ENSF-381: Full Stack Web Development Laboratory

Ahmad Abdellatif
Department of Electrical & Software Engineering
University of Calgary

Objectives

Welcome to the ENSF381 course lab! Below are the detailed instructions for creating and understanding an HTML document. The main objective of this lab is to apply the fundamental concepts of HTML structure, semantic elements, and inline styling.

Groups

Lab instructions must be followed in groups of two students. Submissions from groups with fewer than two students will not be accepted.

Submission

You must submit the complete source code, ensuring it can be executed without any modifications. Also, if requested by the instructor, you may need to submit the corresponding documentation file (e.g., word and image). Only one member of the group needs to submit the assignment, but the submission must include the names and UCIDs of all group members at the top of the code.

Deadline

Lab exercises must be submitted by **11:55 PM on the same day as the lab session**.

Exercise 1: Inspecting HTML Elements

Objective: Learn how to inspect and extract specific HTML elements from a webpage using the browser's Developer Tools.

1. **Open the Wikipedia Page**
   - Open your preferred web browser (e.g., Chrome, Firefox, or Edge).
   - Go to the following URL: University of Calgary - Wikipedia.
2. **Locate the "Faculties" Section**
   - Scroll down the page to find the **"Faculties"** section. This section contains a list of faculties available at the University of Calgary.

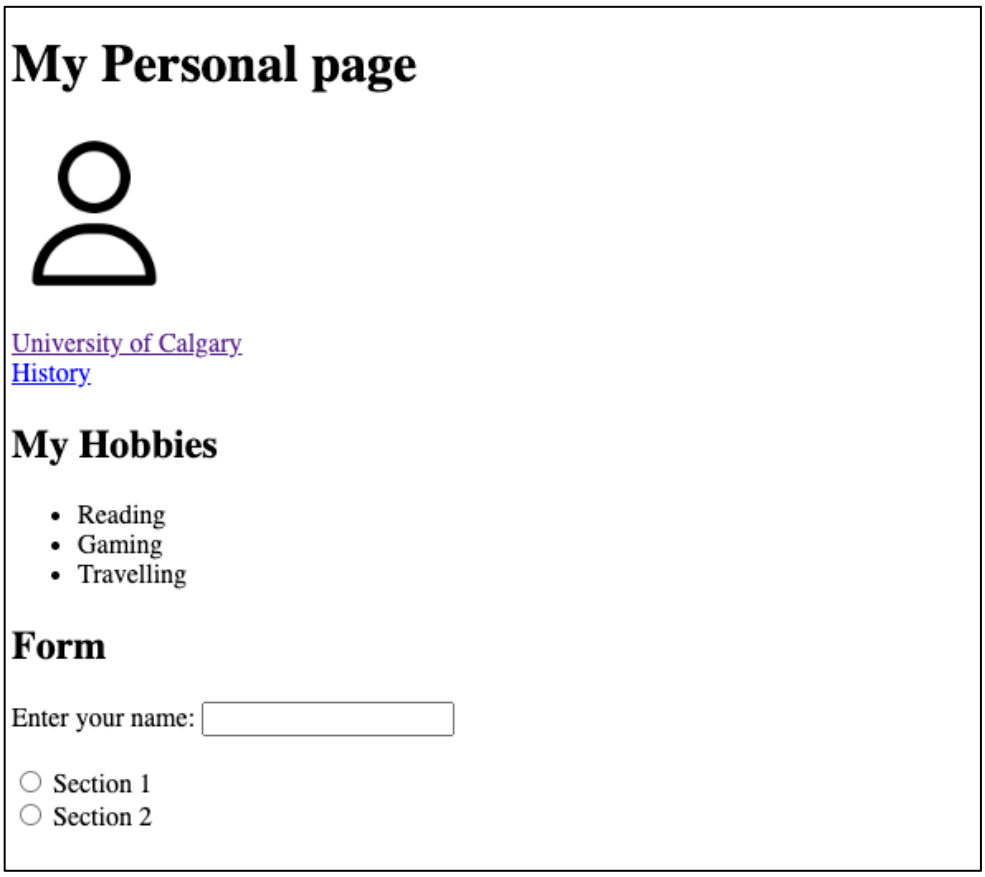
3. **Open the Developer Tools**
   - Right-click anywhere on the page (preferably near the "Faculties" section).
   - From the context menu, select **"Inspect"** (in Chrome and Edge) or **"Inspect Element"** (in Firefox).

   Alternatively, you can open the Developer Tools using the keyboard shortcut:

   - **Windows/Linux:** Press `Ctrl + Shift + I`
   - **Mac:** ⌘ `+ Option + I`
4. **Find the "Faculties" HTML Element**
   - In the Developer Tools panel, hover over different parts of the HTML in the **Elements** tab. The corresponding section on the webpage will be highlighted.
   - Move your mouse until the **"Faculties"** header or its content is highlighted on the webpage.
5. **Expand and View the HTML Structure**
   - If the HTML element corresponding to "Faculties" has a small triangle next to it, click the triangle to expand and see its child elements.
   - Carefully observe the HTML tags and hierarchy. Identify where the "Faculties" section begins and ends.
   - What heading tag (e.g., *<h1>, <h2>, <h3>*) does the 'Faculties' section use? *[write your answer in **the Answer_sheet**]*
   - Take a screenshot of the "Faculties" section highlighted in the Developer Tools panel and add it to the ***Answer_sheet***.

Exercise 2: Personal page

Objective: In this exercise, we will create a simple personal webpage using HTML. The purpose of this exercise is to familiarize yourself with more HTML elements, including lists, links, images, and forms. **All the required images and text are available in the supplementary folder.**



Setting Up the HTML File

- Create a new file and save it as *exercise2.html.*

Creating the Document Structure

1. Add the basic structure of an HTML document:
2. Inside the <head> tag, set the title to “Personal Page”.

Adding the Page Content

1. Add a top-level heading for your page title “My Personal Page” using first level (h1) heading.
2. Add a profile picture to the page as shown in the output. The profile picture is available in the D2L folder for this lab as *profile.png*.

3. Set the width and height of the profile picture to 100px. Ensure the profile picture is in the same folder as your HTML file.
4. Add a link to the University of Calgary website (https://www.ucalgary.ca) as shown in the output.
5. Add a link to the Exercise 3 page (*exercise3.html*) as shown in the output (*exercise2_output.pdf*).
6. Add a second level (*h2*) heading titled "My Hobbies".
7. Create a bulleted list of at least three hobbies using an **unordered list** as shown in the output.
8. Add a section for a form.
9. Include a text box for users to enter their name.
10. Add radio buttons to allow users to select between predefined sections as shown in the output. The user can select one section at a time.

Exercise 3: UofC History page

Objective: This exercise will teach you how to create a structured HTML page using a combination of text, images, and hyperlinks. Specifically, we will create a page like the UofC History page (https://ucalgary.ca/about/our-history). You will learn how to use tables for layout, define sections with headings and paragraphs, and embed images and links.

Setting Up the HTML File

1. Open your code editor and create a new file.
2. Save the file as *exercise3.html.*
3. Ensure that the newly created file is saved in the same location as exercise3.html.

Creating the Document Structure

1. Add the basic structure of an HTML document:
2. Inside the <head> tag, set the title to "Our History".

Explore the Page Layout

1. Open *exercise3_output.pdf*.
2. Observe how the content is divided into sections and arranged into a coherent structure.

Adding the Page Content

1. Inside the `<body>` tag, create a `<table>` to organize the content.

- Add this CSS (place it in the `<head>` section) to make the table structure temporarily visible:

```
<!--debugging style-->
<style>
  table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
  }
</style>
<!--debugging style -->
```

- Set the table width to 100%:

```
<table style="width: 100%;">
  <!-- table content goes here -->
</table>
```

1. Divide the table into **six rows** and **2 columns**:
    - One row for the banner image.
    - One row for the introduction text.

- Four rows for the content sections.

2. Add the banner image (banner.jpg) to the first row of the table. Use a single cell spanning the entire width of the table for the banner.
3. Set the image width to 100% and height to 600px.
4. Use first level heading (h1) for the title of the introduction to make it prominent. The text to insert is: “Like many Calgary success stories ….etc”.
5. Add multiple paragraphs to describe the historical context as shown in the output. Use the paragraph tag for each block of paragraph text. You can find the output text in the *supplementary.txt* file.
6. Add a horizontal line to separate this section from the next as shown in the output.
7. In the next row, use a first level heading (*h1*) for the section title (The roads that brought us here). Then, include the paragraph text shown in the output inside a paragraph tag. **Note:** Use a single cell that spans the entire width of the table for the banner.
8. For the rest of the rows:

    a. Use second level headings (*h2*) for each section title (e.g., “Calgary's urban evolution”) to differentiate them from the main title. The headings are:
        i. Calgary's Urban Evolution
        ii. Changing Face of Campus
        iii. Culture and Society
    b. Add supporting text within <*p*> tags for each section. The text to insert includes:
        i. For “Calgary's Urban Evolution”: “A university and a city growing together side by side.”
        ii. For “Changing Face of Campus”: “Creating an intellectual hub to empower the problem-solvers of the future, today.”
        iii. For “Culture and Society”: “Fueling the cultural and social transformation of a city and region.”
    c. Include an image for each section with dimensions of 570px by 400px.
    d. Provide links within each section for users to explore more information. These links should be displayed as "Read more" at the end of the descriptive text. The links are:
        i. For “Calgary's Urban Evolution” section: https://ucalgary.ca/about/our-history/calgarys-urban-evolution
        ii. For “Changing Face of Campus” section: https://ucalgary.ca/about/our-history/changing-face-campus
        iii. For “Culture and Society” section: https://ucalgary.ca/about/our-history/culture-and-society

9. Comment out the style we applied for debugging in *Step1*.
10. Ensure that the final output of your HTML page matches the provided output screenshot exactly.
11. Ensure that clicking on the ‘history’ link in the Exercise 2 html page redirects to the html page you have created for Exercise 3.

Question 4.1: What is the difference when you set the table width to a fixed width (e.g., 150px) compared to a percentage (e.g., 100%)? Explain how it affects the layout and responsiveness of the webpage.

Submissions

You need to submit the following items:

1. Fill out *Answer_sheet.docx*, including **the names of the group members along with their UCIDs**, a **screenshot of the outputs for Exercise 1 and Exercise 2**, and the **answers to Exercise1 and Exercise 3**. Submit the completed answer sheet (preferably as a pdf file) on D2L.
2. Push the code for Exercise 2 and Exercise 3 to GitHub.
3. Compress the files you created for the exercises into a zip file and upload it on **D2L** along with the answer sheet.

Part 1: Creating a GitHub Repository

1. Open GitHub in your browser.
2. Sign in with your GitHub account. If you do not have an account, create one.
3. Click on the + icon in the top-right corner of the GitHub page.
4. Select "New repository".
5. Fill in the repository details:
    - Repository Name: Provide a descriptive name (e.g., lab01).
    - Description (optional): Add a short description of the project.
    - Visibility: Choose Public.
6. Check the box "Add a README file".
7. Click "Create repository".

Part 2: Cloning the Repository

1. Copy the Repository URL
    - On the repository page, click the green "Code" button.
    - Copy the URL under HTTPS (e.g., https://github.com/your-username/html-lab.git).
2. Open Your Terminal
    - Open a terminal or command prompt on your computer.
3. Clone the Repository
    - Navigate to the folder where you want to store the repository:

```
cd ENSF381
```

    - Run the git clone command with the repository URL:

```
git clone https://github.com/your-username/lab01.git
```

- This will create a local copy of the repository in the specified folder.

Part 3: Adding Code and Output Image

1. Navigate to the Repository
   - In the terminal, move into the cloned repository folder:

```
cd lab01
```

2. Add Your Code and Output Image
   - Place your code files (e.g., exercise2.html) into the cloned repository folder.
3. Stage the Files
   - Use the git add command to stage the files for commit (do this for all required files:

```
git add exercise2.html exercise3.html
```

4. Commit the Changes
   - Write a commit message describing your changes:

```
git commit -m "Add HTML code for lab1"
```

5. Push the Changes to GitHub
   - Push your changes to the remote repository:

```
git push origin main
```