

# Ver.2021 노베이스 모던 웹개발

## - 3. 당장 필요한 것만 배우는 CSS

작성자: 경상국립대학교 컴퓨터과학과 조교 이자룡

### 목차

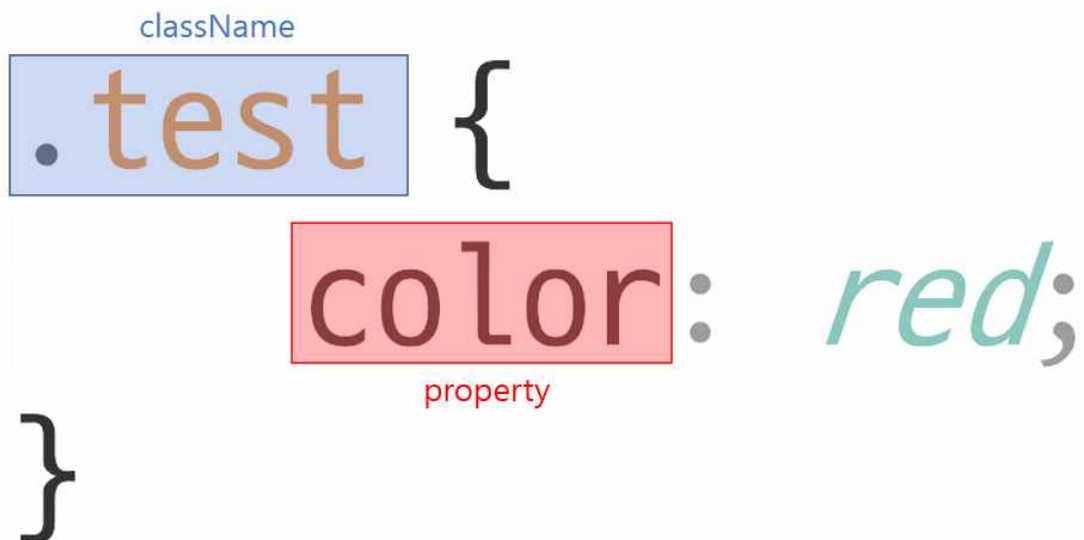
p.2	-	0. CSS의 구조
p.11	-	1. 폰트
		2. 박스모델
p.14	-	2-1. border, padding, margin, background-color
p.18	-	2-2. display: inline, block, inline-block, weight, height
p.21	-	3. position: relative, absolute, fixed
p.29	-	4. 레이아웃: grid
p.34	-	5. 마치며

## 0. CSS의 구조

\* 이 수업을 시작하기 전에 반드시, JavaScript, TypeScript, React hook 기초를 알고 시작하길 바란다. 《ver.2021 노베이스 모던 웹개발》 첫번째 강좌인 『React 시작 전에 알아야 할 JavaScript, TypeScript』, 두번째 강좌인 『React Hook』이 준비되어 있다.

아니, CSS하나 배우는 게 알아야 할 것이 이렇게나 많은 말인가? 그것보다는, CSS를 건드리려면 기본적인 프로그래밍을 할 줄 알고 배우라 이 말이다.

css 는 디자인을 위한 언어이다. 버전 3 기준 문법은 다음과 같다.



The diagram illustrates the structure of a CSS rule. It shows a selector '.test' in a blue box, labeled 'className' above it. This is followed by an opening curly brace '{'. Below the selector, the property 'color' is shown in a red box, labeled 'property' below it. This is followed by a colon ':', the value 'red' in a green italicized font, and a semicolon ';'. The entire rule is enclosed in a closing curly brace '}'.

적용하고 싶은 className 을, 점 찍고 쓴다. className 은 소문자로 시작하는 낙타체다. 중괄호로 감싸고, 쓰고자 하는 프로퍼티 이름을 쓴다. 콜론 : 찍고 값 쓰고 세미콜론 써주면 된다.

바로 적용해보자. CodeSandbox 에서 React TypeScript 템플릿을 사용한다.

참고로, 이번 CSS 강좌는 이 장, CSS의 구조가 핵심이다. 나머지 장들은 덜 중요하다.

```
App.tsx x
1  import "../styles.css";
2
3  export default function App() {
4    return (
5      <div className="App">
6        <h1>Hello CodeSandbox</h1>
7        <h2>Start editing to see some magic happen!</h2>
8      </div>
9    );
10  }
11

styles.css x
1  .App {
2    font-family: sans-serif;
3    text-align: center;
4  }
```

line1: 우리가 지금까지 무시했던 첫 번째 줄. css 파일 import

line5: React에선 무조건 class가 아니라 className 이라고 쓴다. React에서 class는 다른 의미이기 때문이다. React 말고 일반적인 HTML 에서도 className 을 써도 잘 동작한다.

styles.css에서 .App 은 <div className="App">을 의미한다. 즉, 그 안에다가 CSS를 적용한다는 뜻이다.

React에서 CSS를 적용하는 여러 방법이 있지만, 하나의 방법만 표준적으로 사용한다. 바로 CSS 파일을 import하는 것이다.

```
styles.css ●
1  .App {
2    font-family: sans-serif;
3    text-align: center;
4    color: red; /* 이걸 빨간색이다 */
5  }
```

color: red; 를 추가했을 경우. 주석은 // 는 허용하지 않고, 여러줄 주석인 /\* \*/ 만 허용한다.

결과:

## Hello CodeSandbox

Start editing to see some magic happen!

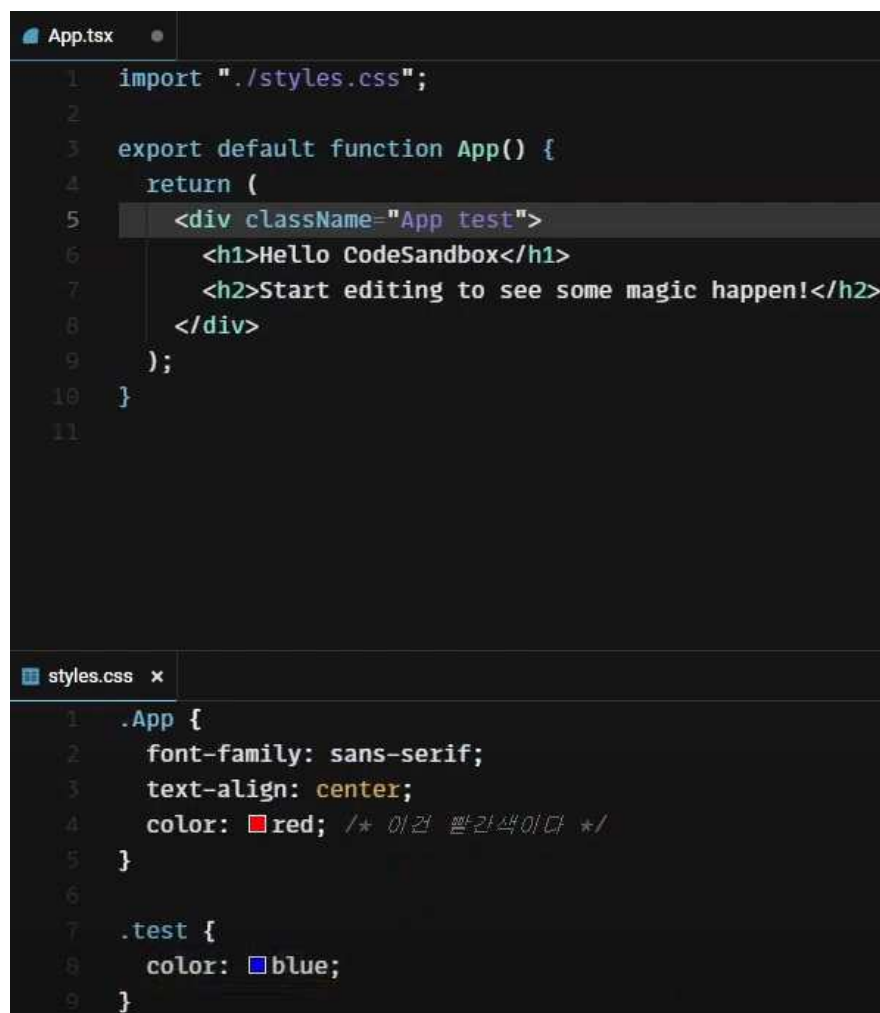
화면 전체의 글자가 빨간색으로 바뀌었다.

축하한다. 여러분의 첫 css 코드를 적용시켜보았다. 이제부터 CSS의 정말 중요한 원리를 하나하나 알아보겠다.

하나의 태그에 여러개의 클래스 적용

CSS 파일 순서배치에 따른 차이

태그 안에서의 순서변경



```
App.tsx
1 import "../styles.css";
2
3 export default function App() {
4   return (
5     <div className="App test">
6       <h1>Hello CodeSandbox</h1>
7       <h2>Start editing to see some magic happen!</h2>
8     </div>
9   );
10 }
11

styles.css
1 .App {
2   font-family: sans-serif;
3   text-align: center;
4   color: red; /* 이걸 빨간색이다 */
5 }
6
7 .test {
8   color: blue;
9 }
```

App.tsx

line5: App 과 test 의 두 개의 className 을 설정했다.

styles.css


line7-9: test 클래스 추가

결과:

## Hello CodeSandbox

Start editing to see some magic happen!

왜 빨간색이 아니고 파란색으로 나왔을까? styles.css 에서 .App이 먼저 나왔기 때문이다. 순서를 바꿔보면 빨간색이 된다.



```
1 .test {  
2   color: blue;  
3 }  
4  
5 .App {  
6   font-family: sans-serif;  
7   text-align: center;  
8   color: red; /* 이건 빨간색이다 */  
9 }
```

결과:

## Hello CodeSandbox

Start editing to see some magic happen!

즉, 나중에 나온 것이 적용된다. 부모자식 관계가 아닌 동등한 관계일 경우엔 나중에 나온 것이 적용된다.

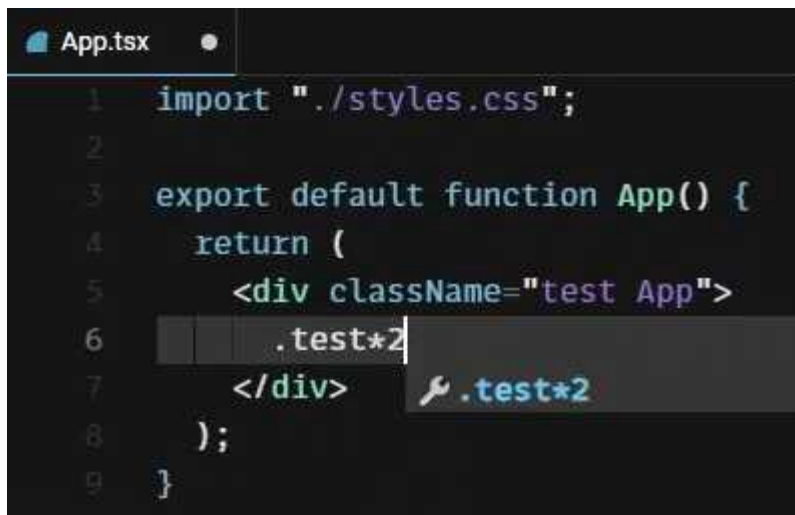
```
import "../styles.css";

export default function App() {
  return (
    <div className="test App">
      <h1>Hello CodeSandbox</h1>
      <h2>Start editing to see some magic happen!</h2>
    </div>
  );
}
```

그러나 이렇게, 태그 안에서 순서를 바꾸는 것은 아무 의미가 없다. 즉, 태그 안에선 순서는 의미가 없다.

Emmet

동일한 css 프로퍼티를 가진 클래스를 한번에 처리  
부모 자식 관계일 경우



```
App.tsx
1 import "../styles.css";
2
3 export default function App() {
4   return (
5     <div className="test App">
6       .test*2
7     </div>
8   );
9 }
```

효율적인 코딩을 위해 다음을 알아보자. 꺾은 괄호 이후에 클래스 이름을 쓰고 별표 찍고 2를  
쓴다음 탭키를 누르면

```
App.tsx
1  import './styles.css';
2
3  export default function App() {
4    return (
5      <div className="test App">
6        <div className="test">*/div>
7        <div className="test">*/div>
8      </div>
9    );
10 }
```

다음과 같이 test 라는 className을 가진 div 태그가 두 개 생긴 것을 알 수 있다. 이것은 **emmet** 이라는 패키지에서 제공하는 기능인데, VSCode 는 기본적으로 emmet 이 포함되어 있다. 우린 vSCode를 설치한 적 없지만 사실 CodeSandbox의 에디터는 VSCode이다. 만약 다른 텍스트 에디터, 예를 들면 sublime text 를 이용할 경우에는 수동으로 설치해줘야 작동한다. **emmet** 을 배워두면 코딩을 매우 효율적으로 할 수 있으니, 검색해서 주요 기능들을 알아두면 좋다.

여러 클래스가 동일한 css 프로퍼티를 가질 땐 다음과 같이 한다.

```
App.tsx
1  import './styles.css';
2
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="test1">test1</div>
7        <div className="test2">test2</div>
8      </div>
9    );
10 }
11

styles.css x
1  .App {
2    color: red;
3  }
4  .test1,
5  .test2 {
6    color: blue;
7  }
```

결과:

```
test1
test2
```

styles.css

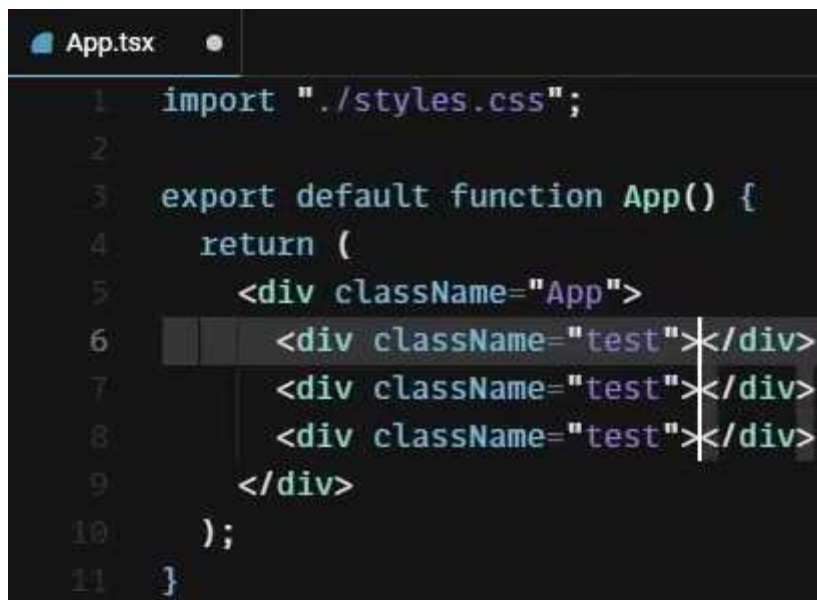
line4 - : .test1과 .test2를 콤마(,)를 사용해 묶어준 것이다.

여기서 왜 파란색이 적용되었을까? css 파일에 배치된 순서때문 아닐까라고 생각 들 수 있는데 이건 부모 자식의 문제다. App 클래스는 부모이고, test1, test2 클래스는 자식이다. **원래 css는 부모의 속성을 그대로 따라간다.** 그래서 만약 test1 과 test2에 아무것도 지정하지 않았을 때 빨간색으로 표시되는 것이다. **만약 부모의 말을 안 듣고 싶으면 자식에서 새로운 값으로 설정해주면 된다.**

## 다중커서

여러개 태그를 하나의 클래스로 지정

vscode에선 다중커서를 지원한다. Alt 누르고 마우스를 클릭 클릭 클릭해주면 된다.



```
App.tsx
1  import "../styles.css";
2
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="test"></div>
7        <div className="test"></div>
8        <div className="test"></div>
9      </div>
10    );
11  }
```

다음과 같이 작성해보자



```
App.tsx
1  import "../styles.css";
2
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="test">test1</div>
7        <div className="test">test2</div>
8        <div className="test">test3</div>
9      </div>
10    );
11  }
12

styles.css x
1  .test {
2    color: red;
3  }
```

결과:

test1  
test2  
test3

원래 클래스는 하나의 태그에만 쓰라고 있는 용도가 아니다. 그래서 이와 같이 여러 개 태그에 하나의 클래스를 지정해서 한번에 처리할 수 있다. 만약 딱 하나만 쓰고 싶으면 className 대신 id라는 걸 사용할 수 있으나, 웬만하면 class만 써주자.

CSS에서 가장 중요한 부분인, CSS의 기본적인 구조는 다 배웠다.

지금부터 주요 프로퍼티들을 다뤄볼 것이다. 모든 프로퍼티를 다루진 않는다. 당장 웹개발에 필요한 프로퍼티들만 다룬다.

그리고 하나 강조하고 싶은데, **여기 있는것들을 전부 외우려고 하지 마라.** 학교 공부하듯 공부하지 말라. 내가 CSS를 배우는 데 추천하는 방법은, **수백번의 실수를 통해 익히고, 작동이 잘 안 될 경우 이 강의를 계속 반복해서 참고**하는것이다. 아무리 열심히 해도 1주일뒤면 까먹을 지식들이라 생각된다.

CSS는 크게 네 부분으로 나뉜다.

1. 폰트
2. 박스모델
3. 포지션
4. 레이아웃

## 1. 폰트

글자와 관련된 부분이다.

font-size      폰트 사이즈를 바꾼다. 단위는 rem을 사용한다.  
color          글자의 색깔이다.  
text-align     정렬할 때 쓴다.

```
App.tsx x
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="test">test</div>
7      </div>
8    );
9  }

styles.css x
1  .test {
2    font-size: 6rem;
3    color: ■tomato;
4    text-align: right;
5    border: 1px solid □black;
6  }
```

test1, test2 를 지워버리고 test로 고쳤다.

css 파일엔 .App 부분을 지워버렸다.

text-align에선 오른쪽정렬을 위해 right 로 설정했다. left, right, center, justify(양쪽 정렬) 중 선택하면 된다.

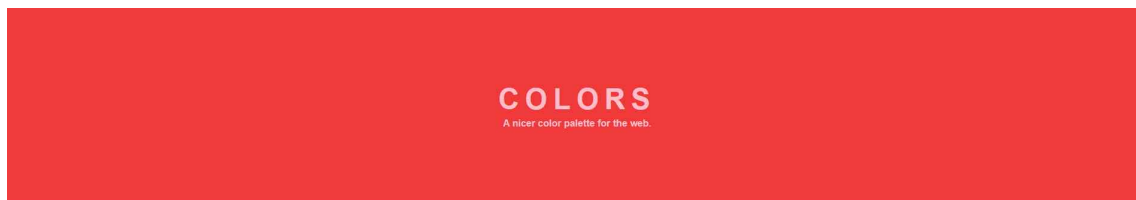
그리고 정확한 결과를 보기 위해 border를 설정해주었다. 나중에 배우게 될 것이다.

결과:



css를 사용하다보면, red, blue, yellow 등 기존의 색이 촌스러운수도 있는데, 세련된 색을 쓰고싶으면 구글 검색을 해보자.

검색어: css color recommendation



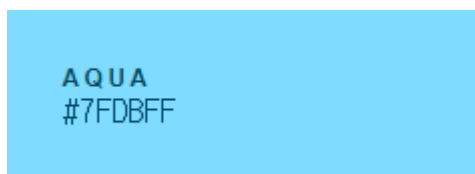
#### The New Defaults

Skining your prototypes just got easier - colors.css is a collection of skin classes to use while prototyping in the browser.

647B minified and gzipped.

NAVY #001F3F	BLUE #0074D9	AQUA #7FDBFF	TEAL #208C8C
PURPLE #800080	ROSEB #FF00FF	MAROON #8B0000	
RED #FF0000	ORANGE #FF8C00	YELLOW #FFFF00	

이것은 상단에 뜨는 아무 사이트나 클릭한 것이다. 보면 꽤 괜찮은 색상들이 있는데, 쓸만한 색상은 헥스 코드로 표현되어 있다. 헥스 코드는 RGB 값을 표현한 번호이며, #으로 시작하고, CSS에서 바로 쓸 수 있다.



이 색상을 가져와보겠다.

```
styles.css ●
1  .test {
2      font-size: 6rem;
3      color: ■ #7FDBFF;
4      text-align: right;
5      border: 1px solid □ black;
6  }
```

색상을 헥스 코드로 입력하면 된다.

결과:



test

## 2. 박스모델

### 2-1. border, padding, margin

모든 태그는 박스로 이루어져있다.

border 테두리. 굵기, 선 스타일, 색깔을 지정한다.

```
styles.css
1  .test {
2    font-size: 6rem;
3    color: #7FDBFF;
4    text-align: right;
5    border: 1px solid black;
6  }
```

우린 이미 사용해서, text 라는 글자 주위에 테두리를 그렸다.

1px 픽셀은 단위이다. 숫자를 올리면 굵어진다.

solid 선 스타일이다. solid 말고는 거의 쓸 일이 없는 것 같다.

black 선 색상이다.

1px solid black 이라고 꼭 순서대로 쓸 필요는 없다. black solid 1px 이라고 뒤죽박죽으로 해도 잘 동작한다.

margin과 padding에 대해 배워보자. 일단, 직관적으로 이해하기 위해 이미지를 보자.



Border는 우리가 만들어본 테두리다.

Margin은 다른 태그 또는 화면과의 여백이다.

Padding은 Border와 내용물 사이의 여백이다.

다음과 같이 코드를 써보았다.

```
App.tsx x
1  import "../styles.css";
2
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="test1">test1</div>
7        <div className="test2">test2</div>
8      </div>
9    );
10 }

styles.css x
1  .test1,
2  .test2 {
3    border: 10px solid black;
4    margin-top: 10px;
5    margin-bottom: 10px;
6    margin-left: 10px;
7    margin-right: 10px;
8  }
```

margin 은 top, bottom, left, right 로 나뉜다.

\* margin-top 이라고 일일이 치고있지 마라. matop 이라고 쓰면 자동완성 리스트가 뜰 것이다. 자동완성을 잘 활용해야 코딩 오타실수도 줄일 수 있다.

이걸 한번에 쓸 수도 있다.

```

1  .test1,
2  .test2 {
3      border: 10px solid black;
4      margin: 10px 10px 10px 10px;
5  }

```

시계방향 순서다. 위, 오른쪽, 아래, 왼쪽

결과:



즉, margin은 두 콘텐츠 사이, 또는 화면의 경계와의 여백이다.

padding 은 다음과 같다.

```

1  .test1,
2  .test2 {
3      border: 10px solid black;
4      padding-top: 10px;
5      padding-bottom: 10px;
6      padding-left: 10px;
7      padding-right: 10px;
8  }

```

margin과 똑같은 문법을 쓴다. 네 개로 나누지 않고, margin 처럼 한 줄로 끝낼수도 있다.



결과:



즉, padding은 콘텐츠와 border 사이의 간격이다.

연습삼아 컬러를 넣어보자.

```
styles.css
1  .test1,
2  .test2 {
3    border: 10px solid black;
4    padding: 10px 10px 10px 10px;
5    background-color: blue;
6  }
```

컬러는 background-color를 쓴다.

결과:



색을 넣어봤다. 자세히 보면, test1과 test2가 웹 화면 한 칸 전체를 차지하고 있다. 왜냐면, <div> 태그는 display 가 block 속성이기 때문이다.

## 2-2. display: inline, block, inline-block, weight, height

display는 해당 태그가 화면에 어떻게 나올지를 정의하는 프로퍼티다.

모든 태그는 기본적으로 다음 세 가지 중 하나를 가진다.

inline	쓰인 영역만 차지함. 크기 지정 불가능
block	가로 전체를 차지함. 크기 지정 가능
inline-block	쓰인 영역만 차지하는것으로 시작하나, 크기 변경 가능. 레이아웃에 많이 사용됨

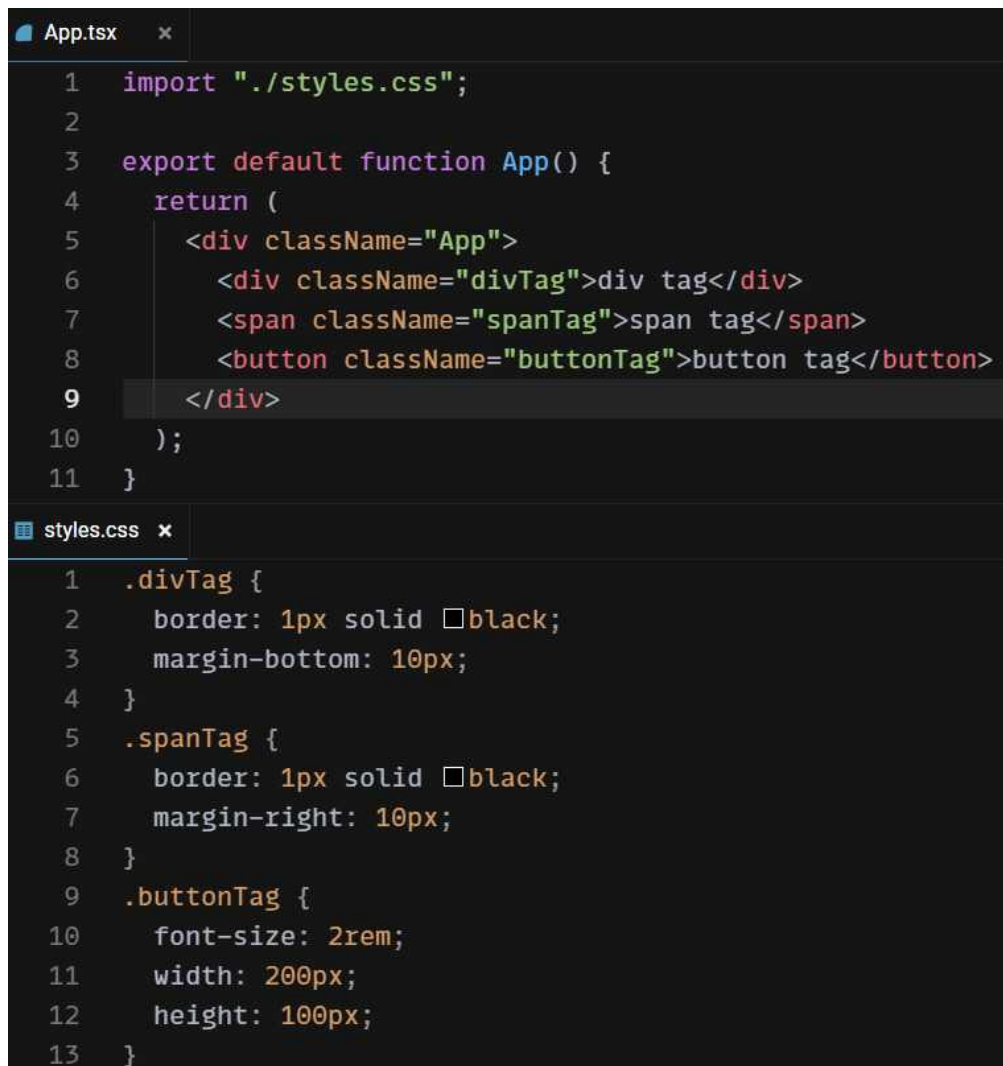
대표적인 태그들은 다음과 같다.

inline: <span>, <a>

block: <div>, <p>, <h1>

inline-block: <button>

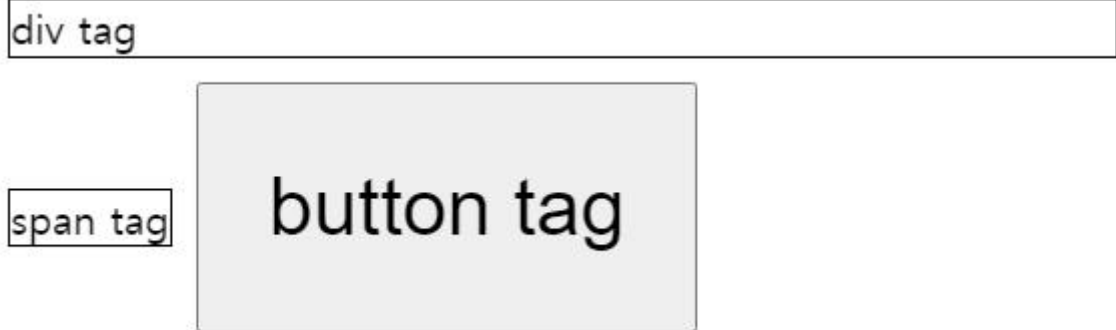
코드로 확인해보자. 다음과 같이 작성했다.



```
App.tsx x
1 import "../styles.css";
2
3 export default function App() {
4   return (
5     <div className="App">
6       <div className="divTag">div tag</div>
7       <span className="spanTag">span tag</span>
8       <button className="buttonTag">button tag</button>
9     </div>
10   );
11 }

styles.css x
1 .divTag {
2   border: 1px solid black;
3   margin-bottom: 10px;
4 }
5 .spanTag {
6   border: 1px solid black;
7   margin-right: 10px;
8 }
9 .buttonTag {
10  font-size: 2rem;
11  width: 200px;
12  height: 100px;
13 }
```

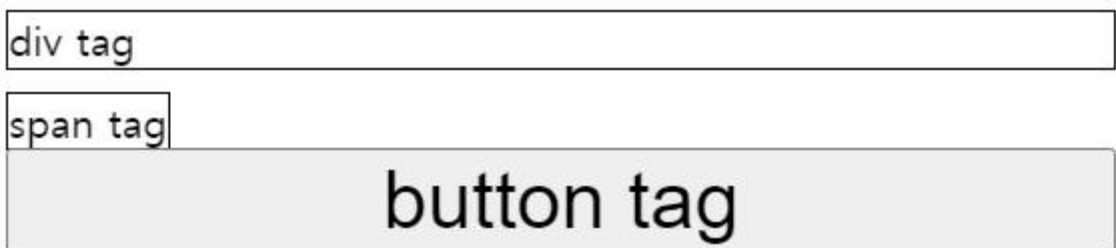
결과:



<div>는 한 줄 전체를 차지한다. display 가 block 으로 설정되어 있기 때문이다.  
<span>은 딱 정해진 영역만 차지한다. display가 inline 으로 설정되어 있기 때문이다.  
inline 으로 정해진 태그는 높이와 너비를 지정할 수 없다.  
<button>은 정해진 영역을 차지하지만 높이와 너비를 지정할 수 있다. width와 height 로 높이와 너비를 설정했다.

weight의 경우, px이 아닌 %로도 조절할 수 있다. 이 경우 화면 전체 너비에서 퍼센트로 계산한다. 그러나 height는 설정할 수 없다.

```
.buttonTag {  
  font-size: 2rem;  
  width: 100%;  
  height: 1%;  
}
```



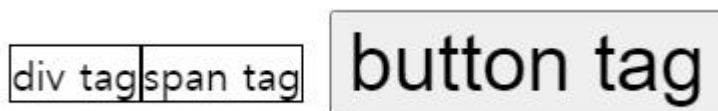
버튼이 화면 너비의 100%를 차지했기 때문에 block과 똑같아졌다. 그러나 높이는 아무리 바뀌도 바뀌지 않는다.

다시 원래대로 돌려놓자.

display의 기본값은 변경 가능하다. block 이었던 <div> 를 inline으로 바꿔보자.

```
.divTag {
  border: 1px solid black;
  display: inline;
}
```

결과:



\* 참고로, 코딩에 익숙하지 않은 사람들이 자주 하는 실수는 모든 것을 <div> 태그로 처리하고 필요할 때 display를 inline, inline-block 으로 바꿔서 해결하는 것이다. 이는 매우 좋지 않은 습관이다. 코드도 길어지고, 가독성도 떨어진다. 딱히 의미없는 영역인데 inline 요소의 무엇인가가 필요하다면 <span> 태그를 사용하고, <div> 와 <span> 으로 모든 걸 해결하지 말고 가능하면 최대한 시맨틱 태그들로 구분해주자.

display 프로퍼티에선 다음도 쓰일 것이다.

- none    화면에서 안보이게 하는 것. JavaScript 를 통해 none을 on, off 하는 방식으로 화면의 태그를 숨기고 나오게 하는 용도로 쓴다.
- grid    레이아웃 만들 때 사용한다. 수업 후반부 레이아웃 장에서 자세히 다룬다.

### 3. position: relative, absolute, fixed

태그의 위치를 옮길 때 사용한다. 단순히 옮기면 되는 간단한 문제가 아니라, "무엇을 기준으로 옮길지" 아는 것이 중요하다. 그 기준은 부모와 밀접한 관련이 있다.

조부모, 부모, 자식태그가 필요하다.

```
App.tsx  x
1  import './styles.css';
2
3  export default function App() {
4    return (
5      <div className="App">
6        <div className="grandDad">
7          Grand Dad
8          <div className="dad">
9            Dad
10           <div className="me">me</div>
11         </div>
12       </div>
13     </div>
14   );
15 }
```

```
styles.css  x
1  .grandDad,
2  .dad,
3  .me {
4    border: 1px solid black;
5  }
```

결과:

Grand Dad
Dad
me

position은 따로 설정하지 않으면 static이다. static일 경우, 움직이는것은 불가능하다.


relative부터 알아보자.

```
.me {
  position: relative;
}
```

다음 코드를 입력해보자. 결과창은 아무 변화가 없을 것이다. 그러나, 움직임을 주면 변화가 생긴다.

```
.me {
  position: relative;
  left: 100px;
}
```

결과:

Grand Dad	
Dad	
	me

100px

헛갈리지 마라. left 면 오른쪽으로 갈 것이다. top, bottom, left, right 를 사용할 수 있다.

여기서도 px 말고 % 를 사용할 수 있다. 그러나 top과 bottom 은 %가 작동하지 않고, left와 right만 작동한다.

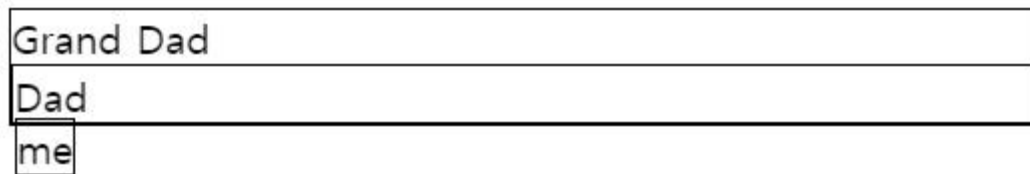
중요한 건, 기준이다. relative는 부모 태그로부터 자신의 위치를 결정한다. 원래 자기가 있었던 위치에서 이동한다.

다음은 absolute 이다. absolute 를 쓸 때 가장 먼저 확인해야하는건, 부모가 static이 아닌지이다. 부모가 static이고 조부모가 static이 아니라면 조부모를 따르고, 전부 다 아니라면 화면의 왼쪽 위 모서리를 기준으로 삼는다.

부모를 relative로 정해주겠다.

```
.dad {  
  position: relative;  
}  
.me {  
  position: absolute;  
  top: 20px;  
}
```

결과:

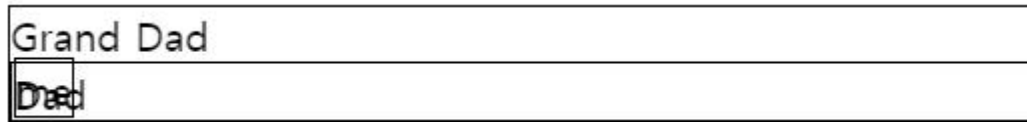


부모인 Dad 를 기준으로 20px 아래로 내려왔다. absolute 는 top, left, right, bottom 모두 %를 사용할 수 있는데, 화면이 아닌 부모태그가 기준인 %이다.

이번엔 부모가 static이면서 조부모가 static이 아닌 경우다.

```
.grandDad {  
  position: relative;  
}  
  
.dad {  
}  
.me {  
  position: absolute;  
  top: 20px;  
}
```

결과:

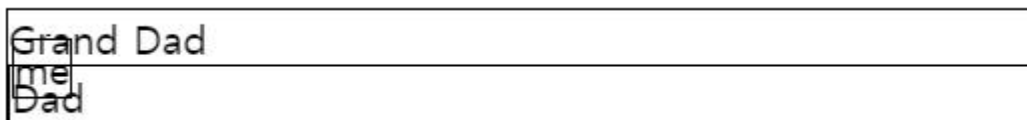


조부모를 기준으로 20px 내려왔다.

조부모도, 부모도 모두 static일 경우는 화면의 왼쪽 위를 기준으로 한다.

```
.grandDad {  
}  
  
.dad {  
}  
  
.me {  
  position: absolute;  
  top: 20px;  
}
```

결과:



마지막으로 배워볼 것은 fixed이다. 이것은 시조 태그의 위치를 기준으로 하며, 스크롤도 무시한다. px과 % 모두 적용 가능하다.

스크롤을 테스트해보기 위해 랜덤 문자를 넣어볼 것이다. 일일이 치지 말고, lorem 이라는 것을 이용하면 된다.

연습해보자. App.tsx 에서 lorem 누르고 tab 키



```
import "../styles.css";

export default function App() {
  return (
    <div className="App">
      <div className="grandDad">
        lorem
        <div className="dad">
          Lorem ipsum dolor sit amet, consectetur adipisicing elit. Fugit quia
          quae, aliquid possimus iste rem harum corporis? Ex sit, amet eaque
          minima repudiandae unde consectetur dolorum praesentium, odio ullam
          doloremque!
          <div className="me">me</div>
        </div>
      </div>
    </div>
  );
}
```

결과:

lorem
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Fugit quia quae, aliquid possimus iste rem harum corporis? Ex sit, amet eaque minima repudiandae unde consectetur dolorum praesentium, odio ullam doloremque!
me

그러나 스크롤을 위해선 아직 한참 부족하다. 엄청나게 써보자. App.tsx 에서 lorem\*10  
쓰고 tab 키

lorem

Lorem ipsum dolor sit amet consectetur adipisicing elit. Repellendus, aliquam facilis architecto totam vel, id explicabo repudiandae ex et sed officia similique qui sint autem quasi, asperiores delectus tempore vitae! Repellendus odit suscipit sed corporis neque recusandae libero cupiditate, dolorem voluptate! Repellendus quos odit impedit nam praesentium harum rem sequi consectetur quam earum voluptatibus deserunt excepturi, iusto eaque atque molestias. Earum quod repudiandae beatae, hic voluptas autem quibusdam? Esse dignissimos possimus voluptatem porro quasi aliquid blanditiis deserunt quo, aliquam autem sed accusantium. Vitae ducimus sunt quidem mollitia. Odio, atque nesciunt. Cupiditate saepe optio quidem doloribus adipisci enim voluptatem debitis at rerum et neque, molestias quo quod non magnam voluptatibus provident earum aliquid illo vitae officiis atque sunt quia! Quas, dicta. Soluta velit accusantium praesentium, beatae, provident dolor quaerat ipsum placeat ea quisquam quae assumenda voluptatem esse laboriosam suscipit voluptatibus libero delectus totam corporis neque similique voluptatum perspiciatis. Rem, corrupti voluptatem. Aut modi corrupti delectus maxime nemo. Quos error, excepturi maxime aspernatur a natus odit itaque unde quas. Quas, autem quaerat! Eum alias, aperiam tempore id animi corrupti quae reprehenderit quia. Repellat eum adipisci debitis ipsum impedit, esse incidunt laborum accusamus libero id eveniet ad velit earum harum, doloremque nemo exercitationem, commodi

스크롤이 생겼다.

```
.me {  
  position: fixed;  
  top: 0px;  
  background-color: blue;  
}
```

fixed로 설정하고, top 을 0px 로, 컬러는 blue 로 설정하자. 참고로, position 값을

줬기에 더이상 block 이 아니기때문에 적당한 글자를 쳐줘야한다.

```
<div className="me">me</div>
```

결과:

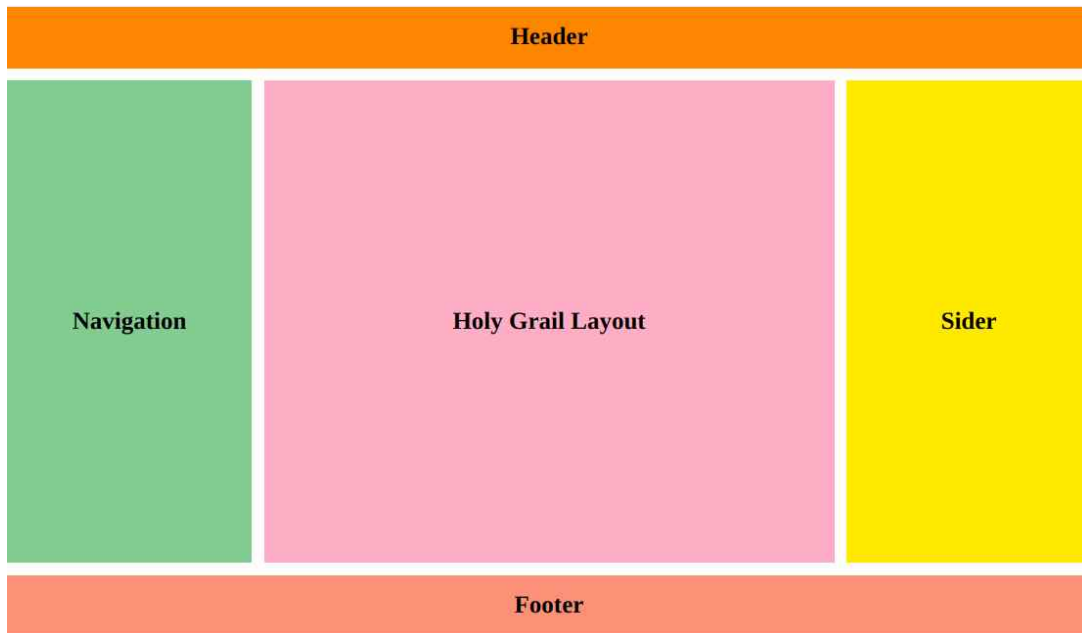
io quidem doloribus adipisci enim voluptatem debitis at rerum et neque, molestias quo quod non magnam voluptatibus provident earum aliquid illo vitae officiis atque sunt quia! Quas, dicta. Soluta velit accusantium praesentium, beatae, provident dolor quaerat ipsum placeat ea quisquam quae assumenda voluptatem esse laboriosam suscipit voluptatibus libero delectus totam corporis neque similique voluptatum perspiciatis. Rem, corrupti voluptatem. Aut modi corrupti delectus maxime nemo. Quos error, excepturi maxime aspernatur a natus odit itaque unde quas. Quas, autem quaerat! Eum alias, aperiatur tempore id animi corrupti quae reprehenderit quia. Repellat eum adipisci debitis ipsum impedit, esse incidunt laborum accusamus libero id eveniet ad velit earum harum, doloremque nemo exercitationem, commodi vitae cupiditate consequuntur nam amet perferendis placeat voluptatum! Minus? Aliquid tempora ea esse perferendis quas adipisci inventore laborum, ex laudantium accusantium atque facilis minus nobis, magnam eveniet unde sit recusandae commodi, amet impedit ab! Nisi quasi illo error pariat. Libero magnam repellat illum obcaecati quia aliquid? Ipsum voluptatibus voluptatem nihil quos doloremque omnis ad ut harum? Alias, veritatis dicta, quo fuga asperiores aut porro commodi ducimus eligendi laudantium expedita! Eius sapiente iusto et nobis ratione? Architecto aliquid et culpa minima, quos doloribus cumque incidunt placeat adipisci! Ducimus sint dolorum numquam reprehenderit. Quisquam, magni et ea corrupti hic officiis veniam.

스크롤 내려도 고정되는 파란 상자가 생겼다.

정리해보면,

static	기본값. 움직임 불가능
relative	부모 태그로부터 자신의 위치 결정됨. 원래 자기가 있었던 위치에서 이동
absolute	부모의 위치가 기준. relative 와는 기준이 다름. 부모가 static이면 부모를 무시함. 부모가 static이고 조부모가 static이 아닐 경우 조부모가 기준
fixed	display는 더 이상 block 아님 시조 태그의 위치로부터 이동. 심지어 스크롤도 무시함. display는 더 이상 block 아님

## 4. 레이아웃: grid



우리가 흔히 보는 웹 화면이다. 이런 레이아웃을 만들어 볼 것이다. grid 를 이용해보자.

```
App.tsx
1  import "../styles.css";
2
3  export default function App() {
4    return (
5      <div className="App">
6        <header className="header">header</header>
7        <nav className="nav">nav</nav>
8        <main className="main">main</main>
9        <aside className="aside">aside</aside>
10       <footer className="footer">footer</footer>
11     </div>
12   );
13 }
```



```
styles.css x
1  .App,
2  .header,
3  .nav,
4  .main,
5  .aside,
6  .footer {
7    border: 1px solid black;
8  }
```

결과:

header
nav
main
aside
footer

이제 nav, main, aside 부분만 옆으로 나누면 될 것이다. 이 셋을 통합하는 부모태그를 만들어줘야한다.

```
<div className="container">
  <nav className="nav">nav</nav>
  <main className="main">main</main>
  <aside className="aside">aside</aside>
</div>
```

container 라고 이름지었다.

```
.container {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
}
```

display를 grid 로 줘야한다.

결과:

header		
nav	main	aside
footer		

grid-template-columns 는 각 columns 으로 나눔을 뜻한다. row 와 column, 행과 열은 자주 헷갈리게 될 것인데, 다음과 같이 외우자.

가로	세로
로우	컬럼
행	열

손으로 화살표를 그려가면서, 가로 세로 로우 컬럼 행 열

어쨌든, 각 column 으로 나뉜 것을 볼 수 있다.

fr 은 뭘까? 비율이다. 세 개의 태그가 부모태그인 <container> 안에 있고, 모두 1:1:1 의 비율을 차지한다.

grid-template-columns 만 바꿔보자.

```
.container {  
  display: grid;  
  grid-template-columns: 150px 1fr 1fr;  
}
```

결과:

header		
nav	main	aside
footer		

화면 크기를 조정해보았다. nav만 150px로 고정이고, 나머지는 nav를 뺀 나머지에서 1:1 비율을 차지한다.

다음과 같이 바꿔보자.

```
.container {
  display: grid;
  grid-template-columns: 1fr 2fr 1fr;
}
```

결과:

header		
nav	main	aside
footer		

각각 1:2:1 비율로 나뉜다.

column이 있다면 row 도 있을 것이다.

```
.container {
  display: grid;
  grid-template-rows: 1fr 2fr 1fr;
}
```

main이 좀 더 많은 비율을 차지하고, row 로 바뀌었다.

header
nav
main
aside
footer

main 에 lorem 을 넣어보자.



```
<main className="main">
  Lorem ipsum dolor sit amet consectetur
  fugit, architecto mollitia consequat
  suscipit ipsa facilis, consequat
  quasi nostrum enim harum.
</main>
```

결과:

header
nav
Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus sed fugit, architecto mollitia consequuntur blanditiis quos libero autem suscipit ipsa facilis, consequat repellat maxime aliquam officiis quasi nostrum enim harum.
aside
footer

main이 커진 만큼, nav와 aside 도 커져 정해진 비율을 그대로 유지한다.

## 5. 마치며

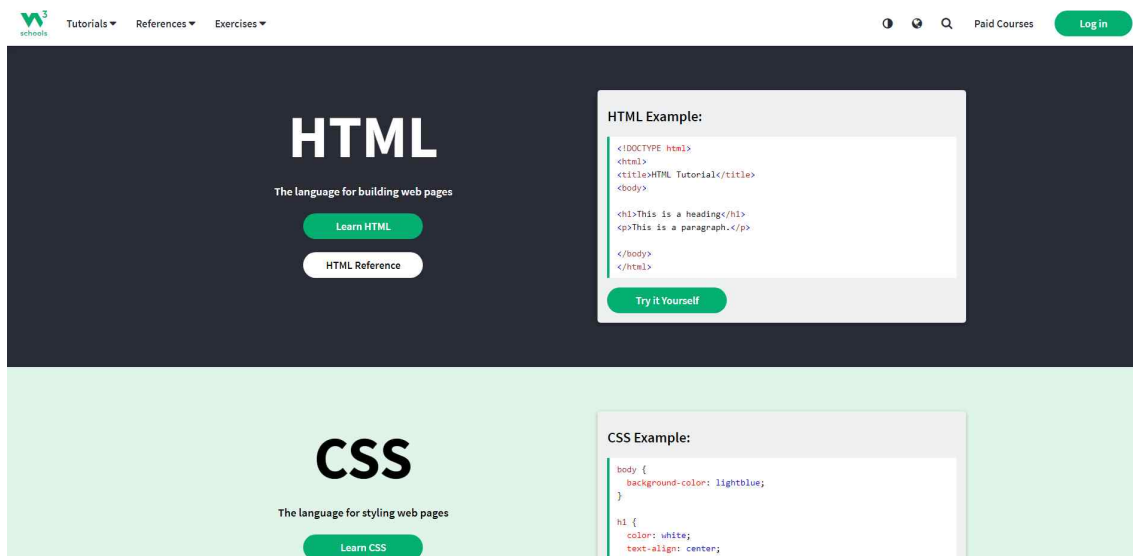
CSS 프로퍼티는 몇개나 될까? 250개 정도 된다고 한다. 이 강의를 모든 프로퍼티 다루는 데 쓴다면, 그것은 국어사전을 만드는 일과 같다. 아무도 국어사전을 공부하진 않는다.

지금까지 설명한 프로퍼티는, 모르면 웹개발할때 매우 곤란한 필수 프로퍼티들이다. 아마 여러분 머릿속에 남아있는 게 거의 없을 것이라 생각한다. 일주일 지나면 전부 까먹을 지식들이다. 이 수업은, CSS를 장난감 다루듯이 막 가지고 놀다가 왠지 작동이 안 될때마다 다시 들으러오길 추천한다.

이 밖에 프로퍼티는 구글 검색을 통해 상황에 맞게 사용하자.

CSS 예제를 찾아보는 최고의 웹사이트 하나만 소개하고 끝내겠다.

W3Schools      <https://www.w3schools.com/>



HTML, CSS, JavaScript 의 훌륭한 자습서 사이트이다.

다음은 Bootstrap 수업이다. 그러나 Bootstrap을 세세하게 다루진 않는다. 정확하게는, 구글링 하는 법과 공식문서 읽는 법에 대한 수업이다.