

- OpenClassrooms courses
 - What is on the page
 - -> you have the title of the course
 - -> then a video
 - -> under the video is text -> that text is what the person in the video is reading off of
 - -> then there are prerequisites
 - -> then there is a table of contents
 - -> you can download the videos from the course
 - The process we use is
 - -> write out the table of contents
 - -> then shell out our ones according to them
 - -> don't download the videos, it's a waste of time and storage
 - The suggested process is
 - -> to through the video
 - -> read through the material
 - -> do the quizzes
- Contents
 - Part #1 - Set Up Your Environment
 - 1. Install Your Front-End Development Tools
 - 2. Edit Your First Lines of Code With Visual Studio Code
 - 3. Explore Additional Visual Studio Code Features
 - 4. Test Your Code With Chrome and Firefox DevTools
 - -> each of these is called a chapter
 - -> and at the end (of this course) there is a quiz
- **1. Install Your Front-End Development Tools (VS code / Firefox / Chrome installations)**
 - -> this course is about installing, configuring and using tools
 - **VS code** -> editor, file explorer, version control, command palette
 - For testing and analysing projects -> **browsers have developer tools (you will need multiple browsers)**
 - **Chrome**
 - **Firefox**
 - **-> this chapter is installing VS Code, Firefox and Chrome**
 - -> VS code
 - <https://code.visualstudio.com> download the operating system
 - complete the installation
 - it's an app on MacOS
 - -> you need to install git if there are certain technical challenges
 - **-> the instructions are to launch install it with the Stable build**
 - -> if needs to be connected to git and you need to see 'clone github repository' on the home screen
 - -> Chrome
 - install Chrome <https://www.google.com/chrome/>
 - this is a more simple installation than VS code
 - -> Firefox
 - install Firefox <https://www.mozilla.org/en-US/firefox/new/>
- **2. Edit Your First Lines of Code With Visual Studio Code**
 - **Introduction / about -> this chapter is on VS code**
 - -> VSCode is an **IDE Integrated Development Environment**
 - -> it contains
 - a code editor (**it's like MS word for code**)
 - a file explorer <- connecting to the files on the computer
 - a built-in terminal <- so you can remain in the same window

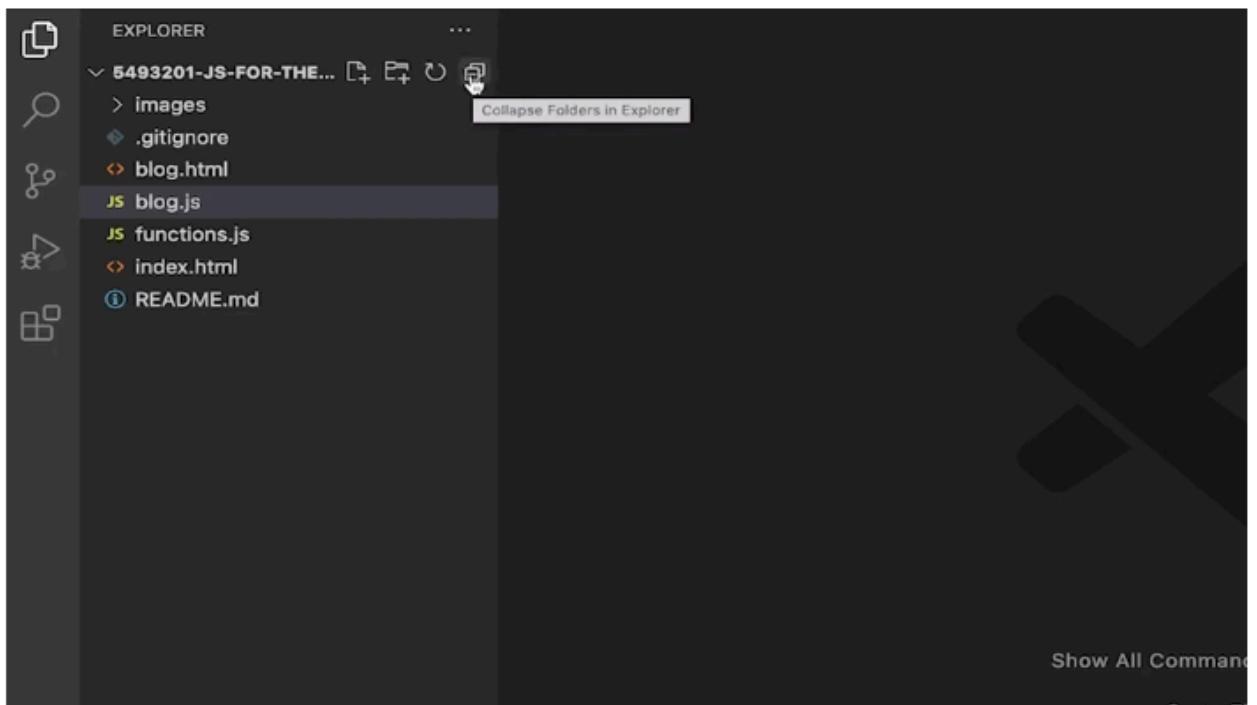
- -> **version control**
 - **VS code integrates with git <- connecting to files in the cloud**
- -> the command palette
 - -> you can learn shortcuts
- -> extensions
 - -> there are extension communities around VS code <- connecting to other developers
- -> then combining all of the elements
 - the next two chapters are
 - configuring VS code
 - using the editor
 - managing files / work folders
 - leveraging the command palette
 - the terminal
 - git
 - VS code with extensions
- -> VS code is an IDE which combines connections to other developers, code in the cloud and files on the computer
- **Configuring VS code**
 - -> 'screencast' is the same as a screen recording
 - -> configuring VS code
 - -> **manage (settings icon) in the bottom left hand > themes > colour themes (Monokai theme)**
 - -> **settings > commonly used > font size <- she's changed this from 12-15**
 - -> **help > get started**
 - -> **there are different tabs at the top of VS code**
 - **VS code is where the code / websites are written**
 - **Chrome / firefox is where they are tested**
 - -> **it's like a problem solving process -> but the problem we're trying to solve is "how do we create this website?"**
 - -> after the video are a lot of screenshots -> which are the same as the video -> but in picture form
 - -> **help <- this is where you can search for whatever feature you want in the IDE**
- **Exploring the code editor**
 - -> getting the code into a code editor
 - -> **github is where the files etc are stored in the cloud**
 - **to clone a repo in VS code**
 - **get a link for the repo**
 - -> to the git repo <https://github.com/OpenClassrooms-Student-Center/5493201-js-for-the-web-gulp> and get a link to clone it
 - -> **code > HTTPS > copy the URL (HTML version)**
 - **then paste the link into VS code (clone git repository) > paste in the link**
 - **pick somewhere to clone a local copy of the repo**
 - exploring the cloned repo in VS code
 - -> one of the files is blog.js
 - -> a file in JavaScript
 - -> you can open the different files which were cloned
 - -> **each colour of the text in the IDE corresponds to a different element of the code**
 - -> **line numbers on the LHS of the IDE**
 - -> **these are used for spotting errors and collaborating with other**

developers

- -> arrows next to sections of code
 - these are for 'code folding', for organisation
- -> autocompletion
 - -> function <- and then suggested code appears
 - -> she's doing this in the JS file which she has open
 - -> **there is an appearance feature called zoom**

- **Managing your work folders**

- managing file extensions
- there are folders nested in folders in the cloned repo files which are open in VS code
 - -> the nested folders can be accessed from VS code
 - -> **you don't need to go into finder to manage those files -> you can add / manage them from the IDE**



- -> there is a root directory there
- -> she's done an example with a new <HTML> file
- -> **.ts, .js, .html**
- -> you can also delete the file (command delete e.g)
- -> **command back space is delete a file on a mac -> and delete the entire row when you are typing text**
 - **the same shortcut you use to delete a line when you type can be used to delete a file in finder**
- -> she's done another example where she's made a new folder
- -> refresh explorer is for new files which come in
- -> **collapse folders (this is the last option -> which collapses all of the open folders)**

- **Leverage the command palette**

- -> bottom left settings icon > **command palette (shift command p)**
- -> different options and functions
- -> clicking on the files opens them (all of them)
- -> **shift command p > close all editors**
 - -> **you can also new file**
 - -> **you can create a Python file**

- -> but in an IDE (compared to a Jupyter notebook), you have more ready access to the terminal
- -> which is more linked to the file structures of your machine

- **Let's recap!**

- topics covered
 - -> installing VS code, chrome, firefox
 - -> VS code -> editor / file explorer / command palette

- **3. Explore Additional Visual Studio Code Features**

- **Handle the Terminal**

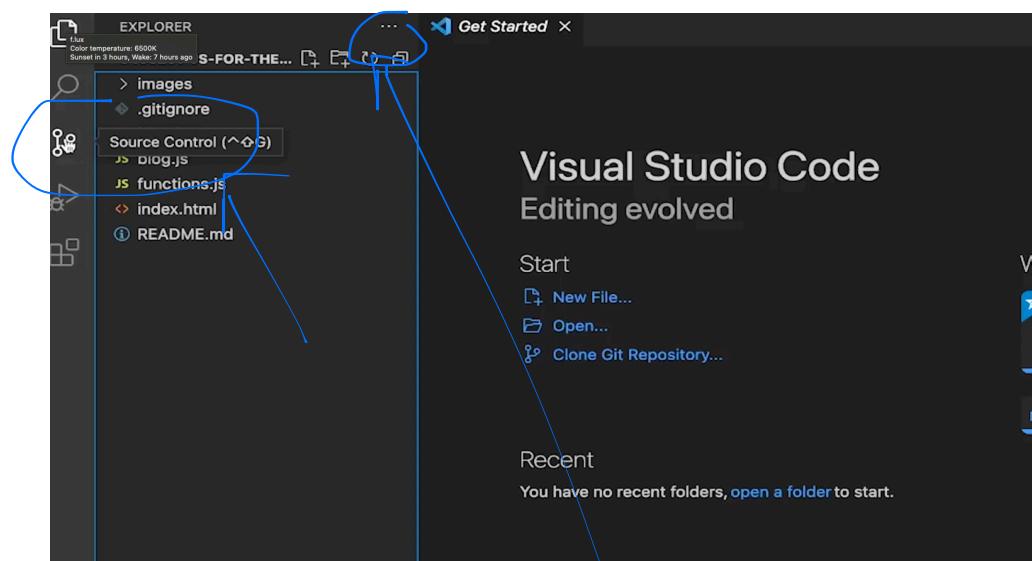
- -> the terminal or command prompt
- -> **VS code has a terminal in it**
- -> **command shift p > create new terminal in active workspace**
 - -> this creates the terminal at the bottom of where we are currently working

A screenshot of the Visual Studio Code interface. The top half shows the Explorer sidebar with files like blog.js, functions.js, index.html, and README.md. The main editor area displays a JavaScript file named blog.js with several lines of code. Below the editor is a tab bar with PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, with TERMINAL selected. At the bottom of the screen is a terminal window showing a command prompt: 'milapaul_0c@mp 5493201-js-for-the-web-gulp %'. A blue arrow points from the text 'this creates the terminal at the bottom of where we are currently working' to the terminal window.

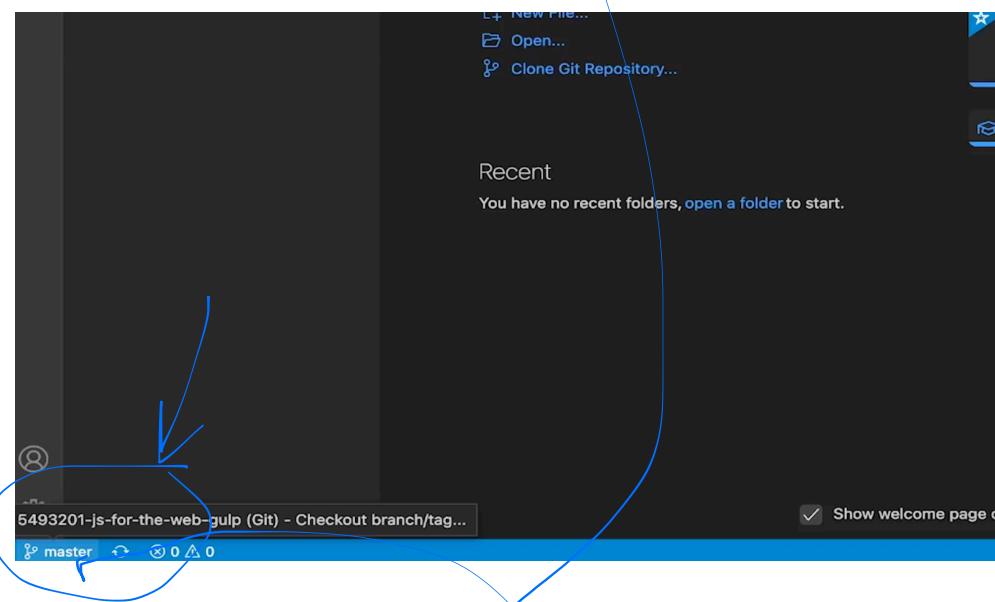
- -> **you can split the terminal (if we are working in two different file directories)**
- -> and then in that terminal you can cd into different directories
- -> the power shell / command prompt
- -> integrating all of the tools into one makes it easier to manage, to increase productivity

- **Use Git**

- -> she has cloned the git repo into VS code
- -> the commands on the LHS of VS code
- -> **source control**
 - this is under git



- -> bottom LHS of the page -> this the branch we are working on in git



- then she makes a local branch in order to edit the code
 - > branch > create branch
 - -> you can also do this from the command palette
 - > then she names the branch -> and this changes
 - > so we have the version of the code on the cloud -> then clone it locally
 - > then change it in our own branch -> then push that version back up to the cloud for version control
 - > -> and this is how we have entire teams of developers writing software
- -> then she makes changes to the file in the branch
 - -> then saves them
 - -> in the source control section of VS code -> the change shows up as a notification on the source control

```

EXPLORER
5493201-JS-FOR-THE-WEB-GULP
> images
> .gitignore
blog.html M
blog.js
functions.js M
index.html
README.md

functions.js > newsletterButton.addEventListener('click') callback
1 const newsLetterButton = document.getElementById('newsletter-button');
2 const newsLetterSection = document.getElementById('newsletter-section');
3 const newsLetterEmail = document.getElementById('newsletter-email');
4
5 let sentEmail = false;
6
7 newsLetterButton.addEventListener('click', () => {
8   let newParagraph = document.createElement('p');
9   if (sentEmail) {
10     return;
11   }
12   const findAlert = document.getElementById('alert');
13   if (findAlert) {
14     findAlert.parentElement.removeChild(findAlert);
15   }
16   if (!newsLetterEmail.value) {
17     newParagraph.textContent = 'Please enter a value!';
18     newParagraph.classList.add('alert-danger', 'w-100', 'p-2');
19     newParagraph.setAttribute('id', 'alert');
20     newsLetterSection.appendChild(newParagraph);
21   }
22
23   newParagraph.textContent = 'E-mail address successfully submitted!';
24   newParagraph.classList.add('alert-success', 'w-100', 'p-2');
25   newsLetterSection.appendChild(newParagraph);
26   newsLetterSection.children[0].children[0].removeChild(newsLetterEmail);
27   newsLetterSection.children[0].children[0].removeChild(newsLetterButton);
28   sentEmail = true;
29 });

```

- when you click on source control, you can see the changes
 - -> to "stage" the changes -> you click + on one of them
 - -> so the file is changed -> this is saved -> then this is saved to the local branch -> then this is committed back to github
 - -> she types in a message to describe the changes which she made

```

SOURCE CONTROL
Message (Enter to commit on 'd...')

Changes 2
blog.html M
JS functions.js M

JS functions.js > newsletterButton.addEventListener('click') callback
1 const newsLetterButton = document.getElementById('newsletter-button');
2 const newsLetterSection = document.getElementById('newsletter-section');
3 const newsLetterEmail = document.getElementById('newsletter-email');
4
5 let sentEmail = false;
6
7 newsLetterButton.addEventListener('click', () => {
8   let newParagraph = document.createElement('p');
9   if (sentEmail) {
10     return;
11   }
12   const findAlert = document.getElementById('alert');
13   if (findAlert) {
14     findAlert.parentElement.removeChild(findAlert);
15   }
16   if (!newsLetterEmail.value) {
17     newParagraph.textContent = 'Please enter a value!';
18     newParagraph.classList.add('alert-danger', 'w-100', 'p-2');
19     newParagraph.setAttribute('id', 'alert');
20     newsLetterSection.appendChild(newParagraph);
21   }
22
23   newParagraph.textContent = 'E-mail address successfully submitted!';
24   newParagraph.classList.add('alert-success', 'w-100', 'p-2');
25   newsLetterSection.appendChild(newParagraph);
26   newsLetterSection.children[0].children[0].removeChild(newsLetterEmail);
27   newsLetterSection.children[0].children[0].removeChild(newsLetterButton);
28   sentEmail = true;
29 });

Stage Changes

```

- -> and the changes have been submitted
- **Make the Most of VS Code With Extensions**
 - -> plugins (extensions)
 - -> <https://marketplace.visualstudio.com> <- this contains the full list of them
 - suggested ones
 - -> colour picker <- colour wheel for colour codes
 - -> open in browser <- opens the file in the browser
 - -> to install them
 - -> **because you are adding another block (an extension) to the IDE**
 - -> search for the one you want to add
 - -> **it's like the app store but for the IDE**



- **Discover additional VS Code extensions**

- -> see cheat sheet notes
- -> for a list of extra suggested extensions

- **Let's Recap!**

- -> this chapter was
 - integrated terminal in VS code
 - git repos (creating branches)
 - installing extensions
- -> the next chapter is testing the code

- **4. Test Your Code With Chrome and Firefox DevTools**

- **different approach -> how to notes**

- **bootcamp isn't about understanding, it's about knowing how to do something**
- **listen for how to do xyz -> make it into how to notes**
- **then use those to do the how to tasks and submit them**
- **MVP**

- **About / intro**

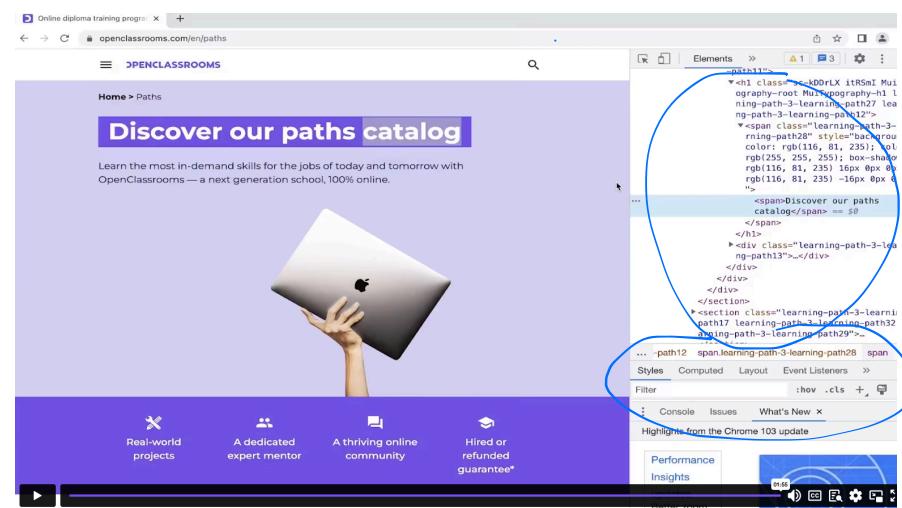
- -> chrome and firefox developer tools
- -> the inspector -> styles
- -> console (troubleshooting)

- **Explore the Inspector - HTML**

- -> command shift c in a browser
- -> it's the HTML of the website
- -> click on parts of it -> it shows the website
- -> right click on parts of the website > inspect

- **Explore the Inspector - Styles**

- -> styles inspector (css -> not the structure, the style of the webpage)
- -> in the inspector, you have HTML on the top and CSS on the bottom
- -> you can change in the styles inspector -> it doesn't change the actual website



top is HTML bottom is CSS

- **Study the Console**

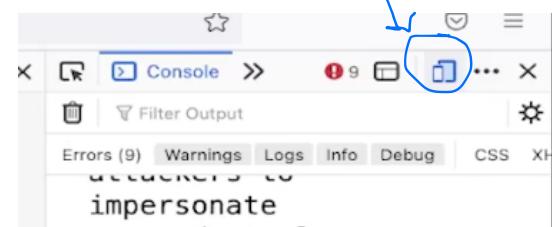
- -> the console
- -> >> > console, or inspector > console
 - -> this isn't a shortcut it's found in the GUI
- -> it contains warnings
 - self-cross site scripting attack
 - -> don't paste code into the console you don't understand (security, people launching attacks through the console)
 - -> breaking the webpage

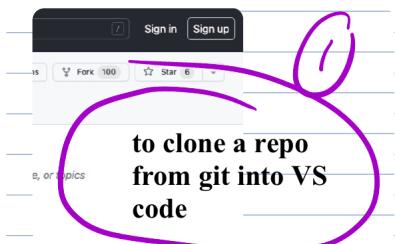
- **Activate Mobile Mode**

- -> there is a mode to test the project on mobile mode
- -> the menu bar at the top of the console
- -> the screen moves to a mobile (you don't have to test it on an actual phone)

- **Let's Recap!**

- -> inspector
- -> the console
- -> mobile mode (simulating browsing from a mobile device)





- clone
a repo

(2)

- to edit files once in the IDE

- edit files

(3)

To type on commands or do something in VS code

command palette to execute file or folder (also an entire line on a text editor)

/ type commands
in IDE

- the command palette (ctrl + enter a shortcut and type in instructions)
- shift command p to get it up → when you have files open

(4)

command shift p > create terminal

- integrated
terminal

To get an integrated terminal in VS code

- to create a branch and submit new changes using git in VS code | (5)
→ we cloned a repo into VS code from git - now →

dev tools, click the console

(6)

To get the console to show up In a b
element -> the console will show up

1

console

/
Creating
branches

→ To clone a repo from git into VS code

- go to the repo on git → get a link for the repo

A screenshot of a GitHub repository page for 'OpenClassrooms-Student-Center / 5493201-js-for-the-web-gulp'. The page shows a list of files and a 'Clone' section with options for 'Local', 'CodeSpaces', and 'Clone'. A circled 'Clone' button is highlighted. Below it, there's a 'HTTPS GitHub CLI' field containing a URL: 'https://github.com/openClassrooms-Student...'. A callout bubble points to this URL with the handwritten note: 'generate an https (in this case) link to the repo'.

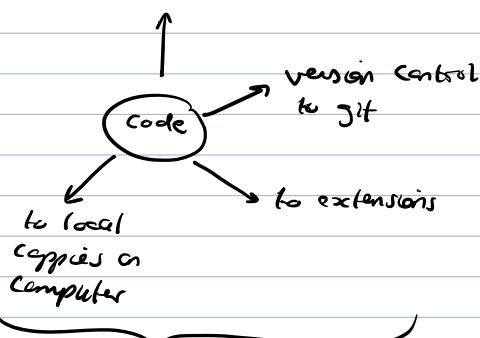
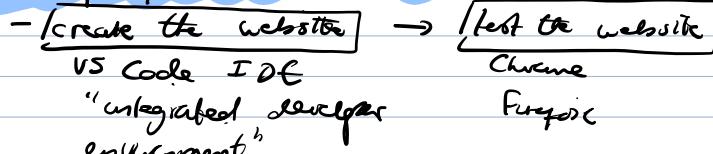
to clone a repo from git into VS code

- get that repo into vs code

- VS code welcome page > clone git repository > paste in the link from GitHub

- choose a directory to store a local version of the file

- software for front-end web dev.



- organised in different tabs / colour schemes
- help tabs for searching whatever you need
- code folding for organisation

"Integrated" developer environment
→ vs a text file

this has access to console

(2)

- To edit files once in the IDE

a terminal within the IDE, rather than using a Jupyter Notebook

```

1 const api = 'https://us-central1-the-new-york-times.cloudfunctions.net/api/v1/articles';
2 const loadButton = document.getElementById('load');
3
4 function getRequest(url) {
5   return new Promise((resolve, reject) => {
6     let request = new XMLHttpRequest();
7     request.open('GET', url);
8     request.onreadystatechange = () => {
9       if (request.readyState === 4) {
10         if (request.status === 200) {
11           resolve(JSON.parse(request.responseText));
12         } else {
13           reject(new Error(`Request failed with status ${request.status}`));
14         }
15       }
16     };
17     request.send();
18   });
19
20 async function getBlogPost() {
21   const titlePromise = getRequest(`${api}/titles`);
22   const loremPromise = getRequest(`${api}/lorem`);
23   try {
24     let [titleResponse, loremResponse] = await Promise.all([titlePromise, loremPromise]);
25     document.querySelector('#main').innerHTML = titleResponse.data[0].content;
26   } catch (error) {
27     console.error(error);
28   }
29 }
30
31 loadButton.addEventListener('click', getBlogPost);
32 
```

this is the cloned git repo open in the IDE
→ you can manage files (new files, new folder) etc. - from the IDE, you don't need to open finder

+ new file (.js, .html, .css, .py)
+ new folder

⑤ refresh explorer for new files
→ collapse all folders open for larger projects

v ... → ...

v ...

→ they are languages / file types
→ command backspace is delete a file or folder (also an entire line in a text editor)

③

To type on commands or do something in VS code

- the command palette (aha, enter a shortcut and type in instructions)
 - shift command p to get it up → when you have files open
 - you can type
 - add new file
 - create new file
 - the help tab at the top is to search the entire IDE to look for something
 - the Shift + Command + P command palette is to do something / perform a function within it
- to get an "integrated" terminal, aka one which opens in VS Code
 - Command + Shift + P > create new terminal in active workspace

④

To get an integrated terminal in VS code

→ in other words, at the bottom of the file we are currently editing

- you can get two different ones open for different file directories at the same time (is like a split screen feature)

```

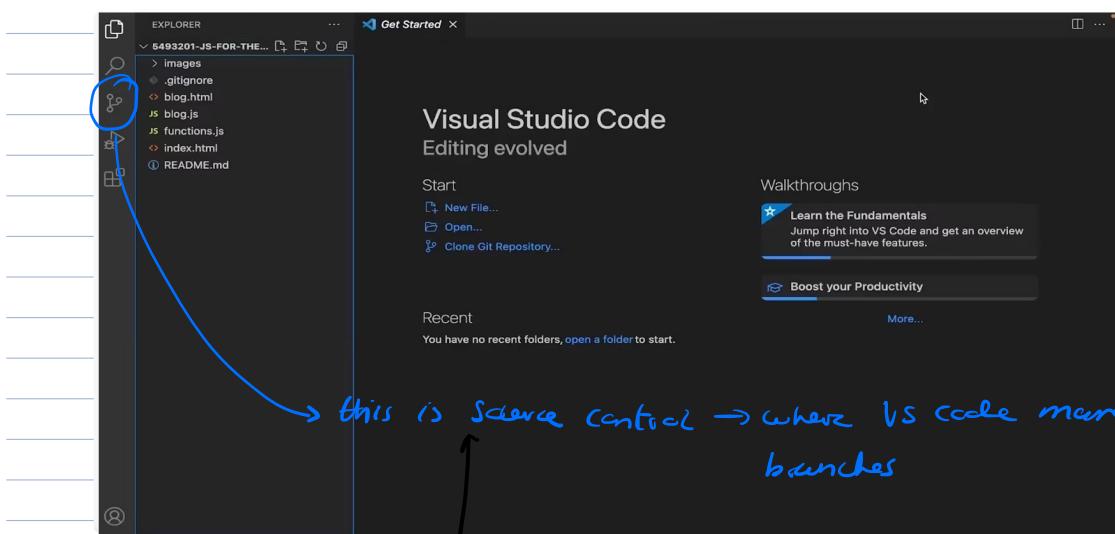
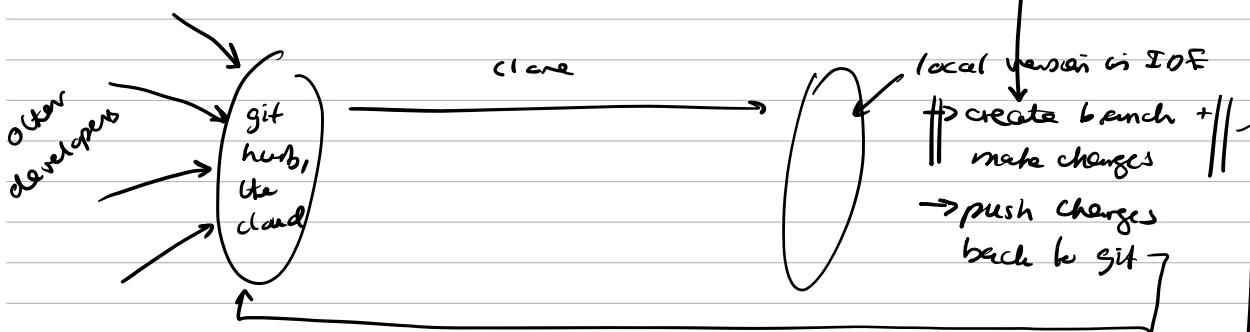
EXPLORER JS blog.js ...
5493201-JS-FOR-THE...
images .gitignore
blog.html
JS blog.js
functions.js
index.html
README.md

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
milapaul_oc@mp 5493201-js-for-the-web-gulp %

```

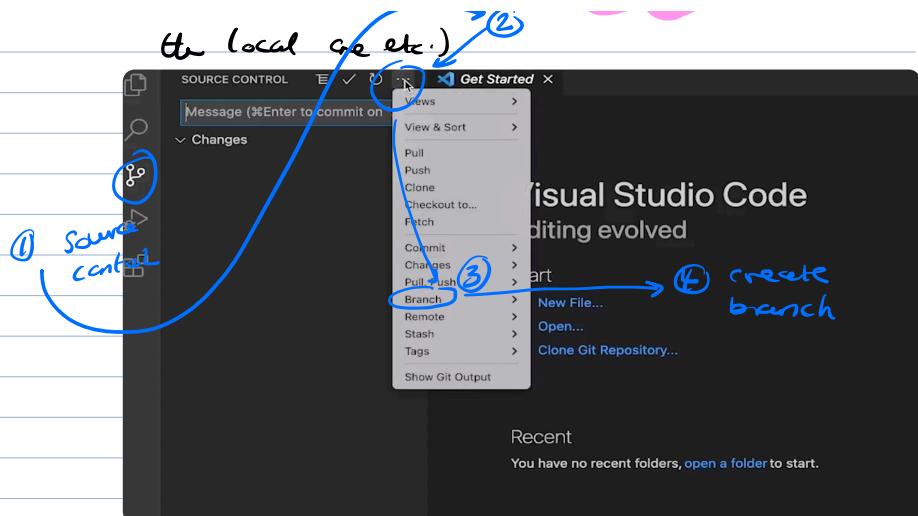
this creates another terminal here

- To create a branch and submit new changes using git in VS code | ⑤
 → we cloned a repo with VS code from git - now



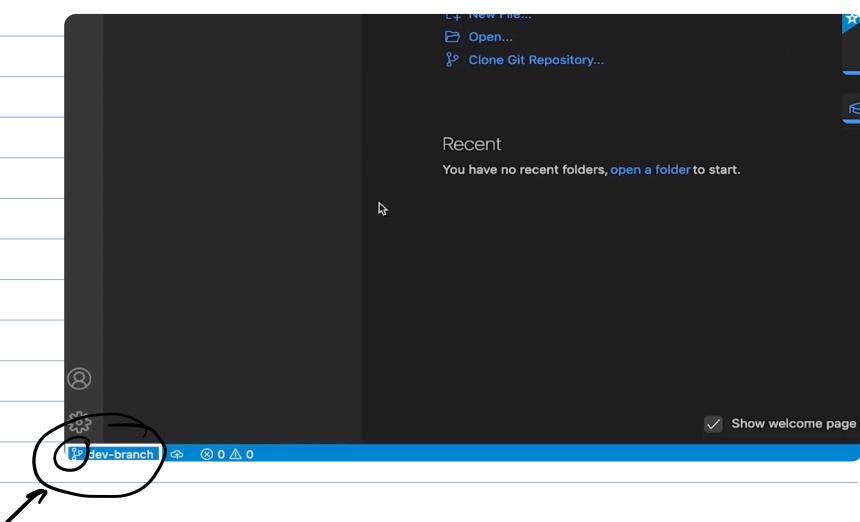
makes a local branch in order to edit the code
 → in Source Control - which Source are we working from (git)

METHOD 1:
THROUGH
SOURCE
CONTROL



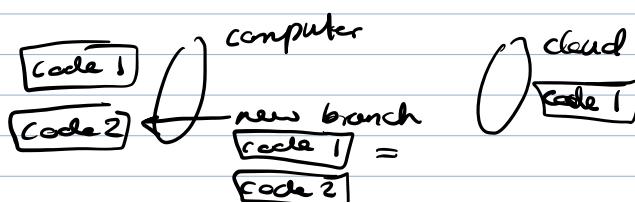
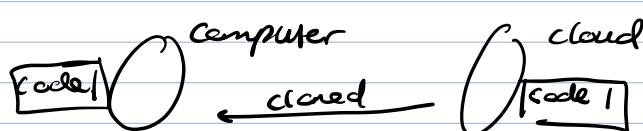
METHOD 2:
COMMAND
PALETTE
AREA
COMMAND
SHIFT P

How you know IT'S WORKED:

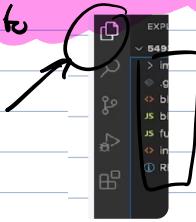


this is the current branch
→ it will have changed and the name will be what
you entered

So now:

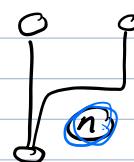
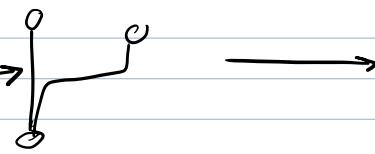


#2 Makes changes to the files and saves them (on the new branch)
→ she goes back to



← then clicks on a .html file
changes it
→ and saves the S's

→ How you know it's worked:

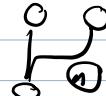


← n = number of changes made

#3 Stage the changes - when you appear them before they are definitely committed to that branch

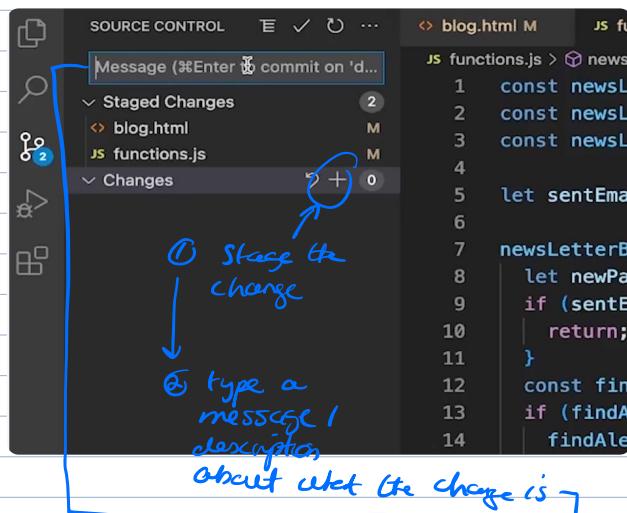


→ in source control, but not



→ click on it and will see a list of those changes

→ you are making changes after clicking + committing them



Message (Enter commit on 'd...)

Staged Changes

- blog.html
- JS functions.js

Changes

① Stage the change

② type a message / description about what the change is

blog.html M JS fu

JS functions.js > news

```
1 const newsL
2 const newsL
3 const newsL
4
5 let sentEma
6
7 newsLetterB
8 let newPa
9 if (sentE
10   return;
11
12 const fin
13 if (findA
14   finde
```

③ click enter

→ then those changes will be submitted and categorised by the IDE, so when people make changes to the versions of the code, you will be able to see what it

and vice versa

- extensions in the IDE create the app store but for plugins



you can add a block to the existing IDE

→ looks like the app store, but for VS code

→ suggested ones - colour picker (color wheel for color codes)

- opens the file in the file browser

→ extra suggested ones

- Live Share - collaborate with others **in real time**, working on the same files simultaneously, allowing for easy long-distance pair programming
- GitLens - **supercharge** VS Code's built-in Git functionality
- Stories - like Instagram stories, but for code!
- Search Hero - all developers use **internet search for answers**, so Search Hero builds in multi-search engine search and translation tools → **be smart about for help**

- dev tools, aka the console

(6)

To get the console to show up In a browser, right click somewhere > inspect element → the console will show up

