mdn web docs

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. Dealing with files discusses some issues you should be aware of so you can set up a sensible file structure for your website.

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, maybe on your Desktop, in your Home folder, or at the root of your hard drive.

1. Choose a place to store your website projects. Inside your chosen place, create a new folder called `web-projects` (or similar). This is where all your website projects will live.
2. Inside this first folder, create another folder to store your first website in. Call it `test-site` (or something more imaginative).

An aside on casing and spacing

You'll notice that throughout this article, we ask you to name folders and files completely in lowercase with no spaces. This is because:

1. Many computers, particularly web servers, are case-sensitive. So for example, if you put an image on your website at `test-site/MyImage.jpg` and then in a different file you

try to invoke the image as `test-site/myimage.jpg`, it may not work.

2. Browsers, web servers, and programming languages do not handle spaces consistently. For example, if you use spaces in your filename, some systems may treat the filename as two filenames. Some servers will replace the areas in your filenames with "%20" (the character code for spaces in URLs), resulting in all your links being broken. It's better to separate words with hyphens, rather than underscores: `my-file.html` vs. `my_file.html`.

The short answer is that 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 hyphens, at least until you know what you're doing. That way you'll bump into fewer problems later down the road.

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. Let's create these now:

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. Using your text editor, create a new file called `index.html` and save it just inside your `test-site` folder.
2. **`images`** folder: This folder will contain all the images that you use on your site. Create a folder called `images`, inside your `test-site` folder.
3. **`styles`** folder: This folder will contain the CSS code used to style your content (for example, setting text and background colors). Create a folder called `styles`, inside your `test-site` folder.
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). Create a folder called `scripts`, inside your `test-site` folder.

Note: On Windows computers, you might have trouble seeing the file names, because Windows has an option called Hide extensions for known file types turned on by default. Generally, you can turn this off by going to Windows Explorer, selecting the Folder options... option, unchecking the Hide extensions for known file types check box, then clicking OK. For more specific information covering your version of Windows, you can search on the web.

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. To demonstrate this, we will insert a little bit of HTML into our `index.html` file, and make it display the image you chose in the article "What will your website look like?" Alternatively, you can choose an existing image at your disposal, on your computer or from the Web, and use it in the following steps:

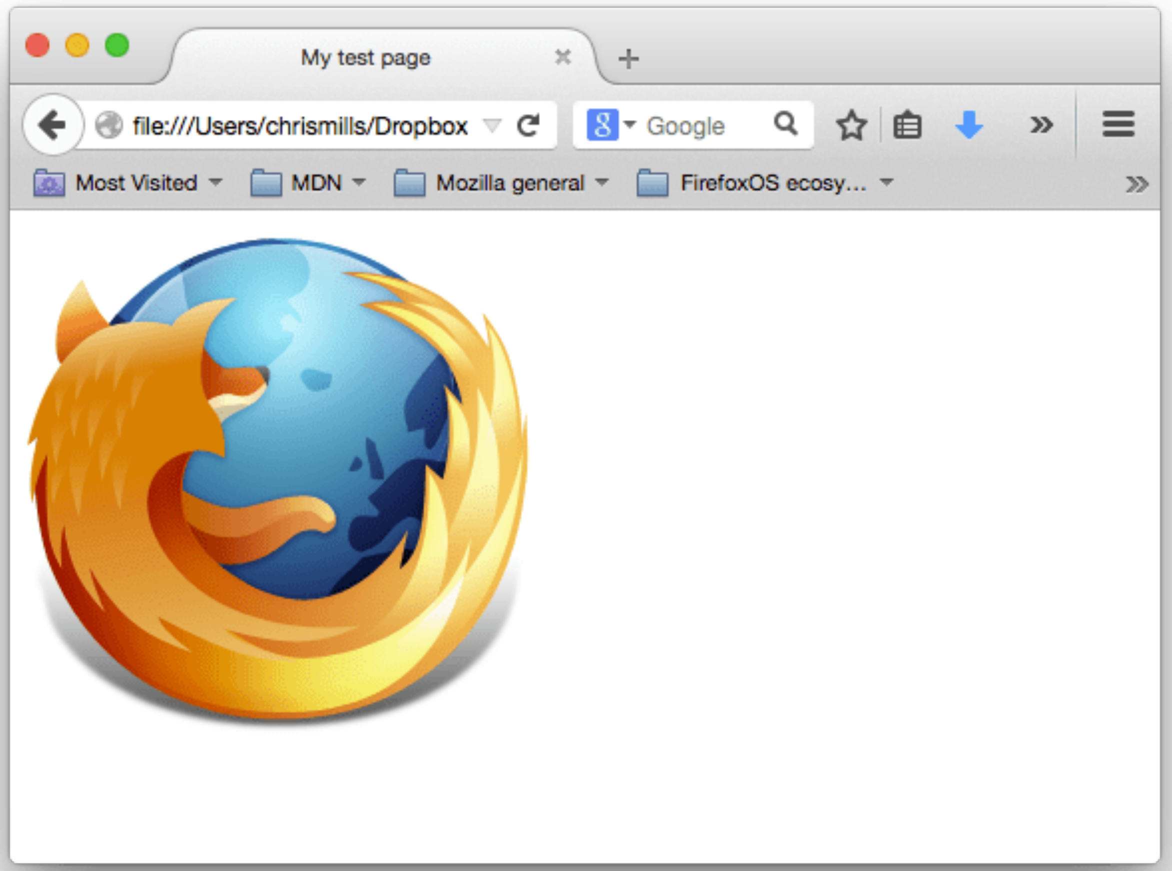
1. Copy the image you chose earlier into your `images` folder.
2. Open up your `index.html` file, and insert the following code into the file exactly as shown. Don't worry about what it all means for now — we'll look at the structures in more detail later in the series.

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>My test page</title>
  </head>
  <body>
    <img src="" alt="My test image" />
  </body>
</html>
```

3. The line `<img src="" alt="My test image">` is the HTML code that inserts an image into the page. We need to tell the HTML where the image is. The image is inside the

images directory, which is in the same directory as `index.html`. To walk down the file structure from `index.html` to our image, the file path we'd need is `images/your-image-filename`. For example, our image is called `firefox-icon.png`, so the file path is `images/firefox-icon.png`.

4. Insert the file path into your HTML code between the double quote marks of the `src=""` code.
5. Change the contents of the `alt` attribute to a [description of the image](#) you are including. In this case, `alt="Firefox logo: flaming fox wrapping the world"`.
6. Save your HTML file, then load it in your web browser (double-click the file). You should see your new webpage displaying your image!



Some general rules for file paths:

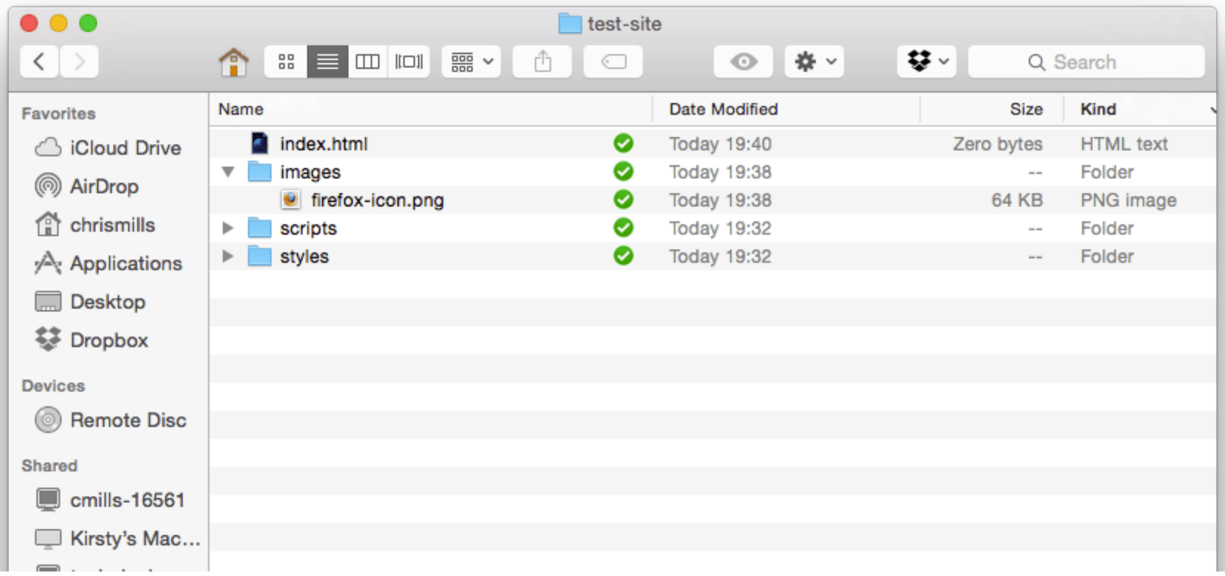
- To link to a target file in the same directory as the invoking HTML file, just use the filename, e.g. `my-image.jpg`.
- To reference a file in a subdirectory, write the directory name in front of the path, plus a forward slash, e.g. `subdirectory/my-image.jpg`.
- To link to a target file in the directory 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`.

For now, this is about all you need to know.

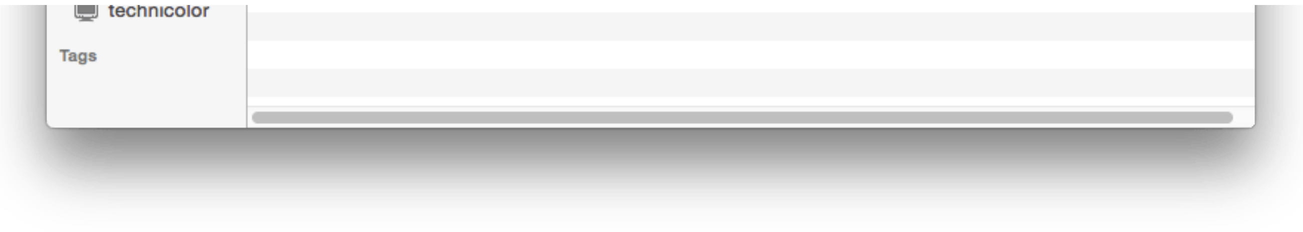
> Note: The Windows file system tends to use backslashes, not forward slashes, e.g. `C:\Windows`. This doesn't matter in HTML — even if you are developing your website on Windows, you should still use forward slashes in your code.

What else should be done?

That is about it for now. Your folder structure should look something like this:



Name	Date Modified	Size	Kind
index.html	Today 19:40	Zero bytes	HTML text
images	Today 19:38	--	Folder
firefox-icon.png	Today 19:38	64 KB	PNG image
scripts	Today 19:32	--	Folder
styles	Today 19:32	--	Folder



This page was last modified on Feb 16, 2023 by MDN contributors.