COMP2203 Lab 3: PHP and Web APIs

Objectives

By the end of this lab, you will have:

- Forked (copied) an existing Git repository.
- Implemented some basic PHP to reduce code/markup duplication.
- Read live data from a Web API and rendered it into a webpage, using PHP.
- Committed your changes into your Git repository, and documented it in a reasonable way.

Grades

- You must record the preparation points in your logbook for a D.
- To get a C, additionally complete part 1.
- To get a B, additionally complete part 2.
- To get an A, additionally complete part 3.

When recording answers in your log book, write the section and question number. **Write clearly and legibly** – scruffy work will lose you marks.

Your mark will improve if you use the correct tools, and write readable, commented code. We are happy to help you find the right tools and coding style.

Preparation

Before beginning the lab, we recommend you take some time to review PHP's include() function.

You should be familiar with data structures in PHP and JSON, and how data structures are converted from JSON for use in PHP.

Part 1: Implementing Basic PHP

In this part of the lab, you will convert the website to a PHP-driven site, and use the include() function to reduce the amount of duplicated code.

Let's start by checking out the lab repo.

a. Fork the lab repo (www.github.com/COMP2203/lab3) into your Github account, so that you have you own copy to work with. Give the new repo a sensible name.

b. Use the Github Client application on the Lab PCs to clone your newly-forked repo into an empty directory in your ECS webspace called comp2203-lab3. The Clone is your working copy, and will be linked to the master repo at Github.

c. Check that the site is running properly in your webspace, by visiting http://users.ecs.soton.ac.uk/[you]/comp2203-lab3.

Now you're ready to start editing webpages.

d. Change the file extension of the existing webpages from .html to .php, so they will be run through the PHP engine on the server.

e. Create a new directory in the repo called "includes". This is where all your include files will live.

f. Analyse the webpages that make up the site, and convert the `<head>` section of the HTML into an include file.
Implement this functionality in all the webpages.

g. Analyse the webpages that make up the site, and convert the footer into an include file.
Implement this functionality in all the webpages.

h. Convert the menu bar into an include. Note that the menu items are assigned an attribute of `class="active"` to indicate which page the visitor is on. Devise a strategy for setting the `class="active"` attribute on the correct menu item for each page. You will need to use a function to achieve this.
Implement this functionality in all the webpages.

i. Use PHP to set the copyright year on all pages. The date should always be the current year.

Get your work marked by a demonstrator before proceeding to part 2.

Part 2: Reading and Rendering Live Data from a Web API

In this section, you will use PHP functions to read live JSON data from a Web API, interpret the data and render it into a webpage.

We have created a Web API for you to use in this lab. It sources live weather data from the Met Office for a variety of locations.

Reference for our API is at https://github.com/COMP2203/lab3/wiki.

a. Use the API reference to work out the URL that returns weather data for a particular location.
   Test this URL by pasting it into your browser.
   Using your browsers View Source function may make the data easier to check.
   If you successfully get data back, then continue.

b. Use PHP's `file_get_contents()` and `json_decode()` functions to convert the JSON data into a PHP data structure.
   Use `print_r()` to verify that PHP can read the data correctly.

c. Render the data into the Weather Placeholder in index.php. The data should be presented in a human-readable format.
   You can build a basic weather monitor, or a complex one – it's up to you.

d. Finally, push (sync) all your commits up to Github.

Part 3: Addressing Data

In this part of the lab, you will be devising ways of accessing data structures. For each piece of data, write one line of PHP to echo the indicated variable.

Here's an example:

```
$people = [
       1 => "Alice",
       2 => "Bob", ←  echo this variable          Answer: echo $people[2];
       3 => "Charlie"
];
```

```
a. $registrations = [
          "Alice" => true,
          "Bob" => true,
          "Charlie" => false ←  echo this variable
   ];

b. $people = [
        1 => [
          "name" => "Alice",
          "registered" => true
        ],
        2 => [
          "name" => "Bob",
          "registered" => true ←  echo this variable
        ]
   ];

c. $people = [
        1 => [
          "name" => "Alice",
          "modules" => [
            1 => [
              "code" => "COMP2203", ←  echo this variable
              "title" => "Application Scripting"
            ],
            2 => [
              "code" => "COMP2202",
              "title" => "Databases"
            ]
          ]
        ],
        2 => //... etc
   ];
```

Now take your log book to your demonstrator for marking.

Your Github settings should remain on the workstation you used this week. So, if you use the same workstation next week, you may save some time.