LOYOLA UNIVERSITY CHICAGO

CENTER FOR TEXTUAL STUDIES AND DIGITAL HUMANITIES

DIGH 402 - Introduction to Digital Humanities Design and Programming

Spring Semester 2014

Week 7

Build your own Class

- PHP class and test script to produce the following output
    - output a user’s username
    - output a user’s firstname and lastname
    - output a user’s age and gender

MVC

Model-View-Controller - Recap Part 1

- defines three parts of an application called model, view, and controller

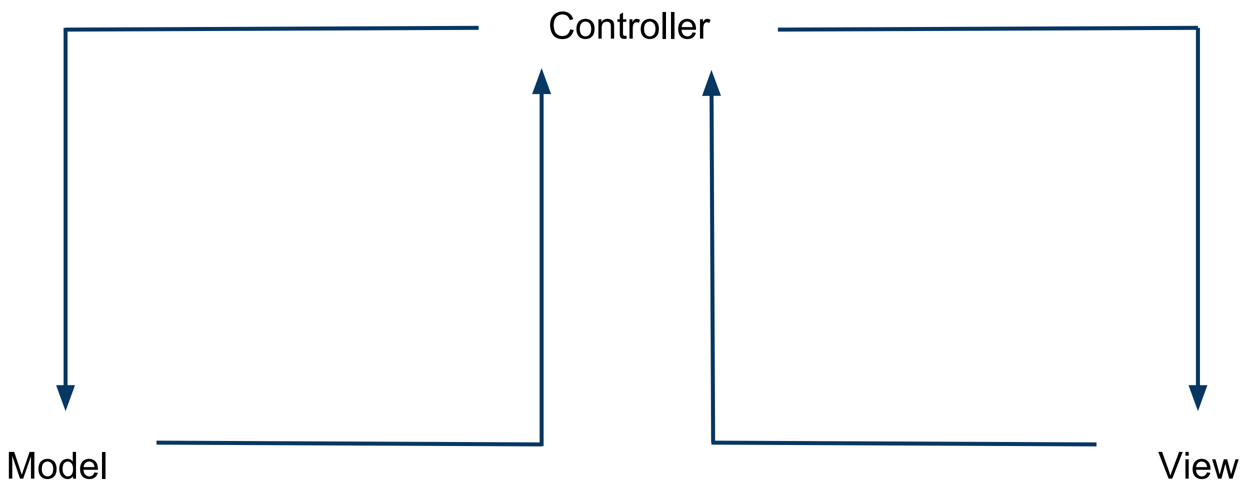
- Model
    - provides the underlying data and methods that offer information to the rest of the app
    - does not define how the app will look or how it will act

- View
    - includes different on-screen UI elements such as buttons, fields, switches...
    - multiple views make up the UI for an app

- Controller
    - manages the interaction and flow between the model and the view
    - handles actions such as user input (keyboard, mouse etc) and sends them to the model or view as required

MVC

Model-View-Controller - Recap Part 2



MVC

Model-View-Controller - Basic Outline for 402 framework

- model
    - connect to the database
    - submit queries to the database (select, insert, update, delete…)
    - update the database…

- controller
    - prepare data for database and view
        - user registration, loading, accounts…
        - load data for content, editing
        - organise data for taxonomy, metadata…
    - load template views for rendering of data
    - security controls and checks
    - load config files for framework
    - load required modules for framework

- view
    - render templates for framework
    - display requested content for users
    - render all UI elements as requested...

Object Oriented Programming

Object Oriented Programming - How to convert our code to OOP

- abstract overview of structure

- classes and inheritance

- what is public, private, protected?

- examples and how it works...

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline

- index.php file (loaded by the web server upon initially opening the home page)

- framework application directory (frame)
    - contains a framework ‘bootstrap’ file
    - directories for files to handle
        - model
        - view
        - controller

- configuration directory (config)
    - config settings for framework
    - any necessary global settings

- system directory (system)

- assets and template (design)

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - index.php file

- we'll use this initial file to setup our framework

- we'll 'require' the frameworks directories, set default paths, set constants

- then load our framework bootstrap loader

    - instantiate an object for the loader class
    - load framework settings
    - initialise the database connection, settings...
    - initialise required sessions for the framework
    - initialise set view theme for the framework
    - initialise the required assets including css, javascript…
    - load controller for the framework
    - load view menus, title…
    - finally render and create the required view for the user

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - directory.php file

- set base framework directory using ‘get current working directory’

- set base folder for framework

- set base folders for framework
    - config
    - design
    - frame
    - system

- set framework system folders

- set framework design folders

- set framework MVC folders

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - bootstrap.php file

- stored in the framework (frame) folder

- called once from the index.php file in the root public directory for our framework and site

- initialises our framework and allows us to control loading of parts of our framework
    - main loader file for framework
    - initialise settings
    - initialise database
    - initialise session (covered later on…)
    - initialise our selected theme for the framework design
    - load framework aspects including menus etc…
    - assign required variables for layout etc…
    - load and output the required layout for our framework

…

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - basic loader.php file

- require our functions.php file for various generic framework functions

- require our constants.php file for framework constants

- require our error_functions.php file for abstracted error handling for framework

- require our controller.php file from our system/library/ to allow loading of our MVC

- ‘Loader’ class to allow us to initialise and load various functions and framework requirements

eg:

  - initialise settings function loaded from bootstrap.php file
  - initialise the database settings and class allowing us to connect to our MySQL database

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - basic settings.php file

- for now we are adding a few basic global settings for our framework

eg:

- setting the title for the framework
- setting the project director…

- we can also set project metadata for the HTML etc

    - keywords, charset, description…

- plus further framework settings such as

    - default language
    - get base URL for project framework

GitHub Code

Object Oriented Programming

Refresher - Using Static Properties and Methods

- we can define class properties and methods that are ‘static’
    - a static method or property can be used without instantiating the object first

- mark as static by putting the ‘static’ keyword after ‘public’ etc

- ‘scope resolution operator’ :: is used to access ‘static’ properties or methods
    - eg: $user = User::get_instance();

- ‘static’ property is a variable that belongs to the class only, not any object
    - isolated from any other property, even another of the same name in an object of this class

- ‘static’ method has no need to access any other part of the class
    - you can’t refer to $this inside a static method (because no object has been created to refer to)

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - basic db.php and config_db files

db.php (Part 1)
- database class for connection and management using PHP’s PDO (PHP data objects) extension. Class contains the following
    - declare various static protected variables
    - setup function for connecting to the database
    - initialise function called to connect to the database within our framework
        - this is called during the bootstrap via the loader class

config_db.php
- create a multi-dimensional array to store connection settings for our framework database
- two arrays including one for development settings and another for production settings

eg:
    - hostname, username, password, database

GitHub Code

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - basic db.php file

db.php (Part 2) - why use PDO instead of mysqli

- more modern extension for connecting to databases through PHP
- PDO has a better interface compared to mysql and mysqli
- PDO has different drivers for different SQL database vendors
- instead of concatenating escaped strings into SQL, PDO binds parameters
    - this is a cleaner and easier way of securing queries
    - allows for performance increase when calling same SQL query with slightly different parameters
- multiple methods for error handling
    - object oriented exception handling
    - consistent style of error handling using PDO

Object Oriented Programming

Object Oriented Programming - Abstract overview of current framework structure

Initial Outline - basic db.php file

db.php (Part 3) - querying the database

- multiple options in PDO for returning result dataset from database
    - use a foreach loop
    - or a while loop
    - or one of the available PDO fetch modes

- PDO also has many built-in options to help fetch results
    - simple fetch()
    - fetchAll() returns an associative array with the field names as keys
    - count rows from query dataset using rowCount()

- we can also use PDO to insert, update or delete records in our database

- it's easy to use PDO statements with parameters

GitHub Code