

Full Stack Development Lab Instructions
ENSF381: Lab02

Investigating HTML tags

Created by: Mahdi Jaberzadeh Ansari (He/Him)
Schulich School of Engineering
University of Calgary
Calgary, Canada
mahdi.ansari1@ucalgary.ca

Week 3, January 22/24, 2024

Contents

1	Introduction	1
1.1	Objectives	1
1.2	Prerequisites	1
1.3	Forming groups	1
1.4	Before submission	1
1.5	Academic Misconduct/Plagiarism	3
1.6	Marking Scheme	3
1.7	Complains	3
2	Exercise A (Creating a repository)	4
2.1	Initialize the Project	4
2.2	Inviting Collaborators to Your GitHub Repository	5
2.3	Deliverable	5
3	Exercise B (Working with images)	6
3.1	Create a favicon	6
3.2	Creating an HTML Page with Image Formats	7
3.3	Deliverable	8
4	Exercise C (Working with text-related tags)	9
4.1	Fixing Issues in an HTML File	9
4.2	Deliverable	9
5	Exercise D (Linking pages)	11
5.1	Required Files	11
5.2	Instructions	11
5.3	Deliverable	12
5.4	HTML Code for index.html	12

1 Introduction

1.1 Objectives

This lab is designed to teach you the fundamentals of creating web pages using only HTML. By the end of this lab, you will have created a basic web page with various sections and HTML elements.

1.2 Prerequisites

1. Basic understanding of computer operations.
2. A text editor (such as Notepad++, Visual Studio Code, or Sublime Text).
3. A web browser (Chrome, Firefox, Safari, etc.) to view your HTML file.
4. GitHub account and an installed Git client

1.3 Forming groups

- In this lab session, you can work with a partner (groups of three or more are not allowed).
- The main goal of working in a group is to learn:
 - how to do teamwork
 - how to not be a single player
 - how to not be bossing
 - how to play for team
 - how to tolerate others
 - how to behave with colleagues
 - how to form a winner team
 - and ...
- Working with a partner usually gives you the opportunity to discuss some of the details of the topic and learn from each other. Also, it will give you the opportunity to practice one of the popular methods of program development called pair programming. In this method, which is normally associated with the “Agile Software Development” technique, two programmers normally work together on the same workstation (you may consider a Zoom session for this purpose). While one partner, the driver, writes the code, the other partner, acting as an observer, looks over his or her shoulder, making sure the syntax and solution logic are correct. Partners should switch roles frequently in such a way that both have equivalent opportunities to practice both roles.
- If you decided to work with a partner:
 - Choose a partner that can either increase your knowledge or transfer your knowledge. (i.e., do not find a person with the same programming skill level!)
 - Please submit only one lab report with both names. Submitting two lab reports with the same content will be considered copying and plagiarism.

1.4 Before submission

- For most of the labs, you will receive a DOCX file that you need to fill out the gaps. Make sure you have this file to fill out.
- All your work should be submitted as a single file in PDF format. For instructions about how to provide your lab reports, study the posted document on the D2L called [How to Hand in Your Lab assignment](#).

- Please note that if it is group work, only one team member must submit the solution in D2L. For ease of transferring your marks, please mention the group member's name and UCID in the description window of the submission form.
- If you have been asked to write code (HTML, CSS, JS, etc.), make sure the following information appears at the top of your code:
 - File Name
 - Assignment and exercise number
 - Your names in alphabetic order
 - Submission Date:

Here is an example for CSS and JS files:

```
/*
=====
Name      : lab2_exe_A.css
Assignment : Lab 2 Exercise A
Author(s)  : Mahdi Ansari, William Arthur Philip Louis
Submission : May 21, 2030
Description : A CSS file for decorating X form
=====
*/
```

- Some exercises in this lab and future labs will not be marked. Please do not skip them, because these exercises are as important as the others in learning the course material.
- In courses like ENSF381, some students skip directly to the exercises that involve writing code, skipping sections such as “Read This First,” or postponing the diagram-drawing until later. That’s a bad idea for several reasons:
 - “Read This First” sections normally explain some technical or syntax details that may help you solve the problem or may provide you with some hints.
 - Some lab exercises may ask you to draw a diagram, and most of the students prefer to hand-draw them. In these cases, you need to scan your diagram with a scanner or an appropriate device, such as your mobile phone, and insert the scanned picture of your diagram into your PDF file (the lab report). A possible mobile app to scan your documents is Microsoft Lens, which you can install on your mobile device for free. Please make sure your diagram is clear and readable; otherwise, you may either lose marks or it could be impossible for TAs to mark it at all.
 - Also, it is better to use the [Draw.io](https://draw.io) tool if you need to draw any diagram.
 - Drawing diagrams is an important part of learning how to visualize data transfer between modules and so on. If you do diagram-drawing exercises at the last minute, you won’t learn the material very well. If you do the diagrams first, you may find it easier to understand the code-writing exercises, so you may be able to finish them more quickly.
- **Due Dates:**
 - You must submit your solution until 11:59 p.m. on the same day that you have the lab session.
 - Submissions until 24 hours after the due date get a maximum of 50% of the mark, and after 24 hours, they will not be evaluated and get 0.

1.5 Academic Misconduct/Plagiarism

- Ask for help, but don't copy.
 - You can get help from lab instructor(s), TAs, or even your classmates as long as you do not copy other people's work.
 - If we realize that even a small portion of your work is a copy from a classmate, both parties (the donor and the receiver of the work) will be subject to academic misconduct (plagiarism). More importantly, if you give exercise solutions to a friend, you are not doing him or her a favor, as he or she will never learn that topic and will pay off for this mistake during the exam or quiz. So, please do not put yourself and your friend in a position of academic misconduct.
 - You can use ChatGPT, but please note that it may provide similar answers for others too, or even the wrong answers. For example, it has been shown that AI can hallucinate, proposing the use of libraries that do not actually exist ¹. So, we recommend that you imagine ChatGPT as an advanced search engine, not a solution provider.
 - In order to find out who is abusing these kinds of tools, we will eventually push you toward the incorrect responses that ChatGPT might produce. In that case, you might have failed for the final mark and be reported to administration.
 - If we ask you to investigate something, don't forget to mention the source of your information. Reporting without reference can lead to a zero mark even by providing a correct answer.

1.6 Marking Scheme

- You should not submit anything for the exercises that are not marked.
- In Table 1, you can find the marking scheme for each exercise.

Table 1: Marking scheme

Exercise	Marks
A	10 marks
B	10 marks
C	15 marks
D	15 marks
Total	50 marks

1.7 Complains

- Your grades will be posted one week following the submission date, which means they will be accessible at the subsequent lab meeting.
- Normally, the grades for individual labs are assessed by a distinct TA for each lab and section. Kindly refrain from contacting all TAs. If you have any concerns regarding your grades, please direct an email to the TA responsible for that specific lab.

¹<https://perma.cc/UQS5-3BBP>

2 Exercise A (Creating a repository)

2.1 Initialize the Project

Last time, we learned how to start Git on a local machine. This time, we will clone an existing project from a remote repository. Cloning is particularly useful when you want to create a local copy of a remote repository to work on. Follow these steps:

- Create a new repository on GitHub named **Lab2**. Remember to:
 - Check the box for **Add a README file**. This file serves as an introduction and overview of your project.
 - Select **MIT License** for your project's license. This is a permissive license that allows others considerable freedom with your code while still crediting you.
- After creating the repository, it should contain two files: a **README.md** and a **LICENSE** file. The **README.md** is where you can write about your project, and the **LICENSE** file contains the license text.
- To clone the repository to your local machine, open your terminal and navigate to the directory where you want the project to be located. In this case, navigate to `~/ENSF381/` (`~` means the user home directory):

```
cd ~/ENSF381/
```

- Then, use the `git clone` command followed by the URL of your remote repository. You can find this URL on your repository's page on GitHub. The command will look like this:

```
git clone https://github.com/yourusername/Lab2.git Lab2
```

Replace **yourusername** with your actual GitHub username. This command will create a directory named **Lab2** in `/ENSF381/` and initialize a Git repository in it, pulling all the content from the remote repository.

- Navigate into your newly created project directory:

```
cd Lab2
```

- To verify that the cloning process was successful, you can list the files in the directory:

```
ls -l
```

You should see the **README.md** and **LICENSE** files, along with any other files that were in the remote repository.

In this section, you've learned how to clone a remote repository, which is a crucial step in collaborative software development. Cloning allows you to work on your projects locally while maintaining a connection with the remote repository.

2.2 Inviting Collaborators to Your GitHub Repository

Collaborating with others on a project is a key aspect of software development. GitHub allows you to add collaborators to your repository. Here's how you can invite your colleagues to the repository you created in the previous steps:

- First, log in to your GitHub account and navigate to the repository you want to share. In this case, it's the **Lab2** repository.
- On your repository's page, locate the **Settings** tab near the top of the page and click on it. This will take you to the repository settings.
- In the settings menu, on the left side of the page, you will find a section called **Manage access**. Click on it.
- On the **Manage access** page, you will see a button labeled **Invite a collaborator**. Click on this button.
- GitHub will prompt you to enter the GitHub username or email address of the person you want to invite. Enter the username or email of your colleague.
- Once you enter the username or email, GitHub will suggest the matching account. Click on the account to select it.
- After selecting the collaborator, you can set the role for them. For most cases, the **Write** access level is sufficient. This allows them to push changes to the repository but doesn't allow actions like deleting the repository.
- Click on **Add collaborator** to send the invitation. Your colleague will receive an email from GitHub with a link to accept the invitation.
- Inform your collaborator to check their email and accept the invitation to collaborate on your repository.

Once your colleague accepts the invitation, they will have access to the repository and can start collaborating with you on the project. This process is crucial for teamwork in software development, allowing multiple people to work together seamlessly on the same codebase. Ask your colleges to clone the repository on their local machines.

Please note: *In the subsequent sections, each exercise should be completed by a single individual. It is imperative to rotate the role of the developer. This means that if Person X performs the steps in Exercise B, then Exercise C must be undertaken by Person Y, and this pattern should continue accordingly. We will check the commits on GitHub to give you marks.*

2.3 Deliverable

1. Add the address of your GitHub repository to your answer sheet file. Please make sure the repository is public and do not delete it until the end of the semester.

3 Exercise B (Working with images)

In this section, we will learn how to create an HTML page that demonstrates various image formats. We will focus on how vector images (SVGs) maintain their quality when resized compared to raster images (JPEG, GIF). The images for this exercise are provided to you via D2L. We are going to create a page that, at the end, seems like Figure 1.

Figure 1: Final output of the images.html file



3.1 Create a favicon

1. Go to <https://www.favicon.cc/>.
2. Design a favicon. You can see what I designed in Figure 2.
3. Download it into your project folder
4. Commit and push with the comment `Add favicon`.

Figure 2: favicon.cc Website



3.2 Creating an HTML Page with Image Formats

1. Create a basic HTML file

- (a) Create an HTML file inside your project folder and name it `images.html`.
- (b) Type the following content into it:

```
<!--
=====
Name      : images.html
Assignment : Lab 2 Exercise B
Author(s)  : Mahdi Ansari, William Arthur Philip Louis
Submission : May 21, 2030
Description : A simple HTML file.
=====
-->
<!DOCTYPE html>
<html>
<head>
  <title>Image Demonstration</title>
  <title>Image Demonstration</title>
  <meta charset="UTF-8">
  <meta name="description" content="An HTML example with img tags
">
  <meta name="keywords" content="HTML, Web Development, Example,
img">
  <meta name="author" content="Your Name">
  <link rel="shortcut icon" href="favicon.ico" type="image/x-icon
">
</head>
<body>
  <h1>Image Format Demonstration</h1>

  <h2>JPEG Image</h2>
  <p>This is a standard JPEG image:</p>
  

  <!-- Continue extra codes here -->
</body>
</html>
```

- (c) If you open your HTML file in this step you must see some texts and an icon that shows it couldn't load the image.
- (d) Create a folder beside the `images.html`, name the folder as `images`, and copy and paste the example images we provided for you (i.e., `example.jpg`, `example.svg`, `example.gif`).
- (e) Refresh the page again and you must see the the first picture.
- (f) Commit and push the 4 new files that you have added to your project so far.

2. Understanding the Basic Structure:

- `<!DOCTYPE html>`: This declaration defines the document to be HTML5.
- `<html>`: The root element of an HTML page.
- `<head>`: Contains meta-information about the document, like its title.

- `<title>`: Specifies the title of the document, shown in the browser's title bar or page's tab.
- `<body>`: Contains the content of the HTML document, such as text, images, links, etc.

3. Adding Headings and Paragraphs:

- `<h1>`, `<h2>`: Headings. `<h1>` is the highest level, and `<h2>` is a subheading.
- `<p>`: Paragraph. This tag is used for writing paragraphs.

4. Inserting Images:

- To insert an image, use the `` tag.
- The `src` attribute specifies the path to the image file.
- The `alt` attribute provides alternative information for the image if a user cannot view it.
- Example: ``

5. Understanding Image Formats and Quality:

- **JPEG**: A common image format for photographs and realistic images. Does not support transparency.
- **GIF**: Used for simple animations and graphics. Supports transparency but limited to 256 colors.
- **SVG (Scalable Vector Graphics)**: A vector image format. Unlike JPEG or GIF, SVGs are not pixel-based. They are composed of mathematical equations that define shapes, paths, and fills. As a result, SVGs can be scaled to any size without losing quality. This is why setting a large width (e.g., `width="2000"`) does not pixelate the image.

6. Add SVG and GIF images:

- For each image format (SVG, GIF), add a subheading (`<h2>`), a paragraph (`<p>`) to describe the image, and then insert the image using the `` tag.
- Make sure the image source (`src`) correctly points to the location of your images. In this example, images are stored in an `images` folder.
- Test the page in a web browser. Resize the browser window to observe how the SVG image retains its quality due to its vector nature.
- Zoom the page to 500% and see how the quality of the SVG image remains the same while the quality of other raster images is diminished.

3.3 Deliverable

Complete the following steps to understand how HTML elements render on a web page and to capture a visual representation of your work:

1. Open the Developer Tools in Google Chrome. You can do this by pressing **F12** on a Windows machine.
2. Navigate to the **Elements** tab. This tab displays the HTML structure of your webpage. Here, you will find all the tags you have used in your HTML document.
3. Hover your mouse cursor over each element in the **Elements** tab. Notice how the corresponding section on the webpage is highlighted. This feature helps you visually connect the HTML code with its rendered output on the page.
4. Locate the `<html>` tag in the **Elements** tab. Right-click on it and select **Capture node screenshot** from the context menu. This action will capture a screenshot of the entire HTML node.
5. Save the screenshot to your desktop. Ensure that the screenshot, which visually represents your HTML structure, is clearly visible.
6. Include this screenshot in your answer sheet file. This will serve as a visual proof of your work and understanding of the webpage's structure.

4 Exercise C (Working with text-related tags)

In this section, we will explore basic text formatting techniques, experiment with ordered and unordered lists, including nested lists, and undertake a series of practical exercises. The anticipated outcome of these activities is depicted in Figure 3.

4.1 Fixing Issues in an HTML File

To enhance your understanding of HTML and gain hands-on experience, follow these steps:

1. Remember to rotate the developer role for this exercise. Each member should get an opportunity to contribute.
2. First, pull the latest changes that your colleagues have made to the project.
3. Download the `news.html` file available on the D2L platform.
4. Add the downloaded file to your project folder. Commit the addition to your Git repository with the message `Add news file` and push the changes to GitHub.
5. Before making any modifications, open the `news.html` file in a web browser to view its current state.
6. Begin updating the file based on the instructions provided within the comments in the HTML code. There are nine specific comments you need to address. Each comment guides you on how to modify or enhance a particular part of the HTML file.
7. Commit and push the final changes to your GitHub repository.

This exercise will not only improve your HTML skills but also familiarize you with the process of interpreting and implementing code modifications based on inline documentation and comments.

4.2 Deliverable

1. Make a screenshot from the `<html>` tag of the final file, as you did in the previous exercise.
2. Include this screenshot in your answer sheet file.

Figure 3: Final output of the `news.html` file

Welcome to My HTML Page

Section 1

Section 2

Section 3

Article Title

This is a paragraph in an article section. It demonstrates text formatting, like **bold**, *italic*, and **marked** text.

Here's a quotation:

This is a blockquote, a section that quotes content from another source.

The article section typically contains text, images, and other content that forms an independent part of a document.

List Examples

Ordered List:

1. First item

2. Second item

1. Second-First item

2. Second-Second item

3. Second-Third item

3. Third item

Unordered List:

Bullet item

First-Bullet item

Second-Another bullet item

Third-Yet another bullet item

Another bullet item

Yet another bullet item

Additional Elements

This section includes a variety of additional HTML elements:

Definition Lists:

A list where each item has a term and definition.

HTML

Hypertext Markup Language.

CSS

Cascading Style Sheets

Tables:


Here is an example of a table.

Monthly Sales Report

Month	Sales	Expenses
January	\$2000	\$1500
February	\$1800	\$1200
Total	\$3800	\$2700

Images:

Adding an image:



Copy right Mahdi Ansari.

10

5 Exercise D (Linking pages)

In this exercise, you will learn to link multiple HTML pages. The anticipated outcome of these activities is depicted in Figure 4.

Figure 4: Final output of the `index.html` file



5.1 Required Files

- `index.html`
- `news.html`
- `images.html`

5.2 Instructions

1. Linking HTML Pages:

- Create and open `index.html` and start by copying and pasting the content of the file from section 5.4.
- Follow the comments and ensure links to `news.html`, and `images.html` are properly set up in the navigation section.
- Verify that each linked file is in the same directory as `index.html`.

2. Testing Navigation:

- Open `index.html` in a web browser.
- Click each link to confirm the correct navigation between pages.

3. Adding Back-Links:

- (a) Open `news.html` and `images.html` in your HTML editor.
- (b) At the **top** of each page, add a hyperlink (`
Back to Home`) to enable navigation back to `index.html`.
- (c) Ensure that these backlinks are correctly functioning by testing them in a web browser.
- (d) At the **bottom** of each page, add a hyperlink (`
Back to Home`) to enable navigation back to the home page! **Please note that this time we haven't provided the file name in href attribute.**
- (e) Right-click on the `index.html` file in File Explorer and test all backlinks. Are all of them correctly functioning?
- (f) Right-click on the `index.html` file in the Explorer pane in VSC and select **Open with Live Server** to see your webpage in action. Test all the backlinks again. Are all of them correctly functioning?

5.3 Deliverable

1. Write your answers to the questions to the items 3e and 3f in the section 5.2 into the answer sheet file.
2. **Creating a JPG Screenshot:**
 - Take a screenshot of the `index.html` page.
 - Include this screenshot in your answer sheet file.
3. **Creating a GIF:**
 - Record a GIF video navigating through the linked HTML pages.
 - You can use a variety of tools for creating GIF images, for example, <https://www.screentogif.com/>.
 - Save this GIF in the project folder.
4. **Uploading to GitHub:**
 - Add the newly created GIF file to the GitHub repository.
 - Commit and push the changes to the `index.html` file with an appropriate message.

5.4 HTML Code for index.html

```
<!--
=====
Name      : index.html
Assignment : Lab 2 Exercise D
Author(s)  : Mahdi Ansari, William Arthur Philip Louis
Submission : May 21, 2030
Description : A simple HTML website with different tags.
=====
-->
<!DOCTYPE html>
<html>
<head>
  <title>Comprehensive HTML Examples</title>
  <meta charset="UTF-8">
  <meta name="description" content="A comprehensive HTML example with various tags">
  <meta name="keywords" content="HTML, Web Development, Examples">
  <meta name="author" content="Your Name">
  <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
</head>
<body>
  <header>
    <h1>Welcome to My HTML website</h1>
  </header>

  <nav>
    <ul>
      <li><!--Add a hyperlink to the News page.--></li>
      <li><!--Add a hyperlink to the Images page.--></li>
    </ul>
  </nav>
</body>
</html>
```