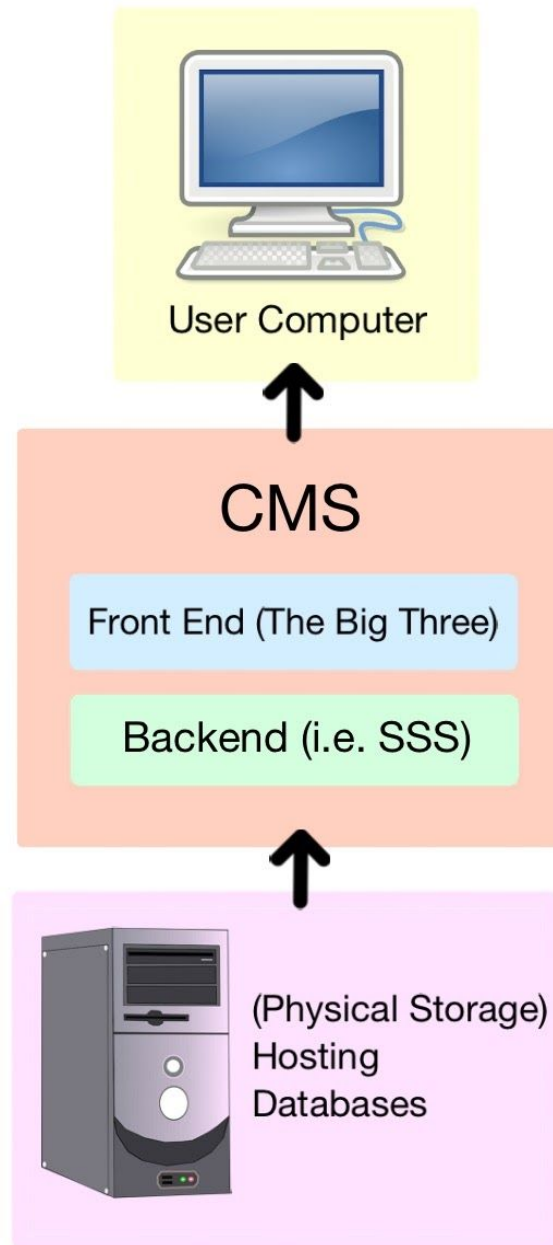


Portfolio Website with Github Pages and p5.js

Before we make our portfolio website, we need to learn a little about how websites work in general.

Making a Website



Websites don't just float in the cloud; there has to be a physical place that their data is stored. This physical storage space is usually taken care of by a web **host**. The web host stores website data on web servers. These web servers are data-centric computers that 'know' how to speak to web browsers through the HTTP protocol. See a [photo of web servers here](#), and see an infographic on [where the internet's data is stored here](#). We are going to be *hosting* our portfolio websites on **Github** (but we'll get to that later).

Most websites use a **Content Management System**, or CMS, to organize data and code for site editors, administrators, content creators. One example of a CMS is Wordpress. A CMS organizes code on the front end (the Big Three) and the backend (such as a server-side scripting language, or SSS). Wordpress uses PHP as it's SSS. Our portfolio websites will be small, so we don't need to use a CMS; we're going to directly deal with the Big Three, but if you're interested in more about CMSs, you can watch some videos from Lynda's [CMS Essential Training series](#).

The Big Three

The **Big Three** web programming languages are **HTML**, **CSS**, and **Javascript**. We will be using all three to build our website.



HTML

The Builder



CSS

The Artist

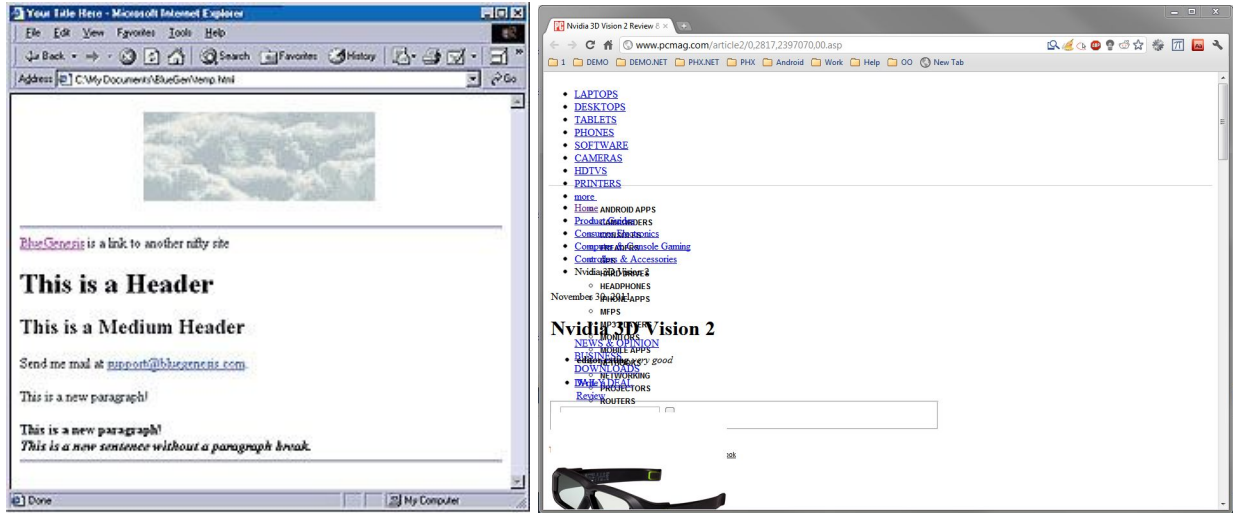


Javascript

The Wizard

HTML

HTML, or hypertext markup language, is the builder. HTML makes up the basic building blocks of websites. If you've seen older websites or incompletely loaded websites (see below), you've seen what plain HTML looks like. It is made up of basic text, headers, links, images, lists, etc.



Left: a basic HTML site.

Right: an incompletely loaded site showing just the HTML code.

HTML code looks like this:

```
<!DOCTYPE html>
<html>
  <head>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/p5.min.js"></scrip
t>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/addons/p5.dom.min.
js"></script>
    <script src="sketch.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>Here is some text.</p>
    <div id="sketch1css">
    </div>
  </body>
</html>
```

You should recognize this from when we ran p5.js code from our computers a few weeks ago.

CSS

Along with HTML, we will use **CSS**, or cascading style sheets. CSS is the artist. CSS stylizes HTML. Using **classes and IDs**, CSS can change text colors, fonts, and sizes; position text, images, and other things wherever you want them; and more. Notice we link to CSS code (i.e. the **stylesheet**) in the above HTML code.

CSS code looks like this:

```
#nametext {
  color: #2f1962;
  text-align: center;
  font-size: 8em;
  font-family: 'Rock Salt', cursive;
  line-height: 70%;
}
p {
  font-family: 'Verdana';
  text-align: center;
  font-size 2em;
  line-height: 200%;
}
```

JavaScript

Finally, **JavaScript** is the wizard. Javascript allows for cool things like animation, interactivity, and Object Oriented Programming. We don't need to worry about learning Javascript though, because P5.js **transcompiles** to Javascript! All that means is P5.js can be *translated* to Javascript through some transcompilation scripts so that the web browser can understand it. Again, you saw these transcompilation scripts when we ran p5.js code from our computers a few weeks ago, but if you've forgotten, they look like this:

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/p5.min.js"></scrip
t>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/addons/p5.dom.min.
js"></script>
  <script
```

```
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/addons/p5.sound.min.js"></script>
```

From top to bottom, there is a p5 transcompilation script, a script for the dom library, and a script for the sound library. You can copy and paste this code into the header (<head>SCRIPTS HERE</head>) of your HTML file.

Github and Preparing our Individual Sketches

All of the individual p5.js sketches you want on your portfolio website need to be added to Github as **repositories**. To do this, you'll need a Github account.

Make a Github Account

Go to <https://github.com/> and create a free account. Github uses a popular **version control system** called **Git**. Version control keeps track of file changes, updates, and versions to avoid this age-old problem:



Github is often called the 'social media' of programmers because it allows for easy open-source collaboration and has lots of forums for coding documentation and Q&As. We'll specifically use Github pages, which is a **website host** through Github.

Make a Repository

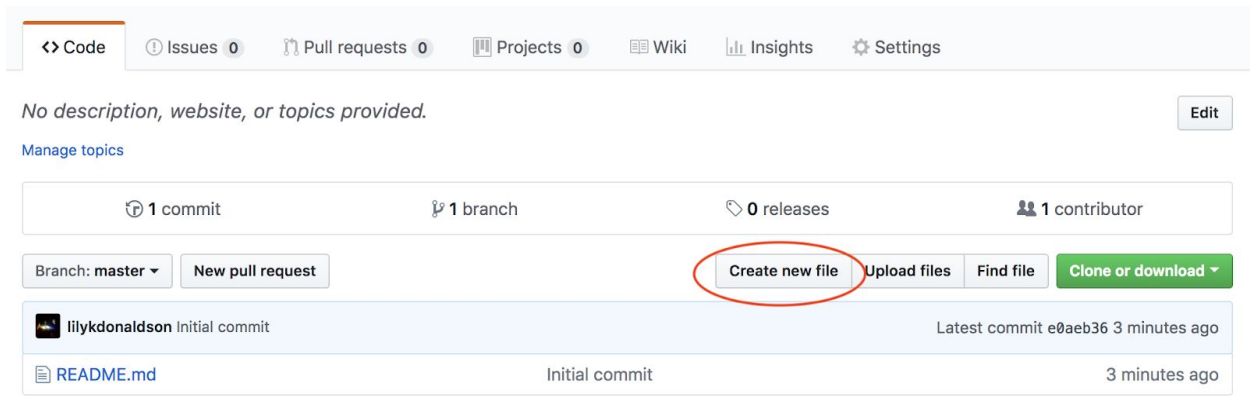
You can think of a **repository** as a folder for all your stuff. We're going to make our first repository to hold one of our p5.js sketches.

1. Click "New" to create a new repository.
2. Name your new repository something unique and specific. It's most common to use lowercase with dashes (-) between the words. For instance, if your p5.js sketch draws butterflies on the screen, maybe name your repository something like "butterfly-p5-sketch."
3. Write a short description of what your sketch does in the "Description" box.
4. Check the box next to "Initialize this repository with a README."
5. Congrats! You've made your first repository.

Putting our Code in the Repository

Often, we push files/code into repositories using the command lab or the Github desktop app, but we'll just be using the Github website for simplicity.

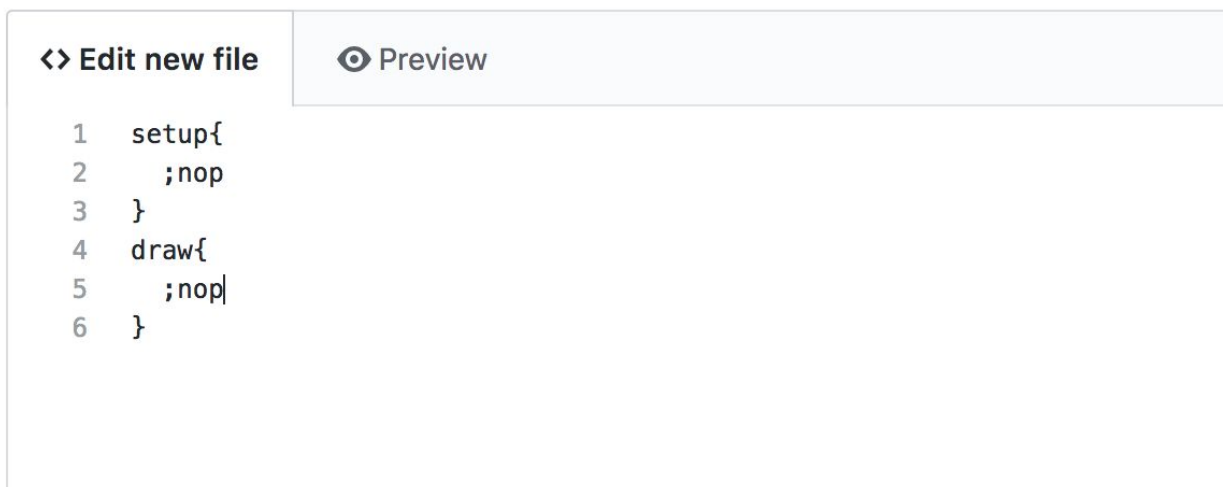
Let's add our existing p5.js drawing to our repository. On your repository's page, click "Create new file."



The screenshot shows the GitHub repository interface. At the top, there are navigation tabs: Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. Below this, there's a section for repository metadata: "No description, website, or topics provided." with an "Edit" button and a "Manage topics" link. A summary bar shows "1 commit", "1 branch", "0 releases", and "1 contributor". Below the summary bar, there are buttons for "Branch: master", "New pull request", "Create new file" (circled in red), "Upload files", "Find file", and "Clone or download". A commit history section shows an "Initial commit" by "lilykdonaldson" with a "README.md" file added 3 minutes ago.

Name your file "sketch.js" noting that you MUST use the .js file extension in the name so that Github knows you are creating a Javascript file. Paste your p5.js code into the text box. Scroll down, and click "Commit new file."

butterfly-p5-sketch2 / or [cancel](#)



The screenshot shows the "Edit new file" interface in GitHub. It has two tabs: "Edit new file" (active) and "Preview". The code editor contains the following p5.js code:

```
1  setup{
2    ;nop
3  }
4  draw{
5    ;nop
6  }
```

Now we'll need to create an HTML file for our sketch so that web browsers can read it. Create another file. Name it **index.html** and paste the following code into the text box:

UPDATED 11/29

```
<!DOCTYPE html>
<html>
```

```
<head>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/p5.min.js"></scrip
t>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/addons/p5.dom.min.
js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.5.11/addons/p5.sound.mi
n.js"></script>
  <script src="sketch.js"></script>
  <style> body {padding: 0; margin: 0;}
</style>
</head>
<body oncontextmenu="return false;">
</body>
</html>
```

Note: you don't need the second script if your sketch doesn't use the **dom library**, and you don't need the third script if your sketch doesn't use the **sound library**. Commit that file.

Add your Sketch to Github Pages

On your repository page, go to settings and scroll down to Github Pages. Select "master branch" and click save.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Source

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more](#).

master branch ▾

Save

GitHub Pages needs to update before we can see our sketch. Once the link on Github Pages settings turns from blue to green (see below), click the link to see your sketch.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is ready to be published at <https://lilykdonaldson.github.io/butterfly-p5-sketch/>.

Source
Your GitHub Pages site is currently being built from the master branch. [Learn more.](#)

master branch ▾ Save

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

✓ Your site is published at <https://lilykdonaldson.github.io/butterfly-p5-sketch/>

Source
Your GitHub Pages site is currently being built from the master branch. [Learn more.](#)

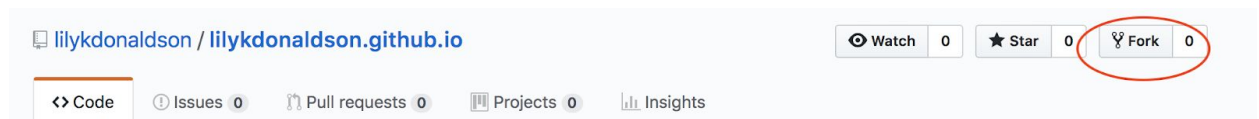
master branch ▾ Save

Save that link: we'll use it later.

Repeat this process of creating new repositories for all the p5.js sketches you want to be on your site.

Putting it all Together

1. Pick a template. [Template One](#). [Template Two](#). [Template Three](#).
2. For Template One, click [here](#). For Template Two, click [here](#). For Template Three, click [here](#).
3. Fork the template.



The template is now a repository on your account!

4. Rename the repo YOURUSERNAME.github.io
5. Edit the code to have your name, your bio, and your sketches. To change the sketches, replace the iframe links with your sketch repository links that you saved earlier.
6. In settings, make sure your repo has Github Pages turned on and directed to the master branch.

7. Your site will be live on YOURUSERNAME.github.io!

Tips

When you want to insert a sketch on your page, use an **iframe** and **div** tags in your index.html file:

```
<div id="unique-css-id-name-for-sketch" style="cursor: move;">
  <iframe src="YourGithubSketchLink.com" width="widthofcanvas"
height="heightofcanvas" scrolling="no" frameborder="0"></iframe>
</div>
```

Make sure to include `style="cursor: move;"` if your sketch is interactive. Once you insert that code into your HTML file, you can stylize the iframe/sketch in your CSS file like this:

```
#unique-css-id-name-for-sketch {
  position: absolute;
  top: 320px;
  left: 40px;
}
```