CSS

### CSS 작성 규칙

크기

= px, vh (스크린 크기 유동적임) -> vh 100 주면 스크린 크기 100에 맞추나는 뜻으로 flex와 연결 됨.

= 태그가 겹칠 시 순차적으로 읽기 때문에 맨 아래 있는게 최종적으로 읽힌다.

External 시

* External css 시 = Rel(relationship) 에 대해 명시해야 한다.
* 셀렉터 방법

자식만 가리키고 싶다면 -> combinator

= p(부모) span(칠드런) {

}

= p > span {

text-decoration: underline;

} (다이렉트)

= p (자식) + span(자식) {

text-decoration: underline;

} (p 다음 span)

= p(자식) ~ span(자식) {

text-decoration: underline;

} (자식 과 자식 바로옆이 아니여도 됨.)

= input[placeholder~=“name”]{

(exactly -> name만 가지고 옴)

}

= input[placeholder (속성)\*=name”]{

(Contains -> name 이 들어가 있으면 다 가지고 옴)

}

색깔

{

color: rgb or Reba(알파 -> 트랜스패런싱)

}

# color picker -> 익스텐션

기본 지정 css

:root{

—main-color: ~ (서류 기본 설정을 해 줌.)

—default-border: 1px solid var(—main-color);

}

P {

background-color: var(—main-color);

}

Pseudo-element (가상 요소)

* 가상 클래스는 별도의 class를 지정하지 않아도 지정한 것 처럼 요소를 선택할 수 있음.

:

= :active -> 버튼을 누르면 색이 변함

= :hover -> 마우스 같다 되기만 해도 색이 변함.

= :focus -> 키보드로 선택 시 색이 변 함.

= :visited -> 링크에만 적용되며 한 번 방문하면 색이 바뀜.

= :focus-within -> focused 자식을 가진 부모 엘리먼트의 상태를 말함 -> focused 가 된 개체가 선택되면 같이 바뀜

= *#login-form* input*:not*([*type*="submit"]) {

border-bottom: 1px solid rgba(0, 0, 0, 0.2);

transition: border-color 0.3s ease-in-out;

}

(로그인 폼이라는 submit 인풋 창에 입력이 안 되면 아래 것들이 실행되지 않는다.)

* 가상 요소는 가상클래스처럼 선택자에 추가되며 존재하지 않는 요소를 존재하는 것처럼 부여하여 문서의 특정 부분 선택이 가능함.

:: -> 세부 속성 및 싱글만 손댈 수 있음.

= p::first-letter{

(첫 대문자만 바꿀 수 있음)

}

= input::placeholder{

(placeholder 이름 바꾸는 거)

}

= p::selection{

(커서 땡기면 변함)

}

Ex.

= “div: nth-child(even or odd){

background-color: teal;

}”

= “div: nth-child(4n+1…){

background-color: teal;

}”

<ul class = “test1”>

<li>이: 이발소에가서 이발을 했다</li>

<li>환: 환장하겠네</li>

</ul>

.test1 ul li: first-child {font-weight:bold}

(클래스를 지정하지 않아도 li의 첫번쨰 자식요소를 선택 제어)

.test1 ul li:: first-letter{font-size:30px; color:blue}

(li내의 첫번째 글자를 감싸고 있는 요소가 없어도 있는 것과 같이 제어)

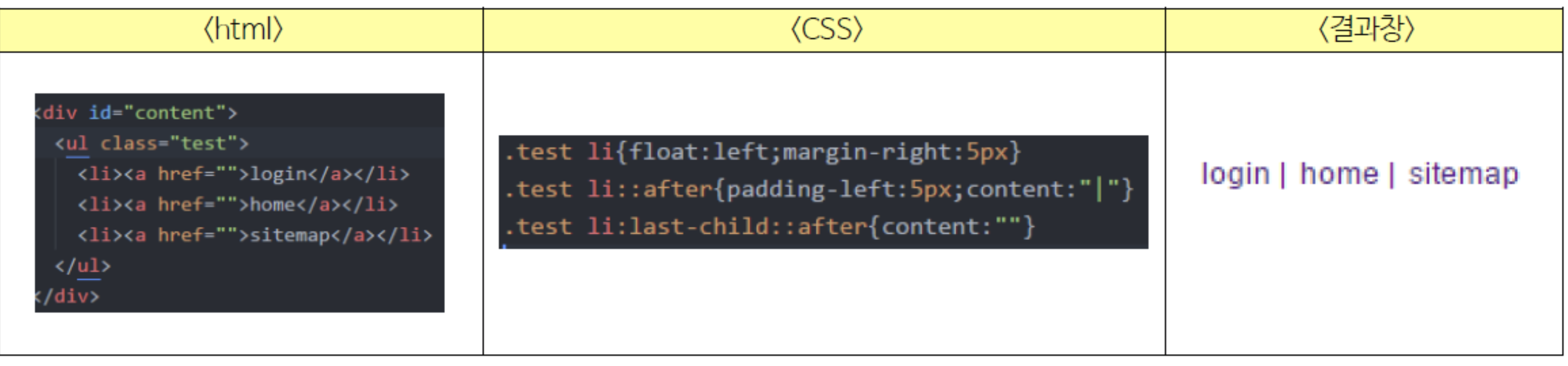
텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트, 테이블이(가) 표시된 사진

자동 생성된 설명

(::before 와 ::after와 꼭 함께 쓰이는 content 가짜 속성 필요 -> HTML 문서에 정보로 포함되지 않은 요소를 CSS에서 새롭게 생성시켜줌)



Content:” “ 로 마무리

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

<a href=””> 만 선택해서 집어넣기 위해 ::를 사용

- 개발자 도구에서 가장 중요한 selector는 'state'다. Styles의 상단 :hov를 누르면 확인 가능

박스 태그

Display: flex; 로 자리 선정 설정한다.

예전 방식

= 대부분의 태그가 block 이다. (display: inline)으로 바꿔줘야 함.

* Block을 display:inline으로 바꾸면 위아래가 사라져 보이지 않는다 -> 해결 위해 display: flex; 존재

= inline 태그 -> span, link, img

* 인라인은 높이와 너비를 가질 수 없어서 위 아래 margin을 가질 수 없다, 갖고 싶다면 display: block으로 바꿔줘야 함.

현재 방식

Ex.

Display: flex;

Justify-content: 주축에 대해 flex 항목들을 정렬하는 방식을 제어 (행 -> 가로 line)

Align-items: 열 row 세로

(flex-direction: column or column-reverse으로 바꿔주면 세로 혹은 가로로 바뀐다.)

텍스트, 녹색, 실외이(가) 표시된 사진

자동 생성된 설명

Space-evenly

Align-self (개별 항목)

* Stretch, center, start, end (속성들)

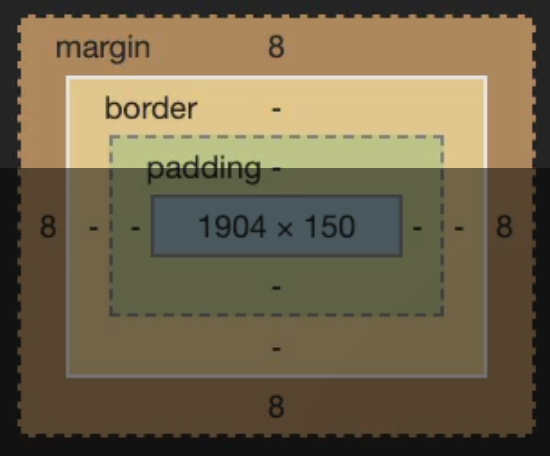
Flex wrap -> flex 시 화면에 따라 박스가 줄어들고 늘어나는 것을 Flex wrap 이라고 하며 “flex-wrap: no wrap” 시 화면에 따라 움직여지지 않는다 / wrap 이면 화면에 따라 움직여 진다.

Box (body>div)

= div 하나로 css 설정 시 바디 통체로 골라짐.

= 두 개를 줄 경우 상하 좌우

= 네 개를 줄 경우 시계방향 순 (상 우 하 좌)



Margin

= space from border of the box to the outside, 박스 경계로부터 바깥에 있는 공간

= collapsing margin -> 박스 경계와 바디 경계가 둘 다 0이면 상하 만 적용되여 같이 밀어냄.

body안에 div의 위 아래 마진이 body의 마진과 만나면 둘 중 큰 값의 마진으로 body에 적용된다

Padding

= collapsing margin 이 일어난다면 패딩을 움직여 바꾸자~

= 패딩을 줄여서 박스를 줄여준다.

Border

= 하나만 쓴다 이유는 별로 안 예뻐서

= border-style or border: ~

**총합**

**= css에서 200px div 만들 시**

**-> width 노 선언 +좌 우 50px padding 시**

**- 50 100 50 = 200px**

**-> width 선언 + 좌 우 50px padding 시**

**- 50 200 50 = 300 px 총합**

**-> width 선언 + 좌 우 50px padding + box-sizing: border-box 선언 시**

**- 내가 padding을 줘도 내 총합 박스 사이즈 늘리지 마.**

**Colorzilla**

**추가 표현**

*#login-form* input[*type*="submit"] {

background-color: var(--yellow);

cursor: pointer;

padding: 20px 0px;

border-radius: 5px;

}

**(cursor = pointer, not-allow, progress 등 다양)**

*#login-form* a {

text-align: center;

text-decoration: none;

color: inherit;

font-size: 13px;

}

**(text-decoration = 텍스트 꾸며주는 속성)**  
text-decoration: underline;

text-decoration: underline dotted;

text-decoration: underline dotted red;

text-decoration: green wavy underline;

text-decoration: underline overline #FF3028;

**(color: inherit -> 부모로부터 기본색을 상속 받는다 -> 로그인 폼으로부터 상속 받는데 기본 색은 루트로 지정해 주지 않는 이상 검정색이다.)**

**폼 태그**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**액션 = 어떤 페이지로 data를 보낼건지 지정할 수 있다.**

**(action = “friends.html” 이면 friends.html로 이동한다.)**

**Method = get 방식 post 방식 두 가지 사용 할 수 있음**

* **Post = 백엔드 서버에 정보를 전송하는 방식 (DB 필요)**
* **Get = 보안에 취약한 방식으로 그냥 쿠키로 다 보내며 페이지를 그냥 넘기는 방식**

**Nav 태그**

**= 큰 페이지 또는 현재 페이지의 다른 부분과 연결되는 네비게이션 링크들의 집합을 정의할 때 사용.**

**= <nav> 요소를 사용하는 일반적인 예로는 메뉴 목차 인덱스 등이 있다.**

**= 글을 읽지 못하는 사용자를 위한 스크린 리더기와 같은 브라우저는 nav 요소를 사용하여 해당 콘텐츠의 초기 렌더링을 생략할지 여부 결정.**

**= vsc -> “nav>ul>li\*4>a**

**Layer 층**

* **Position: fixed**

**= 새로운 레이어를 만들어 레이어를 고정 시킴, 이 때 만들어진 레이어의 포지션 기본 벨류는 이전에 작성된 콘텐츠의 끝을 기준이 됨.**

**= 이를 수정할 수 있음 -> top: 1px, bottom: 1000px 등등**

* **Position: static**

**= 레이아웃이 박스를 처음 위치하는 곳에 두는 것, 기본**

* **Position: relative**

**= element가 처음 생성된 위치를 기준으로 top bottom left right 위치를 조금씩 수정할 수 있다.**

* **Position: absolute**

**= 가장 가까운 relative 부모를 기준으로 이동 -> 부모에 position: relative 를 같이 해주면 속성이 들어간 게 부모가 되고 아니면 바디전체가 부모가 된다.**

**Font**

**= font-weight -> 폰트 굵기**

**=**

**태그 규칙**

**= 왠만하면 클래스로 -> BEM 모드**

**= very very unique -> id로 주자.**

**반응형 웹 만들기**

**= vscode -> ! -> header 내 meta -> name = “viewport” content = “width=device-width, initial-scale=1”**

**(모바일 반응형 -> viewport 란 웹페이지에서 사용자의 보이는 영역을 말 함.)**

**(width -> 뷰포트의 가로 크기를 정함 -> 만약 500 이란 값을 넣으면 500 만큼의 픽샐 개수를 가진 너비로 설정, device-width라고 지정했으니 기기의 스크린 너비에 맞추라는 뜻)**

**(initial-scale -> 페이지 처음 접속 시 보여질 배율)**

**= 뷰포트를 정하지 않으면 width 기본값은 980 -> width:100% 지정 시 980px 값을 가지게 됨.**