PHP Basic & OOPHP

- Concepts & Examples

Magnus White
mpw@ecs.soton.ac.uk

Overview

Aim to:

- PHP basics
- Discuss specific PHP concepts
- Provide examples
- Answer questions

PHP Basics

Some people raised issues in the labs - I want to clarify some of the basics.

- Variables
- Built in PHP Functions
- Functions

Variables

Declare a variable:

```
(string)  $module = ‘This is the COMP2203 module’;

(int)     $number = 2;

(bool)    $show = TRUE;
```

Variables

Declare a variable:

```
(string)  $module = 'This is the COMP2203 module';

(int)     $number = 2;

(bool)    $show = TRUE;
```

Print the variable to the page:

```
echo $module;  result: This is the COMP2203 module

echo $number;  result: 2

echo $show;    result: 1
```

Built in PHP Functions

```php
echo $var;

$array = (‘red’, ‘green’, ‘blue’, 3);
print_r($array);

var_dump ($array);
```

```php
exit;  die;
is_null();
isset();
empty();
foreach();
```

Plus many more, check php.net and W3Schools

Functions

```
function calcTotal(){
    $item1 = 20;
    $item2 = 100;
    $total = $item1 + $item2;
    return $total;
}

echo calcTotal();
Result: 120

echo ‘Total = ‘ . calcTotal();
Result: Total = 120
```

Functions called inside other functions

```
function calcTotal(){
    $item1 = 20;
    $item2 = 100;
    $total = $item1 + $item2;
    return $total;
}

function presentTotal(){
    $total = calcTotal();
    return ‘The total amount is £’ . $total;
}
echo presentTotal();
Result: The total amount is £120
```

Functions - passing in variables

```
function calcTotal($calc1, $calc2){
    $total = $calc1 + $calc2;
    return $total;
}
```

```
function presentTotal(){
    $item1 = 20;
    $item2 = 100;
    $total = calcTotal($item1, $item2);
    return ‘The total amount is £’ . $total;
}

echo presentTotal();
Result: The total amount is £120
```

Variables & Scope

There are different types of variables and they can be accessed from different places

The **Scope** of a variable is the context in which it is defined

This means what has access to that particular variable

Variables & Scope

In the previous function examples, the variables had to be declared inside or passed into the function

Declaring a different state of scope for a variable will allow updated access and restrictions

Variables & Scope

Here the variables are declared as GLOBAL allowing the function access without them having to be declared or passed into the function

Will NOT work

```
$part1 = 'This is ';
$part2 = 'a sentence';

function sentence(){
    $sentence = $part1 . $part2;
    return $sentence;
}

echo sentence();
```

Works correctly

```
$part1 = 'This is ';
$part2 = 'a sentence';

function sentence(){
    global $part1, $part2;
    $sentence = $part1 . $part2;
    return $sentence;
}

echo sentence();
```

Returns: This is a sentence

OOPHP

Object Orientated PHP

Modern way of using PHP

Just like in any other Object Orientated language

Makes code cleaner and avoids duplication

Class & Objects

Class - a blueprint for an object

When you instantiate a class you create an object

When you use the classes methods, you call upon the object you have instanciated

Classes - Example 1

```
class Module {

    public $name;
    public $description;

    public function __construct($n, $d){
        $this->name = $n;
        $this->description = $d;
    }

    public function getInfo(){
        return ‘Module ‘ . $this->name . ‘ is about ‘ . $this-
        >description;
    }
}
```

Instantiate Example 1

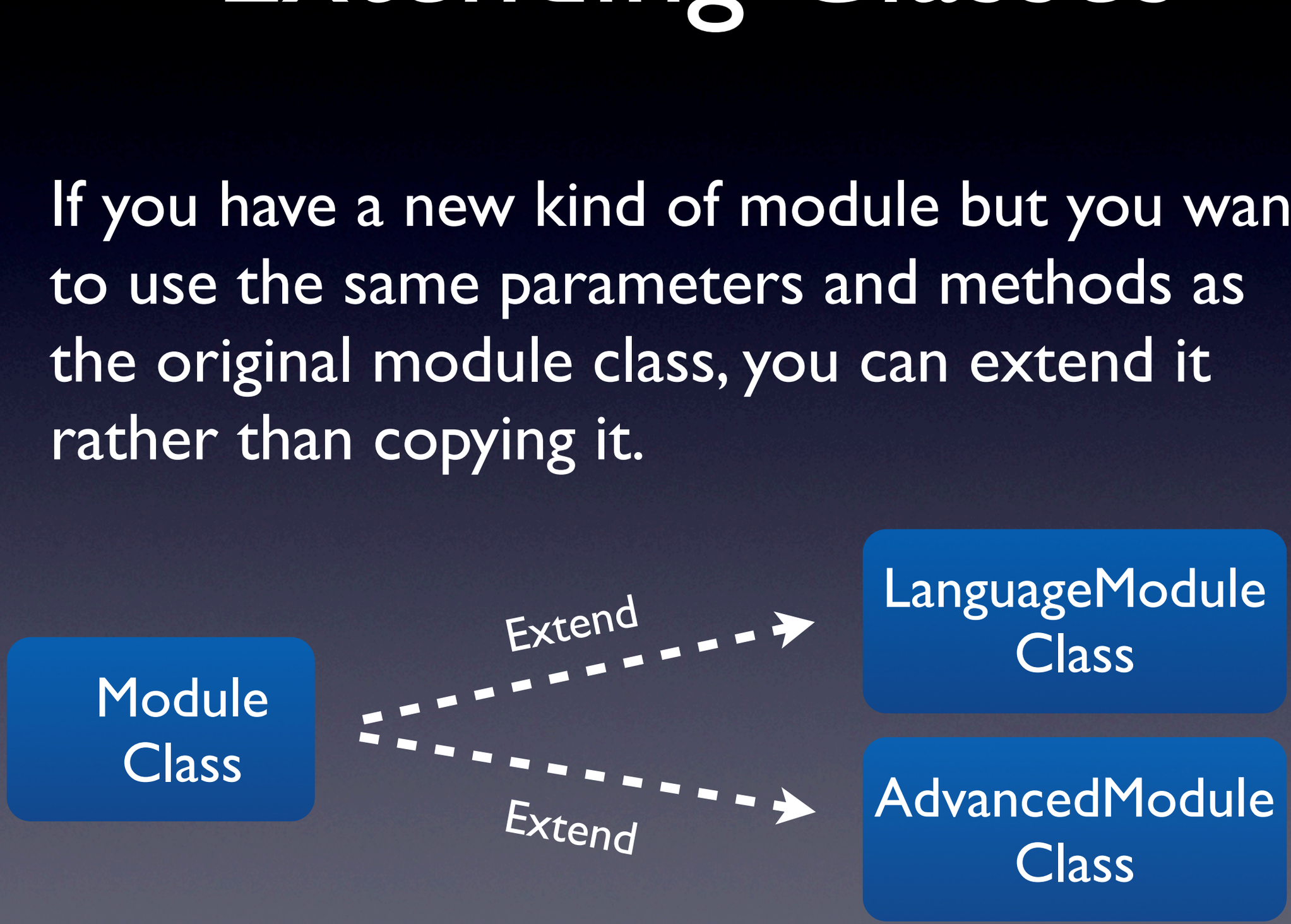
```
// Instantiate Module($name, $description)
$module = new Module('COMP2203', 'application scripting');

// Use getInfo() method
$module->getInfo();

Return: Module COMP2203 is about application scripting

// Print module name - possible as $name is public
return $module->name;

Return: COMP2203
```

Instantiate Example 1 (cont.)

```
// Instantiate Module($name, $description)
$module = new Module('COMP2203', 'application scripting');

// Change the description
$module->description = 'application scripting which has 1
coursework and an exam';

// Use getInfo() method again
$module->getInfo();

Return: Module COMP2203 is about application scripting
which has 1 coursework and an exam
```

Extending Classes

If you have a new kind of module but you want to use the same parameters and methods as the original module class, you can extend it rather than copying it.



Classes - Example 2

```
class LanguageModule extends Module {

    public $lang;

    public function __construct($n, $d, $l = 'English'){
       $this->name = $n;
       $this->description = $d;
       $this->lang = $l;
    }

    public function moduleLanguage(){
       return 'Module ' . $this->name . ' is taught in ' .
       $this->lang;
    }
}
```

Instantiate Example 2

```
// Instantiate LanguageModule($name, $description,
$lang)
$langModule = new LanguageModule('INFO3005',
'application scripting', 'Swedish');

$langModule->getInfo();
Return: Module COMP2203 is about application scripting

$langModule->moduleLanguage();
Return: Module COMP2203 is taught in Swedish
```

A Few Tips

- Google how a function works
- If you copy code from the internet, it is obvious!
- Instead, understand what it does and write it yourself (probably more efficiently)
- In groups, try and break each other's sites - this is invaluable testing

More Information

Use Google!

There are thousands of website with the information you require

The best are:

- www.W3Cschools.com
- www.PHP.net

PHP Coding

- Concepts & Examples

Magnus White
mpw@ecs.soton.ac.uk