CptS 483 HW #3

Last updated 2015-03-09. NOTE: This assignment description will probably be updated before the due date of the assignment.

In this homework, you will create your own MVC framework. Included with this document is a zip file containing sample models, views, and controllers. Your assignment is to build a suitable MVC framework that will glue these files together into a single, coherent, web application. You can see a live demo of this code at http://projects.adamcarter.com/simple_mvc. To this end, the following links might aid you in completing this assignment:

- Auto loading files in PHP
- Reflection in PHP
  - PHP documentation
  - Sitepoint tutorial
  - Another tutorial

Basic Rules

While many of the specific implementation details are left to the individual student, your MVC framework must conform to the following standards:

- Your framework must properly render the supplied files in their entirety. Your project will be graded using a separate set of controllers, views, and models. Resist the temptation to hard code values!
  - To make this easier, it might be a good idea to place all of your supporting MVC files into a separate folder.
- You can assume that all controllers will be in the "controllers" folder and that all views exist in a folder of the controller's name inside the "views" controller. For example, all views for the HomeController will exist at /views/home/{view_name}.
- Along these lines, each controller action can be expected to be in the form "{action}Action". For example, the index action function name should be indexAction().
- Note that all controllers have the ability to accept a "params" variable. This array captures additional information from the browser URL. For example:
  - If the URL is /Home/Foo, $params = [] (empty array)
  - If the URL is /Home/Foo/2, $params = [2]
  - If the URL is /Home/Foo/Bar/Baz/Vam/Zin, $params = [Bar, Baz, Vam, Zin]
- Furthermore, action functions should return a View object (implemented by you). At minimum, the View must keep track of the data to send to the view as well as the optional name of the view (see MovieController.php line 20). When a view name is not specified, default to the name of the controller's function. E.g. if we are rendering the index function, we would look for a view called "index.php"

- When rendering the view, place all values returned by the controller action into a variable called "view_bag".
- When a user requests the site root (i.e. no controller name specified), default to rendering the HomeController's indexAction() function
- Inside each view folder, there might be a "layout.php" file. In this case, be sure to render that file.
    - If a view folder does not contain a "layout.php" file, check the view root (i.e. /views) for a "layout.php" file and render it instead.
    - If no such layout.php file exists, render the controller's view directly (e.g. render /views/home/index.php for HomeController->index() function)

Required Functions

The MVC code bundled with this document calls the following functions, which you will need to implement:

mvc_build_url(CONTROLLER, ACTION, PARAMS [optional])

Similar to CodeIgniter's site_url function. This function will return a string containing a URL to the specified controller and action.

mvc_build_style_url(CSS)

This function will return a string containing a URL to the specified CSS file located in /content/style.

mvc_build_script_url(SCRIPT)

This function will return a string containing a URL to the specified JavaScript file located in /content/scripts.

mvc_redirect(CONTROLLER, ACTION, PARAMS [optional])

Redirect's the user's browser to the supplied controller and action.

Deliverables

You must submit your code via Angel by midnight on Friday, April 17, 2015. You must demo your project before the end of the term during my office hours.

Grading Criteria

Your grade will be determined based on how well it meets the above requirements.