




Branch: master ▾ netcom_advanced_python / day-2 / 2-intro-to-webdev.md Find file Copy path

 t-0-m-1-3 edited image markdown aedce34 2 minutes ago

1 contributor

149 lines (91 sloc) 7.46 KB Raw Blame History  

First things first: planning

Before doing anything, you need some ideas. What should your website actually do? A website can do basically anything,

To begin, you'll need to answer these questions:

1. **What is your website about?** Do you like dogs, New York, or Pac-Man?
2. **What information are you presenting on the subject?** Write a title and a few paragraphs and think of an image you'd like to show on your page.
3. **What does your website look like, in simple high-level terms?** What's the background color? What kind of font is appropriate: formal, cartoony, bold and loud, subtle?

Sketching out your design

Next, grab pen and paper and sketch out roughly how you want your site to look. For your first simple webpage, there's not much to sketch out

<https://mdn.mozillademos.org/files/9239/website-drawing-scan.png>

Choosing your assets

At this point, it's good to start putting together the content that will eventually appear on your webpage.

Text

You should still have your paragraphs and title from earlier. Keep these close by.

Images

Fonts

Dealing with files

A website consists of many files: text content, code, stylesheets, media content, and so on.

When you're building a website, you need to assemble these files into a sensible structure on your local computer, make sure they can talk to one another, and get all your content looking right before you eventually upload them to a server.

Where should your website live on your computer?

- When you are working on a website locally on your computer, you should keep all the related files in a single folder that mirrors the published website's file structure on the server.

- This folder can live anywhere you like, but you should put it somewhere where you can easily find it

You should use a hyphen for your file names. The Google search engine treats a hyphen as a word separator but does not regard an underscore that way. For these reasons, it is best to get into the habit of writing your folder and file names lowercase with no spaces and with words separated by dashes

What structure should your website have?

Next, let's look at what structure our test site should have.

The most common things we'll have on any website project we create are an index HTML file and folders to contain images, style files, and script files.

1. `index.html` : This file will generally contain your homepage content, that is, the text and images that people see when they first go to your site.
2. `images` folder: This folder will contain all the images that you use on your site.
3. `styles` folder: This folder will contain the CSS code used to style your content (for example, setting text and background colors).
4. `scripts` folder: This folder will contain all the JavaScript code used to add interactive functionality to your site (e.g. buttons that load data when clicked).

File paths

To make files talk to one another, you have to provide a file path between them — basically a route, so one file knows where another one is.

Some general rules for file paths:

- To link to a target file in the same directory as the invoking HTML file, just use the filename, e.g. `my-image.jpg`.
- To reference a file in a subdirectory, write the directory name in front of the path, plus a forward slash, e.g. `subdirectory/my-image.jpg`.
- To link to a target file in the directory above the invoking HTML file, write two dots. So for example, if `index.html` was inside a subfolder of `test-site` and `my-image.jpg` was inside `test-site`, you could reference `my-image.jpg` from `index.html` using `../my-image.jpg`.
- You can combine these as much as you like, for example `../subdirectory/another-subdirectory/my-image.jpg`.

HTML basics

- HTML (Hypertext Markup Language) is the code that is used to structure a web page and its content.
 - For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables

So what is HTML?

- HTML is not a programming language; it is a markup language that defines the structure of your content.
- HTML consists of a series of **elements**, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way

<https://mdn.mozillademos.org/files/9347/grumpy-cat-small.png>

The main parts of our element are as follows:

1. **The opening tag**: This consists of the name of the element (in this case, `p`), wrapped in opening and closing

angle brackets. This states where the element begins or starts to take effect — in this case where the paragraph begins.

2. **The closing tag:** This is the same as the opening tag, except that it includes a forward slash before the element name. This states where the element ends — in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
3. **The content:** This is the content of the element, which in this case, is just text.
4. **The element:** The opening tag, the closing tag and the content together comprise the element.

Elements can also have attributes that look like the following:

<https://mdn.mozillademos.org/files/9345/grumpy-cat-attribute-small.png>

- Attributes contain extra information about the element that you don't want to appear in the actual content.
- An attribute should always have the following:
 - A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
 - The attribute name followed by an equal sign.
 - The attribute value wrapped by opening and closing quotation marks

Nesting elements

- You can put elements inside other elements too — this is called **nesting**.

```
<p>My cat is <strong>very</strong> grumpy.</p>
```

Empty elements

Some elements have no content and are called empty elements. Take the `` element

- there is no closing `` tag and no inner content. This is because an image element doesn't wrap content to affect it

CSS basics

- CSS (Cascading Style Sheets) is the code you use to style your webpage
- CSS is not really a programming language. It is not a markup language either — it is a style sheet language. This means that it lets you apply styles selectively to elements in HTML documents

```
p {  
  color: red;  
}
```

Anatomy of a CSS ruleset

Let's look at the above CSS

<https://mdn.mozillademos.org/files/9461/css-declaration-small.png>

- whole structure is called a **rule set**
1. **Selector** The HTML element name at the start of the rule set. It selects the element(s) to be styled (in this case, `<p>` elements). To style a different element, just change the selector.
 2. **Declaration** A single rule like `color: red;` specifying which of the element's properties you want to style.

3. **Properties** Ways in which you can style a given HTML element.

4. **Property value** To the right of the property after the colon, we have the property value, which chooses one out of many possible appearances for a given property (there are many color values besides red).

Note the other important parts of the syntax:

- Each rule set (apart from the selector) must be wrapped in curly braces ({ }).
- Within each declaration, you must use a colon (:) to separate the property from its values.
- Within each rule set, you must use a semicolon (;) to separate each declaration from the next one.

CSS Is based on a box model