**초보자를 위한 HTML & CSS 동작과 원리**

**\* HTML의 기본 형태**

html

head

body

body 는 실제로 웹 페이지에 보여주는 데이터를 표현하는 부분

head 는 화면에 보여지기 위한 추가 정보(메타 정보)를 명시하는 부분

**\* 태그 속성과 값**

<a href=<http://www.naver.com>>네이버로 갑니다</a>

a 는 태그 명 의미

href 는 태그의 속성 명 의미

http... 는 태그 속성의 값을 의미

네이버.. 는 실제 태그가 적용될 텍스트를 의미

**\* HTML 태그**

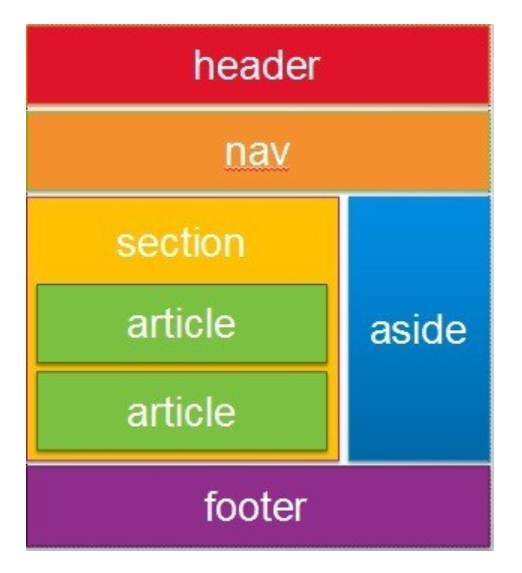
- HTML은 웹 페이지를 표현하는 문서로 적절한 태그를 통해 의미 전달을 잘 할 수 있어야 한다. - HTML5에선 기존 HTML에 있던 font 태그를 더 이상 지원 안한다. 그 이유는 웹 페이지의 스타일은 HTML 영역이 아닌 CSS의 영역이기 때문이다.

- HTML 태그 정보 ref : https://www.w3schools.com/tags/ref\_byfunc.asp

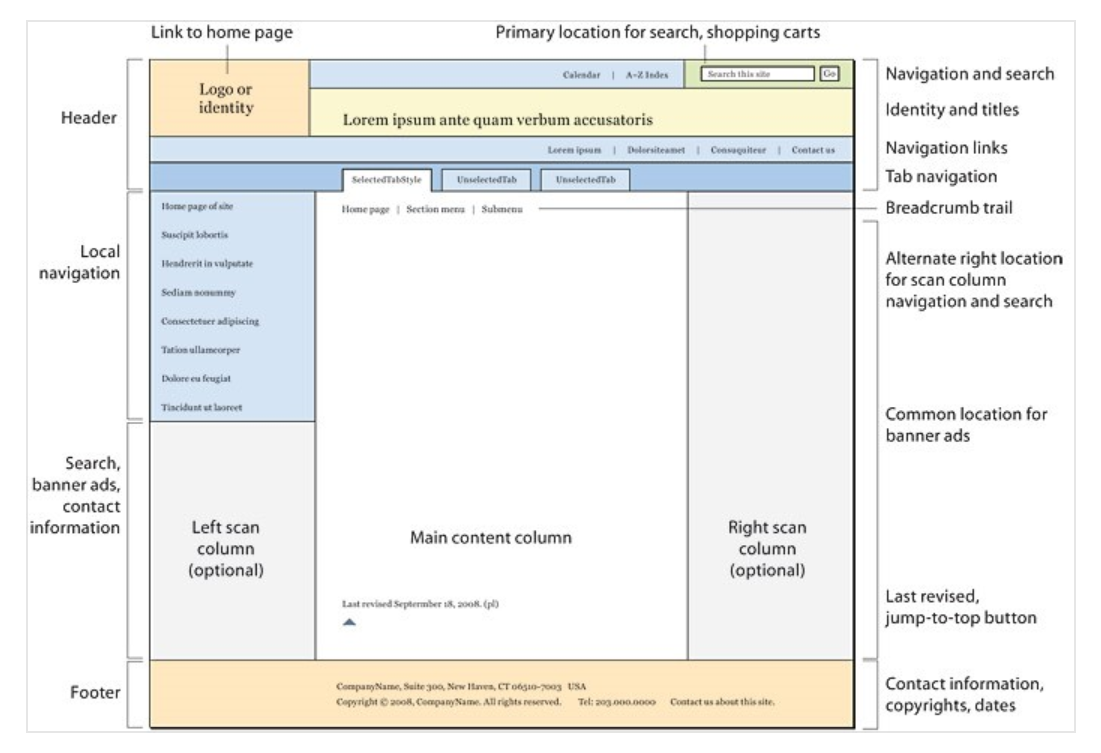
**\* 레이아웃을 위한 HTML 태그**

- 최신 브라우저가 대부분 설치된 모바일 장치에선 HTML5에서 제공하는 header, footer, aside와 같은 태그를 일반적으로 사용한다.

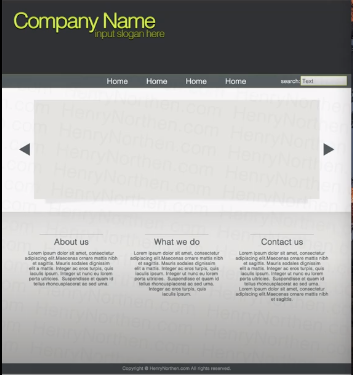
- 일반적으로 PC 환경에선 브라우저 버전이 다양해 HTML5의 최신 태그를 사용하는 대신, div 태그를 사용하고, header, footer, aside 같은 효과를 주기위해 추가 라이브러리를 사용한다.



**\* HTML 구조화 설계**



위의 이미지와 같이 웹 페이지의 구조를 설계하는 것을 HTML 구조화 설계(HTML Structure Design) 라고 한다.



예를 들어 위와 같은 홈페이지의 구조를 설계 한다면?? 아래와 같은 방식으로 설계할 수 있을 것이다. 설계 단계이기 때문에 여러가지 스타일(CSS)은 적용을 안한다.

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <meta name="viewport" content="width=device-width">  <title>JS Bin</title>  </head>  <body>  <header>  <h1>Company Name</h1>  </header>  <section>  <nav><ul>  <li>Home</li>  <li>Home</li>  <li>Home</li>  <li>Home</li>  </ul></nav>  <section>  <button></button>  <!-- Rolling 되는 이미지가 4개라고 가정 한다면.. -->  <div><img src=""></div>  <div><img src=""></div>  <div><img src=""></div>  <div><img src=""></div>  <button></button>  </section>  <section>  <ul>  <li>  <h3>About Us</h3>  <div>Description...</div>  </li>  <li></li>  <li></li>  <li></li>  </ul>  </section>  </section>  <footer><span>Copyright ...</span></footer>  </body>  </html> |

**\* HTML의 속성 ID와 Class**

ID 고유한 속성으로 한 HTML 문서에서 중복 사용 불가

고유한 ID 값이 있으면 하나하나에 특별한 제어 및 검색에 용이하다.

Class 하나의 HTML 문서 안에서 중복 사용 가능

홈페이지 전체적인 스타일을 일관성 있게 지정하기 위해 사용한다.

**\* 개발자 도구(Elements tab)을 활용한 HTML CSS 디버깅**

- 크롬의 개발자 도구를 활용한다(F12 단축키)

- 오픈되는 창의 Elements tab으로 이동 후 좌측의 인스펙션 버튼을 누르면 각 페이지의 부분에 마우스를 올리면 HTML 소스가 이동된다.

- 실제 소스의 속성 값을 바꿔서 페이지 모습을 바꿀 수도 있다.

**\* CSS란?**

CSS는 페이지의 스타일(글꼴, 글자색, 크기 등등)을 정의하는 스타일시트 언어이다.

**\* CSS의 구조**

span {

color : red;

}

div > p {

font-size : 20px;

}

위의 예제를 분석 해보면 아래와 같이 할 수 있다.

selector(선택자) 위 예제에선 span과 div > p (div 태그 하위의 p 태그)가 된다.

property(속성) 위 예제에선 color와 font-size가 된다.

value(값) 위 예제에선 red, 20px가 된다.

**\* HTML 페이지에 CSS를 적용하는 방법**

inline 실제 사용 태그 안에 직접 작성하는 방법

예를 들면 다음과 같다. <span style=”color:red”> Hello </span>

internal 페이지 HEAD 태그 안에 작성하는 방법

예를 들면, <head> <style> span { color : red; } </style> </head>

별도 CSS 파일을 관리할 필요가 없음

서버로부터 CSS 파일을 받기 위해 브라우저가 추가 요청을 보낼 필요가 없음

external 별도의 \*.css 파일을 만들고 HTML 페이지에서 이 파일을 include 한다.

예를 들면,

<head>

<link rel=”stylesheet” type=”text/css” href=”main.css”/> </head>

와 같이 표현이 가능하다.

우선순위는 inline > internal > external이 된다.

**\* CSS에서의 상속 개념**

상위 태그에 적용된 스타일은 하위 태그에도 적용 된다는 개념. 하지만 padding, bordeer와 같이 배치에 관련된 스타일은 하위에 적용되지 않는다.

<div style=”color:red”>

Hello, <p> World!!!</p> <!-- p 태그도 마찬가지로 color 스타일이 적용 -->

</div>

**\* 캐스케이딩 - Computed Style이 결정되는 방식**

CSS는 여러가지 스타일 정보를 기반으로 최종적으로 ‘경쟁’에 의해서 적절한 스타일이 반영된다. 여기서 경쟁이란 하나의 노드(태그)에 복수 개의 동일한 속성, 다른 값을 적용 하려는 경우에 내부적으로 어떤 스타일을 브라우저가 반영 할지 결정하는 것을 의미한다.

선언 방식 inline > internal > external 방식으로 우선 순위 처리

선언 순서 같은 방식으로 선언할 경우 가장 마지막에 선언한 스타일을 처리

선언 레벨 선택자가 구체적인 경우 우선 순위 처리

예를 들어 span 보다는 div > span 이 더 구체적이므로 우선 처리

선택 방식 아이디를 통한 선택이 클래스를 통한 선택보다 우선

예를 들어

<div id=”a” class=”b”> ... </div> 에 대하여

#a { color : red; } 와 .b { color : blue; } 가 있으면 red 적용

**\* Selector를 활용한 DOM 탐색**

태그로 지정 span { color : red; } 와 같이 태그 명을 명시

ID로 지정 # + ID를 명시 e.g. #spanTag { color : red; }

Class로 지정 . + Class를 명시

e.g. div 밑의 spanTag 클래스 명시 시, div.spanTag { color : red; }

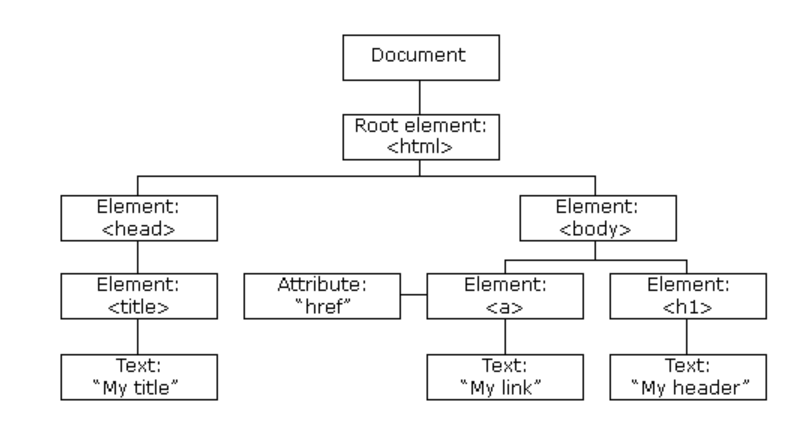
그룹 선택 h1, span, div { color : red; }

하위 요소 공백을 통한 방법 : div span { color : red; }

화살표를 통한 방법 : div > span { color : red; }

N 번째 자식 #divTag > p:nth-child(2) { color : red; } 처럼 nth-child(N) 사용

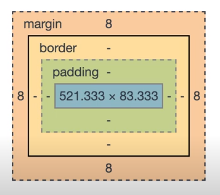
**\* HTML의 DOM 이란?**



웹 페이지가 로딩될 때 브라우저는 위와 같이 페이지를 파싱하여 DOM 이란 오브젝트를 생성한다. 위의 생성된 DOM을 통해 Javascript, CSS는 요소 추가 및 스타일을 적용한다.

**\* BOX 모델을 이용한 엘리먼트 배치**

HTML의 태그 엘리먼트는 사각형 박스 형태를 가지며 다음과 같은 속성을 가진다.



Margin 엘리먼트 간의 간격

인접한 두 개 Block 엘리먼트가 서로 다른 Margin을 가진 경우 더 큰 값이 적용

인접한 두 개 inline 엘리먼트의 경우엔 Margin의 합이 적용

Border 태그 엘리먼트 사각형 박스의 테두리 두께

Padding 태그 테두리와 태그 하위 자식 엘리먼트 사이의 간격

**\* Position 속성을 이용한 Layout**

Position 속성은 HTML 엘리먼트의 위치를 조절하는 속성이다.

기본 값으로 static이 지정된다.

static 기본 값

relative 속성 top, left를 기준으로 이동한다. 예를 들어,

div {

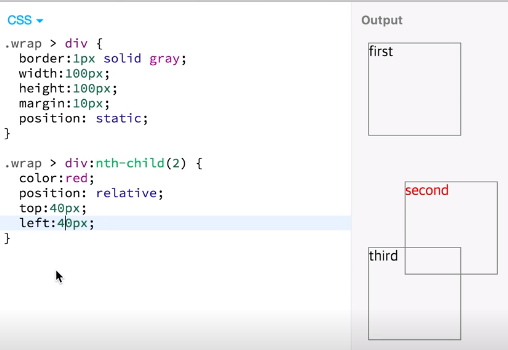
position : relative;

top : 40px;

left : 40px;

}

와 같이 주면 기본 위치 기준으로 40px 아래로, 40px 오른쪽으로 이동한다.



absolute div {

position : absolute;

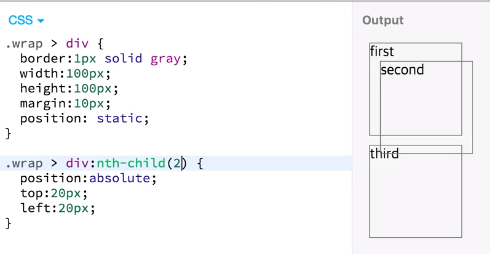
...

}

위와 같이 주면 해당 엘리먼트는 페이지를 그릴 때 무시하게 된다.

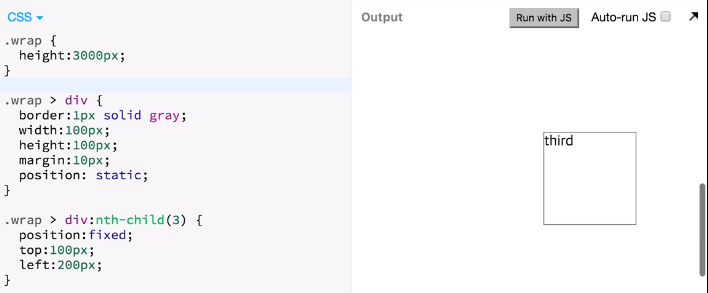
즉, 다른 엘리먼트들이 이 엘리먼트는 없다고 생각하고 그려지게 된다.

속성이 absolute이면 position이 static이 아닌 엘리먼트를 기준으로 그린다.



fixed 스크롤이 생겨 스크롤을 내려도 fixed는 고정이 되었기 때문에 사라지지 않음.

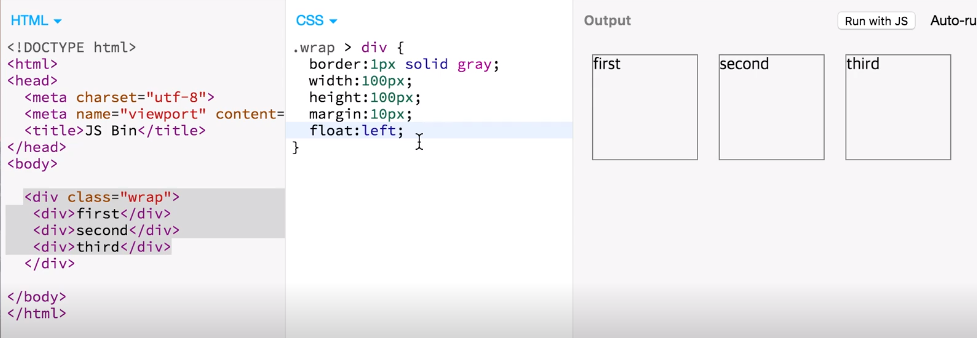
그대로 따라 내려간다. 나머지는 absolute와 비슷하다.



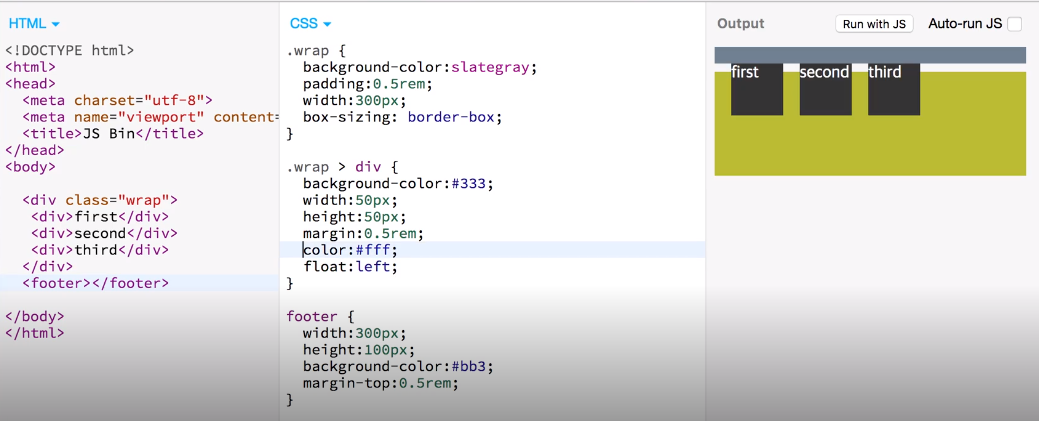
**\* Float 기반 Layout**

Float 속성을 주면 해당 엘리먼트는 위에 떠 있는 엘리먼트 효과가 나게 된다. 때문에 그 하위 엘리먼트가 속성을 가진 엘리먼트가 없다고 판단하여 겹쳐지게 된다.

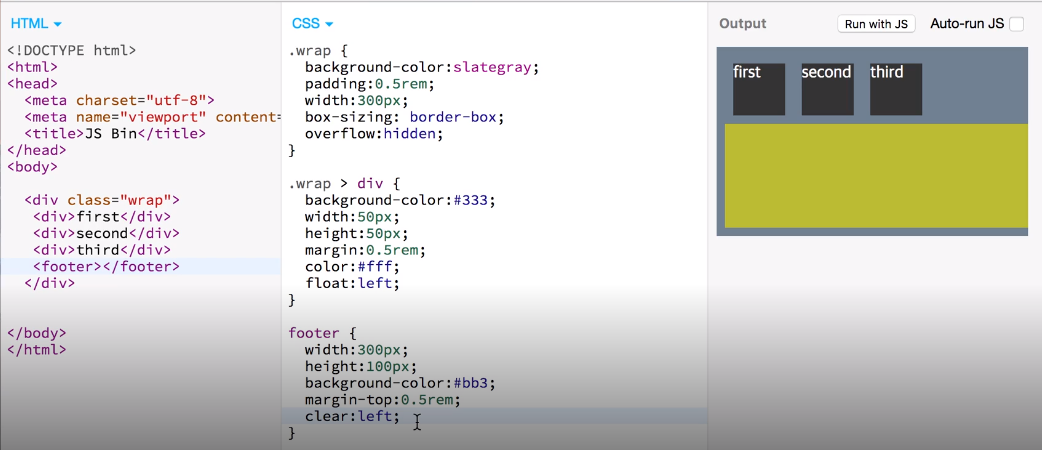
Float 끼리는 겹치지 않는다.



부모 엘리먼트는 그려질 때 자식 엘리먼트의 속성이 Float이면 넓이, 높이를 조절할 때 해당 자식 엘리먼트는 없다고 인식한 채 그려지게 된다. 아래 그림을 보면 wrap의 자식이 모두 float이기 때문에 wrap의 높이가 굉장히 작게 그려지고, footer 또한 위로 올라온 것을 볼 수 있다. 이 문제를 해결하기 위해선 wrap의 속성으로 overflow:auto 또는 overflow:hidden 이라는 속성을 추가 해야 한다.



또한 Float 속성을 가진 엘리먼트의 형제 엘리먼트는 Float인 엘리먼트를 무시한채 Layout 된다고 했다. 하지만 무시하지 않겠다는 의미로 clear 속성을 줄 수 있다. clear : left; 와 같이 속성을 주면 float:left;인 엘리먼트를 무시하지 않겠다는 의미가 된다.

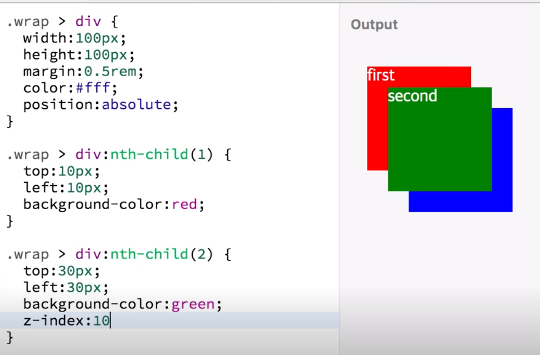


**\* Flex 기반 Layout**

데스크톱 환경에선 브라우저 버전, 호환성 등의 문제로 사용하기 어렵지만, 모바일 환경에서의 최신 브라우저라면 사용 해보는 것도 좋다. 반응형 웹(다양한 해상도, 크기의 디바이스에서 레이아웃을 자동으로 잡아주는 웹)에서 많이 사용한다.

**\* Z-Index 속성의 이해**

포지션 속성 값이 static이 아닌 경우 겹칠 수 있는데, 나중에 선언된 엘리먼트가 가장 위에 나오게 된다. 하지만 이 순서를 바꾸기 위해 Z-Index를 사용할 수 있다. Z-Index의 기본 값은 0이고 더 큰 숫자의 엘리먼트가 위에 위치하게 된다.



Z-Index를 통한 쌓임은 부모 엘리먼트 내에서만 의미를 가진다. 아무리 자식 엘리먼트의 Z 값이 커도 부모의 Z 값이 작으면 부모 레벨 엘리먼트에 의해 가려질 수 있다.

**\* Less Syntax**

CSS는 프로그램 언어라고 하기엔 너무나 불편한 점이 많다. 변수 선언, 재사용 등등이 굉장히 어렵기 때문이다. 이러한 불편함을 해소하기 위해 Less 라는 개념이 등장한다. 단, LESS는 CSS에서 공식적으로 지원 안하므로 LESS를 CSS로 변경 후 배포해야 한다...

변수 선언 @base\_width : 400px;

변수 사용 div { width : @base\_width; }

mixin footer { .div; } 와 같이 정의하면 div의 스타일을 재사용 한다

재사용 하는 .div의 속성 값을 덮어 쓸 수도 있다

Nested Rule 엘리먼트의 선언이 분산되는 것을 막기 위해 아래와 같이 LESS가 지원

.content {

width:@base\_width

li { <!-- .content > li 를 의미한다 -->

list-style:none;

}

}

**\* Transition을 이용한 CSS 기본 애니메이션**

예를 들어 javascript를 통해 특정 엘리먼트 클릭 시 높이를 200px로 변경하게 했을 때 그 에니메이션이 동작하는 시간을 조절하는 속성이다.

아래와 같이 선언할 수 있다.

.box {

<!-- 높이 및 넓이가 변경될 때 2초가 걸리게 한다. 또는 all 2s 를 해도 된다 -->

transition: height 2s, width 2s;

}

그 외 가속도 등등 다양한 속성이 존재한다.

**\* Transform 속성 활용**

transform은 좌표공간을 변형함으로써, 다른 엘리먼트에 영향을 미치지 않고 특정 엘리먼트의 위치를 바꿀 수 있다. 그 중 translate 값을 주는 경우 GPU 연산을 통해 빠르게 브라우저 랜더링을 처리할 수 있는 속성이다. 때문에 position이나 float를 사용하는 것보다 더 빠르게 랜더링이 가능하다.

