

# PHP

Stig Sæther Bakken

Alexander Aulbach

Egon Schmid

Jim Winstead

Lars Torben Wilson

Rasmus Lerdorf

Andrei Zmievski

Jouni Ahto

14-08-2002

<http://www.php.net/>

<<PHP.ppt>>

# Outline

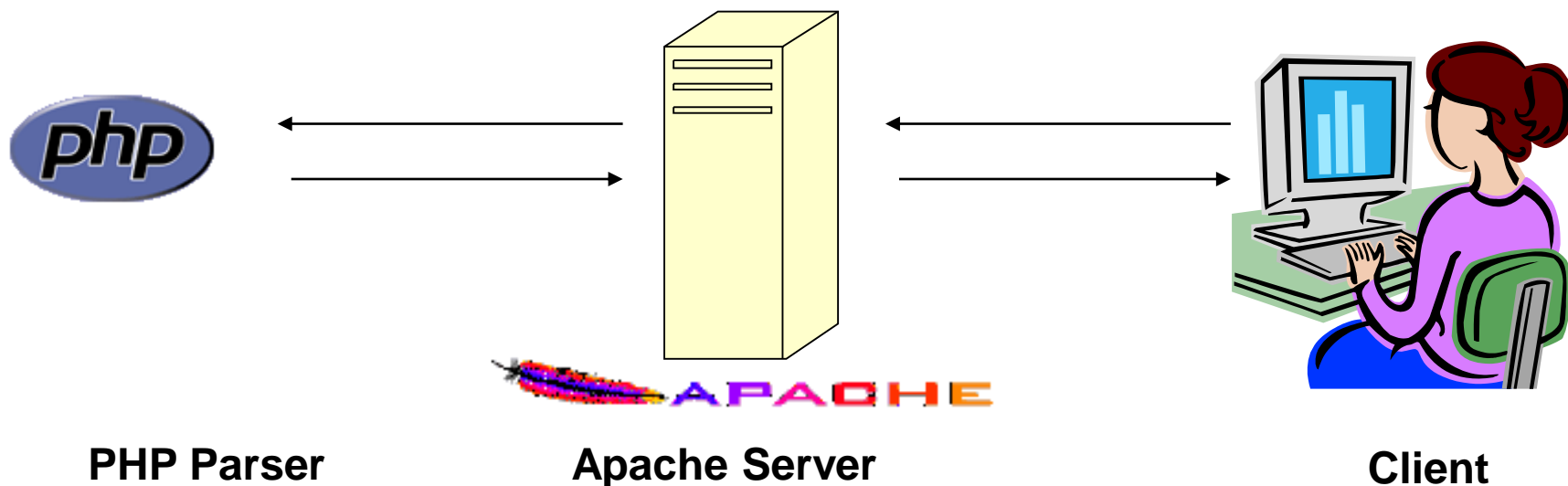
- Introduction
- 基本語法介紹
- 連結資料庫
- 結論

# Outline

- Introduction
- 基本語法介紹
- 連結資料庫
- 結論

# Introduction

- What is PHP?
  - PHP: Hypertext Preprocessor



# Introduction

## ■ What is PHP?

### ■ Example :

```
<html>
<head>
  <title>PHP Sample</title>
</head>

<body>

  <?php
    echo "I am a PHP scripit !!";
  ?>

</body>
</html>
```



# Introduction

- Outputs :
  - HTML
  - Images
  - PDF
  - Flash
  - XHTML
  - XML
- PHP has supports for a wide range of databases :

Adabas D	Ingres	Oracle (OCI7 and OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

# Introduction

## ■ 一個簡單的例子

```
<html>
<head>
  <title>PHP Test</title>
</head>
<body>
  <?php echo "Hello World<p>"; ?>
</body>
</html>
```

```
<html>
<head>
  <title>PHP Test</title>
</head>
<body>
  Hello World<p>
</body>
</html>
```



# Outline

- Introduction
- 基本語法介紹
- 連結資料庫
- 結論



# 基本語法介紹

- Escaping from HTML
  - `<?php . . . ?>`
  - `<script language="php"> . . . </script>`
  - `<? . . . ?>`
  - `<% . . . %>`

# 基本語法介紹

## ■ 指令的排法

```
<?php  
    echo "This is a test";  
?>
```

```
<?php echo "This is a test" ?>
```

# 基本語法介紹

## ■ Comments

- PHP supports 'C', 'C++' and Unix shell-style comments

```
<?php
// This is a one-line c++ style comment
echo "This is a test";

/* This is a multi line comment
   yet another line of comment */
echo "This is yet another test";

# This is shell-style style comment
echo "One Final Test";
?>
```

# 基本語法介紹

## ■ Types

- Boolean
- Integer
- Floating-point number (float)
- String
- Array
- Object

# Types

## ■ Boolean

- To specify a boolean literal, use either the keyword **TRUE** or **FALSE**. Both are case-insensitive.
- `$foo = True; // assign the value TRUE to $foo`

# Boolean

- 哪些東西是False?
  - the boolean **FALSE**
  - the integer 0 (zero)
  - the float 0.0 (zero)
  - the empty string, and the string "0"
  - an array with zero elements
  - an object with zero elements
  - the special type NULL (including unset variables)
  - Every other value is considered **TRUE** (including any resource).

# Types

## ■ Integers

```
$a = 1234; # decimal number  
$a = -123; # a negative number  
$a = 0123; # octal number (equivalent to 83 decimal)  
$a = 0x1A; # hexadecimal number (equivalent to 26 decimal)
```

- The size of an integer is platform-dependent.
- The maximum value of about two value is the usual value (that's 32 bits signed).
- PHP doesn't support unsigned integers.

# Types

- Floating point numbers

```
$a = 1.234; $a = 1.2e3; $a = 7E-10;
```



# Types

- Strings
  - single quoted
    - Ex: 'this is a string'
  - double quoted
    - Ex: "this is a string"
  - heredoc quoted

# double quoted

- Escaped characters

<code>\n</code>	linefeed
<code>\r</code>	carriage return
<code>\t</code>	horizontal tab
<code>\\</code>	backslash
<code>\\$</code>	dollar sign
<code>\"</code>	double-quote

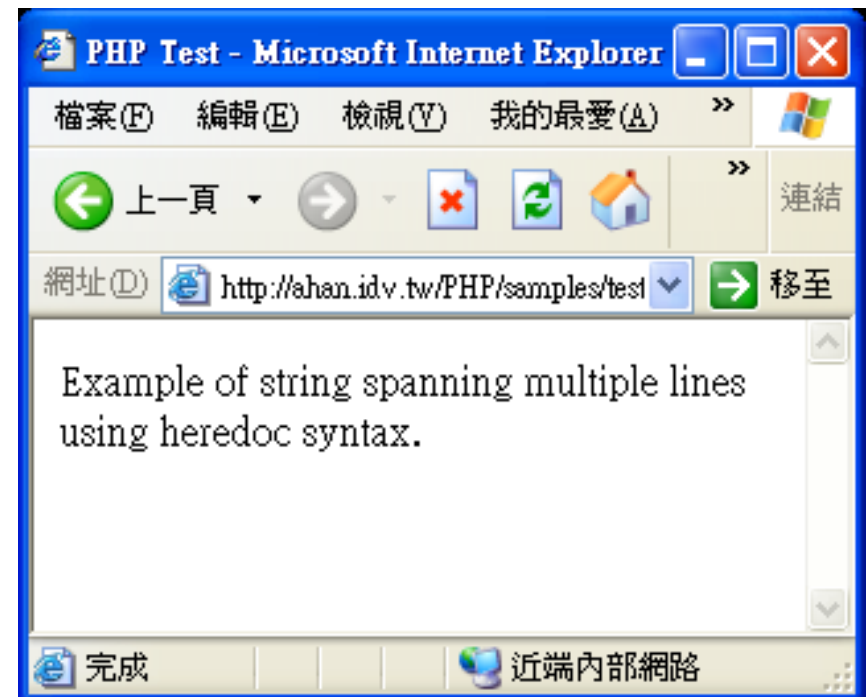
# Heredoc

```
$str = <<<EOD
```

Example of string  
spanning multiple lines  
using heredoc syntax.

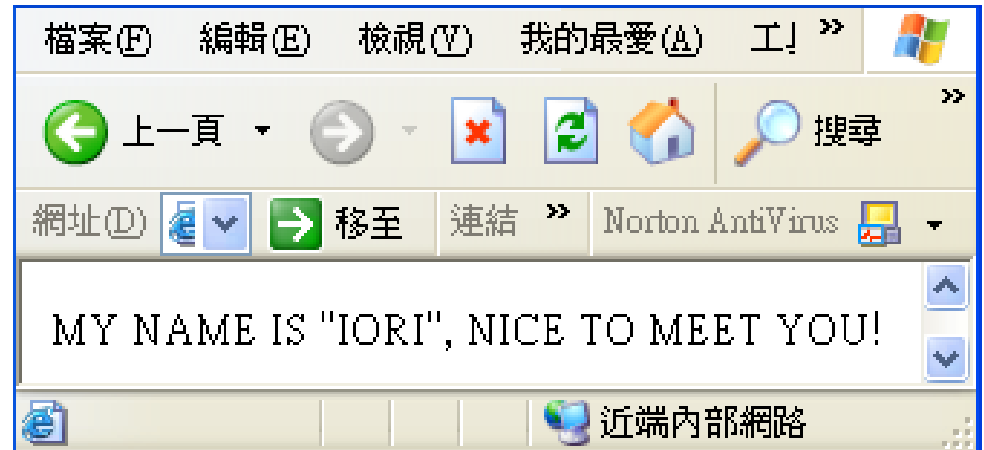
```
EOD;
```

```
echo $str;  
echo "<br>";
```



# Heredoc (Cont.)

```
<?php
$a='IORI';
echo <<<EOD
MY NAME IS "$a",
NICE TO MEET YOU!
EOD;
?>
```



# Variable parsing

```
$beer = 'Heineken';
echo "$beer's taste is great";
// works, "'" is an invalid
// character for varnames

echo "<br>He drunk some $beers";
// won't work, 's' is a valid
// character for varnames

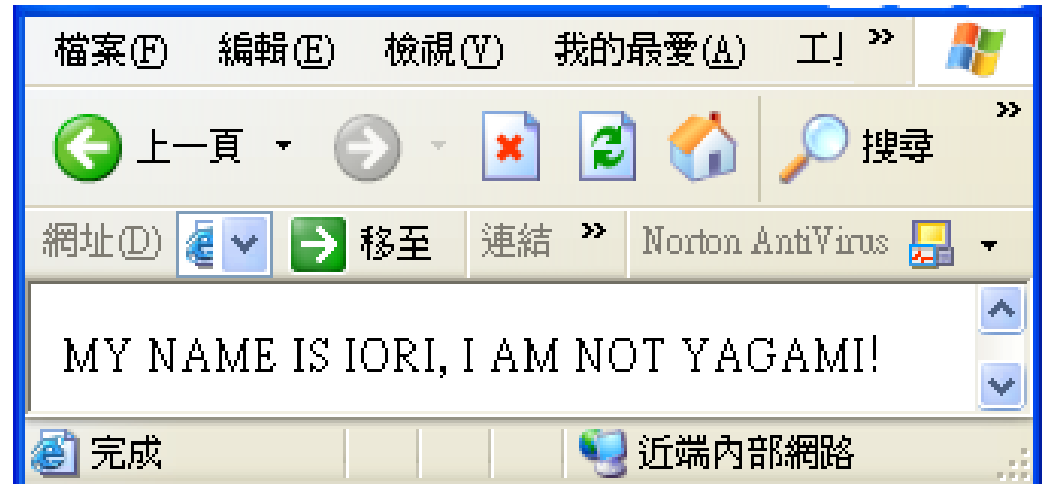
echo "<br>He drunk some ${beer}s";
// works
```



# Variable parsing (Cont.)

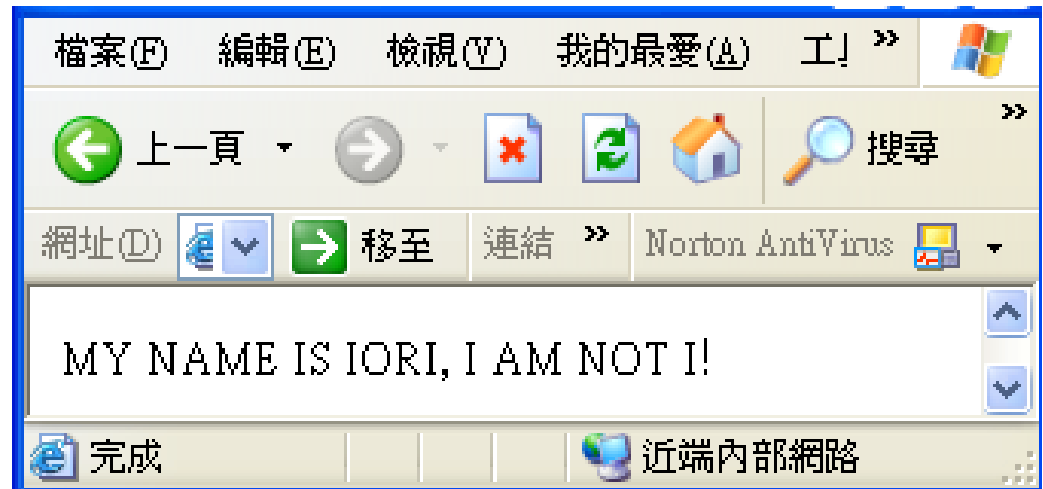
## ■ A complex example

```
<?php
$a='IORI';
$IORI='YAGAMI';
echo <<<EOD
MY NAME IS $a,
I AM NOT {$$a}!
EOD;
?>
```



# String access by character

```
<?php
$a='IORI';
$b=$a{0};
echo <<<EOD
MY NAME IS $a,
I AM NOT $b!
EOD;
?>
```



# String conversion

```

$foo = 1 + "10.5";           // $foo is float (11.5)
$foo = 1 + "-1.3e3";         // $foo is float (-1299)
$foo = 1 + "bob-1.3e3";      // $foo is integer (1)
$foo = 1 + "bob3";           // $foo is integer (1)
$foo = 1 + "10 Small Pigs";  // $foo is integer (11)
$foo = 4 + "10.2 Little Piggies"; // $foo is float (14.2)
$foo = "10.0 pigs " + 1;     // $foo is float (11)
$foo = "10.0 pigs " + 1.0;   // $foo is float (11)

```



# String Operators

- 用 "." 來結合String.

Ex:

```
$a = "Hello ";
$b = $a . "World!";
// now $b contains "Hello World!"
echo $b;
```

```
$a = "Hello ";
$a .= "World!";
// now $a contains "Hello World!"
echo "<br>".$a;
```



# Types

## ■ Array

- `$a = array( 1 => 'one', 2 => 'two', 3 => 'three' );`
- A key is either an integer or a string.
- A value can be anything.

# Array

## ■ 二維陣列

```
<?php
```

```
$a=array("key1" => array("keya" => "value1a",  
                        "keyb" => "value1b"),  
        "key2" => array("keya" => "value2a",  
                        "keyb" => "value2b"));
```

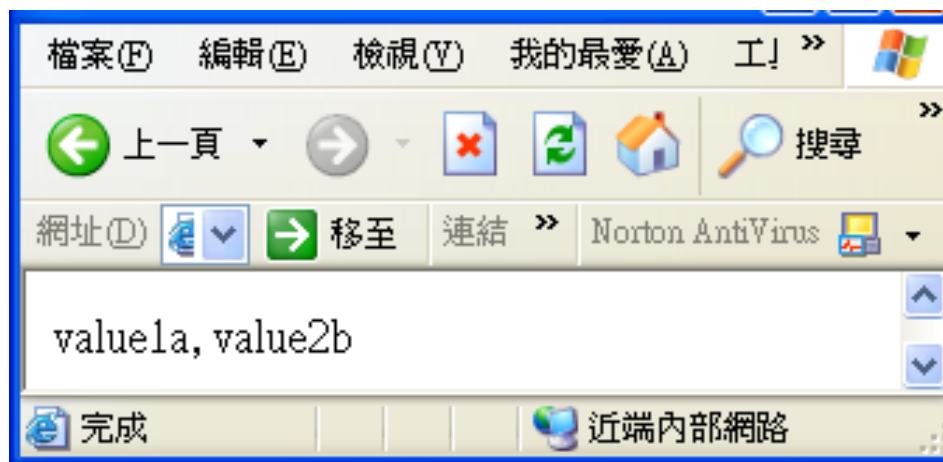
```
echo <<<EOD
```

```
{ $a[key1][keya] },
```

```
{ $a[key2][keyb] }
```

```
EOD;
```

```
?>
```



# Types

## ■ Objects

```
<?php
class test
{
    var $a="test";
}
$b=new test;
echo $b->a;
?>
```



# 變數命名

- case-sensitive
- starts with a letter or underline

Ex: 合法的: \$abc, \$Abc, \$\_4bc

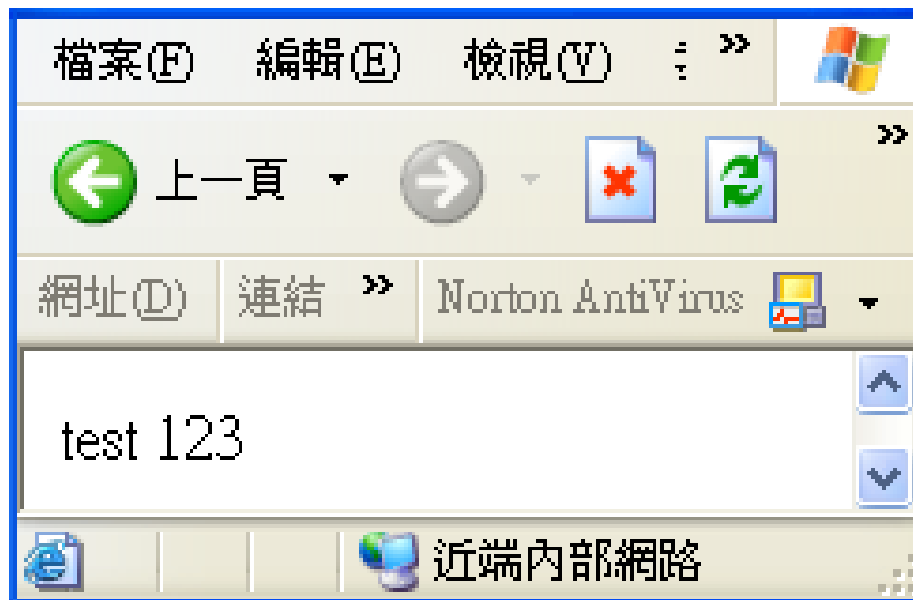
不合法的: \$4bc

# Reference

- To assign by reference, simply prepend an ampersand (&) to the beginning of the variable which is being assigned (the source variable).
- Only named **variables** may be assigned by reference.

Ex:

```
<?php
$a="test";
$b=&$a;
echo $b;
$b=" 123";
echo $a;
?>
```



# Constants

- You can define a constant by using the `define()` function.
- Only scalar data (`boolean`, `integer`, `float` and `string`) can be contained in constants.
- ```
define("CONSTANT", "Hello world.");  
echo CONSTANT; // outputs "Hello world."  
echo Constant;  
// outputs "Constant" and issues a notice.
```

# Expressions

```
$b = $a = 5;
```

```
$c = $a++;
```

```
$e = $d = ++$b;
```

```
$f = double($d++);
```

```
$g = double(++$e);
```

```
$h = $g += 10;
```



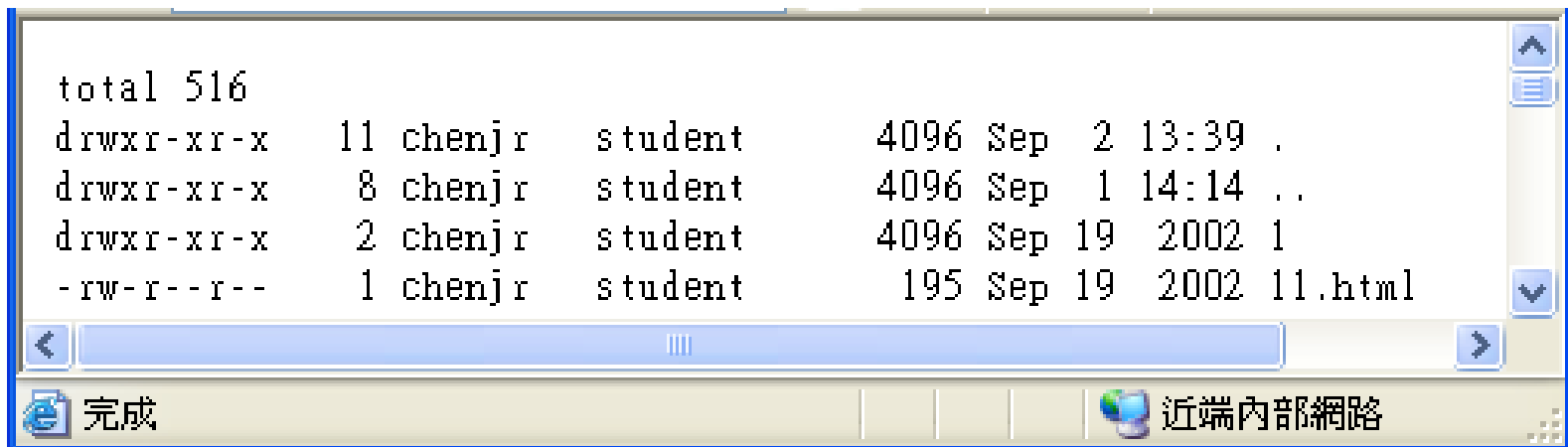
# Comparison Operators

| Example                       | Name                     | Result                                                                                                            |
|-------------------------------|--------------------------|-------------------------------------------------------------------------------------------------------------------|
| <code>\$a == \$b</code>       | Equal                    | <b>TRUE</b> if <code>\$a</code> is equal to <code>\$b</code> .                                                    |
| <code>\$a === \$b</code>      | Identical                | <b>TRUE</b> if <code>\$a</code> is equal to <code>\$b</code> , and they are of the same type. (PHP 4 only)        |
| <code>\$a != \$b</code>       | Not equal                | <b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> .                                                |
| <code>\$a &lt;&gt; \$b</code> | Not equal                | <b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> .                                                |
| <code>\$a !== \$b</code>      | Not identical            | <b>TRUE</b> if <code>\$a</code> is not equal to <code>\$b</code> , or they are not of the same type. (PHP 4 only) |
| <code>\$a &lt; \$b</code>     | Less than                | <b>TRUE</b> if <code>\$a</code> is strictly less than <code>\$b</code> .                                          |
| <code>\$a &gt; \$b</code>     | Greater than             | <b>TRUE</b> if <code>\$a</code> is strictly greater than <code>\$b</code> .                                       |
| <code>\$a &lt;= \$b</code>    | Less than or equal to    | <b>TRUE</b> if <code>\$a</code> is less than or equal to <code>\$b</code> .                                       |
| <code>\$a &gt;= \$b</code>    | Greater than or equal to | <b>TRUE</b> if <code>\$a</code> is greater than or equal to <code>\$b</code> .                                    |

# Execution Operators

- 當變數的值是用esc下面的” ` “括起來時，表示其為server上執行某指令的結果。

ex: `$output = `ls -al`;`  
`echo "<pre>$output</pre>";`



```
total 516
drwxr-xr-x  11 chenj  student   4096 Sep  2 13:39 .
drwxr-xr-x   8 chenj  student   4096 Sep  1 14:14 ..
drwxr-xr-x   2 chenj  student   4096 Sep 19  2002 1
-rw-r--r--   1 chenj  student    195 Sep 19  2002 11.html
```

# Logical Operators

| Example                         | Name | Result                                                                     |
|---------------------------------|------|----------------------------------------------------------------------------|
| <code>\$a and \$b</code>        | And  | TRUE if both <code>\$a</code> and <code>\$b</code> are TRUE.               |
| <code>\$a or \$b</code>         | Or   | TRUE if either <code>\$a</code> or <code>\$b</code> is TRUE.               |
| <code>\$a xor \$b</code>        | Xor  | TRUE if either <code>\$a</code> or <code>\$b</code> is TRUE, but not both. |
| <code>! \$a</code>              | Not  | TRUE if <code>\$a</code> is not TRUE.                                      |
| <code>\$a &amp;&amp; \$b</code> | And  | TRUE if both <code>\$a</code> and <code>\$b</code> are TRUE.               |
| <code>\$a    \$b</code>         | Or   | TRUE if either <code>\$a</code> or <code>\$b</code> is TRUE.               |

# if, else if, else

```
if ($a > $b) {
    print "a is bigger than b";
} elseif ($a == $b) {
    print "a is equal to b";
} else {
    print "a is smaller than b";
}
```

```
#####
```

```
if ($a == 5):
    print "a equals 5";
    print "...";
elseif ($a == 6):
    print "a equals 6";
    print "!!!";
else:
    print "a is neither 5 nor 6";
endif;
```

# for

```
for ($i = 1; $i <= 10; $i++) {  
    print $i;  
}
```

```
for ($i = 1; $i <= 10; $i++) :  
    print $i;  
endfor;
```

# while

```
$i = 1;  
while ($i <= 10) {  
    print $i++;  
}
```

```
$i = 1;  
while ($i <= 10):  
    print $i;  
    $i++;  
endwhile;
```

# do ... while

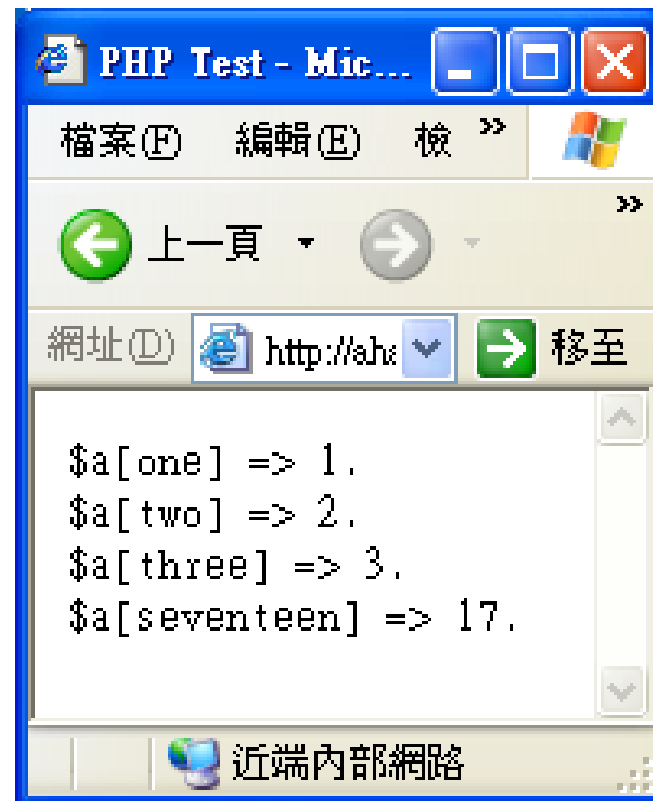
```
do {  
    if ($i < 5) {  
        print "i is not big enough";  
        break;  
    }  
    $i *= $factor;  
    if ($i < $minimum_limit) {  
        break;  
    }  
    print "i is ok";  
    ...process i...  
} while(0);
```

# foreach

- foreach(array\_expression as \$value)  
statement
- foreach(array\_expression as \$key => \$value)  
statement

```
$a = array (
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);

echo "<pre>";
foreach($a as $k => $v) {
    print "\$a[$k] => $v.\n";
}
echo "</pre>";
```





# User-defined functions

- In PHP 3, functions must be defined before they are referenced. No such requirement exists in PHP 4.
- PHP **does not support function overloading**, nor is it possible to undefine or redefine previously-declared functions.

```
function foo ($arg_1, $arg_2, ..., $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
```

# User-defined functions

- Pass an array

```
<?php
```

```
$a=array(5,5);
```

```
function add($input)
```

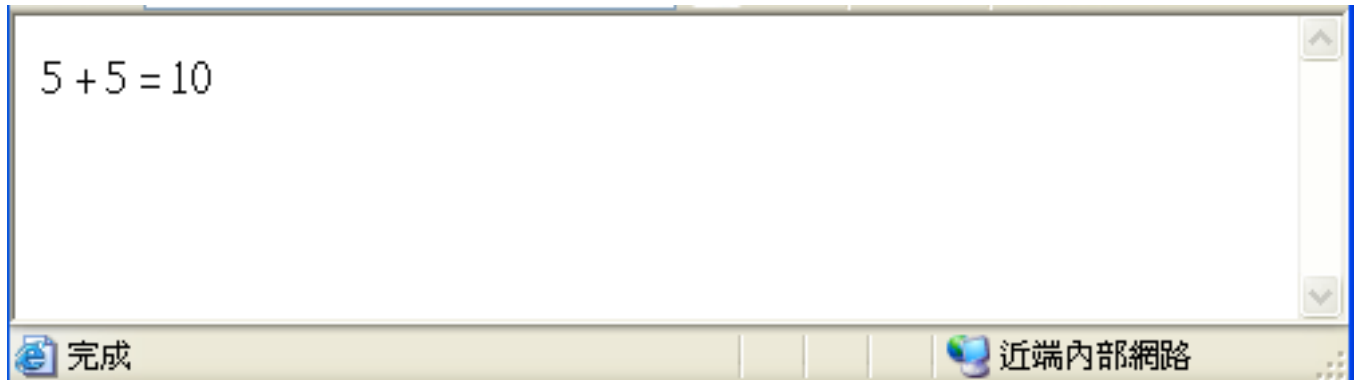
```
{
```

```
    echo "$input[0] + $input[1] = ",$input[0]+$input[1];
```

```
}
```

```
add($a);
```

```
?>
```



```
5 + 5 = 10
```

# User-defined functions

- passed by reference

```
function add_some_extra(&$string)
{
    $string .= 'and something extra.';
}
$str = 'This is a string, ';
add_some_extra($str);
echo $str;
```



# User-defined functions

## ■ Default argument values

```
function makecoffee ($type = "cappuccino")
{
    return "Making a cup of $type.\n";
}
echo makecoffee ();
echo makecoffee ("espresso");
```



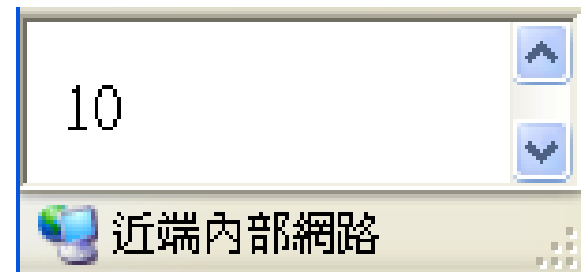
# Default argument values

- Any defaults should be on the right side of any non-default arguments.
  - `function makeyogurt ($type = "acidophilus", $flavour)` ☹️
  - `function makeyogurt ($flavour, $type = "acidophilus")` 😊

# User-defined functions

## ■ Returning values

```
<?php
$a=array(5,5);
function add($input)
{
    return $input[0]+$input[1];
}
echo add($a);
?>
```



# Global Variables

- In PHP global variables must be declared **global** inside a function if they are going to be used in that function.

```
<?php
```

```
$a=1;
```

```
$b=2;
```

```
function add()
```

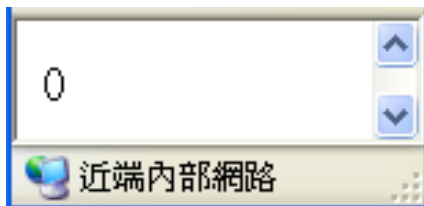
```
{
```

```
    return $a+$b;
```

```
}
```

```
echo add($a);
```

```
?>
```



```
<?php
```

```
$a=1;
```

```
$b=2;
```

```
function add()
```

```
{
```

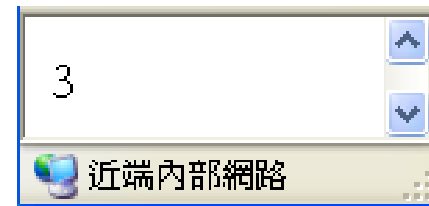
```
    global $a,$b;
```

```
    return $a+$b;
```

```
}
```

```
echo add($a);
```

```
?>
```



# Static variables

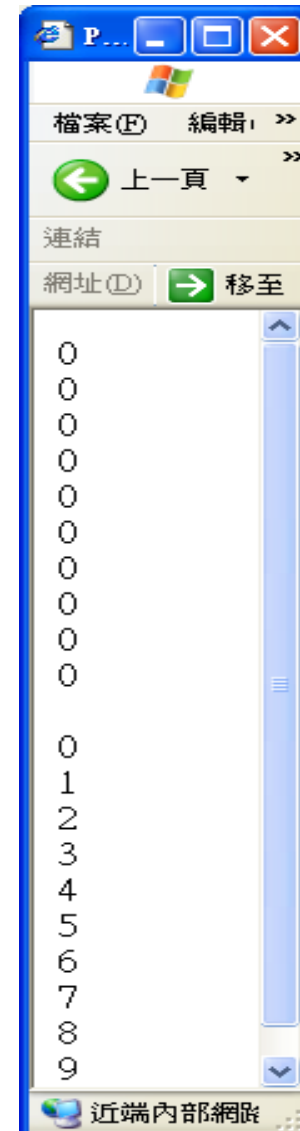
```
function Test ()
{
    $a = 0;
    echo $a;
    $a++;
}

function Test2()
{
    static $a = 0;
    echo $a;
    $a++;
}

for($i=0; $i<10; $i++)
{
    Test();
    echo "<br>";
}

echo "<p>";

for($i=0; $i<10; $i++)
{
    Test2();
    echo "<br>";
}
```





# class

- A class is a collection of variables and functions working with these variables.
- \$this: 和JAVA中的this相同, 指目前這個物件

```
class Cart
{
    var $items; // Items in our shopping cart

    // Add $num articles of $artnr to the cart
    function add_item ($artnr, $num)
    {
        $this->items[$artnr] += $num;
    }

    // Take $num articles of $artnr out of the cart
    function remove_item ($artnr, $num)
    {
        if ($this->items[$artnr] > $num) {
            $this->items[$artnr] -= $num;
            return true;
        } else {
            return false;
        }
    }
}
```

# extends

- multiple inheritance is not supported.
- Classes are extended using the keyword “**extends**”.

```
class Named_Cart extends Cart
{
    var $owner;

    function set_owner ($name)
    {
        $this->owner = $name;
    }
}
```

# Constructors

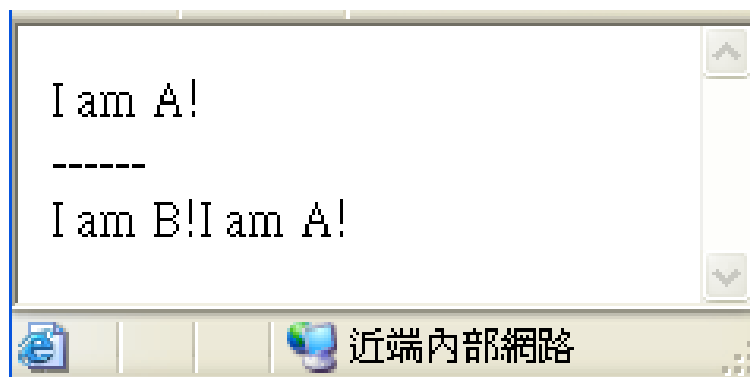
- Constructors are functions in a class that are automatically called when you create a new instance of a class with *new*.

```
// Works in PHP 3 and PHP 4.  
class Auto_Cart extends Cart  
{  
    function Auto_Cart()  
    {  
        $this->add_item ("10", 1);  
    }  
}
```

# :: operator

```
<?php
class A {
    function pr() {
        echo "I am A!";
    }
}
class B extends A {
    function pr() {
        echo "I am B!";
        A::pr();
    }
}
A::pr();
echo "<br>-----<br>";
$b=new B;
$b->pr();
?>
```

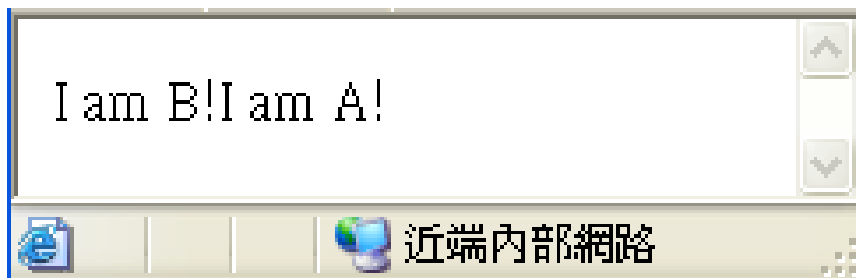
:: 有點像**JAVA**中**static**的觀念，可以讓你直接使用某類別中的成員



# parent

```
<?php
class A {
    function pr() {
        echo "I am A!";
    }
}
class B extends A {
    function pr() {
        echo "I am B!";
        parent::pr();
    }
}
$b=new B;
$b->pr();
?>
```

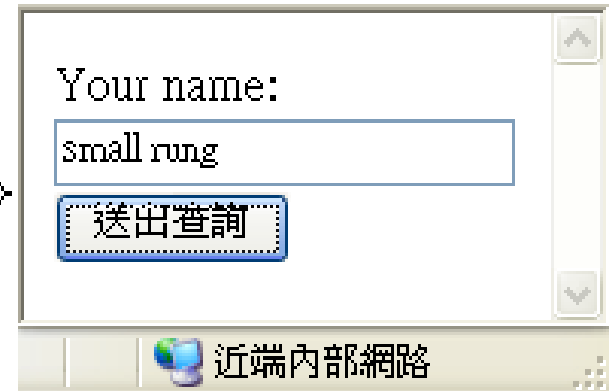
parent和Java中的super一樣，  
在子類別中可以使用父類別  
的成員



# Dealing with Forms

## ■ \$\_POST["field name"]

```
<form action="test.php" method="POST">
Your name: <input type="text" name="name">
<input type="submit">
</form>
```



test.php:

```
<?php
echo "Hi, ". $_POST["name"] ;
?>
```



# Outline

- Introduction
- 基本語法介紹
- 連結資料庫
- 結論

# 連結資料庫

- 在連結資料庫前, 必須先知道你安裝的php支援哪些資料庫.
- 請執行phpinfo(), 找” dbx”, 下面會列出支援的dbms.

## dbx

<b>dbx support</b>	enabled
<b>dbx version</b>	1.0.0
<b>supported databases</b>	MySQL ODBC PostgreSQL Microsoft SQL Server FrontBase Oracle 8 (not really) Sybase-CT

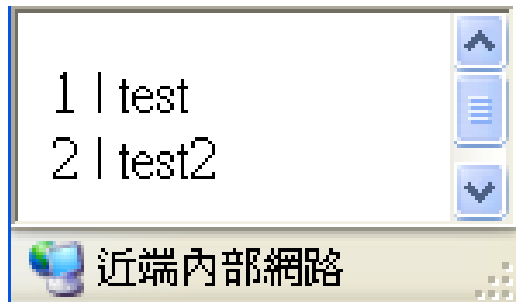


# 連結資料庫

- 接下來以mysql為例:
  - `mysql_connect("db.cse.nsysu.edu.tw");`
    - 連結到mysql所在的主機
  - `mysql_select_db("test");`
    - 選擇要用的資料庫
  - `$result = mysql_query("select * from test");`
    - 執行一SQL查詢, 並把結果放在\$result
  - `$row=mysql_fetch_row($result);`
    - 從\$result中拿出一row來

# 連結資料庫

```
<?php
mysql_connect("db.cse.nsysu.edu.tw");
mysql_select_db("test");
$result = mysql_query("select * from test");
$row=mysql_fetch_row($result);
echo $row[0]." | ".$row[1]."<br>";
$row=mysql_fetch_row($result);
echo $row[0]." | ".$row[1];
?>
```



```
mysql> select * from test;
+----+-----+
| i | c |
+----+-----+
| 1 | test |
| 2 | test2 |
+----+-----+
2 rows in set (0.00 sec)
```

# 連結資料庫

```
<?php
mysql_connect("db.cse.nsysu.edu.tw");
mysql_select_db("test");
mysql_query("insert into test values(3,'test3')");
?>
```

```
mysql> select * from test;
+----+-----+
| i | c      |
+----+-----+
| 1 | test  |
| 2 | test2 |
| 3 | test3 |
+----+-----+
3 rows in set (0.00 sec)
```

# Outline

- Introduction
- 基本語法介紹
- 連結資料庫
- 結論

# 結論

- PHP其實很容易學, 不要被一堆\$給嚇到了!