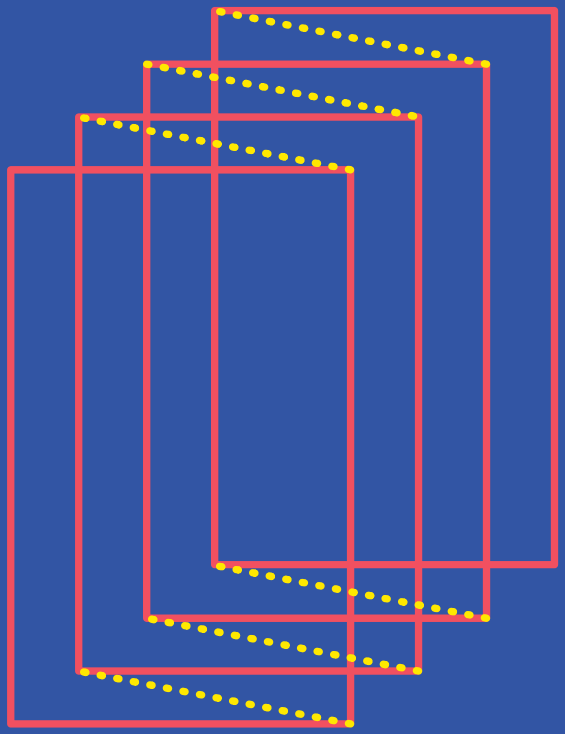
HTML AND CSS
a how-to guide

SWATI MEHTA



EMM LAB GUIDE

EMM LAB

Experimental Methods and Media

Suggested Citation: Swati Mehta, HTML and CSS: A How-to Guide,” Guide. Experimental Methods and Media Lab. Trent University, Peterborough, Ontario. August 2024.

TABLE OF CONTENTS

PREFACE

This guide will introduce you to the world of HTML and CSS. It will teach you how to read, write, and comprehend HTML and CSS, with examples along the way. If you follow these examples, by the end of this guide you will have designed a static website. In the process, this guide will also help you think about code, software and other technical things that are concealed from our day-to-day interactions with technology. These thoughts will hopefully lead to more thinking about art, methods, and how we go about studying and doing things in our very media saturated environments.

The guide is designed for all levels of expertise. This is to say, you can use this if this is the first time you've heard of HTML and CSS, have a rough idea and would like to know more, need a refresher, are an absolute expert, or just curious about what's going on here. You only need a laptop or desktop computer, basic typing skills, and a text editor like notepad or WordPad (available on every computer for free!) to get started.

The guide also has a list of resources: tutorials, videos, zines, to help you further develop HTML and CSS skills. This list is in no way exhaustive or definitive. For everything that's listed here, there's a gazillion more just a search or prompt away. This guide just nudges you in the right direction, so you know what to look for, check if it is accurate, know how to fix it, and make it work for your own requirements.

Let's get started!

PART I: HTML

HTML: LANGUAGE OF THE WORLDWIDE WEB

HTML is an abbreviation for Hypertext Markup Language[1]. It is a language that allows us to view and share websites on the Internet. No matter how sophisticated a website is, it still uses HTML. Hypertext refers to a system of linking documents so they can be shared easily across a network of computers. Markup language provides structure for linking these pages and documents[2]. In other words, HTML describes the structure of a website and uses a wide variety of elements to display content on these pages across web browsers. We'll see how structure and elements work together in HTML when we go through the process of creating a website.

The first version of HTML (i.e., HTML 1) was created by Sir Tim Berners Lee, a physicist at CERN in Switzerland, in 1991, two years after he created the worldwide web in 1989[3]. However, this version of HTML was not officially released until 1995 when it was published as HTML 2.0[4]. HTML-5 was the fifth and final version of HTML released by the World Wide Web Consortium (W3C) in 2008. The current version is known as the HTML living standard and is managed by a consortium of major web browsers i.e., Apple, Google, Microsoft, and Mozilla, known as the Web Hypertext Application Technology Working Group (WHATWG)[5].

1 A Simple Guide to HTML: https://www.simplehtmlguide.com/whatishtml.php

2 HTML Introduction: https://www.w3schools.com/html/html_intro.asp

3 Raggett, D., Lam, J., Alexander, I., & Kmiec, M. (1998). A history of HTML. In Raggett on HTML 4. Addison-Wesley Longman Publishing Co., Inc.

4 HTML History: https://www.w3schools.in/html/history

5 W3C and WHATWG to work together to advance the open Web platform: https://www.w3.org/blog/2019/w3c-and-whatwg-to-work-together-to-advance-the-open-web-platform/

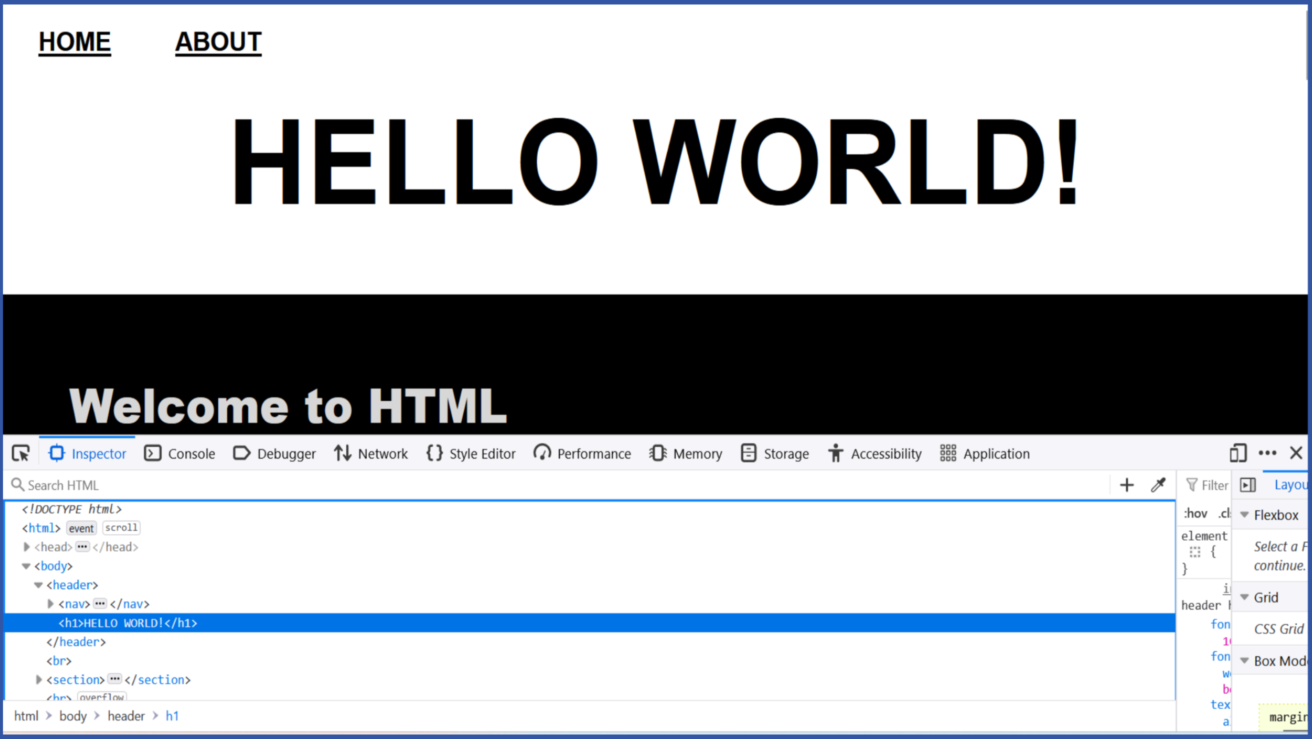
WHY SHOULD WE KNOW HOW TO USE HTML?

Developing websites is extremely easy and often requires minimum knowledge of HTML. Most webhosting platforms, like WordPress, Squarespace, or Wix, have integrated HTML, CSS and a variety of plug-ins into templates and drag-and-drop boxes to create user friendly web development interface. As a result, users like us are less likely to interact directly with HTML when setting up a website. Moreover, generative AI applications like ChatGPT can easily generate near perfect HTML and CSS for more customized options.

When making a website is so easy, why should we take on the extra work of familiarizing ourselves with HTML?

For the same reasons we learn any language. Knowing a language allows us to communicate, comprehend, and create in that language. It is like being added to an inner circle, where suddenly everything starts making more sense than it used to before we became fluent in that language. The ability to read, write, and understand HTML even at a basic level is an entry to the inner circle of the world wide web. An awareness of the structure and elements that create a page can reveal why a website looks and works the way it does.

On a functional level, familiarity with HTML can help you maintain your website without having to run to 'techie' to fix a broken link. Even if you do outsource the task to a web developer, you'll be able to communicate better if you understand how websites work. Though most proprietary webhosting services limit access to the source code, they still allow users to switch between visual and html editors. This means that you'll know exactly where to look when something does not work on the default design template. Most importantly, you will know how to fix it. You could easily ask a generative AI application to write an almost perfect HTML script for you and copy-paste it. The knowledge of HTML, however, will help you evaluate the output, test the AI's capability, and find its limitations. That's the advantage of knowing a language.



```
<!DOCTYPE html>
<html> event scroll
  <head> ··· </head>
  <body>
    <header>
      <nav> ··· </nav>
      <h1>HELLO WORLD!</h1>
    </header>
    <br>
    <section> ··· </section>
    <br> overflow
```

Image 1: Even though we can simply drag-and-drop elements to easily develop a website, there's HTML behind the scenes. Here's a quick way to check: Open your favorite website in any browser. Right click on the page and select 'inspect element' from the menu that opens. A dialog box will open at the bottom of the screen. Can you spot any HTML elements?

It can be difficult to resist the allure of convenience offered by third-party platforms. They are quick and offer all kinds of attractive design templates for websites. All these conveniences come at a cost. Access to templates, plugins or even the ability to customize templates is blocked behind steep subscription plans. Pricing is always subject to change. It can be tough to pack-up and leave if your website already has a dedicated user base. Even if a website is migrated to a different host, the expenses do not go away. Self-coding and hosting a website, in contrast, is cheaper than third party platforms. There is also the significant advantage of owning the source-code. It comes with the convenience of customizing almost every aspect of a website with minimal spending on subscriptions.

Though most proprietary web-hosting platforms make claims of being ethical and sustainable, verifying these claims is often difficult and almost impossible. What we do know is that data-centers power these platforms, and their energy consumption is far removed from being sustainable[6]. This, along with data sharing, privacy, and advertising models used by tech companies creates a paradox of (un)ethical choices for responsible use of technology. Self-hosting provides opportunities to address these challenges. It gives you space to experiment with sustainable alternatives such as using local, decentralized servers or solar powered servers. It also affords more control over the type and amount of data you wish to collect and how do you use it. Self-coding and hosting, therefore open up several possibilities to experiment with alternatives that could in-turn aid attempts to re-imagine and reconstruct existing formations on how we communicate on the internet.

Apart from gaining tech literacy, learning HTML also provides an awareness of how socio-technical systems work. As researchers, we are often looking to unearth and discover relationships and meanings of different subjects and

6 Pasek, A. (2023). Getting Into Fights With Data Centers: Or, a Modest Proposal for Reframing the Climate Politics of ICT. White Paper. Experimental Methods and Media Lab, Trent University, Peterborough, Ontario. https://emmlab.info/Resources_page/Data%20Center%20Fights_digital.pdf

objects in society. This often requires a close reading of texts to understand the conditions that created the text in question. The tricky thing about software is that it obfuscates. In other words, we can never really see or learn all the software that is involved in the functioning of technology because it is hidden to create seamless user experience[7]. Moreover, AI with all its promise of efficiency will further reduce access to these layers of technology and software that mediate our experience. As our social interactions are increasingly being mediated through layers of software, we need the ability to read a language or code that configures these systems.

Artistic expression through code is another way to push technical boundaries. The syntax of different coding languages is full of opportunities for experimentations like code poetry (https://www.sourcecodepoetry.com)

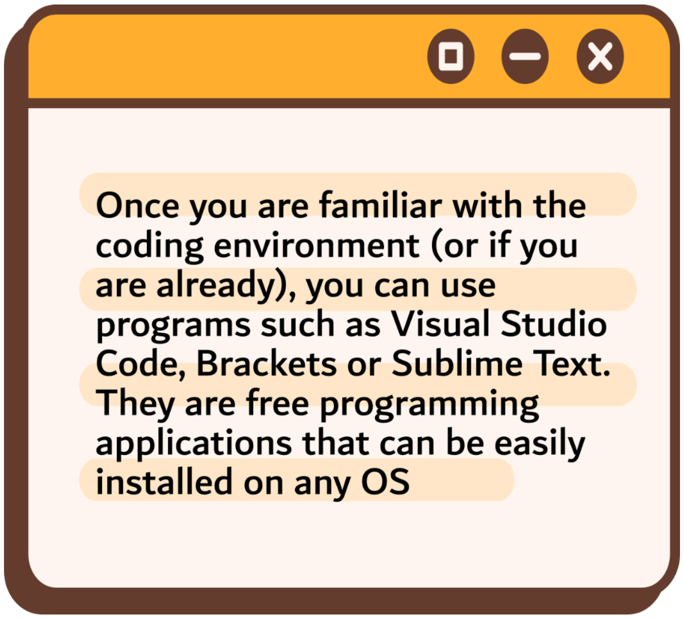
A small peek into these heavily concealed technical systems can give us some sense of how these systems are entangled with our social realities. Knowing any software, a programming language, or a markup language like HTML affords us these visions. Learning HTML is therefore an excellent starting point to understand not just the world wide web but also logics used by coding languages.

7 Fuller, M. (2003). Behind the blip: Software as culture. New York: Autonomedia.

HOW DO WE USE HTML?

TECH REQUIREMENTS

1. Laptop or desktop computer (any operating system)
2. A .txt editor. Most commonly used operating systems (OS) have a default text editor application. It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on Ubuntu.



Once you are familiar with the coding environment (or if you are already), you can use programs such as Visual Studio Code, Brackets or Sublime Text. They are free programming applications that can be easily installed on any OS

3. To host the website online, you'll need an account on a webhosting service. We are using Neocities (https://neocities.org/) in this tutorial but you are welcome to use any similar service for your purpose. It is also possible to set-up your own self-hosting server. The Solar-Powered Media zine[8] is an excellent guide for setting up a solar-powered server using a simple raspberry pi.

[8] Pasek, A, and Benedetta P. (2021). Solar-Powered Media. http://lowcarbonmethods.com/local/zine.html

GETTING STARTED

Like any language, HTML has its set of rules, such as syntax, grammar, and vocabulary, which when used correctly yield a legible webpage. The logic of HTML is simple: there are elements and tags that define categories and content. Different kinds of content tags can be nested under elements. For HTML to work, these elements and tags need to be arranged in a logical order, following all the rules of the language.

Think of it as ordering a sandwich at Subway. To get a sandwich made successfully you need to provide instructions in a very specific order. First, the type of bread and its size defines the structure of your sandwich. This is typically followed by instructions for your preference for cheese and having the selected bread toasted. Once the bread is toasted (or not depending on your preference), you proceed to select the ingredients of your sandwich. Finally, you select the sauce. Anyone who has tried getting a sandwich at Subway, knows that messing up with the sequence of your order is not an entirely pleasant experience. To produce a sandwich without minimum disruption, one must follow the rules of ordering it set by the sandwich maker.

The logic of HTML is not very different, except that it produces a website instead of a sandwich at the end of the process. A fault in the logic can break a website or just frustrate the user by not producing the desired outcome. This is also to say that HTML is as simple as ordering a sandwich. The first attempt can be daunting but once you understand its logic it serves you well.

Let's begin!

1. Create a folder

The first step is to set up a folder which will act as a home for your HTML page and images. This folder, known as the root folder, defines a 'path' for the website i.e., it tells the HTML where to get images or styles sheets.

So go ahead, create a folder on the desktop or any other drive on your computer/laptop that is easily accessible.

2. Create a .html file

   Now that we have a folder, we can go ahead and create a .html file and save it in the root folder. The file extension .html designates the document as a webpage. In other words, it tells a browser that this is a webpage.

   - Open the default .txt editor and a new blank file.
   - Then click on “save as” and make sure you select the folder we just created as the location for this file.
   - Name the file: ‘index.html’ save it and close the file (for now).
   - Go back to the home folder and check if you’ve saved the file correctly.
   - If you see the default web browser icon next to the ‘index.html’ file, you’ve got it right.
   - Now double click to open the ‘index.html’ file. It should open as a tab in your default web browser. Congratulations! You have created a webpage. It’s blank, but it exists!

Let’s cross-check a couple of things before we go further and fill out this page.

- Check the URL of the page: it should consist of the name of your folder followed by the name of the page.
- The page name ‘index.html’ is the default home page of your website.

It is important to note that though this page opens in a web browser, it is not in fact online yet. The .html file is locally hosted on your computer and this link only works on the system you are working on. To be online, this page needs to be hosted on a web server. We'll get to that once your page has some content and is ready to say hello! to the world.

3. Page structure

Let's create a structure for this page.

- Go back to the root folder and right click on the 'index.html' page.
- Hover over the 'open with' option and select the txt editor. Windows users will get an option to 'edit in Notepad' in the main menu after they right click. You can select that instead of the 'open with' option.
- Your index.html file should now be open as a txt file.

The first instruction we enter on this page is:

```
<!DOCTYPE html>
```

This informs the web server that this document is an html file.

Once you've typed this, press enter to get to the next line. Here, we tell the server that the html begins from this point onwards. To do this we use the tag:

```
<html>
```

Now everything that follows under this tag will be read as html for this page till we close the tag. This tag will be closed at the end as we want all our html elements nested under this tag.

HTML 101

Use an underscore (_) instead of a space while naming files, images, and folders

HTML 101

Save all your files, images, and additional folders related to the website in same root folder

HTML 101

All element tags are in lower case e.g., <p> and not <P>

HTML 101

Close tags! Every open tag needs a corresponding closing tag

Once again, press enter and move to the next line after you've typed the <html> tag.

Next, we will define elements that contain details about this webpage. This information is known as 'meta information' or 'metadata'. The information we enter here is not visible to the end user but is used by browsers to render the page correctly. To define this information, we use the <head> tag:

```
<head>
```

We will define some parameters for browsers under this tag.

First, we define the character set i.e., an element that instructs the browsers to read universal fonts and characters on the website.

Next, we instruct browsers on how to size the webpage for different screen sizes and devices.

Then, we give the browsers information that should show up on search engine pages. This is the descriptive text a user will see on a search engine's result page.

Finally, we define the title of the website, i.e. , the name of the page that an end user should see in the address bar and the browser tab.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="A short description of your webpage. It
should not be longer than a sentence or two.">
<title> HTML_Guide </title>
```

Let's close the <head> tag for now. We'll come back to it when it is time to add some CSS.

```
</head>
```

4. Content

Now that we've defined all the meta information, we can start adding content to the page. These parts will be visible to the end user.

First, we define the body with the `<body>` element. All the elements introduced under this tag will form the main body of the webpage.

i. Let's add a header to this page using the `<header>` tag and a heading using predefined heading levels in HTML.

```
<body>
<header>
<h1> HELLO WORLD!</h1>
</header><br>
```

We'll add a basic site navigation later. For now, we can move on to adding more content to the website. Save the file.

ii. Let's add some content to the page!

First, we'll define a new section with the `<section>` element. Next, we add a heading for the section using a predefined heading level in HTML. Then we can add some paragraphs using the `<p>` tag.

```
<section>
```

```
<h1> Welcome to HTML </h1>

<p>This guide will introduce you to the world of HTML. It will teach you how
to read, write, and comprehend HTML. In the process, it will also help you
think about code, software and other technical things that are concealed from
our day-to-day interactions with technical interfaces. These thoughts will
hopefully lead to more thinking about art, methods, and how we go about
studying and doing things in our very media saturated environments. At the
end of this guide, you will have designed a static web page.</p>

<br>
```

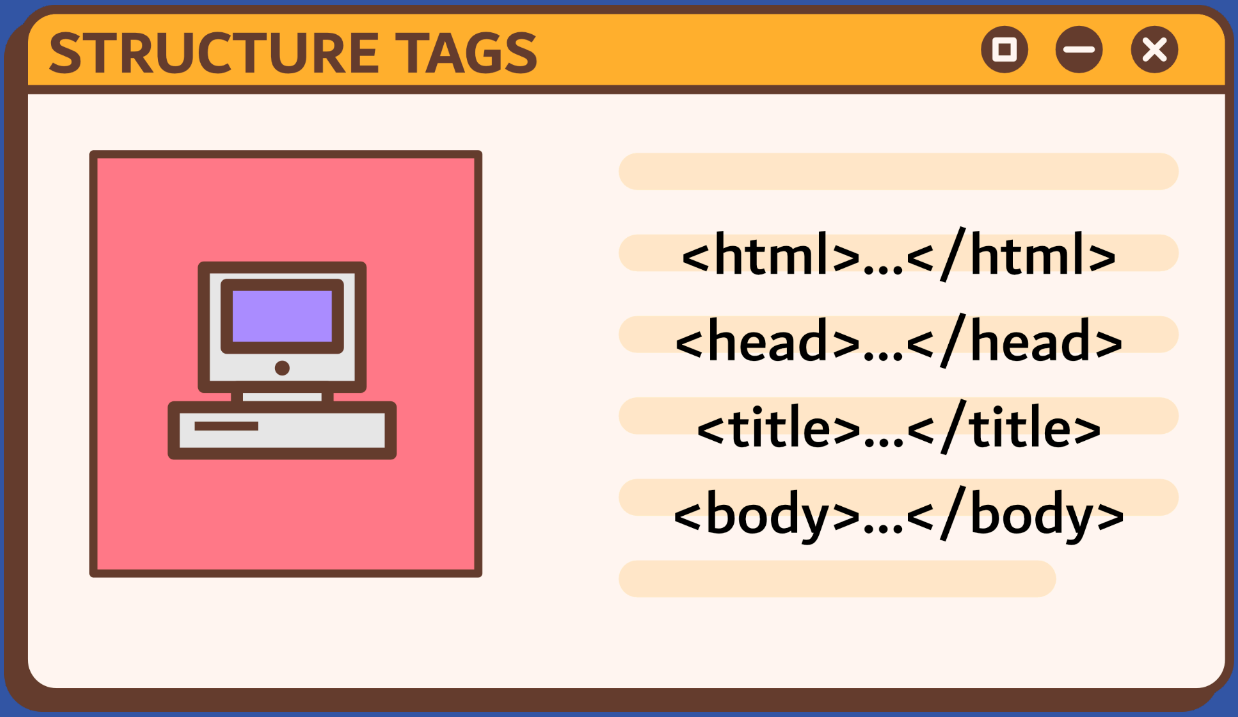
iii. Let's add an image using the `<img>` tag.

First, make sure the image file you want to use is saved in your root folder. Copy the image file name along with the extension i.e., "img_name.png" (or .jpg depending on the image you have saved).

Before adding the image, we define a container for this image using the `<div>` tag. Everything added under the `<div>` container is considered a single unit. This makes it easier to apply different styles to these elements.

```
<section>

<h1> Welcome to HTML </h1>

<p>This guide will introduce you to the world of HTML. It will teach you how
to read, write, and comprehend HTML. In the process, it will also help you
think about code, software and other technical things that are concealed from
our day-to-day interactions with technical interfaces. These thoughts will
hopefully lead to more thinking about art, methods, and how we go about
studying and doing things in our very media saturated environments. At the
end of this guide, you will have designed a static web page.</p>

<br>

<div>

<img src="html_3.png"alt="banner_img"/></img>

</div><br>
```

iv. Let's add external links with the `<a>` tag.



<html>...</html>

<head>...</head>

<title>...</title>

<body>...</body>

Source: https://www.simplehtmlguide.com/cheatsheet.php

COMMON TAGS

<h?> Heading </h?>	(h1 for largest to h6 for smallest)
<p> paragraph </p>	Paragraph of Text
<b> bold </b>	Make text between tags bold
<i> italic </i>	Italicize text between tags
<a href="url"> text </a>	Hyperlink text to another webpage
<div> ... </div>	Divide content into sections
<img src="filename.jpg">	Add an image
<ul> list <li> </li> </ul>	Unordered bullet list
<ol> list <li> </li> </ol>	Ordered list
<br>	Line Break

Source: https://www.simplehtmlguide.com/cheatsheet.php

```
<section>

<h1> Welcome to HTML </h1>

<p>This guide will introduce you to the world of HTML. It will teach you how
to read, write, and comprehend HTML. In the process, it will also help you
think about code, software and other technical things that are concealed from
our day-to-day interactions with technical interfaces. These thoughts will
hopefully lead to more thinking about art, methods, and how we go about
studying and doing things in our very media saturated environments. At the
end of this guide, you will have designed a static web page.

 </p><br>

<div>

<img src="html_3.png"alt="banner_img"/></img>

</div>

<p> The guide is designed to help all levels of expertise. This is to say,
you can use this if this is the first time you have heard of HTML, have a
rough idea and would like to know more, need a refresher, are an absolute
expert, or just curious about what's going on here. You only need a laptop or
desktop computer, basic typing skills, and a text editor like notepad or
WordPad (available on every computer for free!) to get started.<br><br>

The guide also has a long list of resources: <a href=
"https://www.w3schools.com/html/default.asp">tutorials,</a> videos, zines, to
help you further develop these newly acquired HTML skills. This list is in no
way exhaustive or definitive. For everything that is listed here, there's a
gazillion more just a search or prompt away. This guide just nudges you in
the right direction, so you know what to look for, check if it is accurate,
know how to fix it, and make it work for your own requirements.

Let us get started!</p>
```

v. Let's make a list.

There are specific tags for adding lists in HTML. The `<ul>` element tag is used for creating unordered lists. These are typically rendered as bullet points. The `<ol>` element tag is used for ordered lists. These are rendered as numbered list. The `<li>` element under `<ul>` or `<ol>` defines the list item. Let's add an unordered list to see how this works.

```
<section>

<h1> Welcome to HTML </h1>

<p>This guide will introduce you to the world of HTML. It will teach you how
to read, write, and comprehend HTML. In the process, it will also help you
think about code, software and other technical things that are concealed from
our day-to-day interactions with technical interfaces. These thoughts will
```

```
hopefully lead to more thinking about art, methods, and how we go about
studying and doing things in our very media saturated environments. At the
end of this guide, you will have designed a static web page.</p><br>

<div>

<img src="html_3.png"alt="banner_img"/></img>

<br>

</div>

<p> The guide is designed to help all levels of expertise. This is to say,
you can use this if this is the first time you have heard of HTML, have a
rough idea and would like to know more, need a refresher, are an absolute
expert, or just curious about what’s going on here. You only need a laptop or
desktop computer, basic typing skills, and a text editor like notepad or
WordPad (available on every computer for free!) to get started.<br><br>

The guide also has a long list of resources: <a href=
"https://www.w3schools.com/html/default.asp">tutorials,</a> videos, zines, to
help you further develop these newly acquired HTML skills. This list is in no
way exhaustive or definitive. For everything that is listed here, there’s a
gazillion more just a search or prompt away. This guide just nudges you in
the right direction, so you know what to look for, check if it is accurate,
know how to fix it, and make it work for your own requirements.

Let us get started!</p>

<h2> Tech requirements</h2>

<ul>

<li>Laptop or desktop computer (any operating system)</li>

<li>A .txt editor. Most commonly used operating systems (OS) have a default
text editor application.<br>

It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on
Ubuntu. </li>

<li>Laptop or desktop computer (any operating system)</li>

<li>To host the webpage online, you will need an account on a webhosting
service.</li>

</ul>

</section><br>
```

You can reuse the section element to define additional sections on your page. Make sure you close the tag before you begin a new section.

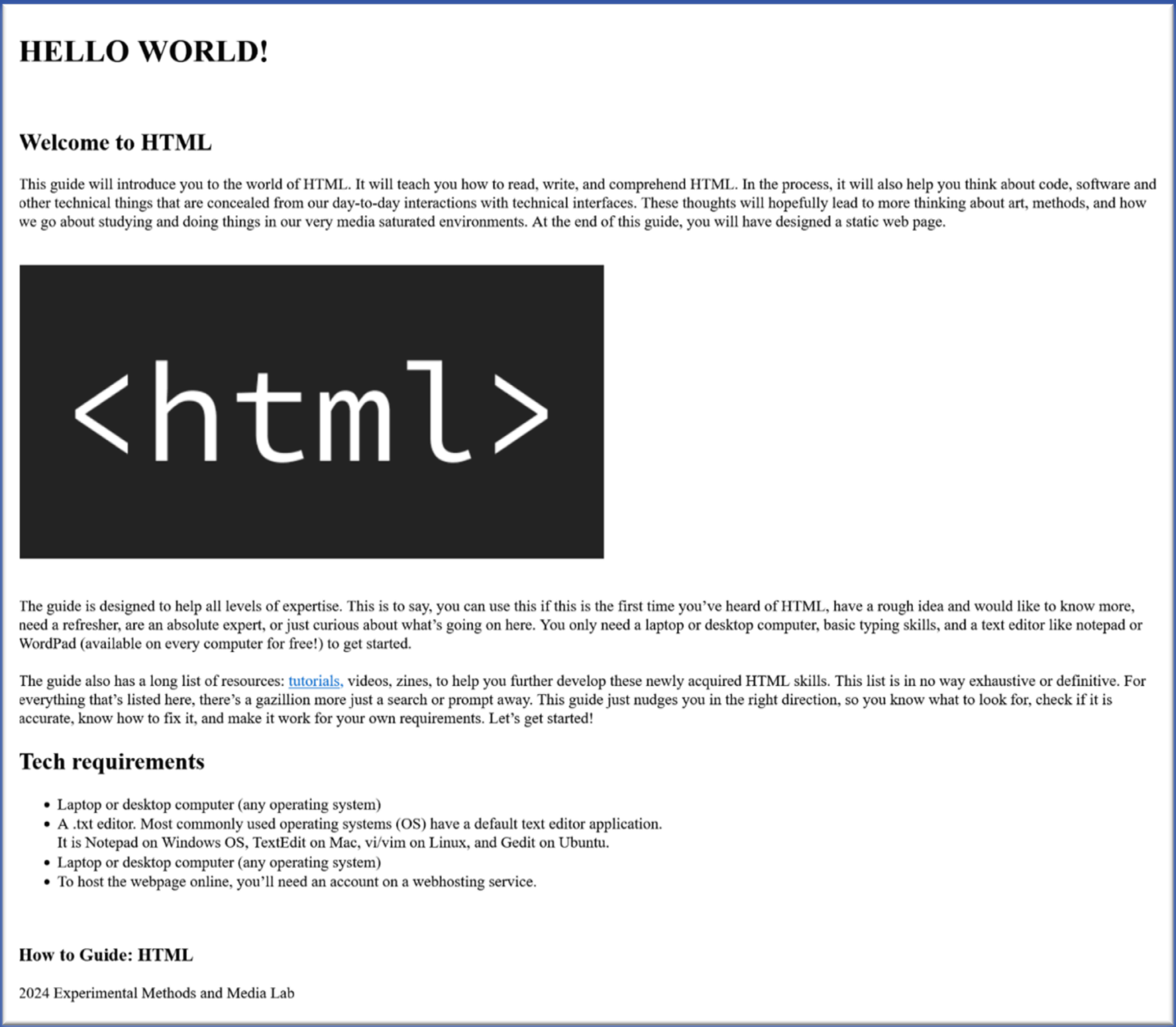
vi. Let's add a footer. A page is incomplete without a footer. We can add a footer at the end of the page using a `<footer>` tag.

```
<footer>

<h3>How to Guide: HTML</h3>

<p>2024 Experimental Methods and Media Lab</p>

  </footer>
```

The page is complete! We can now close the `<body>` and `<html>` tags we opened at the beginning of the page. Save the file and refresh your browser to see changes.

```
<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta name="description" content="A short description of your webpage. It
should not be longer than a sentence or two.">

<title> How_to_Guide </title>

</head>

<body>

<header>

<h1> HELLO WORLD! <br></h1>

</header>

<section>

<h1> Welcome to HTML </h1>

<p>This guide will introduce you to the world of HTML. It will teach you how
to read, write, and comprehend HTML. In the process, it will also help you
think about code, software and other technical things that are concealed from
our day-to-day interactions with technical interfaces. These thoughts will
hopefully lead to more thinking about art, methods, and how we go about
studying and doing things in our very media saturated environments. At the
end of this guide, you will have designed a static web page.

 <br></p>
```

```
<div>

<img src="html_3.png"alt="banner_img"/></img>

<br>

</div>

<p> The guide is designed to help all levels of expertise. This is to say,
you can use this if this is the first time you have heard of HTML, have a
rough idea and would like to know more, need a refresher, are an absolute
expert, or just curious about what’s going on here. You only need a laptop or
desktop computer, basic typing skills, and a text editor like notepad or
WordPad (available on every computer for free!) to get started.<br><br>

The guide also has a long list of resources: <a href=
"https://www.w3schools.com/html/default.asp">tutorials,</a> videos, zines, to
help you further develop these newly acquired HTML skills. This list is in no
way exhaustive or definitive. For everything that is listed here, there’s a
gazillion more just a search or prompt away. This guide just nudges you in
the right direction, so you know what to look for, check if it is accurate,
know how to fix it, and make it work for your own requirements.

Let us get started!</p>

<h2> Tech requirements</h2>

<ul>

<li>Laptop or desktop computer (any operating system)</li>

<li>A .txt editor. Most commonly used operating systems (OS) have a default
text editor application.<br>

It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on
Ubuntu. </li>

<li>Laptop or desktop computer (any operating system)</li>

<li>To host the webpage online, you will need an account on a webhosting
service.</li>

</ul>

</section><br>

<footer>

<h3>How to Guide: HTML</h3>

<p>2024 Experimental Methods and Media Lab</p>

  </footer>

</body>

</html>
```



HELLO WORLD!

Welcome to HTML

This guide will introduce you to the world of HTML. It will teach you how to read, write, and comprehend HTML. In the process, it will also help you think about code, software and other technical things that are concealed from our day-to-day interactions with technical interfaces. These thoughts will hopefully lead to more thinking about art, methods, and how we go about studying and doing things in our very media saturated environments. At the end of this guide, you will have designed a static web page.

The guide is designed to help all levels of expertise. This is to say, you can use this if this is the first time you've heard of HTML, have a rough idea and would like to know more, need a refresher, are an absolute expert, or just curious about what's going on here. You only need a laptop or desktop computer, basic typing skills, and a text editor like notepad or WordPad (available on every computer for free!) to get started.

The guide also has a long list of resources: tutorials, videos, zines, to help you further develop these newly acquired HTML skills. This list is in no way exhaustive or definitive. For everything that's listed here, there's a gazillion more just a search or prompt away. This guide just nudges you in the right direction, so you know what to look for, check if it is accurate, know how to fix it, and make it work for your own requirements. Let's get started!

Tech requirements

- Laptop or desktop computer (any operating system)
- A .txt editor. Most commonly used operating systems (OS) have a default text editor application.
  It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on Ubuntu.
- Laptop or desktop computer (any operating system)
- To host the webpage online, you'll need an account on a webhosting service.

How to Guide: HTML

2024 Experimental Methods and Media Lab

Image 3: Final HTML webpage

5. Adding more pages

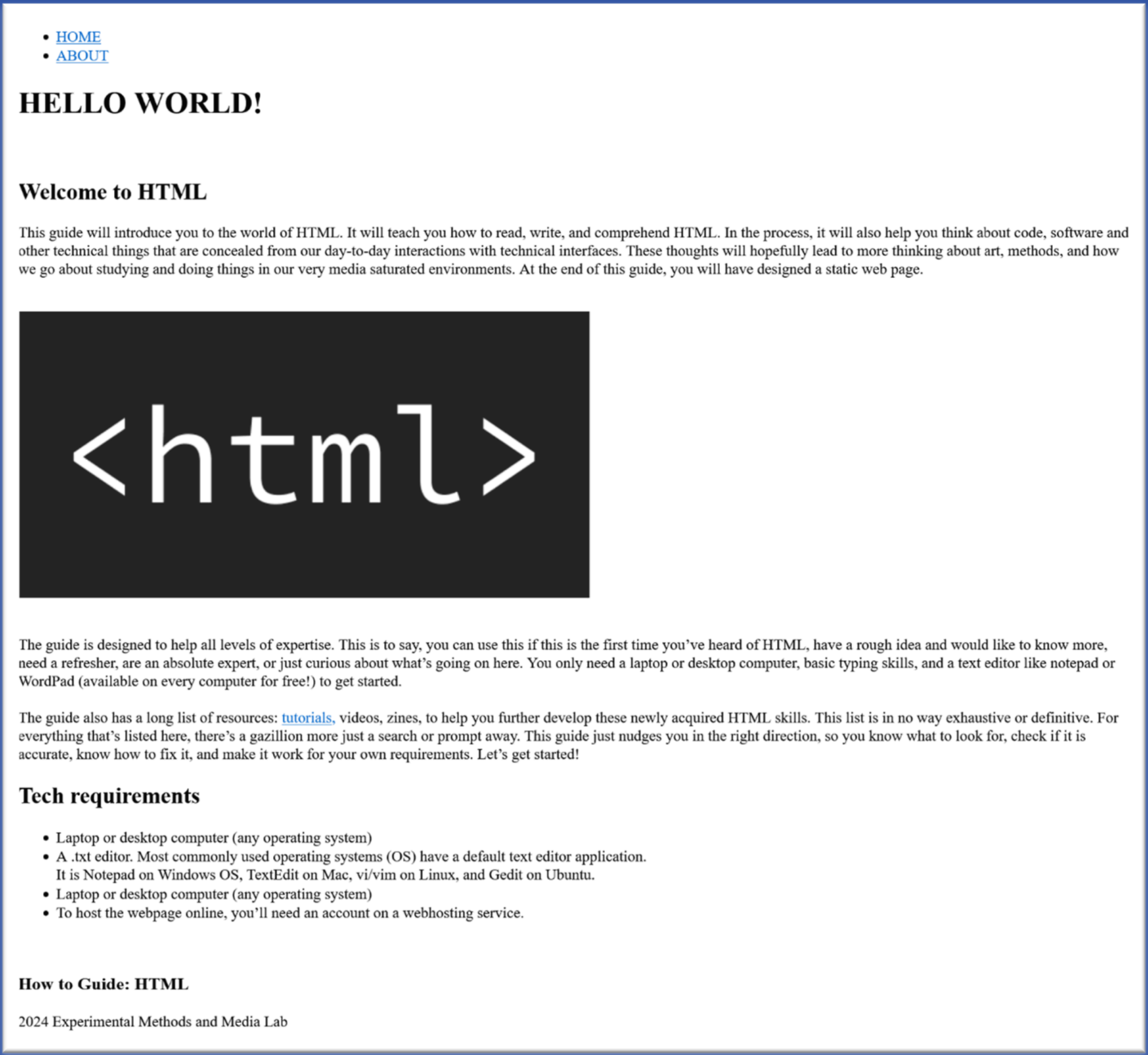
Adding more pages to a website is easy once the index.html page is ready. Let's add an About page to this website to see how it's done.

First, we'll add a navigation to our index.html page. For this, we'll go back to the `<header>` element and add the navigation element <nav> before the heading tag. Then we'll use `<ul>`, `<li><a>` tags to create a menu for the website.

Note that most coding programs automatically intend each line of code. This is done to aid clarity, especially when there are many nested elements. However, it is not necessary to use indentation. Formatting the HTML script has no impact on the final webpage. Web browsers will only read styling and formatting preferences applied through CSS.

```
<body>
<header>
    <nav>
        <ul>
      <li><a href="INDEX.HTML">HOME</a></li>
      <li><a href="ABOUT.HTML">ABOUT</a></li>
      </ul>
    </nav>
<h1> HELLO WORLD! </h1>
</header><br>
```

Save changes. Now refresh the browser to check if the changes are visible on your webpage.



- HOME
- ABOUT

HELLO WORLD!

Welcome to HTML

This guide will introduce you to the world of HTML. It will teach you how to read, write, and comprehend HTML. In the process, it will also help you think about code, software and other technical things that are concealed from our day-to-day interactions with technical interfaces. These thoughts will hopefully lead to more thinking about art, methods, and how we go about studying and doing things in our very media saturated environments. At the end of this guide, you will have designed a static web page.

The guide is designed to help all levels of expertise. This is to say, you can use this if this is the first time you've heard of HTML, have a rough idea and would like to know more, need a refresher, are an absolute expert, or just curious about what's going on here. You only need a laptop or desktop computer, basic typing skills, and a text editor like notepad or WordPad (available on every computer for free!) to get started.

The guide also has a long list of resources: tutorials, videos, zines, to help you further develop these newly acquired HTML skills. This list is in no way exhaustive or definitive. For everything that's listed here, there's a gazillion more just a search or prompt away. This guide just nudges you in the right direction, so you know what to look for, check if it is accurate, know how to fix it, and make it work for your own requirements. Let's get started!

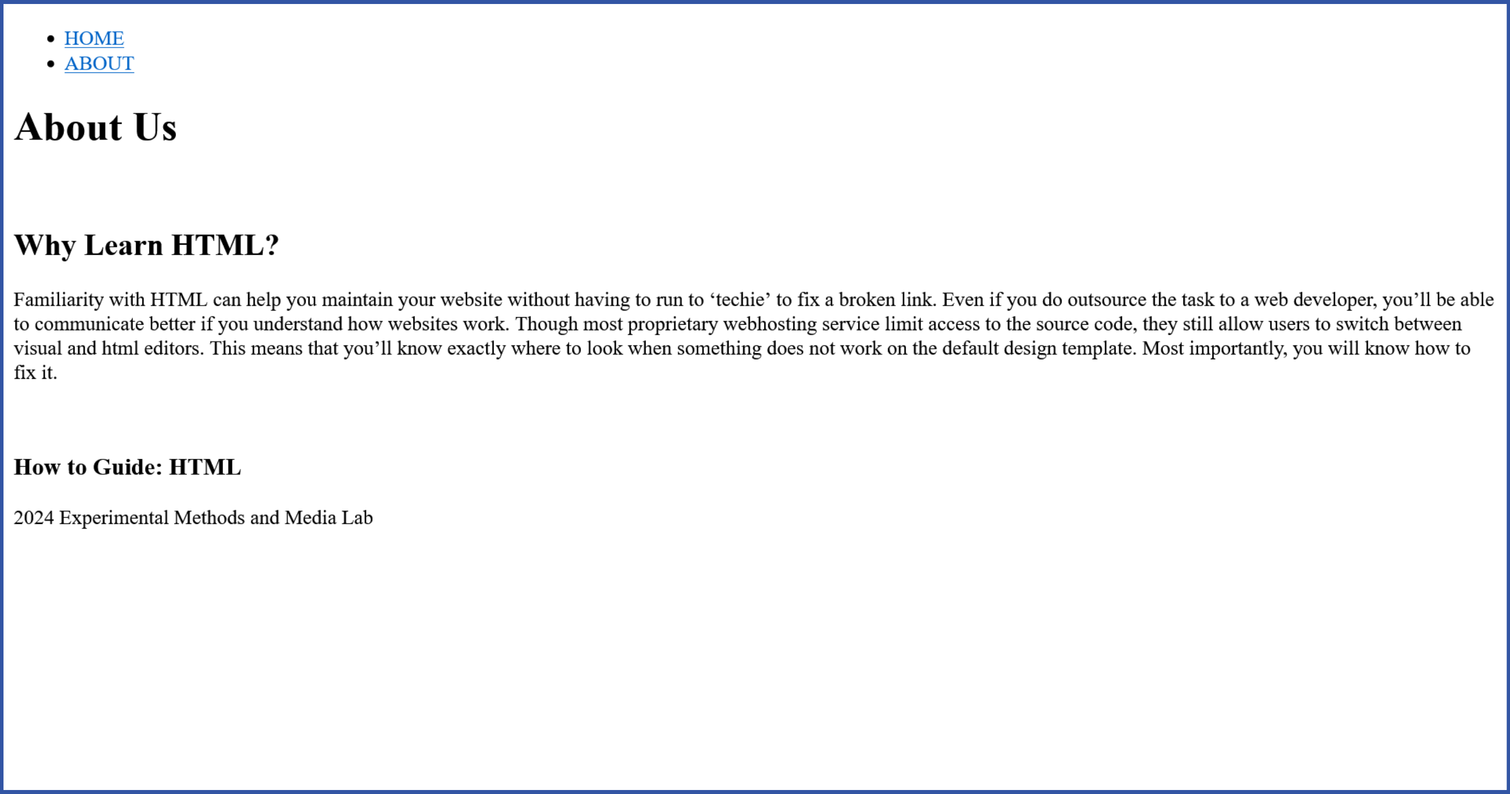
Tech requirements

- Laptop or desktop computer (any operating system)
- A .txt editor. Most commonly used operating systems (OS) have a default text editor application.
  It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on Ubuntu.
- Laptop or desktop computer (any operating system)
- To host the webpage online, you'll need an account on a webhosting service.

How to Guide: HTML

2024 Experimental Methods and Media Lab

Image 4: Webpage with navigation



- HOME
- ABOUT

About Us

Why Learn HTML?

Familiarity with HTML can help you maintain your website without having to run to ‘techie’ to fix a broken link. Even if you do outsource the task to a web developer, you’ll be able to communicate better if you understand how websites work. Though most proprietary webhosting service limit access to the source code, they still allow users to switch between visual and html editors. This means that you’ll know exactly where to look when something does not work on the default design template. Most importantly, you will know how to fix it.

How to Guide: HTML

2024 Experimental Methods and Media Lab

Image 5: Screenshot of the About page

The combination of the <li> tag and <a> tag gives us a list of menu items that are linked to the specific html pages within the website. The home page is linked to the 'index.html' page and the 'About' page is linked to the 'About.html' page, which is what we create next.

Select and copy all the HTML on 'index.html' and paste it into a new .txt file.

Save the new file as 'About.html' in the root folder (where you've saved your index.html file).

Now change the content between html elements to add information relevant to the about us page.

```
<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta name="description" content="A short description of your webpage. It
should not be longer than a sentence or two.">

<title> HTML_Guide </title>

</head>

<body>

<header>

    <nav>

        <ul>

      <li><a href="index.html">HOME</a></li>

      <li><a href="About.html">ABOUT</a></li>

      </ul>

    </nav>

<h1> About Us</h1>

</header><br>

<section>

<h1> Why Learn HTML? </h1>

<p>Familiarity with HTML can help you maintain your website without having to
run to 'techie' to fix a broken link. Even if you do outsource the task to a
```

```
how to fix it.</p>
</section><br>
<footer>
<h3>How to Guide: HTML</h3>
<p>2024 Experimental Methods and Media Lab</p>
  </footer>
</body>
</html>
```

Save all your changes and close the file. Open the page in a web browser.

Click on 'Home' in the menu bar. This should take you to the home page.

Once on the home page, click 'About' in the menu. This should take you back to the About us page.

If everything works, you have successfully created a website! You can add more pages by copy-pasting the HTML into new .html files and editing content on the new pages. Remember to update the navigation on all the pages as you create new pages.

With HTML done, we can now look at styling the pages with CSS.

PART II: CSS

WHAT IS CSS?

As we can see from what we have created so far in this tutorial, plain HTML is, well, plain. It simply makes content legible on a page. The stylistic presentation of this content is accomplished through Cascading Style Sheets (CSS). CSS is a language used to define style sheets for markup languages such as HTML. The word cascading implies that when a style is defined for a parent HTML element, it is automatically applied to all the elements nested under the element[9].

Background colors, fonts, layout text alignment, and other such elements that make a website look complete are defined through CSS. Every website is a combination of HTML and CSS and more, depending on its features and complexity. This makes CSS indispensable to web development. The separation of content and formatting also makes it possible to customize the website for different devices and screens. With the help of specific CSS functions a website can be made responsive, i.e., usable on mobile devices like tablets and phones, and accessible through screen-readers and tactile devices.

9 What is CSS: https://www.simplehtmlguide.com/whatiscss.php

HOW TO USE CSS?

There are three ways to add CSS to a website.

- First, defining the style alongside HTML elements by setting a style attribute within the tag. This is known as inline CSS. The style in this case only applies to the tag to which it has been added. For example,

```
<h1 style="color:white;"> Title </h1>
```

  will only change the color of the content within this `<h1>` to white.

- Second, using the `<style>` tag within the `<head>` tag in HTML. We'll use this approach for styling our website. The CSS in this case applies to the entire page. For example,

```
<head>
<style>
h1{
color: white;
}
</style>
</head>
```

  will change the color of all the `<h1>` tags on the page to white.

- Third, linking an external CSS file to the HTML file. CSS files, like HTML, can be created as plain text files using a .css extension. The external CSS file can be linked to the HTML file using a `<link ...>` tag under the <head> tag.

  For example:

```
<head>
<link rel="stylesheet" href="style.css">
</head>
```

will apply the style sheet defined in the style.css document to the entire website.

A basic style sheet can be used to define style for html tags like `<body>` or `<h1>`. In this case, styles for the `<body>` tag will apply to everything under that tag. To style different parts of the body, CSS can be applied to the different Components such as the `<header>` or `<footer>` tags and the elements nested under them. We'll use this approach when styling our pages.

As you'll see, CSS follows a slightly different syntax; however, it shares some conventions with HTML such as the logic of opening and closing tags. CSS commands are included within these { } parenthesis. The CSS will not work as expected if the closing } is missing. The CSS property being defined is followed by a colon (:)and each line of CSS ends with a semi-colon (;). Missing these or using them incorrectly will also break the CSS. Most coding programs will alert you to such errors while you write your CSS, but it's always good to look out for them as you progress, especially if you are using notepad or similar applications.

STYLING A PAGE

Open the index.html txt file.

In the page structure, add the element tag `<style>` under the element `<head>`. Now everything under the `<style>` tag is read as CSS instructions for the website by the browsers.

```
<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta name="description" content="A short description of your webpage. It
should not be longer than a sentence or two.">
```

```
<title> HTML_Guide </title>
<style>
```

i. Define the border and size for the entire page.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="A short description of your webpage. It
should not be longer than a sentence or two.">
<title> Page Name </title>
<style>
*{
  padding: 0;
   margin: 0;
   box-sizing:"border-box";
}
```

The CSS defined here sets the padding, border, and margin for the entire page. These elements make up the 'box model', which refers to design and layout in CSS. In this model, every HTML element is wrapped by a box.

The CSS attribute 'padding' is the first box that wraps around the content. This attribute is transparent and is used to clear the area around the content. Padding is applied at the top, right, bottom, and left of the content. Different values can be set for each side of the box.

'Border' is the next box and it goes around padding. This is not a transparent attribute, which means that its thickness and color can be defined in CSS and will be visible on the website.

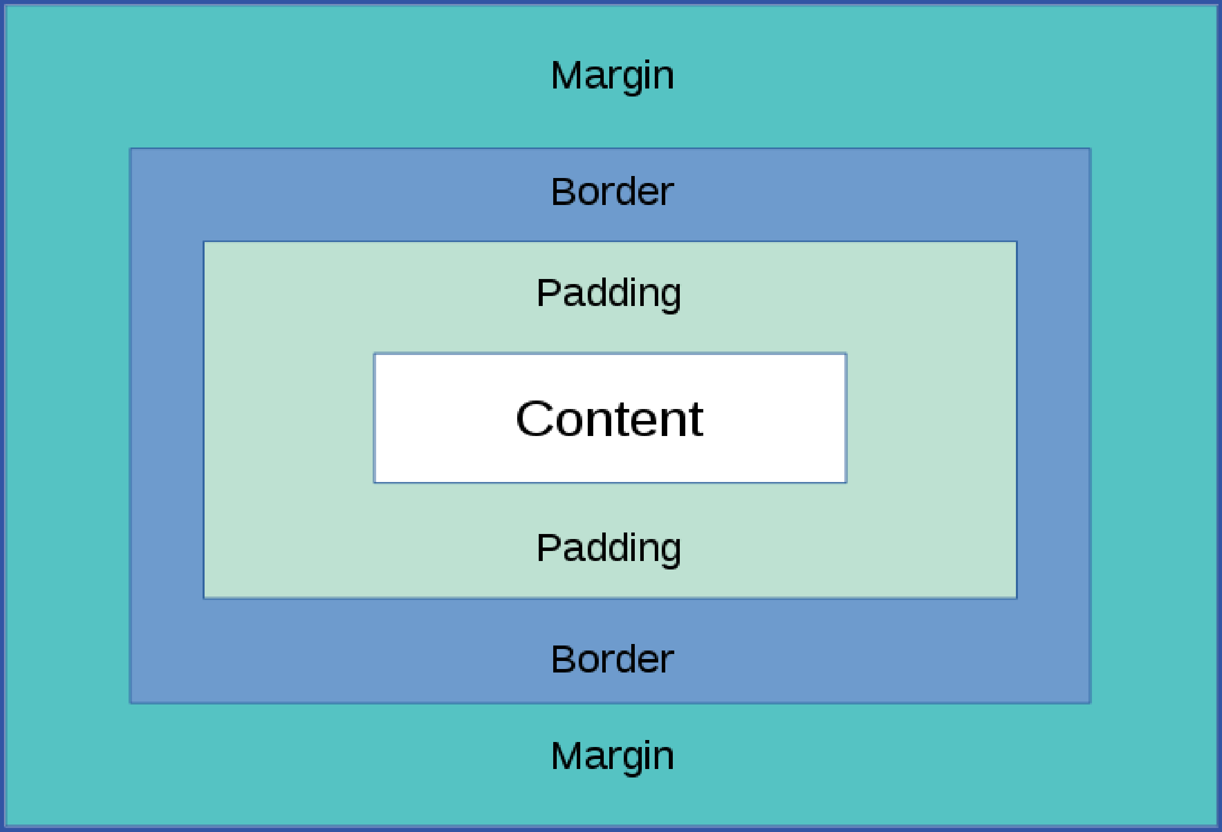


Image 6: Box model of CSS.

Source: Charles Calvert, CC BY-SA 4.0, via Wikimedia Commons

‘Margin’ forms the final and outer most layer in the box model. It is also a transparent attribute and is used to clear the area around the border. Like padding, different values can be set for each side of the margin in CSS.

The ‘box-sizing: border-box’; property in CSS is used to ensure that padding and border are part of an element’s total width and page. This ensures that the element has a uniform size.

We have used all these CSS properties to make sure our entire webpage has no extra space around it in the form of padding or margins and this is included in the total length and width of the page. Now all content on this page will be uniformly aligned and ready to be styled individually.

ii. Define the style for `<body>`:

```
body {
  background: black;
  font-size: 22px;
  line-height: 32px;
  color: white;
  margin: 0;
  padding: 0;
  word-wrap: break-word !important;
  font-family: sans-serif;
}
```

Here, we have defined CSS properties for the ‘body’ element. The styles defined here will apply to all elements under the <body> tag, unless we define different styles for each of these elements. The background, font-size, line-height, and color properties set the default values for each of these properties. The ‘Word-wrap: break-word’ property is used to automatically break and wrap long words on the next line.

The ‘font-family’ property is used to specify the font for the element. This property uses specific font-family names such as ‘arial’ or ‘Helvetica’ as well as a generic-family names such as serif and sans-serif. While either can be used to define the font style in CSS, specific font-family names should be followed by a generic name so that web browsers can select a similar font from the generic family if the font specified in the CSS is unavailable. Each of these values should be separated by a comma in the CSS.

E.g.,

```
body {
font-family: "Times New Roman", Times, serif;
}
```

iii. Style the header:

```
header {
  background: white;
    padding-bottom: 20px;
}
```

HTML elements such as headings <h1> or paragraphs <p> are used in different partsof the body of a page. If we use these tags as our primary CSS category, then all <p> or <h1> elements on the website will have the same style. This may not be ideal if we want to style each section differently. To style elements nested under specific elements, we need to specify that the style being defined should only be applied to elements under a specific category.

For example, to style elements within the <header> we begin the CSS command with the parent element category i.e., “header” followed by the specific nested elements that need to be styled. Then we specify the style.

CSS PROPERTIES

width: 0px;	Width of HTML element
height: 100%;	Height of HTML element - e.g., 20px | 100%
margin: 5px;	Margin - space around an element, or distance between two elements
margin-top: 1px;	Top Margin. Also try -bottom: -left: or -right:
padding: 5px;	Padding - distance between an element's contents and its border
padding-top: 1px;	Top Padding. Also try -bottom: -left: or -right
text-align: left;	Horizontal Alignment - left | center | right
text-decoration: underline;	Text Decorations - e.g., none | underline | line-through
font-family: fontname;	Font Face (Typeface) - e.g., Verdana, Arial, Helvetica
font-size: 16pt;	Font Size or Height - e.g., 12pt | 15px
font-weight: bold;	Font Weight (Boldness) - e.g., bold | normal | 200
color: red;	Element Color - e.g., red | #FF0000
background-color: white;	Background Color of element
background-image: url();	Background Color of element
border-color: yellow;	Border Color of element
border: 1px solid blue;	Width, style and color of border defined together

Source: https://www.simplehtmlguide.com/cheatsheet.php

iv. Let's style the elements within the header:

```
header nav {
  display: inline;
  background: transparent;
  background-size: cover;
  background-position: center;
  text-align: center;
  padding-top: 5px;
  padding-bottom: 10px;
}
header nav ul {
    display: inline-flex;
  list-style: none;
  color: black;
  font-weight: bold;
  text-align: center;
}
header nav ul li {
    width: 100px;
  margin: 10px;
  padding: 5px;
  text-align: center;
}
```

```
header nav ul li a {
text-decoration: underline;
  color: black;
  display: inline-block;
  padding: 5px;
}
header h1 {
    font-size: 100px;
    font-weight: bold;
    text-align: center;
    line-height: 32px;
    color: black;
    padding-bottom:75px;
    padding-top: 50px;
}
```

The ‘display’ property in CSS is used to define how the box element, i.e., the content, is treated on the page. It determines whether a box is an inline element, a block, or a flexible box. The values selected for this property determine the layout for the elements nested under the element being styled.

v. Style the main content section of the website and elements under it:

```
section {
    display: inline-block;
    background: transparent;
    background-size: cover;
    background-position: center;
    color: rgb(218, 218, 218);
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 10px;
    margin: 10px;
}

section h1 {
    text-align: left;
    font-size: 40px;
    font-weight: 900;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 15px;
    padding-bottom: 15px;
    margin: 10px;
}
section p {
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section p a {
  color: rgb(136, 136, 244);
}

section div {
    text-align: center;
    padding-top: 5px;
    padding-bottom: 5px;
}
section h2 {
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    margin: 10px;
}
section ul{
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-bottom: 20px;
    margin: 10px;
}
section ul li{
    text-align: left;
```

```
        font-size: 20px;
        font-weight: 500;
        padding-left: 30px;
        padding-right: 20px;
        padding-top: 10px;
        padding-bottom: 20px;
        margin: 10px;
    }
```

Though we have used RGB values for colors in this CSS, there are other ways in which color can be added as well. Hex codes, CMYK, or generic names of colors like 'red' or 'blue' can be used to set color properties in CSS. A simple web search can help find applications and tools that generate color codes for HTML and CSS. This tutorial used a color picker from: https://htmlcolorcodes.com/color-picker/

vi. Style the footer and elements under it:

```
    footer {
        background: transparent;
        background-size: contain;
        padding-top: 5px;
        border-top: 1px solid rgb(238, 238, 238);
        border-bottom: none;
    }
    footer h3 {
        text-align: center;
        font-size: 10px;
        font-weight: light;
        color:  white;
        padding-left: 20px;
        padding-right: 20px;
        padding-top: 5px;
        padding-bottom: 1px;
    }
    footer p {
        text-align: center;
        font-size: 10px;
        font-weight: light;
        color:  white;
        padding-left: 20px;
        padding-right: 20px;
        padding-top: 5px;
        padding-bottom: 1px;
    }
```

vii. Finally, we use a media query command to make the website responsive i.e., automatically adjust to different screen-sizes. Here we are telling the servers that styles apply to all media devices up to a max-width of 768 pixels. Then we specify instructions for screens that are smaller than this size. All CSS within the media query will only apply to screens that are smaller than 768 px.

```
@media all and (max-width : 768px) {
header {
overflow: hidden;
}
header nav {
    margin: 0;
}
header nav ul {
    float: none;
}
header h1{
    font-size: 50px;
    font-weight: bold;
    text-align: center;
    line-height: 16px;
    color: black;
    padding-bottom:35px;
    padding-top: 25px;
}
section {
    display: inline-block;
    background: transparent;
    background-size: cover;
    background-position: center;
    color: rgb(218, 218, 218);
    padding-left: 10px;
    padding-right: 10px;
    padding-top: 5px;
    padding-bottom: 5px;
    margin: 5px;
}
section h1 {
    text-align: left;
    font-size: 25px;
    font-weight: 900;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 15px;
    padding-bottom: 15px;
    margin: 5px;
}
section p {
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section p a {
  color: rgb(136, 136, 244);
  text-decoration: underline;
}

section div {
    text-align: center;
    padding-top: 5px;
    padding-bottom: 5px;
}
section div img{
    display: flex;
    width: 375px;
    height: auto;
}
section h2 {
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
```

```
section ul{
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section ul li{
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
}
</style>
```

Now, we have all the essential information so we can close the `<style>` tag and save. Your `<style>` element should look like this:

```
<style>
*{
  padding: 0;
   margin: 0;
   box-sizing:"border-box";
}
body {
  background:black;
  font-size: 22px;
  line-height: 32px;
  color: white;
  margin: 0;
  padding: 0;
  word-wrap: break-word !important;
  font-family: sans-serif;
}

header {
    background: white;
    padding-bottom: 20px;
}

header nav {
    display: inline;
  background: transparent;
  background-size: cover;
  background-position: center;
  text-align: center;
  padding-top: 5px;
  padding-bottom: 10px;
}
header nav ul {
    display: inline-flex;
  list-style: none;
  color: black;
  font-weight: bold;
  text-align: center;
}
```

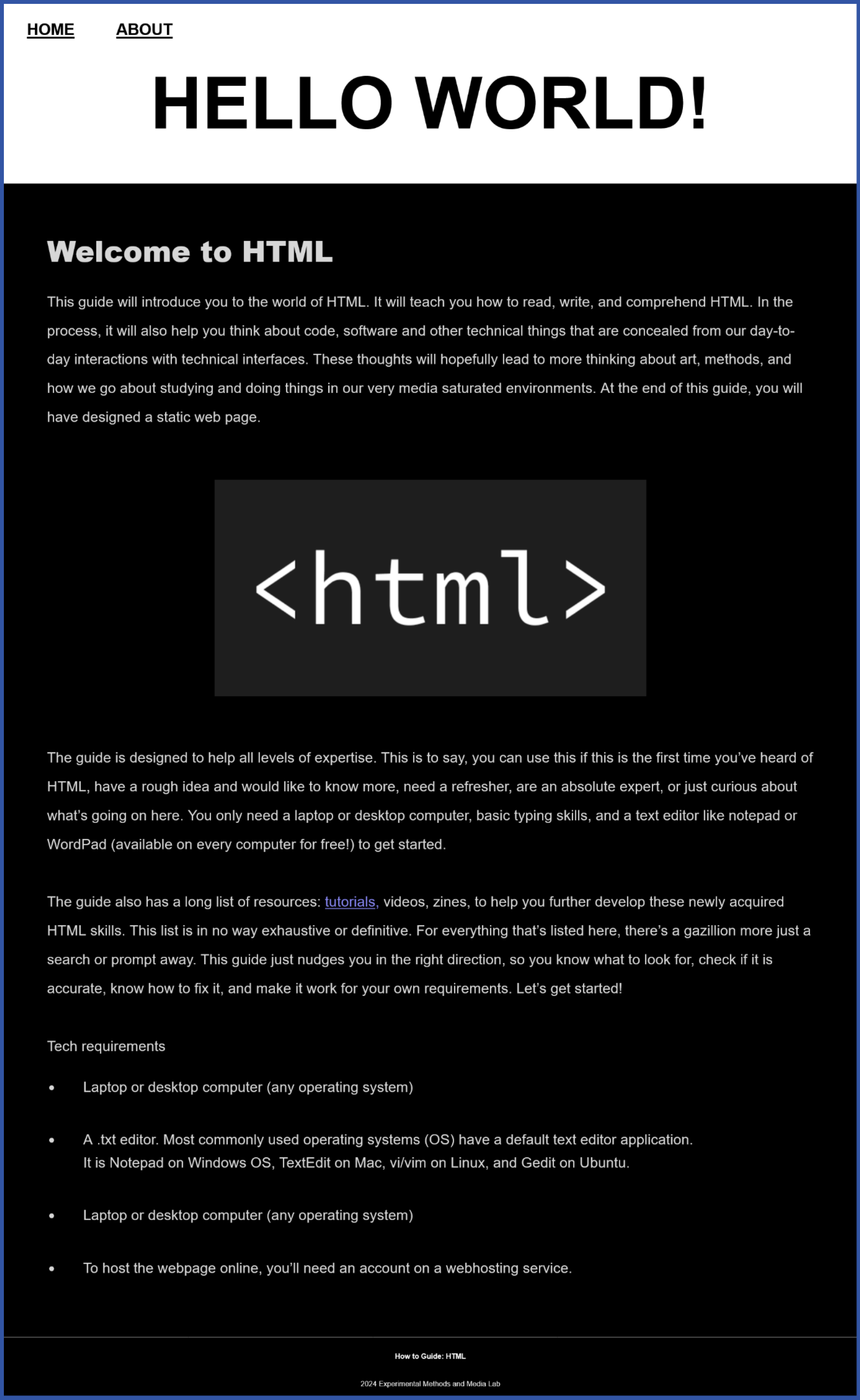
```
header nav ul li {
    width: 100px;
  margin: 10px;
  padding: 5px;
  text-align: center;
}
header nav ul li a {
text-decoration: underline;
  color: black;
  display: inline-block;
  padding: 5px;
}
header h1 {
    font-size: 100px;
    font-weight: bold;
    text-align: center;
    line-height: 32px;
    color: black;
    padding-bottom:75px;
    padding-top: 50px;
}

section {
    display: inline-block;
    background: transparent;
    background-size: cover;
    background-position: center;
    color: rgb(218, 218, 218);
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 10px;
    margin: 10px;
```

```
}

section h1 {
    text-align: left;
    font-size: 40px;
    font-weight: 900;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 15px;
    padding-bottom: 15px;
    margin: 10px;
}
section p {
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section p a {
  color: rgb(136, 136, 244);
}

section div {
    text-align: center;
    padding-top: 5px;
    padding-bottom: 5px;
}
section h2 {
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    margin: 10px;
}
section ul{
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-bottom: 20px;
    margin: 10px;
}
section ul li{
    text-align: left;
    font-size: 20px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
footer {
    background: transparent;
    background-size: contain;
    padding-top: 5px;
    border-top: 1px solid rgb(238, 238, 238);
    border-bottom: none;
}
footer h3 {
    text-align: center;
    font-size: 10px;
    font-weight: light;
    color:  white;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 5px;
    padding-bottom: 1px;
}
footer p {
    text-align: center;
    font-size: 10px;
    font-weight: light;
    color:  white;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 5px;
    padding-bottom: 1px;
}

@media all and (max-width : 768px) {
header {
overflow: hidden;
}
header nav {
    margin: 0;
}
header nav ul {
    float: none;
}
```

```
header h1{
    font-size: 50px;
    font-weight: bold;
    text-align: center;
    line-height: 16px;
    color: black;
    padding-bottom:35px;
    padding-top: 25px;
}
section {
    display: inline-block;
    background: transparent;
    background-size: cover;
    background-position: center;
    color: rgb(218, 218, 218);
    padding-left: 10px;
    padding-right: 10px;
    padding-top: 5px;
    padding-bottom: 5px;
    margin: 5px;
}
section h1 {
    text-align: left;
    font-size: 25px;
    font-weight: 900;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 15px;
    padding-bottom: 15px;
    margin: 5px;
}
section p {
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    line-height: 200%;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section p a {
  color: rgb(136, 136, 244);
  text-decoration: underline;
}

section div {
    text-align: center;
    padding-top: 5px;
    padding-bottom: 5px;
}
section div img{
    display: flex;
    width: 375px;
    height: auto;
}
section h2 {
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 20px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
```

```
}
section ul{
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
section ul li{
    text-align: left;
    font-size: 15px;
    font-weight: 500;
    padding-left: 30px;
    padding-right: 20px;
    padding-top: 10px;
    padding-bottom: 20px;
    margin: 10px;
}
}
</style>
```

Refresh the browser to check changes on your homepage. If everything works, copy everything within the `<style>` tags including the tags themselves and paste it under the `<head>` tag in all the .html pages you've created for the website.



HOME ABOUT

HELLO WORLD!

Welcome to HTML

This guide will introduce you to the world of HTML. It will teach you how to read, write, and comprehend HTML. In the process, it will also help you think about code, software and other technical things that are concealed from our day-to-day interactions with technical interfaces. These thoughts will hopefully lead to more thinking about art, methods, and how we go about studying and doing things in our very media saturated environments. At the end of this guide, you will have designed a static web page.

<html>

The guide is designed to help all levels of expertise. This is to say, you can use this if this is the first time you've heard of HTML, have a rough idea and would like to know more, need a refresher, are an absolute expert, or just curious about what's going on here. You only need a laptop or desktop computer, basic typing skills, and a text editor like notepad or WordPad (available on every computer for free!) to get started.

The guide also has a long list of resources: tutorials, videos, zines, to help you further develop these newly acquired HTML skills. This list is in no way exhaustive or definitive. For everything that's listed here, there's a gazillion more just a search or prompt away. This guide just nudges you in the right direction, so you know what to look for, check if it is accurate, know how to fix it, and make it work for your own requirements. Let's get started!

Tech requirements

- Laptop or desktop computer (any operating system)
- A .txt editor. Most commonly used operating systems (OS) have a default text editor application.
  It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on Ubuntu.
- Laptop or desktop computer (any operating system)
- To host the webpage online, you'll need an account on a webhosting service.

How to Guide: HTML

2024 Experimental Methods and Media Lab

Image 7: Homepage of the website after adding CSS

PART III: HELLO WORLD!

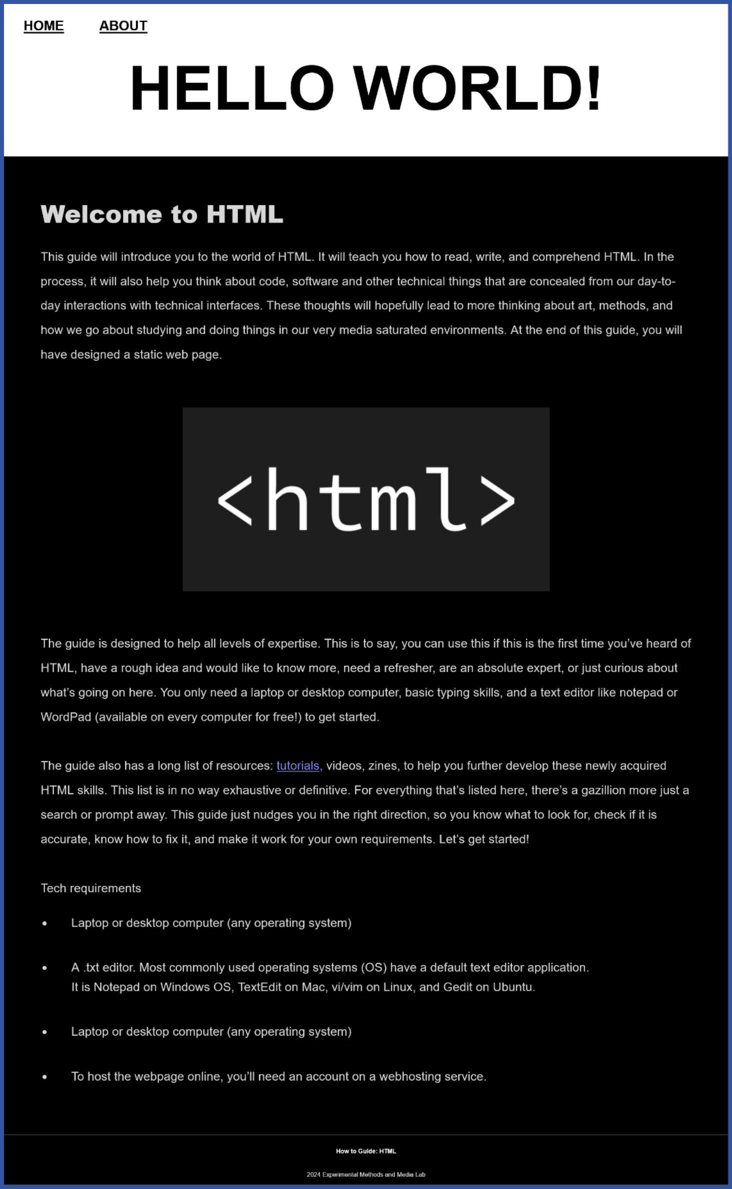
GOING LIVE!

The final step involves moving this website from your computer to the world wide web. To do this you need to set up an account on a webhosting service. We are using Neocities (https://neocities.org/) for this guide, but you are welcome to explore others or use a preferred hosting service.

- Create an account on Neocities
- Neocities has a default index.html page for websites.
- Select all (cntrl+a) the contents on the index.html file you've been working on so far. Copy and paste it on the Neocities' default index.html page. Save!
- Upload additional webpages in the same folder on Neocities

But we are not done yet. We've moved the source code to this new folder on Neocities. This changes the file paths for the images in the HTML. To make everything work correctly, upload all the images you've used on this webpage on Neocities, making sure they are in the same folder as the index.html file. Save!

Finally, click on the view website option to see your website. Now you can copy the URL from the address bar and share it with anyone who wants to see your website.


HOME ABOUT

HELLO WORLD!

Welcome to HTML

This guide will introduce you to the world of HTML. It will teach you how to read, write, and comprehend HTML. In the process, it will also help you think about code, software and other technical things that are concealed from our day-to-day interactions with technical interfaces. These thoughts will hopefully lead to more thinking about art, methods, and how we go about studying and doing things in our very media saturated environments. At the end of this guide, you will have designed a static web page.

<html>

The guide is designed to help all levels of expertise. This is to say, you can use this if this is the first time you've heard of HTML, have a rough idea and would like to know more, need a refresher, are an absolute expert, or just curious about what's going on here. You only need a laptop or desktop computer, basic typing skills, and a text editor like notepad or WordPad (available on every computer for free!) to get started.

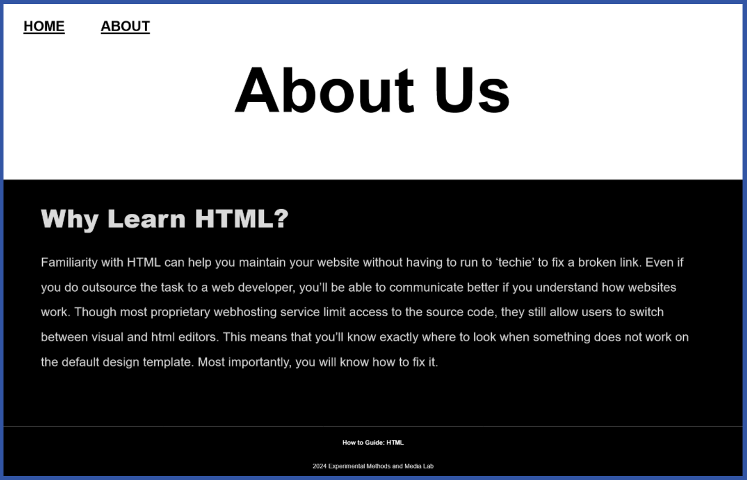
The guide also has a long list of resources: tutorials, videos, zines, to help you further develop these newly acquired HTML skills. This list is in no way exhaustive or definitive. For everything that's listed here, there's a gazillion more just a search or prompt away. This guide just nudges you in the right direction, so you know what to look for, check if it is accurate, know how to fix it, and make it work for your own requirements. Let's get started!

Tech requirements

- Laptop or desktop computer (any operating system)
- A .txt editor. Most commonly used operating systems (OS) have a default text editor application. It is Notepad on Windows OS, TextEdit on Mac, vi/vim on Linux, and Gedit on Ubuntu.
- Laptop or desktop computer (any operating system)
- To host the webpage online, you'll need an account on a webhosting service.

How to Guide: HTML

2024 Experimental Methods and Media Lab


HOME ABOUT

About Us

Why Learn HTML?

Familiarity with HTML can help you maintain your website without having to run to 'techie' to fix a broken link. Even if you do outsource the task to a web developer, you'll be able to communicate better if you understand how websites work. Though most proprietary webhosting service limit access to the source code, they still allow users to switch between visual and html editors. This means that you'll know exactly where to look when something does not work on the default design template. Most importantly, you will know how to fix it.

How to Guide: HTML

2024 Experimental Methods and Media Lab

Image 10: Final website

TROUBLESHOOTING

- Images will not show if the path and file name is incorrect in html. Check file names and path names to ensure you have used them correctly.
- All open tags in HTML and CSS should have a corresponding closing tag.
- Check to ensure all nested elements have been closed.
- In CSS, ensure all attributes have a semi-colon (;) at the end.
- Coding programs like Visual Code editor, Sublime Text or Brackets can quickly point out missing closed tags or other errors.
- If your page gives you a very specific error (like 404 or 501), search the error code to find out the nature of error and then fix it.
- Refer to the resources section for advanced guides and tutorials.

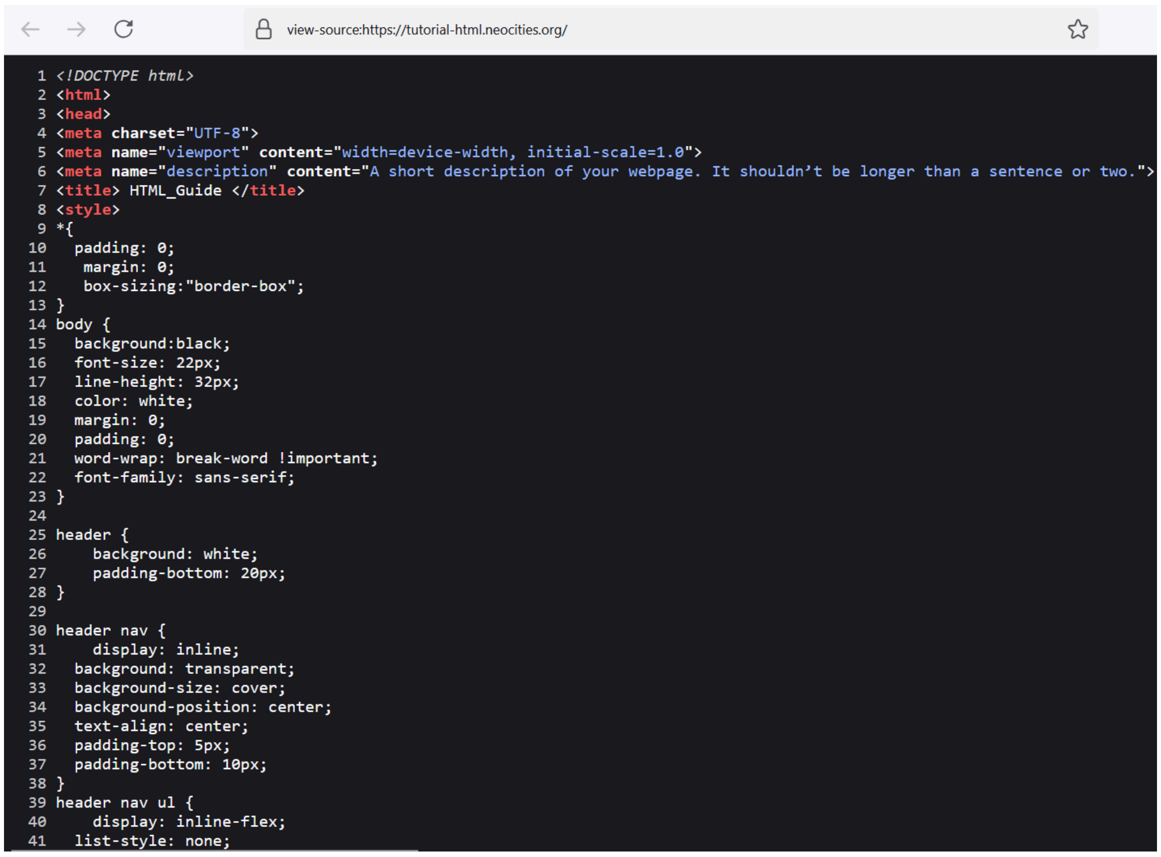
PART IV: RESOURCES

TUTORIAL WEBSITE

The website developed as part of this tutorial can be accessed at: https://tutorial-html.neocities.org/

Right click on the website and select 'View Page Source' . Now you will be able to see the HTML and CSS for this website, the one we've worked on through in this tutorial.

You can select all and copy-paste this page on a .txt file or your preferred programming tool. Simply change the text, images, or any other content within different HTML tags to convert this into your own website. Then follow the 'going live' instructions in this tutorial to publish your website. It is that easy!



```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="A short description of your webpage. It shouldn't be longer than a sentence or two.">
<title> HTML_Guide </title>
<style>
*{
  padding: 0;
   margin: 0;
   box-sizing:"border-box";
}
body {
  background:black;
  font-size: 22px;
  line-height: 32px;
  color: white;
  margin: 0;
  padding: 0;
  word-wrap: break-word !important;
  font-family: sans-serif;
}

header {
    background: white;
    padding-bottom: 20px;
}

header nav {
    display: inline;
  background: transparent;
  background-size: cover;
  background-position: center;
  text-align: center;
  padding-top: 5px;
  padding-bottom: 10px;
}
header nav ul {
    display: inline-flex;
  list-style: none;
```

Image 11: 'View Page Source' screen of the tutorial website

TECH RESOURCES

Sublime Text: https://www.sublimetext.com/3

Brackets: https://brackets.io/?lang=en

Visual Code Studio: https://code.visualstudio.com/

HTML+ CSS color picker: https://htmlcolorcodes.com/color-picker/

Solar powered Media zine: http://lowcarbonmethods.com/zine.html

How to guide: HTML website and source code: https://tutorial-html.neocities.org/

Neocities: https://neocities.org/

HTML TUTORIALS

W3C HTML Tutorial: https://www.w3schools.com/html/default.asp

Structuring the Web with HTML: https://developer.mozilla.org/en-US/docs/Learn/HTML

Simple HTML Guide: https://www.simplehtmlguide.com/

HTML Syntax: https://www.w3schools.com/html/html5_syntax.asp

HTML Cheasheet: https://www.simplehtmlguide.com/cheatsheet.php

Learn HTML (free course): https://www.codecademy.com/learn/learn-html

HTML and CSS Tutorial - Create a Website for Beginners: https://youtu.be/kMT54MPz9oE?si=bfDnS4qcgR8b6TIA

HTML for People: https://www.htmlforpeople.com/

CSS TUTORIALS

W3C CSS Tutorial: https://www.w3schools.com/css/

CSS first steps overview: https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps

CSS Tutorials: https://developer.mozilla.org/en-US/docs/Web/CSS/Tutorials

Hell Yes! CSS!: https://wizardzines.com/zines/css/

Learn CSS (free course): https://www.codecademy.com/learn/learn-css

Vectors from Freepik

EMM LAB GUIDE