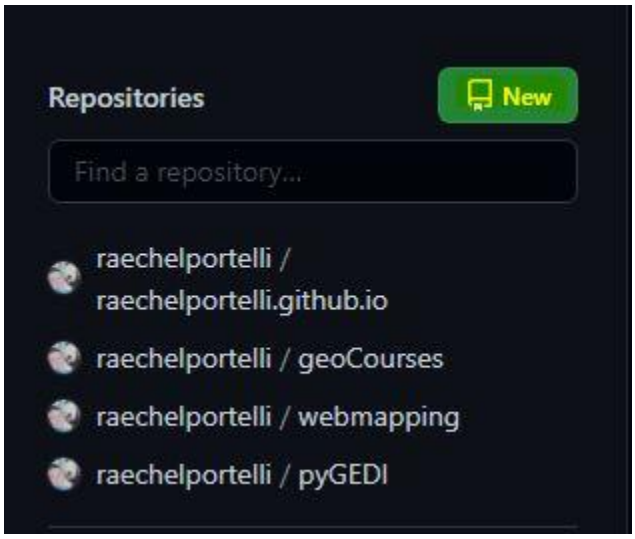
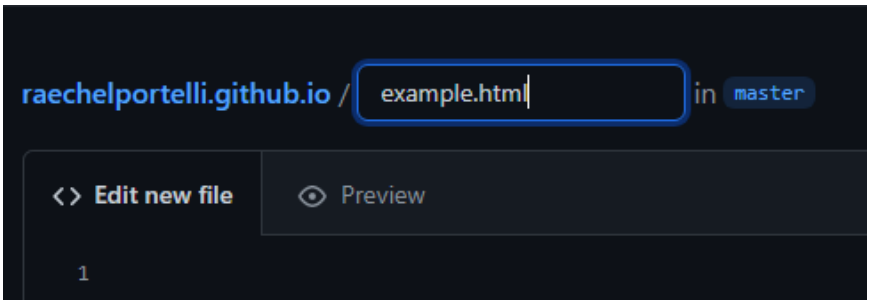
Create your own Github.io website

For resources to help with your webpage building see: https://www.w3schools.com/ and Youtube

1. Create your account with github. Use a professional sounding username as that will become the beginning of your webpage address.
2. On Github create a public repository and call it *username*.github.io you can see in the example, my username is raechelportelli
3. Navigate to the repository and select Add File. This will allow you to create a new html file.





4. I have gone ahead and created a new example.html file. You can edit your file directly in the browser.

5. The basic setup for any HTML webpage is

```
<!DOCTYPE html>
<html>
<head>
<title>A Meaningful Page Title</title>
</head>
<body>
The content of the document......
</body>
</html>
```

6. The head section of the file contains links to other packages, CSS files, or libraries that you can use to improve the website interactivity and design. The body of the HTML contains the content of your webpage.
7. Once you've added the text above to your HTML document, scroll to the bottom of the page and select Commit Changes.

Linking external resources for web page improvement

1. HTML by itself can be rather bland, lacking both design and interactivity. For those reasons, it is important to also learn about CSS and JavaScript (JS). CSS is a very straightforward language for creatively designing the HTML elements in your webpage. From font styles to layouts, CSS does it all. JavaScript on the other hand handles your interactivity needs. JavaScript is a bit more difficult to understand than HTML and CSS, but there are a number of resources available including prewritten packages that can really improve the functionality of your design.
2. First, take a look at CSS. In order to use a css file with your website you will need to create a new file. See above if you've forgotten how to do this. You can name it what you would like, but I would suggest the same root name as your html file. You should use the extension CSS.
3. Let's try a very simple example first. Let's set the font color for all text in the body of your HTML document to blue. In your CSS file add the following lines and Commit Changes.

```
body {
  color: blue;
}
```

4. If you were to go check your HTML file you would see there are no changes. That is because we haven't created a relationship between the HTML file and the CSS file. To do this open your HTML file and add the following line of code within the <head></head> tags.

```
<link rel="stylesheet" href="filename.css">
```

5. It will take a few minutes for Github to reflect your changes. You can keep hitting refresh or the enter button, but it should change to blue text.
6. CSS does more than just fonts though. CSS is also useful for creating layouts for your html documents.

Adding JavaScript Capabilities to your HTML

1. The last thing that we will look at is adding JavaScript capabilities. While you can write your own javascript files from scratch, there are many resources available to help you get started without such commitment.
2. Let's look at adding the Lightbox.js functionality to our web page. FIrst, you will want to download the related files to your own webpage. You can access them here https://lokeshdhakar.com/projects/lightbox2/.
3. Next, you will want to create a link so that the javascript file can be used on your webpage. Add the following lines of code to your html <head> section. Do not change the names of the files this time. Commit Changes

```
     <meta charset="utf-8">
     <link rel="stylesheet" type="text/css" href="css/gallery.css">
     <link rel="stylesheet" type="text/css"
href="css/lightbox.min.css">
     <script src="js/lightbox-plus-jquery.min.js"></script>
     <meta name="viewport" content="width-device-width,
initial-scale=1.0">
```

4. Now that you have the link to the JS established as well as CSS specific to the template, you will need to update the HTML. To see how the template works, you can open the examples that are in the download or read the webpage. It boils down to linking the formatting to the files using the data-lightbox attribute.

For a single image

```
<a href="images/image-1.jpg" data-lightbox="image-1" data-title="My
caption">Image #1</a>
```

For a multi image display

```
<a href="images/image-2.jpg" data-lightbox="set1">Image #2</a>
<a href="images/image-3.jpg" data-lightbox="set1">Image #3</a>
<a href="images/image-4.jpg" data-lightbox="set1">Image #4</a>
```