CHT2520 Advanced Web Programming

Matthew Mantle m.e.mantle@hud.ac.uk

Today's Session - Introduction to Object-oriented Programming

- Basic OO concepts - class, object, instance, property, method
- Two of 'the four pillars of object-oriented programming'
  - Abstraction
  - Encapsulation
  - We'll do the other two on Thursday

What is an object?

- A data structure
  - Like an array
- More complex
  - A group of related variables (properties)
  - functions for working with these variables (methods)
- We specify the structure of an object using a class...

What is an object?

```php
//Film class specifes the structure of Film objects
class Film {
    public $title; //property
    public $year; //property
    function __construct($title, $year){
        $this->title = $title;
        $this->year = $year;
    }
    function getAge(){ //method
        return date("Y") - $this->year;
    }
}
```

```php
//Create a Film object using the class
$filmObject = new Film("Jaws",1975);
echo "<p>The film {$film->title} is {$film->getAge()} years old.</p>";
```

An Array of Objects

```php
$films = [];
$films[] = new Film("Jaws", 1975);
$films[] = new Film("Winter's Bone", 2010);
$films[] = new Film("Do The Right Thing", 1989);

foreach ($films as $film){
    echo "<p>The film {$film->title} is {$film->getAge()} years old.</p>";
}
```

- We can create multiple instances (objects) from a single class.
- In this example each Film object is stored in an array.

Why OOP? - Abstraction

- One of the four pillars of OOP
- Represent complex things simply
  - Don't have to worry about the underlying details
- Example - Creating a new PDO object.
  - We don't need to know the details of how the database connection is made

```
$conn = new PDO('mysql:host=localhost;dbname=cht2520', 'cht2520', 'letmein');
```

Access control modifiers

- Properties can be public, private or protected.
  - Public. The default, any code can access the property.
  - Private. Only code inside the class definition can access the property.
  - Protected. Only code inside the class or child classes can access the property.

What's wrong with public properties?

```php
class Film {
    public $title; //property
    public $year; //property
    function __construct($title, $year){
        $this->title=$title;
        $this->year=$year;
    }
    ...
}
```

```php
$filmObject = new Film("Jaws","Thriller");
echo "<p>The film {$film->title} was made in {$film->year}.</p>"
```

- The `year` property is public
  I can set it to be a non year value (Thriller)

We can use private properties to restrict access?

```php
class Film {
    public $title; //property
    private $year; //property
    function __construct($title, $year){
        $this->title=$title;
        $this->setYear($year);
    }
    function setYear($year)
    {
        if(!is_int($year) || $year<1895){
            throw  new  InvalidArgumentException ( 'Year must be an integer, at least 1895' ) ;
        }
        $this->year =$year;
    }
```

- The only way to change the `year` property is by calling `setYear()`

Encapsulation (data hiding)

- Using private properties to restrict access is encapsulation, one of the four pillars of OOP
- Objects should control their own state
  - Using private properties and getter and setter methods

Practical Work

- Intro to Object Oriented Programming using PHP
- More theoretical than first two weeks
  - We'll build a useful web app using OOP next week