CHT2520 Advanced Web Programming

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

Today's Session - Re-writing the MVC Example to use OOP

- Based on the Basic MVC examples from Week 2

How would we use OOP when building web apps

- We can broadly categorise objects:
  - Domain (Business) objects
    – Represent things from the real world problem domain e.g. Film, Student.
    - Often create these objects using data from a database
  - 'Other' objects
    – Simply used in our application to help structure and manage code e.g. a Controller class
- We'll consider domain objects later this week.

Why OOP?

- It is clear where the function has come from
- Using functions

```
//Load the models file
require("models/film.php");
//Load the controller file
require("controllers/filmController.php");
index(); //from film.php or filmController.php?
```

- Using OOP

```
use controllers\FilmController;
$filmController =  new FilmController();
$filmController->index();
```

The `index()` method is part of `$filmController`

Why OOP?

- We can take advantage of inheritance

```php
abstract class BaseController
{
        protected function loadView($view, $data = []){
                extract($data);
                require("views/" . $view . ".php");
        }
}
```

```php
class FilmController extends BaseController
{
    public function index(){
        $films = $this->filmModel->all();
        $this->loadView("index.view", ["films" => $films]);
    }
```

- Typically we would have a separate controller for each domain class.

Why OOP?

- A `DbConnect` class.
  - 'Seperation of concerns'
  - Static methods for getting/closing the database connection.
  - Uses the singleton design pattern https://en.wikipedia.org/wiki/Singleton_pattern
    - Ensures only one instances of the database connection can exist.
-

Practical Work

- A simple OO MVC app
- You should make sure you are familiar with basic OO concepts - classes, objects, properties, methods, inheritance, namespacing etc. first.