

此投影片為博碩圖書有限公司提供，僅允許學生私下學習用，請勿複製轉載，以免侵犯智財權



第六章 網頁間的資料傳遞

6-1 PHP網頁的狀態管理

6-2 表單和URL參數的傳遞與接收

6-3 Cookie

6-4 Session

備註：可依進度點選小節

6-1 PHP網頁的狀態管理



狀態管理

狀態管理(1/4)



狀態管理（**State Management**）是一種網頁資料管理的方式，資料來源通常由客戶端所啟動或輸入

在網頁中需要保留下來的資料通常儲存在客戶端或伺服器端

我們將狀態管理依儲存位置區分為兩大類介紹：

- 客戶端的狀態管理
- 伺服器端的狀態管理

狀態管理(2/4)



客戶端的狀態管理

- 將資料儲存在客戶端
- 讓使用者執行的 **PHP** 程式需要資料時，不必透過伺服器端，在客戶端即可存取資料。下表是常見的客戶端的狀態管理方法：

方法	說明
表單欄位	使用HTML表單的標籤來傳遞資料到其他的網頁
URL參數	使用URL網址的參數來傳遞資料到其他的網頁
Cookie	將資料儲存在一份檔案並保留在使用者的電腦內

狀態管理(3/4)



伺服器端的狀態管理

- 將資料儲存在伺服器端
- PHP 程式需要透過伺服器端才能取得資料但是其安全性也比較高

方法	說明
Session變數	在伺服器端使用文字檔案來儲存使用者的資料
文字檔案	在伺服器端使用文字檔案來儲存使用者的資料
資料庫	使用資料庫來儲存使用者的資料

狀態管理(4/4)



各種狀態管理方法的適用場合

方法	適用場合
表單欄位	使適合蒐集網站使用者輸入的資訊，例如網站使用者填寫註冊會員表單的資料。
URL參數	適合傳遞沒有安全性考量的資料，例如傳遞討論區的頁數。
Cookie	適合儲存安全性要求不高以及少量的資料，例如購物車內的商品資訊。因為Cookie是儲存於客戶端，所以可能會有洩露安全問題；但是Cookie可以減伺服器端的負擔。
Session	適合儲存有安全性考量的資料，例如帳號、密碼。因為Session是儲存於伺服器端，所以其安全性較佳；但是伺服器的負擔會較重。
文字檔案	適合儲存大量的文字資料。
資料庫	適合儲存大量的並且需要保密的資料，例如會員的個人資料。資料庫是屬於伺服器端的儲存，當資料有安全性考量以及需要永久性保存時，資料庫是最適合的儲存空間。

6-2 表單和URL參數的傳遞與接收



表單的傳遞與接收

URL參數的傳遞與接收

表單的傳遞與接收(1/7)


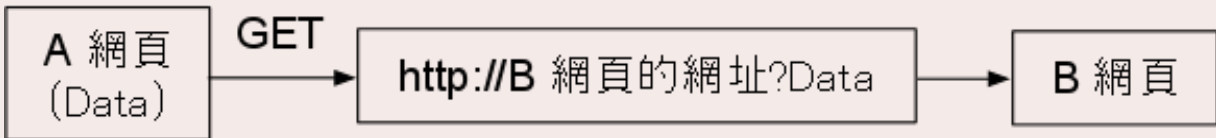


當我們在設計表單時，必須指定表單要傳送的目標頁面與傳送的方式

在<form> 標籤中，**action** 屬性即目標頁面，**method** 屬性即傳送方式。傳送方式分成以下兩種：

表單的傳遞與接收(2/7)



method 屬性	說明
POST	<p>使用HTTP通訊協定的標頭來傳遞表單資料，它允許傳輸大量的資料。</p>  <pre>graph LR; A["A 網頁 (Data)"] -- "HTTP 標頭 (Data)" --> B["B 網頁"]</pre>
GET	<p>使用URL網址的參數來傳遞表資料，其傳輸資料有限，連同URL共255字元。</p>  <pre>graph LR; A["A 網頁 (Data)"] -- "GET" --> B["http://B 網頁的網址?Data"]; B --> C["B 網頁"]</pre>

表單的傳遞與接收(3/7)



接收表單的方式

- 依據傳送方式的不同，在 **PHP** 程式中目標網頁接收表單資料的方式可以使用以下兩種變數來傳遞資料：

傳送方式	接收方式
POST	使用\$_POST["欄位名稱"]指定接收表單內的欄位值
GET	使用\$_GET["欄位名稱"]指定接收表單內的欄位值

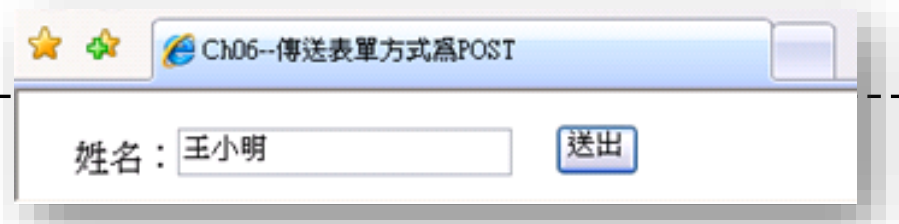
注意：在程式中，\$_POST 和 \$_GET 的英文字母大小寫是有差別的，英文字母必須為大寫，才可正確取得資料。

表單的傳遞與接收(4/7)



使用POST方式傳送表單範例(1/2)

```
<!--傳送表單的網頁-->
<html>
  <head>
    <title>Ch06--傳送表單方式為POST </title>
  </head>
  <body>
    <form name="form1" method="post" action="postform.php">
      姓名：<input name="username" type="text">
        <input type="submit" value="送出">
    </form>
  </body>
</html>
```



表單的傳遞與接收(5/7)



使用POST方式傳送表單範例(2/2)

<!--接收表單的網頁-->

```
<?php
```

```
    echo "Hello, ";
```

```
    echo $_POST["username"];
```

```
?>
```



表單的傳遞與接收(6/7)



使用GET方式傳送表單範例(1/2)

<!--傳送表單的網頁-->

<html>

<head>

<title>Ch06--傳送表單方式為GET</title>

</head>

<body>

<form name="form1" method="get" action="getform.php">

姓名：<input name="username" type="text">

<input type="submit" value="送出">

</form>

</body>

</html>

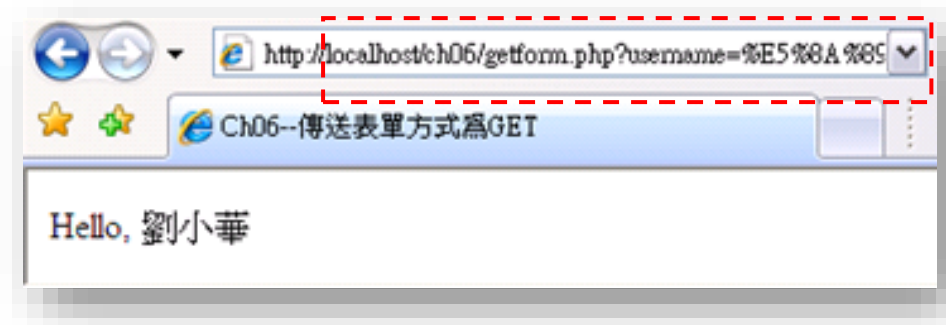


表單的傳遞與接收(7/7)



使用GET方式傳送表單範例(2/2)

```
<!--接收表單的網頁-->  
<?php  
    echo "Hello, ";  
    echo $_GET["username"];  
?>
```



URL參數的傳遞與接收(1/4)

URL參數傳遞資料的方式與上節介紹的 GET method 相同，它同樣是使用 URL 網址的參數來傳遞資料

URL 參數很適合用來傳遞程式間的資料，例如：討論區的頁數

URL參數的傳遞與接收(2/4)

傳遞URL參數

- 資料以字串的方式傳遞，資料必須附加在網址的後面。
- 網址與資料之間必須以「？」符號做為區隔；參數與參數之間必須以「&」符號連結。

使用GET方式接收URL參數

```
<a href="url.php?username=李小龍&sex=男">傳送</a>
```

```
<?php  
    $name = $_GET["username"];  
    $sex = $_GET["sex"];  
?>
```

URL參數的傳遞與接收(3/4)

範例(1/2)

<!--傳送URL參數的網頁-->

<html>

<head>

<title>Ch06--使用URL參數傳遞資料</title>

</head>

<body>

傳送URL參數

</body>

</html>



URL參數的傳遞與接收(4/4)

範例(2/2)

<!--接收URL參數的網頁-->

```
<?php
```

```
    echo $_GET["username"];
```

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

```
    echo $_GET["sex"];
```

```
?>
```



6-3 Cookie



何謂Cookie

存取Cookie資料

何謂Cookie(1/2)



Cookie 是一個小檔案，它用來記錄使用者的登入資訊或使用者的操作資訊，並且儲存於用戶端的電腦中

常見的購物車應用程式，即是使用 **Cookie** 來記錄使用者放入購物車的商品

然而，**Cookie** 並不適合每一個應用，下列是 **Cookie** 的優缺點：

何謂Cookie(2/2)

優點	缺點
Cookie 是儲存於用戶端的磁碟或記憶體，它不會佔用伺服器的資源。	因為Cookie是儲存在用戶端，所以Cookie可能會被竊取，造成安全上的威脅。
Cookie可以長時間儲存於用戶端的電腦，因此在Cookie的生命週期（其生命週期可自行設定）到達前，我們可以不斷取得其Cookie資料，免於重複輸入資料的麻煩。	瀏覽器只能對同一個伺服器儲存20個Cookie，而每個Cookie只有4K Bytes的容量，因此Cookie並不適合儲存大量資料。並且，若瀏覽器不支援Cookie，可能會造成Cookie無法使用。

存取Cookie資料(1/7)



在程式的設定下，我們可以將指定的 **Cookie** 資料儲存於用戶端的電腦之中，並設定該 **Cookie** 的有效時間、儲存路徑與安全性等等。

存取Cookie資料(2/7)



儲存Cookie資料

- 儲存 Cookie 資料要使用 `setcookie()` 函數將資料寫入 Cookie 中

```
<?php
    setcookie("名稱", "內容值");
?>
```

- 若要設定 Cookie 的有效時間、儲存路徑與安全性時，則需要在 `setcookie()` 函數內指定其參數值。

```
<?php
    Setcookie("名稱", "內容值", 有效時間, "儲存路徑", "網域", "安全性");
?>
```

- 若沒有指定 Cookie 的有效時間，那在關閉瀏覽器後，所儲存的Cookie 會消失。反之，Cookie 在有效期間過後才會自動刪除

存取Cookie資料(3/7)



Setcookie()函數的參數說明如下：

參數	說明
名稱	Cookie的名稱。
內容值	Cookie儲存的內容。
有效時間	Cookie的有效時間，格式為時間戳記。例：time()+3600 表示有效時間為 1 個小時 (3600秒=60(秒/分)*60(分/小時))。
儲存路徑	Cookie 的儲存路徑。從「php.ini 網頁伺服器設定檔」內可以看到“session.cookie_path”顯示的預設的路徑。若沒有設定儲存路徑，則依照預設的路徑為主。預設路徑通常為「/」，表示對於伺服器上所有檔案與目錄都是有效的儲存路徑。(通常php.ini檔案路徑都在『C:\WINDOWS』底下，若想快速搜尋到該檔案的話可以從「開始→執行」輸入php.ini即可開啟該檔案。)
網域	Cookie的有效網域。若指定「www.test.com.tw」為有效網域，那麼Cookie只存取於www.test.com.tw；若不指定網域或主機，那麼預設的有效網域是產生Cookie的主機名稱。
安全性	0：允許使用HTTP傳送Cookie 1：只有使用HTTPS才能傳送Cookie 安全性的預設值為0。

存取Cookie資料(4/7)



Setcookie()函數的注意事項

- 在呼叫 `setcookie()` 函數之前不可以對瀏覽器有任何的輸出，否則 `setcookie()` 會發生錯誤
- 但是，如果必須在呼叫 `setcookie()` 函數之前呼叫 `echo()` 時，那麼我們就用 `ob_start()` 函數來解決錯誤的發生

函數	說明
<code>ob_start()</code>	<code>ob_start()</code> 函數的用意是打開輸出緩衝區。當打開了緩衝區， <code>echo()</code> 函式內的字串不會輸出到瀏覽器，而是保留在內部緩衝區，直到使用 <code>flush()</code> 函式或者 <code>ob_end_flush()</code> 函式，其字串才會輸出至瀏覽器。

存取Cookie資料(5/7)



讀取Cookie資料

```
<?php  
    echo $_COOKIE["名稱"];  
?>
```

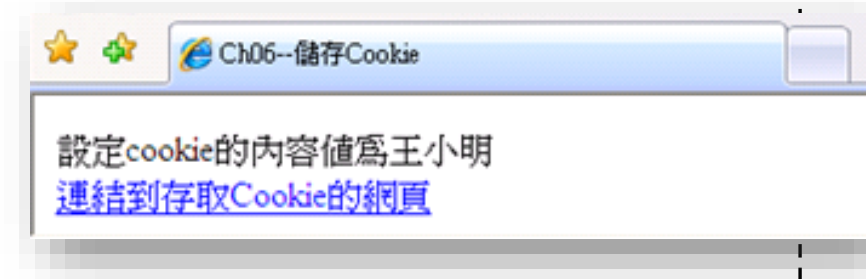
- 注意：在程式中，\$_COOKIE的英文字母大小寫是有差別的，英文字母必須為大寫，才可正確取得資料

存取Cookie資料(6/7)



範例(1/2)

```
<?php
    ob_start();
    echo ("設定cookie的內容值為王小明");
    setcookie("username", "王小明", time()+3600); //儲存Cookie，並設定
    ob_end_flush();                               //Cookie的有效期限為 1 個小時。
?>
<html>
    <head>
        <title>Ch06--儲存Cookie</title>
    </head>
    <body>
        <br>
        <a href="cookie.php">連結到存取Cookie的網頁</a>
    </body>
</html>
```

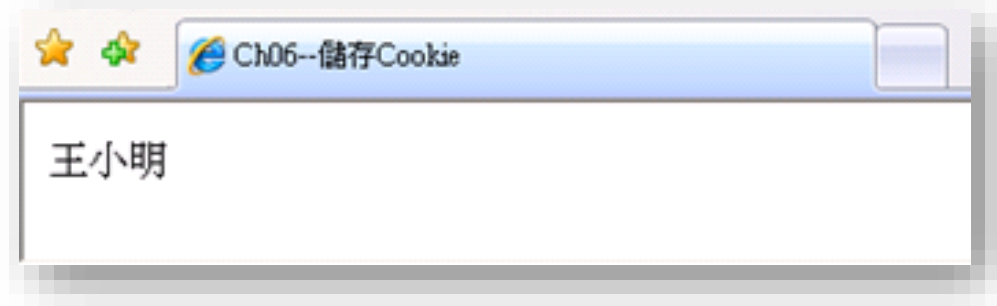


存取Cookie資料(7/7)



範例(2/2)

```
<?php  
    echo $_COOKIE["username"];  
?>
```



6-4 Session



何謂Session

啟動與存取Session

Session的有效時間

刪除Session

何謂Session(1/3)



Session 的意思是「使用者進入網站到使用者離開網站的整個過程」

在 PHP 程式中，使用 Session 變數可以在伺服器端保存使用者的狀態資料

- 以會員系統為例，當使用者登入系統後，伺服器端會產生一個空間來保存使用者登入的會員資料，這個就是 Session

何謂Session(2/3)



當使用者用瀏覽器連線到網站時，網站伺服器會指定一個 **SessionID** 給這次的連線

- 在預設的狀態下，**SessionID** 會用 **Cookie** 的方式儲存在用戶端
- 同一個使用者瀏覽各個網頁時都可以取得同一個 **SessionID**，換句話說，**SessionID** 就像各個網頁的父節點，每個網頁都可以繼承 **SessionID**

當瀏覽器關閉時候，**Cookie** 隨之刪除，因此同一台電腦的後一位使用者也就無法得到上一個使用者的 **SessionID**，因此 **Session** 適用於有安全性考量的資料

何謂Session(3/3)



此外，一個瀏覽器開啟網站就會有一個 Session，二個瀏覽器開啟相同網站就會有二個 Session，以此類推

因此，多個瀏覽器開啟同一個網站，其 Session 都是不相同的

啟動與存取Session(1/4)



啟動Session與設定Session變數

- 在 PHP 程式中要使用 Session，必須先呼叫 `session_start()` 函數來啟動
- 啟動 Session 之後，我們就可以使用 `$_SESSION` 變數的方式去設定 Session 的資料
- 程式碼範例如下：

```
<?php
    session_start();
    $_SESSION[“名稱”] = “要設定的值”;
?>
```

注意：在程式中，`$_SESSION`的英文字母大小寫是有差別的，英文字母必須為大寫，才可正確存取資料。

啟動與存取Session(2/4)



取得Session變數值

- 設定了 Session 變數後，若要在別的網頁存取 Session 變數值時，必須先呼叫 `session_start()` 函數
- 之後再呼叫 `$_SESSION["名稱"]` 即可取得變數值
- 程式碼如下：

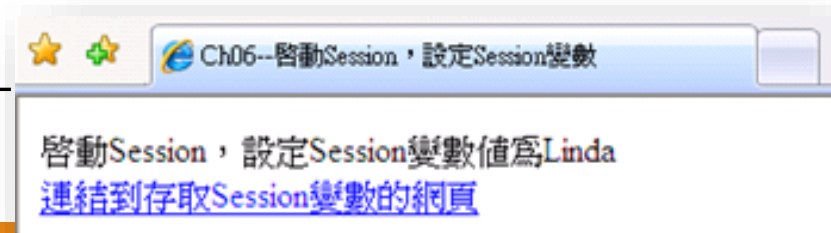
```
<?php
    session_start();
    echo $_SESSION["名稱"];
?>
```

啟動與存取Session(3/4)



範例(1/2)

```
<!--啟動Session，設定Session變數-->
<?php
    session_start(); //啟動Session
    $_SESSION["username"]="Linda"; //設定Session變數
?>
<html>
    <head><title>Ch06--啟動Session，設定Session變數</title></head>
    <body>
        啟動Session，設定Session變數值為Linda<br>
        <a href="getsession.php">連結到存取Session變數的網頁</a>
    </body>
</html>
```

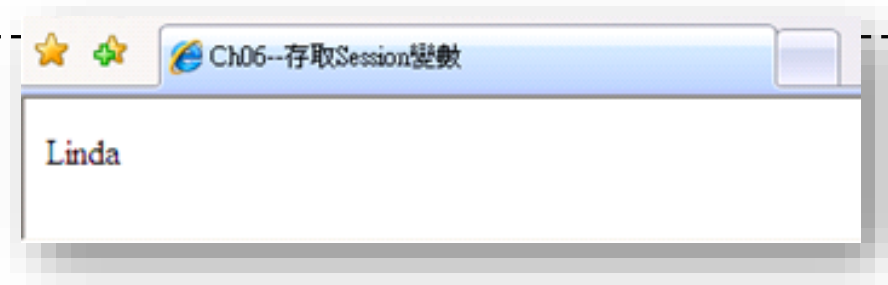


啟動與存取Session(4/4)



範例(2/2)

```
<!--存取Session變數-->
<?php
    session_start(); //啟動Session
    if(!isset($_SESSION["username"])){ //判斷Session是否存在
        echo "Session變數不存在"; }
    else{
        echo $_SESSION["username"]; }
?>
```



Session的有效時間



Session 是有時限性的，在 <php.ini> 設定檔中，預設 Session 的有效期限值為 1440（24*60 秒）

- 假設當使用者開啟網站，啟動了 Session，而使用者若在24 分鐘內都沒有在網站內執行任何動作的話，Session 即會自動消失

若要修改有效期限，則可以針對下面的參數值作修改

```
session.gc_maxlifetime = 1440
```

刪除Session(1/3)



使用者關閉網頁後，Session 就等於失效了，但是，除了關閉網頁外，還是有兩種方法可以刪除 Session

- 刪除指定的Session

```
<?php
    unset($_SESSION["名稱"]);
?>
```

- 刪除目前所有的Session

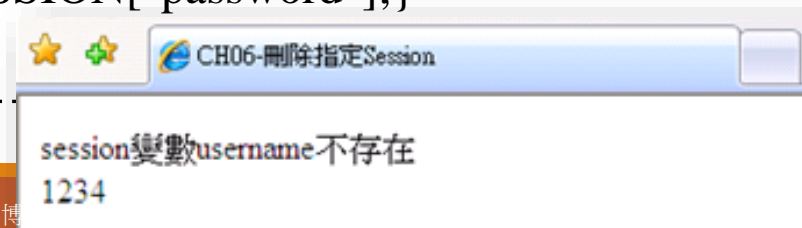
```
<?php
    session_unset();
?>
```

刪除Session(2/3)



刪除指定Session範例

```
<!--刪除指定Session-->
<?php
    session_start(); //啟動Session
    $_SESSION["username"]="Linda"; //設定Session變數
    $_SESSION["password"]="1234"; //設定Session變數
    unset($_SESSION["username"]); //刪除指定的Session變數
    if(!isset($_SESSION["username"])){ //判斷Session變數是否存在
        echo "session變數username不存在<br/>";
        echo $_SESSION["password"];}
    else{
        echo $_SESSION["username"]."<br/>";
        echo $_SESSION["password"];}
?>
```



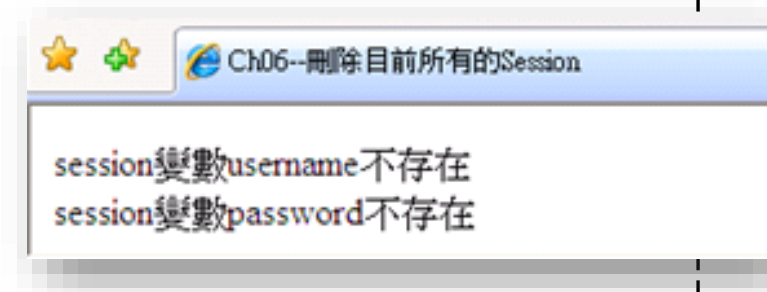
刪除Session(3/3)



刪除目前所有的Session範例

```
<!--刪除目前所有的Session-->
<?php
    session_start(); //啟動Session
    $_SESSION["username"]="Linda"; //設定Session變數
    $_SESSION["password"]="1234"; //設定Session變數
    session_unset(); //刪除所有的Session

    if(!isset($_SESSION["username"])&!isset($_SESSION["password"]))
    {
        echo "session變數username不存在<br/>";
        echo "session變數password不存在" ;
    }else{
        echo $_SESSION["username"]."<br/>";
        echo $_SESSION["password"];
    }
?>
```



本章結束

Q&A討論時間
