HTML, CSS, JS

Hyper Text Markup Language = HTML

- Hyper Text : 하이퍼링크를 통해 순서대로 페이지를 보는게 아니라 원하는 페이지로 바로바로 갈 수 있는 것
- Markup Language : 태그(마크, 표시)를 이용하여 문서나 데이터의 구조를 명시하는 언어

즉 HTML은 웹 페이지의 구조 혹은 데이터 작성을 위한 마크업 언어이다.

태그라는 것을 통해 여기는 제목이구요, 여기는 내용이구요~이런 것들을 명시하는 것

Cascading Style Sheets = CSS

- Style Sheets : 웹페이지의 스타일과 관련된 모든 것을 정해둔 문서
- Cascading : 위에서 아래로. 스타일들이 우선순위로 적용된다.

즉 CSS는 웹 페이지에 관한 다양한 스타일을 정의할 때 쓴다.

제목은 여기 배치하구요, 여백은 어느정도로 하구요, 색깔은 어떻게 하구요~~등의 스타일을 정한다.

Javascript : 웹을 이용하는 유저와 상호작용하기위한 기능을 추가할 때 쓰는 언어

즉

- HTML은 웹페이지의 구조를

- CSS는 웹 페이지의 스타일을
- JS는 웹 페이지의 동작을

담당한다.

About HTML

HTML의 기초문법

태그를 통해 여기는 제목이구요 ~ 여기는 내용이구요 등등을 나타낸다.

이 때,

제목 : 좀있으면알바감

이 아니라

`<제목>좀있으면 알바감</제목>`

으로 태그로 감싸서 내용을 나타낸다. 좌측 태그를 시작태그, 태그로 감싸진 것을 내용(contents), 우측 태그를 종료태그라 하며 이 묶음을 HTML 요소 (Elements)라고 한다.

HTML 문서 구조

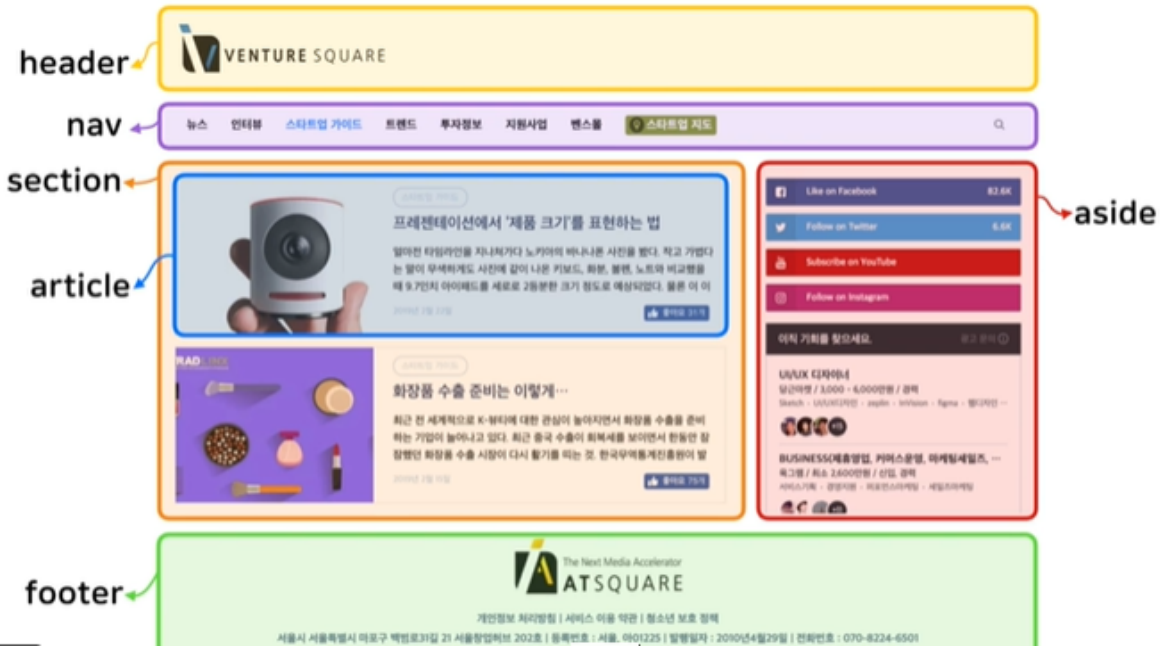
HTML문서는 정해진 '양식'이 있음. 다음과 같다.

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="utf-8">
        <title>문서이름같은거</title>
    </head>
    <body>

    </body>
</html>
```

- <!DOCTYPE html> : 문서 형식을 정의할 때 쓰임. 반드시 있어야 함. 이 문서는 html로 작성한거야 라고 브라우저에게 알려줌.
- <html lang="ko"> : 전체 html을 감쌈, 이 태그 밖에 다른 태그는 존재할 수 없으며 전체 문서에서 한 번만 쓰임. 이 페이지의 주 언어가 korean임을 lang을 통해 명시. 즉 이 태그는 본격적인 태그의 시작이며 사용하는 주 언어를 정의함
- <head> : html문서에 대한 정보를 담는다. 문서 내에서 단 하나 존재하며 위치는 html 태그 바로 아래.
- <meta charset="utf-8"> : 문서에 관련된 정보를 담는태그.
- <title> : 웹페이지의 제목을 담는 태그
- <body> : html에서 실질적으로 보여지는 부분. 문서에 하나 존재하며 위치는 head태그 아래.

레이아웃과 관련된 기본 태그

시멘틱 태그 : 의미를 가지고 있는 태그　ex) nav, section, header, footer....



- <header> : 제목이나 소개 등을 담기 위해 쓰는 태그
- <nav> : 내비게이션 역할, 페이지 이동을 위한 메뉴를 주로 담고 있음
- <section> : 기준에 따라 구간을 구분하기 위해 사용
- <article> : 주 내용을 담기 위한 태그
- <aside> : 광고나 사이트의 주변 부분에 해당하는 내용을 담는 태그
- <footer> : 일반적으로 웹사이트의 아래에 사이트의 추가정보를 담음

이 때, '배치'는 따로 CSS를 통해 구현한다. html만 사용하면 다음과 같은 모습임

텍스트와 관련된 태그

이에 해당하는 대표적인 태그는 제목 태그, 본문 태그, 글자와 관련된 태그들이 있다. 물론 태그로 감싸지 않아도 <body>태그안에 써둔 텍스트는 잘 보이긴 하지만, 나중에 CSS로 스타일을 먹일 때 굉장히 불편하므로 특별한 경우가 아니라면 태그로 감싸자.

- 제목 태그 : 제목을 나타내고 싶을 때 사용, 1~6까지 중요도에 따라 쓴다. <h1> ~ <h6>까지
- 본문 태그 : 주로 쓰는 건 <p>,
, <pre>태그.
    1. <p> : paragraph, 단락, 문단을 나타낸다. 즉 얘로 감싸진 내용이 한 문단이 되는 거임
    2.
 : 줄바꿈을 해줌. Enter의 기능. 얘는 종료태그(즉 </br>)이 없음. 이렇게 종료태그가 없는 녀석들을 Empty Element라고 부름
    3. <pre> : preformatted, 형식화된. <p>태그의 경우 감싼 텍스트 중에서 Enter를 통해 줄바꿈을 해도 웹페이지에는 줄바꿈이 안 된채로 나오지만, 얘는 텍스트가 작성된 그대로 웹페이지에 출력됨. 이게 무슨 말이냐고?

        ```
        <p>그니까 내가 여기서 엔터를 해볼꺼야
            근데 문서상에는 한 줄로 나온다니까?</p>
        ```

        이건 웹페이지에서 줄바꿈없이 나오지만, <pre>태그로 감싸면 내가 입력한 그대로 나온다는 말입니다. 즉 적은 그대로 출력함.

- 글자와 관련된 태그 :
    1. 태그 : 태그로 감싼 단어 or 문장에 Bold체를 먹임
    2. 태그 : 태그로 감싼 단어 or 문장을 이탤릭체로 바꿈
    3. <sub>태그 : 태그로 감싼 단어 or 문장을 조금 아래로 내림. 로그의 밑 등을 표현할 때 유용
    4. <sup>태그 : 태그로 감싼 단어 or 문장을 조금 위로 올림. 수학의 지수 등을 표현할 때 유용
    5. <ins>태그 : 태그로 감싼 단어 or 문장에 밑줄을 추가
    6. <del>태그 : 태그로 감싼 단어 or 문장에 취소선을 추가el>태그 : 태그로 감싼 단어 or 문장에 취소선을 추가

링크 태그

<a>태그가 링크태그의 역할을 함. 다른 태그와는 다르게 필수로 들어가는 속성이 있음.

```
※ 속성이란?
: 시작 태그의 괄호 사이에 들어가며, 태그 이름 바로 뒤에 한 칸 띄우고 작성함.
태그에 관한 추가적인 정보를 제공. html의 모든 태그는 속성을 가질 수 있음!

※ 속성의 형태
키 = "값"
반드시 값을 쌍따옴표 or 따옴표로 감쌀 것
```

a태그는 링크주소를 속성으로 줘야 한다.

```
<a href="www.google.co.kr">구글</a>
```

태그가 여러 속성이 있을 땐 띄어쓰기로 구분함. 속성의 값을 따옴표나 큰따옴표로 감싸지 않았다면 띄어쓰기로 구분을 할 때 문제가 발생할 수 있으니, 큰따옴표나 따옴표로 값을 감쌀 것

예를 들어,

```
<a 집=경기도 용인 이름=수노>수노</a>
```

이렇게하면 나는 집이라는 속성에 값으로 "경기도 용인"을 주고 싶었지만 실제론 경기도까지만을 값으로 인식하고 용인은 키로 인식할 수 있다는 것. 따라서 경기도 용인을 큰따옴표로 감싸란 말임

절대url vs상대url

- URL : 인터넷에서 HTML페이지, CSS문서, 이미지 등 자원의 위치를 나타냄
- 절대URL : 접근하는 최초 시작점부터 경유한 경로를 모두 기록하여 자원의 위치를 나타냄
- 상대URL : 기준점을 기준으로 상대적인 경로를 기록하여 자원의 위치를 나타냄

ex)

target 속성

링크를 클릭했을 때 해당 페이지를 어디에 열지 정하는 속성(새 창에서 열기, 새 탭에서 열기 등)

대표적으론

```
target = "_self"              -> 현재 탭에서 링크를 여는 속성
target = "_blank"             -> 새 탭(창)에서 링크를 여는 속성
```

멀티미티어와 관련된 태그

- 이미지 태그 : <img src = "이미지url" alt="사진 설명"> 종료태그없음. src속성은 기본으로 가짐. alt속성은 불러올 이미지가 없거나 불러오는데 실패했을 경우 대신 표시되는 문장. 이 외에도 height, width속성 등을 통해 높이같은 걸 조정할 수 있으나 이는 CSS로 조정하기를 권장

테이블

<table>태그 : 일정한 형식과 태그가 있으므로 이를 잘 활용해서 작성해야 한다.

일반적인 표의 구성은 다음과 같다.

이 전체를 HTML 코드로 바꾸면?

정리하자면,

- <table> : 표 전체를 감싸는 태그
- <tr> : 표에서 행을 구분하는 태그
- <th> : 표의 행 내부에 제목 셀을 담는 태그
- <td> : 표의 행 내부에 데이터 셀을 담는 태그

ex)

```
<table>
    <tr>
        <th>이름</th>
        <th>나이</th>
    </tr>

    <tr>
        <td>조상현</td>
        <td>24</td>
    </tr>

    <tr>
        <td>우태윤</td>
        <td>25</td>
    </tr>
</table>
```

표

성별	남	여
학년	3	
이름	수노	곽두팔

이렇게 3이 있는 셀을 합친 것처럼 셀을 합쳐야 할 땐 어떻게 할까?

속성을 이용한다.

- rowspan = "숫자" : "숫자"만큼 셀이 행을 점유
- colspan = "숫자" : "숫자"만큼 셀이 열을 점유

리스트

장보기 목록같은거 생각하면 됨. 두가지로 분류할 수 있다.

1. 순서없는 목록(Unordered List)
2. 순서있는 목록(Ordered List)

1의 경우

```
<ul>
  <li>아이템1</li>
  <li>아이템2</li>
  <li>아이템3</li>
</ul>
```

2의 경우

```
<ol>
  <li>아이템1</li>
  <li>아이템2</li>
  <li>아이템3</li>
</ol>
```

목록과 관련있는 속성

모두 순서있는 리스트와 관련되어있다.

- 태그에 가능한 속성
    1. start="숫자" : 리스트가 시작하는 숫자를 정함
    2. type="문자" : 순서를 시작하는 문자를 정함(a부터 시작한다 등)
    3. reversed : 순서를 반대로 시작. 다른 속성과 달리 키만 써서 사용
- 태그에 가능한 속성
    1. value="숫지" : 해당하는 리스트 아이템의 번호를 임의로 지정

form 태그

: 폼에 포함되는 다양한 입력 양식 태그들을 감싸줌. 예를 들어 지원서를 작성할 때, 모든 내용이 담겨있는 종이라고 생각하면 됨. form태그는 다양한 속성들을 가지며, action속성과 method속성은 필수적으로 사용해야 함.

- action : 데이터를 보낼 URL을 지정
- method : 보내는 방식을 지정("get" or "post"가 값으로 들어감)

이 때 method속성의 값으로 오는 get과 post는 둘다 우리가 폼에 입력한 데이터를 서버에 보냄. 그러나 get은 데이터를 URL 끝에 붙여 눈에 보이게 보내고(네이버 검색창에 어떤 단어를 검색하면, 내가 검색한 단어가 주소창의 맨 끝에 있다), post는 데이터를 URL에 적지 않고 내부에 숨겨서 보낸다. get은 데이터 조회만을 목적으로 할 때 주로 쓰고, post는 서버에 있는 데이터를 쓰거나 수정, 삭제할 떄 주로 사용하는 것과 연관이 있다.

input 태그

form 태그 중에서 가장 많이 쓰이는 태그. 사용자에게 입력을 받기 위해 사용된다. 종류가 다양한데, 이는 type속성에 의해 결정된다. name속성도 가질 수 있으며, 서버로 데이터를 전송할 때 입력받은 데이터들을 name속성을 통해 구분한다. input태그를 통해 입력받은 녀석이 'name속성의 값'을 꼬리표로 달아서 전송되는 것이라 생각하면 됨.

placeholder속성은 input에 아무 것도 입력되지 않았을 때 값으로 적어줬던 녀석이 나타난다.

사실 우리가 input에 값을 입력하면 value라는 속성에 그 값이 저장된다고 생각할 수 있다. 그러나 input태그에 value속성을 직접 적어서 value="Sanghyun"처럼 한다면, 이 속성값(Sanghyun)을 초기값처럼 둘 수 있게 된다. 그냥 밑에 거 참고하셈ㅋㅋ

```
<input> 태그의 value 속성은 <input> 요소의 초깃값(value)을 명시합니다.

value 속성은 <input> 요소의 type 속성값에 따라 다른 용도로 사용됩니다.

- “button”, “reset”, “submit” : 버튼 내의 텍스트를 정의함.
- “hidden”, “password”, “text” : 입력 필드의 초깃값을 정의함.
- “checkbox”, “image”, “radio” : 해당 입력 필드를 선택 시 서버로 제출되는 값을 정의함.

또한, <input> 요소의 type 속성값이 “file”인 경우에는 value 속성을 사용할 수 없습니다.
```

label 태그

: for속성과 함께 사용되며, input태그의 id속성의 값이 for속성의 값으로 들어감(input태그가 아니라 select태그 등에도 사용가능). input태그의 입력칸에 꼬리표를 붙여주는 느낌의 태그. 이 태그로 감싼 내용물을 클릭하면 얘랑 연결된 input태그에 바로 값들을 입력할 수 있게 활성화됨. 이렇게 클릭해야만 입력가능한 건 아니고, 직접 input칸을 눌러서 입력할 수도 있음

ex)

```
        <label for="userid">아이디 : </label>
        <input type="text" id="userid" name="ID" placeholder="ID">
```

div 태그

: 태그들을 구분짓고 나누기위해 사용되는 태그

select 태그

: 여러 개의 선택지를 제시하고 싶을 때 쓰는 태그. 각 선택지를 감싸는 <select>태그와 선택지 하나하나를 감싸는 <option>태그로 이루어짐

ex)

이 녀석 역시 name속성을 가질 수 있으며, input태그와 동일하게 이 name속성의 값은 사용자가 선택한 옵션에 꼬리표를 달아 전송하는 역할. 또한 각각의 <option>태그들 역시 value속성을 가질 수 있으며, 이 value값이 <select>태그의 name속성과 매칭된다.

textarea 태그

: 한번에 많은 글을 입력받을때 사용한다.

button태그

: input태그의 type값으로 button을 준 것과 동일하게 버튼을 생성한다. button태그는 html요소를 button태그 내부에 담을 수 있어 유용하다. 그 예는 다음과 같다.

```
<button type="submit">
    <img src="dog.png"
</button>
```

CSS

선택자를 통해 선택한 html 요소에 스타일을 먹임

HTML에 CSS를 적용시키는 방법

1. Link style
2. Embedding style
3. Inline style

1. Link Style

: HTML외부의 CSS파일을 불러온다. 가장 일반적인 방법.

어떻게? : <head>태그 안에서 다음과 같이 <link>태그를 사용

```
<head>
    <meta charset="UTF-8">
    <title>무야호</title>
    <link rel ="stylesheet" href ="css파일이름.css">
</head>
```

이 때 rel속성은 relation의 약자로써, 현재 문서와 연결문서 사이의 관계를 나타낸다. href는 연결할 문서의 URL.을 적어준다.

2. Embedding style

: HTML의 <head>태그에 <style>태그를 작성하여 이 내부에 CSS를 작성

ex)

```
    <head>
        <meta charset="UTF-8">
        <title>무야호</title>
        <style>
            p{
              color: red;
              background-color: green;
            }
        </style>
    </head>
```

3. Inline style

: 스타일을 먹이고 싶은 HTML요소에 직접적으로 style속성을 작성하는 방법

ex)

```
<body>
  <h1 style="color: green; background-color: gray">안녕하세욧!</h1>
</body>
```

당연한 소리지만 이렇게 작성한 css는 해당 요소에만 적용됨! 다른 요소들엔 적용 X

선택자

"이 문단은 폰트는 맑은 고딕으로, 글씨크기는 18로, 색깔은 파란색으로 해주세요."

라는 문장이 있을 때 "이 문단은"이 선택자에 해당하는 개념이다. 풀어서 말하면, 스타일을 적용하고자 하는 HTML요소를 선택하는 역할을 해준다.

선택자는 기본적으로 h1, p, span과 같은 태그들을 사용할 수 있다. 근데 이런 식으로 h1태그를 선택했다면, 문서 내의 모든 h1태그에 스타일이 적용된다. 또한, 콤마(,)를 이용해 동시에 여러 선택자를 고를 수 있다.

ex)

```
h1, p, span{
    color: red;
}
```

또한, 다음 방법을 통해 "as라는 클래스를 가진 요소의 자식 요소들 중 a태그"를 선택할 수 있다. 하위선택자 라고 부르는 듯?

```
.as a{
    color: red;
}
```

단순 선택자

1. 타입 선택자

해당 태그를 갖는 모든 요소에 스타일 적용. 방금 본 것과 같다.

ex)

```
p{
    color: red;
}
```

문서 내의 모든 <p>태그에 color: red가 적용됨

2. id선택자

id로 스타일을 적용. 해당 id값을 갖는 태그만 적용되는데, 문서 내에 같은 id를 갖는 태그는 존재할 수 없다. 즉 같은 id값을 갖는 태그가 중복되어선 안 됨. 딱 하나를 집어서 스타일 멕이기 좋음. 한놈만 팬다는 마인드. 타입선택자와는 다르게 #을 통해서 id선택자임을 쏼라쏼라

ex)

```
#jo{
    color: red;
}

이렇게하면 어딘가에

<p id="jo">앗!!!</p>
처럼 id값으로 jo를 갖는 요소에 스타일이 멕여짐
```

3. 클래스 선택자

class로 스타일을 적용. html요소 중 클래스 속성값을 가진 녀석들을 골라내어 멕인다. 얘는 .을 통해서 클래스 선택자임을 쏼라쏼라

ex)

```
.ss{
    color: red;
|

이렇게하면 어딘가에
```

```
<p class="ss">어머나</p>
<h2 class="ss">무야호</h2>

처럼 클래스 값으로 ss를 갖는 모든 요소에 스타일이 멕여짐
```

참고 : 한 요소에 띄어쓰기를 통해 여러 클래스를 적용할 수 있다.

```
<p class="human student">무아호</p>
```

human이라는 클래스와 student라는 클래스가 무야호라는 요소에 적용된다.

4. 전체 선택자

모든 요소에 스타일 적용, 따라서 속도 저하 가능성 있음, 쓰지 않기를 권장

ex)

```
*{
    color: red;
}

모든 요소에 스타일이 적용됨
```

5. 속성 선택자

특정 속성을 소유하는 모든 요소에 스타일을 적용

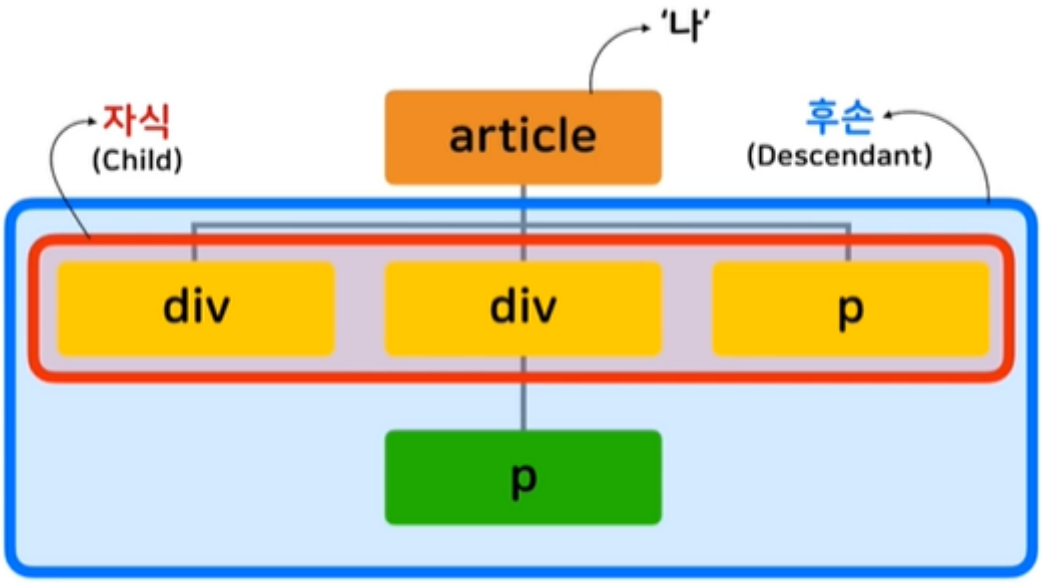
ex)

```
a[target="_blank"]{
  color: red;
}

이렇게하면 문서 내의 a태그 중 target="_blank"라는 속성을 갖는 녀석들에게 스타일이 적용됨.

당연한 얘기지만
```

```
.as[target="_blank"]{
    color: red;
}

이렇게하면 문서 내에서 as라는 클래스값을 가진 녀석 중 target="blank"라는 속성을 갖는
녀석들에게만 스타일이 적용됨!
```

복합선택자

우선 html요소들 간의 부모자식개념을 알아야 함. 직관적으로 이해하기 쉬움



'나'가 <article>태그라고 할 때, 어떤 걸 자식태그라 하고 어떤 걸 후손이라 하는지 딱 이해가 될거임. 참고로 이 상황은 코드로 보면

```
<article>
  <div></div>
  <div><p></p></div>
  <p></p>
</article>
```

의 상황임.

1. 자식 선택자

선택자 A의 모든 자식중 **선택자 B**와 일치하는 요소를 선택

ex)

```
선택자 A > 선택자 B{
    color: red;
}

예를 들면

article > p{
  color: red;
}
```

2. 후손 선택자

선택자 A의 모든 후손중 **선택자 B**와 일치하는 요소를 선택

ex)

```
선택자 A 선택자 B{
   color: red;
}

예를 들면

article p{
    color: red;
}
```

Pseudo 클래스

요소의 특별한 상태를 지정할 때 씀

다음과 같이 쓴다.

```
선택자:pseudo-class{
    속성: 속성값;
}
```

pseudo-class 즉 가상의 클래스라는 건데, 이 가상 클래스를 지정해서 스타일을 멕이는 거다. 그냥 찾아서 공부해보장

값, 단위, 색

숫자값과 백분율

CSS는 `속성: 값`이라는 한 짝으로 묶이는데 이 때 값으로 올 수 있는 것은

1. 숫자값
2. 키워드
3. 색

이 있으며, 속성에 따라 쓸 수 있는 값들이 정해져 있다. 이 중 `숫자값`은 단위가 중요한데, 대표적인 단위는

1. px (참고: 웹 브라우저는 기본적인 font-size가 16px)
2. em
3. rem

이 있다. 이 중 em과 rem은 px와 달리 상대적인 길이를 나타내는데, em은 현재 스타일이 지정된 요소의 font-size기준이고, rem은 최상위 요소의 font-size기준이다.

rem을 예로 들면, html문서에서 최상위 요소는 html태그이므로 이 태그에 지정된 스타일 중 font-size가 32px라면 1rem = 32px

아니 근데 그냥 px쓰지 왜 em, rem을 쓰는걸꽈?

우리가 쓰는 브라우저 환경이 기기(컴퓨터, 모바일 등)에 따라 다르므로, 반응형으로 만들기 위해 상대적인 개념인 em, rem을 사용함. 제일 권장하는 건 rem

또한 `%`라는 단위도 있는데, 이 녀석도 상대적인 길이로 보통 이미지나 레이아웃의 너비나 높이를 지정할 때 쓴다.

색상

`blue`, `erange`, `powderblue`와 같은 키워드로 색상을 표현가능하나 이는 한계가 있음. 키워드를 제외하고 색을 표현하는 3가지 방법이 있음

1. hex code : #dd3333 같은 걸 말하는 거임
2. RGB : rgb(0, 0, 0), rgb(255, 0, 0)등으로 색을 표현하는 걸 말함
3. hsl

또한 hex code와 RGB는 알파값이란 걸 추가함으로써 `투명도`를 조절할 수 있다.

hex code의 경우 뒤에 숫자를 추가하는 것으로, (ex. #00ff00 → #00ff0050)

RGB의 경우 a를 추가하여 rgba(red값, green값, blue값, a값)으로 표현하는 것으로 가능. a값으론 0~1까지의 숫자를 넣을 수 있음

텍스트 관련 속성

폰트와 관련된 속성

1. font-size
2. font-familty
3. font-style
4. font-weight

가 있다.

font-family

폰트의 종류를 정함. 한 단어로 된 폰트명은 따옴표 없이 사용가능, 띄어쓰기나 하이픈(-)이 포함된 이름은 따옴표로 감싸서 사용. 사용자의 컴퓨터엔 내가 쓰는 폰트가 없을 수도 있으므로, 여러 개를 지정하여 적용시킨다(따라서 맨 마지막에는 모든 os/브라우저에 있는 일반 글꼴을 둔다)

(ex: font-size: a, b, c;)

그러면 내 컴퓨터에 있는 폰트만 적용가능한가? No. 폰트파일을 함께 업로드해 경로를 설정하거나, 웹 폰트를 활용해 쓸 수도 있다. 이 중 웹 폰트는 링크를 통해 쉽게 폰트를 불러오는 방식이다. html파일에 링크를 통해 css파일을 연결하는 방식과 비슷하다.

font-style, font-weight

font-style로 가능한 것

1. normal : 기본 글자체
2. italic : 이탤릭체. 이탤릭체를 지원하는 폰트에서만 정상적으로 사용가능
3. oblique : 글자를 기울임

fint-weight : 폰트의 굵기를 조정, 속성값으로 bold 또는 100 ~ 900사이의 숫자를 줄 수 있음 bold는 700에 해당함. normal은 400에 해당

font-size, family, style, weight등을 하나하나 따로 적어줄 필요 없이

font속성을 사용해서 한 방에 적을 수도 있음. 이는 따로 css font property를 검색할 것

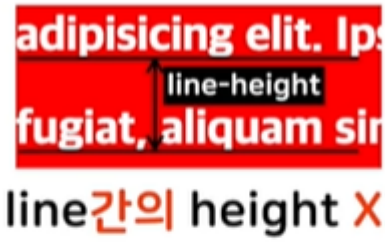
텍스트 정렬과 관련된 속성

text-align

텍스트를 좌,우,중앙 정렬. 값으론 left, center, right가 올 수 있음. 참고로 웹 페이지를 기준으로 정렬되는 게 아니라 적용되고 있는 본인 요소를 기준으로 정렬됨

line-height

문장 사이의 간격을 조정(줄간간격 조정). 값으로는 `px가 붙는 숫자` 또는 `그냥 숫자`가 올 수 있는데, 그냥 숫자가 오면 상대적인 간격을 의미하게 된다. 즉 이 경우 요소의 폰트 사이즈가 16px이고 line-height값으로 2가 오면 간격은 32px가 됨.



line간의 height X

line의 height O

letter-spacing

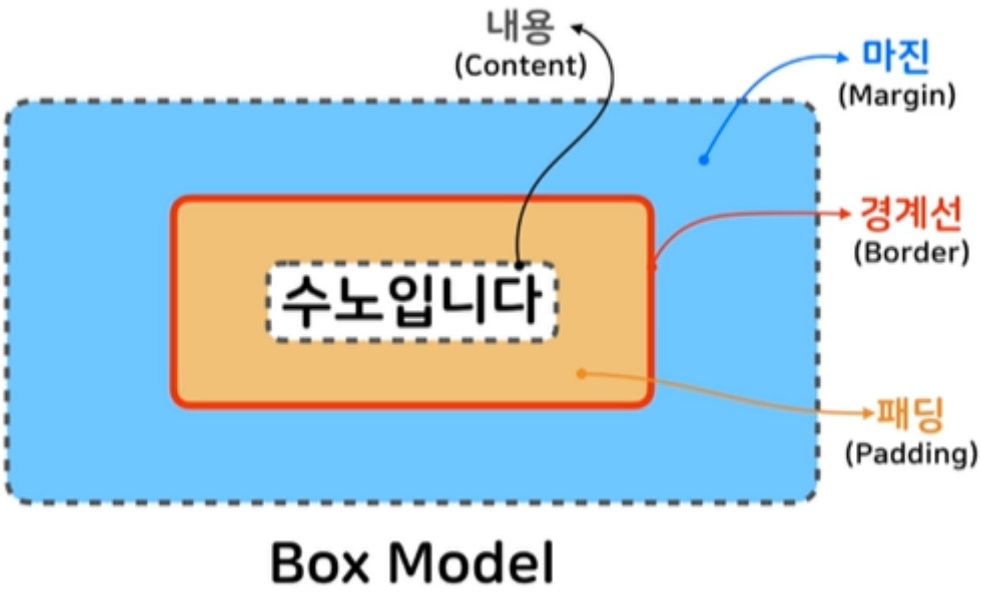
자간 즉 글자와 글자 사이의 간격을 의미.

text-indent

문단의 시작부에 이 속성의 값만큼의 들여쓰기를 함

박스 모델

HTML의 모든 요소는 `상자 형태`를 가진다.



- Content : 텍스트, 이미지같이 우리가 태그 사이에 담은 실제 내용물
- Border : Content를 감싸고 있는 테두리
- Padding : Content와 Border사이의 여백
- Margin : Border밖의 여백

박스 모델과 관련된 CSS 속성에 대해 알아보자.

우리가 보통 요소의 크기를 정의할 땐 width, height속성을 많이 쓴다. 이는 Content의 크기에 해당하는 속성이다.

border

보통 3가지의 속성을 사용한다.

1. border-style : 테두리 선의 형태 관련. 값으로 주는 개수에 따라 다르게 적용됨. 검색해서 알아볼것
2. border-width : 두께 관련

3. border-color : 색깔 관련

이 3가지 속성을 따로따로 할 필요없이, border라는 속성을 사용해 한 방에 멕일 수도 있음.

ex)

```
border: 4px solid red;
```

검색해서 알아볼 것.

추가 : border-radius속성 : 경계선을 둥글게 표현가능

padding & margin

상하로 붙어있는 요소들에 margin속성을 주면 CSS의 마진상쇄가 일어남. 큰 쪽 마진을 따라간다!

box-sizing

- box-sizing: content-box(기본으로 이렇게 돼있음)

→ width(height) = content size가 됨

- box-sizing: border-box;

→ width(height) = content size+ padding+border가 됨

위치와 관련된 속성1

display 속성

요소가 보여지는 방식을 지정하는 속성이다. 대부분의 HTML요소들은 display: block 또는 display: inline 이라는 속성값을 기본적으로 가진다.

- `display: block;`을 기본적으로 갖는 태그들 : 항상 새로운 줄에서 시작을 하며, 따로 너비를 지정하지 않아도 width: 100%를 기본으로 가진다. , <h1>~<h6>, <p>, <header>등이 있다.

> 💡 참고 : width: 100%는 부모 요소의 너비에 맞춰 늘어나는 것이다. %가 상대적인 개념이니까.

- `display: inline;` 을 기본적으로 갖는 태그들 : 새로운 줄에서 시작하지 않으며 필요한 만큼의 너비만 가짐 → 즉 요소의 content크기만큼의 너비만 가짐. <a>, , <image>등이 해당함

이 두 종류의 요소(block, inline)는 가질 수 있는 속성이 다르다. **block element**의 경우 width, height, margin, padding이 가능하지만 **inline element**의 경우 width, height, margin-top, margin-bottom이 불가능하다. 이런 불편함을 display: inline-block;을 통해서 해소가능하다. **inline-block**은 width, height, margin-top, margin-bottom이 가능하다.

display: none; 도 줄 수 있는데 이렇게하면 웹페이지에 해당 요소가 출력되지 않는다.

position 속성

요소의 위치를 정의하는 속성이다. 값으로는 static, relative, absoulute, fixed가 올 수 있다.

1. static : 기본값으로 설정되는 녀석으로, 얘 쓰면 좌표 속성을 사용할 수 없다

2. relative : 상대 위치를 뜻하며, 기본 위치(요소가 기본적으로 원래 있어야 할 위치)를 기준으로 좌표 속성을 사용할 수 있게 한다
3. absolute : 절대 위치를 말하며, 부모나 조상 중 relative, absolute, fixed가 선언된 곳을 기준으로 좌표 속성을 사용할 수 있게 한다. 부모나 조상 중 위 항목이 선언된 녀석이 없다면 최상단 태그인 body태그가 기준이 됨
4. fixed: 보이는 화면을 기준으로 좌표 속성을 이용해 위치를 고정할 수 있게 함. 스크롤을 움직일때마다 따라 움직이는 것들이 이걸 활용한 것임

좌표 속성?

left: 20px;

top: 20px;

등으로 위치를 조정할 수 있게 하는 속성을 말한다

z-index 속성 : 파워포인트나 포토샵같은 데서 그림들이 겹칠 때 우선순위를 정해서 어떤 게 제일 앞에 보이게 하는지와 같은 기능. 이 속성값으로 준 숫자가 높을수록 우선순위 up

위치와 관련된 속성2 - flexbox

기존에 display나 position등을 통한 웹의 레이아웃 개발은 생각할 부분이 많고 한계가 어느 정도 존재해 개발이 좀 어려웠다. 이를 개선할 수 있는 방법으로 CSS에서 flex box라는 것이 나왔다.

flex box

부모 요소인 flex container와 자식 요소인 flex item으로 구성된다. flex box를 사용하고 싶으면 정렬하고 싶은 요소에 `dlsplay: flex;`라는 css를 멕이면 된다. 이 때, `dlsplay: flex;`가 멕여진 요소가 flex container, 즉 부모 요소가 되며 그 자식 요소들이 flex item(자식 요소)이 된다.

flex box의 핵심은 이 녀석은 가로or세로의 정해진 방향을 기준으로 요소들을 정렬한다는 것이다 →

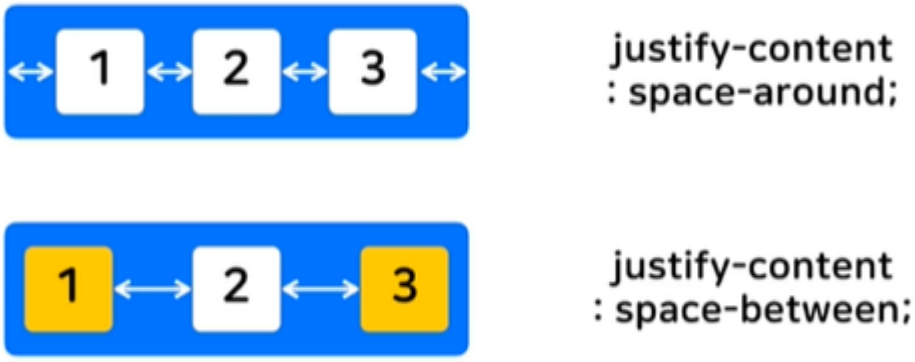
이를 통해 웬만한 웹 레이아웃들을 구현할 수 있다.

이때, flex box에서 부모/자식 요소 별로 쓸 수 있는 속성들은 다르며 각각 다음과 같다.

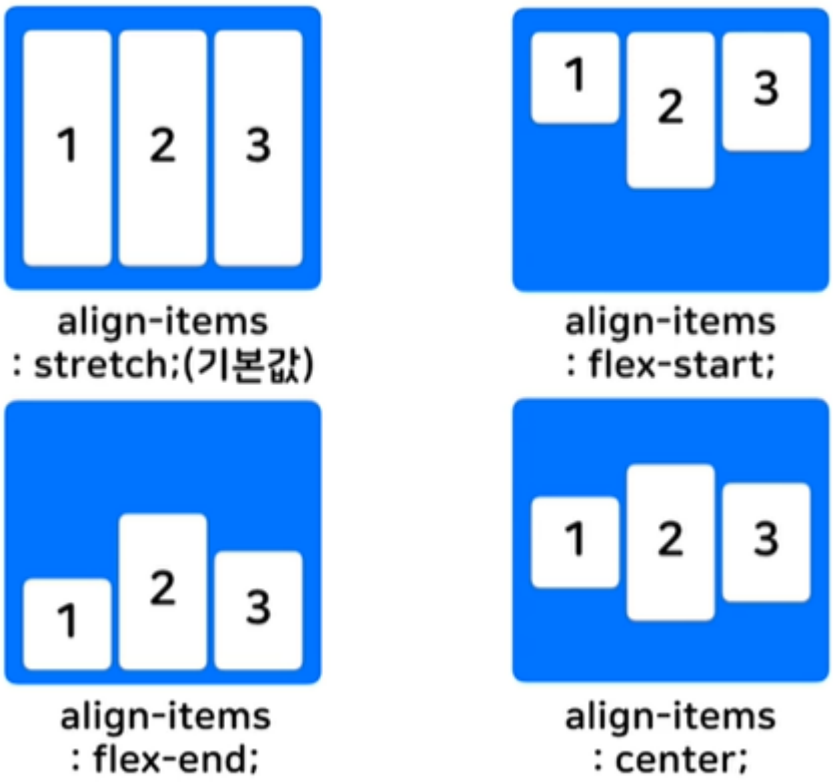
부모 요소(flex Container)	자식 요소(flex item)
flex-direction	flex-grow
flex-wrap	flex-shrink
justify-content	flex-basis
align-items	flex
align-content	

- **flex-direction** : flex 컨테이너 안의 item들의 방향을 정함. 따로 지정하지 않으면 기본으론 row가 들어감. row이면 왼쪽에서 오른쪽으로 정렬하고 column이면 위에서 아래로 정렬하는데 각각의 값 뒤에 `-reverse`가 붙으면 방향이 반대로 정렬됨.

- **flex-wrap** : flex item이 flex container를 벗어났을 때 줄을 바꿔주는 속성. 따로 지정하지 않으면 nowrap이 들어감. 따로 값을 wrap으로 지정해주면 벗어날 때 알아서 줄바꿔줌. 참고로 flex-direnction과 flex-wrap을 `flex-flow: row wrap;`을 통해 한 번에 설정가능함.
- **justify-content** : flex-direction으로 설정된 방향을 기준으로 수평으로(즉 같은 방향으로) item을 정렬하는 방법을 정함. 따로 지정하지 않으면 flex-start가 기본으로 들어가며 center, flex-end라는 값을 지정해줄수 있음. flex-direction이 row라고 할 경우 flex-start는 좌측정렬, center는 중앙정렬, flex-end는 우측정렬을 의미하게 됨. 이 외에도 space-around 또는 space-between을 값으로 줄 수 있다.
  - space-around : 시작과 끝에 item을 하나씩 두고 그 사이의 남은 공간을 동일한 간격으로 배치
  - space-between : 시작과 끝을 기준으로 동일한 간격만큼 item들을 배치



- **align-items** : flex-direction으로 설정된 방향의 **수직방향**으로 item을 정렬하는 방법을 정함. 따로 지정하지 않으면 stretch가 기본으로 들어가며 flex-start, flex-end, center라는 값을 지정해줄 수 있다.



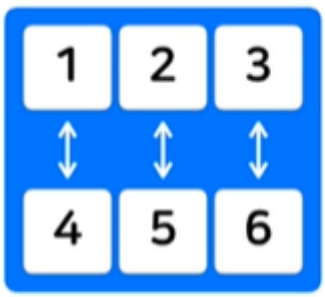
- **align-content** : flex-direction으로 정해진 방향에 수직으로 여러 줄들이 생길 때 item들을 정렬하는 방법을 정함. 따로 지정하지 않으면 stretch가 기본으로 들어가며 flex-start같은 값들을 지정할 수 있다.

align-content
: stretch;(기본값)

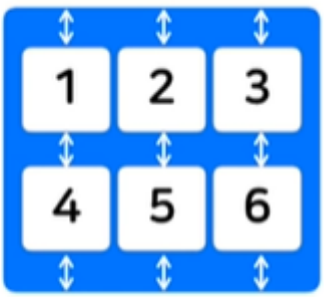
align-content
: flex-start;

align-content
: flex-end;

align-content
: center;



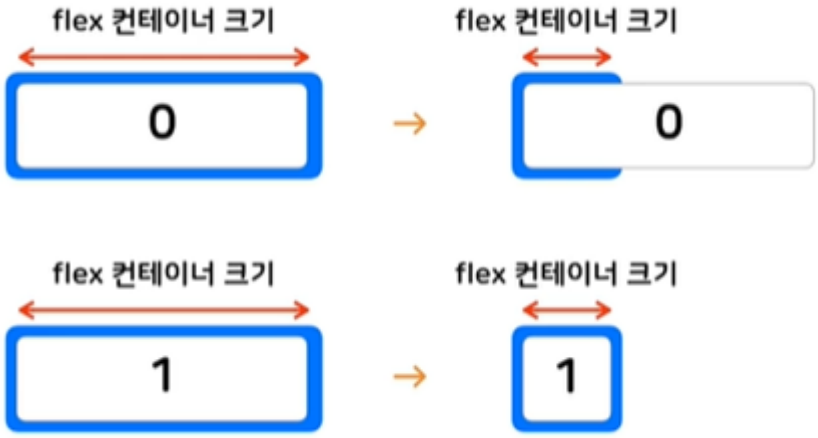
align-content
: space-between;



align-content
: space-around;

- **flex-grow** : flex item의 확장과 관련. 단위없는 숫자를 값으로 쓰며 따로 지정하지 않으면 기본으로 0이 들어간다. 값이 0일 경우 flex container가 커져도 item은 그대로 원래 크기를 유지하지만, 1이상의 값을 가질 경우 item의 원래 크기와 상관없이 container를 채우기 위해 item이 커진다. 같은 flex container안의 item들이 1이상의 flex-grow값을 모두 가지면, 가장 큰 수를 가진놈이 제일 커진다.

- **flex-shrink** : flex item의 축소와 관련. 단위없는 숫자를 값으로 쓰며 따로 지정하지 않으면 기븐으로 1이 들어감. 만약 값이 0이면 container가 item보다 작아져도 item크기는 그대로지만, 1이면 container가 item보다 작아질 때 item도 그만큼 맞춰서 작아진다.



- **flex-basis** : flex item의 기본 크기를 결정. 따로 지정하지 않으면 기본으로 auto가 들어감. 따로 10px, 20px등등의 값을 지정해줄 수 있음.
- **flex** : flex-grow, flex-shrink, flex-basis를 한 번에 지정가능한 속성. `flex: 1 0 auto;` 등으로 사용(순서는 grow - shrink - basis)

상속과 우선순위

상속이란?

조상이나 부모 요소에 적용된 CSS속성을 자식이 물려받은 것. 그러나 모든 CSS속성이 상속되는 것은 아니다. width, height, margin, padding, display등등은 상속되지 않음. 이런 속성들을 상속받길 원하면 속성값으로 inherit를 주면 된다.

우선순위

CSS속성간 적용 우선순위가 있음. 3가지의 규칙이 있다.

1. 중요도
2. 명시도
3. 선언 순서

1. 중요도

<head>태그 내의 <style>태그

<head>태그 내의 <style>태그 내의 @import문

<link>태그로 연결된 CSS

<link>태그로 연결된 CSS내의 @import

문브라우저 디폴트 스타일시트

순으로 우선순위가 있다 위일수록 높음

2. 명시도

!important

인라인 스타일(inline style)

아이디 선택자

클래스, 속성, 가상클래스 선택자

태그 선택자

전체 선택자

상속

순으로 우선순위가 있다. 위일수록 높음. 살짝..더 구체적일수록? 더 범위가 좁을수록? 우선순위가 높은 느낌

3. 선언 순서

나중에 선언된 스타일이 우선 적용된다!

ex)

```
p{
    color: red;
}
p{
    color: green;
}
```

→color: green이 적용된다는 말임