INFO 151
Web Systems and Services

Week 7 (T2)

Dr Philip Moore

Dr Zhili Zhao

Course Overview

Weeks 1 – 3

- Introduction to Web Systems and Services
- Creating Web-Pages and Web-Sites with a Markup Language
- Introductory HTML 4 and HTML 5 with CSS

Weeks 4 – 6

- Client-Side Web Programming
- Object Oriented Programming
- Introductory and further JavaScript

Weeks 7 – 9

- Server-side Programming
- Introductory PHP
- Introduction to Database, SQL, and MySQL

Course Resources

- The sources of information and resources for JavaScript may be found at:
  - The w3schools.com web-site
  - The url: https://www.quanzhanketang.com/
- The w3schools.com web-site has limited resources for PHP and MySQL
  - The recommended course textbook for PHP and MySQL (including JavaScript) is:
    - Sams Teach Yourself PHP, MySQL & JavaScript All in One SIXTH EDITION
    - This book is available in the University Library

Review of JavaScript
and
Client-Side Systems

Overview

- In earlier tutorials we have considered:
  - Client-side server systems and programming
  - HTML and JavaScript
- In this session we will
  - Briefly review JavaScript and Client-Side Server Systems
  - Introduce server-side systems and programming in PHP
  - Introduce running PHP scripts in the NetBeans IDS
  - Introduce PHP and the PHP syntax looking at:
    - Strings and string methods including string formatting using `printf` and `sprintf`
    - Variable substitution in strings
    - Regular expressions
- There are significant similarities between JavaScript and PHP
  - We will identify the similarities and differences

PHP *vs* JavaScript

- Comparing PHP and JavaScript:
  - PHP and JavaScript serve different purposes in website development
  - PHP is a server-side scripting language
  - JavaScript is a client-side scripting language
- In practical ‘real-world’ website design and creation
  - Dynamic websites are created when we use the functions of both languages together
- In general:
  - JavaScript relates to the ‘look-and-feel’ of the website
  - PHP provides additional functionality including data processing and security

JavaScript

- JavaScript is NOT a fully-fledged programming language
- JavaScript is essentially a client-side application
- It offers limited scope for interactions with computer systems resources
- Importantly
  - It cannot natively (simply) connect to database systems
  - It is not a suitable language for processing inputs and data
- JavaScript provides a basis for
  - User interaction with forms
  - Controlling how user data is displayed in a web-browser

JavaScript and Client-Side Validation

- Client-side validation with JavaScript is optional but there are benefits which include
  - Faster response to the user (than in server-side validation)
  - Reduced loads on web-servers and network traffic
  - Implementation as interactive validation
    - Inputs and errors are checked as they occur
    - Field-by-field checking and reporting with individual error messages
- Client-side validation should be restricted to
  - Improving speed of response
  - Reducing the computational overhead (loading requirements)
  - Adding features

JavaScript and Web-Based Applications

- There are many common uses for JavaScript in web-based applications which include
- Simple interaction with form data
  - JavaScript is often used to calculate values
  - Display these values in an input widget (such as a warning dialog)
- Enhancing user interaction with web-pages with for example
  - Drop-down menus
  - Mouseover changes (the cursor is hovering over a hyperlink)
  - Dialog boxes (which may display information or a warning)
- Customising the web-browser output and using information from the browser to enhance the presentation

JavaScript and Events

- Most of the JavaScript functions are focused on events
- An event is an action that can be trapped through JavaScript code
- For example events commonly include:
  - A mouse passing over an object
  - A window opening
  - A user clicking a button on a form

JavaScript

- This review of JavaScript and client-side systems is not detailed or comprehensive
- Details may be found at:
  - The W3schools web-site (https://www.quanzhanketang.com/js/default.html)
  - Sams Teach Yourself PHP, MySQL & JavaScript All-in-One Sixth Edition

JavaScript and Client-Side Validation

JavaScript should **NEVER** replace server-side verification and validation

Server-side Programming
and
Hypertext Preprocessor (PHP)

The Advantages of Server-Side Scripting

- Client-side scripting requires:
  - Browsers need plugins and scripting technologies
  - Browsers may not support JavaScript (unusual) or it may be disabled
- Server-side scripting using the PHP scripting language:
  - Can reduce the load time for web pages
- Server-side scripting enhances security because:
  - Scripting takes place on the server the script itself is not sent to the browser (thus assisting in the prevention of copying / cloning/ or being scrutinised)
  - Server-side scripting offers greater protection for user privacy and is the preferred option for e-commerce / membership applications / and social media sites

PHP and Server-Side Server Systems

- Unlike JavaScript PHP is a server-side language
- PHP provides greatly enhanced functionality
  - As such it is similar to a fully functional high-level programming language such as Java or C++
- For example
  - PHP provides the capability to interact with database systems
  - Database systems and PHP will be introduced in week 8
    - In this course the focus is on MySQL
- Server-side systems
  - Provide enhanced security in validation and input checking
  - Provide less interactivity (by design)

Running PHP Scripts

(Review) Running PHP Programs

- To run a PHP program:
  - The XAMPP server must be started and running (in the background)
  - Use the Apache server
  - For MySQL programs the MySQL server must be started and running
- *JavaScript* project files are stored in the NetBeans Projects directory (folder) generally located in 'users' in the 'C' drive
- *PHP* project files are stored in the *htdocs* folder located in the XAMPP server directory which is generally located on the 'C' drive
- Recall: the *path* to the XAMPP server must be set in the PHP project in the NetBeans IDE

Php_Example - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Files | Services

- Php_Example
  - Source Files
    - index.php
  - Include Path
    - C:\xampp\php

Navigator

- html
  - head
    - meta
    - title
  - body
    - h2

Filters:

index.php

Source | History

```
1  <!DOCTYPE html>
2  <!--
3  PHP_Example
4  -->
5  <html>
6      <head>
7          <meta charset="UTF-8">
8          <title>PHP_Example</title>
9      </head>
10     <body>
11         <h2>This is text using HTML</h2>
12         <?php
13         echo "This is a simple example PHP
14         ?>
15     </body>
16 </html>
17
```

Applications are running

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

XAMPP Control Panel v3.2.2

Modules

Service	Module	PID(s)	Port(s)	Actions			
	Apache	6392 5168	80, 443	Stop	Admin	Config	Logs
	MySQL	2220	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

Config | Netstat | Shell | Explorer | Services | Help | Quit

```
11:19:39 [main]    XAMPP Version: 7.2.5
11:19:39 [main]    Control Panel Version: 3.2.2  [ Compiled: Nov 12th 2015 ]
11:19:39 [main]    You are not running with administrator rights! This will work for
11:19:39 [main]    most application stuff but whenever you do something with services
11:19:39 [main]    there will be a security dialogue or things will break! So think
11:19:39 [main]    about running this application with administrator rights!
11:19:39 [main]    XAMPP Installation Directory: "c:\xampp\"
11:19:39 [main]    Checking for prerequisites
11:19:44 [main]    All prerequisites found
11:19:44 [main]    Initializing Modules
11:19:44 [main]    Starting Check-Timer
11:19:44 [main]    Control Panel Ready
11:19:49 [Apache]  Attempting to start Apache app...
11:19:50 [mysql]   Attempting to start MySQL app...
11:19:54 [Apache]  Status change detected: running
11:19:55 [mysql]   Status change detected: running
```

17:1 INS

11:35 11/07/2018

Index of C:\xampp\htdocs

file:///C:/xampp/htdocs/

Index of C:\xampp\htdocs\

[parent directory]

Name	Size	Date Modified
dashboard/		31/07/2018, 13:59:03
img/		31/07/2018, 13:59:03
Php_HTML/		31/07/2018, 19:42:17
Php_HTML_1/		31/07/2018, 19:44:38
Php_test_1/		31/07/2018, 18:39:18
Php_Test_2/		31/07/2018, 18:41:38
Php_Test_3/		31/07/2018, 18:47:15
Php_Test_4/		31/07/2018, 18:52:05
Php_Test_5/		31/07/2018, 18:54:01
Php_Test_6/		31/07/2018, 19:01:59
webalizer/		31/07/2018, 13:58:58
xampp/		31/07/2018, 13:59:03
applications.html	3.7 kB	06/07/2018, 10:30:40
bitnami.css	177 B	27/02/2017, 09:36:00
favicon.ico	30.2 kB	16/07/2015, 16:32:32
index.php	260 B	16/07/2015, 16:32:32

- HTML and PHP project folders
- Note: do not change or delete the XAMPP system folders

Php_for - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Files | Services

- Php_for
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- html
  - head
    - meta
    - title
  - body
    - h2

index.php

Source | History

```
1  <!DOCTYPE html>
2  <!--
3  PHP for loop example
4  -->
5  <html>
6      <head>
7          <meta charset="UTF-8">
8          <title>PHP for loop example</title>
9      </head>
10     <body>
11         <h2>An example of a PHP for loop</h2>
12         <?php
13             for ($x = 0; $x <= 10; $x++) {
14                 echo "The number is: $x <br>";
15             }
16         ?>
17     </body>
18 </html>
19
```

Select web browser

Run script

Output - Browser Log

19:1 INS

Slide 28 of 71

JavaScript *vs* PHP
and
Programming in PHP

JavaScript and PHP

- The basic approach to programming using PHP and JavaScript is very similar
- There are many functions in PHP that are not available in JavaScript
- The syntax of PHP is similar to JavaScript
- There are syntax differences – the key differences are:
  - Variables in PHP are prefixed with a **$** sign
  - Local variables must be declared in JavaScript (**not** required in PHP)
  - Different opening and closing **<tags>** are used
  - String concatenation in JavaScript use the (**+**) sign / PHP uses a period (**.**)
- In the following slides the language basics of PHP and JavaScript are compared

Syntax Comparison (1)

Language Component	PHP	JavaScript
Opening and closing tags	<?php … >	<script> … </script>
Block statement	{…}	{…}
Multi-line comment	/* comment*/	/* comment*/
Single-line comment	// comment	// comment
Constant declaration	define (“a”, 1);	cons a = 1;
Variable declaration	Not required	Required for local variables (var a = 1;)
Variable assignment	$a = 0;	a = 0;
Assignment shortcut style	$a += 5;	a += 5;
Variable typing	at runtime	at runtime
Statement terminator	;	; (or the end of a line)
Equality type and value testing	Double-equals, ===	Double-equals, ===

Syntax Comparison (2)

Language Component	PHP	JavaScript
Equality type and value testing	Triple-equals, ===	Triple-equals, ===
Inequality testing	!=	!=
Strings	“string” ‘String’	“string” ‘String’
String constants	\n and \t	\n and \t
String concatenation	$a = $b . $c;	
Boolean values	true / false	true / false
Logical AND	&&	&&
Logical OR	||	||
Logical NOT	!	!

Programming in PHP

HTML JavaScript Template

```
<!DOCTYPE html>
<!- comment -->
<html>
<head>
<title></title>
</head>
<body>
      <!- enter HTML here -->
      <script>
      //enter JavaScript program code here //a comment example
      </script>
</body>
</html>
```

An HTML PHP Template

```
<!DOCTYPE html>
<!- comment -->
<html>
<head>
<title></title>
</head>
<body>
      <!- enter HTML here -->
      <?php
      //enter PHP program code here //a comment example
      # enter PHP program code here //a comment example
      ?>
</body>
</html>
```

Programming and PHP

- In JavaScript we have introduced
  - Sequential operations
  - Iteration operations (Loops)
  - Selection operations
- As for the majority of high-level programming languages (such as Java, C, C++, etc)
  - The essential structure of the processes is very similar
  - There are some syntax differences (we will identify these when strings and the programming functions are introduced for PHP)

PHP and Strings

PHP and Strings

- Recall that in week 4 we considered the JavaScript strings and string methods
- There are many similarities between JavaScript and PHP in the approach to using strings and string literals
  - There are differences in the way PHP uses strings and string literals
  - The differences relate to the syntax rather than the basic methods

String Literals

- A common task in a PHP script
  - Is to output literal sequences of characters to create messages
  - Such as error messages that appear on HTML pages
- A literal sequence of characters (a *string literal* – or simply a *string*) using both double ("…") and single ('…') quotes.

```
print 'This works'
print  "This works the same"
```

  - Both methods produce the same output
  - The syntax is the same for both PHP and JavaScript

Comparing JavaScript and PHP

- In JavaScript string output is achieved as follows

```
document.write("a string" + "another String")
var n = 5;
document.write("a string" + "another String" + n);
```

- In PHP string output is achieved using both **`echo`** and **`print`**
  - The difference between echo and print is **`echo`** can output more than one parameter (parameters comma separated)

```
print "Hello world!";
```

`echo "Hello world!";` (the same output as `print`)

`echo "Hello world!", 42;` (the output with a parameter)

Escaping in Strings

- Escaping special characters is a PHP string operates as for a JavaScript String
- Consider the following strings:

  `'This text, its’ created under JavaScript’` (creates an error)

  `'This text, its\’ created under JavaScript’` (correct)

  `'This text, its’ created under PHP’` (creates an error)

  `'This text, its\’ created under PHP’` (correct
- As can be seen the method and the result is the same

Newline Characters

- Unlike other programming languages PHP allows newline characters to be included directly in a string literal
- The following example shows the variable `$var` assigned with a string that contains a newline (`\n`) character

```
//this is ok. $var has a newline character
$var = 'The quick brown fox
jumps over the lazy dog'
```

- This feature is used in constructing SQL statements (used in MySQL) that are easier to read in PHP source code

String Literals and Database

- A common task in a PHP script is to access and operate on data in a *Relational Database Management System* (or *RDMS*)
- The following example Is used to construct a PHP statement in the *Structured Query Language* (or *SQL*)

```
$query = "SELECT max(order_id)
          FROM orders
          WHERE cust_id = $custID";
```

  - The variable (**`$query`**) is used to output the result

  The example shows the use of a *structured string* to produce a database search query – we will introduce RDMS (MySQL) in a later tutorial

Whitespace in Strings

Whitespace

- Whitespace is treated the same in PHP and JavaScript - PHP has additional methods and operators - an example is *trimming whitespace*
- There are tree trim functions: ***`ltrim`*** ***`rtrim`*** ***`trim`***
  - The syntax is:
  - ***`string trim(string subject[,string character_list]`*** (use ***`trim`*** *(or)* ***`rtrim`*** *(or)* ***`ltrim`***
- The *trim* functions return a copy of the *subject* string
  - ***`trim()`*** removes both leading and trailing whitespace characters
  - ***`ltrim()`*** removes leading whitespace characters
  - ***`rtrim()`*** removes trailing whitespace characters

Whitespace Trim Examples

- The following three examples show the effect of the trim methods

```
$var = trim(" Tiger Land \n");
```

  - The ***trim()*** result (output) = "Tiger Land" (no leading or trailing whitespace)

```
$var = ltrim(" Tiger Land \n");
```

  - The ***ltrim()*** result (output) = "Tiger Land \n" (no leading whitespace)

```
$var = rtrim(" Tiger Land \n");
```

  - The ***rtrim()*** result (output) = " Tiger Land" (no trailing whitespace)
- A range of characters may be specified using the optional *[character_list]* using two periods (..) as follows

```
$var = trim("30 JULY 2018, "0..9");
```
(trims digits and spaces)

```
print $var;
```
(prints "JULY")

PHP String Methods

Typical String Methods

String Methods and Operations
• Operations related to the length of a string
• Padding strings
• Changing the case of text (lowercase to uppercase (or) uppercase to lowercase)
• Comparing strings
• Finding and extracting substrings from strings
• Finding the position of a substring within a string
• Replacing characters and substrings
• Translating characters or substrings in a subject string

PHP String Methods

- `// is an inline comment in PHP`
- `echo strpos("Hello world!", "world");//output 6`
- `echo str_replace("world", "Dolly", "Hello world!");//output Hello Dolly!`
- `echo strlen("Hello world!");//output 12`
- `echo str_word_count("Hello world!");//outputs 2`
- `echo strrev("Hello world!");// output !dlrow olleH`

- Details of PHP methods can be found at:
  - https://www.quanzhanketang.com/php/php_string.html
  - Details of PHP may be found at *Learn PHP*

Php_String_Methods - Apache NetBeans IDE 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Services | Files

- Php_String_Methods
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- PHP
- CSS
  - Elements
    - SELECTOR
- HTML
  - html
    - head
      - meta
      - title
    - body
      - div
        - h4

index.php

Source | History

```
<!DOCTYPE html>
<!--
A PHP script to show some common string methods
s
-->
<html>
    <head>
        <meta charset="UTF-8">
        <title>String Methods</title>
    </head>
    <body>
        <div style="color: gold">
            <h4>
                The PHP script shows some typical commonly used string methods
            </h4>
        </div>
        <?php
            echo strpos("Hello world!", "world"); // outputs 6
            echo '<br>';
            echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly
            echo '<br>';
            echo strlen("Hello world!"); // outputs 12
            echo '<br>';
            echo str_word_count("Hello world!"); // outputs 2
            echo '<br>';
            echo strrev("Hello world!"); // outputs !dlrow olleH
        ?>
    </body>
</html>
```

Web Browser | String Methods

18:13 INS

Apache NetBeans 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Services | Files

- Php_String_Methods
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- PHP
- CSS
  - Elements
    - SELECTOR
- HTML
  - html
    - head
      - meta
      - title
    - body
      - div
        - h4

index.php

Source | History

```
<!DOCTYPE html>
<!--
A PHP script to show some common string methods
s
-->
<html>
    <head>
        <meta charset="UTF-8">
        <title>String Methods</title>
    </head>
    <body>
        <div style="color: gold">
            <h4>
                The PHP script shows some typical commonly used string methods
            </h4>
```

String Methods

http://localhost/Php_String_Methods/index.php

100%

The PHP script shows some typical commonly used string methods

6
Hello Dolly!
12
2
!dlrow olleH

Web Browser

18:13 INS

Printing and Formatting Strings

PHP Variable Substitution in Strings

Variable Substitution in PHP Scripts

- Variable substitution in a string literal (using （"…"）) replaces the variable name with the value of the variable
- An example is shown in the following code:

```
$number = 45;
$vehicle = "bus";
$message = "This $vehicle holds $number people";
print $message;
```

- The output is:

  "This bus holds 45 people"

`printf`
and
`sprintf`

printf and sprintf

- PHP provides advanced printing (output) methods for situations where a more complex output format is required
  - The **printf** and **sprintf** methods are modelled on 'C'
  - The syntax is as follows

```
String sprintf (string format [,mixed args … ])
String printf (string format [,mixed args … ])
```

  - The following example prints: "Result = 3.14"

```
$variable = 3.14159 //or PI
printf("Result: '%.2f'\n", $variable);
```

`printf` and `sprint` Differences

- There is a difference in the way `printf` and `sprintf` functions in PHP
- `Sprintf`
  - Is returned as a string
- `printf`
  - Sends the output directly to the output buffer
  - PHP uses the output buffer to build an HTTP response.
- The use of these output methods will be decided by the requirements of specific web-site design

`printf` and `sprintf` operator types

Operator Type	Function Description
%%	A literal percent character
%b	An integer formatted as a binary character (the number 5 in the binary system is: **101**)
%c	An integer formatted as an ASCII character
%d	An integer formatted as a signed decimal number (can hold numbers: **1** and **-1**)
%u	An integer formatted as an unsigned decimal number (can only hold numbers: **1**)
%o	An integer formatted as an octal number (the base 8 number system)
%x (or) **%X**	An integer formatted as a hexadecimal number (the base 16 number system / using lowercase and uppercase letters)
%f	A float formatted with a specified number of decimal places
%s	A string

ASCII Codes

- Brief History of ASCII code:
  - The American Standard Code for Information Interchange, or ASCII code, was created in 1963 by the "American Standards Association" Committee or "ASA"
  - The agency changed its name in 1969 by "American National Standards Institute" or "ANSI" as it is known since
- ASCII codes relate to characters and numbers plus other symbols (e.g., all keys on the computer keyboard have an ASCII code.
- For a comprehensive discussion and list of ASCII codes see:

https://theasciicode.com.ar

Variable Substitution in `printf`

Variable Substitution in **printf**

- The following PHP code example shows:
  - The creation of a variable ($n = 12)
  - Variable substitution with the **printf** string formatting
  - The output using %f (prints a floating point number)

```
<?php
    $n = 12;
    printf("%f", $n);
?>
```

- The output is:

  "12.000"

- The following slides show a worked **printf** examples and the output

Php_printf_example - Apache NetBeans IDE 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Services | Files

- Php_printf_example
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- PHP
- CSS
  - Elements
    - SELECTOR
- HTML
  - html
    - head
      - meta
      - title
    - body
      - div
        - h4
      - hr
      - hr
      - hr

index.php

Source | History

```
<!DOCTYPE html>
<!--
A PHP printf example showing formatted output
-->
<html>
    <head>
        <meta charset="UTF-8">
        <title>PHP printf example</title>
    </head>
    <body>
        <div style = "color: royalblue">
            <h4>
                A PHP script which demonstrates the printf operators
            </h4>
        </div>
        <hr>
        <?php
            $number = 9;
            $str = "Beijing";
            printf("There are %u million bicycles in %s.",$number,$str);
        ?>
        <hr>
        <?php
            $num1 = 123456789;
            $num2 = -123456789;
            $char = 50; // The ASCII Character 50 is 2
            // Note: The format value "%%" returns a percent sign
            printf("%%b = %b <br>",$num1); // Binary number
            printf("%%c = %c <br>",$char); // The ASCII Character
            printf("%%d = %d <br>",$num1); // Signed decimal number
            printf("%%d = %d <br>",$num2); // Signed decimal number
```

Web Browser | PHP printf example

50:1 INS

Php_printf_example - Apache NetBeans IDE 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Services | Files

- Php_printf_example
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- PHP
- CSS
  - Elements
    - SELECTOR
- HTML
  - html
    - head
      - meta
      - title
    - body
      - div
        - h4
      - hr
      - hr
      - hr

index.php

Source | History

```
            printf("There are %u million bicycles in %s.",$number,$str);
        ?>
        <hr>
        <?php
            $num1 = 123456789;
            $num2 = -123456789;
            $char = 50; // The ASCII Character 50 is 2
            // Note: The format value "%%" returns a percent sign
            printf("%%b = %b <br>",$num1); // Binary number
            printf("%%c = %c <br>",$char); // The ASCII Character
            printf("%%d = %d <br>",$num1); // Signed decimal number
            printf("%%d = %d <br>",$num2); // Signed decimal number
            printf("%%e = %e <br>",$num1); // Scientific notation (lowercase)
            printf("%%E = %E <br>",$num1); // Scientific notation (uppercase)
            printf("%%u = %u <br>",$num1); // Unsigned decimal number (positive)
            printf("%%u = %u <br>",$num2); // Unsigned decimal number (negative)
            printf("%%f = %f <br>",$num1); // Floating-point number (local settings aware)
            printf("%%F = %F <br>",$num1); // Floating-point number (not local sett aware)
            printf("%%g = %g <br>",$num1); // Shorter of %e and %f
            printf("%%G = %G <br>",$num1); // Shorter of %E and %f
            printf("%%o = %o <br>",$num1); // Octal number
            printf("%%s = %s <br>",$num1); // String
            printf("%%x = %x <br>",$num1); // Hexadecimal number (lowercase)
            printf("%%X = %X <br>",$num1); // Hexadecimal number (uppercase)
            printf("%%+d = %+d <br>",$num1); // Sign specifier (positive)
            printf("%%+d = %+d <br>",$num2); // Sign specifier (negative)
        ?>
        <hr>
    </body>
</html>
```

Web Browser | PHP printf example

50:1 INS

Php_printf_example - Apache NetBeans IDE 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects | Services | Files

- Php_printf_example
  - Source Files
    - index.php
  - Include Path
    - C:\xampp

Navigator

- PHP
- CSS
  - Elements
    - SELECTOR
- HTML
  - html
    - head
      - meta
      - title
    - body
      - div
        - h4
      - hr
      - hr
      - hr

index.php

```
3  A PHP printf example showing formatted output
```

PHP printf example

http://localhost/Php_printf_example/index.php

A PHP script which demonstrates the printf operators

There are 9 million bicycles in Beijing.

%b = 11101011011110011010001010 1
%c = 2
%d = 123456789
%d = -123456789
%e = 1.234568e+8
%E = 1.234568E+8
%u = 123456789
%u = 18446744073586094827
%f = 123456789.000000
%F = 123456789.000000
%g = 1.23457e+8
%G = 1.23457E+8
%o = 726746425
%s = 123456789
%x = 75bcd15
%X = 75BCD15
%+d = +123456789
%+d = -123456789

Regular Expressions

Regular Expressions and Strings

- Many programming languages (including PHP) there use *regular expressions* which enable sophisticated pattern matching
- However: *regular expressions* are less efficient (greater computational overhead) than other native PHP methods
- An important use of *regular expression* is to manage input strings and input validation

Regular Expressions and Strings

- Many programming languages (including PHP) there use *regular expressions* to enable sophisticated pattern matching
- However: *regular expressions* are less efficient (greater computational overhead) than other native PHP methods
- The following simple example shows a *regular expression*

```
if (ereg( "cat", "raining cats and dogs" ) )
```

  - The *regular expression* cat matches the *subject* string and the output prints "Found '*cat*'";
- An important use of *regular expression* is to manage input strings and input validation

Review

- In this tutorial we have:
  - Briefly reviewed JavaScript and Client-Side Server Systems
  - Introduced server-side systems and programming in PHP
  - Introduced running PHP scripts in the NetBeans IDS
  - Introduced PHP and the PHP syntax looking at:
    - Iteration operations (loops)
    - Strings and string methods including string formatting using `printf` and `sprintf`
    - Variable substitution in strings
    - Regular expressions
  - Demonstrated the similarities and differences between JavaScript and PHP

Conclusions

- There are similarities and differences between JavaScript and PHP
  - The differences are mainly in the language syntax
  - Using the incorrect syntax will produce errors which are hard to find
- In the following tutorial we will extend our introduction to PHP