (大量的文字)、DATE (日期)、DATETIME (日期及時間)、TIMESTAMP (時間戳記)。

其它欄位名稱分別為：name、email、mobile 和 address，型態皆使用 VARCHAR。VARCHAR 的意思是不定長度的字串，使用 VARCHAR 要記得「一定要設定長度」，沒設長度在建立資料表時會發生錯誤。設定長度是限定字串的最大長度，所以在設定時應該寬裕一點，以免到時候資料放不下而被截斷。

欄位設定好之後，儲存引擎使用「MyISAM」，校對使用「utf8_general_ci」，按「儲存」按鈕建立。

The screenshot shows the MySQL database management interface. At the top, there's a navigation bar with icons for '瀏覽' (Browse), '結構' (Structure), 'SQL', '搜尋' (Search), '新增' (Add), '匯出' (Export), '匯入' (Import), '權限' (Permissions), '操作' (Operations), '追蹤' (Tracking), and '崩潰器' (Crash Reporter). Below this, the '資料表名稱' (Table Name) is set to 'address_book', and the '欄位' (Columns) tab is selected. The '新增' (Add) button is highlighted.

The main area displays a table structure with columns: '名稱' (Name), '類型' (Type), '長度/值' (Length/Value), '預設值' (Default Value), '編碼與排序' (Encoding and Collation), '屬性' (Attributes), '空值 (Null)' (Null), '索引' (Index), and 'A' (Action). The columns are: 'sid' (INT, PRIMARY), 'name' (VARCHAR(255)), 'email' (VARCHAR(255)), 'mobile' (VARCHAR(255)), and 'address' (VARCHAR(255)).

Below the table structure, there are settings for '資料表備註' (Table Comment), '編碼與排序' (Encoding and Collation) set to 'utf8_general_ci', and '儲存引擎' (Storage Engine) set to 'InnoDB'. The 'PARTITION 定義' (Partition Definition) section is empty.

At the bottom right, there are buttons for '預覽 SQL' (Preview SQL) and '儲存' (Save), with the '儲存' button highlighted.

- ① 填入各欄位名稱。
- ② 選擇適當的資料類型。
- ③ 設定資料長度，尤其是 VARCHAR。
- ④ 編碼與排序選擇「utf8_general_ci」。
- ⑤ 選擇儲存引擎，通常使用「InnoDB」。
- ⑥ 點按「儲存」。

MySQL 的設計是儲存引擎是低耦合的，可以選擇使用不同的儲存引擎（每個資料庫在選定儲存引擎後就不能更換）。常用的儲存引擎為「MyISAM」，其特色是快速，一般網站使用這個就可以了。若要有 transaction（交易）的功能，可以使用「InnoDB」。

所謂「校對編碼」，是資料以什麼編碼來看待。校對編碼從頭到尾都統一用 utf8 以避免不必要的亂碼發生。每個資料欄都有「校對」的設定，只要設定資料表的校對即可。

欄位名稱可不可以使用中文？可以，資料庫名稱、資料表名稱和欄位名稱都可以是中文，不過不建議使用中文，以避免造成不必要的困擾。

如果在建立資料表的過程，發現當初設想的欄位數太少，可以在「新增？個欄位」下填入要增加的欄位數，並按「執行」。若欄位數太多，沒用到的只要欄位名稱留白即可。

新增資料

點選上方選單的「新增」按鈕，進入新增資料的頁面，並填入所欲新增的資料，再按「執行」按鈕新增。sid 欄位的值可以不用填，因為我們已經設定它為 Auto Increment（自動增加）。

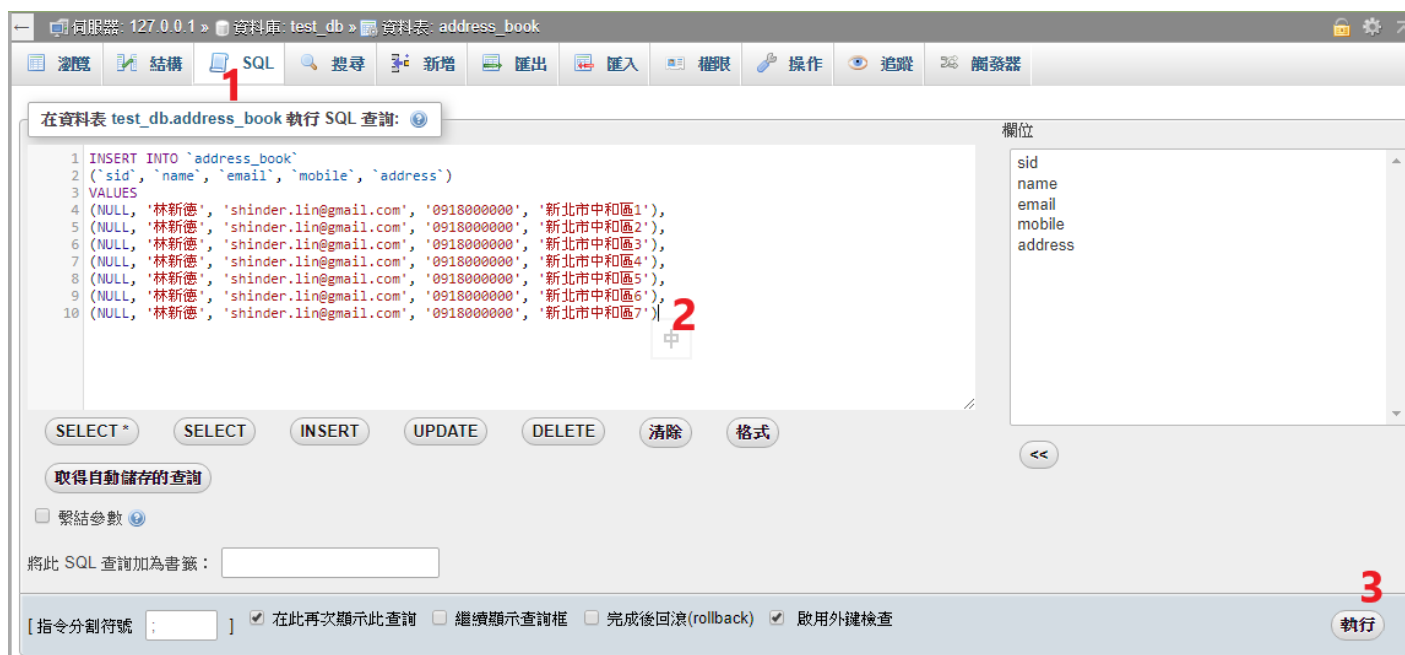
The screenshot shows the phpMyAdmin interface. On the left, the database tree view shows 'test_db' selected, with 'address_book' highlighted under the '新增' (Add) button. The main panel shows the '新增' (Add) form for the 'address_book' table. The form has five fields: 'sid' (int(11)), 'name' (varchar(255)), 'email' (varchar(255)), 'mobile' (varchar(255)), and 'address' (varchar(255)). The 'name' field contains '林新德', 'email' contains 'shinder.lin@gmail.com', 'mobile' contains '0918000000', and 'address' contains '新北市中和區'. The 'sid' field is empty. At the bottom right, there is a '執行' (Execute) button. Red numbers 1, 2, 3, and 4 are overlaid on the image to indicate the steps: 1 points to 'address_book' in the sidebar, 2 points to the '新增' button, 3 points to the 'name' field, and 4 points to the '執行' button.

- ① 點選資料表。
- ② 點選「新增」分頁。
- ③ 填入各欄的值。
- ④ 點按「執行」。

點按「執行」後，將顯示插入狀況的訊息。完成新增資料，同時會顯示其 SQL 語法。

```
INSERT INTO `address_book`  
(`sid`, `name`, `email`, `mobile`, `address`)  
VALUES  
(NULL, '林新德', 'shinder.lin@gmail.com', '0918000000', '新北市中和區');
```

請注意，上述的欄位名稱都使用「`」包裹，字串的值用單引號「'」包裹。phpMyAdmin 的新增頁面每次可以新增 2 筆資料，如果有多筆資料，用其新增頁面就顯得較為不方便。可以使用文字編輯器，將之前新增資料的 SQL 敘述複製貼上，再加以編寫。



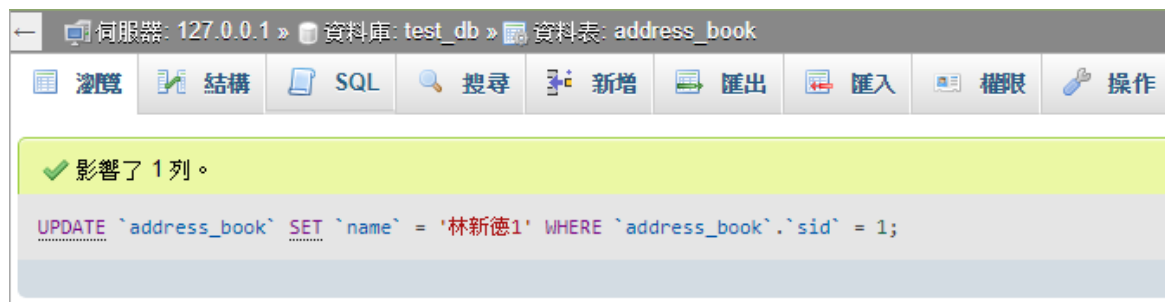
修改資料

新增一些資料後，點按資料表名稱。再按某筆資料前的編輯圖示以進行修改。



- ① 點選「瀏覽」分頁。
- ② 點按「編輯」，以進入編輯畫面。
- ③ 或在欲編輯的欄位內雙擊，可「快速編輯」。

修改 name 欄位的資料後，按「執行」。



```
UPDATE `address_book` SET `name`='林新德 1'
WHERE `address_book`.`sid`=1;
```

使用一張表，WHERE 後面的資料表名稱可以省略，通常寫成：

```
UPDATE `address_book` SET `name`='林新德 1' WHERE `sid`=1;
```

phpMyAdmin 的安全性設定

MySQL 預設的使用者是 root，密碼預設為 root (XAMPP 預設密碼為空字串)。

設定使用者及密碼

在開發時，由於是測試環境，使用 root 及無密碼是沒問題的。如果有隱私問題，或者要直接用發佈時使用的帳號及密碼，就必須設定使用者和密碼。

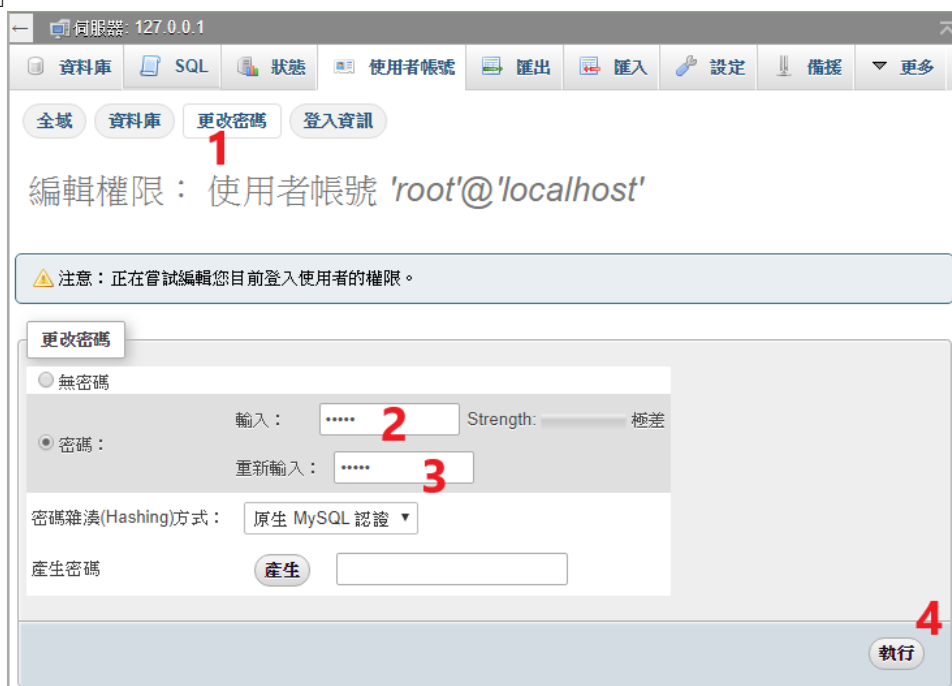
在第一次變更密碼時，最好用紙筆記下密碼，倘若忘記密碼就可能要重新安裝 MAMP。

先到 phpMyAdmin 的主目錄頁面，點選「權限」，此時可以看到「使用者一覽」裡有兩個授權的使用者都是 root，可是「從哪登入」的主機是不同的。只要主機 IP 或網域名稱看起來不一樣，就視為不同的主機（不管它們最後是不是指向相同的主機）。



- ① 點選「首頁」（主目錄）。
- ② 點按「使用者帳號」分頁。
- ③ 點按某帳號的「編輯權限」。

點選「編輯權限」圖示，進入權限編輯及密碼設定頁面，找到「更改密碼」，輸入密碼兩次（在此密碼為 admin，要上線的 Service 不要使用此密碼，應使用強度高的密碼），點按「執行」。



變更密碼後，可以看到成功更改的字樣及對應的 SQL 敘述，重新載入頁面後就會發現「拒絕存取」（Access denied）。

MySQL 使用者「root@localhost」的密碼更改後，phpMyAdmin 還是使用舊密碼登入，所以被拒絕存取。我們必須修改 phpMyAdmin 的設定檔，位於

「/Applications/MAMP/bin/phpMyAdmin」的「config.inc.php」。注意，另一個檔名有點雷同的「config.sample.inc.php」，就如其名是個 sample，參考用，改這個是無效的。

用文字編輯器打開「config.inc.php」之後，在密碼設定行加入密碼：

```
$cfg['Servers'][$i]['user'] = 'root';  
$cfg['Servers'][$i]['password'] = '密碼';
```

config.inc.php 存檔後，phpMyAdmin 重新載入就可以正常操作 MySQL 了。忘記密碼怎麼辦呢？重新安裝應該是最快的方式。

不將密碼放在 config.inc.php

某些情況您可能不希望直接開 phpMyAdmin 就可以操作 MySQL，希望有個登入的管控。此時可以改用不同的 auth_type，原來是使用「config」的方式。在 config.inc.php，我們可以找到這行：

```
$cfg['Servers'][$i]['auth_type'] = 'config';
```

除了 config，我們還可以使用 http 或 cookie。把上行的「config」改成「http」就是使用 http 的方式，使用者名稱和密碼並不放在 config.inc.php 裡，而是登入時輸入。

若使用 cookie，必須多設定一個 blowfish 加密的字串：

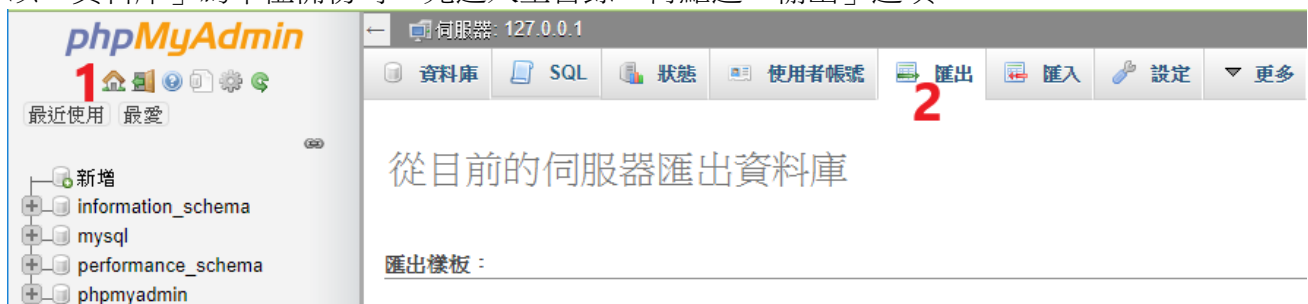
```
$cfg['blowfish_secret'] = 'just_keyin_something';  
$cfg['Servers'][$i]['auth_type'] = 'cookie';
```

MySQL 資料備份

在此介紹用 phpMyAdmin 備份的方式。

以 phpMyAdmin 的輸出做備份時，依範圍大小分成以「資料庫」為單位，或者以「資料表」為單位。

以「資料庫」為單位備份時，先進入主目錄，再點選「輸出」選項。



- ① 點選「首頁」（主目錄）。
- ② 點按「匯出」分頁。

以「資料庫」為單位備份時，快速選項可匯出所有資料庫。自訂選項可選擇某些資料庫。



- ① 點選「自訂」。
- ② 選取要匯出的資料庫。
- ③ 點按「執行」按鈕匯出（其餘項目皆不須變更）。

以「資料表」為單位備份時，先點選左側資料庫，再點選「輸出」選項。接著在「檢視資料庫的備份概要」的左側就可以選擇要輸出的資料表。



- ① 點選欲匯出的資料庫。
- ② 點選「匯出」分頁。
- ③ 點選「自訂」。
- ④ 選取要匯出的資料表。
- ⑤ 點按「執行」按鈕匯出。

單一「資料表」備份，作法類似。下圖是顯示為文字的方式。



- ① 點選欲匯出的資料表。
- ② 點選「匯出」分頁。
- ③ 點選「自訂」。
- ④ 點選「以文字顯示輸出結果」。
- ⑤ 點按「執行」按鈕匯出到頁面上。

以下方框內容為匯出的 SQL 內容，很明顯的備份的方式是輸出 SQL 敘述（其中一為註解），使用的語法不外乎「CREATE」和「INSERT」。注意，在預設的情況，建立資料表時會先判斷是否已存在同名稱的資料表，若不存在才建立。

```
-- phpMyAdmin SQL Dump
-- version 3.4.5
-- http://www.phpmyadmin.net
--
-- 主機: localhost
-- 產生日期: 2018 年 12 月 04 日 14:38
-- 伺服器版本: 5.5.16
-- PHP 版本: 5.6.8

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

--
-- 資料庫: `test_db`
--
--
```

```
--
-- 表的結構 `address_book`
--

CREATE TABLE IF NOT EXISTS `address_book` (
  `sid` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `email` varchar(255) NOT NULL,
  `mobile` varchar(255) NOT NULL,
  `address` varchar(255) NOT NULL,
  PRIMARY KEY (`sid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=9 ;

--
-- 轉存資料表中的資料 `address_book`
--

INSERT INTO `address_book` (`sid`, `name`, `email`, `mobile`, `address`) VALUES
(1, '林新德', 'shinder.lin@gmail.com', '0918000000', '新北市汐止區'),
(2, '林心得1', 'shinder.lin@gmail.com', '0918000001', '新北市汐止區1'),
(3, '林心得2', 'shinder.lin@gmail.com', '0918000002', '新北市汐止區2'),
(4, '林心得3', 'shinder.lin@gmail.com', '0918000003', '新北市汐止區3'),
(5, '林心得4', 'shinder.lin@gmail.com', '0918000004', '新北市汐止區4'),
(6, '林心得5', 'shinder.lin@gmail.com', '0918000005', '新北市汐止區5'),
(7, '林心得6', 'shinder.lin@gmail.com', '0918000006', '新北市汐止區6'),
(8, '林心得7', 'shinder.lin@gmail.com', '0918000007', '新北市汐止區7');
```

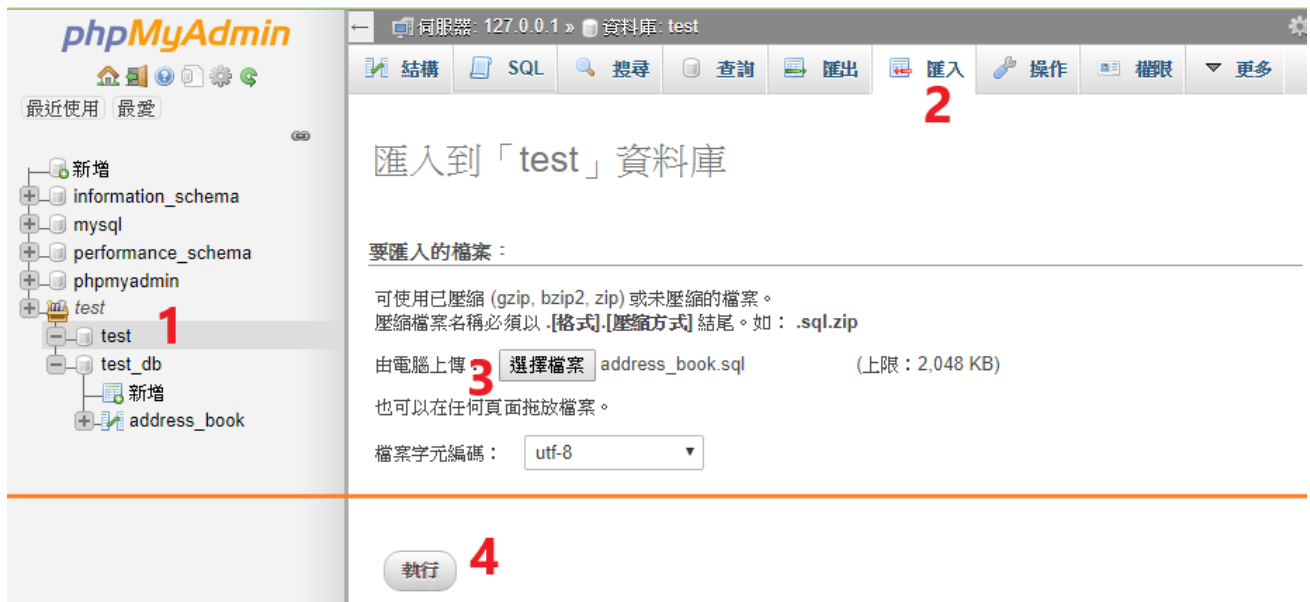
除了以 SQL 格式做輸出，也可以使用其它格式。不過其它格式大多是為了將資料轉移到不同的軟體（如 Excel）。以資料庫備份的角度應該使用 SQL 格式。

使用 phpMyAdmin 做資料還原

在 phpMyAdmin 做匯入時，同樣依範圍大小分成以「資料庫」為單位，或者以「資料表」為單位。

以「資料庫」為單位匯入時，先進入主目錄，再點選「匯入」選項。

以「資料表」為單位匯入時，先點選左側資料庫，再點選「匯入」選項。



- ① 選取標的資料庫。
- ② 點選「匯入」分頁。
- ③ 點按「選擇檔案」，選擇所要匯入的 SQL 檔。
- ④ 點按「執行」按鈕匯入。

載入的 SQL 檔應該小於 2M（或 8M）。如果大於 2M 時，可以用文字編輯器將檔案分割成幾個小檔案；或者調整伺服器 PHP 可上傳的檔案大小。若資料備份時檔案遠大於 2M 時，可以使用 BigDump（<http://www.ozeroov.de/bigdump/>）。

使用 PDO API

參考專案

<https://github.com/shinder/mmmh57-php>

<https://bitbucket.org/lstd0125/mfee19-php/src/master/>

資料庫連線

```
<?php
$db_host = 'localhost';
$db_name = 'proj54';
$db_user = 'root';
$db_pass = 'root';
$db_charset = 'utf8';
$db_collate = 'utf8_unicode_ci';

$dsn = "mysql:host={$db_host};dbname={$db_name};charset={$db_charset}";

// pdo 連線設定
```

```

$pdo_options = [
    PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_PERSISTENT => false,
    PDO::ATTR_EMULATE_PREPARES => false,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES {$db_charset} COLLATE
{$db_collate}"
];

try {
    $pdo = new PDO($dsn, $db_user, $db_pass, $pdo_options);
} catch(PDOException $ex) {
    echo 'Connection failed:'. $ex->getMessage();
}

if(! isset($_SESSION)){
    session_start();
}

```

使用 PDO::query

```

<?php
require __DIR__. '/__connect_db.php';
$page_name='product_list';
$per_page = 4;
$params = [];

$cate = isset($_GET['cate']) ? intval($_GET['cate']) : 0;
if(isset($_GET['cate'])){
    $params['cate'] = $cate;
}

// 用戶要看第幾頁
$page = isset($_GET['page']) ? intval($_GET['page']) : 1;

// 取得分類資料
$c_sql = "SELECT * FROM categories WHERE parent_sid=0";
$c_stmt = $pdo->query($c_sql);
$cates = $c_stmt->fetchAll(PDO::FETCH_ASSOC);

$where = ' WHERE 1 ';
if(! empty($cate)){
    $where .= " AND category_sid=$cate ";
}

// 取得總筆數
$t_sql = " SELECT COUNT(1) FROM products $where";
$total_rows = $pdo->query($t_sql)->fetch(PDO::FETCH_NUM)[0];

```

```

$total_pages = ceil($total_rows/$per_page); //總頁數

// 取得商品資料
$p_sql = sprintf("SELECT * FROM products $where LIMIT %s, %s ",
    ($page-1)*$per_page, $per_page);
$p_stmt = $pdo->query($p_sql);

```

使用 PDO::prepare

```

<?php
require __DIR__ . '/__connect_db.php';

$result = [
    'success' => false,
    'code' => 400,
    'info' => '參數不足',
    'postData' => [],
];

if(isset($_POST['email']) and isset($_POST['password'])){
    $result['postData'] = $_POST;

    // 去掉頭尾空白, 然後轉小寫
    $email = strtolower(trim($_POST['email']));
    // 密碼編碼, 不要明碼
    $password = sha1(trim($_POST['password']));

    $sql = "SELECT `id`, `email`, `mobile`, `address`, `birthday`,
`nickname`
            FROM `members` WHERE `email`=? AND `password`=?";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([
        $email,
        $password,
    ]);

    // 影響的列數 (筆數)
    if($stmt->rowCount() == 1){
        $result['success'] = true;
        $result['code'] = 200;
        $result['info'] = '登入完成';

        $_SESSION['user'] = $stmt->fetch(PDO::FETCH_ASSOC);
    } else {
        $result['code'] = 410;
    }
}

```

```
        $result['info'] = 'Email 或密碼錯誤';
    }
}
echo json_encode($result, JSON_UNESCAPED_UNICODE);
```

通訊錄製作

計算總筆數

```
$t_sql = "SELECT COUNT(1) FROM address_book";
$total = $pdo->query($t_sql)->fetch(PDO::FETCH_NUM)[0];
```

依頁碼讀取資料

```
$per_page = 5; // 每頁幾筆
$page = isset($_GET['page']) ? intval($_GET['page']) : 1; // 用戶要看第幾頁

$total_pages = ceil($total/$per_page); // 總頁數
$page = $page > $total_pages ? $total_pages : $page;
$page = $page < 1 ? 1 : $page;

$sql = sprintf("SELECT
    `sid`, `name`, `email`, `mobile`, `address`, `birthday`
    FROM address_book
    ORDER BY `sid` DESC
    LIMIT %s, %s", ($page-1)*$per_page, $per_page);
$stmt = $pdo->query($sql);
$ar = $stmt->fetchAll(PDO::FETCH_ASSOC); // 一次讀取全部
```

掛載 Bootstrap css framework

<http://getbootstrap.com/>

```
<link rel="stylesheet" href="bootstrap/css/bootstrap.css">
<link rel="stylesheet" href="bootstrap/css/bootstrap-theme.css">
<script src="lib/jquery-3.4.1.js"></script>
<script src="bootstrap/js/bootstrap.js"></script>
```

分離選單部份

```
<?php include __DIR__ . '/__navbar.php' ?>
```

以表格顯示資料清單

```
<table class="table table-bordered">
```

分頁按鈕 pagination

```
<ul class="pagination">
  <?php for($i=1; $i<=$total_pages; $i++): ?>
    <li class="page-item <?= $i==$page ? 'active' : '' ?>">
      <a class="page-link" href="?page=<?=$i?>"><?=$i ?></a>
    </li>
  <?php endfor ?>
</ul>
```

分頁按鈕參考 (JavaScript)

```
// 處理頁碼按鈕
(function(page, totalPages, prevNum){
  let beginPage, endPage;
  if(totalPages <= prevNum*2+1){
    beginPage = 1;
    endPage = totalPages;
  } else if(page-1 < prevNum) {
    beginPage = 1;
    endPage = prevNum*2+1;
  } else if(totalPages-page < prevNum) {
    beginPage = totalPages-(prevNum*2+1);
    endPage = totalPages;
  } else {
    beginPage = page-prevNum;
    endPage = page+prevNum;
  }
  output.beginPage = beginPage;
  output.endPage = endPage;
})(page, output.totalPages, 3);
```

在選單上標示目前頁面

```
<ul class="nav navbar-nav">
  <li class="<?= $page_name=='data_list' ? 'active' : '' ?>">
    <a href="data_list.php">資料列表</a>
  </li>
  <li class="<?= $page_name=='data_insert' ? 'active' : '' ?>">
    <a href="data_insert.php">新增資料</a>
  </li>
</ul>
```

新增資料的表單

```

<form method="post">
  <div class="form-group">
    <label for="name">姓名</label>
    <input type="text" class="form-control" id="name" name="name">
  </div>
  <div class="form-group">
    <label for="email">email</label>
    <input type="email" class="form-control" id="email" name="email">
  </div>
  <div class="form-group">
    <label for="mobile">手機號碼</label>
    <input type="text" class="form-control" id="mobile" name="mobile">
  </div>
  <div class="form-group">
    <label for="birthday">生日</label>
    <input type="text" class="form-control" id="birthday"
name="birthday">
  </div>
  <div class="form-group">
    <label for="address">地址</label>
    <input type="text" class="form-control" id="address" name="address">
  </div>

  <button type="submit" class="btn btn-default">Submit</button>
</form>

```

新增資料

```

<?php
require __DIR__ . '/__connect_db.php';
$result = [
  'success' => false,
  'code' => 400,
  'info' => '參數不足',
  'postData' => [],
];

if(isset($_POST['name']) and isset($_POST['email'])){
  $result['postData'] = $_POST;

  $s_sql = "SELECT 1 FROM `address_book` WHERE `email`=?";
  $s_stmt = $pdo->prepare($s_sql);
  $s_stmt->execute([ $_POST['email'] ]);
  if($s_stmt->rowCount()==1){
    $result['code'] = 420;
    $result['info'] = 'Email 重複';
    echo json_encode($result, JSON_UNESCAPED_UNICODE);
    exit;
  }
}

```



```

    $sql = "INSERT INTO `address_book` (
        `name`, `email`, `mobile`, `address`, `birthday`
    ) VALUES (?, ?, ?, ?, ?)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([
        $_POST['name'],
        $_POST['email'],
        $_POST['mobile'],
        $_POST['address'],
        $_POST['birthday']
    ]);

    // 影響的列數 (筆數)
    if($stmt->rowCount()==1){
        $result['success'] = true;
        $result['code'] = 200;
        $result['info'] = '資料新增完成';
    } else {
        $result['code'] = 410;
        $result['info'] = '資料沒有新增';
    }
}
echo json_encode($result, JSON_UNESCAPED_UNICODE);

```

資料排序

```

$sql = sprintf("SELECT * FROM `address_book` ORDER BY `sid` DESC LIMIT %s, %s",
    ($page - 1) * $per_page,
    $per_page
);

```

使用 prepare() 和佔位符號？可以自動跳脫單引號排除 SQL injection

表單欄位檢查

```

<form name="form1" method="post" onsubmit="return checkForm();">
<!--表單內容 -->
</form>

```

```

<script>
function checkForm(){
    if(document.form1.name.value.length<2){
        alert('請填寫正確姓名');
        return false;
    }
}

```

```
}  
</script>
```

檢查 email 格式

```
function checkForm(){  
    if(document.form1.name.value.length<2){  
        alert('請填寫正確姓名');  
        return false;  
    }  
    if(! validateEmail(document.form1.email.value)){  
        alert('email 格式不正確');  
        return false;  
    }  
}  
// http://stackoverflow.com/questions/46155/validate-email-address-in-javascript  
function validateEmail(email) {  
    var re =  
/^((([^\<>()\\[\]\\. ,;: \s@"]+(\.[^\<>()\\[\]\\. ,;: \s@"]+)*)|("[.+")@((\[[0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\]|([a-zA-Z\ -0-9]+\.)+[a-zA-Z]{2,})))$/;  
    return re.test(email);  
}
```

後端驗證 email 格式

<https://www.php.net/manual/en/filter.examples.validation.php>

```
<?php  
$email_a = 'joe@example.com';  
$email_b = 'bogus';  
  
if (filter_var($email_a, FILTER_VALIDATE_EMAIL)) {  
    echo "Email address '$email_a' is considered valid.\n";  
}  
if (filter_var($email_b, FILTER_VALIDATE_EMAIL)) {  
    echo "Email address '$email_b' is considered valid.\n";  
} else {  
    echo "Email address '$email_b' is considered invalid.\n";  
}  
?>
```

清單頁移除用圖示

```
<td>  
    <a href="javascript: delete_it(<?= $r['sid'] ?>)">  
        <i class="fas fa-trash-alt"></i>  
    </a>  
</td>
```

刪除資料前的詢問

```
<script>
    function delete_it(sid){
        if(confirm('刪除編號為 ' + sid + ' 的資料?')){
            location.href = 'data_delete.php?sid=' + sid;
        }
    }
</script>
```

移除資料

```
<?php
require __DIR__ . '/__connect_db.php';

$cid = isset($_GET['cid']) ? intval($_GET['cid']) : 0;

if(empty($cid)){
    header('Location: ./data_list.php');
    exit;
}

$sql = "DELETE FROM `address_book` WHERE `cid`=$cid";

pdo->query($sql);

if(isset($_SERVER['HTTP_REFERER'])){
    // 從哪裡來回哪裡去
    header('Location: ' . $_SERVER['HTTP_REFERER']);
} else {
    header('Location: ./data_list.php');
}
```

編輯資料的 icon

```
<td>
    <a href="data_edit.php?cid=?= $r['cid'] ?>">
        <i class="fas fa-edit"></i>
    </a>
</td>
```

編輯資料：先讀取該筆資料

```
<?php
require __DIR__ . '/__connect_db.php';
```

```
// 沒有參數的話，直接回列表頁
if(empty($_GET['sid'])){
    header('Location: data_list.php');
    exit;
}
$sid = intval($_GET['sid']);

$sql = "SELECT * FROM `address_book` WHERE `sid`=$sid";
$stmt = $pdo->query($sql);

// 如果沒有拿到資料，就返回列表頁
if($stmt->rowCount()<1){
    header('Location: data_list.php');
    exit;
}

// 拿出一筆資料 (只有一筆)
$row = $stmt->fetch(PDO::FETCH_ASSOC);
```

修改資料

```
<?php
require __DIR__ . '/__connect_db.php';

$result = [
    'success' => false,
    'code' => 400,
    'info' => '參數不足',
    'postData' => [],
];

if(isset($_POST['sid']) and isset($_POST['name']) and
isset($_POST['email'])){
    $result['postData'] = $_POST;

    // 不是要修改的這一筆，若找到相同的 email
    $s_sql = "SELECT 1 FROM `address_book` WHERE `email`=? AND
`sid`<> ?";
    $s_stmt = $pdo->prepare($s_sql);
    $s_stmt->execute([
        $_POST['email'],
        $_POST['sid'],
    ]);
    if($s_stmt->rowCount()==1){
        $result['code'] = 420;
        $result['info'] = 'Email 重複';
    }
    echo json_encode($result, JSON_UNESCAPED_UNICODE);
    exit;
```

```

    }

    // \[value\-\d\]
    $sql = "UPDATE `address_book` SET
`name`=?, `email`=?, `mobile`=?, `address`=?, `birthday`=? WHERE `sid`=?" ;

    $stmt = $pdo->prepare($sql);

    $stmt->execute([
        $_POST['name'],
        $_POST['email'],
        $_POST['mobile'],
        $_POST['address'],
        $_POST['birthday'],
        $_POST['sid']
    ]);

    // 影響的列數 (筆數)
    if($stmt->rowCount() == 1){
        $result['success'] = true;
        $result['code'] = 200;
        $result['info'] = '資料修改完成';
    } else {
        $result['code'] = 410;
        $result['info'] = '資料沒有修改';
    }
}

echo json_encode($result, JSON_UNESCAPED_UNICODE);

```

XSS Attack

避免 XSS 攻擊的兩種方式

```

<!-- PHP: strip_tags() -->
<td><? = htmlentities($row['address']) ?></td>

```

PHP 產出 JSON 的方式

```

<?php
$ar = array(
    'name' => '彼德潘 / peter',
    'age' => 30,
    'other' => array(56, 'aaa', 72)
);
echo json_encode($ar, JSON_UNESCAPED_UNICODE|JSON_UNESCAPED_SLASHES);
echo json_encode($ar, JSON_UNESCAPED_UNICODE+JSON_UNESCAPED_SLASHES);

```

json_decode()

```
$str = '{"name":"peter"}';  
$br = json_decode($str);  
var_dump($br);  
echo $br->name;  
$cr = json_decode($str, true);  
echo $cr['name'];
```

自訂回傳的 JSON 格式

```
{  
  "success": true,  
  "items_num": 25,  
  "pages_num": 5,  
  "per_page": 5,  
  "page": 2,  
  "data": [  
    {},  
    {},  
    {},  
    {}  
  ]  
}  
/*  
{  
  "success": false,  
  "code": 1000,  
  "error": "參數錯誤"  
}  
*/
```

兩層分類選單參考

```
<?php  
require __DIR__ . '/parts/connect_db.php';  
$stmt = $pdo->query("SELECT * FROM categories ORDER BY sid DESC");  
$raw_data = $stmt->fetchAll();  
$first = [];  
// 把第一層的資料放到陣列裡  
foreach($raw_data as $r){  
    if($r['parent_sid']==0){  
        $first[] = $r;  
    }  
}  
  
// 把第二層的資料放到陣列裡  
foreach($first as &$f){  
    foreach($raw_data as $r){  
        if($f['sid']==$r['parent_sid']){  
            $f['children'][] = $r;  
        }  
    }  
}
```

```
    }  
  }  
}  
  
echo json_encode($first);
```

多層選單結構組成

```
<?php  
if(! isset($pdo)){  
    require __DIR__ . '/config/__connect_db_jd2.php';  
}  
  
function getTagTree($pdo) {  
    $sql = "SELECT * FROM `tags` ORDER BY `sequence`, `parent_sid`,  
`sid`";  
    $rows = $pdo->query($sql)->fetchAll();  
  
    $dict = [];  
    foreach($rows as $k=>$v){  
        $dict[$v['sid']] = &$rows[$k];  
    }  
  
    $tagTree = [];  
    foreach($dict as $sid=>$item){  
        if($item['parent_sid']!=0){  
            $dict[$item['parent_sid']]['children'][] = &$dict[$sid];  
        } else {  
            $tagTree[] = &$dict[$sid];  
        }  
    }  
    return $tagTree;  
}  
$tag_cates = getTagTree($pdo);
```

用戶密碼編碼使用

password_hash() (<https://www.php.net/manual/en/function.password-hash>)
password_verify() (<https://www.php.net/manual/en/function.password-verify>)

git clone 專案

```
cd /Applications/MAMP/htdocs/  
git clone https://github.com/shinder/mmmh57-php.git  
cd mmmh57-php  
# git pull origin
```

SQL 參考

-- SQL的註解

-- 合併查詢說明

```
SELECT *  
FROM `products` JOIN `categories`;
```

 -- 合併查詢：通常不這樣子用，說明用

-- 合併查詢

```
SELECT `products`.*, `categories`.`name`  
FROM `products`  
JOIN `categories`  
ON `products`.`category_sid` = `categories`.`sid`;
```

-- 資料表別名

```
SELECT p.*, c.`name`  
FROM `products` AS p  
JOIN `categories` AS c  
ON p.`category_sid` = c.`sid`;
```

-- 資料集欄位別名

```
SELECT p.*, c.`name` AS 分類名稱  
FROM `products` AS p  
JOIN `categories` AS c  
ON p.`category_sid` = c.`sid`;
```

```
SELECT p.*, c.`name` 分類名稱  
FROM `products` p  
JOIN `categories` c  
ON p.`category_sid` = c.`sid`;
```

-- LEFT JOIN

```
SELECT p.*, c.`name` cate_name  
FROM `products` p  
LEFT JOIN `categories` c  
ON p.`category_sid` = c.`sid`;
```

```
SELECT p.*, c.`name` cate_name  
FROM `categories` c  
LEFT JOIN `products` p  
ON p.`category_sid` = c.`sid`;
```

-- 取出某欄為空值

```
SELECT p.*, c.`name` cate_name  
FROM `products` p  
LEFT JOIN `categories` c  
ON p.`category_sid` = c.`sid`  
WHERE c.`sid` IS NULL;
```

-- 取出某欄不為空值

```
SELECT p.*, c.`name` cate_name  
FROM `products` p  
LEFT JOIN `categories` c  
ON p.`category_sid` = c.`sid`
```



```
WHERE c.`sid` IS NOT NULL;

--
SELECT * FROM `products` WHERE `author`='吳睿紘';
SELECT * FROM `products` WHERE `author` LIKE '吳睿紘';

SELECT * FROM `products` WHERE `author` LIKE '陳%';

SELECT * FROM `products` WHERE `author` LIKE '%陳%';

SELECT * FROM `products` WHERE `author` LIKE '%計%' OR `bookname` LIKE '%計%';

SELECT * FROM `products` WHERE `author` LIKE '%科技%' OR `bookname` LIKE '%科技%';

--
SELECT * FROM `products` WHERE sid=6 OR sid=2 OR sid=3;

SELECT * FROM `products` WHERE sid IN (6, 2, 3);

SELECT * FROM `products` WHERE sid IN (6, 2, 3) ORDER BY sid DESC;

SELECT * FROM `products` WHERE sid IN (6, 2, 3, 10, 15) ORDER BY RAND();

-- 計算總筆數
SELECT COUNT(*) FROM `products`;

SELECT COUNT(sid) FROM `products`;

SELECT COUNT(1) FROM `products`;

-- SUM() 總合
SELECT * FROM `products` WHERE sid IN (1,2);

SELECT SUM(`price`) FROM `products` WHERE sid IN (1,2);

-- GROUP BY: 群組
SELECT `category_sid`, COUNT(1) num FROM `products` GROUP BY `category_sid`;
-- MySQL 8 無法執行
SELECT *, COUNT(1) num FROM `products` GROUP BY `category_sid`;

-- MySQL 8 的錯誤訊息: #1055 - Expression #1 of SELECT list is not in GROUP
BY clause and contains nonaggregated column 'mfee19.products.sid' which
is not functionally dependent on columns in GROUP BY clause; this is
incompatible with sql_mode=only_full_group_by
-- 解決方式: https://stackoverflow.com/questions/41887460/select-list-is-not-in-group-by-clause-and-contains-nonaggregated-column-inc
-- SET GLOBAL sql_mode=(SELECT
REPLACE(@@sql_mode, 'ONLY_FULL_GROUP_BY', ''));

SELECT p.`category_sid`, c.`name`, COUNT(1) num
```

```
FROM `products` p
    JOIN `categories` c
        ON p.`category_sid`=c.`sid`
GROUP BY p.`category_sid`;

SELECT p.`category_sid`, c.`name`, COUNT(1) num
FROM `products` p
    LEFT JOIN `categories` c
        ON p.`category_sid`=c.`sid`
GROUP BY p.`category_sid`;

-- 訂單編號 10 的產品名稱
SELECT o.member_sid, o.order_date, d.*, p.bookname
FROM `orders` o
    JOIN `order_details` d
        ON o.sid=d.order_sid
    JOIN `products` p
        ON p.sid=d.product_sid
WHERE o.sid=10

-- 日期的格式 'YYY-MM-DD'
-- $today = date("Y-m-d H:i:s");
-- time()
-- strtotime()
SELECT * FROM `orders` WHERE `order_date` > '2017-01-01';
-- '2017' > '2016-06-01 11:15:53'
-- '2017' > '2018-06-01 11:15:53'

SELECT * FROM `orders`
    WHERE `order_date` >= '2017-04-07'
    AND `order_date` < '2017-04-08'

-- 累加
UPDATE `products` SET `price`=`price`+1 WHERE sid=1

-- Datetime 不要用 + 運算
SELECT NOW(), NOW()+100000;

-- 子查詢
SELECT product_sid sid FROM `order_details` WHERE `order_sid`=10;

SELECT * FROM `products` WHERE sid IN (
    SELECT product_sid FROM `order_details` WHERE `order_sid`=10
);

SELECT * FROM `products`
    JOIN
        (SELECT product_sid, price od_price
            FROM `order_details` WHERE `order_sid`=10) od
        ON `products`.sid = od.product_sid;

-- 款式，型號，顏色

-- 購物車：多種型態的商品、多個購物車
```

-- 查看某個會員的訂購記錄

```
SELECT *  
  FROM `orders` o  
 JOIN `order_details` d ON o.`sid`=d.`order_sid`  
 WHERE o.member_sid=1;
```

```
SELECT *  
  FROM `orders` o  
 JOIN `order_details` d ON o.`sid`=d.`order_sid`  
 WHERE o.member_sid=1 AND `order_date` > '2017-04-01';
```

```
SELECT *  
  FROM `orders` o  
 JOIN `order_details` d ON o.`sid`=d.`order_sid`  
 JOIN `products` p ON p.sid=d.product_sid  
 WHERE o.member_sid=1 AND `order_date` > '2017-04-01';
```