

# Tribute - make a website

a short guide to creating webpages

## leave your mark on the world

Long ago, or not really long ago, it was common for people to make personal websites.

It took Orpheus the dinosaur a long time to make one of her own, possibly because she doesn't have fingers :( (She will not confirm or deny whether it was actually a Tumblr blog dedicated to posting fanfiction and raccoon memes. But hey, no judgement.)

In this day and age, spaces like these are becoming less common. Orpheus is here to help you ~~embrace cringe~~ build a fanpage dedicated towards something you really, really like - like a tribute!

You'll learn:

- What to make your website about
- How to make it look nice
- How to publish it for other people

## what to make?

Think about something you really enjoy. A TV show, or a video game, a book, an artist, or even food.

Personally, Orpheus really likes playing this video game called Undertale. She wants to show you how you can make your own cool website for something you like.

Consider what's special about your chosen topic and how you could apply it to a website. Specifically, things like:

- Fonts?

- Theme/colour scheme?
- Music and audio?

If it's a game, think about what the in-game menus and interface look like. If it's a book, think about the design of the cover. If it's an artist or band, think about what their albums look like, or the app or device you usually use to listen to their music.

Don't restrict yourself to what you think a normal website might look like. Be creative! Orpheus really likes the sites on Neocities, for instance.

## how to make it

Now that you have an idea, how do you actually turn it into a website?

There are many ways, but they all boil down to the very basics of HTML and CSS. In essence - HTML is the skeleton of the website, and CSS is the stuff that makes it look nice.

Let's use the example of Undertale from before. Undertale has a fairly simple design, and uses pixel art and a monospace font. It's also characterised by lots of black boxes with white outlines. If we bring all those aspects together, we can make a cool video game tribute.



Let's quickly break this down into separate components. First, though, everything you see here is inside this basic HTML file.

index.html

```
<!DOCTYPE html>
<head></head>
<body>
<!-- all the cool stuff goes here! -->
</body>
```

1. **Image:** That picture of Heidi the raccoon

```
<img
  src = "https://cdn.hackclub.com/019d30f1-9e41-7d88-ab94-b22ce6fc68ce/image.png"
  style = "display: block;
    margin-left: auto;
    margin-right: auto;
    margin-top: 16px;
    margin-bottom: 16px;
    height: 160px;">
</img>
```

2. **Box:** The box with flavour text.

index.html

```
<div
  style = "width: 100%;
    border-width: 4px;
    border-style: solid;
    border-color: white;
    color: white;
    padding: 16px;
    font-family: monospace;">
  A raccoon approaches.
</div>
```

3. **Everything else:** All that text on the bottom.

index.html

```
<div
  style = "width: 100%;
    display: flex;
    flex-direction: row;
    color: white;
    font-family: monospace;
    padding-top: 8px;
    padding-bottom: 8px;">

  <span>ORPHEUS LVL 0</span>

  <span
    style = "margin-left: auto;
      margin-right: auto;">
    HP

  <span
    style = "background-color: #FEC700;
      padding: 8px;
      width: 24px;
      display: inline-block;">
  </span>
  20 / 20
</div>
```

Put all of those components in a div with styles `background-color: black; padding: 16px;` and you're good to go!

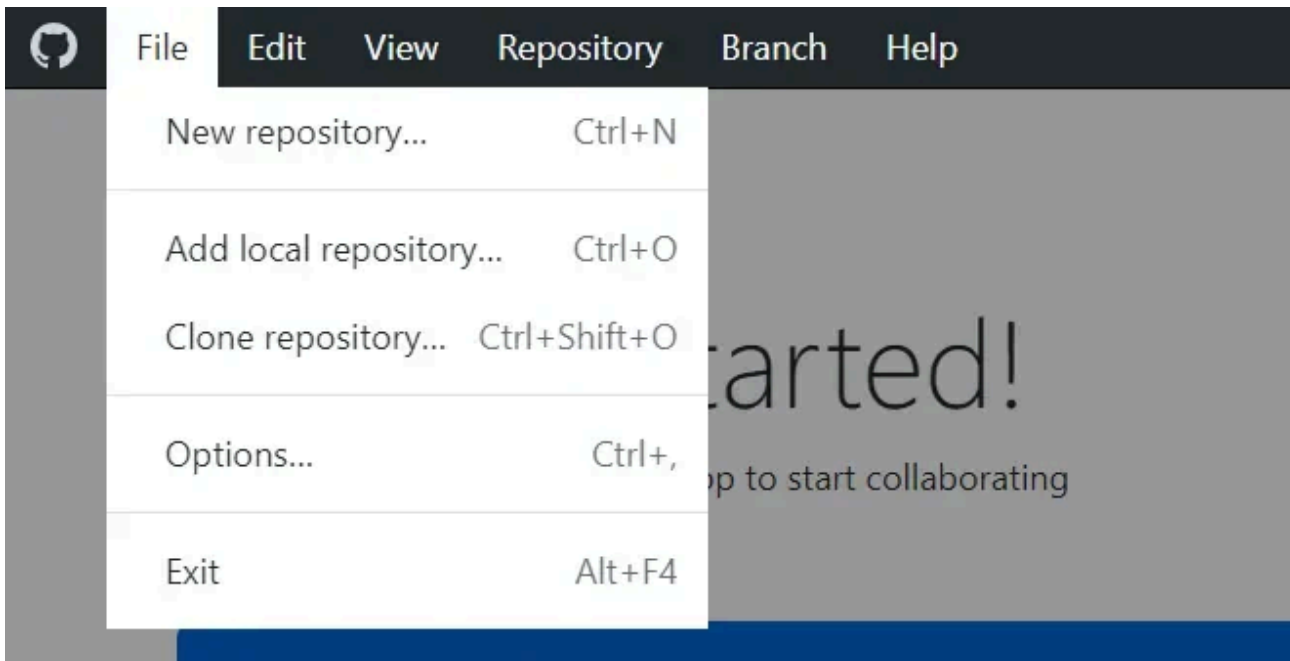
Everything inside `style = ""` is what's known as inline CSS. You can either keep your CSS styles inline in your HTML, or you can put it in a separate .css file.

But wait, you ask: how do I do this?

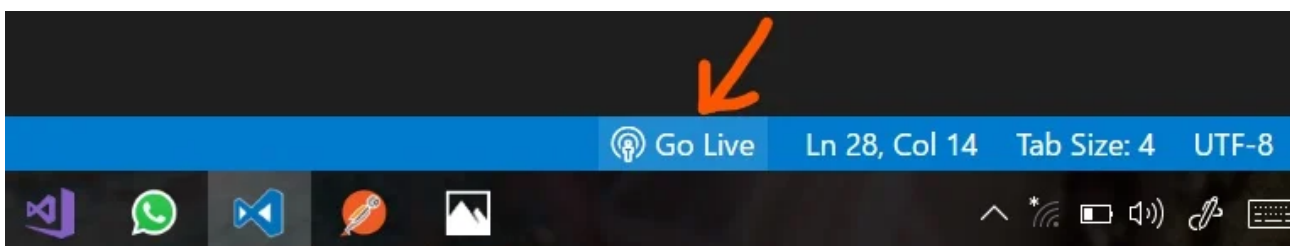
There's a few different options for storing your code. One example is GitHub, which lets you store your code and make changes to it as well as see other people's code.

1. Sign up for GitHub [here](#).

2. Download GitHub Desktop [here](#). Open it and log in with the GitHub account you just made!
3. Make a new repository using the GitHub Desktop UI, by pressing File > New Repository. Call it something like “my-website” - make your repository **public** and initialise it with a README.md. This will make a new folder on your device, typically in the **Documents/GitHub** folder.



4. Install [Visual Studio Code](#), and set up [Hackatime](#). (You might have this set up already, which is great!)
5. Open VS Code and open the folder that GitHub Desktop made for you (in the Documents folder!) using File > Open Folder, or press Ctrl/Cmd + O.
6. While in VS Code, make a new file called index.html in that folder.
7. Install the [VS Code Live Server extension](#). This will let you see your website in the browser! At any point in development, click the 'Go Live' button to start serving your website.



8. Now, if you start writing your HTML in index.html, you'll be able to see your work live.

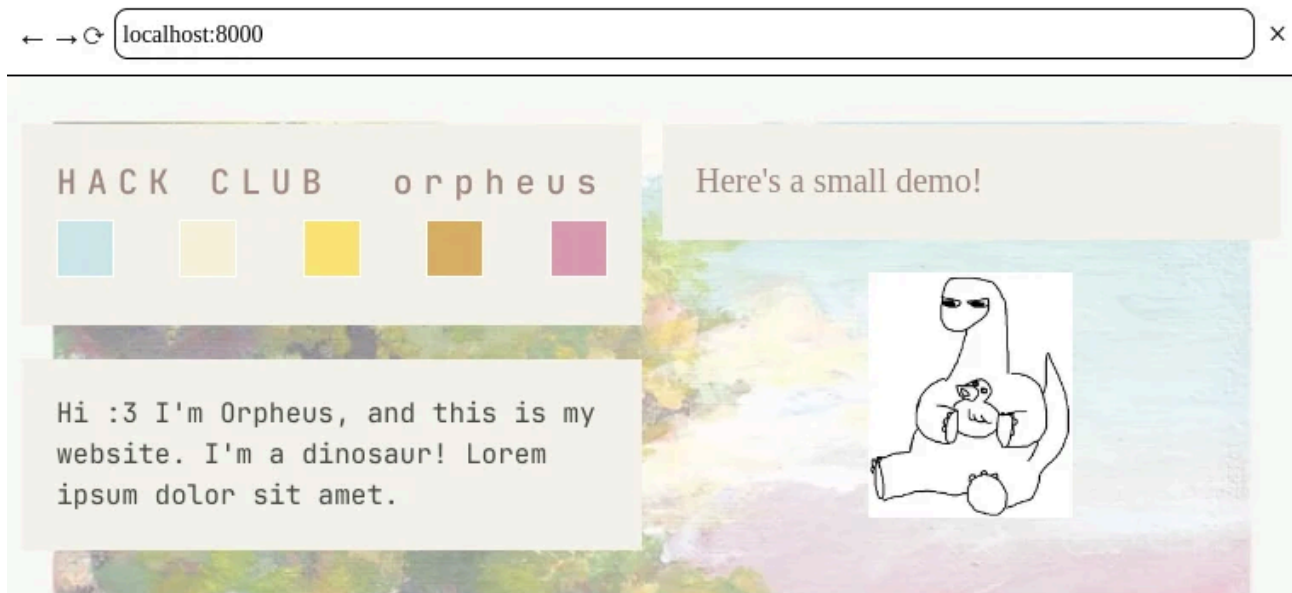
## another example [↗](#)

Orpheus also really likes the album Citrona by Flipturn. She wants to make a website for herself that's based off this album.

What stands out about this, anyways?

1. The palette of the painting and the colour used for the text at the bottom.
2. The sans-serif font and spacing (kerning) between letters.
3. The off-white box in the background, that looks vaguely like a Polaroid.





The issue is sometimes, your inline CSS styles get a bit complicated when you're making your website. That's why you can put your CSS styles in a different file. Name it something useful - like styles.css.

Then you can tell your HTML where to look for CSS styles. In the head section, add the following line (if you called your CSS file something different, update the name)

index.html

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

Inside your CSS file, you can now make changes to elements. For example:

styles.css

```
/*
this will affect the <body></body> tag.
you can use this to style any HTML tag, like <p>, <div>, <span>, <a>, etc.
*/
body {
  width: 80%;
}

/*
this will affect elements with the 'cool' class, as shown by the '.'
e.g <div class = "cool"><div>

classes can be applied to multiple elements
*/
.cool {
  color: white;
}

/*
this will affect elements with the 'very-cool' id, as shown by the '#'
e.g <div id = "very-cool"><div>

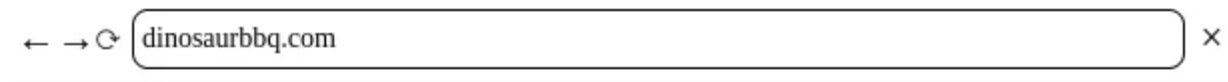
ids should only be applied to one element
*/
#very-cool {
  background: white;
}
```

It's basically the exact same as inline styles, but in a different file.

Now, go forth and make your website yourself! Remember, Neocities is a great place to find inspiration.

## making your website real

It makes Orpheus really sad when she clicks on a link and sees this:



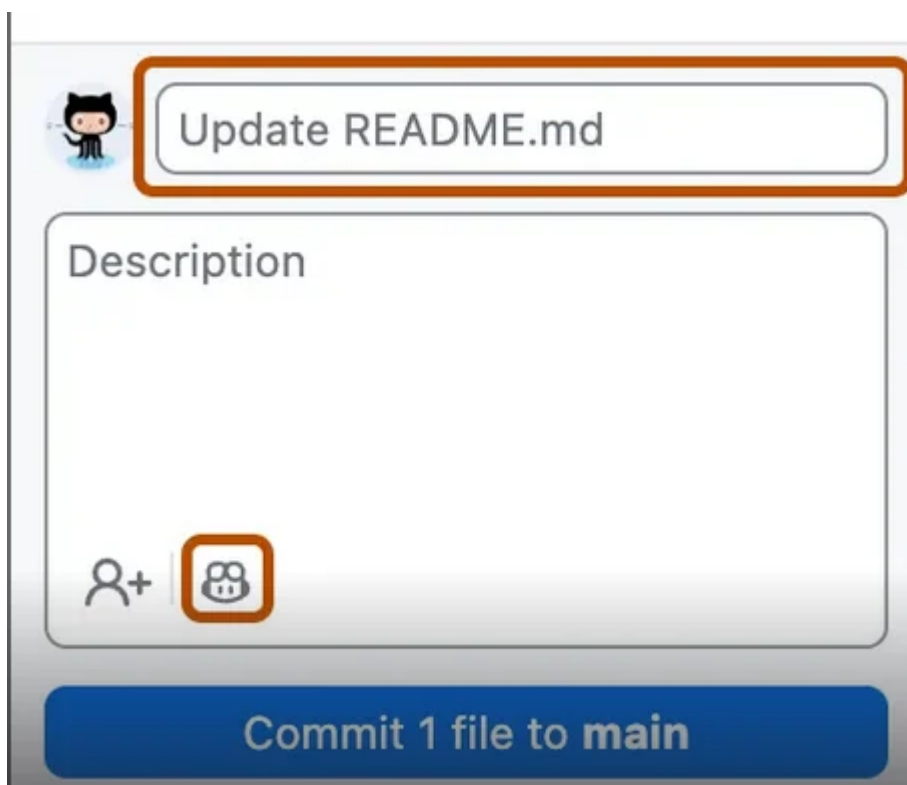
# 404

page not found! this website was probably deleted.

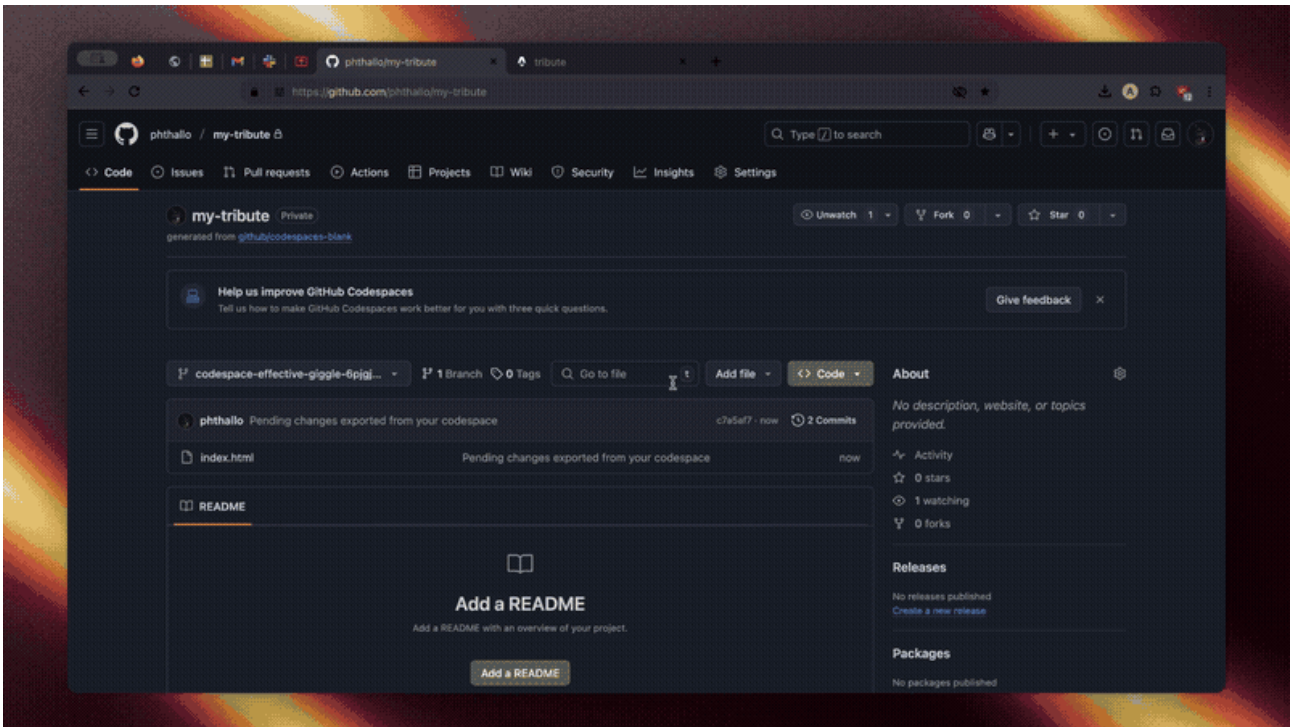
Someone probably put a lot of effort into making it, right? And now it's nothing...

Here's where GitHub saves the day again! GitHub is both a website that you can use to both store your code, but it also offers a feature that lets you host a website on it. It's called GitHub Pages.

1. First, let's make your code public. Open GitHub Desktop again - you should see a summary of your project changes.
2. On the left hand side, you'll see an option to 'commit your file'. This just means save your changes! Give it a descriptive commit message, such as 'add initial files', then press 'Commit to main'.



3. After that - let's make your changes appear on GitHub itself! Press the 'push origin' button to push your changes to your GitHub repository.
4. Go to GitHub.com and find your repository! Follow these steps to publish your website permanently online. You'll be given a link in the format `YOURUSERNAME.github.io/nameofyourrepository`. Send this link to anyone who you want to show off your project too!



## You're done!

Congratulations on finishing Tribute.

If you feel confident, you can move on and start your own project that builds on what you learned here.

If not - no worries! Feel free to ask for help in the Hack Club Slack.

*This guide was originally written by Phthallo for Tribute.*