Lab Homework 1: Partials

This lab is about putting together the concepts we discussed in lecture to help you assemble your own *programmer's toolkit* for creating dynamic web sites. In this lab you'll learn how we can leverage PHP code embedded within our HTML to generate all sorts of content.

1. Learning Objectives

- Use partials (PHP includes) to reduce redundant code.
- Leverage variables to customize partials.
- Reduce redundant code and improve code readability with PHP user defined functions.
- Learn how to properly cite resources for this class.
- Build your *programmer's toolkit*: Explore how these coding principles open the possibilities of the content that you can construct with PHP.

2. Deadline

Lab Homework	Deadline	Slip Days	Credit	Solution
All Parts	Sun 2/2, 11:59pm	Max: 2 days	20 points (completion)	Provided

1. What you need.

   Your textbook. You'll probably want to use it as a reference to complete this homework.

   If you plan on using your personal laptop to work on your assignments this semester, you should bring it. Otherwise you can use a lab computer. If you plan to use the lab computers, please **bring a USB drive that you can use to save your work**.

2. **Sign the attendance sheet before you leave**.

   It is your responsibility to sign the attendance sheet before you leave. If it wasn't handed to you, **ask to sign it**. No signature on the attendance sheet will result in an immediate 0 for this lab's attendance grade. Forgetting to sign the attendance sheet will not be considered a valid excuse to have your lab attendance corrected.

3. Work together.

   Work with your peers to complete this lab. Talk aloud, discuss, troubleshoot problems. **You are encouraged to work together so long as you do your own work and you don't give away answers.** Every student should have the opportunity to learn this material. If you give away the answers, you're taking away their learning opportunity.

4. Submit.

   When you're finished, follow the instructions in **submit.md** to submit your assignment.

Part I: Welcome!

1. Check Your Section Number

In past semesters, students have made the mistake of attending the wrong lab section.

1. Please go to Student Center and open your enrolled courses.
2. Show this page to the TA to sign the attendance sheet; a TA must see and confirm that you are currently enrolled in the appropriate section.

Note: Other platforms such as CoursePad and Scheduler are not synced with the university and you should not be looking at these sites to check enrollment.

In the case you are not in the right section, please go to the correct lab section now.

2. Ice Breaker!

Objective: **Paper Tower**

1. Form teams of 4-5 students.
2. A TA will hand each team a scratch piece of paper.
3. Each team has exactly five minutes to construct the tallest freestanding structure using no other materials.
4. The winning team is the team with the tallest tower!
5. Now get to know your team in the next section...

3. Introductions

You'll be in this lab section with the same people all semester long. Take a moment to introduce yourself to those around you. I want you to get to know one another so you can help each other out. You might want to exchange contact information too.

Students who have a peer support network in class often do better. **Use this opportunity to build your 2300 support network**: sit in the same location when doing your 2300 homework (just do your own work), ask your peer to fill you in when you miss class, ask your peer to critique your project design!

Remember, there are as many A+'s available in this class as students; everyone can get an A+. Helping your peers will not in any way shape or form hurt your chances at earning an A+. Everyone benefits when we support each other!

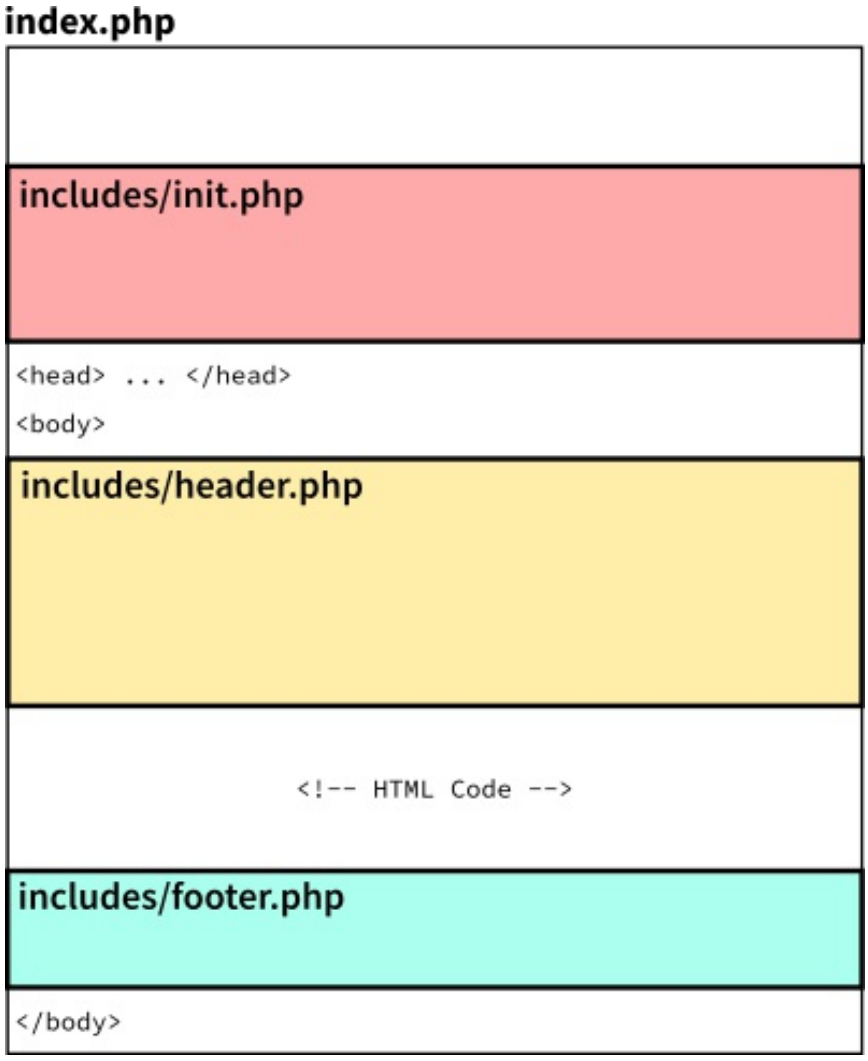
4. Clone your Lab Repository

1. Clone: `git@github.coecis.cornell.edu:info2300-2020sp/YOUR_GITHUB_USERNAME-lab01.git`.
   Replace **YOUR_GITHUB_USERNAME** in the URL with **your GitHub username**.
2. Have the cloned repository open **as a project in VS Code**.

Part II: Reusing Code with Partials

1. Partials (PHP Includes)

Partials save us time by reducing duplicate code. The website in your lab repository currently makes poor use of partials. For example, the structure of **index.php**'s partials should look like what is shown below.



- The **init.php** partial contains only PHP. The file declares variables and functions that will be used in all the pages. It currently checks the PHP version and configures error output. (*Hint:* You can declare variables here if you like.)
- The **header.php** partial *should* contain the code which creates the navigation bar which is included on the top of every web page.
- The **footer.php** partial *should* contain the code that creates the footer that is located on the bottom of every page.

Objective: We've already provided the *init.php* partial. **Complete the partial plan above by creating the *header.php* and *footer.php* partials and use them in every page for this site.**

2. Customizing Partials

The *header* partial should indicate to the user which page they are currently viewing in their browser. It currently looks like this:

Objective: You will change the *header* partial to programmatically indicate the current page like this:

2.1. Planning

Objective: As always, **plan first, and then code**. Write your plan as pseudocode as a comment at the top of *header.php*.

If you need to, you can write comments in PHP like this:

```
/*
  This is a multi-
  line comment in PHP.
*/

// This is a single line comment in PHP.
```

You may not know where to start here. That's normal. Try brainstorming ideas how you might use the tools you have in your programmer's toolbox to solve this problem. Talk to other students and discuss ideas. Don't be too specific here and discuss actual code... you'll still want to solve this yourself that way you learn more. You may also have to make several plans if your first few fail. Again, this is normal.

Hint: You can solve this using only variables (the stuff you've learned in class). You don't need conditional statements, arrays, functions, loops (the stuff you haven't learned in PHP yet) to do this.

2.2. Implementation

Objective: Once you've got your plan, it's now time to code it up! Programmatically identical the current page in the navigation bar.

Note: Undefined variable notices/warnings are okay.

Part III: Reusing Code with Functions

Managing and maintaining the content of static HTML pages can take a lot of time. It's often easier to store the content in some way (later we'll learn how to do this with databases) and then use PHP to place that content in the HTML. In this part, we'll add another tool to your *toolkit*: demonstrate how we can leverage functions and arrays to generate HTML content.

A benefit of using a function is that it shortens code that you would otherwise would have repeatedly displayed. This makes code concise and neat which improves readability. Functions also allow us to reproduce the same action we want while decreasing room for error, as we only need to make changes to arguments (parameters) taken in.

Take a look at the code for the Standards page. Observe that there is an array of the standards for this class.

If we wanted to output these standards to the screen we *could* write some code like this:

```
<li><?php echo htmlspecialchars($standards[0]); ?></li>
<li><?php echo htmlspecialchars($standards[1]); ?></li>
<li><?php echo htmlspecialchars($standards[2]); ?></li>
<li><?php echo htmlspecialchars($standards[3]); ?></li>
<li><?php echo htmlspecialchars($standards[4]); ?></li>
<li><?php echo htmlspecialchars($standards[5]); ?></li>
...
```

The many repeated parts just takes up space. Using a function, we can have the same display while using less than 5 lines of code (in comparison to the 20 lines we have now). Another benefit of a function is that it can help us prevent mistakes that commonly are made when just copying and writing several lines of code.

1. Plan

Here is a plan for how we might remove the repeated lines of code.

At the top of **standards.php**:

```
function print_standard( standard )

  prints out standard enclosed in a <li> tag

end function
```

In the HTML body:

```
for each ( standard in standards )

  call print_standard(standard)

end for each loop
```

2. Code it!

We've already created the `print_standard()` function declaration for you. We've also coded up the code for the the HTML body.

Objective: You just need to complete the inside of the `print_standard()` function.

You might feel lost here. That's normal. Take it slow and think through things. Discuss at a high level with your peers if you need to. Work together and support one another!

Part IV: Citations

If you use resources created by other people, you need to cite that resource. Not citing that resource is *plagiarism* and is considered academic misconduct.

You are required to cite *all* resources according the course citation policy. Failure to cite **all resources** according to this policy will result in a minimum of a one letter grade deduction on an assignment.

Objective: Visit the course website and read the course citation policy now. (Yes, you need to read it again if you took 1300, it's changed!)

1. Citation Rules

Objective: Create a new citations.php page and add it to the nav bar.

Objective: Your citation page should include the two *main* citation rules for the class:

1. Every resource that you use that you did **NOT** create (external resource) must have a citation visible on the screen near the resource **AND** must include a citation to the resource in the code as a comment.
2. Every resource that you use that you created (**non**-external resource) must have a citation in the code as a comment.

Hint: The bolded text here has semantic meaning. You probably want to use `<strong>` or `<em>`.

2. Citation Examples

Objective: Provide an example for each citation rule on this page.

1. For the first rule, find a resource from somewhere on the internet.
2. For the second rule create your own resource. (*Hint:* take a selfie.)

Part V: Validation

Objective: Your HTML and CSS must validate. Check that each page validates before you submit your assignment.

Hint: Validate HTML by viewing the PHP page in the browser, then view the HTML source in the browser. Copy the source and paste it into the HTML validator: https://validator.w3.org/nu/#textarea

Contributors

The following individuals made contributions to this assignment.

- Kyle Harms
- Sharon Jeong