Web Site Development with Visual Studio Code

Visual Studio Code vs. Glitch for Web Site Development

Glitch and Visual Studio Code both offer features and tools for Web site development.

Glitch Strengths:

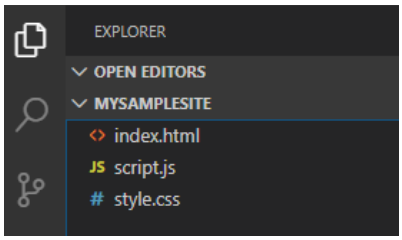
- **On-line coding:** Works from any computer or Chromebook, no setup required on your machine
- **Simplicity of project creation**: Glitch simplifies the creation of a new site by providing New Project --> hello-webpage
- **Quick Start with environment requirements**: For Flask, Glitch will automatically install required support files/frameworks from a requirements file, whereas Visual Studio Code requires setting up a virtual environment for Flask
- **Easy to Collaborate/Remix**: great for teams and for finding examples to try
- **Automatic Web site deployment:** Your project is automatically deployed to *yourprojectname*/glitch.me and ready to share with your friends, view on your phone, etc.

Visual Studio Code Strengths

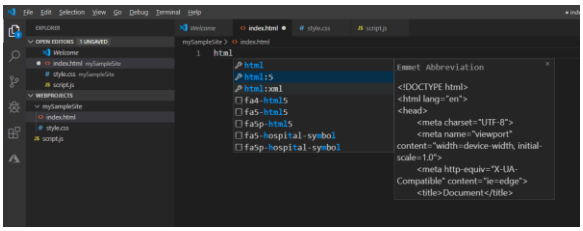
- **IntelliSense:** Code Completion, parameter info, list members, and more
- **Code snippets:** code templates you can insert by typing the first letter(s) of an abbreviation
- **Debugging:** essential tools for stepping through code and identifying/fixing problems
- **Full-featured editor**
- **Good documentation/tutorials:** https://code.visualstudio.com/docs

Create a Web Site in Visual Studio Code

- **Create a folder to organize all your Web projects**: In Finder (Mac) or File Explorer (Windows) create a folder "WebProjects" in your Documents folder
- **Create a folder for your practice Web site:** inside "WebProjects", make folder "mySampleSite"
- **Open your "mySampleSite" folder in Code**: On Mac drag the folder to VS Code in the Dock; on Windows, right-click and select "open with Code". It opens a Workspace named MYSAMPLESITE
- **Inside Visual Studio Code add file structure for a new Web site**:
    - Opening a folder in VS Code, opens a Workspace named after the folder but with all caps: MYSAMPLESITE
    - In VS Code's Explorer (if you don't see it use cmd/ctrl-shift-E) Select MYSAMPLESITE then click the "add file" icon, name the file index.html; add another file named "style.css", and a third file named "script.js". Make sure those 3 files are inside the mySampleSite folder.
    - Your folder/files list should look like this:



- **Create html template in index.html**: VS code provides "snippets" to greatly reduce typing. Select index.html to edit. Start typing '*html*' to bring up a list of abbreviations. Hover over *html:5* to see the code it represents. Hit Enter/Return (or Tab if tab completion turned on in settings) to incorporate the snippet. You now have a basic Web page with all required elements



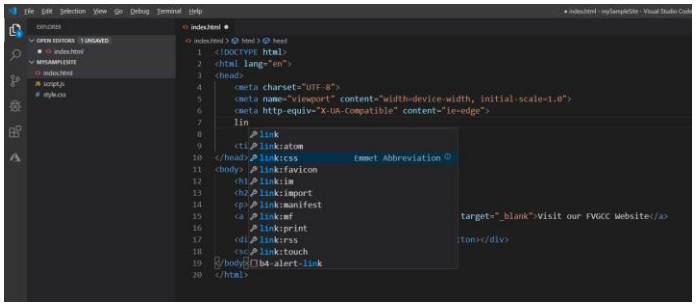
- **Add some html elements** between the <body></body> tags. Remember each element is enclosed in matching tags. Opening and closing tags are between < >; closing tag name starts with </

  Example:

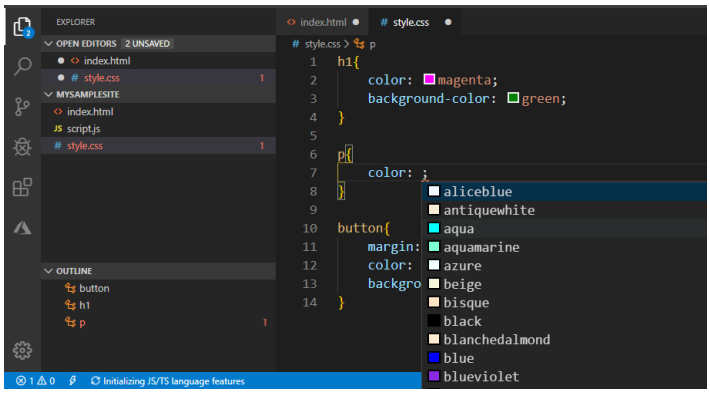
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>My Sample Site</title>
</head>
<body>
    <h1>Hello everyone!</h1>
    <p>This is a very basic Web page</p>
    <a href="https://foxvalleygirlscodingclub.com" target="_blank">FVGCC</a>
</body>
</html>
```

  - **Link your stylesheet**: inside the <head> start typing *lin* just above <title> to see the available link snippets; select *link:css*. This uses the default name *style.css*



- **Edit Stylesheet:**
    - Edit style.css to add one or two styles, perhaps for the <h1> and <p> tags:
        - Type the tag name, followed by 'curly braces' { }, between the braces list each attribute *name:value* followed by a semicolon. Visual Studio Code gives lots of context-sensitive help in setting values. Start typing *col* and click on *color* in the list; the cursor will be at insertion point to select color from displayed list:



Note the nice outline in the lower left that shows what styles are defined; you can click here to quickly go to one of your styles

- **Add Live Server Extension:** Live Server is an Extension that lets us see our Web pages in the Browser. Click **Extensions** icon in left sidebar; in search bar type "Live Server"; select the one by Ritwick Dey (should be 1st listed in search results); click Install
- **View your index.html:** with index.html open in your editor, right-click and select "Open with Live Server". Your Browser will open and display the page.
- **See Changes Live:** make a change to your index.html page and/or the style.css page while your site is still open in the Browser. Save the changes and observe them applied "live"

Debugging Web Site JavaScript in Visual Studio Code

Although you can use DevTools in Chrome for some debugging, VS Code provide a better interface for that

Install Debugger for Chrome

Select the Extensions icon in left margin, in search bar type *"Debugger for Chrome"* and then Install

Add HTML to reference JavaScript

We will add some simple HTML and JavaScript to our site to demonstrate debugging

Add this code to your index.html right above the closing </body> tag:

```
<div>
    First Name: <input type="text" id="firstName" value="FVGCC"><br>

    <button onclick="greet()">Be Greeted</button>

</div>

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

HTML Code explanation:

- tags enclose a section that contains an input element and a button element
- <input> element has a "value" attribute which is what will display on the page; most importantly to the code it has an "id" attribute which the JavaScript uses to reference that element. An input element doesn't have a closing tag
-
 is just a line break for spacing
- <button> element has an "onclick" attribute which triggers an onclick event when the button is clicked; the value of the onclick attribute is "greet()" and will execute a function named greet()
- <script></script> tags can enclose in-line script or, as in this case, link to a separate JavaScript file referenced in the "src" attribute. In our example, it links to the script.js file you created. This is where the greet() function will be defined. The best placement for the script element is usually just above the body tag because scripts can reference elements on the page and will fail if the page isn't loaded yet

Add JavaScript

- Select your script.js file to edit
- Define the greet function:

```
function greet() {
    var firstName = document.getElementById("firstName").value;
    alert("Greetings and Salutations! " + firstName);
}
```

Code Explanation

- JavaScript's getElementById function finds an element on the page having an id attribute matching the argument; in this case, it gets the input element we defined with id="firstName"
- We want the value of the input element, not the element itself, so we need .value
- Alert creates a pop-up displaying a string

Configure the Debugger

In Visual Studio Code, debugging configurations are stored in a launch.json file which enables selecting various configurations from the debug menu. The Chrome debugger extension can be configured to launch from a file or a host.

Add Configuration: from menu Debug --> Add Configuration. Select "Chrome: Launch" from the dropdown. This creates and opens the launch.json file with the entry:

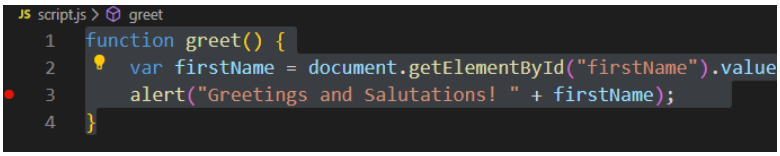
```
{
      "type": "chrome",
      "request": "launch",
      "name": "Launch Chrome",
      "url": "http://localhost:8080",
      "webRoot": "${workspaceFolder}"
  },
```

- I had trouble running the debugger with the above configuration because the 8080 port used in the localhost url is not the one my system was using. Instead of trying to start from the url add another configuration to start from the index.html file
- Debug --> add configuration --> Chrome: Launch will add a second entry to launch.json. Modify it to open index.html by changing it to this:

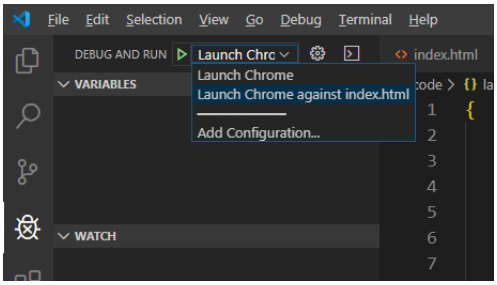
```
{
      "type": "chrome",
      "request": "launch",
      "name": "Launch Chrome against index.html",
      "file": "${workspaceFolder}/index.html"
 }
```

Start Debugging

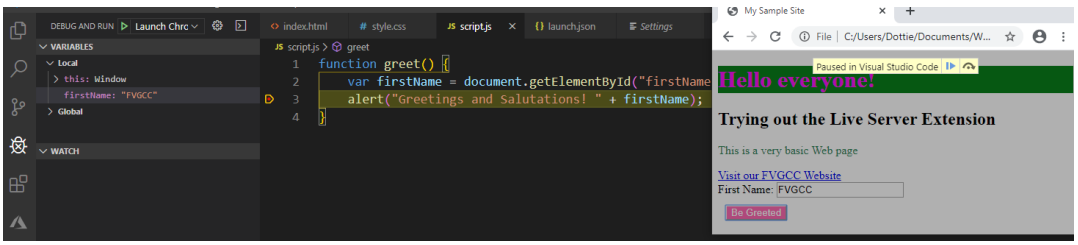
- **Set a breakpoint** in the script.js by clicking to the left of a line number. Set it on the alert line; you will see a red dot to mark the breakpoint



- **Start the debugger:** click the little "bug" icon in the left margin in VS Code to get the Debug and Run toolbar to display at top. Select Launch Chrome against index.html to select the configuration. Then F5 to start debugging

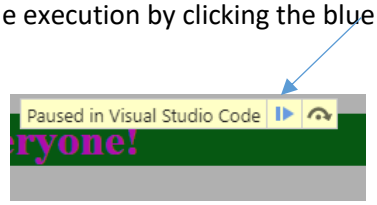


- **Call the JavaScript:** enter a different First Name or leave as is; click the button. The Browser shows "Paused in Visual Studio Code" at the top of the page. Visual Studio Code opens the JavaScript file and shows the point where code is stopped. In the left panel, you see the local variables, including the value of firstName. Hovering over the variable in the code also shows the value



- **Change the Value of a variable:**

    - In the Variables --> Local area, right-click firstName, choose “set value” and change to something else. Hover over firstName in the code and see that it has been changed
- Continue execution by clicking the blue arrow in the Browser next to “Paused in Visual Studio Code”



- You will see the new value of the first name displayed

Bootstrap Features in Visual Studio Code

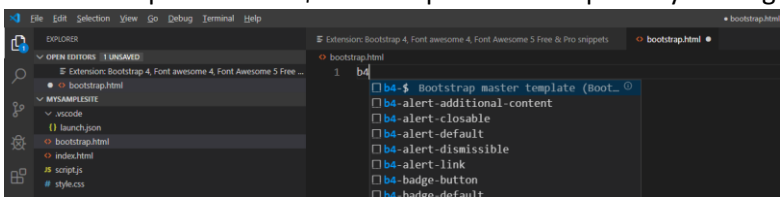
In our previous projects in Glitch we added Bootstrap to make our Web sites responsive to different size devices, add functionality, themes, and more.

Visual Studio Code provides an Extension for Bootstrap that makes it easy to incorporate

Add the Bootstrap Extension:

Extensions icon in left margin --> in search window type “Bootstrap 4 font” --> select the Extension named “Bootstrap 4, Font awesome 4, Font Awesome 5 Free & Pro Snippets” by Ashok Koyi and install it

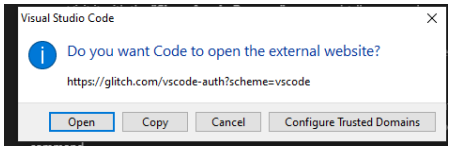
Add Bootstrap to a new HTML Page

- Click the New File icon next to the name of your site in Explorer; name the file bootstrap.html
- Edit the new empty bootstrap.html file. On the first line type *b4* to get a list of all the code snippets for Bootstrap. Select the $ Bootstrap master template by clicking it in the list
- 

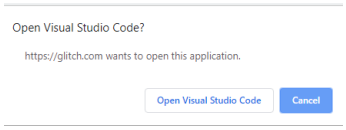
- This creates a full HTML page with all the required structure including the links to the Bootstrap CSS and JavaScript
- Add a few Bootstrap features:
    - **Nav menu**: Position cursor just below <body> and type *b4-nav* and select b4-nav-complete
    - **Jumbotron**: Position cursor below the last ; type b4-jumbotron and select b4-jumbotron-default
- **View the page:** right-click and select “Open with Live Server”
- **Experiment!**
    - Edit the elements you added; create some new ones
    - **Switch themes**:
        - Bootswatch.com; select a theme you like, click “download” and choose “bootstrap.css” from the list
        - Copy or move the file from the downloads folder to your Web site folder (which is something like: C:\Users*myname*\Documents\WebProjects\mySampleSite). It should appear in your list in the Explorer view in VS Code.
        - Change the stylesheet link to point to your file: comment out the original stylesheet link; add `<link rel=”stylesheet” href=”bootstrap.css”>`
        - ```
          <!-- Bootstrap CSS -->
          <!--  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.
          iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous"> -->
           <link rel="stylesheet" href="bootstrap.css">
          ```
          ```

Connecting Visual Studio Code to Glitch

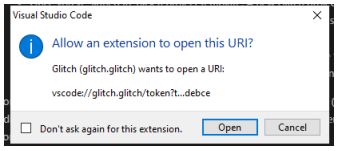
- Log into your GitHub Account (https://github.com
- Sign into Glitch (https://glitch.com) using the GitHub sign-in option
- Add the Glitch Extension to Visual Studio Code:
    - Select the Extensions icon in leftmost panel, type “Glitch” in search bar
    - Select the one published by Glitch with the Glitch logo and click “Install”
    - Review the documentation that shows in the main window when the Extension is selected, especially the “Quick Start” section
- Connect to Glitch from VS Code:
    - Cmd-shift-p or ctrl-shift-p to bring up command palette in VS Code
    - Select Glitch: Show Commands
    - Select Sign In --> Sign In on Glitch.com



    - ‘Open’ to initiate a request to glitch.com; you will then see this prompt:



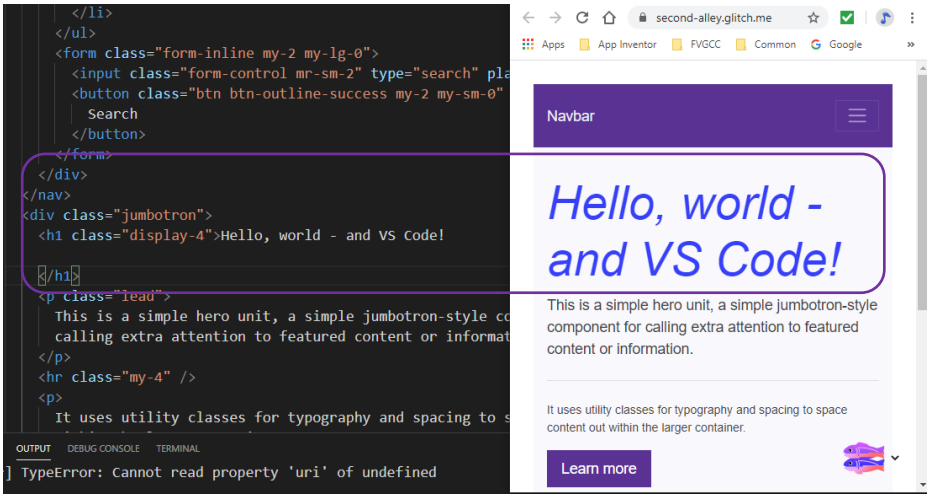
    - “Open Visual Studio Code” sends the request back to your machine:



    - “Open” brings you back to Visual Studio Code where you will see a message in the bottom right: 

Use VS Code to open your Web Project that you created in Glitch

- Open Command Palette (ctrl/cmd+shift+P) again and select Glitch: Show Commands, then Open Projects. You should see a list of your Glitch projects!
- Select your Web site project to open it in VS Code editor
- As you make changes to a file in VS Code it will show up in Glitch.


second-alley.glitch.me
Apps
App Inventor
FVGCC
Common
Google
Navbar
Hello, world - and VS Code!
This is a simple hero unit, a simple jumbotron-style component for calling extra attention to featured content or information.
It uses utility classes for typography and spacing to space content out within the larger container.
Learn more
OUTPUT
DEBUG CONSOLE
TERMINAL
TypeError: Cannot read property 'uri' of undefined