User Interfaces – Lab # 3 **Version 2**

420-2W6-AB

Worth:	This lab will count towards the 'lab' portion of your final grade at 12.5%
Due:	March 19, 2022 (by 'end of day')
Hand In:	A .zip file containing your work onto LEA in the appropriate place
Late Penalty:	Late submissions lose 2.5% per day to a maximum of 7 days. Nothing is accepted after 7 days and a grade of zero will be given.

"Your submitted work must be clear, complete, and YOUR OWN. You must be prepared to explain any of your work to me in person. Failure to be able to defend your work, or do a similar question in front of me in person can/will void any grade you get on this lab."

Goals:

- Develop and practice intermediate CSS layout skills using Floats and Flexbox
- Translate static design documents into a living website.

Contents

User Interfaces – Lab # 3 **Version 2**

420-2W6-AB

Part 1: Exercises (10%)

If you've been keeping up with in-class exercises over the past few weeks, you've already completed Part 1 of this homework. If not, **you should start here** before attempting Part 2 – the skills practiced in these exercises are going to be necessary to complete Part 2.

For each of the following, you should attempt all of the problems – for full marks, you should complete at least **75%** of them (i.e., you can leave one or two incomplete for each exercise set). I**nclude your solutions as separate subfolders in your submission .zip**. (that is, create a folder for each exercise set, and copy your solutions to those exercises into the corresponding folder – have all three of those folders be inside of the folder that you compress to a .zip and submit to me)

- Week 4: https://michaelhaaf.github.io/2W6-W23/lectures/week4.html#exercises
  - Odin Project: CSS Foundations Practice Exercises (.zip)
  - MDN: Typesetting a Homepage (instructions) (starter files .zip)
  - Odin Project: CSS Box Model Practice Exercises (.zip)

- Week 5: https://michaelhaaf.github.io/2W6-W23/lectures/week5.html#exercises
  - Float exercises: **Complete at least 3 of 4**. Some of the links on my website were broken, I have now fixed them.

- Week 6: https://michaelhaaf.github.io/2W6-W23/lectures/week6.html#exercises
  - Odin Project: CSS Flexbox Practice Exercises (.zip)

Part 2: Create a Landing Page (80%)[1]

For this assignment, you will create an entire web page from a design we've provided for you.

The design is specified in a set of image files:

- an image of the completed website
- an image detailing the fonts and colors required
- some assets (placeholder image, icons) have been provided for you as well.

Get your project as close as you can to the design, but do not worry about getting it pixel-perfect (it is often the case that designs provided to you in real life have mistakes, imperfect measurements, etc.). Don't get out your ruler or count pixels to find the exact margins between the various sections.

Once your layout is done, **you must substitute your own content into this design**. This website should be a continuation in the series of website about animals you have created – you should insert real images for the placeholders, write actual content about the animals, etc.

One more thing:

- The "LOGO" text in the top left corner of the webpage should be an .svg image as well. You should find an svg that seems representative of the content of your webpage (so if this is a page about Lions, you should find a nice .svg that's a graphic of a Lion, or at least a large cat)

A note about images on the web

You do not have the legal right to use just any image that you find on the web. There are many free images to be found, but make sure that the image you use is actually free for you to use, and make sure to credit the creator of the image in your project. Some good places to find free-to-use images on the web include Pexels, Pixabay, and Unsplash. To credit the image, **include the url of the image as a comment in your html.** It's actually a problem with the Design Specifications that I shared with you that there is no room for attributing the authorship correctly – we will return to this problem in Lab 4.

[1] Assignment description (not content) adapted from https://www.theodinproject.com/lessons/foundations-landing-page -- even though the design and the requirements of this project are different than ours, the resources provided for this project should be useful to you as well.

Hints

- There are likely small details about this assignment that we have not have encountered in our class. *This is by design.* These details are minor, and easily searched (e.g. google `css rounded corners`, or better yet, search the topic on the MDN developer docs/w3schools/css-tricks.com/internetingishard.com/other resources I've provided in class. **Real-life, professional developers use external resources *constantly* for things that they have been doing for years.** At this point it is not expected that you will have everything in this course memorized.

- There are many ways to tackle a project like this, and it can be overwhelming to look at a blank HTML document and not know where to start. Suggestion: take it one section at a time. The website you're creating has 4 main sections (and a header+footer), so pick one and get it into pretty good shape before moving on. Starting at the top is always a solid plan.

- For the section you're working on, begin by getting all the content onto the page before beginning to style it. In other words, do the HTML and *then* do the CSS. You'll probably have to go back to the HTML once you start styling, but bouncing back and forth from the beginning will take more time and may cause more frustration. (Note: you don't need to use more than one stylesheet. Using only one CSS file is adequate for this project).

- **Do not worry about making your project look nice on a mobile device, or even at different zoom levels. We'll learn how to deal with that later.**
- HINT: The design specification has a defined width of **1440px.** Your website should also be 1440px wide. Don't worry about how it behaves at other viewport widths or zoom-levels.

Part 3: Alternative stylesheets for troubleshooting (10%)

When web designers provide web developers with template designs like we have provided for you in Part 2 of this assignment, there is *always* back-and-forth between the developers and the designers about the requirements of the project. Sometimes the developers are able to provide helpful feedback about more reasonable style/font/layout decisions, and vice versa.

One way to quickly test a variety of alternative design choices is to include an easy way to swap/add/remove CSS stylesheets from a webpage. You can use the same code I provided you in Lab 2 to turn the hamburger icon into a button that can toggle themes. **Focus on getting the hamburger icon to match the design specification first, then add the toggling behavior.** Once this is working, you should add one stylesheet to your webpage:

1. layouts.css
2. ~~Custom.css~~ not included, we will do this in Lab 4 instead

The theme toggler will allow these stylesheets to overwrite the default sheet (**style.css**) which you completed in Part 1. The requirements for each new stylesheet are elaborated below.

layouts.css (10%)

This stylesheet will allow you to easily identify the structural elements of your web page. In addition to the normal **style.css** rules, **layouts.css** should add the following styles:

1. Flex containers should be identified with a border of solid color (ex.: red, green, blue, yellow, purple, etc). An example is shown in the figure below.
2. Flex items should be identified with a dashed or dotted border.
3. An item can be a Flex container and item at the same time. In this case it will have a colored border that is dashed or dotted.
4. Block level elements / Float elements that are not inside Flex items or containers will have a black solid border.