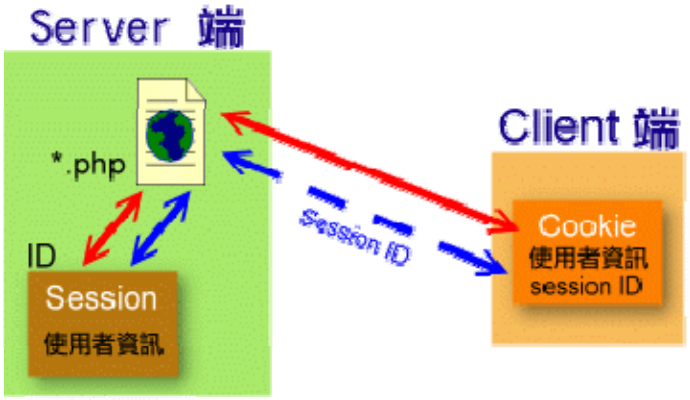
SESSION 與 COOKIE

引用自：王勝雄，台中市網【PHP 程式語言編寫】研習課程
網址：http://km.tceb.edu.tw/~wsx/php/index.htm

Session 與 Cookie 運作原理

在互動式網頁中，Session 及 Cookie 提供了「記憶」有關使用者資訊的能力，例如當我們到某一些網站的時候，網頁會呈現您「第 n 次光臨」或是在網路書店選書，但還沒確定下單時，過程中選的書如何被「記憶」下來呢？在一般的互動性網頁大都運用了所謂的 Session 或 Cookie 的功能來處理這些工作。

即然是「記憶」使用者的資訊，除了用「記憶體」之外(但一般不致於用它，畢竟容量比較小)，主要的就是利用「檔案」來儲存資訊，所以 Session 或 Cookie 其實都是一個「小」檔案，但這個檔案儲存的位置不同，「安全性」也不同；先以下圖來說明：



檔案位置

Session 是將使用者資訊儲存在 Server 端暫存檔中，儲存的位置是依照 php.ini 的設定：

```
session.save_path = /tmp
```

而 Cookie 是儲存在使用者自己的電腦裡(Client 端)，儲存的位置一般是放在

- C:\Documents and Settings\電腦帳號\Cookies (Windows NT/2000/XP)
- 或 C:\Documents and Settings\帳號\Local Settings\Temporary Internet Files\
- 或 C:\windows\Cookies (Windows 98)

檔案名稱＆內容

Session 的檔案名稱「可能」是 ：

sess_3dd484f2bab6a2d2509e9850dae3b897
(就是 sess_ 開頭，加上 32 字元的亂數雜湊編碼所組成的檔名)

它的內容可能是：

checkls:3:"yes";var1ls:1:"4";var2ls:1:"5";

其實 PHP 的 Session 內容就是「變數名稱｜變數類型：長度：內容；」的組合

Cookie 的檔案名稱「可能」 是：

tceb@km.tceb.edu[1].txt (就是 主機使用者帳號@網址.txt)

它的內容可能是：

phpbb2km_data (cookie 名稱）
s%3A0%3A%22%22%3B (cookie 值）
km.tceb.edu.tw/ (cookie 網域及路徑)
1536
16956672
29648268
4116192768
29574842
*

在一般 Cookie 的檔案中，每一筆記錄有 8 個訊息，前 3 個訊息依序為：

1. 瀏覽的網站在 Client 端所要紀錄的 cookie 名稱(以 PHP 程式設計觀之，其為變數名稱，上面的範例就是指「phpbb2km_data」)
2. cookie 值(變數值，例如「s%3A0%3A%22%22%3B」)
3. 網站的網域與路徑位置(例如：「km.tceb.edu.tw/」)，其他的訊息大致還有「安全性、cookie 有效日期、修改日期、建立日期、建立者」等等訊息

Cookie 的內容事先經過編碼，一般來說，除非透過軟體來解讀(例如：IECookiesView) ，一般是無法了解其意涵的。

傳遞 Session

由於 Cookie 是存在 Client 端，所以在不同程式之間的傳遞都可以藉由使用者本身的檔案來存取所需的資訊，但 Session 是儲存在 Server 端，除非使用者告訴 Server Session 資訊檔案名稱是什麼，不然是無法傳遞「Session 值」(就是 sess_後面加的 32 字元)， 因此一般來說，會利用兩種方式來傳遞「Session 值」:

1. 透過 URLs(因網路安全考量，現在大多不用此種方法)
2. 透過 Cookie

Cookie 的限制

因為 Cookie 常常用來存取使用者的資訊，為了怕被拿來濫用，或是佔用太多硬碟空間，所以對 Cookie 做出了以下的限制：

- 每個使用者的瀏覽器只能支援(存取)300 個 Cookie
- 每個瀏覽器只能針對同一個伺服器存取 20 個 Cookie
- 每個 Cookie 的大小最多僅 4k Bytes 的容量
- 有些瀏覽器可以把 Cookie 的功能關掉，若關掉後 Cookie 就不能使用

Session 函式

當 PHP 在使用 Session 的功能之前，由於 php.ini 的一些設定值會影響 session 函數的使用，例如：設定項目「session.save_path」預設為 /tmp，其目的在指定 session 檔案存放路徑;而設定項目「session.auto_start」預設為 0(關閉)，其目的在指定 session 是否自動啟動。上述的設定可以在官方版的 PHP 使用手冊中查詢關鍵字「php.ini」得到 PHP directives 的完整詳細說明。

如果要改變這些設定，可直接修改 php.ini 再重新啟動 Web Server，也可以利用 ini_set( )函數來設定，設定方式如下：

```
ini_set ("參數名稱", "新的參數值");
```

使用 Session 時，不外乎三個程序：

1. 啟動 session 功能：使用 session_start( ) 函數
2. 註冊 session 變數：使用 $_SESSION 陣列變數
3. 清除 session 變數或檔案：使用 unset( )或 session_destroy( )函數
   (使用 $_SESSION=array( ) 可以清除所有的 session 變數)

接下來，先介紹 session 各函數的語法，再用範例的方式來綜合介紹如何使用 session:

session_start( )函數

說明：建立一個 session 或在已啟動 session 的狀況下，繼續目前 session id 的「傳值」狀態，這類似 GET、POST 或 cookie 一樣。

注意：如果使用的狀態是屬於 cookie base 的 session 傳遞方式，之後程式的開頭都必須使用 session_start( )函數。 另外，在使用 session_start( )函數時，在 session_start( )之前都不可以有任何的結果輸出，例如不可以在 session_start( )之前使用 echo 輸出變數或將 session_start( )嵌入在網頁中，其他 html 語法先輸出，否則會出現錯誤訊息。

範例：

程式 chap7-2-1.php

```
01:  <?php
02:      session_start();   //啟動 session
03:      echo '>>第一頁<<';
04:
05:      $_SESSION['color'] = '藍色';
06:      $_SESSION['city'] = '台中市';
07:      $_SESSION['time'] = date("Y 年 m 月 d 日  H 時 i 分 s 秒");
08:
09:      // 這是連結到另一個 php 程式，
10:      // 如果可以使用 cookie 方式，
11:      // 則利用以下方式連結到「第二頁」的程式
12:
13:      //注意這裡用單引號
14:      echo '<br /><a href="ch7-2-1a.php">第二頁(cookie)</a>';
15:  ?>
```

範例說明：

第 02 行：啟動 session，此時 PHP 會將 session name 設為 PHPSESSID(預設值)，並且將 session id 的值存在 cookie 裡，同時建立一個 session 檔案。

第 05、06、07 行：利用$_SESSION 將變數「color」的值記錄到 session 檔案裡。

session_destroy 函數

說明：結束 session，並將儲存 session 的檔案刪除。

注意：須先啟動 session_start( )函數才可正常運作。

範例：

```
01:    <?php
02:        session_start( );              // 起始 session
03:        session_unset( );              // 清除 session 內所有的變數
04:        session_destroy( );            // 結束 session
05:    ?>
```

設定 session 變數的值

說明：在 session 中註冊一個變數，並且指定它的值

用法：$_SESSION[‘變數名稱’] = 指定的值;

範例：

```
$_SESSION['name'] = $std_name;
```

範例說明：在 session 中註冊一個名為 name 的變數，且令儲存 $std_name 的值

刪除 session 變數的值

說明：刪除一個在 session 中註冊的變數

用法：unset($_SESSION[‘變數名稱’]);

範例：

```
unset($_SESSION['name']);
```

範例說明：在 session 中刪除一個名為 name 的變數

Cookie 函數

在 PHP 中，要使用 Cookie 的指令只有一個：

setcookie (string name [, string value [, int expire [, string path [, string domain [, int secure]]]]])

換個寫法：

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

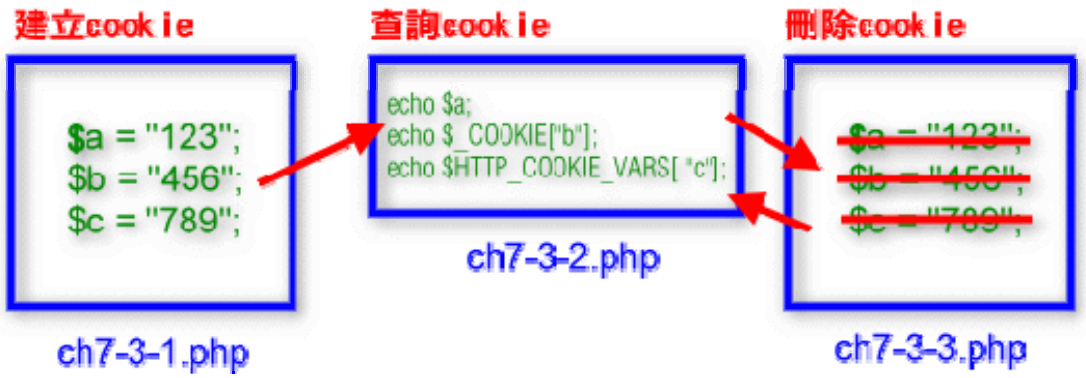
除了 Cookie 的「變數名稱」參數一定要有之外，其餘可省略；參數的說明如下表：

參 數 名 稱	參 數 說 明
Cookie 名稱(Name)	Cookie 的名稱，以 PHP 來說，就是變數的名稱，例如：cookiename 就好比 $_COOKIE['cookiename']
Cookie 值(Value)	就是 Cookie 名稱的值
留存時間(Expire)	Cookie 的留存時間，以「秒」為單位，例如要留存 30 天，那就要用「time( )+30*24*60*60」 time( )函數表示取得現在的時間+30 天*24 小時*60 分*60 秒) ；如果沒有設定，當瀏覽器關掉時，cookie 也跟著結束。
路徑(path)	指在哪些目錄下的檔案可以使用 Cookie
網域名稱(Domain)	用來設定哪些網域可以使用 Cookie
安全性(Secure)	如果結合 HTTPS 安全傳輸協定(SSL)，就是「1」，一般則為「0」。

使用 setcookie( )函數時，得先注意下列事項：

1. 與 session_start( )函數的狀況一樣，在使用 setcookie( )函數時，之前都不可以有任何的結果輸出，例如不可以在 setcookie( )之前使用 echo 輸出變數或將 setcookie ( ) 嵌入在網頁中，其他 html 語法先輸出，否則會出現錯誤訊息。
2. cookie 變數的值可以用 $_COOKIE["名稱"] 的方式取得。
3. 如果沒有設定留存時間，雖然有指定 cookie 名稱及數值，但因為一設定就過期，結果是無法讀取您寫入的 Cookie。
4. 如果要刪除 cookie 的話，直接把要刪除 cookie 名稱的值設定為空白即可。

範例解說：



程式 ch7-3-1.php

```
01:    <?php
02:        setcookie ("a", "123", time()+1800);
03:        setcookie ("b", "456", time()+1800);
04:        setcookie ("c", "789", time()+1800);
05:
06:        echo '<a href="ch7-3-2.php">查詢 Cookies</a>';
07:    ?>
```

程式 ch7-3-1.php 的第 01、02、03 行各註冊一個 cookie 變數及值，存活時間均設為 1800 秒(約 30 分鐘)，而 time( )函數是取得現在時間的 unix 時間格式。

程式 ch7-3-2.php

```
01:    <?php
02:        echo "<br> a ->".$_COOKIE["a"];
03:        echo "<br> b ->".$_COOKIE["b"];
04:        echo "<br> c ->".$_COOKIE["c"];
05:
06:        echo '<br /> <a href="ch7-3-1.php">重新建立 Cookies</a>';
07:        echo '<br /> <a href="ch7-3-3.php">刪除 Cookies</a>';
08:    ?>
```

程式 ch7-3-2.php 顯示三個 cookie 變數的值。

程式 ch7-3-3.php

```
01:    <?php
02:        setcookie ("a", "");
03:        setcookie ("b", "", time()-1800);
04:        setcookie ("c", "", time()-1800);
05:
06:        echo '<a href="ch7-3-2.php">查詢 Cookies</a>';
07:    ?>
```

程式 ch7-3-2.php 的第 01 行讓 $a 被清除；第 02、03 行倒扣存活時間，使 cookie 過期失效；如果 cookie 的內容都被清除，cookie 的暫存檔會一併刪除。

此外，cookie 的應用一定要使用者的瀏覽器能接受 cookie 的存取，有些人會因為安全的理由將 cookie 關掉，所以檢查使用者有沒接受 cookie 的存取是使用前必要的工作，而這項檢查功能要如何做到呢？其實很簡單，就在 cookie 正式使用前來個「模擬測試」，就是先送出一個測試的 cookie ，再檢查這個 cookie 值是否存在，如果不存在，則表示 cookie 沒開，如果存在，則表示有開，並順手將測試的 cookie 刪除 ；實際範例如下：

程式 ch7-3-4.php

```
01:    <?php
02:        setcookie ("testcookie", "test123", time() + 1800);
03:
04:        if ( $_COOKIE['testcookie'] == "test123" )
05:        {
06:            setcookie ("testcookie" );
07:            echo "使用者瀏覽器接受  cookie !!";
08:        }
09:        else
10:            echo "注意！使用者瀏覽器不接受  cookie !!";
11:    ?>
```

程式 ch7-3-4.php 的第 02 行送出一個 cookie；第 04～09 行檢查 cookie 的值是否存在；如果存在，則清除 cookie，否則告知 cookie 未開。

安全性議題

Session 安全的問題

原因簡單，就是當使用者的 cookie 沒打開的時候，必須利用 URLs 方式傳遞，就是類似：

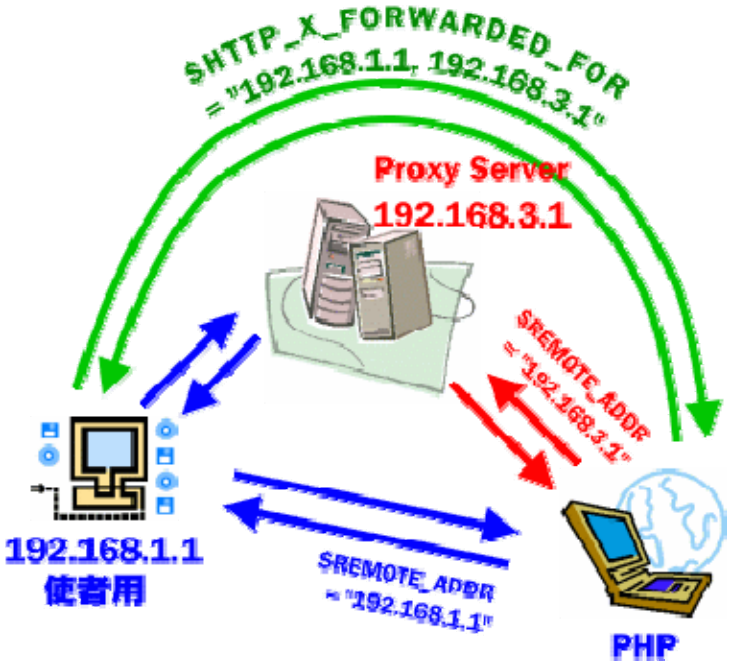
ch7-2-1a.php?PHPSESSID=8234c3f661bd1eac450c23d5650e0881

這時，如果使用者沒有在正常登出後清除 session 時，其他人可能會盜取這一組 URLs 來取得使用者的權限或資訊，雖然這字串很長，這不容易記，但不怕一萬，只怕萬一，所以不管 session 是使用 cookie 或 URLs 方式傳遞 session id，進一步的驗證都是必要的，目前最常用的驗證方法是加上使用者 IP 的資訊。

這個方法簡單的說，就是在使用者登入時，在 session 資訊中即加入使用者的 IP，若有不同的程式啟動相同的 session 時，即連帶檢查是否來自同一個 IP，如果發現不同時，就拒絕使用，並立即刪除這個 session 的檔案。

在 PHP 中，有個 $REMOTE_ADDR 的全域變數會紀錄使用者的 IP，但如果使用者有透過防火牆或代理伺服器(PROXY Server)，則$REMOTE_ADDR 僅會紀錄最後一個連線到 PHP 程式的主機，例如，假設使用者的電腦 IP 位址是 192.168.1.1 ，且使用者沒有設定 proxy 的話，$REMOTE_ADDR 的值為「192.168.1.1」；如果使用者的瀏覽器有設定 proxy，而這 proxy server 的 IP 是 192.168.3.1，則 $REMOTE_ADDR 會是「192.168.3.1」而非「192.168.1.1」，此時，就得用 $HTTP_X_FORWARDED_FOR

變數不可，但這一變數是在透過代理伺服器時才會產生，而且它會將所有經過的代理主機的 IP 記錄下來，所以取用時，也要經過簡單處理，如下圖說明：



所以，驗證程式得先判斷 $HTTP_X_FORWARDED_FOR 是否存在，如果存在，就取第一個 IP 位址，否則就直接用 $REMOTE_ADDR 來驗證；舉例如下：

程式 ch7-4-1.php：儲存使用者的 IP

```
01:    <?php
02:       session_start();   //啟動 session
03:
04:       echo ">>> 第 一 頁 <<< <br />";
05:       if ( !empty($_SERVER["HTTP_X_FORWARDED_FOR"]) )
06:       {
07:          echo "代理主機：".$_SERVER["HTTP_X_FORWARDED_FOR"];
08:          $temp_ip = split(",", $_SERVER["HTTP_X_FORWARDED_FOR"]);
09:          $user_ip = $temp_ip[0];
10:       }
11:       else
12:          $user_ip = $_SERVER["REMOTE_ADDR"];
13:
14:       echo "<br /? 使用者 IP : $user_ip ";
15:       $_SESSION['user_ip'] = $user_ip;
16:
17:       //注意這裡用單引號
18:       echo '<br /><a href="ch7-4-2.php?">第二頁</a>';
19:    ?>
```

程式 ch7-4-1.php 的第 02 行啟動 session ；第 05～08 行利用$_SERVER 研判將 HTTP_X_FORWARDED_FOR 變數是否有值，若有，則利用 split 函數切割字串，第一個分割字串即使用者 IP；第 09～11 行：如果 HTTP_X_FORWARDED_FOR 變數是空的，表示沒有經過代理主機，直接利用$_SERVER["REMOTE_ADDR"]取得使用者 IP；第 14 行：將使用者 IP 置入 session 中。

程式 ch7-4-2.php

```
01:    <?php
02:        session_start();    //啟動 session
03:
04:        echo ">>> 第 二 頁 <<< <br />";
05:        if ( !empty($_SERVER["HTTP_X_FORWARDED_FOR"]) )
06:        {
07:            $temp_ip = split(",", $_SERVER["HTTP_X_FORWARDED_FOR"]);
08:            $user2_ip = $temp_ip[0];
09:        }
10:        else
11:            $user2_ip = $_SERVER["REMOTE_ADDR"];
12:
13:        echo "<BR />原來 session 的 IP:".$_SESSION["user_ip"];
14:        echo "<br />目前使用者 IP : $user2_ip ";
15:
16:        if ( $_SESSION["user_ip"] != $user2_ip )
17:        {
18:            echo "您不是原來登入的 IP，請正常登入!!<br>";
19:            session_destroy ();
20:        }
21:        else
22:            echo "OK! <br />";
23:
24:        //注意這裡用單引號
25:        echo '<br /><a href="ch7-4-1.php">第一頁</a>';
26:    ?>
```

程式 ch7-4-2.php 的第 02 行啟動 session；第 05～10 行，取得目前使用者 IP；第 15～19 行研判 IP 來源是否相同，如果不是，顯示警告，並終結 session。