Intro to web development

For any web development you will use a combination of three programming languages. HTML, CSS, and JavaScript. HTML is used to provide the basic structure, CSS or Cascading Styling Sheet provides the styles, layout, and presentation of the website while JavaScript is used to add advanced functionality and control the behaviour of various elements. We will be using HTML and CSS for our creation today to build a functional and professional looking basic website.

Login into your Replit account and click create new. Search HTML and choose the HTML, CSS, JS auto refresh option. This will automatically create three files. index.html, styles.css, and script.js. The index.html page will have some code already written in it but for our purposes of learning from the beginning we're going to delete all lines of code except for the very first and start fresh. We will need to keep the name as index.html while working on the Replit site.

The first thing we need to know about HTML is that it uses tags such as **<html>** and **<body>.** Tags are always written between <> and almost always come as a set with a closing tag that has an extra forward slash **</html>** or **</body>.** These tags denote new sections of the page, different data entry types, etc. every page will begin with the tag **<!DOCTYPE html>** so when the file is read by the web browser it knows it's a webpage written in html.

After this the initial setup of any new page will be written between the <html> tags. on Replit as soon as you create the opening tag, the closing tag will be automatically created. Write in the **<html>** tag and then hit enter then add in the **<head>** and **<body>** tags as shown below.

```
<!DOCTYPE html>

<html>
  <head>

  </head>

  <body>

  </body>

</html>
```

The indentation we see here is very important and tells us that both the head and body sections of our website are inside of our html section however are not inside of each other as they are on the same indentation.

The **<head>** section of the website is not used for much but there are two important tags we're going to use today. The first is the **<title>** tag. Whatever is written in the title section will show up as the title in the tab of your browser. Add in the **<title>** tag inside the header. Since it's a short section you don't need to hit enter and can keep the closing tag on the same line.

```
<title> SNC2D </title>
```

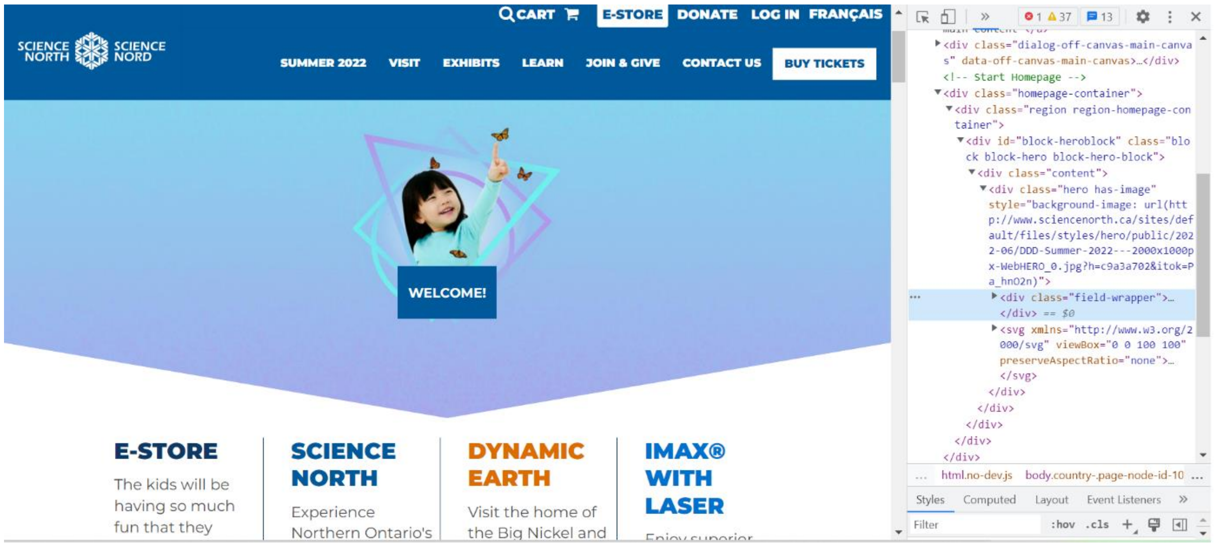
The other tag we'll add into the head section of our webpage is the **<meta>** tag. The Meta tag won't contain any information that will appear on the webpage but instead can contain information about the author and also has important information to help the browser know how to control the dimensions and scaling. We're going to always add in the following line exactly.

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

The meta tag is one of the few that has information contained inside the tag itself rather than between two tags. We will see this a little late on as well with images and to an extent links.

```
<head>
   <title> SNC2D </title>
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
 </head>
```

Next, we'll move on to the body of the webpage. This is where most of our code is going to go. The most important tag to know to help with our organization is **<div>**. Have a look at the image below of the homepage of the Science North website and its code on the right hand side. You can do this exercise yourself on any page by right clicking and inspecting the code.



```
▶<div class="dialog-off-canvas-main-canva
 s" data-off-canvas-main-canvas>…</div>
 <!-- Start Homepage -->
▼<div class="homepage-container">
  ▼<div class="region region-homepage-con
   tainer">
    ▼<div id="block-heroblock" class="blo
     ck block-hero block-hero-block">
      ▼<div class="content">
        ▼<div class="hero has-image"
         style="background-image: url(htt
         p://www.sciencenorth.ca/sites/def
         ault/files/styles/hero/public/202
         2-06/DDD-Summer-2022---2000x1000p
         x-WebHERO_0.jpg?h=c9a3a702&itok=P
         a_hnO2n)">
          ▶<div class="field-wrapper">…
           </div> == $0
          ▶<svg xmlns="http://www.w3.org/2
           000/svg" viewBox="0 0 100 100"
           preserveAspectRatio="none">…
           </svg>
         </div>
       </div>
     </div>
   </div>
 </div>
```

This webpage is divided into three very visible sections in the screen shot. The blue header section, light blue image section, then the info section with the white background. Each division may be subdivided as well to help with the organization of elements which we'll see more of as we go. On the right-hand side, you can see may of the cascading tags in this section are the <div> tag creating many subdivides.

For starters in our body, we're going to add in five sections to build our homepage. You can add in one <div> tag and hit enter to get the auto generated closing tag on a new line then copy and paste the set 4 more times.

```
<body>
   <div>

   </div>

   <div>

   </div>
   .
   .
   .
</body>
```

Each set looks the same, so we'll add in an id for each section to help identify each portion. This will also be important for when we go to add styles to our page. Add the following ids to match the code below.

```
<body>
  <div id = "header">

  </div>

  <div id = "navBar">

  </div>

  <div id = "row1">

  </div>

  <div id = "row2">

  </div>

  <div id = "footer">

  </div>
 </body>
```

Inside the header we'll use the **<h1>** tag to add in a header. There are 6 tags from h1 to h6 dedicated to writing headers, each getting a bit smaller in text size.

```
<div id = "header">
   <h1> Climate Change </h1>
</div>
```

At this point we can hit run on our project, and we'll see a blank white page with the word Homepage written on it. Not very exciting but this is the start of your first ever webpage. Your title will appear in the tab above as well.

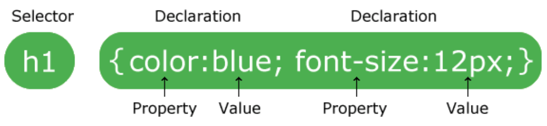
The window to view your webpage is quite small so click the icon in the right corner that opens it in a new page if you'd like.

At this point we will bring in the 2nd file created for us **style.css** so we can start to make our webpage look nicer. Let's begin again by deleting all the code and having a fresh start.

In your CSS we will write many rules. Each will consist of a **selector** to indicate the section of the doc you are styling as well as a **declaration** and **value**. The selector * is the universal selector and applies to all elements on the page.

We'll begin with the universal selector * followed by an opening curly bracket then hit enter to move the closing bracket to a new line. We're going to add in two declarations to apply to all sections of our webpage: **box-sizing: border box;** and **margin: 0;**.

```
* {
  box-sizing: border-box;
  margin: 0;
}
```



Border-box means that the border is included as part of the total width we eventually give each section. This helps with alignment. Setting the margin to 0 ensures all sections are up against each other and there's no weird spaces throughout your webpage as the margin is the space around each section.

Next, we can add in some colour for our header. I like to use the website https://htmlcolorcodes.com/ to find the perfect colours. Looking at a light grey I'm going to choose the colour with the hex value F1F1F1 for my header background colour. We can also add in some style for our h1 section as followed.

```
#header {
   background-color: #F1F1F1;
}

h1 {
   text-align: center;
   padding: 20px;
}
```

The first important thing to note here is the difference between how we've identified the two sections. Any section with an id is identified using the hashtag out front followed by the id name. If we used div instead then all divs would have the same style which we don't want.

In this case I use h1 for the 2nd selector as I plan to have all text written using h1 to follow this style, no hashtag if it's not an id.

Looking deeper at the styles themselves you'll see a box appears next to the colour which is another great alternative to finding that perfect colour. **Background-color** and **text-align** declarations are self explanatory. As you can see any declaration that is two words is separated by a dash and never a space and colour is spelt using the American spelling color.

The **padding** declaration indicates the amount of space to be added around the text itself. As this is the only text in our header section, we're effectively deciding the height of this division.

You'll notice looking at your webpage it's currently not updating to our new styles. This is because we have to go back into our html file and link in this style sheet, so it knows to use these styles with the code we've written. Going back to the **<head>** section of our page, add in the following line.

<link rel="stylesheet" href="style.css">

Now that our style sheet is linked to our code your webpage should update to match the styles we're written.

Let's move on to our navigation bar. Here we're going to create four links that will eventually take us to various existing webpages or pages that either we've created. For this we need the **<a>** tag, write in one and hit tab instead of enter to keep it on the same line then copy and paste so you have four ready to go.

```
<div id = "navBar">
    <a>  </a>
    <a>  </a>
    <a>  </a>
    <a>  </a>
 </div>
```

The <a> tag is another where we'll write information inside of the tag itself. In the opening tag after the a, we write href ="url". Between the tags we write what we want to show up as text on the webpage. Write in the following

```
<a href="https://www.sciencenorth.ca/home"> Science North </a>
<a href= "url"> page 2 </a>
<a href= "url"> page 3 </a>
<a href= "url"> page 4 </a>
```

Here the first link will work and take us to the Science North home page. If you test it out from your open window, just hit back to return to your page view. The other links are currently placeholders to be used for future pages we create and won't take us anywhere now.

For the style of these we can head back into our style.css file and we're going to add in the selector for a. Copy in the following.

```
a {
   display: block;
   background-color: black;
   padding: 20px;
   color: white;
   width: 25%;
   float: left;
   text-align: center;
}
```

Background-color, color, text-align, and padding we've already talked about. We can use colour names as well as hex values for colours.

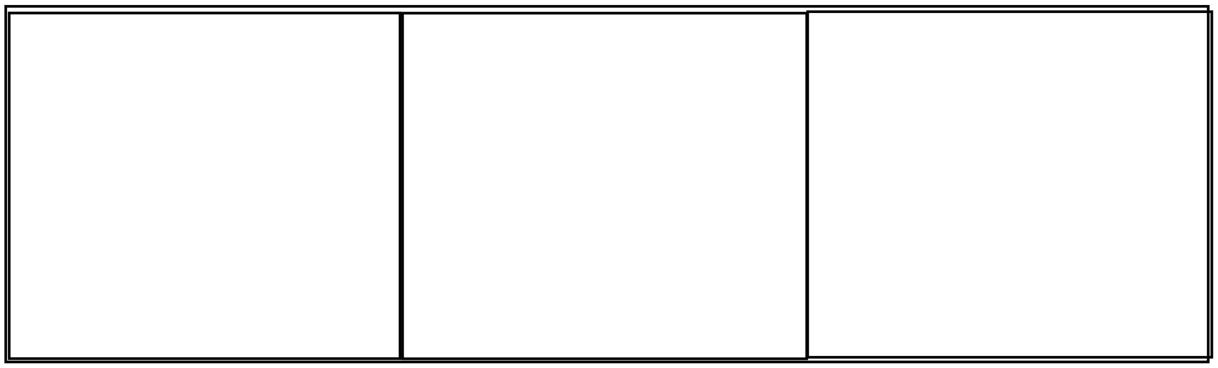
Display: block will make it so each link is a block element and not just the text, it will include some space around it as well.

Setting the **width** to 25% and **float** left will make it so each link block is a quarter of the page width and the first goes as far left as it can, the 2nd will go as far left as it can which will make it go up against the first link. Same for the 3rd and 4th.

With our links we're going to add in a special selector for what happens when we hover over them with our cursor. for this we need to add in **a:hover**. We're not going to do anything special here just swap the colours so we're able to see which link we're over at any given time. The block display will be really evident once this is done.

```
a:hover {
   background-color: white;
   color: black;
}
```

We're finished with the navigational bar for now. Let's move ahead to our first row of content. The first row we are going to sub divide into three columns using the **div** tag again. The outer rectangle below represents the division row 1. Inside this are three divisions side by side that we'll name column.



In your html file add a <div> inside of our row1 and give it the **id = "column"**. We'll use this column to write some text in.

```
Step 1:

<div id = "row1">
     <div id = "column">

     </div>

 </div>
```

```
Step 2:

<div id = "row1">
     <div id = "column">
         <h2> Title</h2>
     </div>

 </div>
```

```
Step 3:

<div id = "row1">
     <div id = "column">
       <h2>Title</h2>
       <p>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
         This is some text <br>
       </p>
     </div>

 </div>
```

<h2> is a header that's a bit smaller than **<h1>**, **<p>** is for paragraph and is used for any general text. **<br>** is for a break and moves the following text to a new line. If we were to simply write the text on a new line that would not show up as so on our webpage.

Now copy and paste the column div two more times ensuring both new columns are inside of our row but now inside of each other. When you refresh you preview window all three columns should currently be stacked one on top of each another. That means it's style time.

Much like we did with the links, we are going to change the width of each division section and float them next to each other. We'll change the colours and align the text as we've been doing.

```
#column {
   background-color: #8d8e94;
   width: 33.33%;
   float: left;
   color: black;
   text-align: center;
   padding: 50px;
}
```

Now let's add in some content to our three columns

```
<div id = "column">
       <h2> What is Climate Change </h2>
       <p>
         Climate change refers to long-term shifts
         in temperatures and weather patterns.
         When looking at the history of the Earth we
         see there have been natural shifts over time,
         but since the 1800s human activity has
         been the main cause of climate change.
      </p>
</div>
```

```
<div id = "column">
       <h2> What are the causes? </h2>
       <p>
         The primary driver of climate change is the
         burning of fossil fuels like coal, oil, and gas.
         Burning these creates greenhouse gas emissions
         which change our global climate by damaging
         the protective ozone layer in our atmosphere,
         and trapping in the sun’s heat making it hotter.
       </p>
</div>
```

```
<div id = "column">
       <h2> What can be done? </h2>
       <p>
         To combat the increase of greenhouse gases
         global leaders have held conventions and
         implemented sustainable development goals
         among other changes. The biggest of these
         changes is the switch to sustainable energy
         sources like solar and wind over fossil fuels.
       </p>
</div>
```

Moving on to Row 2 where we’ll add some images into our page.

For starters you must find and download an image or two you’d like to use for your webpage. Do an image search and find one now. In your project workspace on Replit, click the icon to add a new folder and call it images. Drag and drop your image(s) into this folder. The folder is not necessary but an important aspect of organization.

Inside our Row2 add another <div> with the id picture. Inside this div we are going to use the **<img>** tag. This is another tag where the info goes inside so there is no closing tag to pair with this one. It might seem strange to create a whole other division for the image however it's much easier to arrange divisions than any other element.

After img we want to write src = "name of the image". Because our image in is a folder we write foldername/imagename.extension. This is followed by what shows up as the alternate text in case there is an issue. The entire block of code looks like this:

```
<div id = "row2">
     <div id = "picture">
       <img src="images/earth.jpg" alt="Earth">
     </div>
</div>
```

I've used an image of the earth and have named it earth.jpg and saved it in my images folder.

Going to the styles now we're going to centre this image in our row and make it exactly half of the row. We do this in two steps.

First set the picture division to have a width of 50%. It will become 50% of row2's width since it's inside of row2. Row2 is 100% of the page width as we never changed its width so the picture division will be half the page. After this we will set the img itself to be a width of 100%. This will make it 100% the width of the division it's in and not 100% of the page width. We'll also want to set the background of the row to a certain colour as half of it currently won't be covered.

```
#row2 {
   background-color: #F8F8F8;
}
#picture {
   width: 50%;
   margin: auto;
}

img {
   width: 100%;
}
```

Setting the **margin to auto** will centre the image as it will ensure the margin is the same on either side.

Finally, we finish this page with the footer division. In the footer for now simply add:

<h1> Thank you for visiting </h1>

Since it's an h1 header the styling is already taken care of in the CSS. Add a background colour for the id footer and your first webpage is complete.

Making a website

In your project workspace create a new folder called page2. Click on the 3 dots next to your html and css files and duplicate them. Drag them into the page 2 folder and rename them to page2.html and style2.css. Inside the page2 folder create a new images folder and put an image inside of this folder to be displayed as well.

Open the page2.html file inside the head division, in the link tag change the file name to style2.css then inside the header division change homepage to say Page 2. This will let us know that the links are working as it will look slightly different from the homepage.

Go back to your original index.html file and locate our links, the <a> tags. Where it says url in quotations next to page 2, change that to say the foldername/filename.html much like we did for the photo earlier.

```
 <a href= "page2/page2.html"> page 2 </a>
```

When you click on this link now it in the preview of our homepage it will take us to the second page. In the page2.html file lets change one of the links to take us back to the homepage. In order to do this we have to be able to go backwards a bit. Instead of going into a new folder we have to tell the code in order to find the index.html file we must first go out of the folder we're in then open the index file. We do that by writing two dots before the forward slash. The code for the link back to the homepage in the page2.html should look like:

```
<a href= "../index.html"> Homepage </a>
```

Follow the same steps and add in a 3rd page. When linking between pages we'll have to go out of the folder we're in then into a new folder so it will look like ../foldername/filename.html.

Congratulations you have a working website made entirely from scratch. The exciting thing is there are 1000s of possibilities now that you can do to really make a more professional looking webpage. To help you as you get started an excellent resource is:

https://www.w3schools.com/html/default.asp
https://www.w3schools.com/css/default.asp

Here you can find many advanced styling options, we've only scratched the surface in this tutorial. There are so many resources online to help as well so get creative because if you can imagine it, you can create it.

Follow this link to view a sample page created using this tutorial: https://replit.com/@MacSTA/Science-North-Webpage?v=1

Your Task

In order to finish your website, you need at least Page 2 to have some unique content. For this you must research and report on **one natural and one human activity** driving climate change and the role Canada plays in these activities wherever possible. Try to be unique in both your activities of choice and your page development.

You can begin by simply keeping the same layout as the homepage and changing the text or starting the page from scratch with an entirely new look. This is your chance to let your creativity shine through.