CSC 337, Fall 2013
Assignment 5
Due: Wednesday, October 23 at 22:00:00

In this assignment you'll be writing some small PHP programs and continuing to make use of your HTML and CSS skills.

You'll be working with PHP integers and strings, using control structures, writing some simple functions, and doing a little input from files. At first glance you might think that you'll need PHP's arrays for at least part of this assignment, but you really don't. To encourage you to think more flexibly about what can be done with strings and to avoid headaches with arrays before we've studied them, **there's an assignment-wide restriction: you may not use PHP arrays for this assignment**.

Until we talk about how to secure your `/cs/cgi/people/NETID/public_html` directory, DO NOT use it for testing. Instead, test locally on your own machine. If testing on your own machine is not an option for you, let us know; and we'll get you set up with a secured directory on lectura ASAP.

Problem 1. (8 points) `toss.php`

PHP's `int rand($min, $max)` function generates a random integer between `$min` and `$max`, inclusive. Coin tosses can be simulated with `rand(0,1)`—if it returns 0, that's heads; if 1, that's tails.

In this problem you are to write a PHP program that writes HTML and CSS **on standard output** that shows a visualization of the result of 10,000 coin tosses. Here's the visualization for a sequence of twenty tosses:

The above represents this sequence of tosses: THTTTHTTHHHTTTTHHTTH. That is, there was a tails, a heads, three tails in a row, a heads, two tails in a row, and so on. N-in-a-row heads or tails is shown with `font-size: Nem`. Heads are shown as black text on white; tails are the opposite. Note the rounded corners on both and the one-pixel border on the heads. Output a `<wbr>` after each count to allow line wrap after each count.

Here's a function I used when developing my solution:

```
function toss()
{
    return rand(0,1);

    static $last = 0;

    $last = !$last;
    return $last;
}
```

If the initial return is commented out, the function returns 1, 0, 1, 0, ... on successive calls. The `static`

keyword causes the value of `$last` to be saved across function calls; this language feature comes from C. Java doesn't have a method-level counterpart for it.

We'll run your program like this,

```
% php toss.php > x.html
```

using the command-line output redirection operator (**>**) to capture `toss.php`'s standard output in a file. In the case above that file is `x.html`. We'll then open that file with a browser.

Problem 2. (6 points) `pattern.php`

Assignment 1's problem 1, `pattern.html`, asked you to create an HTML file that produced a particular pattern and suggested that you do that by writing a program. Most of you did it by writing a Java or Python program. For this problem you are to write a PHP program that produces HTML whose rendering produces that pattern. It's probably a simple matter to transliterate your Java or Python code into PHP, and that's fine. If you already knew PHP and wrote `pattern.php` for Assignment 1, just turn it in!

`$a5/pattern.html` is what my `pattern.php` produces. Feel free to look at the HTML and browse it with Chrome DevTools.

Problem 3. (12 points) `create.php`

`toss.php` and `pattern.php` are simple programs that produce interesting graphical output by using a large amount of repetition. For this problem you are to cook up something of the same sort on your own.

You might already have some ideas, but here are some elements that you might use:

- random numbers
- absolute positioning with CSS
- `rgba(...)` values, taking advantage of overlapping background colors
- various CSS sizing properties

If you want to bring in some CSS 3 things that we didn't talk about, that's fine, too. Maybe do some stuff with `transform`, to get element boxes drawn at non-right angles.

The key characteristic your `create.php` must have is the use of repetition to produce an interesting visual effect; it should generate body of HTML (with CSS) that wouldn't be practical to create by hand, which is true for both `toss.php` and `pattern.php`.

If you're short for ideas, come and see us! As usual on problems where you're asked to do something original, I'll be happy to give you feedback on possibilities you have in mind.

We'll run your program like this:

```
% create.php > create.html
```

Remember that `echo` writes to standard output; so if you generate your HTML and CSS with `echo`, the program invocation above should work just fine. If your program uses a data file, be sure to include it in your `a5.zip`.

Problem 4. (26 points) `blog.php`

For this problem you are to write a PHP program that writes HTML and CSS for a single-page blog that's based on the contents of a text file, `blog.txt`, and that essentially matches the styling shown in `$a5/blog*.png`.

`blog.txt` contains blocks of lines, with each block representing a single blog entry. Here's an example of the block of lines for one entry:

```
New pictures!
08/28/13
Donec et convallis tellus.
.i Mauris a sem erat.
.b Etiam in facilisis turpis, sit amet placerat sapien.
.li raffle.jpg
Class aptent taciti sociosqu ad litora torquent per conubia
nostra, per inceptos himenaeos.
.ri folly.jpg
Nullam tincidunt metus justo, ornare luctus enim aliquam
pellentesque. Suspendisse ante nisl, imperdiet non tortor et,
mollis lacinia nisl.
.p
.u Donec luctus porta dolor, blandit tincidunt arcu feugiat
posuere.
In tempus, lorem non facilisis vehicula, nisi nisl sodales erat,
at suscipit massa mauris vitae purus. Donec sit amet justo sed
ligula placerat congue. Cras a justo dignissim, suscipit tortor
nec, varius dolor. In egestas erat ac augue facilisis iaculis.
.end
```

The first and second lines are the title and date of the entry, respectively. Characters in the title bounce upwards slightly when they are moused over! Titles are shown with `font-family: cursive`. Your browser's `cursive` might be different from mine; that's fine. Note that the dashed border curving around the bottom and right of the title extends to the left and top edges of the `antiquewhite` area.

Note that each entry's date is displayed on a tab that extends to the right. Don't bother to examine whether dates of entries are in ascending order. Note that a date like `08/26/13` is transformed into aug 26 2013, split across two lines. Don't make that complicated! Assume that dates are always eight characters long: `MM/DD/YY`. Here's a line of code from my solution that helps with the dates:

```
$months = "JanFebMarAprMayJunJulAugSepOctNovDec";
```

The third and following lines of a block are the text of the entry. A line containing only "`.end`" indicates the end of the entry. Along with plain text to display as-is there is markup with various directives, each indicated by a dot at the start of a line (e.g., `$line[0] == "."`). Here are the directives:

`.i .b .u` The text that follows on this line is to be displayed in italics, bold, or underlined, respectively. Assume that for these the third character is always a blank and that the text to display in the specified form starts with the fourth character.

Note that there is no way to apply multiple styles, like bold and italic, to a piece of text.

`.p` Start a new paragraph by generating "`<p>`".

`.li .ri` The text beginning at the fifth character is the name of an image to be displayed using a left or right float. Assume the image has a 4:3 aspect ratio. Display the image 100 pixels wide with a one-pixel border. If the user hovers over the inline image, it should enlarge to be 450 pixels wide, have a border with rounded corners, be slightly below where it was and be roughly horizontally centered with respect to the space occupied by the article. It's OK if there's some flickering when the cursor is in certain positions over the small image. If a picture near the end of an entry is enlarged, it may extend below the entry itself—that's OK, too.

Use a `clear` to ensure that images always run down the sides and are never adjacent to each other.

Refer to the ***Error handling and implicit limits*** section starting on page 8 of the Assignment 1 write-up to remind yourself of the way I'd like you to think about those topics. Just as that section states, you should assume that `blog.txt` is well-formed, i.e., error-free. Here are some examples of **malformed** lines for which *behavior is undefined*:

```
.ri
.lix.jpg
.i .b .u you be me
.endd
```

Your program should be able to handle any number of blog entries. If you try to read the title line for the next entry and `fgets` returns `false`, then it's time to wrap-up and exit. Assume that `blog.txt` has at least one entry.

Note that the titles in the My Posts box are links to the entries and are in reverse order—the first link is for the last blog entry; the last link is for the first blog entry. Links in the My Posts box are always in black, not underlined, and show no evidence of whether they've been visited or not. Blog entries have a "top" link in their lower right hand corner that takes you to the top of the page. You might first think you need an array to handle that reversal of ordering, but you do not; your brain and a string is all you need! (In fact, I claim that's sufficient for *any* computation!)

Roughly 80% of your score on this problem will be based on a test using `$a5/blog.txt`; the remaining points will be based on results with one or more other versions of `blog.txt`.

Your solution should use `fopen("blog.txt", "r")` to open the input file. (In other words, the program should assume that `blog.txt` is in the current directory.)

Here's a link for a video demo of the interactive features: `http://youtu.be/rzpiMVVFp4I` It has some significant audio dropouts/skips; I'll redo it if time permits, but I believe it's adequate to see the bouncing of characters in entry titles and the enlargement of images when hovering over them.

You'll see that `$a5/blog1.png` doesn't show the full text corresponding to `$a5/blog.txt`. `$a5/blog2.png` shows the whole thing but zoomed way out.

Hints: My solution has about 80 lines of PHP code, not counting comments and a multi-line `echo` (like on slide 41) that outputs some boilerplate and an embedded stylesheet. If your solution starts to grow significantly longer than that, you might be making things too hard or making poor use of PHP's capabilities. (Come and see us!) A CSS note: my solution does not use a float for the My Posts box.

Problem 5. (5 points EXTRA CREDIT) `ascii.php`

`$a5/ascii.png` is a screenshot of part of the output of "`man ascii`". `ascii.php` reads lines on standard input and writes HTML on standard output that when rendered shows the text of the lines read on standard input. That sounds simple enough but here's the twist: **the HTML does not contain the characters being shown**. Instead, you are to use the technique shown on slides 152 and following, with `ascii.png` as the `background-image` for a sequence of spans, each of which represents a single character of the input. I didn't mention the term in lecture but things like `nav_logo161.png` as shown on slide 153 are often called a *sprite sheet* or *CSS sprites*. For this problem `ascii.png` is your sprite sheet!

Assume the input only contains characters with ASCII codes 32-126 inclusive. Output a `<br>` in response to newlines.

Here's one way to see if you understand the problem: If the input is only the word `Zebra`, the HTML that's output would not contain a "Z", a "z", or a character reference to a "z". Instead it would contain five `span`s that reference `ascii.png` as their `background-image` and use `background-position` to show the appropriate glyph.

I haven't yet implemented a solution for this problem myself so I've got no sample result to show you.

Problem 6. Extra Credit `observations.html`

Submit a styled-as-you-like HTML file named `observations.html` with...

(a) (1 point extra credit) An estimate of how long it took you to complete this assignment. To facilitate programmatic extraction of the hours from all submissions, please enclose the number of hours in a `span` with `class=hours`, like this:

```
I spent <span class=hours>8.5</span> hours on this assignment.
```

Other comments about the assignment are welcome, too. Was it too long, too hard, too detailed? Speak up! I appreciate all feedback, favorable or not.

(b) (1-3 points extra credit) Cite an interesting course-related observation (or observations) that you made while working on the assignment. The observation should have at least a little bit of depth. Think of me saying "Good!" as one point, "Interesting!" as two points, and "Wow!" as three points. I'm looking for quality, not quantity.

Aside from my stretch last fall substituting for Dr. McCann in 352 it had been several years since I'd taught, and I'd forgotten how much I valued the feedback I'd gathered with this same question on assignments in years past. Therefore, I'm giving you a chance to go back in time and repeat (a) and (b) above for each of assignments `a1` through `a4`. If you do this for any of the previous assignments, make it clear which assignment(s) you're writing about.

Turning in your work

Use the D2L Dropbox named `a5` to submit a zip file named `a5.zip` that contains all your work. If you submit more than one `a5.zip`, we'll grade your final submission. Here's the full list of deliverables:

`toss.php`
`pattern.php`
`create.php` **and** any data files that it uses

`blog.php`
`ascii.php` (for extra credit)
`observations.html` (for extra credit)

Note that all characters in the file names are lowercase.

Have all the deliverables in the uppermost level of the zip. It's ok if your zip includes other files, too.

Miscellaneous

Remember the assignment-wide restriction: you may not use PHP arrays for this assignment! It is true that a PHP string behaves a lot like an array of characters, but there are no restrictions on the use of strings. (Note: That's **not** a hint to use the array-like behaviors of strings as a workaround for the no-arrays restriction!)

The HTML slides, the CSS slides, PHP slides covered through Oct 16, and in-class examples should provide all you need to complete this assignment. Not considering `create.php`, if you find yourself wanting to use aspects of CSS or PHP we haven't covered, you're probably overlooking something and/or making a problem more difficult than intended.

Point values of problems correspond directly to assignment points in the syllabus. In total this assignment represents 5.2% of your final grade in this course.

`$a5` follows the convention of `$a`*`N`* on the previous assignments.

Remember that late assignments are not accepted and that there are no late days; but if circumstances beyond your control interfere with your work on this assignment, there may be grounds for an extension. See the syllabus for details.

My estimate is that it will take a typical CS junior from five to nine hours to complete this assignment. If you're a CS senior, you'll probably do it in less. If you've taken only a single programming class, you might be on the high end or beyond.

Keep in mind the point value of each problem; don't invest an inordinate amount of time in a problem or become incredibly frustrated before you ask for a hint or help. Remember that the purpose of the assignments is to build understanding of the course material by applying it to solve problems. If you reach the seven-hour mark, regardless of whether you have specific questions, it's probably time to touch base with us. Give us a chance to speed you up! **Our goal is that everybody gets 100% on this assignment AND gets it done in an amount of time that is reasonable for them.**

None of your results need to validate with either the HTML5 validator or the "Jigsaw" CSS validator, but you may find the CSS validator to be helpful in diagnosing mysterious problems with CSS. (The typical mysterious CSS problem is that a declaration is ineffective, like those on slide CSS slide 16.)

Remember that I award Bug Bounty points; if you find problems, there's a potential reward for reporting them. I prefer that bugs be reported by mail to me. I'll put up a " FAQs and Corrections" post on Piazza and update it as things come in.

Use the `a5` folder for any Piazza posts about the assignment.