Our HTML Cookbook

HTML Day 2024 | Toronto, Canada

Hosted by
Tiana Dueck
Ross Zurowski
Garry Ing

html.energy

Welcome

Welcome to HTML Day Toronto. :)
This cookbook contains some ingredients to get you in motion with your website today. Maybe you already have these ingredients in your schema, or maybe you've never cooked with them before! Either way, by the end of our session today, you'll have a lovely gourmet website to enjoy and share.

With love,
Tiana, Garry, and Ross

Associated links:
https://volvox.observer (by Tiana)
https://html.green (by Garry & Ross)
https://html.energy (by Laurel & Elliott)
https://sundaysites.cafe (by John & Tiana)

Thank you Vector Festival for including us in your programme!

Recipe #1: HTML

The recipe for HTML can be approached in a vast variety of ways! Here, I'll provide the ingredients and you're free to mix however many of them together that you'd like, as well as leave out what you're not drawn to.

Step 1: The Structure
Here are the only required ingredients in this recipe...

```
<!DOCTYPE html>
<html>

<head>

<meta name="viewport" content="width=device-width, initial-
scale=1" />

<style>
/* For our CSS. See Recipe #2. */
</style>

</head>

<body>

<!-- This is where you put your HTML elements! Example: -->
<h1>Hello world!</h1>
<span class="spanOne" id="spanOne"></span>

</body>

</html>
```

This structure of our website is the mixing bowl for our recipe. From here, we can through as many HTML ingredients into our <body> bowl. We might not know exactly what these ingredients are right away, but we can experiment with a dash of this and that. Plus, something nice about digital recipes such as this is that we can easily take things out.

HTML Ingredients

Here are some of our freshest HTML ingredients :

`<h1>Header</h1> (<h2> / <h3> / <h4>)`
The header elements help us label sections on our website.

`<p>Paragraph</p>`
The paragraph tag holds your body text.

`<main>Main Content</main>`
A bowl within a bowl! Use the main element if you might want your content to be a bit more separated from your body, which can sometimes help when organizing your styling.

`<div>Content Division</div>`
A div is a bowl that holds a section of elements that you want to stick together and separate from other elements.

`<a href="https://anchor.com">Anchor</a>`
The anchor element creates links and can be used within other elements.

`<img src="html.jpg" style="width: 100%;"/>`
The image element asks you to point to the source of an image with a link to its web or local address. It will probably need some styling. Starting with making the width 100% can help.

```
<li>
  <ol>ordered list element</ol>
  <ul>unordered list element</ul>
</li>
```
The list element holds ordered or unordered list children. Make lists with these elements! They can hold other elements.

HTML Ingredients

```
<figure style="max-width: 300px;">
  <img src="html.jpg" style="width: 100%;"/>
  <figcaption>Figure Caption</figcaption>
</figure>
```

A figure is a nice place to hold an image that you'd like to place inside a border and/or explain with a caption. Styled something like this, the image will resize to fit inside.

```
<details><summary>Details/Summary</summary>
<!-- Content goes here. (This is a comment btw) -->
</details>
```

The details/summary elements are basically a fun div that folds away into a summary, and can be clicked to show its contents. It's really fun for hiding surprises and organizing info.

```
<br>
```

Place your text on a new line within an element with this.

```
<hr
style="border: none;
border-bottom: 1px solid black;">
```

Add a horizontal line to divide up sections. Alternatively, look at borders in the CSS recipe. You'll just want to style the top or bottom border of this element.

```
<header><!-- Elements in your Header --></header>
```

You can place content you want to include in your header section within this element.

```
<footer><!-- Elements in your Footer --></footer>
```

You can place content you want to include in your footer section within this element.

HTML Ingredients

```
<table>
  <caption>Table Caption</caption>
  <thead>
    <tr>
      <th scope="col">Heading 1</th>
      <th scope="col">Heading 2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">Item</th>
      <td>Info</td>
    </tr>
    <tr>
      <th scope="row">Item</th>
      <td>Info</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <th scope="row" colspan="1">Total</th>
      <td>Result</td>
    </tr>
  </tfoot>
</table>
```

A table is a nice way to organize content. You'll want to make it reactive to its container with CSS. It can be simplified down to just the <table> <tr> <td> elements if you want.

```
<span class="spanOne" id="spanOne">Span</span>
```

Within a text friendly element (eg. <p> or), surround text you want to style uniquely within a span and give it a class or id so you can call for it specifically in your CSS.

Recipe #2: CSS

On to CSS! We can add CSS to our HTML recipe when we want to add things like colour, layout, and some interesting visual moments to our bowl. We can add CSS directly to our mixing bowl within the <style> element (see recipe #1) like so:

```
<style>
body {
  font-family: Hershey-Noailles-Futura, monospaced;
  color: #006700;
  margin: 0;
}

h1 {
  font-size: 32px;
}

.spanOne {
  color: #000000;
  width: 100px;
}

#spanTwo {
  width: 100%;
  height: auto;
}
</style>
```

There are different ways of selecting things in our bowl. We can select the <body> bowl or specific elements like <h1>. If we had a class like "spanOne" we can select all the elements with that **class (the same class can be used to identify multiple elements)** and if we had an **id (id names are unique to a single element)** of "spanTwo" we can select that specific element, as shown above. A single page webpage is special, but it can be nice to give CSS its own file connected to our HTML via the <link> element in the <head> of our HTML:

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

CSS Ingredients

Here are some CSS ingredients you can use towards adding your own style to your webpage, starting with some basics and then diving into some examples.
With any CSS attribute, the format is as such:
`element { attribute: value; }`
Some attributes and their values include:
`color (color name, hex, rgb)`
`background-color (color name, hex, rgb)`
`font-family (font name, eg. Arial, monospace, serif)`
`font-size (number in px, pt, em, %)`
`padding (1 to 4 number values in px, pt, em, %)`
`margin (1 to 4 number values in px, pt, em, %)`
`display (block [good for img elements], flex, grid, inline...)`
`position (absolute, static, relative, fixed, sticky)`
`width / height / max-width / min-height / etc (number values in px, pt, em, %)`

Some styles that can be nice to start with:
`body { margin: 0 auto; padding: 0px; margin: 0px; font-family: Times, serif; font-size: 12pt; }`
Some attributes that are nice to set up in your body element.

`p, h1, h2, h3, h4 { margin: 0px; }`
Text elements have a margin value by default, remove or change it like this.

`img { display: block; width: 100%; }`
Displaying an img element as a block will remove its default margin. Setting the width will fit it to its parent div.

`a { text-decoration: none; color: green; }`
`a: hover { text-decoration: underline; font-style: italic; }`
Customize your hyperlinks.

CSS Ingredients

CSS can get really detailed! Let's explore some examples of how you can use it...

`a { color: red; }`
Select all HTML elements of a specific type. All <a> elements will be selected and coloured red.

`.spanOne { color: green; }`
Select all elements with class="spanOne" and apply a style for those elements.

`#spanTwo { color: blue; }`
Select the element with id="spanTwo" and apply a style to the element.

`header:hover { background-color: red; }`
Apply a style to an element when the cursor is hovering on the element using :hover, added to the end of the selector.

`.spanOne:active { color: orange; }`
Apply a style to an element when the you click on it. A pseudo-class for when an element is interacted with like :hover. There's also :focus and :visited.

`.spanOne a { font-weight: bold; }`
Combine selectors to ne more specific of what you want styled like selecting a specific container and all the elements in that container.

`.spanOne p:first-child { margin: 0; }`
Select the first element amongst a set of elements.

CSS Ingredients

.spanThree:has(p) { background-color: yellow; }
Select an element and apply styles if it contains specific elements or properties.

p, li, a { color: red; }
Select multiple elements and apply the same style.

.menu { display: flex; }
Change the layout of elements by applying another flow.

.sectionOne { margin: 20px; padding: 40px; }
Add empty space away from the element using margin. Add negative space inside an element using padding. Add spacing to specific edges using more specific properties like margin-left, padding-bottom.

.sectionOne { border-radius: 50px; }
Add rounded corners to elements. You can create circle using border-radius: 50%.

.sectionOne { width: 100vw; }
vw and vh units are the viewport width and height of the browser. 100vw/vh means 100% of the viewport.

a { transition: color 2s; }
Smooth the transition between two visual states by adding a transition to a property and the time it takes.

body { font-family: Hershey-Noailles-Futura, monospaced; color: #006700; }
Change the fonts being used by listing the fonts in order of what should be presented if available.

CSS Ingredients

`body { background-image: url("html.jpg"); }`
Apply a background image to an element.

`.sectionFour { background: linear-gradient(45deg, blue, red); }`
Apply a gradient as the background to an element.

Animation is another fun thing to experiment with in CSS! Animate any attributes by customizing the format below...

```
.animatedElement {
  animation: example 4s infinite;
}

@keyframes example {
  0% {background-color: red;}
  50% {background-color: green;}
  100% {background-color: red;}
}
```

Want to edit your mobile version of the site?
`@media only screen and (min-width: 600px) { css here }`
Note that any values already set outside of this will still be in effect unless changed within this section. You can alter the screen size to match whatever screen you're styling for.

In a pinch, you can style your HTML inline:
`<h1 style="color: blue">Hello World!</h1>`

CSS Grid Basics

```
<!DOCTYPE html>
<html>
<head>
<style>
.grid-container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  background-color: green;
  padding: 10px;
}
.grid-item {
  background-color: ivory;
  border: 1px solid rgba(0, 0, 0, 0.8);
  padding: 20px;
  font-size: 30px;
  text-align: center;
}
</style>
</head>
<body>
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
</div>
</body>
</html>
```

This grid will have 3 columns and automatically form rows. We have to tell our computer which div is the grid container and which are its items so that it can style them according to their class, which we call in our styling. Change or add values of the attributes to your liking! While tables are good for organizing information, grids are more flexible for formatting the webpage layout.

Recipe #3: Prompts

Not sure where to start? Find some inspiration here...

1. What's a website the wildlife in the park would want to read? A website for a worm? A webpage for a sparrow?
2. Find a lesser used HTML element: what uncovered beauty hides within its brackets?
3. Make a website like a slinky.
4. Embody the spirit of a food product of your choosing.
5. Write a webpage about the history of your neighbourhood, your closest park, or maybe just your relationship to it.
6. Listen: make a website about the sounds you're hearing right now.
7. Write a story within HTML tables: do you read left to right? Top to bottom? What do gaps mean? Make multiple tables or nest tables within tables.
8. Make a choose your own adventure story using the <details> and <summary> elements.

Prompts

9. Make a website about your journey to the park today. What did you see along the way? What was memorable?
10. Go beyond a single page: what's the strangest way you can <a>.
11. Make a site that makes you feel calm.
12. Make a site that changes over time.
13. Write a short poem about your favourite kind of bird. But! It needs to use more than 20 different types of HTML tags. Consider <mark>, <del>, <meter>, <sup> and <sub>.
14. Break out of ASCII: create a page that celebrates your favourite weird Unicode characters, like ❑, ☻, ✜, ⊳ — bust out all the accents you can find!