Systems Programming & Scripting

Lecture 16: PHP Types

PHP Types

- PHP provides support for the following primitive types:
  - boolean
  - integer
  - float
  - string
  - array
  - object
  - resource: reference to an external resource.
  - null

```
<?php
$boolVar = TRUE;
$intVar = 7;
$floatVar = 2.56;
$stringVar = “Hello”
?>
```

Boolean Type

- Boolean can have two values: TRUE or FALSE (case insensitive).
- The following is also considered as a FALSE value: 0, 0.0, “”, “0”, an array with zero elements and NULL.
- Everything else is considered TRUE, e.g. -1

Integer Type

- Only signed integers are supported in PHP.
- Integer size is platform independent. However usual maximum value is two billions (signed).
- Constants PHP_INT_SIZE and PHP_INT_MAX constants hold information about integer PHP representation.

Integer Type (cont'd)

- If an overflow occurs, the integer value will be interpreted as a float.
- (int) or (integer) can be used to cast to an integer value.
  - Usually not required as casting is done automatically.

Floating Point Numbers

- The float type is the same as double.
- float size is platform independent.
  - Common maximum: ~1.8e308 with a precision of roughly 14 decimal.
- For converting to float, the value is first converted to integer and then to float (apart from when converting from string).

string Type

- Simple way to define a string is to use a single quote.
- ‘ can be included by escaping it with a \

```
<?php
Echo ‘This is a test. This is how to include
  a \’. But \n won’t output a new line.’
?>
```

This is a test. This is how to include a ’. But \n won’t output a new line.

- Including the string in a double quote provides support for more escape sequences .

Arrays

- In PHP, an array is an ordered map associating *values* with *keys*
  - Can support many data structures: array, list, hash table, dictionary, stack and queue.
- The array() construct can be used to create an array composed of *key* => *value* pairs.
- Don’t have to be of the same type.
- *key* can be an integer or a string.
- *value* can be any PHP type.

```
<?php
$arrayVar = array(
  “position” => “manager”,
  “firstName” => “peter”,
  “surname” => “john”,
  “age” => 38) ;

echo $arrayVar[“firstName”]; // peter
echo $arrayVar[“age”];       // 38
?>
```

Another Example

```
<?php
 $house = array (
           “food” => array( 1 => “bread”,
                            2 => “vegs”,
                            3 => “fruits”),
           “people” => array( 1,2,3,4,5),
           “rooms” => array(“bedroom”,
                            “livingroom”)
         );
?>
```

Classes & Objects

- PHP supports the object-oriented programming paradigm.
- A class definition contains variables (defined by *var)* and functions (defined by *function*).

Class Example*

```
<?php
  class Cart {
      var $items;// Items in shopping cart

      // Add $num articles of $artnr tocart

      function add_item($artnr, $num) {
          $this->items[$artnr] += $num;
      }
```

* www.php.net

Cont. Class Example*

```
    // Take $num articles of $artnr out of the cart

        function remove_item($artnr, $num) {
            if ($this->items[$artnr] > $num) {
                $this->items[$artnr] -= $num;
                return true;
            } elseif ($this->items[$artnr]
                        == $num) {
                unset($this->items[$artnr]);
                return true;
            } else {
                return false;
            }
        }
    }
?>
```

*www.php.net

Creating objects

- Objects are instances of classes that are created using *new*

```
<?php
  $cart = new Cart();
  $cart->add_item("10", 1);

  $another_cart = new Cart();
  $another_cart->add_item("0815", 3);
  ?>
```

www.php.net

Using the Departmental Server

- A PHP-enabled, departmental web server:

http://www2.macs.hw.ac.uk/

- It reads user files from

<HOME>/public_html

- And displays them under the URL

http://www2.macs.hw.ac.uk/~<USER>/

- Eg: http://www2.macs.hw.ac.uk/~hwloidl/hello.php