Web App Template for Beginners

Start with this one. Learn to build a local web app that you can modify in your future projects.

Background

Each web app has a *frontend* and *backend*. Frontend is usually for UI and design and runs on browser, on user's own computer. P5.js sketches are frontend.

Backend is behind-the-scenes code that runs on server. It is stores and organizes data and delivers your app to users, ie. clients. If 10 people access your website, there are 10 frontends in action but only 1 backend.

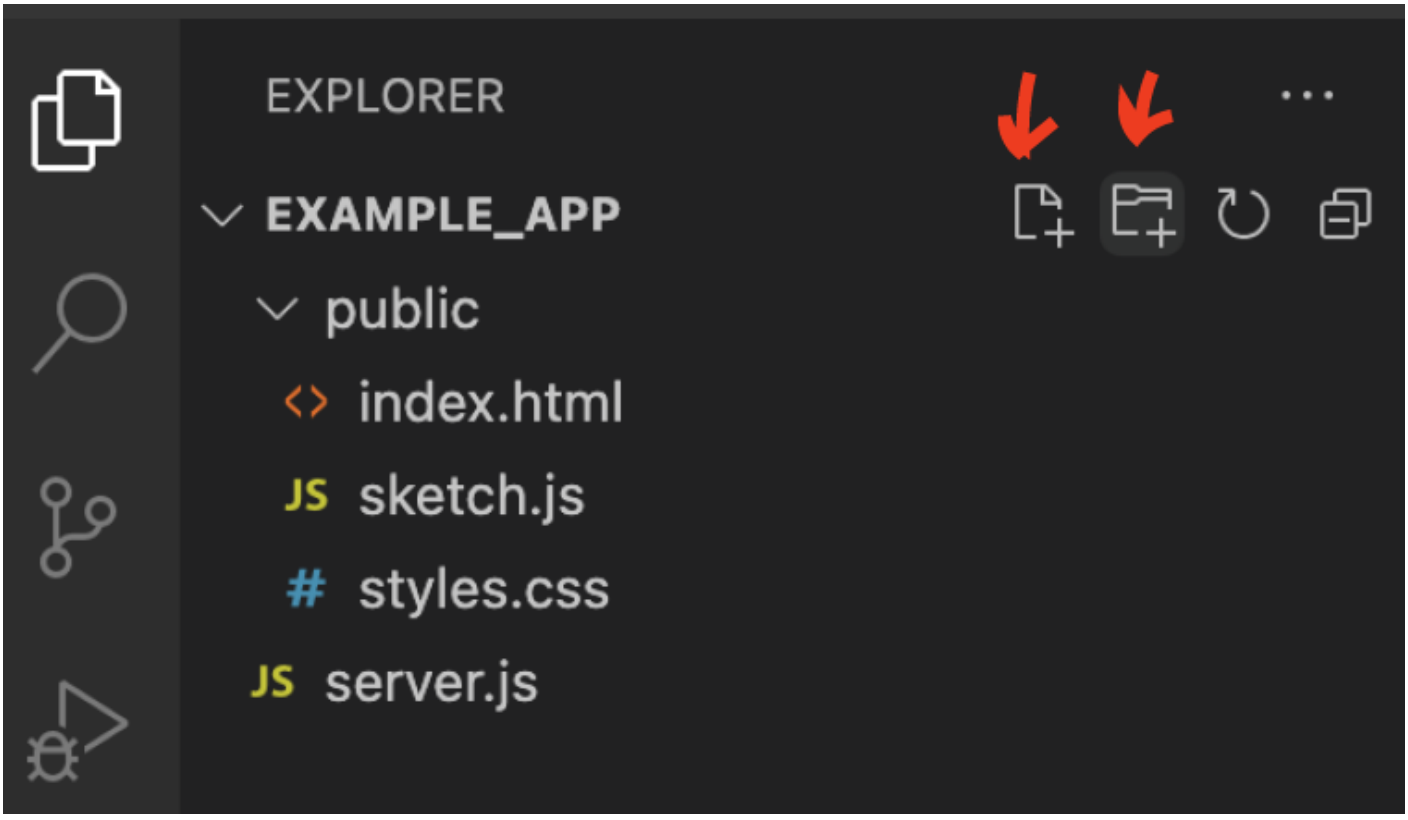
Backend is often built with Node.js or Python. Here we use Node.js.

Setup

1) Get a code editor if you don't have one yet. I like **Visual Studio**.

2) Create a folder for your project on your laptop. For example, a folder called "example_app" in you Documents folder.

3) For Visual Studio: Open example_app folder and create the following file setup by clicking the folder and file icons:



So there are three empty files in “public” subfolder and an empty server.js outside that in the root folder.

Alternatively you can create the setup outside your code editor, however Visual Code has made it really easy to create code files from the scratch so that’s why I recommend it.

Server with Node and Express

4) Install **Node**.

5) Watch videos 12.1. and 12.2. of this excellent **tutorial** by Daniel Shiffman. He explains what Node and Express are and how to get started with them. You are welcome to follow along, but steps 6-10 will give you the same results.

6) Open the command line on your computer. On Mac, go to Applications/Utilities/Terminal.

On command line, type `cd`. Then in Finder, select "example_app" folder and drag it to command line. It gives you the path to that folder automatically. Press enter.

Now you are operating inside that folder using command line. It should look like this:

```
MB-C1179:~ vjussila$ cd /Users/vjussila/Documents/CTL_teaching/example_app
MB-C1179:example_app vjussila$
```

7) On command line, type `npm init`. Answer the questions by typing to the command line and pressing enter after each question. This creates a package.json file that makes the project easier for others to manage and install. If confused, check the tutorial on step 5.

When done, type `npm install express`. This installs Express to this project folder (we only want it to live in this folder, not everywhere on your computer).

8) Go to your empty `server.js` file. Add the following code:

```
const express = require("express");
const app = express();
const server = app.listen(3000);
app.use(express.static("public"));
console.log("It works");
```

Here we are telling the backend to:

-Use Express framework

-Set our server to local port 3000

-Serve files that are in the folder called "public"

-Print "It works" when the server is running

9) Type `node server.js` on command line and press enter. This is how you tell Node to run a file called server.js. It should print "It works" on the command line.

10) Double-check by going to address *localhost:3000* on your browser. No errors? Good!

Now you should have a basic server running! But we don't have anything that the server could show. We'll fix it next.

HTML setup

11) Go to your empty html file and add the following code:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>My test project</title>
  </head>
  <body>
    <h1>My first title!</h1>
  </body>
</html>
```

12) Go to *localhost:3000* and refresh. You should see a minimalist white page with the text *My first title!*

If you get any errors, check your server.js and index.html files again. It's very easy to make a spelling mistake!

Client Javascript setup

13) Now we are only serving html. Let's add the p5 library to serve some p5.js , ie. Javascript!

In your html file, add the following line of code in the `<head>` section of your html, after the `<title>` line:

```
<script src="https://cdn.jsdelivr.net/npm/p5@1.4.0/lib/p5.js"></script>
```

So the section now looks like:

```
<head>
  <meta charset="utf-8" />
  <title>My test project</title>
  <script src="https://cdn.jsdelivr.net/npm/p5@1.4.0/lib/p5.js">  </script>
</head>
```

This line of code gives us access to p5.js library in our project. Note that you can add other Javascript libraries in a similar fashion, like **ML5** for machine learning or **Three.js** for building 3D visuals.

14) Our Javascript will live in the `sketch.js` file. First we need to reference that file in out HTML so that there is a connection. You can think about this way: HTML file is like the frame of a painting, and JS is what happens on the canvas of the painting. We need both!

In `index.html`, add the following line to the `<body>` section:

```
<script src="sketch.js"></script>
```

So that it looks like:

```
<body>
  <h1>My first title!</h1>
  <script src="sketch.js"></script>
</body>
```

15) In the empty `sketch.js`, paste the following code:

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(100);
  rectMode(CENTER);
  strokeWeight(3);
  stroke(255, 0, 0);
  fill(255, 192, 203);
  rect(100, 100, 200, 200);
}
```

Here, we are first drawing a grey background of 400 x 400 pixels. Then we add a pink rectangle with red outline to the center of the canvas. For more p5.js help, see their **reference**.

16) Go to *localhost:3000* and refresh the page. You should see text *My first title!*, grey background and the red-pink rectangle. Your first web app with backend and frontend!

Note

This is a local server and local project. Currently it only lives on your computer. In order to make a public web app that anyone can access, you need to *deploy* it. There will be a tutorial for this later. :)

Next

Try to add a paragraph of text to your page. **Guide**

Try to add an image to your page. **Guide**

Try to change the color of the rectangle with a mouse click. **Guide**

Revision #18
Created 4 January 2022 17:48:44
Updated 20 June 2023 16:09:47