



4주차 CSS

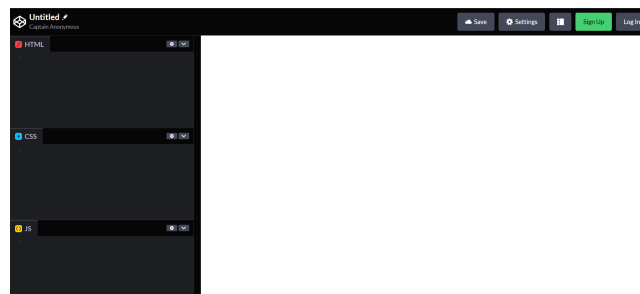
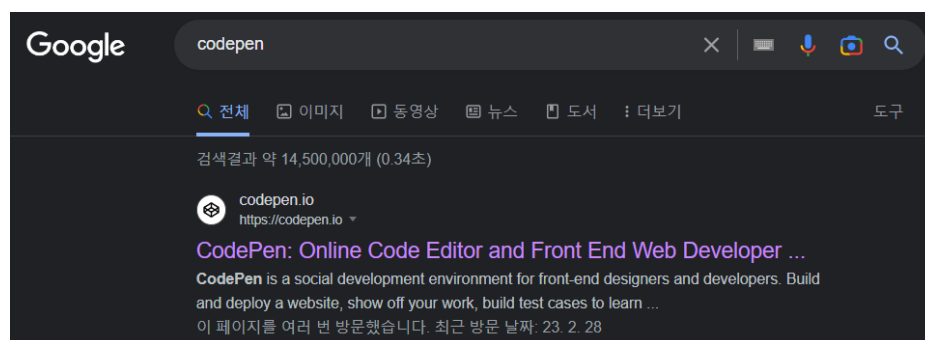
☼ 상태	승인
≡ 설명	CSS 속성 (width,margin,text등)

CSS 속성

CSS에는 굉장히 많은 속성이 존재한다. 또한 많은 속성마다 다른 옵션들이 존재하는데 모두를 공부하고 외우는 것은 현실적으로 거의 불가능하다. 따라서 대표적인 속성들을 실습해 보는 것을 목표로 해 보자.

본 교재에서 소개하지 않는 다른 속성들은 이전에 소개했던 [MDN](#) 문서에서 찾아볼 수 있다.

[CSS](#) 속성을 조금 더 효율적으로 학습하기 위해, [VS CODE](#) 를 잠시 내려놓고 이전에 잠깐 소개하였던 [CODEPEN](#) 을 활용해 보자.



우측상단 Sign Up 왼쪽의 버튼을 눌러 보는 방식을 변경한 뒤 실습을 진행하자.



실습 시 주의사항 ?

책의 예제를 따라한 후에는 꼭 다른 태그들이나 다른 속성의 값들을 넣어 보면서 실습해보도록 하자.

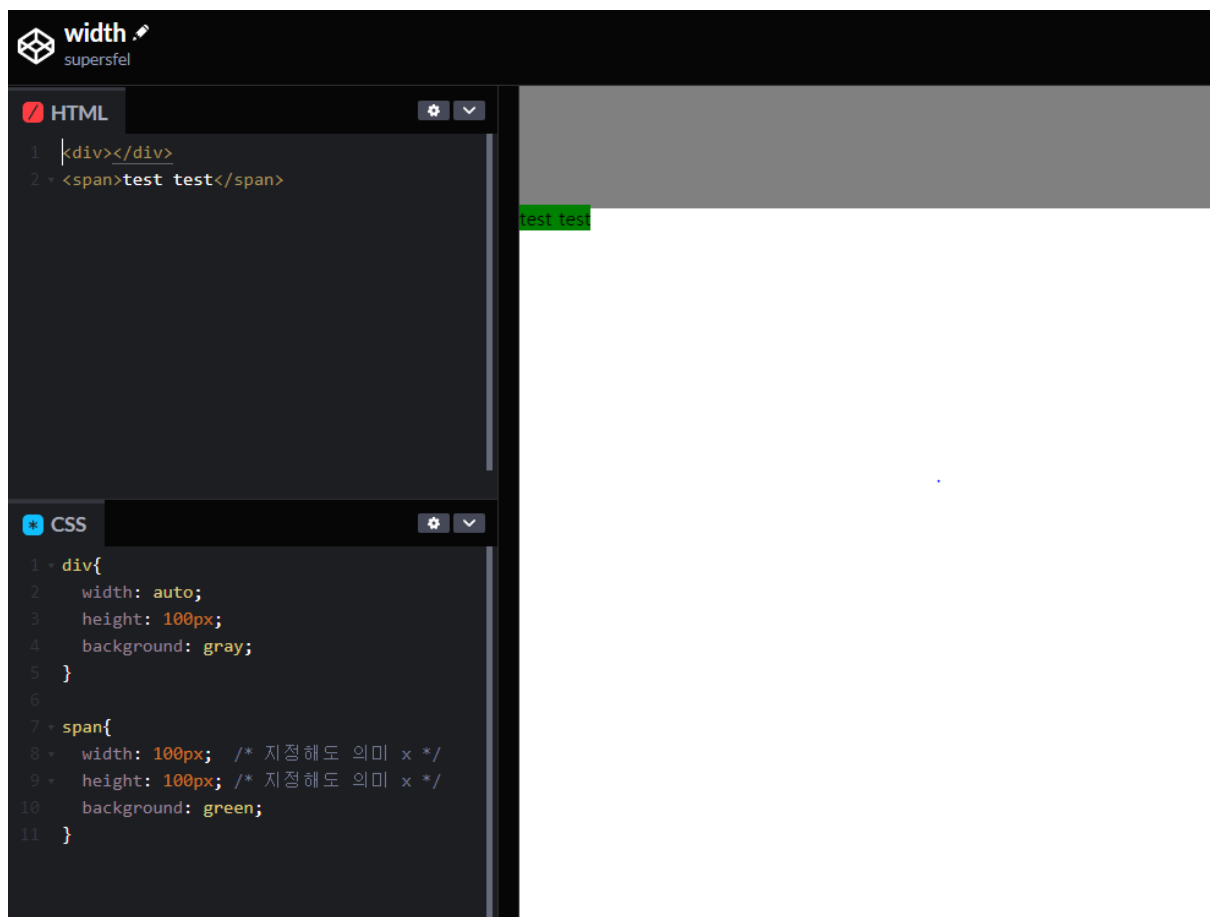
특히, CSS 속성의 값들을 이리 저리 바꾸어보면서 어떠한 동작을 하는지 유의깊게 보자.

width

요소의 가로 길이를 정할 수 있는 속성이다. block 속성과 inline 속성의 기본으로 가지게 되는 값이 다르다.

block 기본 : 100%

inline 기본 : 0



height

요소의 높이를 정할 수 있는 속성이다.

block 기본 : 0px

inline 기본 0

>> 인라인요소는 가로세로를 설정해도 적용이 되지 않는다.

max-width

요소의 최대 가로넓이

min-width

요소의 최소 가로넓이

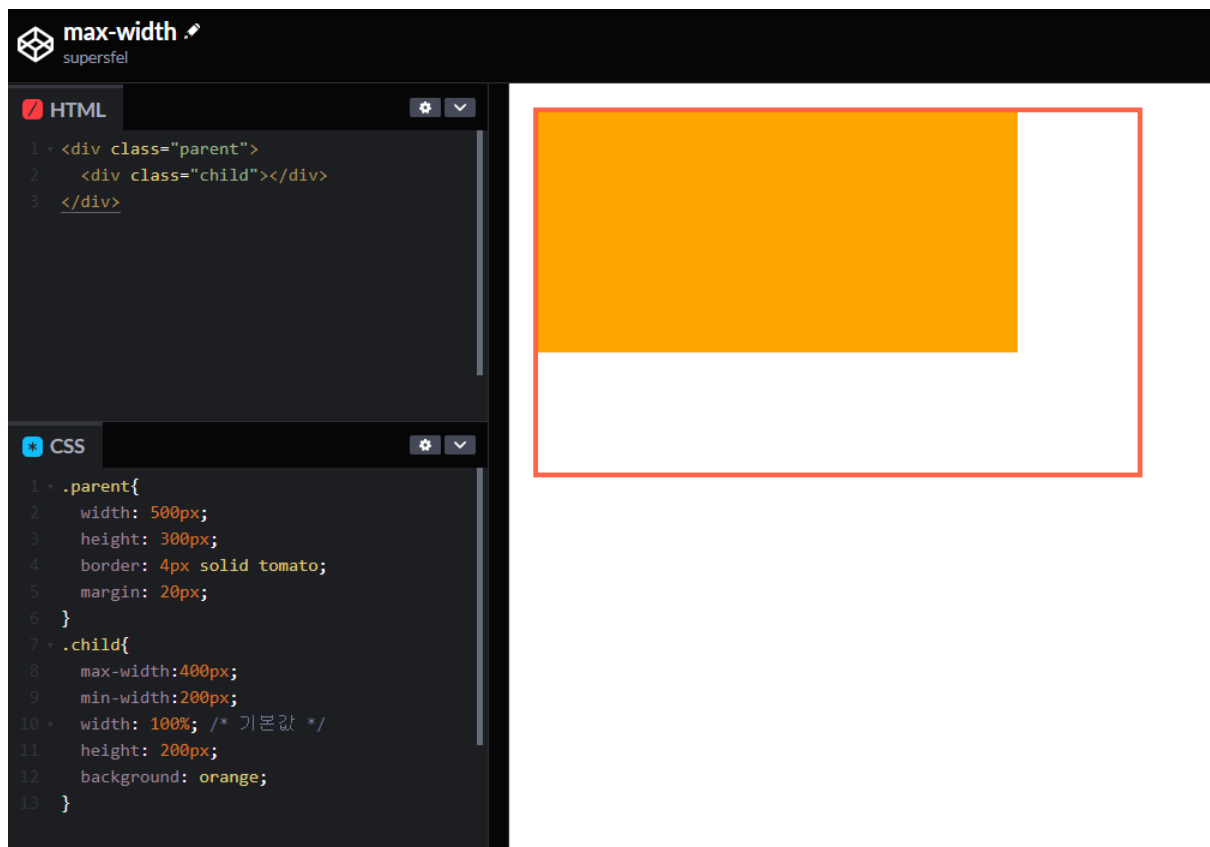
max-height

요소의 최대 세로넓이

min-width

요소의 최소 세로넓이

>> 기본값이 **none** 이다.



위 속성들은 **width** 가 무한정 작아지거나 커져서 화면을 망치는 걸 방지해준다.

margin

요소의 외부(바깥) 여백을 지정해주는 특성이다.



단축특성단축특성??

top,left,right,bottom 을 아우를수 있는 속성이며 음수의 값도 사용할 수 있다

margin : 10 px 20 px 30 px 40 px

위 우 아래 좌

margin : 10 px 20 px 30 px

위 [좌,우] 아래

margin : 10 px 20 px

[위,아래] [좌,우]

margin : 10 px

[위,아래,좌,우]

>>시계방향임을 알수있다

개별 특성으로 적는 방법

1. margin-top :
2. margin-bottom :
3. margin-left :
4. margin-right :

```
HTML
1 <div class="parent">
2   <div class="child"></div>
3 </div>

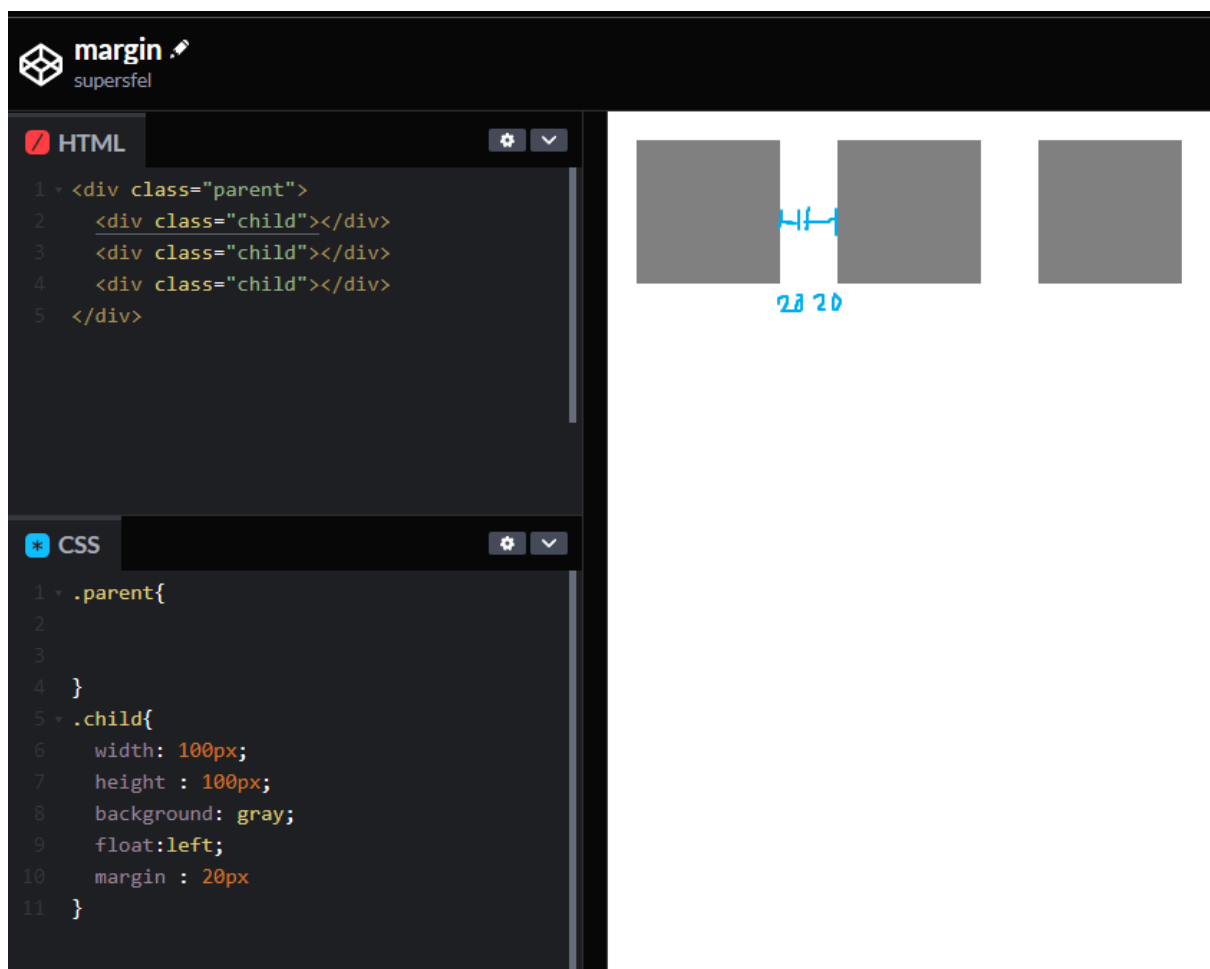
CSS
1 .parent{
2   width: 400px;
3   height: 200px;
4   border: 4px solid red;
5 }
6
7 .child{
8   width: 100px;
9   height: 100px;
10  border : 4px solid;
11  margin: 50%;
12 }
```

| **margin** 에 **%** 를 적을 시 위처럼 부모 요소의 값을 가져온다는 것 을 확인할 수 있다.

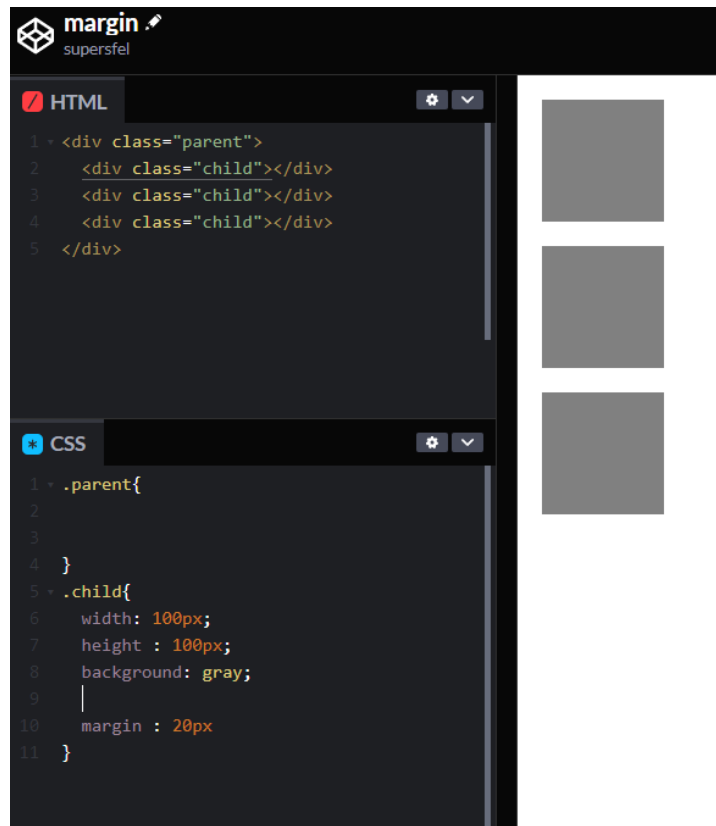
마진중복(병합,Collapse)

마진의 특정 값들이 중복되어 합쳐지는 현상으로 아래와 같은 경우에 발생한다.

형제 요소의 margin-top과 margin-bottom이 만날때
부모요소의 margin-top과 자식요소의 margin-top이 만났을 때
부모요소의 margin-bottom과 자식요소의 margin-bottom이 만났을 때

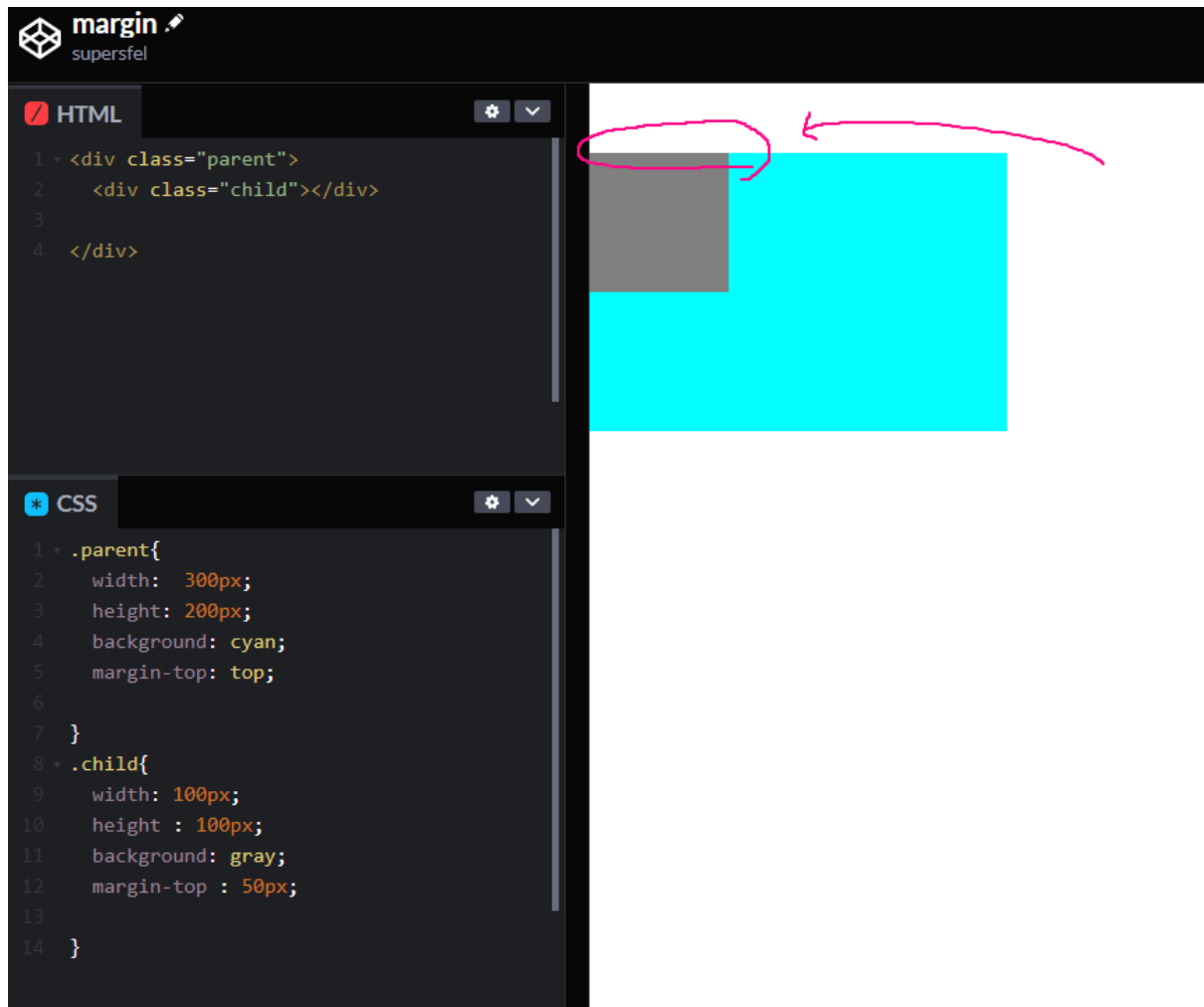


단, 위와 같이 형제 요소의 좌-우로는 중복이 되지 않는다.



위의 경우 `margin-top` 과 `margin-bottom` 이 만났다는 것을 알 수 있다.

마진 중복 계산법

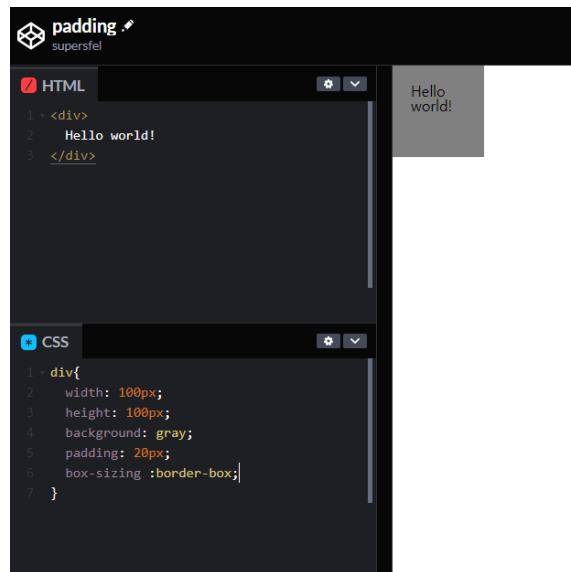


부모와 자식간의 관계일때, 즉 중복이 위 그림처럼 겹쳐서 만나지게 될 때 **마진중복** 현상이 일어나게 된다!

padding

요소의 '내부여백'을 지정하는 속성

마진과 위,우,아래,좌 쓰는방법이 동일하다.



padding 예시

border

요소의 테두리선을 지정하는 속성

마진, 패딩과 같은 방식으로 상하좌우 선택이 가능하다.

border-width : 선의 두께

border-style : 선의 종류

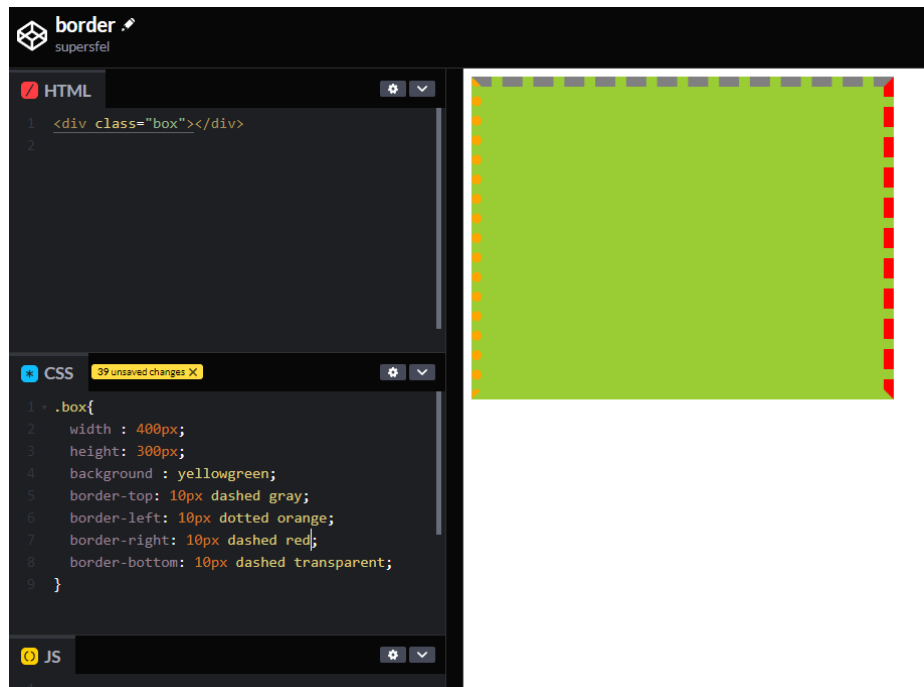
none : 선없음
 hidden : 선 없음
 solid : 실선
 dotted : 점선
 dashed : 파선
 double : 두줄선
 groove : 홈이 파여있는 모양
 ridge : 솟은 모양
 inset : 요소 전체가 들어간 모양
 outset : 요소 전체가 나온 모양

border-color : 선의 색상



border를 추가하면 크기가 선의 크기만큼 커진다!

요소의 크기가 20px * 20px 이라고 할때, border를 2px로 4방향에 준다면 요소의 크기는 24px * 24px가 된다.



`transparent` 는 배경색과 같은 색으로 테두리를 지정해주는 것이다.

`boxsizing`

요소의 크기계산 기준을 지정하는 것

`content-box` : 너비,높이만으로 요소크기를 계산

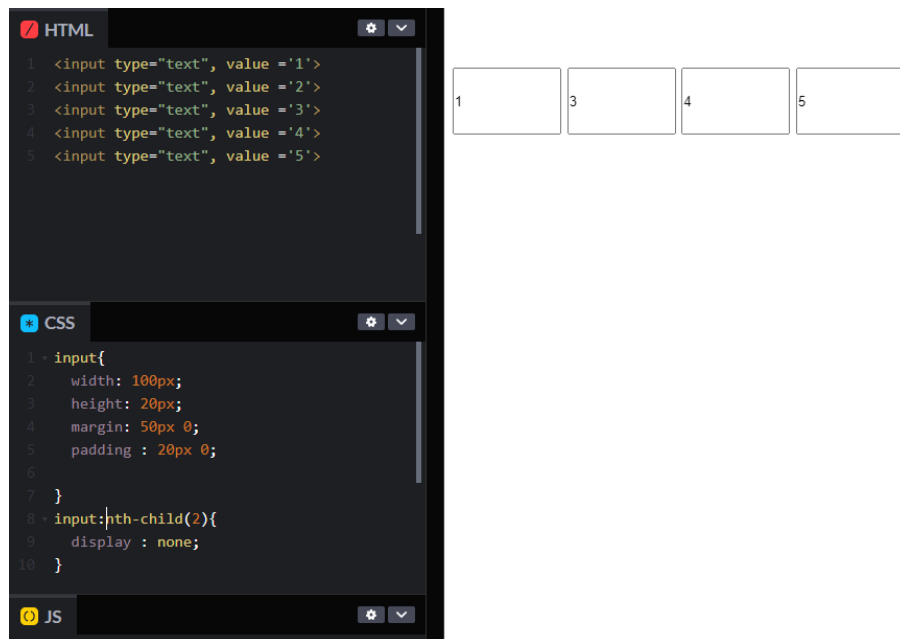
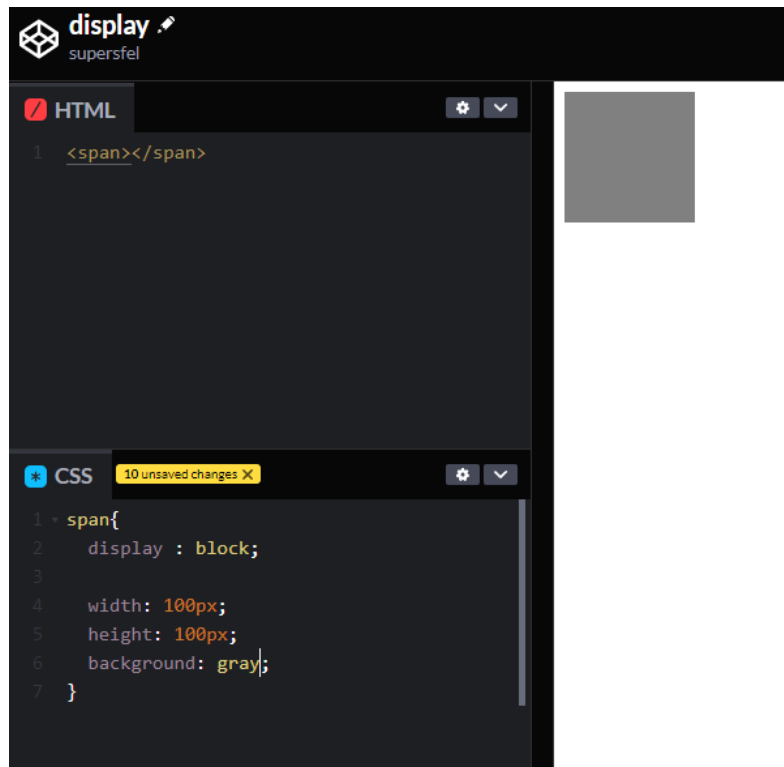
`border-box` : 너비,높이에 padding,border을 포함하여 요소의 크기를 계산

`display`

요소의 박스타입(유형) 결정

`block` `inline` `inline-block` (`input`)

즉, `inline` 이나 `block` 으로 설정되어 있는 태그들의 유형을 바꿀 수 있다.



display : none

display 를 none 으로 설정하면 해당 요소를 사라지게 하는 것이 가능하다.
이를 이용하여 Modal 창 등을 구현하는데 사용할 수 있다.

overflow

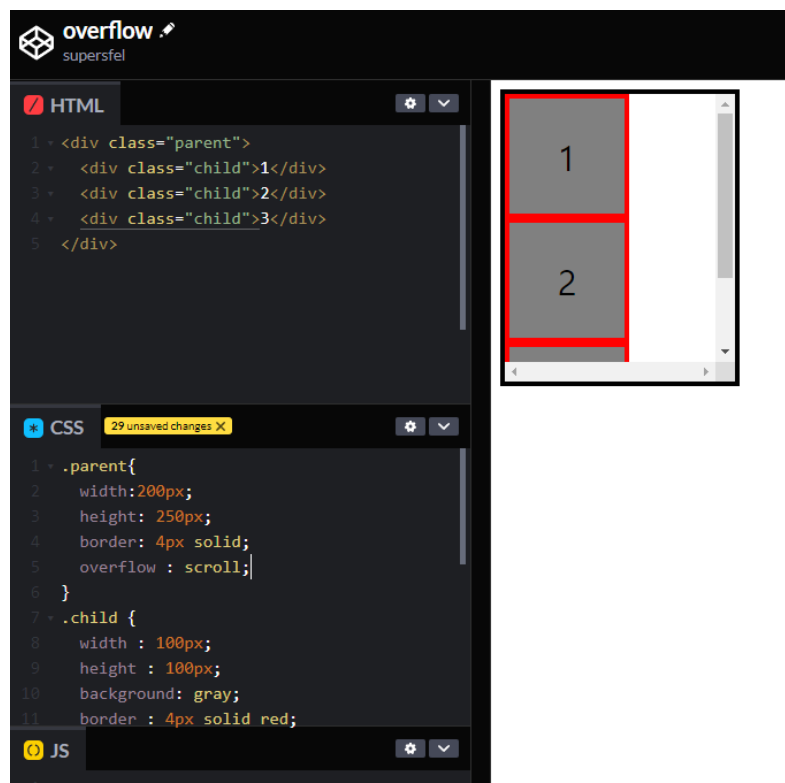
요소의 크기 이상으로 내용이 넘쳤을 때, 내용의 보여짐을 제어

visible : 그대로 표시

hidden : 넘친부분을 잘라냄

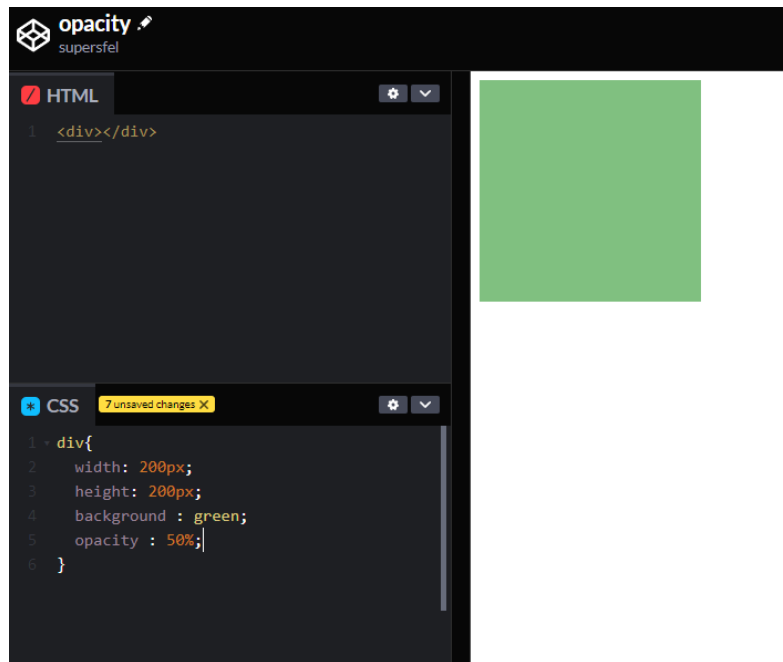
scroll : 강제로 스크롤바를 만듦

auto : 넘치는 부분이 있는 경우에만 스크롤바를 만듦



opacity

투명도를 설정



font

font-style : 글자 기울기 지정

font-weight : 글자 두께 지정

font-size : 글자 크기 지정

line-height : 줄높이(줄간격) 지정

font-family : 글꼴 지정

font : 기울기 두께 크기 / 줄높이 글꼴;

>> /(슬래시) 가 들어가는 부분이 있다.



/ (슬래시) 가 들어가는 이유 ?

크기와 줄높이 모두 **px** 값으로 설정이 가능하기 때문에 헷갈림을 방지하기 위함이다.
또한 **font-size** 와 **font-family** 는 필수로 입력해야 사용이 가능하다.



글꼴을 관리해 봅시다.

font-style

font-style : 스타일 없음

italic : 이탤릭체

oblique : 기울어진 글자

font-weight

normal : 두께(400)
bold : 두껍게 (700)
bolder : 부모요소보다 더 두껍게
lighter : 부모요소보다 더 얇게
숫자 : 100~900까지의 100단위의 숫자 9개로 설정

100-Thin(Hairline)
200-Extra Light(Ultra Light)
300-Light
400-Normal
500-Medium
600-Semi Bold(Demi Bold)
700-Bold
800-Extra Bold(Ultra Bold)
900-Black(heavy)

- bolder
100~300 >> 400
400~500 >> 700
600~900 >> 900
- lighter
100~400 >> 100
600~700 >> 400
800~900 >> 700

border와 lighter은 상대적으로 작동한다!

font-size

% : 부모 요소의 비율로 지정

단위 : px,em,cm 등 지정

line-height

줄의 높이를 설정하는 속성

normal : 브라우저의 기본 정의를 사용

숫자 : 요소 자체 글꼴 크기의 배수로 지정

단위 : px,em,cm 등 단위로 지정

% : 요소 자체 글꼴 크기의 비율로 지정



글꼴을 관리해 봅시다.

글꼴을 관리해 봅시다.

font-family

font-family[글꼴후보1,글꼴후보2...], 글꼴계열;



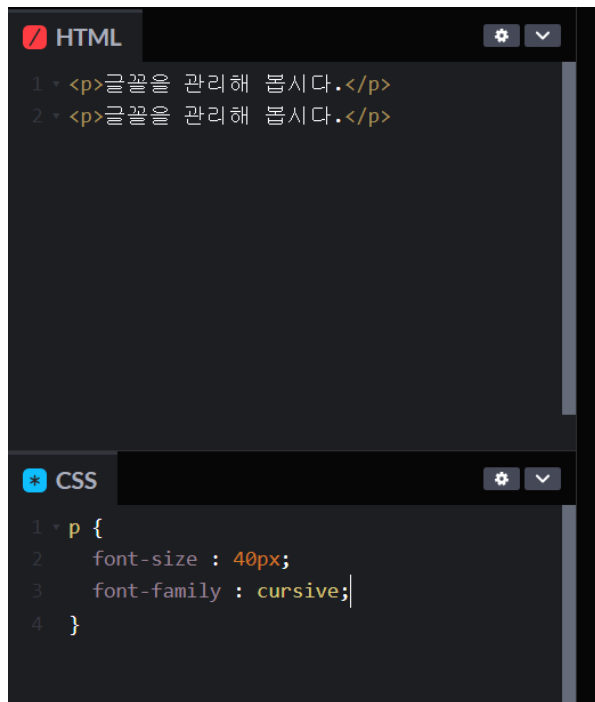
웹에서의 font !

폰트라는 계열은 용량이 굉장히 큰 편이라서 웹에 담아두는것이 아닌, 글꼴이름을 올려둔 후 사용자 브라우저에서 찾는 개념이다.

사용자가 글꼴후보1이 없다면, 글꼴후보 2로 넘어가게 되는 것이다. 아무것도 없다면 마지막 , (쉼표)이후에 오는 고딕체 계열중에서 아무거나 넣는다는 의미이다.

아래는 글씨체 종류이다.

serif - 바탕체 계열
sans-serif - 고딕체 계열
monospace - 고정너비 글꼴계열
cursive - 필기체 계열
fantasy - 장식(재미있는 문자 표현을 포함하는)글꼴계열



글꼴을 관리해 봅시다.
글꼴을 관리해 봅시다.

문자(TEXT) 관련 속성

color : 문자의 색상을 지정

색상이름 : 브라우저에서 제공하는 색상의 이름 (red, blue)

Hex 색상코드 : 16진수 색상[#000000]

RGB : [rgb(255,255,255)]

RGBA : [rgba (255 , 0 , 0 , .5)] << 마지막은 투명도

HSL : 색상, 채도, 명도 [hsl(120,100%,50%)]

HSLA : 색상, 채도, 명도, 투명도 [hsla(120,100%,50%, .3)]

! 실제에서 Hex색상코드, RGB, RGBA를 많이 사용하게 된다!



글꼴을 관리해 봅시다.

글꼴을 관리해 봅시다.

글꼴을 관리해 봅시다.

text-align

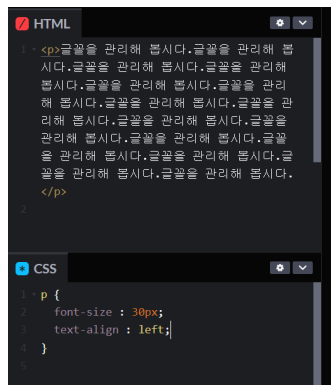
문자 정렬 방식을 지정

left-왼쪽 정렬

right-오른쪽 정렬

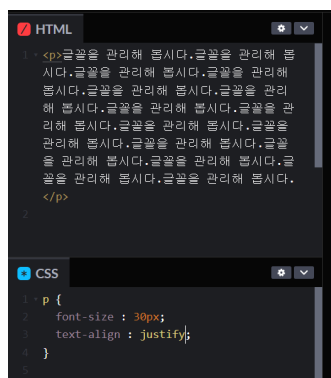
center - 가운데 정렬

justify - 양쪽 맞춤



글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해
봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을
관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.
글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해
봅시다.

아래와 같이 justify를 사용하면 우측이 딱 정렬되게 된다.



글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해
봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을
관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.
글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해 봅시다.글꼴을 관리해
봅시다.

text-decoration

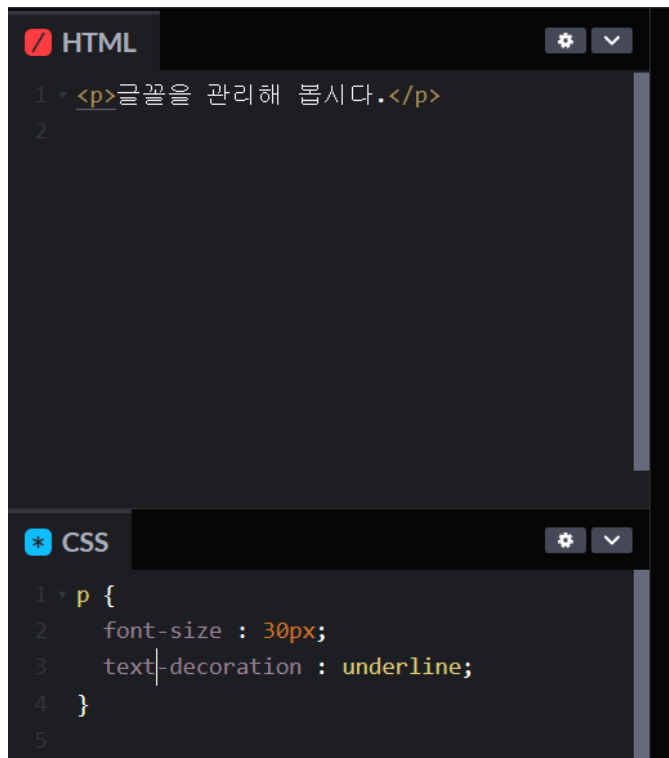
문자의 장식(line)을 설정

none : 선 없음

underline : 밑줄 지정

overline : 윗줄 지정

line-through : 중앙선 지정

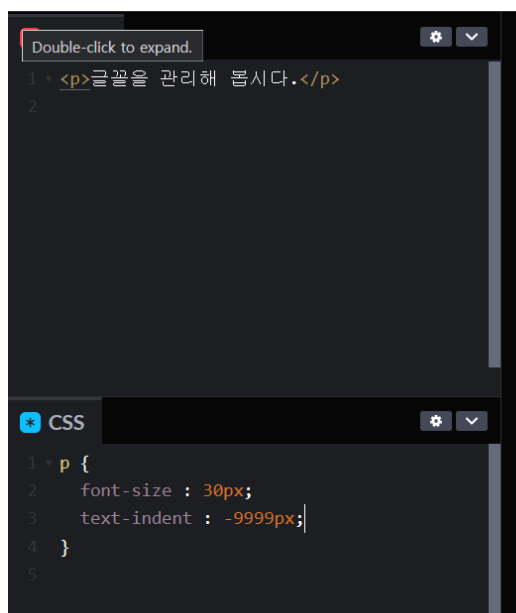


글꼴을 관리해 봅시다.

text-indent
들여쓰기를 지정하는 속성

음수를 사용할 수 있다!! >> 내어쓰기

특히, 내어쓰기 기능을 사용하면, 글자가 화면에서 출력되지 않게 하는 스킬로써 사용이 가능하다



float

요소를 좌우 방향으로 띄움(수평 정렬)

```
<article>

  <div class="picture">

  </div>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque
    quis velit accumsan, consequat risus vel, suscipit mi. Phasellus non
    risus risus. Morbi sed est id purus vehicula porta. Sed accumsan arcu
    quis egestas convallis. Integer imperdiet lectus vel nisi molestie
    cursus. Quisque a turpis neque. Sed ultrices sapien nec leo lacinia, in
    tempus massa condimentum. Aliquam erat volutpat. Sed hendrerit fringilla
    quam maximus fermentum. Praesent sed nisi nisl. Cras a congue nulla.
    Praesent non iaculis nibh. Etiam commodo turpis sed finibus lacinia.
  <div class = "text">
    Curabitur mattis orci at magna luctus aliquet. Fusce finibus tortor
    cursus porttitor condimentum. Fusce vitae tortor finibus nulla tempor
    vehicula. Suspendisse vehicula elementum velit, nec tincidunt metus
    ultrices at. Phasellus dictum euismod elit semper scelerisque. In
    laoreet magna non magna vulputate finibus.
  Suspendisse ex dui, venenatis
    sit amet libero sed, faucibus tempus lorem. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque
    quis velit accumsan, consequat risus vel, suscipit mi. Phasellus non
    risus risus. Morbi sed est id purus vehicula porta. Sed accumsan arcu
    quis egestas convallis. Integer imperdiet lectus vel nisi molestie
    cursus. Quisque a turpis neque. Sed ultrices sapien nec leo lacinia, in
    tempus massa condimentum. Aliquam erat volutpat. Sed hendrerit fringilla
    quam maximus fermentum. Praesent sed nisi nisl. Cras a congue nulla.
    Praesent non iaculis nibh. Etiam commodo turpis sed finibus
  lacinia.
    Curabitur mattis orci at magna luctus aliquet. Fusce finibus tortor
    cursus porttitor condimentum. Fusce vitae tortor finibus nulla tempor
    vehicula. Suspendisse vehicula elementum velit, nec tincidunt metus
    ultrices at. Phasellus dictum euismod elit semper scelerisque. In
    laoreet magna non magna vulputate finibus. Suspendisse ex dui, venenatis
    sit amet libero sed, faucibus tempus lorem.
  </div>
</article>
```

```
article{

}

article .picture{
  width : 200px;
  height: 150px;
  background : gray;
  float : left;
  margin-right : 20px;
  margin-bottom : 10px;
}
article .text{

  clear : left; /*float 해제*/
}
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque quis velit accumsan, consequat risus vel, suscipit mi. Phasellus non risus risus. Morbi sed est id purus vehicula porta. Sed accumsan arcu quis egestas convallis. Integer imperdiet lectus vel nisi molestie cursus. Quisque a turpis neque. Sed ultrices sapien nec leo lacinia, in tempus massa condimentum. Aliquam erat volutpat. Sed hendrerit fringilla quam maximus fermentum. Praesent sed nisi nisl. Cras a congue nulla. Praesent non iaculis nibh. Etiam commodo turpis sed finibus lacinia.

Curabitur mattis orci at magna luctus aliquet. Fusce finibus tortor cursus porttitor condimentum. Fusce vitae tortor finibus nulla tempor vehicula. Suspendisse vehicula elementum velit, nec tincidunt metus ultrices at. Phasellus dictum euismod elit semper scelerisque. In laoreet magna non magna vulputate finibus. Suspendisse ex dui, venenatis sit amet libero sed, faucibus tempus lorem. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque quis velit accumsan, consequat risus vel, suscipit mi. Phasellus non risus risus. Morbi sed est id purus vehicula porta. Sed accumsan arcu quis egestas convallis. Integer imperdiet lectus vel nisi molestie cursus. Quisque a turpis neque. Sed ultrices sapien nec leo lacinia, in tempus massa condimentum. Aliquam erat volutpat. Sed hendrerit fringilla quam maximus fermentum. Praesent sed nisi nisl. Cras a congue nulla. Praesent non iaculis nibh. Etiam commodo turpis sed finibus lacinia. Curabitur mattis orci at magna luctus aliquet. Fusce finibus tortor cursus porttitor condimentum. Fusce vitae tortor finibus nulla tempor vehicula. Suspendisse vehicula elementum velit, nec tincidunt metus ultrices at. Phasellus dictum euismod elit semper scelerisque. In laoreet magna non magna vulputate finibus. Suspendisse ex dui, venenatis sit amet libero sed, faucibus tempus lorem.

none : 요소 띄움 없음

left : 왼쪽으로 띄움

right : 오른쪽으로 띄움

float : 방향

float을 사용하면 꼭 해제를 사용해야한다.

해제를 안하면 요소들이 겹치는 현상이 일어난다.

```
clear : both
```

를 사용하면 left, right가 모두 해제된다.

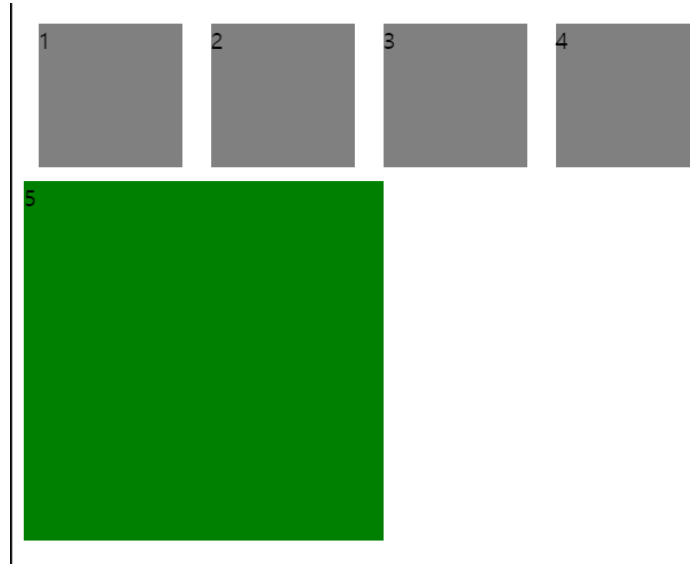
```
<div class='clearfix'>
  <div class="float-box">1</div>
<div class="float-box">2</div>
<div class="float-box">3</div>
<div class="float-box">4</div>
</div>
<div class="new-box">5</div>
```

```
.clearfix::after{
  content:"";
  clear : both; /* 제거하면서 실습*/
  display:block;
}

.float-box{
  width : 100px;
  height : 100px;
  background : gray;
  margin : 10px;
  float : left;
}

.new-box{
```

```
width : 250px;
height : 250px;
background : green;
}
```



float 해제

float 속성이 적용된 요소의 주위로 다른 요소들이 흐르게 되는데, 이를 방지하기 위해서 꼭 해제를 해주어야 한다.

1. 형제요소에 `clear : (left, right, both)`를 사용하여 해제
2. 부모요소에 `overflow : (hidden, auto)`를 사용하여 해제
3. 부모요소에 `clearfix`클래스를 추가 (**** 제일 추천 ****)

1번같은 경우, 다음 형제요소가 없는 경우 다음요소를 추가해야하는 문제가 발생한다.

2번같은 경우, `float`과 전혀 다른 속성을 이용하는 방법이므로 편법같은 방법이다.

따라서 3번을 제일 추천한다.

```
<div class="parent clearfix">
<div class : "child"></div>
<div class : "child"></div>
</div>

.clearfix::after {
content:"";
clear:both;
display : block;
}
```

```
.child{
float : left;
}
```

위처럼 미리 `clear` 할 부분을 클래스로 만들어 놓고, 태그에 `clearfix` 를 붙여서 사용하는 방식이다.

clearfix가 있는 태그안에는 무조건 float이 있는 형제요소들만 포함되어야한다.

💡 float 해제가 어렵다면 ?

요소를 좌측,우측 혹은 가운데 정렬하는것은 `float` 으로만 할 수 있는것은 아니다. `flex-box` 나 `grid` 를 통해서도 정렬이 가능하다. 추후 배우겠지만 `flex-box` 혹은 `grid` 를 사용하는 것이 보다 편리하다.

단, `flex-box` 는 크롬 등 비교적 최신 브라우저에서만 지원하기에 모든 브라우저를 대상으로 하는 페이지를 대상으로 개발을 한다면 사용에 적합하지 않다.

아래는 `flex css` 요소를 사용할 수 있는 웹 브라우저 목록이다

Chrome	Edge	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS	Samsung Internet	Opera Mini	Opera Mobile	Browser for Android	Android Browser	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
4-20			10-11.5													
21-28		3.1-6	2-21	12.1			3.2-6.1									
29-109	12-109	6.1-8	22-27	15-16	6-9		7-8.4						2.1-4.3			
110	110	9-16.2	28-109	17-94	10		9-16.2	4-19.0		12.1		4.4-4.4.4				2.5
111-113		16.4-TP	111-112			110	16.3	20	all	73	13.4	109	110	13.1	13.18	3.1

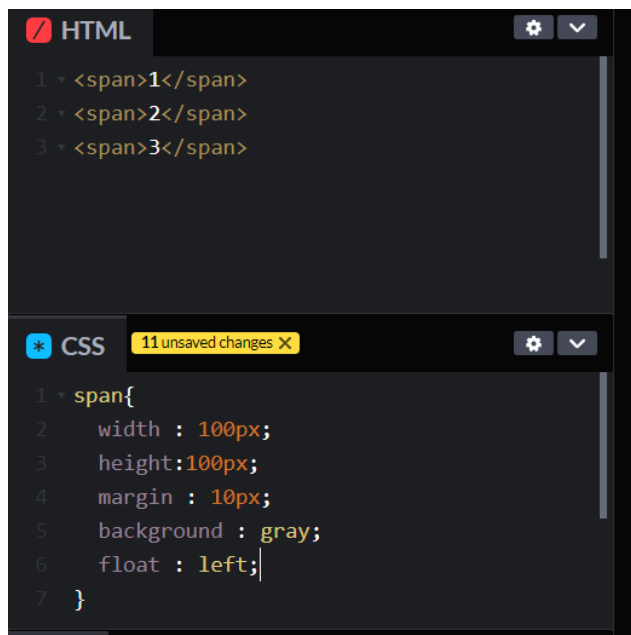
아래는 `grid css` 요소를 사용할 수 있는 웹 브라우저 목록이다.

Chrome	Edge	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS	Samsung Internet	Opera Mini	Opera Mobile	Browser for Android	Android Browser	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
4-28			2-39													
29-56			40-51	10-27												
57	12-15	3.1-10	52-53	28-43	6-9		3.2-10.2	4-5.4								
58-109	16-109	10.1-16.2	54-109	44-94	10		10.3-16.2	6.2-19.0		12-12.1		2.1-4.4.4				2.5
110	110	16.3	110	95	11	110	16.3	20	all	73	13.4	109	110	13.1	13.18	3.1
111-113		16.4-TP	111-112				16.4									

display 수정

`float` 속성이 추가된 요소는 `display` 속성의 값이 대부분 `block` 으로 수정된다.

`flex`, `inline-flex` 는 `block` 으로 변화하지 않는다! 대신 `inline`, `inline-block` 등 대다수의 값이 바뀐다.



The screenshot shows a code editor with two panels. The top panel is titled 'HTML' and contains three lines of code: `1 1`, `2 2`, and `3 3`. The bottom panel is titled 'CSS' and shows a CSS rule for `span` with the following properties: `width : 100px;`, `height:100px;`, `margin : 10px;`, `background : gray;`, and `float : left;`. The CSS panel also indicates '11 unsaved changes'.

```
HTML
1 <span>1</span>
2 <span>2</span>
3 <span>3</span>

CSS
1 span{
2   width : 100px;
3   height:100px;
4   margin : 10px;
5   background : gray;
6   float : left;
7 }
```

