CS 328 - Week 9 Lab Exercise - 2025-03-27/28

Deadline

Due by the end of lab.

Purpose

To practice more with CSS flexbox and/or grid layout, and to practice creating a PHP postback-style document that can create a form and obtain data from that form when it is submitted (and also practice a little with a server-side include along the way).

How to submit

Submit your files using `~st10/328submit` on nrs-projects, each time entering a lab number of **89**.

Requirements

- You are required to work in **pairs** for this lab exercise.
  - This means **two** people working at **ONE** computer, one typing ("driving"), one saying what to type ("navigating"),

    while **BOTH** are looking at the **shared** computer screen and **discussing** concepts/issues along the way.
- Make sure **BOTH** of your names appear in each file submitted.
- When you are done, before you leave lab, **BOTH** of you should submit appropriate versions of these files using **`~st10/328submit`** on nrs-projects, with a lab number of **89**.
- You are expected to follow the style standards from the posted "CS 328 CSS Coding Standards so far" (at https://nrs-projects.humboldt.edu/~st10/s25cs328/328-css-coding-standards.pdf).
- You are expected to follow the style standards from the posted "CS 328 PHP Coding Standards so far" (at https://nrs-projects.humboldt.edu/~st10/s25cs328/328-php-coding-standards.pdf).

Start your file `328lab09.php`

- Use the course HTML template to start up a PHP document **`328lab09.php`**:

  `cp ~st10/html-template.html 328lab09.php`
- Fill in the opening comment block, putting in your **names**, the last modified **date**, and the **URL** that can be used to run your document.
  - (You *will* lose some credit if this URL does not work when I or the grader paste it into a browser!)
- **Within** the **`head`** element, edit the **`title`** element, giving it appropriate content.
- **Within** the **`head`** element, after the `link` element specifying that this document is being styled by `normalize.css`, add a second **`link`** element following course style standards specifying that this document will then be further styled using a file **`lab9.css`** that you are about to create for this lab exercise.
  - **Do not include any inline or internal CSS rules in your `328lab09.php`.**

- Within the **`body`** element, include an **`h1`** element with appropriate content.
- **Within** the **`footer`** element near the end of the **`body`** element, add a **`p`** element whose content includes your names.

You are now going to add to this so that **`328lab09.php`** will be a postback document, so that it **both** creates a form **and** crafts a response to that form when it is submitted (as demonstrated in the posted Week 9 Lecture 2 example **`328lect09-2.php`**).

What form?

The form your PHP is to create when it is initially called should be the **`form`** element in the provided file **`328lab09-form.txt`**. (Why did I give this file a suffix **`.txt`**? To help stress that it is not an entire HTML document -- this file contains JUST a form element !)

- Copy this file into the same directory as your file `328lab09.php`:

  ```
  cp ~st10/328lab09-form.txt .  # DON'T FORGET the SPACE and PERIOD!!
  ```

- Read over the **`form`** element in this file; you'll be writing CSS rules to style it in **`lab9.css`** (see below).
- To try out one of PHP's functions that allow for so-called server-side includes, you should include the contents of this file into your **`328lab09.php`** **NOT** by hand-pasting them in, but instead by **calling** the following function in the appropriate part of the **`if`** statement you will be adding to your **`328lab09.php`**:

  ```
  require_once("328lab09-form.txt");
  ```

  - This can go in a regular PHP tag, and has the side-effect of including the contents of the specified file in the resulting PHP response.
  - (Fun facts: this is supposed to throw a fatal error if it cannot find the file, and if you accidentally were to try to use this to include the same file more than once within a PHP, this is supposed to make sure that it would only include its contents once.)

What goes in `lab9.css`?

Your **`lab9.css`** should meet the following requirements:

- Include a comment including at least your **file's** name, *your* **names**, and the last-modified date.
- If you change any of the default foreground and background colors, make sure the result is **at least WCAG 2 AA Compliant** based on the tester at: https://snook.ca/technical/colour_contrast/colour.html
- Include rule(s) adding visible, attractive **borders** to at least **`328lab09-form.txt`**'s:
  - **`form`** element,
  - its **`div`** with **`id="ice_cream_choice"`**
  - its **`div`** with **`id="more_prefs"`**
- Include rule(s) that will result in this **form** ***not*** taking up the entire width of the body, and being **centered** within the body.
- Include rule(s) that will result in the **submit button** being **centered** within the form.
- Add rule(s) using either **CSS flexbox layout** or **CSS grid layout**, your choice, to noticeably layout the elements within the **`div`** with **`id="ice_cream_choice"`**.

- Add rule(s) using either **CSS flexbox layout** or **CSS grid layout**, your choice, to noticeably layout the elements within the **`div`** with **`id="more_prefs"`**.
- Add additional rules *if* desired to further attractively format elements of this document.
- Make sure your resulting **`lab9.css`** validates as valid CSS.

What form response?

When its form is submitted using `method="POST"`, your PHP should craft a response that *cautiously* displays, within a **`ul`** element, the values submitted from that form.

- REMEMBER! **Never trust user data!** Appropriately use **`htmlspecialchars`** and/or **`strip_tags`** so that you don't unwittingly execute user-injected inappropriate code or other badness.
- IN ADDITION: function **`trim`** expects a string, and removes and leading or trailing blanks from that string, and returns the resulting string.
  - This is also useful for more pragmatic sanitizing -- removing and accidental leading or trailing blanks that users probably-accidentally entered into, for example, input fields and textareas is a good practice, as well.
- FUN FACT: I don't *think* you'll need it for handling this form, but you can see IF an array KEY exists in an associative array using:

  ```
  array_key_exists($desired_key, $desired_array)
  ```

- FUN FACT: I don't *think* you'll need it for handling this form, but you can see IF a variable or array reference has a value set for it using:

  ```
  isset($var_or_array_ref)
  ```

How can one strict-validate the HTML resulting from a PHP?

Beware -- if you put the URL of this PHP directly into the validator at https://html5.validator.nu/, it looks like it does regular-HTML validation (not strict-validation) of just the response containing its form. This is not a bad start, but it is not strict-validation, nor does it check your PHP's response with a submitted form's values.

So, do the following to strict-validate your PHP's resulting HTML:

- Put your **`328lab09.php`**'s URL in a browser and view its source, copy and paste that source into a file named **`328lab09-1.xhtml`**, and put the URL of your **`328lab09-1.xhtml`** into the validator.
- Put your **`328lab09.php`**'s URL in a browser **and fill out and submit its form**, *then* view that *response*'s source, and copy and paste that *response*'s source into a file named **`328lab09-2.xhtml`**, and put the URL of your **`328lab09-2.xhtml`** into the validator.

BEFORE you leave lab:

Make sure that you **both** have copies of the files:

- **`328lab09.php`**
- **`328lab09-form.txt`** (since it may be included in **`328lab09.php`**'s response using **`require_once`**)
- **`lab9.css`**
- **`328lab09-1.xhtml`**, **`328lab09-2.xhtml`**

...and you BOTH submit these using **`~st10/328submit`** on nrs-projects, with a lab number of **89**.