Internet and Web-Based Systems Development Lecture 7

Control Structures and Functions in PHP

Lecture 7

- Control structures
  - if
  - if-else
  - if-else if
  - switch
  - for
  - while
- Combining a form and its result
- Creating and Using Functions

Introduction

- PHP is a server side scripting language running on the Web server
- Shares the properties of common programming languages
  - Logic
  - Structure
- Syntaxes are very similar to C++ and Java
- PHP also possesses the concepts of control structures and functions, as in other programming languages

if structure

- Modify the “roll a dice” program to illustrate how **if** structure can be used
- When the program rolls one, a special message will be displayed

The Ace Program

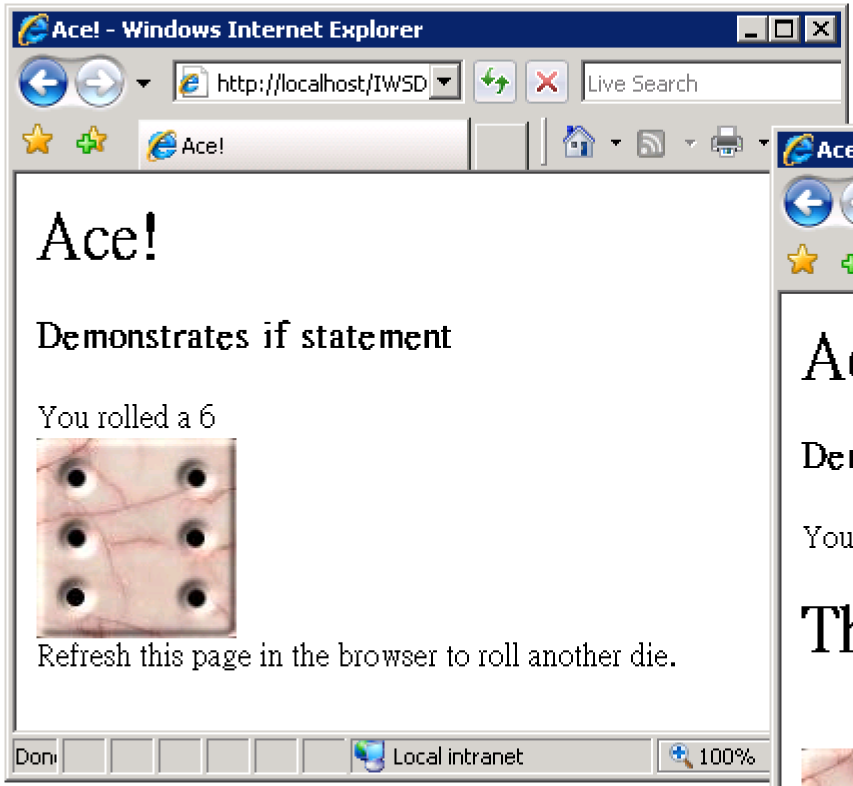
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
   Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
   transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Ace!</title>
</head>
<body>
<h1>Ace!</h1>
<h3>Demonstrates if statement</h3>
<?php
$roll = rand(1,6);
print "You rolled a $roll";
```

ace.php (Page 1 of 2)

```
if ($roll == 1){
  print "<h1>That's an ace!!!!!</h1>";
}
print "<br>";
print "<img src = \"die$roll.jpg\“ />";
?>
<br />
Refresh this page in the browser to roll another die.
</body>
</html>
```

ace.php (Page 2 of 2)

The Ace Program – Results

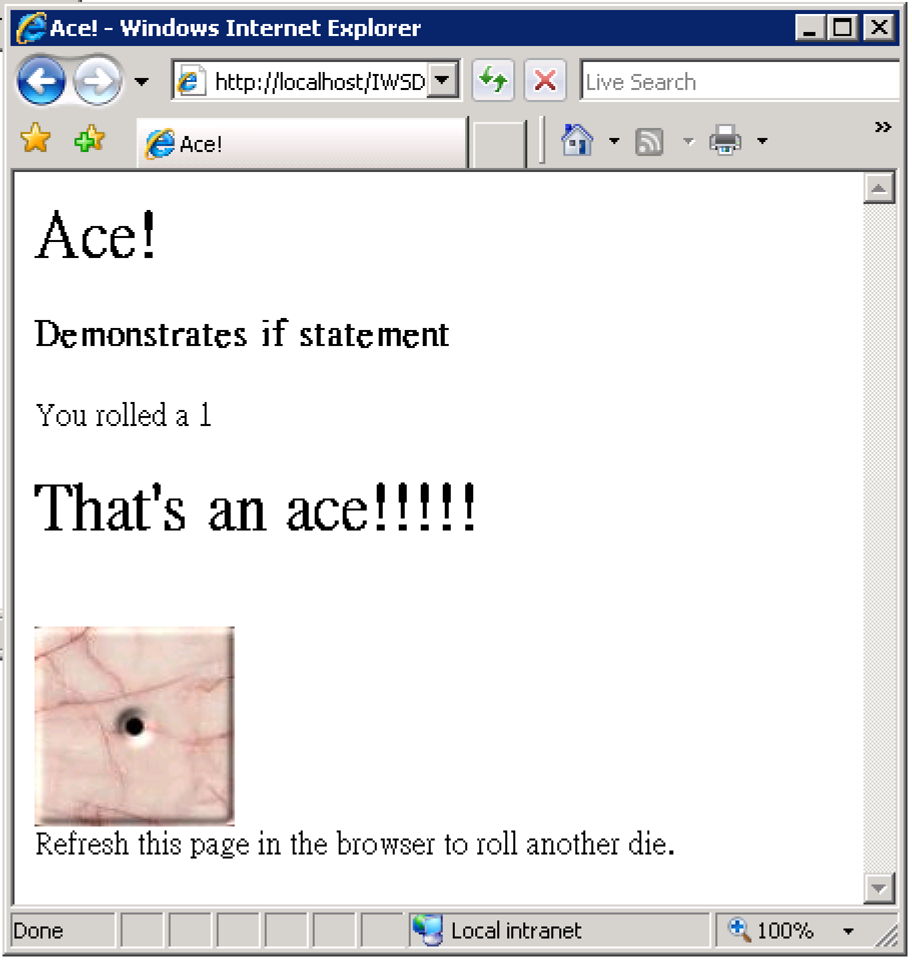


Ace!

Demonstrates if statement

You rolled a 6

Refresh this page in the browser to roll another die.



Ace!

Demonstrates if statement

You rolled a 1

That's an ace!!!!!

Refresh this page in the browser to roll another die.

if structure

- The syntax is basically the same as in C++ and Java
- Note that the condition checking for equality is indicated by two equal signs (==)
- Single equal sign (=) means assigning the value from the right to the variable in the left
- For example,

  $x = 5;

    - Put 5 into the variable x

Comparison Operators

Operator	Description
==	equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
!=	not equal to

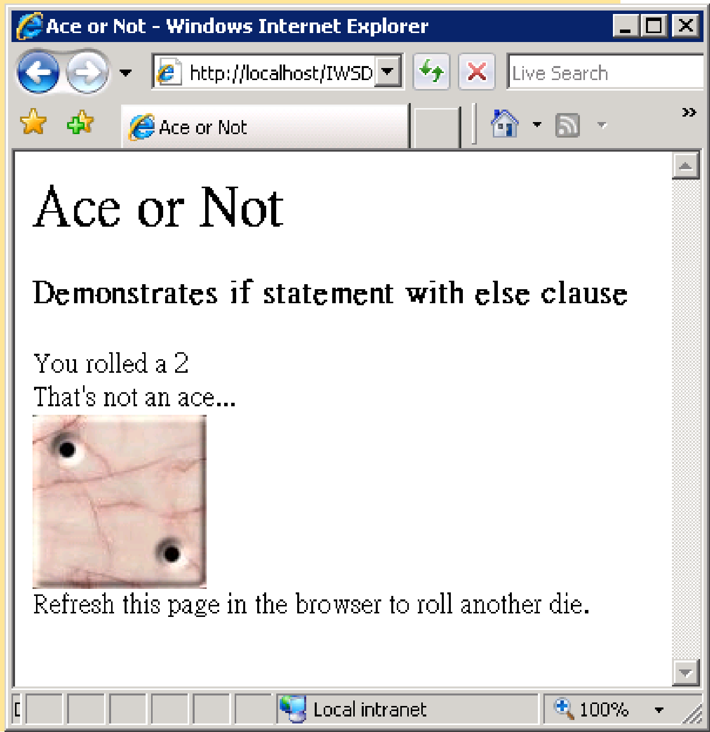
if-else structure

- Sometimes you want the program to do one thing if the condition is true, and something else if the condition is false
- The dice program is enhanced so that the web page displays a message also when the outcome is not *one*

The Ace or Not Program

```
...
<h1>Ace or Not</h1>
<h3>Demonstrates if statement with else clause</h3>
<?php
$roll = rand(1,6);
print "You rolled a $roll";
print "<br />";

if ($roll == 1){
  print "<h1>That's an ace!!!!!</h1>";
} else {
  print "That's not an ace...";
}

print "<br>";
print "<img src = \"die$roll.jpg\" />";
?>
<br>
Refresh this page in the browser to roll another die.
...
```

aceOrNot.php



Ace or Not

Demonstrates if statement with else clause

You rolled a 2
That's not an ace...

Refresh this page in the browser to roll another die.

if-else if structure

- Often you will find yourself working with more complex data
- For example, you might want to respond differently to each of the six possible rolls of a dice

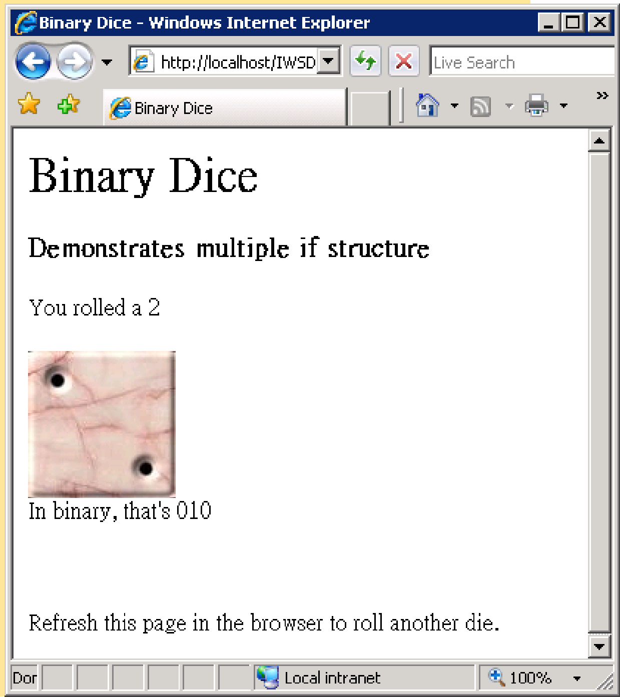
Binary Dice Program

binaryDice.php (Page 1 of 2)

```
...
<?php
$roll = rand(1,6);
print "You rolled a $roll";
print "<br>";

if ($roll == 1){
  $binValue = "001";
} else if ($roll == 2){
  $binValue = "010";
} else if ($roll == 3){
  $binValue = "011";
} else if ($roll == 4){
  $binValue = "100";
```

binaryDice.php (Page 2 of 2)

```
} else if ($roll == 5){
  $binValue = "101";
} else if ($roll == 6){
  $binValue = "110";
} else {
  // This block will never be run
  print "I don't know that one...";
}
print "<br />";
print "<img src = \"die$roll.jpg\" />";
print "<br />";
print "In binary, that's $binValue";
print "<br />";
print "<br />";
print "<br />";

?>

…
```



Binary Dice

Demonstrates multiple if structure

You rolled a 2

In binary, that's 010

Refresh this page in the browser to roll another die.

switch structure

- Sometimes you have to compare one variable to a number of possible values
- The following program illustrates how to use the **switch** structure to display the Roman numeral representation of the dice roll instead of the binary version

Switch Dice Program

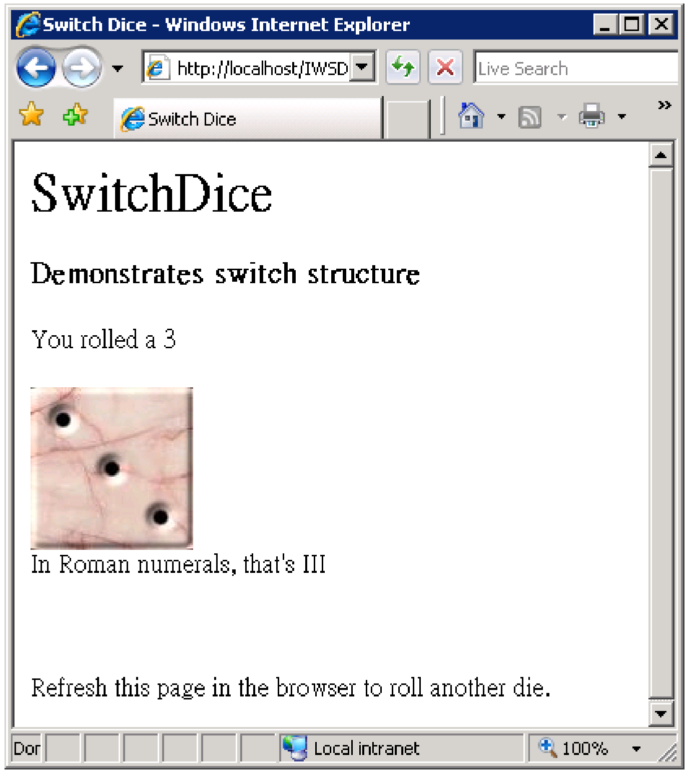
switchDice.php (Page 1 of 2)

```
<?php
$roll = rand(1,6);
print "You rolled a $roll";
print "<br>";
switch ($roll){
  case 1:
    $romValue = "I";
    break;
  case 2:
    $romValue = "II";
    break;
  case 3:
    $romValue = "III";
    break;
```

switchDice.php (Page 2 of 2)

```
  case 4:
    $romValue = "IV";
    break;
  case 5:
    $romValue = "V";
    break;
   case 6:
    $romValue = "VI";
    break;
  default:
    print "This is an illegal die!";
}
print "<br />";
print "<img src = \"die$roll.jpg\" />";
print "<br />";
print "In Roman numerals, that's $romValue";
print "<br />";
print "<br />";
print "<br />";
?>
```

Switch Dice Program


Switch Dice - Windows Internet Explorer

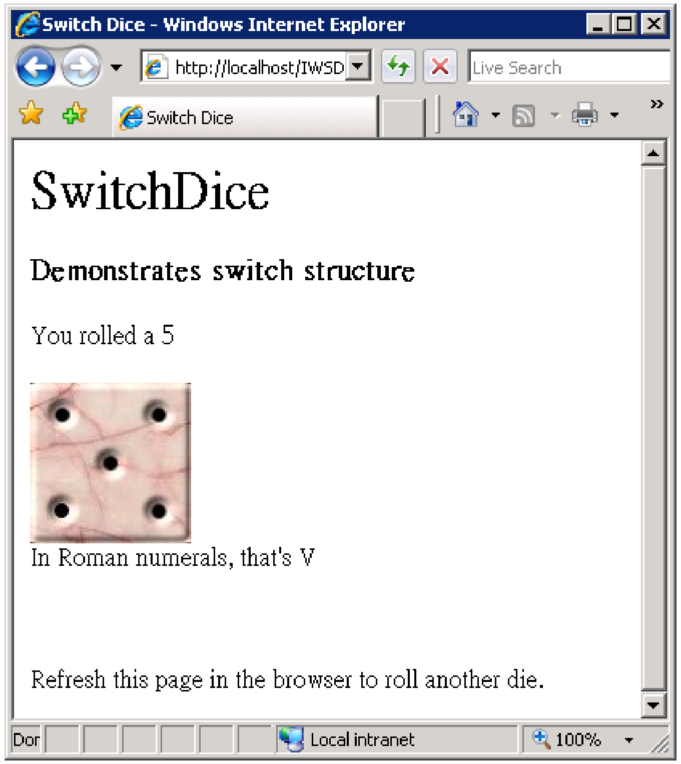
SwitchDice

Demonstrates switch structure

You rolled a 3

In Roman numerals, that's III

Refresh this page in the browser to roll another die.


Switch Dice - Windows Internet Explorer

SwitchDice

Demonstrates switch structure

You rolled a 5

In Roman numerals, that's V

Refresh this page in the browser to roll another die.

Combining a form and its result

- Most of your PHP programs up to now have had two distinct files
  - One contains the XHTML Form (.html)
  - One contains the PHP code (.php)
- Tedious to keep track of two separate files
- Use the **if** statement to combine both the form and the processing code into one page

Hi User Program

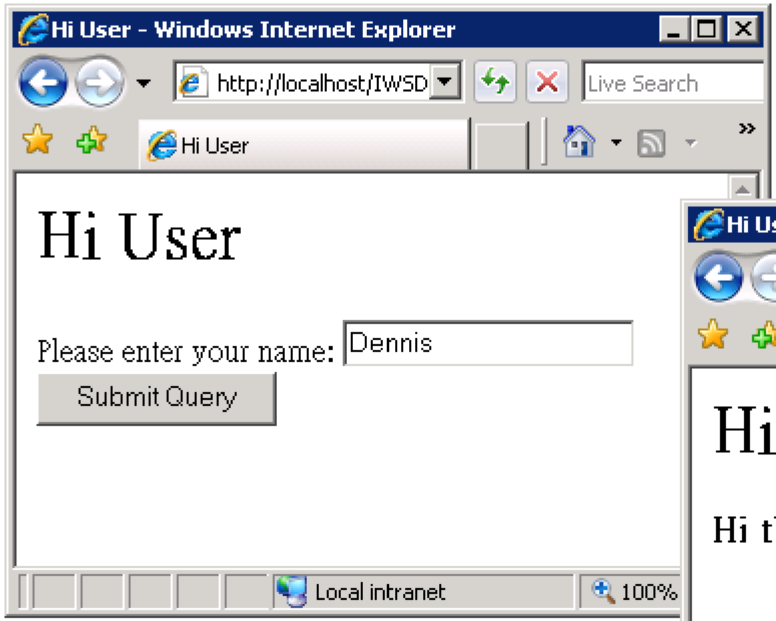
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Hi User</title>
</head>
<body>
<h1>Hi User</h1>
<?php
$userName = $_GET["userName"];
if (empty($userName)){
?>
```

hiUser.php (Page 1 of 2)

```
<form>
 Please enter your name:
 <input type = "text"
      name = "userName" />
 <br />
 <input type = "submit" />
</form>
<?php
} else {
 print "<h3>Hi there, $userName!</h3>";
}
?>
</body>
</html>
```

hiUser.php (Page 2 of 2)

Hi User Program


Hi User - Windows Internet Explorer

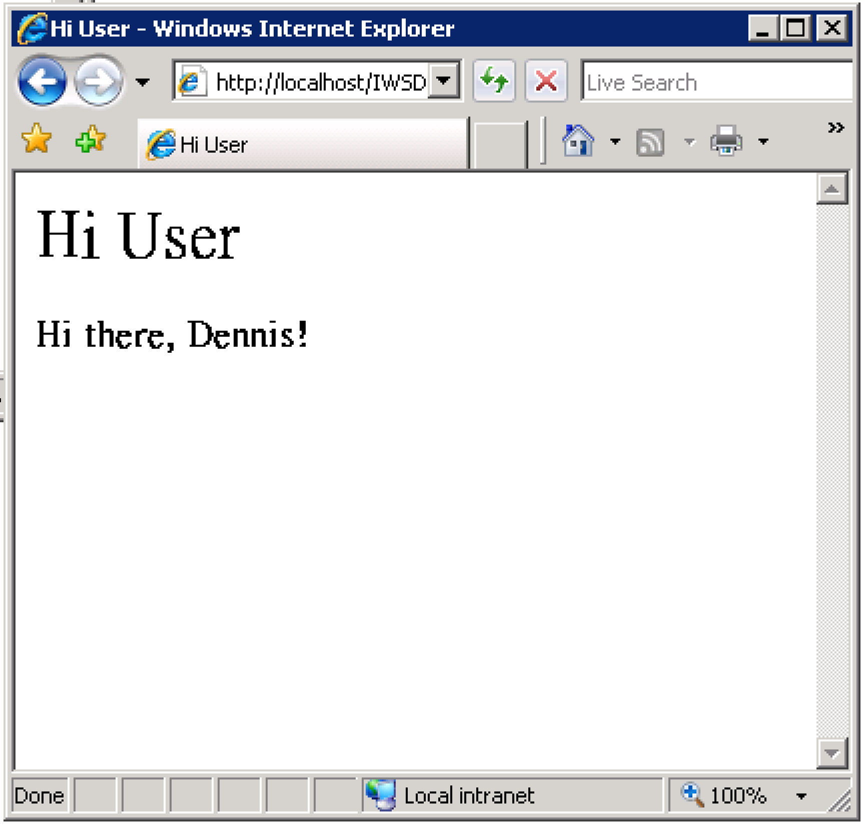
http://localhost/IWSD

Hi User

Please enter your name: Dennis

Submit Query

Local intranet 100%


Hi User - Windows Internet Explorer

http://localhost/IWSD

Hi User

Hi there, Dennis!

Done Local intranet 100%

Hi User Program

- If the <form> tag does not have the attributes action and method, by default, the form data will be sent back to itself
  - That means, it sends a HTTP request with the data to the server, requesting to load a page that is the same as itself and the method used is GET

Hi User Program

- The function empty() returns the value ***true*** if the specified variable is empty or ***false*** if it has a value
  - empty means
    - String value is ""
    - Numeric value is 0
- The condition empty($userName) will generally be ***true*** if this is the first time this page has been called
  - If it is ***true***, the program should generate a form so the user can enter his or her name
  - If it is ***false***, that means the user has entered a name so the program greets the user using that name

Creating Functions

- Code segments that are repeatedly used are normally organized into functions
- A function is like a miniature program
- The following PHP page demonstrates how functions are written and used

Creating Functions

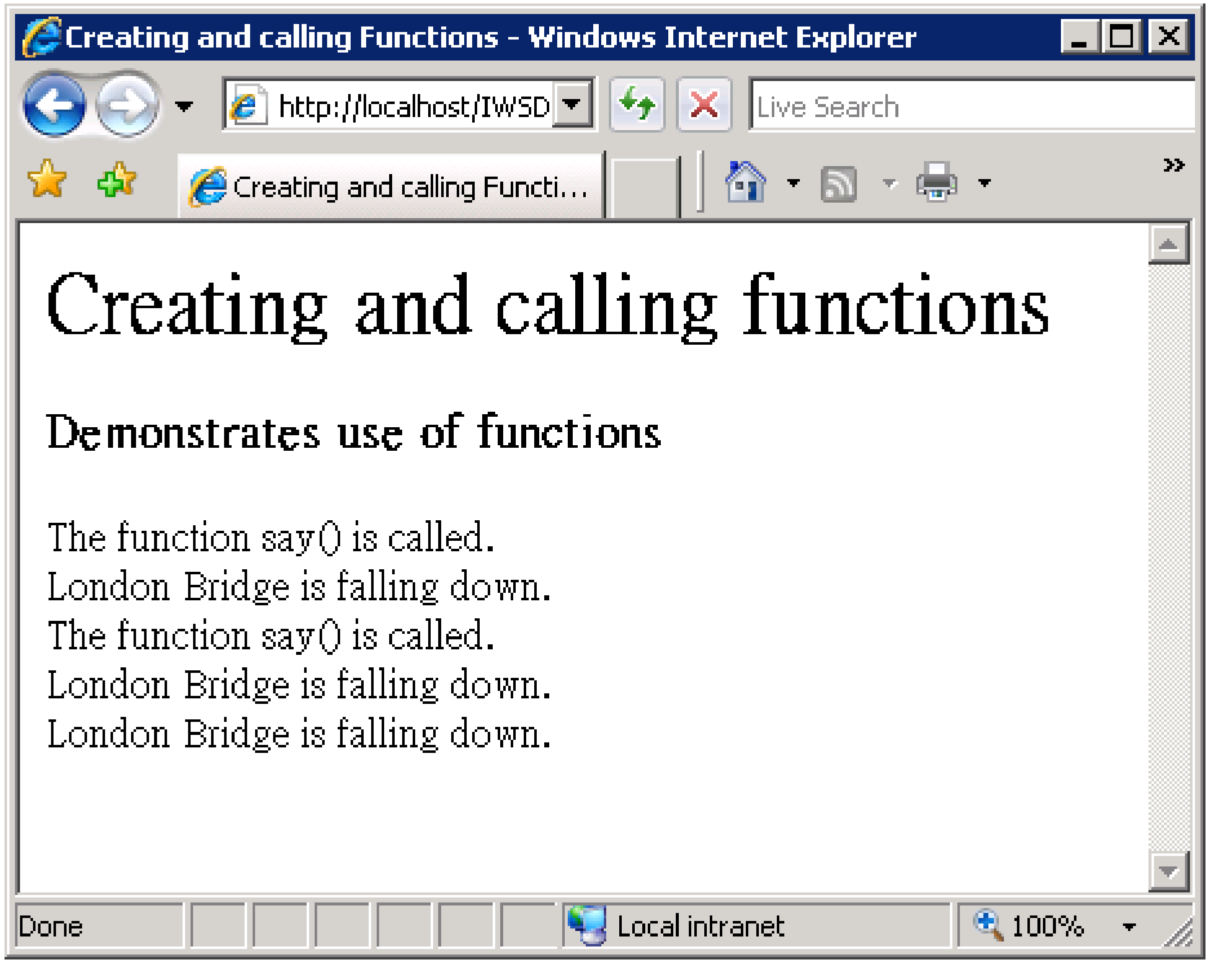
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
    Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Creating and calling Functions</title>
</head>
<body>
<h1>Creating and calling functions</h1>
<h3>Demonstrates use of functions</h3>
<?php

say();
sing();
```

functions.php (Page 1 of 2)

```
say();
sing();
sing();

function say(){
    print "The function say() is called.<br />";
}

function sing(){
    print "London Bridge is falling down.<br />";
}

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

functions.php (Page 2 of 2)

Creating Functions


Creating and calling Functions - Windows Internet Explorer
http://localhost/IWSD
Live Search
Creating and calling Functi...
Done
Local intranet
100%

Creating and calling functions

Demonstrates use of functions

```
The function say() is called.
London Bridge is falling down.
The function say() is called.
London Bridge is falling down.
London Bridge is falling down.
```

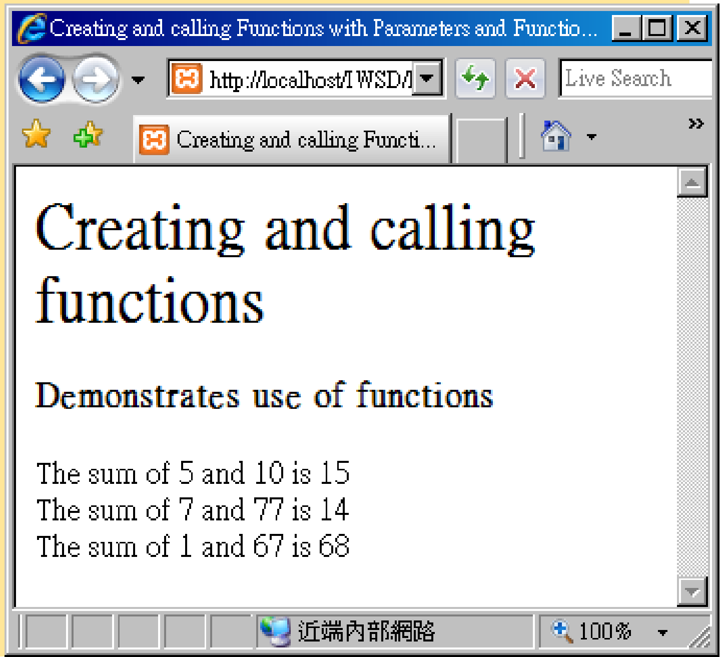
Creating Functions

- Unlike C++, the function declaration does not necessarily written at a position before the function is called
- Note the function declaration structure
  - Use the keyword function followed by the function’s name and a set of parentheses ()
  - A pair of brace { } to combine a series of code lines (statements) into one function

Using Parameters and Function Values

sum.php

```
...
<?php
print "The sum of 5 and 10 is " . sum(5, 10);
print "The sum of 7 and 77 is " . sum(7, 7);
print "The sum of 1 and 67 is " . sum(1, 67);

function sum($num1, $num2){
   $result = $num1 + $num2;
   return $result . "<br />";
}
?>
...
```


Creating and calling Functions with Parameters and Functio...
http://localhost/IWSD/
Live Search
Creating and calling Functi...
近端內部網路
100%

Creating and calling functions

Demonstrates use of functions

```
The sum of 5 and 10 is 15
The sum of 7 and 77 is 14
The sum of 1 and 67 is 68
```

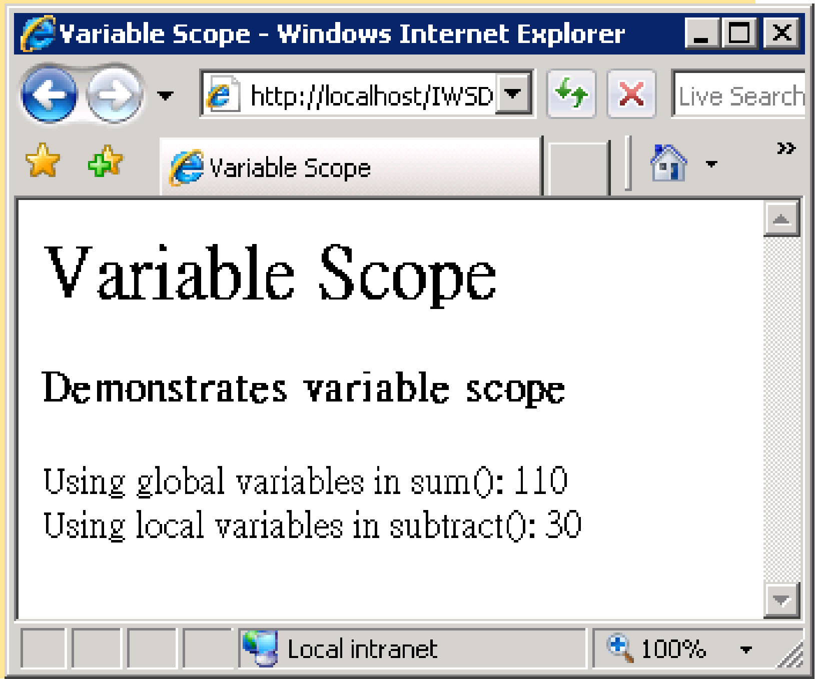
Using Parameters and Function Values

- To pass data to a function, use the parameter list
  - Each parameter is separated by a comma
  - To write a function
    - E.g., **`function sum($num1, $num2, …) {…}`**
  - To use a function
    - E.g., **`sum(1, 3);`**
- The variable created inside the function dies as soon as you leave the function
- Use the keyword **`return`** before the end of the function to pass a value back to the point where the function is called

Variable Scope

variableScope.php

```
<?php
   $a = 10;
   $b = 100;
   print "Using global variables in sum(): " . sum();
   print "Using local variables in subtract(): " . subtract();

   function sum() {
          global $a, $b;
          $result = $a + $b;
          return $result . "<br />";
   }
   function subtract() {
          $a = 50;
          $b = 20;
          $result = $a - $b;
          return $result . "<br />";
   }
?>
```



Variable Scope

Demonstrates variable scope

```
Using global variables in sum(): 110
Using local variables in subtract(): 30
```

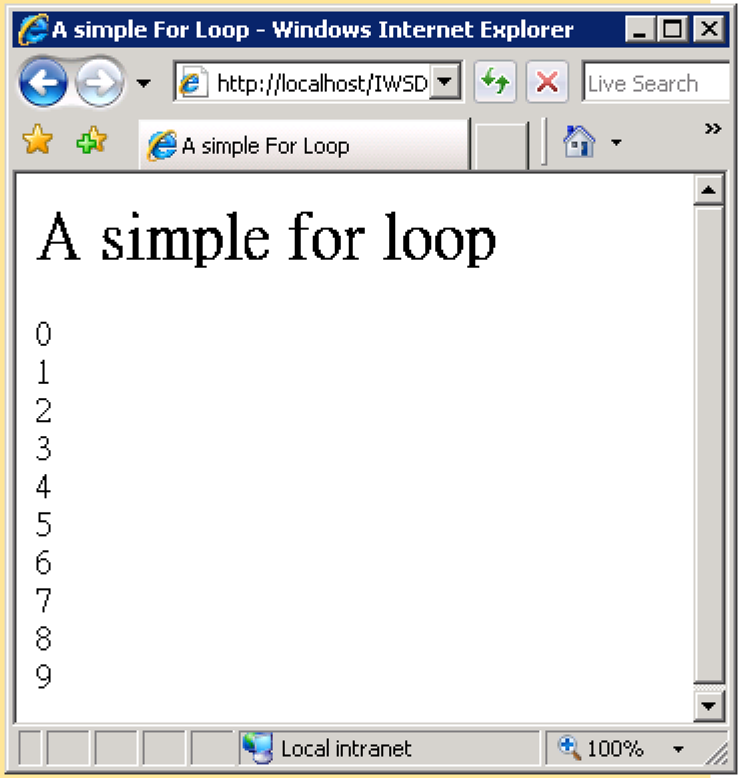
Variable Scope

- In PHP, you must explicitly request that a variable be global inside a function
- Use the keyword global to refer to a variable outside the function and in the *main level*
- If you do not do this, a new local variable with the same name will be created at the *function level*

Looping – **for** structure

for.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>A simple For Loop</title>
</head>
<body>
<h1>A simple for loop</h1>
<?php

    for ($i = 0; $i < 10; $i++){
      print "$i <br />\n";
    }

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


A simple For Loop - Windows Internet Explorer

http://localhost/IWSD

A simple for loop

```
0
1
2
3
4
5
6
7
8
9
```

Looping – **for** structure

- For performing repetitive task
- General format

```
for (initialization; LoopContinuationTest;
  increment) {
   statement(s)
}
```

- Example

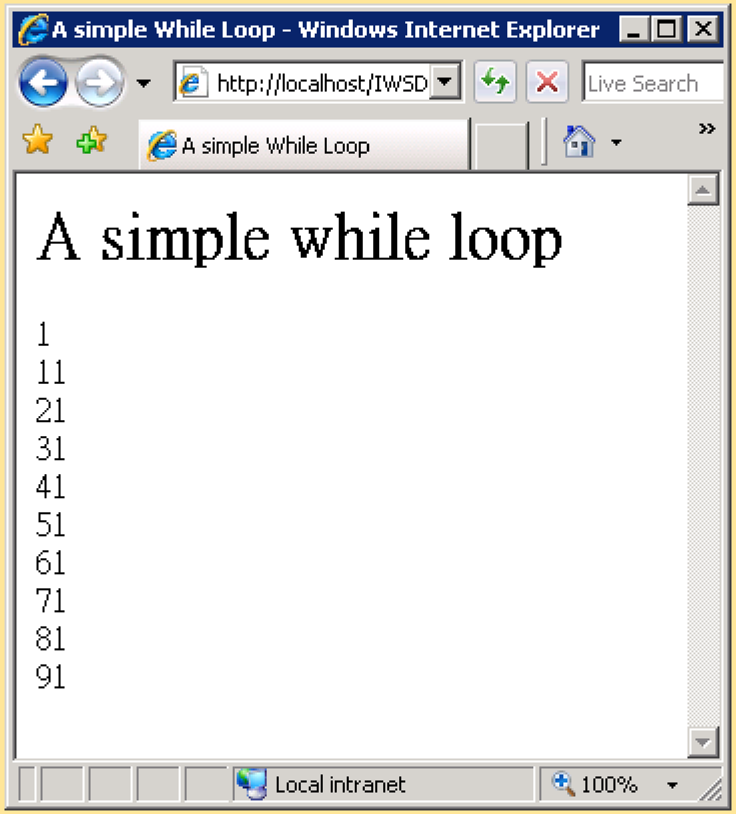
```
for($counter = 1; $counter <= 1000;
  $counter++) {
  print $counter;
}
```

  - Prints integers from 1 to 1000

Looping – **while** structure

while.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>A simple While Loop</title>
</head>
<body>
<h1>A simple while loop</h1>
<?php

$i = 1;
while ($i <= 100){
  print "$i <br />\n";
  $i += 10;
}
?>
</body>
</html>
```



A simple while loop

```
1
11
21
31
41
51
61
71
81
91
```

Looping – **while** structure

- for loops can usually be rewritten as while loops

```
initialization;
while (loopContinuationTest) {
  statement(s)
  increment;
}
```