HTML: skeleton (Mark-up language, what sort of content is in our page)

CSS: muscles (Design language, how does that content look like)

JS: brain (programming language, it makes web interactive, click and something happens)

1. In the same folder then you open VSC file.
2. The folder name should be lower case.
3. The file name should be lower case.

The circle means you didn’t save the file.

You need to save them to make them work!

Control+C : save

If I break all the roles, the browser will always try to show me something.

The browser doesn’t tell me there is some error on the html file.

Tag: put a tag to a text, that tag maybe can be a title / link

VSC close with <>

<food>kimchi</food> = the browser understands that kimchi is food, and this is HTML

* Beginning + name of tag + end of beginning + the content of tag + finish of content (/)

<li> list item

[Ordered list] OL

[Unordered list] UL

VSC prettier: if you make mistakes, it helps you. So every-time you saved,

Install > setting > editor > format on Save > check

<a> : anchor

Going to another website

If you put some website, it would be a link.

But <a> need more information than other links like <title> <h1>

“where we are going?” -> attributes (adding information to our text)

Href: HTTP reference or hyperlink reference

<a href=“http://google.com”>Go do Google</a>

Href is only can be combine with “anchor”.

Target

Target=“\_self” : you can open on a same tap. (deport)

Target=“\_blank” : you can open on a different tap.

Image: the self closing tag

<img />

<img>doesn’t have contents here </img>

<img src=“(put image address)” />

Src only works on image.

Local image that you have (not on the internet)

<img src=“logo.jpg” />

If the image file is not in the same folder…

<img src=“img/logo.jpg” />

**[Structuring HTML document]**

Every HTML documents HAVE TO start with…

<!DOCTYPE html>

Open our HTML tag

<html>

(content)

<head> (invisible, all the extra information)

<title> Home – my first website.</title>

</head>

<body>

</body>

</html>

[Invisible tags]

They show how the browsers show document

Mega : extra data (title, base, link, style, and script elements)

1. Content
2. Name

<meta name=“description” content=“This is my website />

Sometime the browser breaks so always put…

<meta charset=“utf-8” />

We want to help search engines like google

<html lang=“kr”>

og:image

the image that you can see when you share in KAKAO-TALK

<link

Rel=“shortcut icon”

Sizes=“16x16 32x32 64x64”

Href=“https://nomadcoders.co/m.png”

/>

\*When you search google, add the keyword “mdn”

\*never use w3 schools

<p>2<sup>5</sup></p>

* Select <sup> and control+d
* Change them same time

[Form]

<input type=“password” />

You don’t need to remember all the attributes.

<tagname attrName=”value”>(content) </tagname>(and put whatever you want)

<label for=””> same with <input id=””>

**ID**: an attribute that you can put anything inside of the body because it is a “unique identifier”.

The rule for ID is they can get only one ID per tag, and they need to be unique.

\*everything can have ID but only image can have src, type can only be on input.

[Content sectioning]

▸ Non-semantic tag

the tag doesn’t mean anything on its own but it has function.

<div> (division) is a box and can be placed next each other = “header”

if you want to put everything under each other, put them inside of DIV (box).

\*div doesn’t allow to be next to them! (block)

<span> is for short text, inline container

<span>(short text)</span>

\*span allow to be next to them! (inline)

<p> is for paragraph

<p></p>

\*paragraph doesn’t allow to be next to them! (block)

▸ Semantic tag

header, main, footer, address

<footer>

&copy; 2021 Dana

[CSS]

Cascading Style Sheet (CSS): they are read from top to bottom

▸ Put HTML code and CSS code on the same HTML file.

Add code <style>(CSS code)</style>

If they have id, point with #

It should be on the <head>, before all the <body> content

1. Point at the element : selector
2. Write whatever you want : property and value

<style>

h1 {

(property name)color: blue; (property value)

font-size: 20px; (finish with ; no space)

font-weight: 800;

fond-style: italic;

text-align: center;

}

</style>

\*there are not spaces

\*everything must end with ; (property name: propertyu value;)

▸ Seperate the HTML from the CSS. (more professional)

Make CSS file (.css)

Connect HTML file and CSS file with <link> tag

<link href=”styles.css” rel=”stylesheet” /> (Rel: relationship)

body {

background-color: red;

}

**BOX (block)**

<style>

div {

height: 150px;

width: 150px;

}

</style>

<body>

<div></div>

<div></div>

</body>

Block: paragraph, div, header

Inline: span(tiny text), link, image, code = share same length

display: block;

it change from inline to block.

display: inline;

it change from block to inline.

<style>

div {

display: inline;

height: 150px; (X diappeared)

width: 150px; (X it doesn’t work)

}

**inline** does not have height and width. \*they only have margin left and right.

**block** does have height and width.

features of block: margin, padding, border

**Margin**: space form the border of the box to the outside (default 8px)

margin: 0;

or

margin-top: 20px;

margin-bottom: 20px;

margin-right: 15px;

margin-left: 15px;

= margin: 20px 15px; (two value = top,bottom / right,left

margin: 20px (one value= all the sides)

margin: 20px 5px 12px 9px; (four value= top / right / bottom / left)

\*collapsing margins

when the white box’s border and the purple box’s are same, the margin becomes one.

So if we want separate them, we need “padding”.

**Padding**: space from the border of the box to the inside.

**Border**: border of the box

\*means everything!

<style>

\* {

border: 2px solid black;

}

</style>

**Class**: grouping each selectors

.something = name of class

.tomato = (class=”tomato”

#tomato = (id=”tomato”)

Some of them should be one way and some of them should be another way.

<style>

#tomato1,

#tomato2,

#tomato3,

* .tomato

.btn {

Padding: 5px 10px;

Border-radius: 5px;

}

.tomato {

background-color: tomato;

color: white;

~~padding: 5px 10px;~~

~~border-radius: 5px;~~

}

.teal {

background-color: teal;

~~padding: 5px 10px;~~

~~border-radius: 5px;~~

}

</style>

<span class(instead of id)=”tomato”>hello</span>

<span class=”btn tomato”>hello</span>

<span class=”btn tomato”>hello</span>

<span class=”btn teal”>hello</span>

<span class=”btn teal”>hello</span>

**Inline-block**: it allows you to have blocks with height, full margin, they are next to each other.

!But it becomes many problems. It has some default space, and old.

Inline-block do not support Responsive Design, it changes in different sizes.

**Flexbox**: they have three rules and they are important.

Rule 1. You do not talk to the children. Only talk to the parents(body).

And then you can make many other properties.

Justify-content: center;

Justify-content: flex-end;

Justify-content: space-evenly; (=some space distribute everywhere evenly)

This calculations are not for you.

Rule 2. Main axis is horizontal (default)

Property ex) justify-content: space-between

Rule 3. Cross axis is vertical (default)

Property ex) align-items: center;

Vh: viewport(screen) height

Body {

Height: 100vh; = 100% screen

\*flip

Flex-direction: column;

It makes “main axis” become vertical, and “cross axis” become horizontal.

Flex-wrap: nowrap;

It calls wrapping and makes everything in the same line.

Flex-wrap: wrap-reverse;

Flex-direction: column-reverse; or row-reverse;

It reverse 1-2-3 > 3-2-1

\*the default of direction of “flex-direction” is row. So main axis of default is horizontal.

<style>

html {

background-color: tomato;

}

Body {

Margin: 20px;

Padding: 20px;

Background-color: thistle;

}

div {

height: 150px;

width: 150px;

padding: 10px;

border: 2px dashed black;

}

span {

border-style: dotted;

color: teal;

padding: 20px;

}

#first {

background-color: whitesmoke;

}

</style>

<body>

<div id=”first”>

<div id=”second”>

<div id=”third”>

<div id=”forth”>

</div>

Color: teal / wheat /

.father {

Flex-direction: column;

align-items: center;

justify-content: space-between

flex-wrap: wrap;

}

position: fixed;

top: 00px;

bottom:00px;

right: 00px;

left: 00px;

ignore other <div>, different layer

position: relative;

top: 00px;

bottom:00px;

right: 00px;

left: 00px;

When you want to move something a little bit to the top/ bottom / right / left.

position: absolute;

top: 00px;

bottom:00px;

right: 00px;

left: 00px;

it will aline itself according the closed relative parents.

+position: absolute; to the parent

**Pseudo Selector**

more complex way to pointing elements like pointing certain elements in the class,

div: last-child{

background-color: teal;

}

OR

div: nth-child(2){

background-color: tomato;

}

even (2,4,6,8...) = 2n

odd (1,3,5,7...) = 2n+1

3n+1

p span {

color: teal;

}

<div>

<p>

<span>inside</span>

</p>

</div>

(p is parent and span is child)

span {

background-color: yellogreen;

text-decoration: underline;

}

p span {

color: white;

text-decoration: none;

}

↓

div > span

direct children <div>

div span {

text-decoration: underline;

}

<div>

<span>hello</span>

<p>

<span>inside</span>

</p>

</div>

\*div is parent and span is child

p + span

p + span {

text-decoration: underline;

}

<div>

<p>

<span>inside</span>

</p>

<span>hello</span>

</div>

\*span next to p so it’s a brother

p ~ span

<div>

<p>

<span>inside</span>

</p>

<address>hi</address>

<span>hello</span>

</div>

\*p is span’s brother but not just next to it

<style>

input {

border: 1px solid wheat;

}

input: required {

border-color: tomato;

}

</style>

<body>

<div>

<form>

<input type=”text” placeholder=”username” />

<input type=”password” required placeholder=”password” />

</div>

</body>

**attribute selector**

input [type=”password”] {

background-color: thistle;

}

input [placehoder~=”name”] {

background-color: pink;

}

<body>

<div>

<form>

<input type=”text” placeholder=”First name” />

<input type=”text” placeholder=”Last name” />

<input type=”password” required placeholder=”password” />

</div>

</body>

Assignment

width: 50%;

display: flex;

justify-content: space-between;

flex-wrap: wrap;

https://replit.com/@danhee25/ElasticLightblueMicroprogramming#style.css

(mine: you have to fix it)

https://replit.com/@danhee25/YouAreAwesomeBlueprint-2#styles.css

**State**

button: active{

background-color: tomato;

}

button: hover {

background-color: tomato;

}

\*hover means when the mouse is on top, not clicking. But you lose all style.

input: focus{

background-color: tomato;

}

\*focus is when it is selected.

<style>

a: visited {

color: tomato;

}

a: hover {

color: teal;

}

a: focus {

color: turquoise;

}

</style>

<body>

<a href=”https://apple.com”>Go to apple</a>

</body>

form: focus-within {

border-color: seagreen;

}

\*Focus-within is a state for when an element has child. So when the children of the form is focused…

form: hover input: focus {

background-color: sienna;

}

Input:: placeholder {

color: yellowgreen;

}

\*::placeholder allows to modify ‘placeholder’ because something you want to change placeholder.

p::selection {

color: white;

background-color: yellowgreen;

}

p::first-letter {

color: white;

font-size: 50px;

background-color: yellowgreen;

}

p::first-line {

color: white;

font-size: 50px;

background-color: yellowgreen;

}

**Transitions**

a {

color: white;

background-color: tomato;

text-decoration: none;

padding: 3px 5px;

border-radius: 5px;

transition: background-color 10s ease-in-out, color 5s ease-in-out; OR all 5s ease-in-out;

}

a:hover {

color: tomato;

background-color: wheat;

~~transition: all 5s ease-in-out;~~

}

\*Transition property should add to the element without the state.

\*it should be root element, not on the state.

Transition: all 1s cubic-bezier (0.6, 0, 0.735, 0.045);

Transition: background-color 1s cubic-bezier(0.6, 0, 0.735, 0.045);

**Transformation**

<style>

img {

border: 1px solid black;

border-radius: 50%; (it will be circle)

transform: rotateY (30deg); (3D)

}

</style>

<body>

<img scr=”img/logo.jpg” />

</body>

transform: scaleX(4) ;

transform: scale(2,2);

transform: rotateX (00deg) rotateY (00deg) rotateZ (00deg);

transform: translate(-10px);

transform: skew(45deg);

\*it doesn’t modify any other siblings like box elements, margins, paddings.

\*transformation is kind of 3D transformation.

<style>

Img {

Transition: transform 5s ease-in-out;

}

Img: hover {

Transform: rotateZ(90deg) scale(0.5);

}

\*transition should be on the state (root).

https://developer.mozilla.org/ko/docs/Web/CSS/transform

**Animation**

@keyframes (animation name) {

from {

transform: rotateX(0);

}

to {

transform: rotateX(360deg);

}

}

Img {

animation: (animation name) 5s ease-in-out;

}

[several stages]

@keyframes CoinFlip {

0% {

transform: rotateY(0);

}

50% {

transform: rotateY(180deg) translateY(100px);

}

100% {

Transform: rotateY(0);

}

}

Img {

animation: CoinFlip 5s ease-in-out infinite;

width: 50% (making image little bit small)

}

https://animate.style/

display: flex;

height: 100vh;

align-items: center;

justify-content: center;

**Assignmgnet**

<link rel="stylesheet" href="style.css">

@keyframes (animation name) {

from {

transform: rotateX(0);

}

to {

transform: rotateX(360deg);

}

}

@keyframes CoinFlip {

0% {

Transform: rotateY(0);

}

20% {

Transform: scale(2);

}

40% {

Border-radius: 0px;

Transform: rotateY(180deg) translateY(100px);

border-color: tomato;

}

60% {

Transform: scale(5);

}

80% {

Transform: rotateY(0) translateY(0px);

}

100% {

Transform: scale(5);

}

}

<style>

img {

border: 1px solid black;

border-radius: 50%; (it will be circle)

transform: rotateY (30deg); (3D)

}

Animation delay

(mine)

<https://replit.com/@danhee25/YouAreAwesomeBlueprint-4#style.css>

Assignment

Songs list = ul li tag

1. 전체 layout
   * 휴대폰 화면을 기준으로 크게, header, main-song, buttons, song-list 로 나눌 수 있습니다.
2. header
   * header의 title은 flex와 justify-content를 이용해 가운데 정렬을 구현할 수 있습니다.
   * text-align을 사용해도 괜찮습니다.
   * header에서 삭제버튼은 absolute를 사용해 구현할 수 있습니다.
   * position: absolute 를 위해 header에 position: relative 선언하는 것을 잊지마세요.
3. main-song
   * main-song 부분은 cover와 song data로 나눌 수 있습니다.
   * main-song에 flex 선언과 justify-content로 가로축을 일정 간격 맞춰줍니다.
4. buttons
   * buttons에는 font awesome에서 제공하는 아이콘들을 적절히 활용해줍니다.
   * buttons 역시 flex와 justify-content로 일정 간격 맞춰줍니다.
   * button의 입체효과는 border-radius와 border-bottom-width를 이용하면 쉽게 구현할 수 있습니다.
5. song-list
   * song-list 부분은 ul 태그를 이용해 구현해줍니다.
   * 세로로 정렬하기위해 flex-direction: column을 선언해줍니다.
   * 각 song은 main-song 파트와 마찬가지로 cover와 data로 나누어줍니다.

.songlist\_\_m {

display: flex;

}

<li class="songlist\_\_t">

<span class="third\_\_box"></span>

<span class="third\_color">

<span class="songlist\_\_name">Try Everything

<span class="songlist\_\_singer">Shakira

</span>

</span>

</span>

</li>

.gitignore is a file that you want to ignore.

<!-- (something) --!>

Programmer only can see the code.

**BEM (block element modifier)**

They have an easier way of reading CSS.

.btn{}

.btn\_\_price {}

.btn--orange {}

--is modifier.

.brn--big {}

<a class=”btn btn—tomato”>

.btn\_\_play {}

.btn\_\_like {}

.btn\_\_plus{}

<a class=”btn btn--big btn--orange”>

<span class=”btn\_\_price”>$99.9</span>

<span class=”btn\_\_text”>Subscribe</span>

</a>

Heroicons website (icon)

FontAwesome

**Font family**

Font-family: -apply-system,

Put middle

Justify-content: center;

Width: 33%;

.status-bar\_\_column: nth-child(2) {

Display: flex;

Justify-content: center;

}

.status-bar\_\_column:last-child{

Display:flex;

Justify-content: flex-end;

Align-items: center;

}

https://fonts.google.com/

<ul class="songlist\_box">

<li class="songlist\_box\_m">

<span class="songlist\_box\_m\_col"></span>

<span class="songlist\_box\_m\_content">

<span class="songlist\_box\_m\_content\_top">Matargasti

<span class="songlist\_box\_m\_content\_btm">Mohit Chauhan

</span>

</span>

</span>

</li>

box-shadow

[gradutation]

2nd picture = index.html (index = first page)

Press ! and ENTER

<head>

<title>Welcome to Kokoa Talk</title>

</head>

<body>

<div id=”status-bar”>

<div class=”status-bar\_\_column”></div> (it has three columns)

<div class=”status-bar\_\_column”>

<span>No service</span>

<i class="fas fa-wifi"></i>

</div>

<div class=”status-bar\_\_column”>

<span>18:43</span>

</div>

<div class=”status-bar\_\_column”>

<span>11%</span>

<i class="fas fa-battery-quarter"></i>

<i class="fas fa-bolt"></i>

</div>

</div>

<header class=”welcome-header>

<h1 class=”welcome-header\_\_title>Welcome to Kokoa Clone</h>

<p class=”welcome-header\_\_text”>If you have a Kokoa Account, log in with your email or phone number.</p>

</header>

<form id=”login-form”>

<input type=”text” placeholder=”Email or phone number” />

<input type=”password” placeholder=”Password” />

<input type=”submit” value=”Log in” />

<input type=”submit” value=”Sign in” />

</form>

<a href=”#”>Find Kakao Account or Password</a>

</body>

<html last>

<script src="https://kit.fontawesome.com/e65c8db91a.js" crossorigin="anonymous"></script>

<link rel=”stylesheet” href=”style.css”>

= link:css

Reset css

Make a new css file and copy

@import “reset.css” to the css file.

:not()

Body:not(div)

<form action=””>

Which page are you going to send this data.

<form method=”get or post”>

* + 1. Post is to send to the server on the backhand.
    2. Get is insecure, you shouldn’t send username and password.

nav>ul>li\*4>a

<nav>

<ul>

<li><a href=””></a></li>

<li><a href=””></a></li>

<li><a href=””></a></li>

<li><a href=””></a></li>

</ul>

</nav>

If you want to fix position whenever you scrolled, position:fixed;

Box-sizing: border-box;

You do not want to modify your box size. Whenever you add padding size.

If you want to add padding but you don’t want to increase the box.

Display: block;

Make <span> to <div>shape

❤

If you want icon position on top of the other icon,

Position: absolute;

Put parents

Position: relative;

(remember) Aline itself with the closest relative parents.

(transform: translate X)

Align-items: center;

Text-transform: uppercase;

If you need ABCDE for design, they should be lower case in html and you can change in CSS

Place center

Margin-right: auto;

Margin-left: auto;

OR

Justify-content: center;

Align-items: center;

z-index shows the order of the layers

it’s for absolute position or fixed position

z-index: 2;

order only works on flex children.

order: 0;

order:1;

OR

Flex-direction: row-reverse;