



Web Site ?  $\Rightarrow$  just text file (code)

text file  $\rightarrow$  browser  $\rightarrow$  web site

내가 쓴 코드를 브라우저가 웹사이트로 생성.

To do? 어떤 종류의 text를 어디에 쓰는가?

HTML?

웹사이트는 2-3개의 언어로 이루어짐 html, CSS, JavaScript

Super simple

browser에게 content들을 설명해주는 역할  $\Rightarrow$  HTML만 가능

CSS?

Cascading Style Sheets

HTML과 반드시 함께 사용된다.

browser에게 웹사이트가 어떻게 보여야 하는지에 대해 알려줌

JavaScript?

html - 뼈대 CSS - 근육

JavaScript - 뇌

brain of website  $\Rightarrow$  사용자와 상호작용 (interactivity)

프로그래밍 언어이다.

• ✕ 파일명, 폴더명 반드시 소문자로

브라우저는 HTML 파일에 에러가 있다고 알려주지 않는다.  
=> 뭐가 잘못되는지 모름.

html, CSS, JS를 Googling 할때 mdn을 블여 구글링.

W3School 참고하지 말것

필요한 TAG를 구글링하여 찾아보기 다양한 속성들도 알수 있음

수업에서는 form TAG 해보았음 => 정보를 제출하기 위한 대화형 칸트를  
을 포함하는 문서 구획

\* id : 태그당 하나만 사용하는 Unique 값 (고유값)

## Semantic HTML

Semantic Tag : 이름만 보아도 의미 짐작 가능 (div처럼 기능은 있으나 content를 나눌 때)

div로 채우기보다 Semantic Tag를 최대한 활용하는 것이 코드 해석에 편함

=> 나, 팀원들이 보기에도 이해하기 쉽게 만들것

<header> <footer> 등이 존재.. mdn 검색

<main> <address>

<p> 긴 문단 </p> <span> 짧은 문장 </span>

attribute에는 큰따옴표 사용 할것. 항상

# CSS

- 위에서부터 차례로 적용 (cascading style sheets)

선택자(selector) 와 효과

태그 지정 후 원하는 속성값 사용

box

div, header, main 등으로 box를 만들 수 있다.

block level element - 옆에 아무것도 올수없음

inline element - 같은줄에 여러개 올수있음 (글, 링크, 이미지)

대부분의 box는 block, block이 아닌것을 기억하는게 편함 - Span, a, Image

block만 가지는 특징 : 높이와 너비를 가진다. (inline은 없다.)

\* Margin, padding, border

margin - box의 border(경계) 바깥에 있는 공간

margin은 두개 쓰면 위, 아래와 좌, 우 마진 값을 설정할 수 있다. (값이 하나면 사방에)  
너개 쓰면 위 → 오른쪽 → 아래 → 왼쪽 (사계방향)

위, 아래에서 일어나는 (collapsing margin : box끼리의 경계가 같을 때 일어남 [위, 아래])  
=> 하나의 margin값으로 통일되어 버린다.

padding - box의 경계로부터 안쪽에 있는 공간

\* id명 선택자 → #id명 {}

border - box의 경계, 여러속성이 있으나 웃eng어서 한개만 쓰는편언어! solid black

## selector \*은 모든 항목 선택

(CSS는 위에서부터 아래로 실행되므로 같은내용을 다시 쓸 필요가 없다는 것을 기억하자)

Inline element에도 Margin과 padding이 적용될까?

⇒ padding은 정상작용, inline은 높이와 너비가 존재하지 않으므로 Margin이 양쪽만 적용

요소를 가리킬수 있으면서 중복을 허용하는 class 속성.

id ⇒ unique Class ⇒ 여러 요소에서 사용할 수 있음.

선택자  $\begin{cases} \#hi & - id = "hi" \\ .hi & - class = "hi" \end{cases}$

하나의 요소에는 여러개의 class가 존재할 수 있다. 반면 id는 한개만 있어야 한다.

Class를 통해 CSS의 중복코드를 제거할 수 있다.

Vs code 디자인 선택

Ctrl + d + 방향키

```
.btn {  
    padding: 5px 10px;  
    border-radius: 5px;  
}  
  
.tomato {  
    background-color: tomato;  
    color: white;  
}  
  
.teal {  
    background-color: teal;  
}  
/style>  
</head>  
  
<body>  
    <span class="btn teal">hello</span>  
    <span class="btn tomato">hello</span>  
    <span class="btn teal">hello</span>  
    <span class="btn tomato">hello</span>  
    <span class="btn teal">hello</span>  
    <span class="btn teal">hello</span>  
    <span class="btn tomato">hello</span>  
    <span class="btn teal">hello</span>  
</body>
```

→ class를 통해 중복 style 설정

여러개 가능

# display 속성

block

inline

inline-block - block을 inline처럼 보이게 함, width와 height 가질 수 있음

↳ 오래되고 치명적인 오류, 단점이 많아서 별로다. (default 값이 지향대로임)  
반응형 design도 지원하지 않음..

⇒ flex box ⇒ 2d 레이아웃에서 아주 잘 작동, 놓고 싶은 곳에 놓을 수 있음.

rule 1. 자식 element에는 어떤 것도 적지 않을 것, 부모 element에만 명시 해야 함

rule 2. 주축(Main axis), 교차축(Cross axis)

수평(변경 가능)      수직(변경 가능)

\* 단위 vh → 100vh는 화면 높이의 100% (Viewport height)

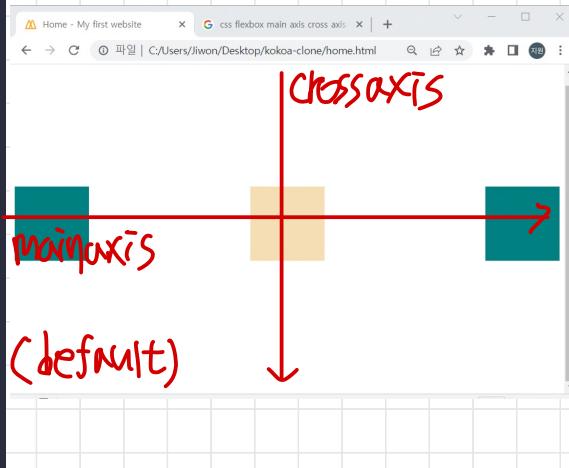
```
<style>
  body {
    height: 100vh;           ← height가 설정안되었으면 적용
    margin: 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }

```

```
div {
  width: 300px;
  height: 300px;
  background-color: teal;
}
```

```
#second {
  background-color: wheat;
}
</style>
```

- mainaxis에 적용되는 속성  
- crossaxis에 적용되는 속성



## flex box - part 2

- flex-direction : column  $\Rightarrow$  주축은 수직이되고 고차축은 수평이됨 (default: row)
- flex container 안에 있는 자식요소도 flex container가 될 수 있다.

```
<style>
  body {
    height: 100vh;
    margin: 20px;
    display: flex;
    flex-direction: column; 주축, 고차축 변경
    justify-content: center; 고차축, 하중
    align-items: flex-end; 주축 해당
  }
```

```
<style>
  body {
    height: 100vh;
    margin: 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }

  div {
    display: flex;
    justify-content: center;
    align-items: center;
    font-size: 28px;
    width: 300px;
    height: 300px;
    background-color: #teal;
  }
}
```

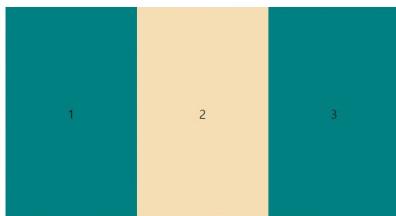
자식요소도  
의외로 flex container가  
될 수 있음

=>

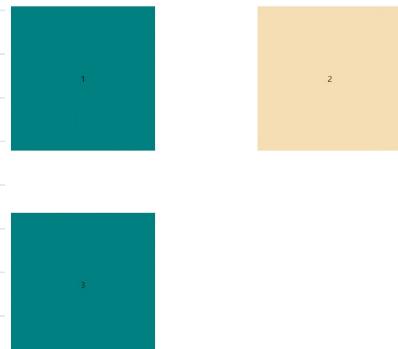


- flex-wrap : nowrap (default)  $\Rightarrow$  한 줄로 맞추기 위해 width  $\frac{1}{3}$ 이 변동될 수 있음  
wrap  $\Rightarrow$  명시된 width 사이즈를 반영, 초과한 한 줄에 넣고  
불가능 하면 다음줄로 넘김.

nowrap



wrap



flex direction: column-reverse (가로줄 바꾸기)  
row-reverse

**fix wrap : wrap-reverse (움직이는 순서 바꿈)**

**Position** - 레이아웃보다는 위치를 아주 조금 움직일 때 쓴다. **default = static**

`position: fixed;` 대상의 위치 고정(위,아래로 스크롤 해도)

`top, bottom, left/right` 를 통해 위치를 정하면 `block` 이든 `inline` 이든 신경쓰지 않고 배치된다. (다른 layer 가 둘)



position : relative => element가 처음 위치한 곳을 기준으로 수정

element가 처음 놓인 자리에서 상하좌우로 움직임.

position : absolute ⇒ 가장 가까운 relative 부모를 기준으로 이동  
relative한 부모를 찾지 못하면 body를 기준으로

pseudo Selector - 세부적으로 element를 선택 할 수 있음

=> pseudo selector를 사용하는 것이 class와 id를 사용하는 것보다 훨씬 좋은 방법이다.

## First Child

:)ast child 마지막

## :nth-child() 선택

```
body {  
    height: 100vh;  
    margin: 50px;  
}
```

```
span {  
    background-color: tomato;  
}
```

```
span:nth-child(3n+1) {  
    background-color: teal;  
}
```

## Combinatorics

```
<style>
    body {
        height: 100vh;
        margin: 50px;
    }

    span {
        background-color: yellowgreen;
        padding: 5px;
        border-radius: 10px;
    }
}

div>span {
    text-decoration: underline;
}

div p span {
    color: white;
}

</style>
end>

dy>
<div>
    <span>hello</span>
    <p>
        abcdefu abcdefu abcdefuabcdefuabcdefu abcdefu abcdefu abcdefu
        abcdefu abcdefu abcdefu. <span>inside</span>
    </p>
</div>
ody>
```

`p+span $ $` = P 바로 다음으로 오는 형제 관계의 span을 가지킴 (not Inside)

# pretty useful

## pseudo selector part 2

४

$P \sim \text{Span}\{3\} = \text{Span}$ 이  $P$ 의 형제이지만 바로 뒤에 오지 않을 때

**attribute selectors** - TAG의 속성을 통해 선택하기

```
        }
        input:optional {
            border: 1px solid wheat;
        }
        input:required {
            border-color: tomato;
        }
        input[placeholder~="name"] {
            background-color: pink;
        }
    </style>
</head>

<body>
    <div>
        <form>
            <input type="text" placeholder="First name" />
            <input type="text" placeholder="Last name" />
            <input type="password" required placeholder="password" />
        </form>
    </div>
</body>
```

Name 韓國人豆丁

First name  Last name  password

# States - 상태에 따라 CSS변화 가능

```
<meta name="description" content="This is my website" />
<style>
    button:active {           ← 버튼이 active 상태(클릭 된) 일때
        background-color: #tomato;
    }
</style>
</head>
```

```
<body>
    <div>
        <button>hello</button>
    </div>
```

```
        <style>           ← button에 커서가 올라가 있을 때
            button:hover {
                background-color: #tomato;
            }
        </style>
```

```
<style>
    input:focus {           ← input이 선택되었을 때
        background-color: #tomato;
    }
</style>
</head>
```

```
<body>
    <div>
        <input type="text" />
        <input type="text" />
    </div>
```

```
        <style>
            a:visited {           ← 방문했던 링크 일 때
                color: #tomato;
            }
        </style>
```

```
</head>

<body>
    <div>
        <a href="https://apple.com">Go to apple</a>
    </div>
```

```
title="Name" description="Content">This is my  
<style>  
    form {  
        border: 1px solid #salmon;  
        display: flex;  
        flex-direction: column;  
        padding: 20px;  
    }  
  
    form:focus-within {  
        border-color: #seagreen;  
    }  
</style>  
<head>  
<body>
```

→ 자식 요소가 focus 되었을 때  
부모 요소의 형식 바뀜

```
<form>  
    <input type="text" name="" id="">  
    <input type="text" name="" id="">  
    <input type="text" name="" id="">  
</form>  
  
form:hover input { → form이 hover 상태일 때 input 요소 변경  
    background-color: #sienna;  
}
```

→ form이 hover 되고  
input이 focus 일 때  
input 색상 바뀜

# Color and Variables

## Color system

- hexadecimcal color - #000000 (16진수)

- RGB - (0,0,0)

- RGBA (0,0,0,0.5)

color picker - chrome extension

변수를 이용해 Color 설정하기 (Custom property, Variable)

```
<style>
  :root {
    --main-color: #fcce00;      - root
  }

  p {
    background-color: var(--main-color);
  }

  a {
    color: var(--main-color);
  }
</style>
```

여기 main color 를 써서 적용

색 뿐만 아니라 border 등 다양한 효과 저장 가능

# ADVANCE CSS

Transition - 어떤 상태에서 다른 상태로의 변화를 애니메이션으로 만드는 방법

```
<meta name="description" content="THIS IS MY WEBSITE" />
<style>
  a {
    color: wheat;
    background-color: tomato;
    text-decoration: none;
    padding: 3px 5px;
    border-radius: 5px;
    font-size: 15px;
    transition: background-color 10s ease-in-out, color 5s ease-in
  }
  a:hover {
    color: tomato;
    background-color: wheat;
  }
</style>
<head>
</head>
<body>
  <style>
    a {
      color: wheat;
      background-color: tomato;
      text-decoration: none;
      padding: 3px 5px;
      border-radius: 5px;
      font-size: 15px;
      transition: all 5s ease-in-out;
    }
    a:hover {
      color: tomato;
      background-color: wheat;
    }
  </style>
</body>

```

10초동안 바뀐다.  
배경색  
ease in-out  
color

\* transition은 반드시 State가 아닌 element 선택자에 존재해야 함.

\* transition은 변화가 존재하는 항목에 적용되므로 State를 바꿨을 때 적용된다.

애니메이션을 통해 변화하는 모든 항목 지원 가능하다.

애니메이션의 속도를 linear, ease, ease-in 등으로 변경 가능하다.  
Cubic-bezier(0,0,0,1) 윤선을 통해 가속/감속을 customize 할 수 있다.  
hover에 특히 유용함.

## Transformation → 3d 변환 가능

속성이 많음 img등을 3d 관점에서 다양하게 변화 시킬 수 있음.

- transformation은 다른 box element, 이미지에 영향을 끼치지 않는다.
- transition과 결합하여 사용할 수 있다.

```
<style>
  img {
    border: 2px solid black;
    border-radius: 50%;
    transition: transform 5s ease-in-out;
  }

  img:hover {
    transform: rotateZ(90deg); img가 hover 될 때 90도 회전
  }
</style>
</head>

<body>
  
  <span>hahahahahahaha</span>
</body>
```

transfor<sub>z</sub> transition을 이용해 사용

## Animations 1 - 원하는 만큼 애니메이션 만들기 및 재생

```
<head>
  <style>
    @keyframes superSexyCoinFlip { → 애니메이션 정의
      from {
        transform: rotateX(0);
      }
      to {
        transform: rotateX(360deg);
      }
    }
    img {
      border: 5px solid black;
      border-radius: 50%;
      animation: superSexyCoinFlip 6s ease-in-out;
    }
  </style>
</head>
<body>
  
</body>
```

img의 animation 불러오기 (infinite 속성으로 무한반복 가능)

## Animation 2. transform 사용법

```
<style>
  body {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
  }
  @keyframes superSexyCoinFlip {
    0% {
      transform: rotateY(0);
    }
    50% {
      transform: rotateY(180deg) translateY(100px);
      opacity: 0; 투명도.
    }
    100% {
      transform: rotateY(0);
    }
  }
  img {
    border: 5px solid black;
    border-radius: 50%;
    animation: superSexyCoinFlip 6s ease-in-out infinite;
  }
</style>
```

기운데로 옮기기

%로 진행 정도에 따라 animation 설정 가능

Media Query - CSS만을 이용해서 스크린의 사이즈 알 수 있음  
=> 스크린 size에 따라 CSS 바꿀 수 있음

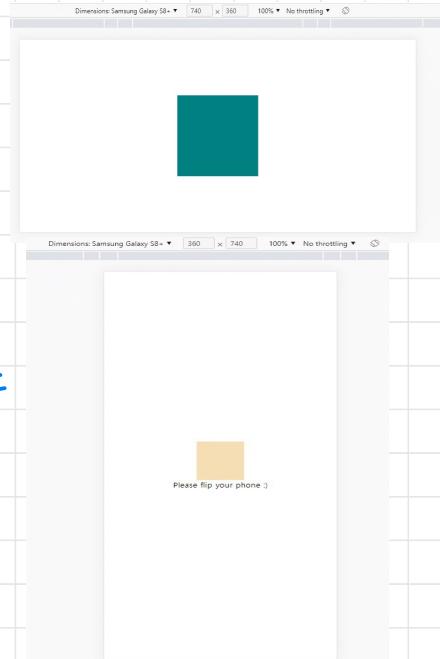
```
<head>
<style>
  body {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    flex-direction: column;
  }
  div {
    background-color: teal;
    width: 200px;
    height: 200px;
  }
  @media screen and (max-width: 600px) {
    div {
      background-color: tomato;
    }
  }
  @media screen and (min-width: 601px) and (max-width: 1200px) and (orientation: portrait) {
    div {
      background-color: wheat;
    }
  }
  span {
    font-size: 36px;
  }
  @media screen and (orientation: landscape) {
    span {
      display: none;
    }
  }
</style>
</head>
<body>
  <div></div>
  <span>Please flip your phone :)</span>
</body>
```

미디어 쿼리 사용법

width 600 높으면

601 {screen < 600, phone 세로로

→ 가로모드 일 때 display:none



\* Git - 변경 내용 확인, Version Control system

\* Git hub - 변경 내용 업로드

## Github

repository - 코드를 넣을 폴더 (변경내용과 히스토리를 갖고 있다.)

Commit - 파일을 저장하고 삽입한 Pointing time  
들여다보면 삭제, 추가된 변경사항 확인 가능 (식별은 없지만 한다.)

### 2way

1. 컴퓨터에 폴더 생성 후 해당폴더를 github에 올리는 방법
2. 처음부터 github에 repository 만드는 방법 - Good way

Github desktop - 초보자들이 github로 작업을 하기 위한 도구 - 작동원리를 쉽게 이해할 수 있다.

repository를 컴퓨터에 Clone → VsCode로 파일 열기

README.md - 서식이 있는 문서 작성 (Mark down)  
모든 Repository가 갖고 있어야 한다.  
#을 통해 제목설정

⇒ Github desktop에서 commit [ 반드시 title 써어야 함 ]

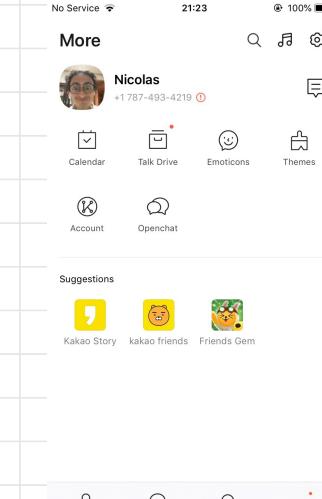
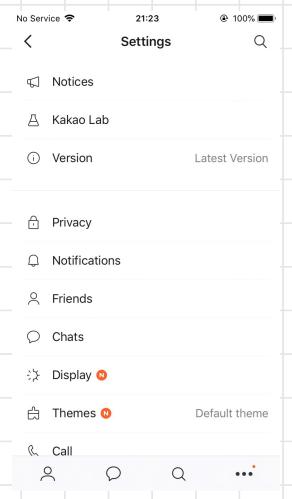
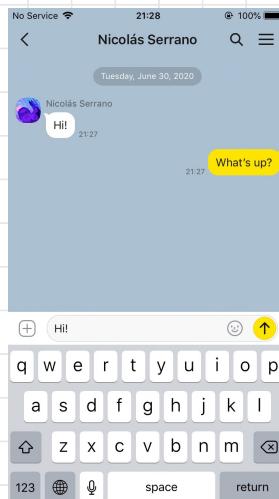
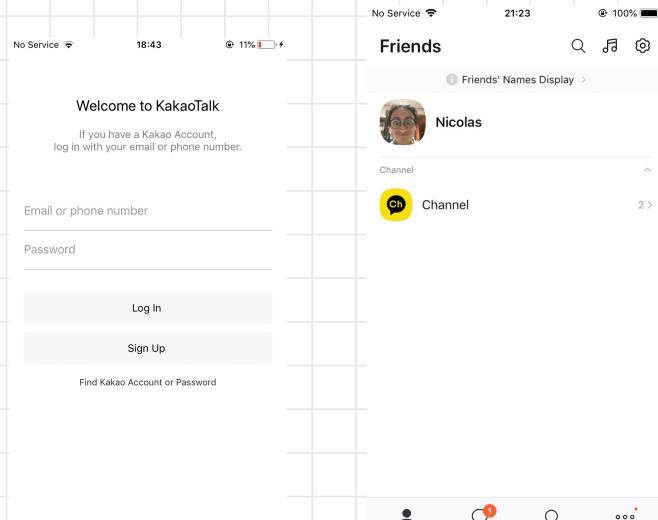
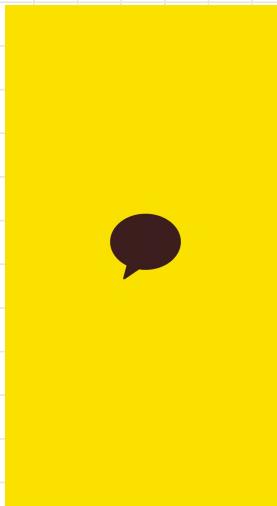
⇒ PUSH Github에 변경사항 전달

Git이 컴퓨터에서 업데이트 된 것을 알게됨

# CLONING TIME

• \*.fitIgnore 파일을 만들어 파일명쓰면 fit에서 해당파일 무시

구현 할 것



\* 대부분의 웹서버는 index.html을 디폴드로 찾아보도록 설정되어 있다.

class 명은 너무 표준적이지 않은것으로.

## BEM (Block Element Modifier)

→ 좀더 쉽게 읽히는 CSS 형식을 가지는 것

→ Id와 class가 헷갈릴때 많음, class로 통일

```
/* Block component */  
.btn {}  
  
/* Element that depends upon the block */  
.btn__price {}  
  
/* Modifier that changes the style of the block */  
.btn--orange {}  
.btn--big {}
```

클래스가 너무 커질수 있어 비선호 하는 이용자도 있다.

```
<a class="btn btn--big btn--orange" href="https://css-tricks.com">  
  <span class="btn__price">$9.99</span>  
  <span class="btn__text">Subscribe</span>  
</a>
```

## Icons (직접 아이콘 구하기 아 SVG (픽셀없는 파일 only 수학))

Font Awesome, Hero font 등에서 사용

\* <script> TAG는 항상 </body> 뒤에 (마지막)에 쓸 것

(CSS) font-family 를 통해 폰트 지정 가능, Google Fonts 사용 가능

Reset CSS - CSS파일으로 대부분의 태그에 margin: 0, padding: 0, border: 0 등을 가진다. 브라우저에 의해 적용되는 스타일을 없애 완전한 초기 상태로 시작

CSS not - 뭔가가 적용되는 걸 원치 않을 때 사용

cursor - 마우스 포인터 설정

color: inherit;

부모로부터 상속받는 값

## form의 중요한 속성

action - 어떤 페이지로 데이터를 보낼건지 지정할 수 있다.

method POST - 백엔드 서버에 정보를 전송하는 방식

GET - 보안에 취약함, Username, password를 이 방식으로 보내면 안된다.

No Service

18:43

11% 🔋⚡

log in page 완성

Welcome to KokoaTalk

If you have a Kokoa Account, log in with  
your email or phone number.

Email or phone number

Password

Log In

[Find Kokoa Account or Password](#)

# friends

Google 검색엔진 → Navigation 찾아서 UI의 li 안에 있는 link를 가져오게 설정

## Navigation 만들기

shortcut을 이용해 빠르게 HTML 작성 가능 (VScode)

nav>ul>li\*4>a



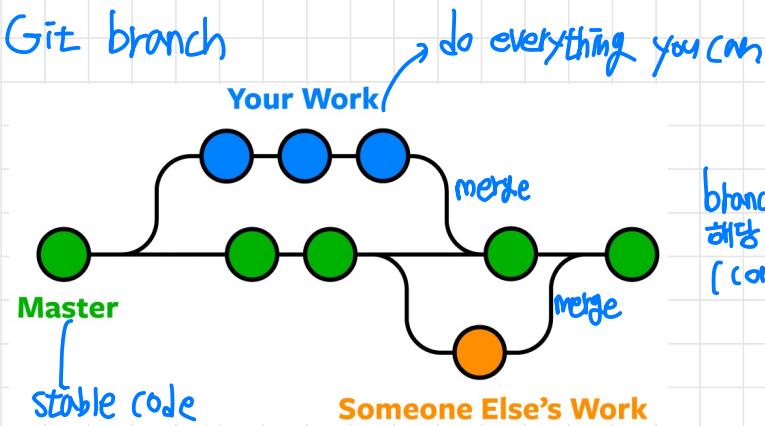
```
<nav>
  <ul>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
  </ul>
</nav>
```

```
.nav{
  position: fixed;
  bottom: 0;
  width: 100%; screen의 100%로.
  background-color: #f9f9fa;
  padding: 20px 40px;
  box-sizing: border-box;
  border-top: 1px solid rgba(121, 121, 121, 0.3)
}
```

Navigation bar 위치를  
하단에 고정

→ width를 설정하고 padding을 줌기 때문에  
box 전체의 크기가 width + padding이 되어버림  
box-sizing을 통해 box size를 맞춰 더듬어나는 것을 막음.

- ※ 사각형을 원으로 만들 때 border-radius 값을 width의 절반으로 사용.
- ※ 공통으로 사용되는 요소들은 Components로 따로 저장하여 page마다 재사용 할 수 있다. => 시간, code 간결
- ※ 같은 code가 여러번 반복되면 변수로 저장하여 사용하자.
- ※ position: fixed, display 작업시 layer가 생겨는데 z-index를 통해 순서 변경 가능
- ※ flex display 사용시 order: 으로 순서 바꿀 수 있음
- ※ animation의 last keyframe을 유지하기 => forwards
- ※ will-change → browser에게 변화될 사항을 미리 알려줌.
- ※ :focus-within property는 선택자 내부의 element 가 focus 됐을 때를 의미함



branch를 바꾸면 vscode에서도 해당 branch 내용으로 바뀜  
(commit 까지 원래했을 때)

## \* Github pages

Github에서 무료로 Static 호스팅을 할 수 있게 해준다.

⇒ 누구나 자신의 웹사이트를 무료로 업로드 할 수 있다. (Github에서 free URL 제공)

\* Static website? only with HTML, CSS, JavaScript (only front-end, can't back-end)

### 사용법

1. 이중(gh-pages) Branch 만들기 (반드시 해당 이름) (repository - public)

2. publish branch

URL → USERNAME.github.io / REPOSITORY NAME

### how to update?

1. Master branch에서 수정사항 수정 후 Commit

2. gh-pages branch로 이동

3. branch TAB의 update from master 선택 (Master에 있는 모든 commit 가져옴)

4. push