Django Full stack Web Development

-Jose Portilla-

Part 1: Front-End

Here is some useful info for you to get started:

1. Please watch the course overview lecture, lots of useful info there!

2. Our QA forums can be found by going to your course dashboard (top right button when viewing a lecture)

3. Our student chat channel is here: https://discord.gg/TztE6B8

4. Video guide to the chat room: https://www.youtube.com/watch?v=bkH89OJ001M

4. Our notebooks can be found as resources in the Course Overview lecture.

5. Info on certification: https://support.udemy.com/hc/en-us/articles/229603868-Certificate-of-Completion

6. The best place to ask questions is in the QA forums, not the messaging system, please keep in mind I may just ask you to repost your question in the QA forums if you message me here. (It’s just easier for me to manage that way.)

Thank you so much,

Jose

(This is an automated message, but I do see replies).

Django Full-stack bootcamp. Part A- Front End

[Course Introduction 4](#_Toc11269024)

[**A/ FONT-END** 5](#_Toc11269025)

[I/ Introduction: 5](#_Toc11269026)

[1/ Front-end 6](#_Toc11269027)

[2/Back-end: 6](#_Toc11269028)

[II/ HTML: 7](#_Toc11269029)

[1/ Instroduction: 7](#_Toc11269030)

[2/ HTML – Level 1 7](#_Toc11269031)

[a/ Basic: 7](#_Toc11269032)

[*W3school.com* 7](#_Toc11269033)

[*Developer.mozilla.org* 7](#_Toc11269034)

[b/ Basic Taging: 7](#_Toc11269035)

[c/ List: 8](#_Toc11269036)

[d/ Divs an Spans 8](#_Toc11269037)

[e/ Attribute: 8](#_Toc11269038)

[2/ HTML level 2: 8](#_Toc11269039)

[a/ Table: 8](#_Toc11269040)

[b/ Form and Labels: 8](#_Toc11269041)

[c/ Radio button, dropdown list and text area: 9](#_Toc11269042)

[III/CSS: 10](#_Toc11269043)

[1/Introduction : 10](#_Toc11269044)

[2/ CSS\_lv\_one: 10](#_Toc11269045)

[3/ CSS Level two: 11](#_Toc11269046)

[a/ Font: 11](#_Toc11269047)

[b/ Box model 12](#_Toc11269048)

[IV/ Bootstrap: 13](#_Toc11269049)

[1/ Overview: 13](#_Toc11269050)

[2/ Button 14](#_Toc11269051)

[3/ Forms 14](#_Toc11269052)

[4/Navbar 14](#_Toc11269053)

[5/Grids: 15](#_Toc11269054)

[V/ JavaScript 17](#_Toc11269055)

[1/ Introduction: 17](#_Toc11269056)

[2/ Basic data type and funtion(), method(). 17](#_Toc11269057)

[3/ Comparison and Operation in JavaScript: 18](#_Toc11269058)

[4/ Control flow 18](#_Toc11269059)

[5/ Loops: 19](#_Toc11269060)

[6/ Functions: 19](#_Toc11269061)

[7/Arrays 20](#_Toc11269062)

[8/ Object 20](#_Toc11269063)

[VI/ Document Object Model: 21](#_Toc11269064)

[1/ Introduction: 21](#_Toc11269065)

[2/ DOM interaction: 21](#_Toc11269066)

[2/ Content Interaction: 21](#_Toc11269067)

[3/ Event 22](#_Toc11269068)

[VII/ jQuery 3.x 22](#_Toc11269069)

[1/ Introduction: 22](#_Toc11269070)

[2/ Basic 22](#_Toc11269071)

[3/ jQuery Events: 23](#_Toc11269072)

[4/ Front-End Project 25](#_Toc11269073)

# Course Introduction

* The course is designed for all level, including beginner.
* IN **front-end**, the course includes:
  + HTML
  + CSS
  + Bootstrap
  + JS (JavaScript)
  + jQuery
* **Back-end**:
  + Python
  + Django
  + SQLite
* General curiculun:
  + HTML, CSS, JS
  + DOM and Jquery
  + Python
  + Django and SQLite
  + Website clone
* Usefull ways to share my code:
  + Codepen
  + Codepad
  + A gist on Github
  + A screenshot
* Atom development environment: Jose highly recommend this IDE for this course.
  + Atom was built with a lot of front-end technology: HTML, FS, CSS and Node.JS <<< Electron (framework) for building flatform apps using web technologis.

# **A/ FONT-END**

## I/ Introduction:

* What is web:
  + URL: When we type in an URL address, browser send a packets (?) including ­IP address to let te sever know which website are we want to connect
  + ISP: Internet Service Provider.
  + The server send back website splited into sevenral packets. Packets don’t care how they get to the browser. The only thing matter is final destination. Mean all packets reach to browser the fastest way it can, each packet may go in a different way.
  + When all packets reaassemble >> the website is showed.
* Full-stack:
  + Front-End: Is what user see on the website.
    - HTML
    - CSS
    - JavaScript
    - Jquery and Bootstrap are built using the previous three.
  + Back-end: Technologies used to decide what to show in Front-End
    - The Language
    - The Framework
    - The Database
    - Php, Node.js, Ruby/rails, Java, Python, etc.

### 1/ Front-end

* HTMP: HyperText Markup language
  + Every website have HTML, it is the structure of a page.
  + View by click view Page source.
* CSS: Cascarding Style Sheets
  + IS the styling of the website
  + Color, fonts, borders
  + Not madatory but most website have it.
* JavaScript:
  + Allow us to add interactivities to the website, including program logic.
  + Any site with interactivity uses JavaScript in some way. **Otherwiese**, the site is “static” (Tĩnh, không tác động thay đổi được từ phía người dùng, kiểu các trang quảng cáo, banner PR,...) .

### 2/Back-end:

* Back-end in cludes the Language, the Framework, the Database
* In the course, we use:
  + **Python** as the language.
    - This language is simple, poweful, scalable, have many libraries. (Can have lot of extension on AI, Big Data, machine learning,.. )
  + **Django** as the Framework.
    - Have a lot of website run on: Printerest, Instagram,...)
  + **SQLite** as the Database.
    - Come up with Django and Python. Most suitable.

## II/ HTML:

### 1/ Instroduction:

*#Never copy code from word file (.doc or .docx). The annoying*

*double quotation in word is different from in coding. Only notepad maybe accepted. This error did drive me crazy for a few seconds. LOL.*

### 2/ HTML – Level 1

#### a/ Basic:

* First, we can create project folder by click add folder. IN each project folder can have sub file for project file and sub project folder.
* Use get full path to get link to the HTML file.
* An HTML file have a head and body:
  + Head is first section: Meta data. In the level one we will use a lot of CSS and JaveScript.
    - <title> Tag.
    - In <meta> tag: What character set we are using.
  + Next in <body>:
    - First we just know about comment in HTML: <!—*Comment content t-->. (#Shortkey: ctrl + forward slash*)

### *W3school.com*

### *Developer.mozilla.org*

#### b/ Basic Taging:

*#Shortkey: ctrl + back slash << open/close side tab (project tab)*

* We have heading 1, heading 2, etc by <h1> This is heading 1 </h1>. Heading 6 is the smallest level.
* We have paragraph for <p> this is the paragram </p>.:
  + Each blog go with a tag is a separate line. Without separate tag, they are in the same line.
* There are 2 ways to get the text bold/italic:
  + First: we put them btw tag <b></b> or <i></i>. It’s easy to remember but Atom do not have auto type for this. So there is a second way.
  + Second: Instead of i and b, we use <strong> and <em> (Stand for *strong* and *emphasys)*

#### c/ List:

* There are 2 type of list in HTML :
  + Ordered list:
    - Put in btw <ol> tag
    - Each item go with a number.
  + Unordered list:
    - Put in <ul> tag
    - Each item go with a bullet, no number
  + List item: <li>

*#Note that to add an item, the item have to go btw <li> tag.*

* + Nested list: We can nest in a list another list.

#### d/ Divs an Spans

* <div> </div> and <span> are use to add effect to particular contents.
  + Divs are to group of contents. And link them to something called **class.** Spans are to create a separated text inside thosedivs

#### e/ Attribute:

* First, let add some image in to the website by the <a> tag.
  + This <a> tag have two attribute:
    - src: stand for source
    - alt: stand for alternative
    - We can actually show the image by a link from internet. An online link from google to save memory from sever(?). And put the link of local file in the alternative (which can be a text “the\_object not found”).

### 2/ HTML level 2:

#### a/ Table:

* Tag <table>, <thead>, <th>, <tr>- Table row, <td> - table data. Attribute boder= “Number”.

#### b/ Form and Labels:

<form class =” ” action =” Link\_to\_some\_webpage ” method= “get” >

<input id=”” type =”” name =”” value = “”>

</form>

(?) why does action take us to the homepage instead of the specified link ?

There are various type of “type” in input. See full detail in w3school or developer.mozilla.com.

<label for=”a\_name:> The\_Label </label>

<input id=”a\_name” type=”” name=”” value=”” placeholder=”” *required*>

*#The id\_name should be unique across all over the file.*

*# the “name” will be assign the value of value/input value.*

#### c/ Radio button, dropdown list and text area:

* Radio button*:*

*<label name = “radio\_button”>Yes</label>*

*<Input id=”radio\_button” type=”radio” name=”yes\_or\_no” value=”Yes”>*

*<label for =”radio\_button\_2”>NO</label>*

*<input id=”radio\_button\_2” type=”radio” name=”yes\_or\_no” value=”No”>*

*#If we want only one radio button to be chosen. Set the same value to the attribute name .*

* Drop down button:

*<select name=”Address”>*

*<option value=”Binh\_Thanh\_dist”> Binh\_Thanh\_dist >*

*<option value=”Dist\_1”> Dist\_1>*

*<option value=” Dist\_2”> Dist\_2>*

*</select>*

* Textarea:

*<textarea name=”” rows=”a\_number” cols=” a\_number”></textarea>*

## III/CSS:

### 1/Introduction :

* CSS - Cascading Style Sheets is used to set how content in HTML is displayed.
* HTML file is the file contain content of the web. And the CSS code is stored is a separate css file which is link to HTML.
* A HTML file can be link to different CSSs file. Each combination can create a totally different webpage.
* In this section Jose also introduce Boostrap.
* Syntax:

\*html:

<link rel="" href="#">

- rel= relationship href : link

\*CSS:

- Comment by /\* Comment \*/ or shortkey ctrl + /

/\* selected tag {

prperty: Value;}

### 2/ CSS\_lv\_one:

* Link a css file to html file by put tag link in to head of html file. Attribute rel: relationship, href: link to
* Comment in CSS use: /\* *comment* \*/
* General format in CSS:

Tag\_name{

Property\_value: how you want it to be displayed

}

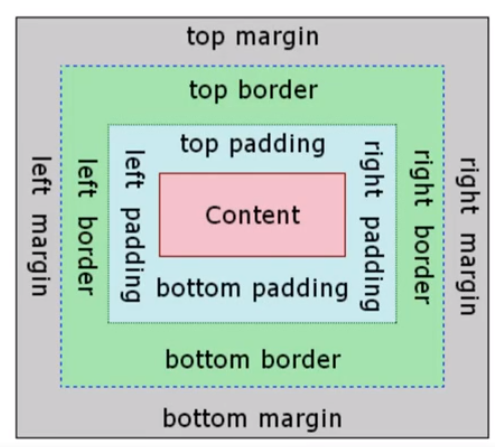
* + A simple property name: color. There’re 4 way to use property color:
    - 1/ Use literal text of color (Limmit amount of built-in color)
    - 2/ Use RGB code by color picker .
    - 3/ Can use rgba, a stand for alpha. Mean transparency << in this case it’s similar to blur
    - 4/ Use Hex code: #AB31D3
* To apply style:
  + to specific section: use <div> <span> (<span> is inside <dib>)
  + To a group of element: .class\_name - apply to all element with class of class\_name.
  + To a single element: #id\_name- apply to element with id of id\_name
  + 3 usefull way:
    - h3 + ul {}: all <ul> which is after <h3>
    - ul a{}: all <a> which is inside <ul>
    - ul a[href="#"] {}Hierarchy in CSS: all <a href="#"] > which is inside <ul>
    - tag.class\_name
    - tag.id\_name
  + Normal call.
  + Class
  + Id
  + Name of id and class must start by a letter.
* Some useful properties in this lecture:
  + Color
  + Border
  + Text
  + Background
* background: (Can use color or picture or even url of picture with "url(link). This link can be an image in the same directory or an image with online url. Differ from HTML, file path to image in PC cannot be use here .

### 3/ CSS Level two:

#### a/ Font:

* To chose font, there’re 2 ways:
  + Chose by property *font-family: “Font\_name”*.
  + Chose by specify character: *font-family: character(tính chất)*.
* To adjust size, style, weight, alignment.
* Not every font have all styles available, and either available on both OS, Windown. We have to check on cssfontstack.com.
* We can check either a font/ which font is available on a specific OS. Check by Google “list of font available on windows/Mac OS”.
* Font can be add online to html. Syntax are on *fonts.google.com* and *fontlibrary.org*. Just copy as instruction.

#### b/ Box model

* Top >> right >> bottom >> left (1).
* Content >> padding >> border >> margin (2)
* Syntax to call:
  + (*1) + (2): measurement*
  + *(2): Top size >> right size >> bottom size >> left size*
  + *(3): auto #auto center*
* 
* There’re 2 ways to adjust the element in box model:  
  + 1/ Directly type in number. For instance: 1px/ 10px/ 100px.
  + 2/ Use percentage “%”. For instance. Left-margin:

## IV/ Bootstrap:

### 1/ Overview:

*#NOTICE: The Version Jose uses in this section is v3.3.7. Keep that in mind because later versions use really difference class names.*

*Just focus on the version of this bootstrap. Focus on understanding how bootstrap works, how to reference documentation.*

*We will have to update the knowledge because content in this section is really limited as well. Then just Don’t be afraid of time consuming for learning new version, we’ll have to spend much more time than just 1,5 hour of the section anyway.*

* Bootstrap is a common framework for Front-end Developer:
  + Inversion of Contrl
  + Default Behaviour
  + Extensibility
  + Non-modifiable Framework Code
* A large part of bootstrap is not memorization, so this section will focus on common use of bootstrap and how to reference documentation in our owe use.
* CDN- Content Deliver Network, it’s a way to use boostrap which is similar to using font from fonts.google.com
* We can either download Bootstrap to our local storage for some purpose. But generally, just use the CDN
* IN Bootstrap, the most important thing is that we need to understand how to reference needed document, not to remember all the classes (because We cannot). And it’s much more effective to understand how references works.

*#Below are some Idea when I come over boostrap website*

* In bootstrap, there’re ton of written code for elements. Just check them out in element from bootstrap menu.
* Bootstrap mostly use *class* to call an existed element. Mostly change needed will be apply by changing name of the class. *(#Not sure yet)*
* We can overwrite written property by link to our own css file. The new condition will be apply in higher priority.

### 2/ Button

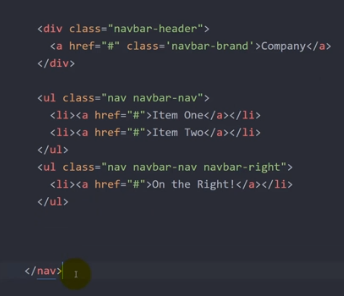
* This is so ez so I don’t talk much.
* Just go to bootstrap website and grab the code to html file. Remember that there are many existed button there.
* No matter how many button created, user only can chose 1 button over a website. Notice this!

### 3/ Forms

* To make the input box look nicer, use class “*form-group*”.
* To add some little notice, use *<small>* tag.
* In dropdown selection ( *<select>* tag), use option *multiple.*
* To disable option, use option *disable.*
* We can put some element together into a *<fieldset>, <legend>* tag*.* The *<fieldset* tag not only group the elements together but also create a boder around those.

### 4/Navbar

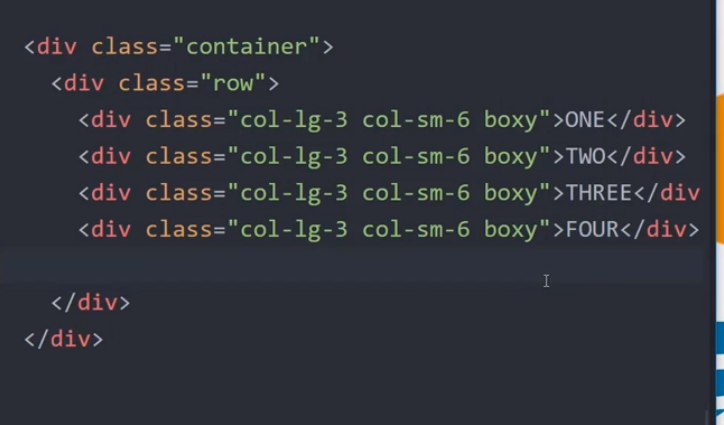
* <nav> tag is to control navigation bar (<nar class= “navbar navbar-default> in bootstrap) of the website.
* Navigation bar usually have:
  + 1/ a header which mostly is a brand and the navigation components. <div class = .”navbar-header”>
  + 2/ The component of navigationbar. Class:<div> <ul class =”nav navbar-nav”>



*#Question: I am having trouble with create spacing between border of jumbottron and navigation bar. Is it a border or what? I tried margin-top and padding-top but weird when changing size of page. See my stuff in navbar.html and navbar.css,*

### 5/Grids:

* Grids system is one of the most fundamental features of Bootstrap
* Bootstrap tools through classes is convinient.
* Grids system is much more than that.
* The system provides the core mechanism by which using bootstrap allows websites to look good through multiple device of multiple screen sizes.



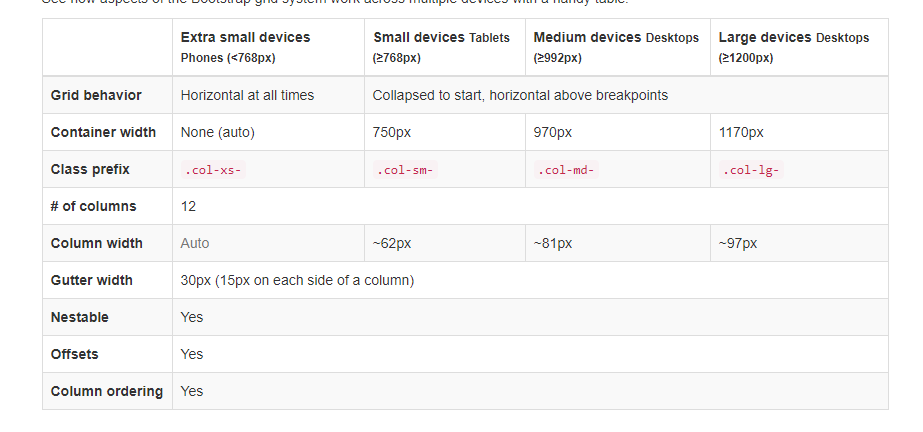


* To summon grids, we use class *row.* Each time the class is called, the selected tag’s area width is devided into 12 parts:
  + The original width is divided into 12 parts.
  + A grid can be nested into an other grid
* The overall syntax of class\_name is:

*col-Screen\_size-Number\_of\_columns*

*Eg: col-md-6, col-lg-12*

*# Bootstrap 4 have the extra large screen*

* The default size in bootstrap:
  + Extra small: <768px
  + Smal: >= 768px
  + Medium: >=992px
  + Large: >=1200px
  + 

## V/ JavaScript

### 1/ Introduction:

* First of all, JavaScript is a full program language, not like HTML (*Markup language*) and CSS (*Style sheet language*).
* Almost all browser have a built in console of JavaScript so we don’t need install anything.
* Some first impression:
  + Comment: //
  + Number includes number either with or without decimal point
  + “10” can be compare to number, which is showed later
  + Booleans are lower case: *true* and *false*
  + undefined and null
  + clear the console: clear()
* Pay attention the use of semicolon in JavaScript
* Error with syntax ( *invalid or unexpected token*) will stop the whole js file from executing. *Undefined* just stop blocks of code go after itself

### 2/ Basic data type and funtion(), method().

* Number operation:
  + Addition
  + Subtraction
  + Multiplication
  + Division
  + Power of
  + Modulo: %
* String:
  + Cancatenate: Same as Python, using “*+*” sign
  + Check length: *string.length*
  + Begin new line: *\n*
  + Tab: *\t*
  + Escape quote by back slash \
  + Indexing: Count from 0, as Python. *But can we reverse ? <<can’t*
  + String in JS cannot be directly multipy by an asterisk like in Python
* Variables:
  + We use syntax: var var\_name = value
  + *undefined* and *null*
    - undefined mean something is created and no previous value was found.
    - null mean none value was assignd to a variable
* *console.log()* is to output something to the console. Quite similar to *print()* in Python
* *prompt()* is similar to *input()* in Python. A string is return. But in JS, if input is a number, the return string value of this can be compare to a number bet nomal comparision

*#JS compare them by auto converse 2 value to the same type*

### 3/ Comparison and Operation in JavaScript:

* Comparison:
  + Comparison is almost the same as Python. Except compare number and string-ed number. To fix that, add an additional equal sign: "===" ,"!=="
  + So in JS, *1 == true* and *0 == false* >>> true. The value when in the same data type is similar. But *1===true* and *0 === false* return false. Same to *undefined == null.*
  + We have another special case: *NaN==Nan* >>> false. This’ll be discuss later
* Operators:
  + And: &&
  + Or: ||
  + Not: !

***#Not like in Python***: multiple comparison without and/or/not is operated from left to right

* + *In JS: 13==13==1 mean (13 == 13) ==1 which is true == 1 >>> final return value :* ***true***
  + *In Python: 13 ==13 ==1 mean 13== 13 and 13 ==1 >>>final return value:* ***false***

### 4/ Control flow

* *If (condition){*

*//execute some code*

*}else if(condition two){*

*//execute some other code*

*}else{*

*//execute some backup code*

*}*

### 5/ Loops:

* While loops suntax:

*While (condition){*

*//Execute some code while the condition is true*

*}*

* The ***break*** statement have to same use in Java as in Python
* For loops in JS includes 3 types:
  + For
  + For/in
  + For/of
* For loop syntax (Not yet learn about for/in and for/of)

*for (statement ;statement2; statement3){*

*//Execute some code while the condition is true`*

*}*

*#with:*

*Statement 1 is what happen before the loop*

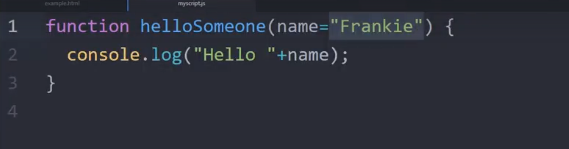
*Statement 2 is the condition to be checked*

*Statement 3 is the code that’d be executed after each loop*

### 6/ Functions:

*#Notice: IN some case, JS accept 2 type of date. For eg: “2” + 2 = 22. But this is not what we should expect for cause it would lead to later error ?*

* Everything here is exactly the same as in Python. Except there isn’t any **global** statement.
* Semicolon is between statement



### 7/Arrays

* Syntax: *var variable = [value 1, value 2, value 3, value 4,…]* (max length-4,294,967,295).
* A version of Python list in JavaScript. Are mutable, sequences, can be nested, indexing.
* The data can be put in the same line or in multiple lines, this is not limited like in Python
* Iteration in JavaScript: *for (x in interatable\_object){ }*
* Replacement of pop and append: *pop( )* and *push( ).*

### 8/ Object

* Object is exactly like in Python but when call a value, the key must be put in a double quotes.
* Objects have no order inside
* Use *console.dir(object\_name)* to print out everything in the object to the console.
* In Object, we can build method with syntax *method\_name : function() {}*
* **This** keyword is somehow similar to *self* in class of Python.
  + ***Self*** keywork in Python use in classes, refer to the class
  + ***This*** keywordin JavaScript is used in object of JavaScript, refer to the Object
* This have to be run all in Text editor. When come out to console, only Objects with key and value can be call, methods are not. We have to re execute them.

## VI/ Document Object Model:

### 1/ Introduction:

* **Document Object Model (DOM)** is a programming interface for HTML, XML and SVG documents. It provides a structured representation of the document as a tree
* The DOM defines methods that allow access to the tree, so that they can change the document structure, style and content.

### 2/ DOM interaction:

* First important document attributes:
  + *document.UR* : Grab the websit3 address
  + *document.body:* grab everything in the body of HTML file
  + *document.head:* Grab everything in the head of HTML file
  + *document.links*: Grab all the links that in the page
* Method for grab specific elements:
  + *document.getElementById(“id\_name”)*
  + *document.getElementByClassName(“class\_name”)*
  + *document.getElementByTagName(“tag\_name”)*
* The below method work the same way as the above but accept any CSS style selector (id with the *#hashtag*, class *.dot* with the and tag)
  + *document.querySelector(“selector\_name“)* – return the first element
  + *document.querySelectiorAll(“selector\_name“ )* – return all elements

*#I have a misunderstanding of calling element ( . and # for class and id). This shows the dangerous of recently non-sense, non-discipline and ridiculous attitude. Shame on you Hoang*

### 2/ Content Interaction:

* Includes texts, HTML codes and attributes
* *variable\_name.textContent* - This returns just the text. Anychange here will return **a text**.
* *variable\_name.innerHTML* – This is quite similar to the above and easy to make coder confused. This returns the actual html inside the tage *<>This\_HTML\_is\_reutrned</>* . Changes here return texts and html action if exist.
* *variable\_name.getAttribute(“attribute\_name”)* - This returns the original attribute. The attribute cannot be directly change like textContent, innerHTML. The ***get*** and ***set*** method are in 2 separated method.

#*Is this mean We cannot directly change the value of attribute from here ?*

* *variable\_name.setAttribute(“attribute\_name”,”new\_value”)* - This allowed you to set an attribute

### 3/ Event

* Key word: *Listen to an event*
  + *Variable\_name.addEventListener(“event\_name”,function);*

*# Pay attention that the funtion go in syntax function() { block of code to be executed}* ***no function name was set***.  *We can also use a existed external function but without parenthese.*

* There are many possible events:
  + Clicks (“click”)
  + Hovers ( “mouseover” and “mouseout”)
  + Double Clicks (“dblclick”)
  + Drags
  + Much more in <http://developer.mozilla.org/en-US/docs/Web/Events>

## VII/ jQuery 3.x

### 1/ Introduction:

* Take the CDN (Content deliver network) *script* from website <http://code.jquery.com> . Notice that the URL inside block of code link to the jQuery code. We can see them by pause the link to address bar.

### 2/ Basic

* In jQuery, we use the dollar sight $, which is quite similar to document.querySlectAll.
* Syntax:
  + Call elements : *&(“element\_name”)*. (Var a = *&(“element\_name”)*.)

*#this return an object which contain a list of all element match element\_name. It’s quite similar to array but actually not it.*

* To access specific item of the elements, we use method .eq() : *a.eq(element\_index)*. Indexing can use reverse indexing: ***a.eq(-1)***
* *$('div').eq(index)* and *$('div')[index]* is worth noticing here. The first code will return a Jquery object while the second return a HTML DOM element(node) – which cannot be use with some jQuery method.
  + Change css: *a.css(“property\_name”,”new\_style”)*
  + This accept JavaScripy object for multiple change in css code.:
    - Var newCss = {

‘color’ : ‘white’,

‘background’ : ‘blue’,

‘border’: ‘20px solid black’

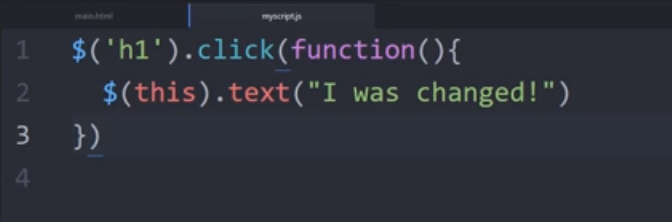
}

* + - *a.css(newCss)*
* Affect the content:
  + Variable\_name.textContent >>> *a.text(type\_thing\_you\_want\_to\_change\_here\_or\_leave\_it\_blank)*
  + Variable\_name.innerHTML >>> *a.html(sam\_as\_above)*.
* Affect attribute: *a.eq().attr(“attr\_name”,”value”)*
* Affect value:
  + First way: use affect attr as above with “value” in “attr\_name”
  + Second: *a.eq()..val(“new\_value”)*
* Affect Class:
  + *addClass().* Each class can only be add once.
  + *removeClass()*
  + *toggleClass()* This is to turn on and off. Mean non-exist >> add, oposite >> remove.

*# In this practice, I try to add both class at the same time. It did happen with a space between 2 classes. But only one was execute as both class affect on same property : “color” and “background”. But the .turnRed was always more priority than .turnBlue*

*Link: In H:\Dropbox\2019.04.12 Coder\2019.05.08 Jose\_Django-Python\2019.05.11\_Practical\_example\7.Jquery\Part1\_MyDocument.html*

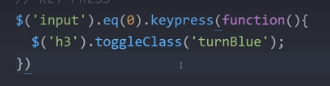
### 3/ jQuery Events:

* <https://Code.jquery.com/category/events>
* 

*$(‘h1’).click(function(){*

*$(this).text(“I was changed”)*

*})*

*  *<<< this is interesting*
* Syntax for event:
* $().eq().event\_name(function{

*#Block\_of\_code*

}

* We can also add event listener by the syntax :

*$().eq().on(“event”,function(){*

*#block\_of\_code*

*} )*

* The animation are avaiable in jQuery. Open the effects on code.jQuery.com to see all the animation. Some eg:
  + fadeOut(time)
  + slideUp(time)
* *Event object.*

### 4/ Front-End Project

* Check out functions at the bottom of the JavaScript files, they help us to understand table indexing.
* Prevent checking outside of the table ? indexing can return out-table value?
* *index()* method return index of the selected element
  + The .closest selector traverses up the DOM to find the parent that matches the conditions.
  + the .find selector traverses down the DOM where the event occurred, that matches the conditions.
  + **Notice** that find() and closet() is similar to & selector, which return a list of all element matched with condition. Seem like an array but not actually an array.

*#Eg: Below is the way to find the element with “.title” class when click In the button. Logic here is*

*Step 1: from button, traverses up to the closest parent elements with class name “.element”.*

*Step 2: from the element with class name “.element”, traverses down and pick the closest elements with class “.title*”



* 