FEND 2021

0] Introduction To Front End Development

-0.1- Understanding how is the internet works

-0.2- Understanding how a website works

-0.3- Understanding FEND technologies

-0.4- Understanding job roles and titles related to FEND

-0.5- Differentiate FEND from Back End Development

-0.6- Understanding how is the browser displaying websites

-0.7- Understanding how to set up a development work environment

-0.8- Learning how to use a text editor to write your code

-0.9- Learning how to use Git the source code manager

-1.0- Learning how to use chrome developer tools

1] HTML

-1.1- What is HTML?

-1.2- The history of HTML

-1.3- The anatomy of HTML Element

-1.4- The structure of HTML Document

-1.5- Block vs inline HTML Elements

-1.6- Text vs Structural HTML Elements

-1.7- Style Element Vs Inline Styling

-1.8- Static, Dynamic & Responsive Web Designs

-1.9- HTML Page Layout

-2.0- HTML5 & Semantic Elements

2] CSS

-2.1- What is CSS?

-2.2- Inline, Internal & External CSS

-2.3- CSS Position Property

-2.4- CSS Display Property

-2.5- CSS Media Queries

-2.6- CSS Flexbox Module

-2.7- CSS Grid Module

-2.8- CSS Bootstrap Library

-2.9- CSS Specificity

-3.0- CSS Import Keyword

3] JavaScript

-3.1- What is JavaScript?

-3.2- The History of JavaScript

-3.3- Variables

-3.4- Primitive Data Types

-3.5- Strings

-3.6- Expressions

-3.7- Loose 3

-3.8- Comments

-3.9- Functions

-4.0- Arguments

-4.1- Return Values

-4.2- Variable Scope

-4.3- Coding Conventions: Spacing

-4.4- The if statement

-4.5- The if/else statement

-4.6- Ternary operator

-4.7- The while loop

2-4.8- The for loop

-4.9- The break statement

-5.0- Comparison Operators

-6.1- Logical Operators

-6.2- Truthy vs Falsey

-6.3- The array data type

-6.4- Objects

-6.5- Built-in Objects

-6.6- ECMAScript 2016 Language Specification

-6.7- ES6 Variables

-6.8- ES6 Arrow Function

-6.9- ES6 Classes

-7.0- Document Object Model DOM

Chapter 01
Introduction To Front End Development

0.1- Understanding how is the internet works

Introduction to the Internet

- **1969 --** *The Internet started as a military project of the USA government in what is called ARPANET (Advanced Research Projects Agency NETwork) to connect more than one computer.*

- **1970 --** *Two of the pioneers(fathers of the internet) Vint Cerf & Bob Kahn had taken a new step toward what we know as the Internet by designing NCP (Network Control Protocol) allowing data to be transmitted between different computers.*

- **1972 --** *Ray Tomlinson invented the format of the current Email addresses by using @ to separate the user name from the hostname.*

- **1974 --** *the same two internet fathers Vint & Bob designed the TCP (Transmission Control Protocol), a decentralized network connecting computers allowing them to transmit the data as small pieces called packets or blocks.*

- **1984 --** *DNS(Domain Name System) is designed to allow networking using user-friendly names instead of IP addresses which are hard to remember.*

- **1991 --** *the first web page was published to introduce the worldwide web. Check what was it like at this link >>>*

- *Also, in the same year, the first content-based search protocol was launched giving the power of checking any webpage content and not only the webpage name.*

- **1993 --** *the first graphical web browser was launched for the public and it was named Mosaic.*

- **1995 --** *SSL(Secure Socket Layer) was developed allowing secure transactions and purchasing online(the first thing that was ordered online was pizza, by the way). In the same year, JavaScript was introduced by Brendan Eich, by the name LiveScript.*

- **1996 --** *Hotmail was launched.*
- **1998 --** *Google was launched.*
- **2001 --** *Wikipedia was launched.*
- **2003 --** *Skype was launched.*
- **2004 --** *Facebook was launched.*

- **2005 --** *Youtube was launched.*
- **2006--** *Twitter was launched.*

So, what is the internet?

A lot of Internet users may use it daily but when it comes to understanding what it is, a lot of them will have no answer. You can think of the internet as a wire(yes, you read it right) and any computers connected to this wire can communicate with each other. Every device connected to this wire has a unique address called IP(Internet Protocol) address which is a bunch of numbers used to identify every device on the internet allowing them to find each other. Since using these numbers was very hard for average internet users, a DNS(Domain Name System) is designed to translate every IP address to an easy-to-reme5mber domain name.
Computers that are connected to the internet may be Servers(have a direct connection to the internet) or Client PCs(have an indirect connection through ISP which stands for Internet Service Provider).

What is WWW?

The World Wide Web *is one of the software layers used on the internet besides a lot of other software layers like mail exchange and file transfer. A collection of technologies and protocols allow users to surf web pages from a server-side to a client-side.*
Many people think that www means the internet, but we can consider it a software part of the internet.

0.2- Understanding how a website works

As we are here for some basic knowledge about web development, let's go deep into this world and find what's going under the hood while surfing any website.
The whole process started with the user opening the web browser to visit a web page, we are all familiar with the web browser but we may not have any awareness of what the browser does.

What does the browser do?

The browser is a piece of software(a program) with some cool capabilities:

- The browser has a GUI(Graphical User Interface) which allows a user to visualize web page content like text, images, videos ...etc.

- The browser has what is called an address bar where a user can type the URL(Uniform Resource Locator) for any website and it will be changed to its corresponding IP(Internet Protocol) using what is called DNS(Domain Name System) to reach the content of this specific webpage.

- The browser has access to the files of the requested web page by using what is called HTTP/S(HyperText Transfer Protocol/Secured) to communicate with the server-side asking for the specific files.

- The browser has a Rendering Engine(an independent piece of software) that is responsible for parsing or processing two kinds of files coming from the server, HTML and CSS files.

- The browser has a JavaScript Engine or interpreter(another independent piece of software) which is responsible for parsing or processing JavaScript files coming from the server.

- The browser has access to local storage & cookies on the user machine to store the needed data for faster interaction and offline usage of the requested website.

0.3- Understanding FEND technologies

The Terminology of Web Development

1- Client:

Any device with **a browser** and internet access through ISP(Internet Service Providers) is called a Client.

EX: Personal Computer, Laptop, Tablet, Smart Phone.

2- Browser:

A piece of software responsible for helping internet users to find web pages by using the address bar to type the **URL**(Uniform Resource Locator) then hit enter or go to **request** files from the **server** then when the server **response** is sent back, the browser visualizes these files inside its GUI(Graphical User Interface) in form of text, images, videos, ...etc. the whole process is done using what called **HTTP/S**(Hypertext Transfer Protocol/Secured).

EX: Chrome, Firefox, Safari, ..etc.

3- URL

URL stands for Uniform/Unique/Universal Source Locator, a text string designed to give every web page a unique address to easily be found and reached by users. A link is a common name for a URL.

EX: www.google.com

4- Request

A request is a process done by a browser to get the resource files (**HTML, CSS, JAVASCRIPT**) of a specific webpage using HTTP/S protocol.

A request is a simple text-based message consisting of a header and body supplying the server-side with the needed data to accept communication.

EX:

```
Request URL: https://mail.google.com/sync/u/0/i/s?hl=en&c=52
Request Method: POST
Status Code: 200
Remote Address: 172.217.19.37:443
Referrer Policy: no-referrer-when-downgrade
```

5- Server:

A Server is a computer(hardware) with more computing power and storage than a personal computer. As the client PC(hardware) has a browser(software), the server has its package of software(including database) helping it to communicate with the client-side.

6- Response:

a process is done by server software to send the resource files(HTML, CSS, JAVASCRIPT) of a specific webpage using HTTP/S protocol to a client-side browser.

7- HTTP/S:

HTTP/S(HyperText Transfer Protocol/Secured) is a network protocol that allows the client-side and the server-side to communicate in a certain way. You can think of it as a contract between them. HTTPS is the secured version of HTTP which allows specific types of secured connection, especially with money transactions online. Nowadays, secured connections are very encouraged to avoid data leaks and cyber-attacks.

8- HTML:

HTML(HyperText Markup Language) is the language used to build web pages. The word HyperText refers to the nature of HTML documents allowing it to use Hyperlinking from a webpage to another by clicking links. The HTML document is a plain text document structur1ed in a specific way and ends with the extension .htm or .html.

EX: to write a paragraph using HTML

```
<h1> Hello World!</h1>
Result: 1
                        Hello World!
```

9- CSS:

CSS(Cascading Style Sheets) is a language used to give a webpage its look(colors, fonts, ...etc) inside the browser GUI. Applying the CSS styles is done by linking HTML & CSS files together or putting these styles inside HTML document/elements.The CSS document is a plain text document structured in a specific way and ends with the extension .css.

EX: to give the paragraph in the last example a red color

```
<p2> Hello World!</p>
Result:
 Hello World!
```

10- JavaScript:

JavaScript is **a programming language** used for giving a webpage its dynamic functionality on the client-side. It is used mainly in the browser to enable manipulating web page content through the DOM.

11- Programming Language:

A program5

10.4- Understanding job roles and titles related to FEND

What type of developer do you want to be?

Let me help you a little by exploring different types of web developers then you can choose by yourself.

Web Designer
If "designer" is in the title, the job is designing. Deciding and implementing how websites look and work. "Web" is in the title because the job is specifically focused on the web. Specific skills would be design-tools-of-choice, HTML, CSS, and light JavaScript.

Front End Developer
This job is focused on HTML, CSS, JavaScript, and light backend work. Not design. The lack of "designer" in the title is intentional. Because the job doesn't require design, deeper skill in the other technologies is implied. You likely have a grasp on some concepts beyond the core technologies, for instance, regression testing or performance.
A synonym might be Front End Engineer. I tend to think of that as requiring a deeper and more specific skill set, possibly with a more narrow focus or at a higher level.
Technology-specific job titles may also be appropriate here, like "JavaScript Developer" or "JavaScript Engineer" for a job where that is primarily what needs to be done. Although, none of the front-end technologies live in a bubble so I generally prefer Front End developers.

UI Designer
This job is more about designing and less about implementation. Good at design-tools-of-choice with perhaps only light HTML and CSS skills. A synonym might be a Visual Designer.

UX Designer
A specific focus is on studying and researching how people use a site. Then ushering changes for the better through the system and testing the results. May not have or need any design or implementation skills. All jobs should care about user experience, but this job lives it.

Interaction Designer
Primarily design, just like a UI Designer, but with a specific focus on how things are used and movement.

Web Developer
This job is focused on back-end work and working with languages specific to the web, like PHP, ASP, Ruby, Python, etc. Medium skill in database/server work, medium skill in JavaScript, light skills in HTML. This is very53 different from a Front End Developer as there is little working with the design and heavier on programming concepts and concerns, like security and structure.
Synonyms could be Web Programmer or Web Application Developer.

Full-Stack Developer
This job is a combination of front and back-end work. Good crossover people are needed at organizations and this is a high-end job.

Content Strategist
Rather than working directly on implementation, this job is about the structural design of websites. Things like the taxonomies, metadata, scheduling, and analysis of content. A synonym might be Information Architect. They might work with people who work with content in a more general way like a Writer, Copywriter, or Editor.

IT Technician
This job works with actual computers and tech equipment. A hardware person.

Dev Ops
The job bridges the gap between IT and Developers. They handle things like server software, version control, deployment, build processes, and testing servers/processes. I wish this had a more job-title-y feeling to it. As it stands it sounds like what you would call the whole team of people with this job.

Product Manager
This job is about guiding the site as a whole (or a major feature of the site) toward a better future. Largely dealing with people and

planning. A Project Manager would be similar but smaller in scope and possibly a temporary role rather than a full job title.

Customer Service Representative

This job is about communicating directly with users of the site to provide help. Then triaging bugs/problems to the internal team. Also understanding/communicating the voice and vibe of the community around the site.

SEO Specialist

This is a big enough sub-industry that it can be its job. SEO Specialist role is to find and implement the keywords needed by search engines algorithms to index any webpage and make it appear as search results relevant to these keywords.

Source>>

0.5- Differentiate FEND from Back End Development

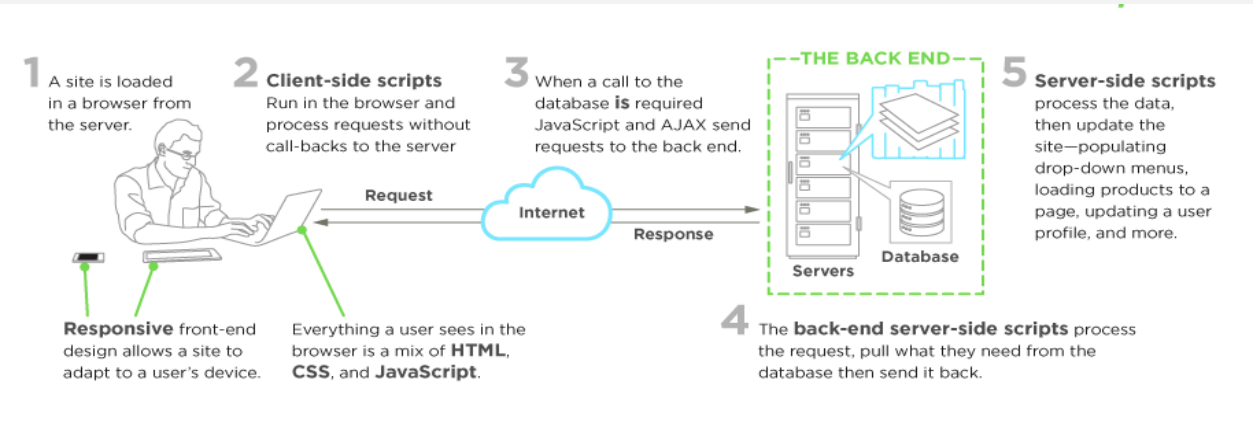
What is front-end development?

Front-end web development, also known as client-side development, is the practice of producing HTML, CSS, and JavaScript for a website or Web Application so that a user can see and interact with them directly. The challenge associated with front-end development is that the tools and techniques used to create the front end of a website change constantly and so the developer needs to constantly be aware of how the field is developing.

The objective of designing a site is to ensure that when the users open up the site they see the information in a format that is easy to read and relevant. This is further complicated by the fact that users now use a large variety of devices with varying screen sizes and resolutions thus forcing the designer to take into consideration these aspects when designing the site. They need to ensure that their site comes up correctly in different browsers (cross-browser), different operating systems (cross-platform), and different devices (cross-device), which requires careful planning on the side of the developer.

[Source>>]

Recap: front end development is all codes required to generate a UI(User Interface) a website visitor can interact with.



[source>>]

What is back-end development?

Back-end Development refers to the server-side of development where you are primarily focused on how the site works. Making updates and changes in addition to monitoring the functionality of the site will be your primary responsibility. This type of web development usually consists of three parts: a server, an application, and a database. Code written by back-end developers is what communicates the database

information to the browser. Anything you can't see easily with the eye such as databases and servers is the work of a back-end developer. Back-end developer positions are often called programmers or web developers.

There are two main build tools to learn front-end development:

1- Frameworks:

A front-end framework is a structural frame for development providing developers with a ready-to-use piece of code functionality with the ability to adapt this functionality with additional user-written code. In the world of web design, to give a more straightforward definition, a framework is defined as a package made up of a structure of files and folders of standardized code (HTML, CSS, JS documents, etc.) which can be used to support the development of websites, as a basis to start building a site.

EX: Vue.js, React.js, Angular.js

2- CMS:

CMS(Content Management System) is a system designed to help users create and manage their websites. A CMS helps webmasters manage the many different resources, content types, and various data that make up modern websites. As a developer, you can create a website without the need to write any code(cool thing?).
EX: WordPress, Joomla, Drupal, ...etc.

Now you must23 think about it and tell yourself why do not learn CMS and start to build websites(since no code is required)? The answer is since CMS is built upon a framework(a package of HTML, CSS, and JavaScript files), eventually, you need to learn the basics of these languages if you need to customize your website for your specific needs or to go anywhere with your front end development skills and career goals.

0.6- Understanding how is the browser displaying websites

Once the communication process is completed successfully, the client-side request ended with a response from the server and the files are ready to be processed, the rendering process is done in four stages:

[1] HTML Parsing Stage

1- Process HTML markup and build the DOM tree.

2- Process CSS markup and build the CSSOM tree.

[2] Render Tree Stage

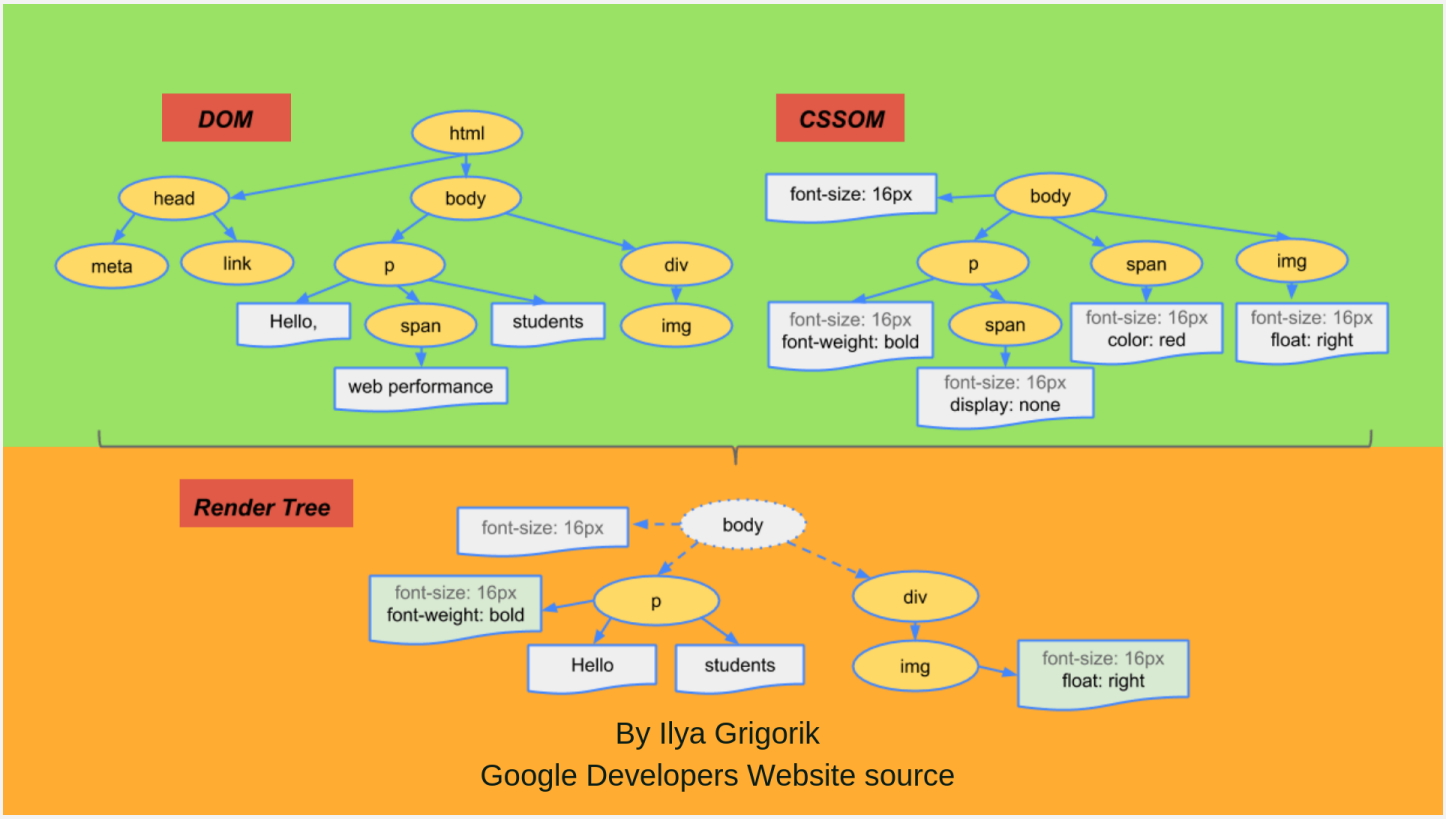
3- Combine the DOM and CSSOM into a render tree.

[3] Layout Stage

4- Run layout on the render tree to compute the geometry of each node.

[4] Paint Stage

5- Paint the individual nodes to the screen.



By Ilya Grigorik

Google Developers Website source

0.7-Understanding how to set up a development work environment

Set up your work environment

Before you can learn anything, you have to know what tools are needed for your development:

1- a text editor:

A text editor is a program that allows you to open, view, and edit plain text files. To write any code you need to use a text editor. As a developer, you may need some features to help you with your workflow, productivity and facilitate your life. Choosing the right editor is quite a useful step toward being a smart and productive developer. There are a lot of text editors out there but some of them are cool and useful like VSC(Visual Studio Code) , my favorite one. Sublime Text and Atom are also worth checking. Once you download VSC, you are ready to write your first HTML document.

2- a terminal:

A terminal is a type of CLI(Command Line Interface) used to type commands and run it(the very old computers were using this CLI).

So, why do we need to use terminals while learning front-end development?

***A lot** of web developer tools operate through the command line. Not everything can be downloaded & installed like an application, so an understanding of shell commands will save you the headache of learning how to install the latest new tools for developers. Also, it allows using Git the Source Code Manager/Version Control System. This is a very imperative thing to learn in today's development. Nodejs(A back-end framework) & NPM(Node Package Manager) also need a terminal.*

The needed shell for us as developers is BASH(Bourne Again Shell) and the preferable version control system is Git.

3- Google chrome developer tool

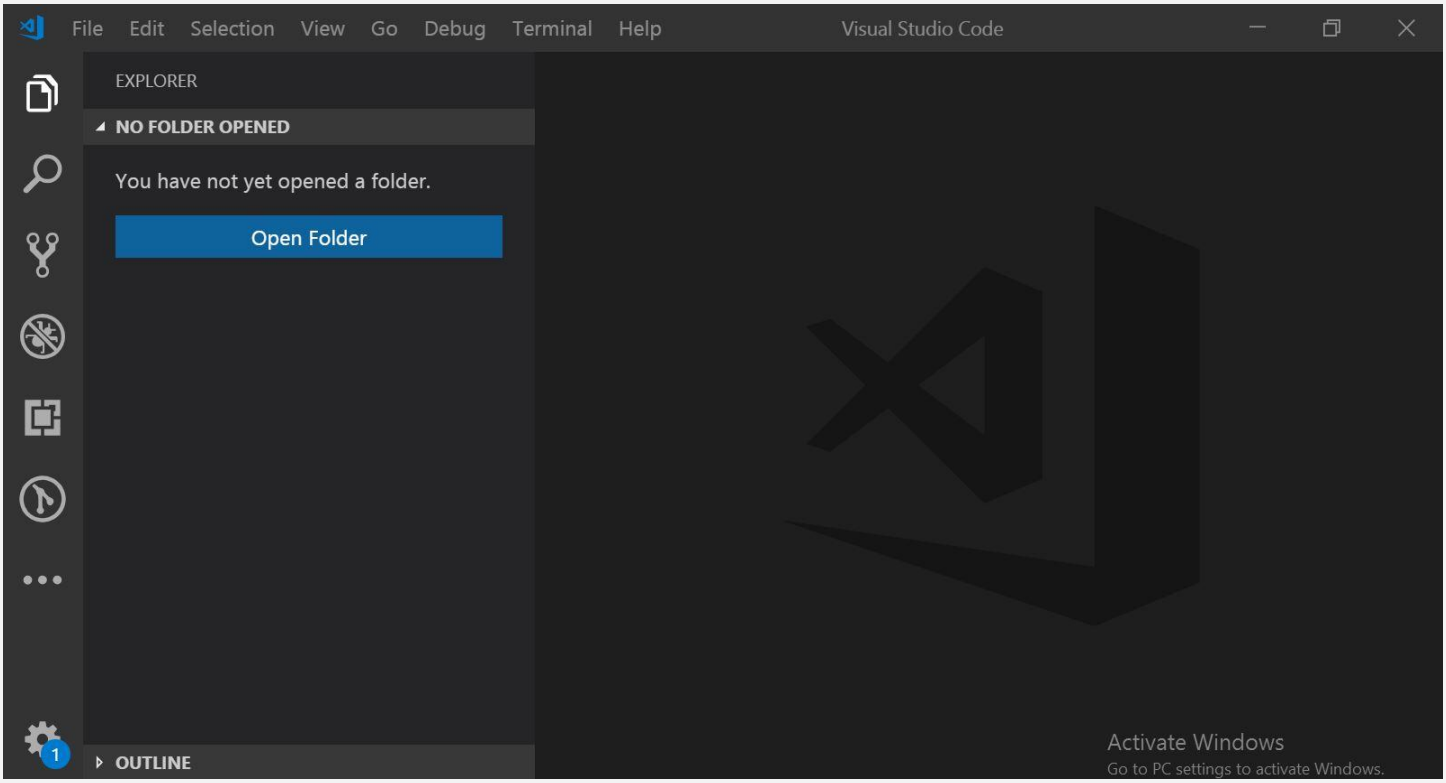
Once you start developing, you have to be able to follow your work progress and see the end product of your work. Chrome Dev Tool is a great place to start since it is a built-in feature of any chrome browser. To use the tool just hit “ctrl + shift+ i” or go to chrome option →> more tools →>developer tool.

0.8- Learning how to use a text editor to write your code

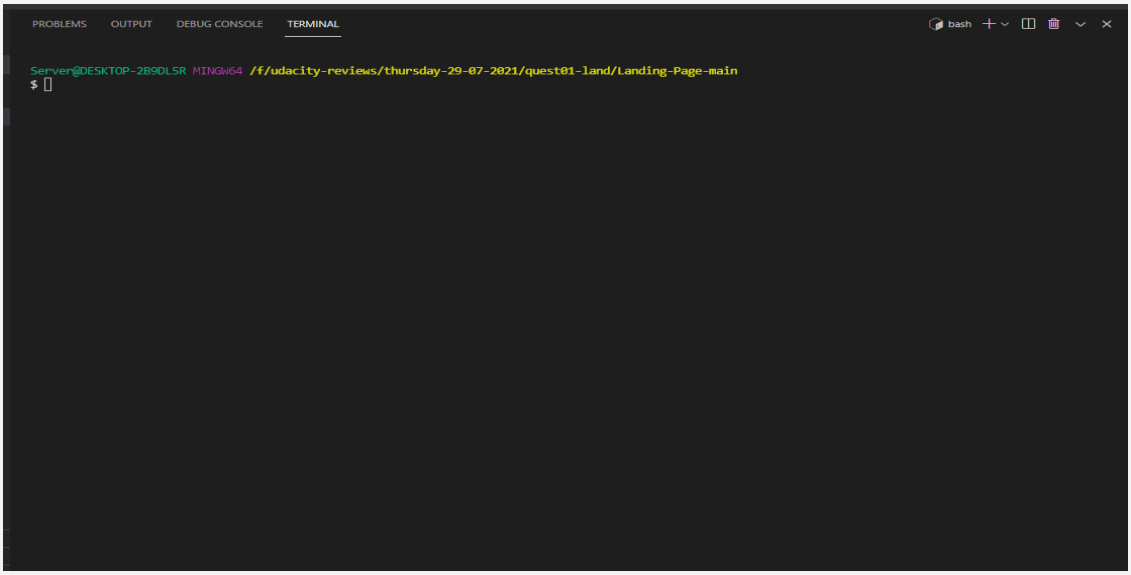
The text editor of choice is the Visual Studio Code

Follow the steps to download the program:

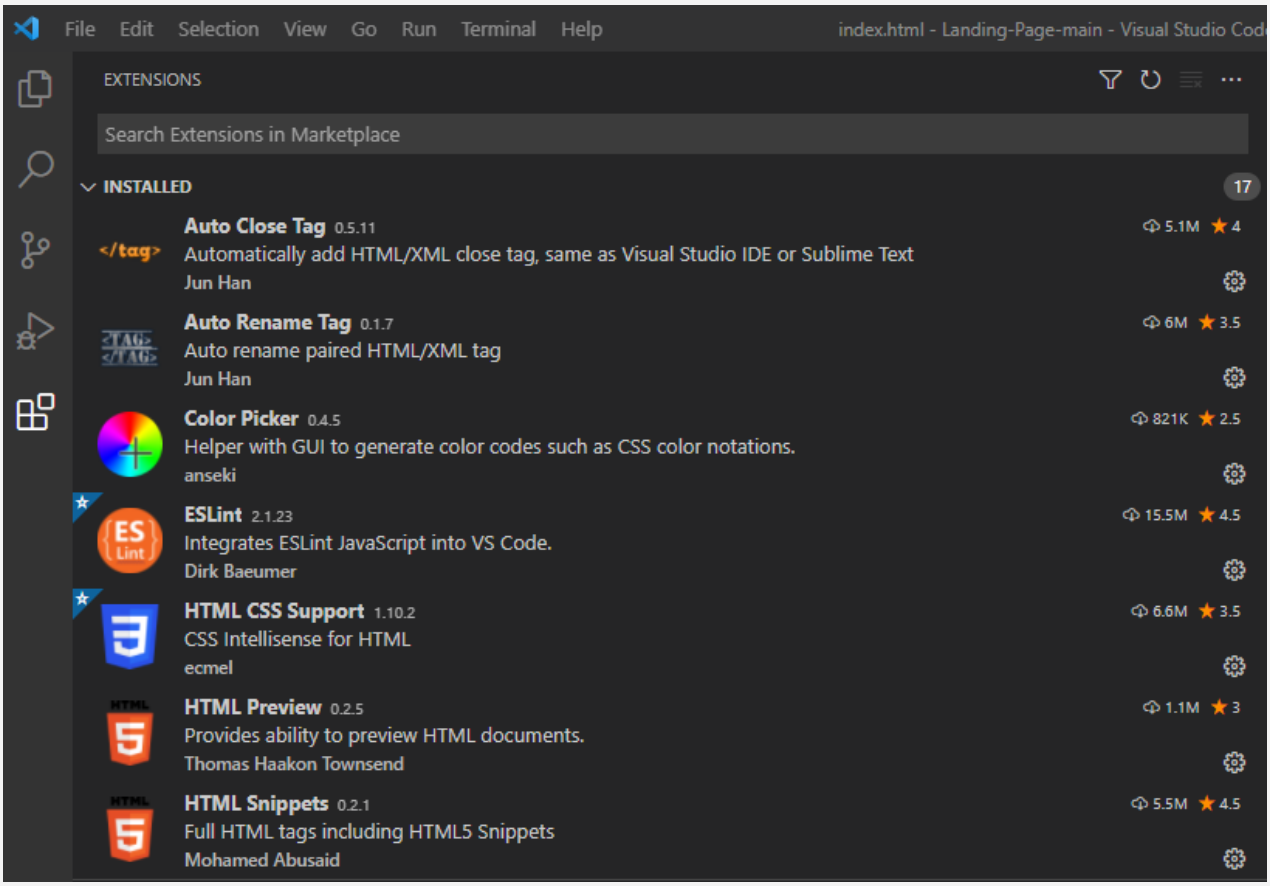
1. Go to https://code.visualstudio.com/download
2. Choose the version relevant to your operating system.
3. Visual Studio Code User Interface



4. Visual Studio Code Terminal



5. Visual Studio Code Extensions



0.9- Learning how to use Git the source code manager

Git is a content tracker or a source control manager which means it's responsible for tracking all changes in your work folders.

- To add git to any local folder you work with >>>

```
$ git init
```

This command will create a hidden git tracking folder and your project work folder is called a repository.

So, what is GitHub?

GitHub is a website that allows you the developer to upload your repositories for remote usage.

- Access your project remotely.
- Collaboration with other developers.

Can I use git without Github?

Yes, git is totally independent of GitHub and you may use it alone.

Set up git terminal inside VSC

To work with git you may use an external terminal window to type your commands using the terminal inside of Visual Studio Code is more convenient to your workflow. To use git inside VSC, go to git website to download the software

- **https://git-scm.com/downloads**
- Go to view menu and click terminal
- Git is a normal shell with basic commands you can use like

The most used shell commands:

>> go to the root folder

```
- $ cd ~
```

>> go to the parent folder

```
- $ cd ..
```

>> go to drive d

```
- $ cd d:/
```

>> list all files and folders inside a location

```
- $ ls
```

>>create this file inside your working folder

- `$ touch <file-name>.<file-extension>`
>> remove this file inside your working folder

- `$ rm <file-name>.<file-extension>`
>> create a folder inside your working folder

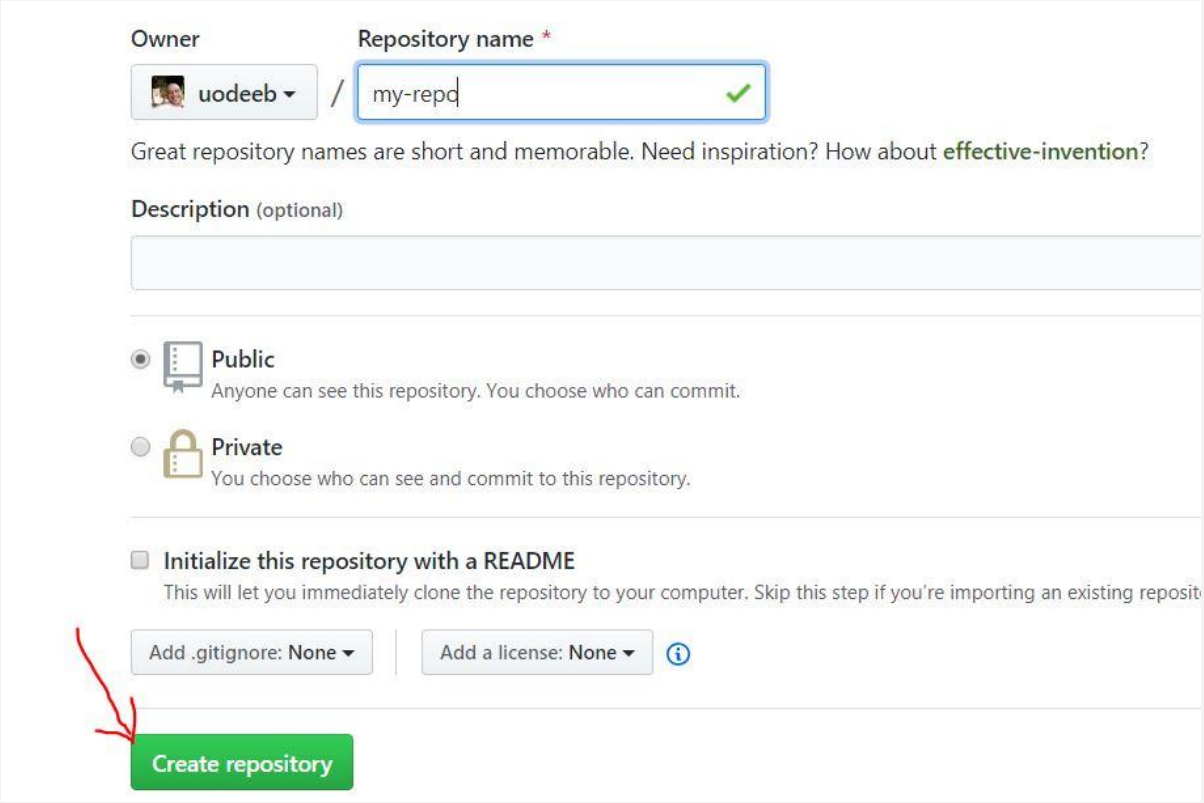
- `$ mkdir <directory-name>`

What is GitHub?

A step by step guide
ADD A PROJECT TO GITHUB

A] create your GitHub repo

1- GO TO >> www.github.com
2- open an account
3- At the top right >>> press new repository
4- Name your repository
5- create your repository



B] on your git bash command line

6- USING YOU GIT BASH >>>go to your project directory

USER MINGW64 /[YOUR-WORK-DIR]

```
$
```

7- to start tracking your folder with git>>>Type in the command line >>>>

```
$ git init
```

8- to add your folder to your GitHub repo >>>Type in the command line >>>>

```
$ git remote add origin <your-repo-url>
```

9- to check your remote repo >>>Type in the command line >>>>

```
$ git remote -v
```

You can see >>>

https://github.com/<user-name>/<repo-name>.git(fetch)

https://github.com/<user-name>/<repo-name>.git(push)

10-to add all your folder files to be tracked >>> Type in the command line >>>>

```
$ git add .
```

11- to commit all changes >>> Type in the command line >>>>

```
$ git commit -m "<name your commit to easily track>"
```

12- to push your files to the remote repo >>> Type in the command line >>>>

```
$ git push origin master
```

Check the video here >>>https://www.youtube.com/watch?v=pKQ8IlsGMsY&feature=youtu.be

1- Git Terminology

Version Control System (VCS) or Source Code Manager (SCM):

A VCS allows you to: revert files back to a previous state, revert the entire project back to a previous state, review changes made over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more.

Commit (snapshot):

Git thinks of its data like a set of snapshots of a mini file system. Every time you commit or save the state of your project in Git, it basically takes a picture of what all your files look like at that moment and stores a reference to that snapshot.

Repository (repo):

A directory that contains your project work, as well as a few files (hidden by default in Mac OS X) which are used to communicate with Git. Repositories can exist either locally on your computer or as a remote copy on another computer.

Working Directory:

The files that you see in your computer's file system. When you open your project files up on a code editor, you're working with files in the Working Directory. This is in contrast to the files that have been saved (in commits!) in the repository. When working with Git, the Working Directory is also different from the command line's concept of the current working directory which is the directory that your shell is "looking at" right now.

Checkout:

When content in the repository has been copied to the Working Directory. It is possible to checkout many things from a repository; a file, a commit, a branch, etc.

Staging Area or Staging Index or Index:

A file in the Git directory that stores information about what will go into your next commit. You can think of the staging area as a prep table where Git will take the next commit. Files on the Staging Index are poised to be added to the repository.

SHA:

A SHA is basically an ID number for each commit. It is a 40-character string composed of characters (0–9 and a–f) and calculated based on the contents of a file or directory structure in Git. "SHA" is shorthand for "SHA hash". A SHA might look like this:
e2adf8ae3e2e4ed40add75cc44cf9d0a869afeb6

Branch:

A branch is when a new line of development is created that diverges from the main line of development. This alternative line of development can continue without altering the main line. Going back to the example of save point in a game, you can think of a branch as where you make a save point in your game and then decide to try out a risky move in the game. If the risky move doesn't pan out, then you can just go back to the save point. The key thing that makes branches incredibly powerful is that you can make save points on one branch, and then switch to a different branch and make save points there, too.

Source >>> www.udacity.com

2- Configure git before start using it

As Git tracks any modifications in our working directory, it adds some vital information related to the author who made these modifications. So, you have to let Git knows your name and email before starting any track.

```
# sets up Git with your name
$ git config --global user.name "<Your-Full-Name>"
```

```
# sets up Git with your email
$ git config --global user.email "<your-email-address>"
```

3- Start tracking a folder

To start tracking any working folder, we have to use this git command to initialize a [.Git] hidden folder with all the needed files to track modifications automatically.

```
$ git init
```

4- Work with an existing repository

Sometimes we need to work with a project already hosted on GitHub. Use the clone(identical copy) command with the URL of your GitHub repo.

```
$ git clone <path-to-repository-to-clone>
```

5- Once you start your work with Git, you will definitely need to use a command to check what you have done. To check your work status use

```
$ git status
```

Note: It's a good practice to use this status command after any commit command you do to get a clear view of your work progress and the modification you have done.

6- after you do some modifications and commit them, you may need to check about these commits. Use the command git log to do so

```
$ git log
```

Note: By *default*, this command displays:

- the SHA[a unique id for each commit]
- the author[who did the commit]

- the date[when this commit took place]
- and the message[the description of your commit]
- You may use the[--oneline] flag to show the result in one line containing the first 7 numbers from the SHA and the message.

```
$ git log --oneline
```

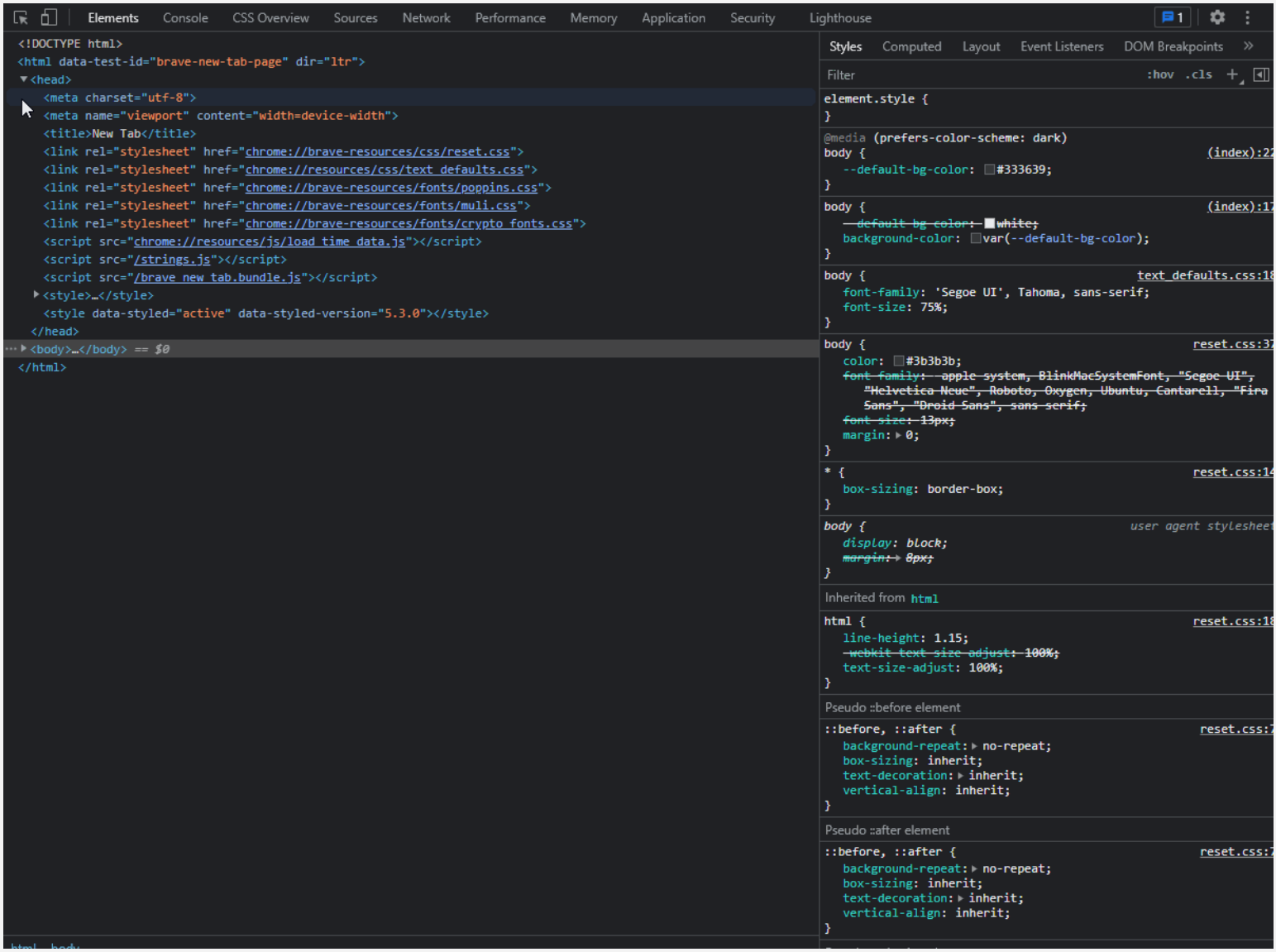
```
$ git log --oneline --decorate --graph --all
```

For all Git commands documentation try this link

1.0- Learning how to use chrome developer tools

To start working with your developer tool, go to your browser and click ctrl + shift + i or point at any place inside the browser and right click then choose inspect.

The developer tool will look something like this:



The most used taps inside the dev-tool are:

1. ### The Elements Tap

   Where you can find the HTML elements of your website. You can temporarily change the elements or their content but those changes will disappear if you refresh the page.
   Also, beside the elements you can find the styles applied to those elements.

2. ### The Console Tap

   Where we can check our JavaScript code for errors.

3. The Source Tap

   Where we can see all files of the website

4. The Network Tap

   Where we can check the requests and responses between the website and the server.

Chapter 02
HTML

1.1- What is HTML?

HTML(HyperText Markup Language) is the foundation of any modern website. It uses the markup technique to add a tag to the content to allow the browser to visually represent the content in a certain way.

H T M L

HyperText Markup Language

HyperText >>

Means linked text. Text is the most dominant part of any webpage and the HyperText is the core of any website. The HyperText is the clickable text where you can click to navigate to another webpage or another part of the same page.

Markup>>

Means to add a mark/tag to the web page content to help the browser to understand the nature of those content.

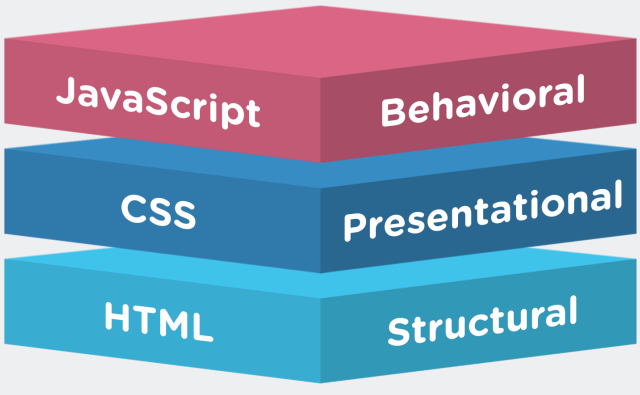
Language>>

A set of keywords with a special meaning in the language context.

So, is HTML a programming language?

No, HTML is not a programming language as it is used to structure(mark up) the data of a webpage but it's not responsible for doing any real functionality. It doesn't modify or manipulate data in any way. HTML can't take input and produce output. Think of it this way: you can't compute the sum of 2 + 2 in HTML; that's not what it's for. This is because HTML is not a programming language.

So, what is the difference between HTML, CSS, JavaScript?



If we think about a website as a human being, you can consider the HTML as the naked body, the CSS is the clothes he wears, and JavaScript is the things he can do like moving, speaking, ...etc.

1.2- The history of HTML

HTML is a very evolving markup language and has evolved with various versions updating. Long before its revised standards and specifications are carried in, each version has allowed its user to create web pages in a much easier and prettier way and make sites very efficient.

HTML 1.0 was released in 1993 with the intention of sharing information that can be readable and accessible via web browsers. But not many of the developers were involved in creating websites. So the language was also not growing.

Then comes the **HTML 2.0**, published in 1995, which contains all the features of HTML 1.0 along with that few additional features, which remained as the standard markup language for designing and creating websites until January 1997 and refined various core features of HTML.

Then **comes HTML 3.0**, where **Dave Raggett introduced** a fresh paper or draft on HTML. It included improved new features of HTML, giving more powerful characteristics for webmasters in designing web pages. But these powerful features of new HTML slowed down the browser in applying further improvements.

Then comes **HTML 4.01**, which is widely used and was a successful version of HTML before HTML 5.0, which is currently released and used worldwide.

HTML 5 can be said for an extended version of HTML 4.01, which was published in the year 2012.

source>>

1.3- The Anatomy of HTML Element

HTML Element anatomy

1- HTML Element is marked up in this way:

```
<opening tag> content </closing tag>
Ex: <p> Hello World!</p>
```

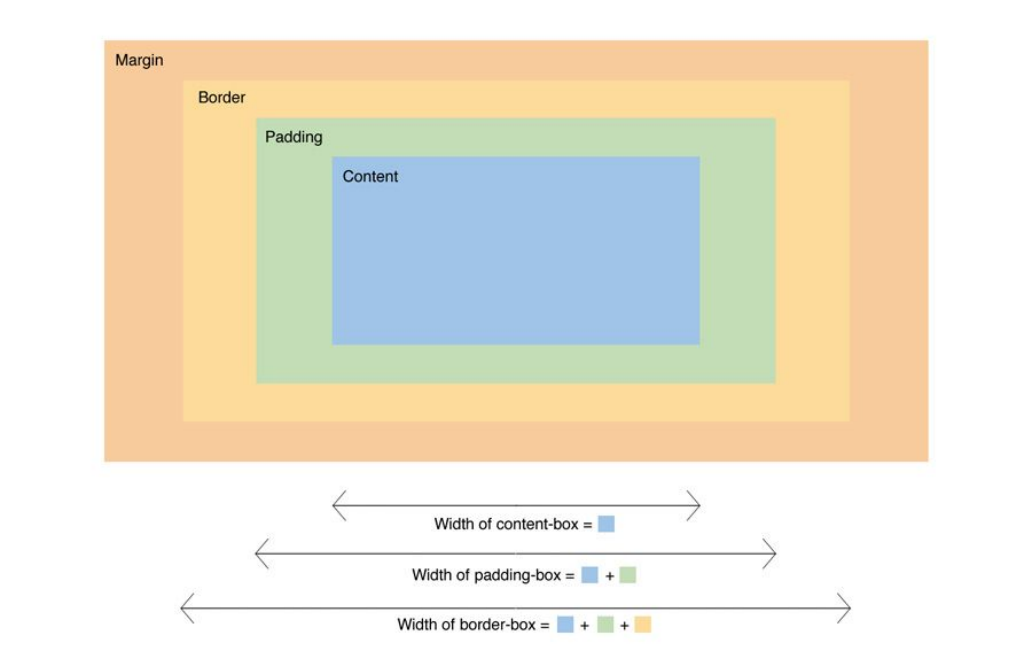
2- HTML Elements may have attributes

Attributes is a key-value pair used to add some additional information about a specific element.

To add an attribute you have to write it inside the opening tag like this

```
<opening tag key="value"> content </closing tag>
Ex: <p class="my-first-paragraph"> Hello World!</p>
```

3- HTML elements have what is called a MODEL BOX which is the geometrical boundaries of any element when it is parsed by the browser. The box model looks like this:



4- HTML element may have no closing tag if it has no content.

Ex: <img/> or <img> , <meta/> or <meta>

1.4- The structure of HTML Document

5- the main structure of any HTML document is

Source: www.w3schools.com

HTML elements reference

NOTE:

This is a reference to the most needed HTML elements Typed in the right syntax. You may have to try typing it yourself while developing to get familiar with it but when you are stuck and need to confirm the right syntax you may check this code samples.

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
<br>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<br>
<h2>HTML Links</h2>
<p>HTML links are defined with the a tag:</p>

<a href="https://www.w3schools.com">This is a link</a>
<br>
<h2>HTML Images</h2>
<p>HTML images are defined with the img tag:</p>

<img src="memix.png" alt="W3Schools.com" width="180" height="100">
<br>
<h2>HTML Buttons</h2>
<p>HTML buttons are defined with the button tag:</p>

<button>Click me</button>
<br>
<h2>An Unordered HTML List</h2>

<ul>
 <li>Coffee</li>
 <li>Tea</li>
 <li>Milk</li>
</ul>

<h2>An Ordered HTML List</h2>

<ol>
 <li>Coffee</li>
 <li>Tea</li>
 <li>Milk</li>
</ol>
<br>
<p>The pre tag preserves both spaces and line breaks:</p>

<pre>
  My Bonnie lies over the ocean.

  My Bonnie lies over the sea.

  My Bonnie lies over the ocean.

  Oh, bring back my Bonnie to me.
</pre>
```

```
<br>
<p>I am normal</p>
<p style="color:red;">I am red</p>
<p style="color:blue;">I am blue</p>
<p style="font-size:50px;">I am big</p>
<h1 id="center-head" style="text-align:center;">Centered Heading</h1>
<p style="text-align:center;">Centered paragraph.</p>
<br>

<p>This text is normal.</p>
<p><strong>This text is bold.</strong></p>
<p><em>This text is emphasized.</em></p>
<p><i>This text is italic.</i></p>
<p><mark>This text is highlighted.</mark></p>
<br>
<p>Browsers usually insert quotation marks around the q element.</p>

<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>
<br>
<!-- This is a comment -->
<p>This is a paragraph inside a hidden comment.</p>
<!-- Comments are not displayed in the browser -->
<br>
<h2>Local Links</h2>

<p><a href="page.html">HTML Images</a> is a link to a page on this website.</p>
<br>
<h2>Link Colors</h2>

<p>You can change the default colors of links</p>

<a class="colored-link" href="page.html" target="_blank">HTML Images</a>
<br>
<h2>Image Links</h2>
<p>The image is a link.<strong> click to go to layout using float page</strong></p>


<a href="layout-float.html">
 <img src="memix.png" alt="HTML tutorial" style="width:42px;height:42px;border:0">
</a>
<p>We have added "border:0" to prevent IE9 (and earlier) from displaying a border around the
image.</p>
<br>
<h2>Image Size</h2>

<p>In this example, we specify the width and height of an image with the width and height
attributes:</p>

<img src="girl.jpg" alt="Girl in a jacket" width="500" height="600">
<p>Use the style attribute to specify the width and height of an image:</p>
<img src="girl.jpg" alt="Girl in a jacket" style="width:500px;height:600px;">
<h2>Images in Another Folder</h2>
```

```
<p>It is common to store images in a sub-folder. You must then include the folder name in the src attribute:</p>

<img src="/img/girl.jpg" alt="girl" style="width:128px;height:128px;">
<h2>Animated Images</h2>
<p>The GIF standard allows moving images.</p>

<img src="EGYPHS-H-N-Y.gif" alt="egyphs" style="width:300px;height:300px;">
<br>
<h2>Floating Images</h2>
<p><strong>Float the image to the right:</strong></p>

<p>
<img src="EGYPHS-H-N-Y.gif" alt="Smiley face" style="float:right;width:200px;height:200px;">
A paragraph with a floating image. A paragraph with a floating image. A paragraph with a floating image.
</p>

<p><strong>Float the image to the left:</strong></p>
<p>
<img src="EGYPHS-H-N-Y.gif" alt="Smiley face" style="float:left;width:200px;height:200px;">
A paragraph with a floating image. A paragraph with a floating image. A paragraph with a floating image.
</p>
<br>
<br>
<br>
<br>
<br>
<br>
<br>
<br>
<h2>HTML Tables</h2>

<p>HTML tables start with a table tag.</p>
<p>Table rows start with a tr tag.</p>
<p>Table data start with a td tag.</p>

<hr>
<h2>1 Column:</h2>

<table>
 <tr>
   <td>100</td>
 </tr>
</table>

<hr>
<h2>1 Row and 3 Columns:</h2>
<table>
 <tr>
   <td>100</td>
```

```
   <td>200</td>
   <td>300</td>
 </tr>
</table>

<hr>
<h2>3 Rows and 3 Columns:</h2>
<table>
 <tr>
   <td>100</td>
   <td>200</td>
   <td>300</td>
 </tr>
 <tr>
   <td>400</td>
   <td>500</td>
   <td>600</td>
 </tr>
 <tr>
   <td>700</td>
   <td>800</td>
   <td>900</td>
 </tr>
</table>

<hr>
<br>
<h1>My <span style="color:red">Important</span> Heading</h1>
<br>
<div class="cities">
   <h2>London</h2>
   <p>London is the capital of England.</p>
   </div>

   <div class="cities">
   <h2>Paris</h2>
   <p>Paris is the capital of France.</p>
   </div>

   <div class="cities">
   <h2>Tokyo</h2>
   <p>Tokyo is the capital of Japan.</p>
   </div>
   <br>
   <h2>Same Class, Different Tag</h2>
<p>Even if the two elements do not have the same tag name, they can have the same class name, and
get the same styling:</p>

<h2 class="city">Paris</h2>
<p class="city">Paris is the capital of France.</p>
<br>
<h2>Difference Between Class and ID</h2>
```

```
<p>An HTML page can only have one unique id applied to one specific element, while a class name can be applied to multiple elements.</p>

<!-- A unique element -->
<h1 id="myHeader">My Cities</h1>

<!-- Multiple similar elements -->
<h2 class="city">London</h2>
<p>London is the capital of England.</p>

<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>

<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>
<br>
<h2>HTML Iframes</h2>
<p>An iframe is used to display a web page within a web page:</p>

<iframe src="layout-float.html"></iframe>
<br>
<h2>Use JavaScript to Change Text</h2>
<p>This example writes "Hello JavaScript!" into an HTML element with id="demo":</p>

<p><strong id="demo"></strong></p>

<script>
document.getElementById("demo").innerHTML = "Hello JavaScript!";
</script>
<br>
<h2>Text Input</h2>

<form action="/action_page.php">
 First name:<br>
 <input type="text" name="firstname">
 <br>
 Last name:<br>
 <input type="text" name="lastname">


<p>Note that the form itself is not visible.</p>

<p>Also note that the default width of a text input field is 20 characters.</p>
<h2>Radio Buttons</h2>


 <input type="radio" name="gender" value="male" checked> Male<br>
 <input type="radio" name="gender" value="female"> Female<br>
 <input type="radio" name="gender" value="other"> Other


   First name:<br>
```

```
   <input type="text" name="firstname" value="Mickey">
   <br>
   Last name:<br>
   <input type="text" name="lastname" value="Mouse">
   <br><br>
   <input type="submit" value="Submit">
   <br>
   <select name="cars">
       <option value="volvo">Volvo</option>
       <option value="saab">Saab</option>
       <option value="fiat">Fiat</option>
       <option value="audi">Audi</option>
     </select>
     <br><br>
     <input type="submit">
     <br>
     <h2>Textarea</h2>
<p>The textarea element defines a multi-line input field.</p>

<form action="/action_page.php">
 <textarea name="message" rows="10" cols="30">The cat was playing in the garden.</textarea>
 <br>
 <input type="submit">
 </form>
  <p>If you click the "Submit" button, the form-data will be sent to a page called
"/action_page.php".</p>
 <h2>The autocomplete Attribute</h2>

 <form action="/action_page.php" autocomplete="on">
   First name:<input type="text" name="fname"><br>
   Last name: <input type="text" name="lname"><br>
   E-mail: <input type="email" name="email" autocomplete="off"><br>
   <input type="submit">
 </form>
  <p>Fill in and submit the form, then reload the page to see how autocomplete works.</p>
 <p>Notice that autocomplete is "on" for the form, but "off" for the e-mail field.</p>
<br>
<h2>The button Element</h2>

<button type="button" onclick="alert('Hello World!')">Click Me!</button>
```

Please check this GitHub repo for the full reference project

https://github.com/uodeeb/my-repo