**NSWI096 – Internet**

5th December 2011

# PHP

**Martin Svoboda**

Department of Software Engineering
Faculty of Mathematics and Physics
**Charles University in Prague**

XRG
XML Web Engineering
research group

# PHP

- PHP = Hypertext Processor
  - http://www.php.net/
- Versions
  - 4
  - 5
    - Improved model of classes
- Language syntax
  - Based on C++ language
  - Generally case sensitive

# PHP

- Hello world example
  - <html>

```
<head><title>My Page</title></head>
<body>
    <p> <?php echo 'Hello world!'; ?> </p>
</body>
</html>
```

```
<p> Hello world! </p>
```

# Script Structure

- PHP source file
  - HTML

```
<?php
    // PHP statements
?>
```

  - HTML
- There are also alternative ways…

# Statements

- Statements separator
  - ;
- Blocks of statements
  - { … }

# Comments

- One-line comments
  - // Comment …
  - # Comment …
- Multi-line comments
  - /* Comment … */

# Variables

- Variables
  - **$variable**
- Variable references
  - **&$variable**
    - $a = 0; $b = &$a; $a = 0; // $a = $b = 0
- Variable variables
  - **$$variable**
    - $name = 'var'; $var = 0; $$name; // $$name = 0
    - $map[9] = 'var'; $var = 0; ${$map[9]}; // ${$map[9]} = 0

# Variables

- Global variables
  - **$GLOBALS**
    - Array of global variables
  - **global** $var, …;
- Super global variables
  - $_GET, $_POST, …
  - These predefined variables are always visible

# Constants

- bool **define**(string $name, mixed $value)
  - Defines a named constant with the given value
- bool **defined**(string $name)
  - Checks whether a given named constant exists
- Magic constants
  - __FILE__
    - The full path and filename of the processed file
  - __LINE__
    - The current line number of the processed file

# Strings

- String literals
  - "…"
    - Translates variables and special sequences
      - \\ backslash  \n new line  \t tab  \$ dollar
    - $name = "Peter"; echo "$name"; // Peter
  - '…'
    - $name = "Peter"; echo '$name'; // $name
- Language constructs
  - **echo** $string;
  - **print** $string;

# Arrays

- Arrays
  - Maps associating values to keys
  - Arrays can be multidimensional
- Construction
  - $map = **array**(key => value, …)
    - Keys may only be integers or strings
    - Default keys are integers starting from 0

# Arrays

- Examples
  - $var = array('zero', 1 => 'one', 5 => 'five', 'six');
    - 0 => 'zero', 1 => 'one', 5 => 'five', 6 => 'six'
  - … $var[] = 'seven';
    - …, 7 => 'seven'
  - … $var['eight'] = 'eight';
    - …, 8 => 'eight', 'eight' => 'eight'

# Types

- Data types
  - **Integer**
    - $a = 0b1111; $a = 017; $a = 15; $a = 0xF;
  - **Float**
    - $a = 1.2; $a = 2.3e4;
  - **String**
    - $a = "value"; $a = 'value';
  - **Boolean**
    - true, false

# Types

- Data types
  - **Array**
    - Maps associating keys and values
  - **Object**
    - Objects as instances of classes
  - **Resource**
    - References to external resources (files, …)
  - **NULL**
    - Special type for NULL values

# Operators

- Arithmetic operators
  - + - * / % ++ --
- String operators
  - .
- Assignment operators
  - = += -= *= /= %= .=
- Comparison operators
  - == === != <> !== < <= >= >

# Operators

- Bitwise operators
  - & | ^ ~ << >>
- Logical operators
  - and or xor && || !
- Ternary operator
  - … ? … : …
- Error control operator
  - @…

# Control Structures

- Conditional statements
  - **if** (expression) { … }

    **elseif** (expression) { … }

    **else** { … }

# Control Structures

- Switch construct
  - **switch** (expression) {

```
        case value:
            ...
            break;
        default:
            ...
    }
```

# Control Structures

- Loops
  - **while** (expression) { … }
    - Prints all digits from 0 to 9

      $i = 0; while ($i < 10) { echo $i; $i++; }
  - **do** { … } **while** (expression);
    - Prints 0 and the remaining digits from 1 to 9

      $i = 0; do { echo $i; $i++; } while ($i < 10);
  - **for** (expr1; expr2; expr3) { … }
    - Prints all digits from 0 to 9

      for ($i = 0; $i < 10; $i++) { echo $i; }

# Control Structures

- Loops
  - **foreach** (array_expression as $value) { … }
    - $map = array('zero', 'one', …, 'nine');

      foreach ($map as $value) {

          echo $value.' '; // zero one two …

      }
  - **foreach** (array_expression as $key => $value) { … }
    - foreach ($map as $key => $value) {

          echo "$key=$value "; // 0=zero 1=one 2=two …

      }

# Control Structures

- Loops
  - **break**;
    - Ends execution of the current control structure
  - **break** int;
    - Ends the specified number of nested structures
    - Break 1 behaves like an ordinary break
  - **continue**;
    - Continues the loop with the next iteration

# Control Structures

- Importing other source files
  - **include** $file;
    - Includes and executes the specified file
  - **require** $file;
    - Produces a fatal error upon failure
  - **include_once** $file;
    - Avoids repeated imports of the same file
  - **require_once** $file;

# Functions

- Calling functions
  - $result = function($val1, $val2);
- User-defined functions
  - **function** myFunction($var1, $var2, …) {

    …

    **return** $value;

    }

# Functions

- Default arguments
  - **function** myFunction($var1, $var2 = value) { … }
- Variable number of arguments
  - **function** myFunction() { … }
  - Accessing arguments
    - func_num_args() – number of passed arguments
    - func_get_arg() – returns the specified argument
    - func_get_args() – returns an array of all arguments

# Functions

- Passing variable references
  - `function myFunction(&$var) { … }`
- Variable functions
  - `$function();`
    - `$function = 'myFunction';`

      `$function(); // Calls myFunction()`

# Classes

- Class definition
  - **class** MyClass {

    public $var = 'value';

    myMethod($param) { … **$this**->var; … }

    }
- Object instantiation
  - $instance = **new** Class();
  - $instance->var;
  - $instance->myMethod($value);

# Classes

- Static properties and methods
  - class MyClass {

    public **static** $var = 'value';

    **static** myMethod($param) { … }

    }
- Accessing static items
  - MyClass::myMethod($value);
  - MyClass::$var;
  - **parent**::…, **self**::…

# Classes

- Visibility
  - **public** – unrestricted access
  - **protected** – visible inside the given class and in derived classes using the inheritance
  - **private** – visible only inside the given class
- Default visibility
  - Properties – visibility must be always declared
  - Methods – public

# Classes

- Constants
  - class MyClass {

    **const** NAME = 'value';

    }
  - MyClass::NAME;

# Classes

- Inheritance
  - class SimpleClass { … }

    class DerivedClass **extends** SimpleClass { … }
  - Abstract classes and methods
    - **abstract** class MyClass { … }
    - Prevents class instantiation
  - Final classes and methods
    - **final** class MyClass { … }
    - Prevents class inheritance / method overriding

# Classes

- Special methods
  - Constructors
    - function **__construct**(…) { … }
    - Parent constructors are not (always) called implicitly
  - Destructors
    - function **__destruct**(…) { … }
  - Printing
    - function string **__toString**() { … }

# Classes

- Interfaces
  - **interface** MyInterface {
    function myMethod();
    }
  - All methods must be declared as public
  - Interfaces can be extended like classes
- Usage
  - class MyClass **implements** MyInterface {
    function myMethod() { … }
    }

# Special Variables

- Super global variables
  - $_POST, $_GET, $_COOKIE
  - $_REQUEST
    - Union of $_GET, $_POST and $_COOKIE arrays
  - $_FILES
  - $_SESSION
  - $_SERVER, $_ENV

# Inputs

- $_POST
  - Sample form
    - <form action="index.php" method="post">
      <input type="text" name="param" />
      <input type="text" name="item[0]" />
      </form>
  - Accessing form data
    - **$_POST**['param']
    - **$_POST**['item'][0]

# Inputs

- $_GET
  - URL parameters
    - index.php?param=value&item[0]=123
  - Accessing parameters
    - **$_GET**['param']
    - **$_GET**['item'][0]
  - HTML forms
    - <form action="index.php" method="get">…</form>

# Inputs

- $_FILES
  - Sample form
    - <form action="index.php" method="post" enctype="multipart/form-data">

      <input type="file" name="param" />

      </form>
  - Accessing files
    - **$_FILES**['param'] = array('name' => …, 'type' => …, 'tmp_name' => …, 'error' => 0, 'size' => …)

# Cookies

- $_COOKIE
  - Setting cookies
    - bool **setcookie**(string $name, string $value)
    - All cookies must be sent before any output is generated
  - Accessing cookies
    - **$_COOKIE**['name']

# Sessions

- $_SESSION
  - Managing sessions
    - bool **session_start**() – initializes session data
  - Accessing items
    - **$_SESSION**['item'] = 'value';

# Special Functions

- void **exit**(), void **exit**(int $status)
  - Terminates the current script execution
    - Value 0 is used as the successful termination
- void **die**()
  - Equivalent to exit construct
- bool **isset**(mixed $var, …)
  - Determines if all variables are set and not NULL
- void **unset**(mixed $var, …)
  - Destroys all specified variables

# Array Functions

- int **count**(mixed $var)
  - Counts all elements in the given array
- array **array_values**(array $input)
  - Returns all values of the given array
- array **array_keys**(array $input)
  - Returns all keys of the given array
- bool **in_array**(mixed $needle, array $haystack)
  - Checks whether a value exists in the given array

# Array Functions

- bool **sort**(array &$array, int $flags)
  - Sorts values of the given array
  - Always assigns new keys
- bool **asort**(array &$array, int $flags)
  - Sorts values of the given array
  - Preserves the original string keys

# String Functions

- void **echo**(string $str1, …)
  - Outputs one or more strings
- int **print**(string $str)
  - Outputs a string
- int **strlen**(string $str)
  - Returns the length of the given string
- string **sprintf**(string $format, mixed $args, …)
  - Returns a string produced according to the format
    - %d decimal, %f float, %s string

# String Functions

- string **substr**(string $str, int $start, int $length)
  - Returns the specified substring of the string
- mixed **str_replace**(mixed $search, mixed $replace, mixed $subject)
  - Replaces all occurrences of the search string with the replacement string
- string **str_pad**(string $input, int $length, string $string = " ", int $type = STR_PAD_RIGHT)
  - Pads the input string to a certain length

# String Functions

- string **chr**(int $ascii)
  - Returns a string with the specified character
- int **ord**(string $str)
  - Returns the ASCII value of the first character
- string **strtolower**(string $str)
  - Makes the given string lowercase
- string **strtoupper**(string $str)
  - Makes the given string uppercase

# String Functions

- array **explode**(string $delimiter, string $str)
  - Splits the string into substrings by the delimiter
- string **implode**(string $glue, array $pieces)
  - Joins the array elements using the glue string
- string **md5**(string $str)
  - Calculates the MD5 hash of a string
- string **sha1**(string $str)
  - Calculates the SHA1 hash of a string

# String Functions

- string **addslashes**(string $str)
  - Quotes special characters with backslashes
    - ' \'  " \"  \ \\
- string **stripslashes**(string $str)
  - Returns a string with backslashes stripped off
- string **htmlspecialchars**(string $str)
  - Translates special characters to HTML entities
    - & &  " "  ' '  < <  > >

# Regular Expressions

- int **preg_match**(string $pattern, string $input)
  - Performs a regular expression match
- mixed **preg_replace**(mixed $pattern, mixed $replacement, mixed $input)
  - Performs a regular expression search and replace

# Numeric Functions

- int **intval**(mixed $var, int $base = 10)
  - Returns the integer value of the given variable
- int **rand**(int $min, int $max)
  - Generates a random integer between given limits
- float **round**(float $value, int $precision = 0)
  - Returns the value rounded to a specified precision

# Date Functions

- string **date**(string $format, int $time = time())
  - Returns a formatted string with a given time
    - 'Y-m-d' → '2011-11-05'; 'j. n. Y' → '5. 11. 2011'
    - 'H:i:s' → '19:05:30‘
- int **time**()
  - Return the current Unix timestamp
- int **mktime**(int $hour, int $minute, int $second, int $month, int $day, int $year)
  - Returns the Unix timestamp for the given values

# Filesystem Functions

- resource **fopen**(string $name, string $mode)
  - Opens the specified file
    - 'r' read only mode, 'r+' read and write
    - 'w' or 'w+' truncates a file or creates a new one
- bool **fclose**(resource $handle)
  - Closes an open file resource
- int **filesize**(string $filename)
  - Gets the size for the given file

# Filesystem Functions

- string **fread**(resource $handle, int $length)
  - Reads up to specified bytes from the given file
- int **fwrite**(resource $handle, string $string)
  - Writes the contents of the string to the given file

# Other Functions

- void **header**(string $string)
  - Sends a raw HTTP header
    - Headers have to be sent before any output is generated
- bool **mail**(string $recipient, string $subject, string $message, string $headers)
  - Sends an email to the specified recipient
- bool **phpinfo**()
  - Outputs information about PHP configuration

# Conclusion

- Discussed issues
  - Language syntax
    - Variables, types, constructs, operators, classes
  - Function reference
- Additional issues
  - Databases
  - Extensions
  - Frameworks