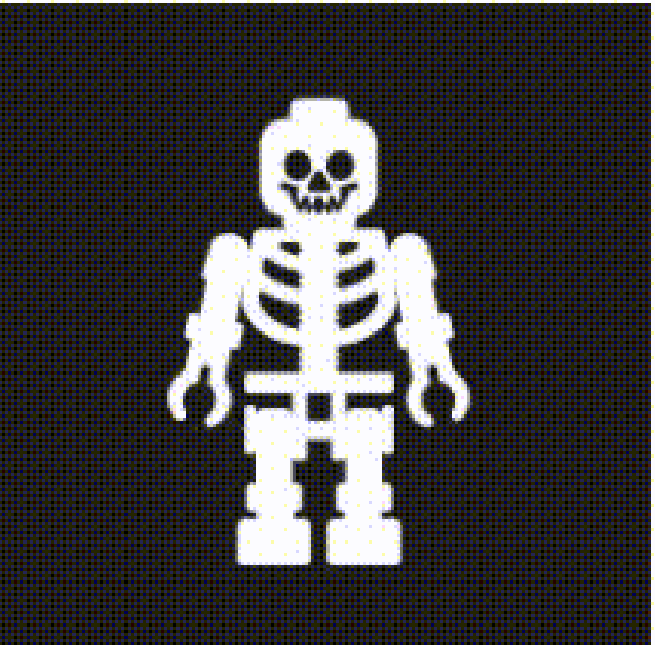
WEB

HTML
structure



CSS
presentation/appearance

JavaScript
dynamism/action

UI / UX

User Interface / User Experience

UI, 사용자 환경

사용자가 특정 콘텐츠를 이용할 때 만나는 접점

UX, 사용자 경험

사람들이 특정 콘텐츠를 사용할 때 UI에서 느낄 수 있는 사용자 경험

웹 브라우저

Web Browser

웹 페이지를 확인하기 위해 브라우저가 필요합니다.

Internet Explorer는 22년 6월 15일 공식 지원이 종료됨에 따라 웹 개발 시 Edge를 고려하여 구현합니다.

웹 에디터

Web Editor

Notepad++

Sublime Text

Atom

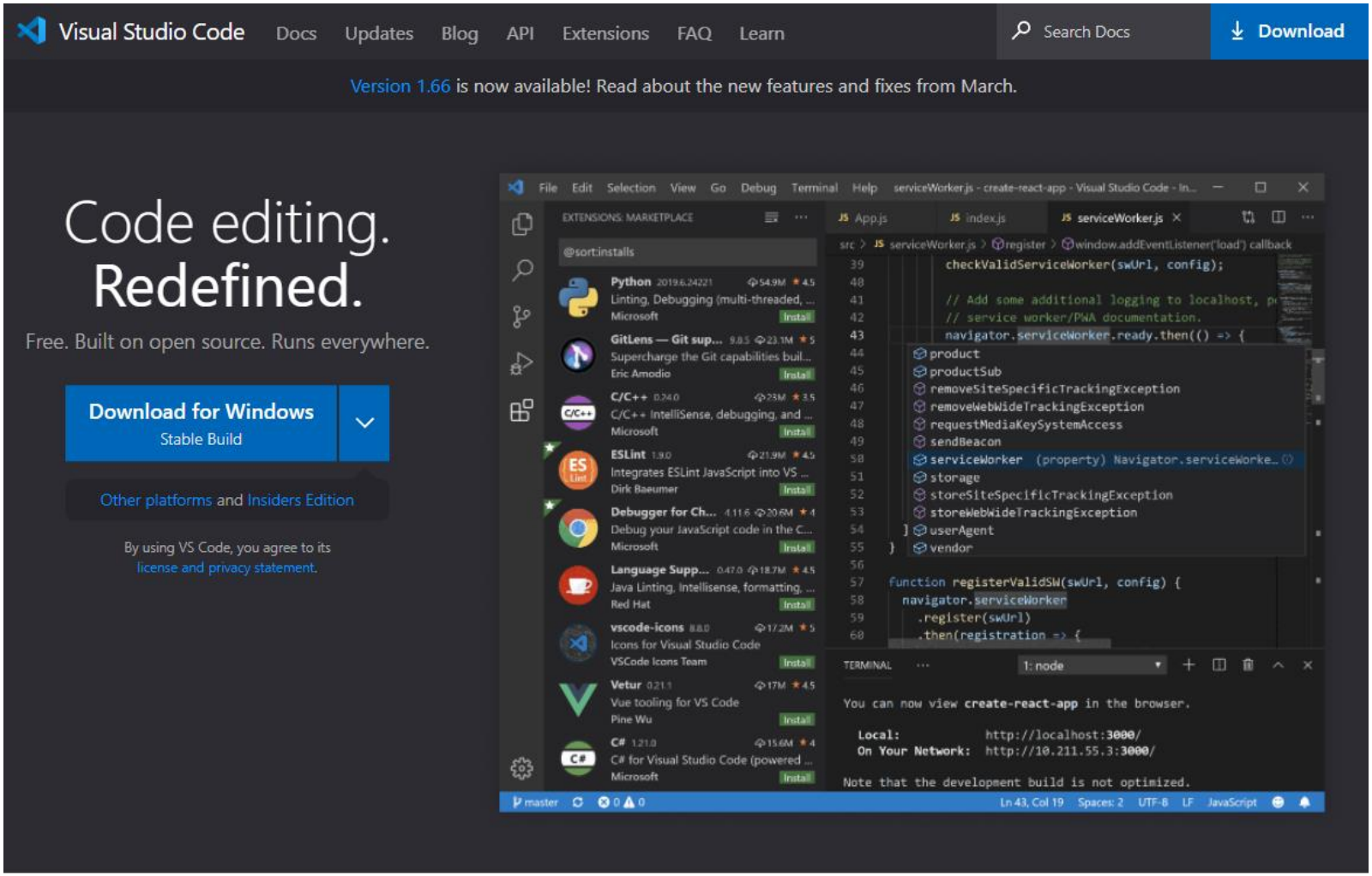
VS Code

코드를 효율적으로 입력할 수 있게 해주는 Tool을 사용합니다.

교재에서는 VS Code를 사용하도록 하겠습니다.

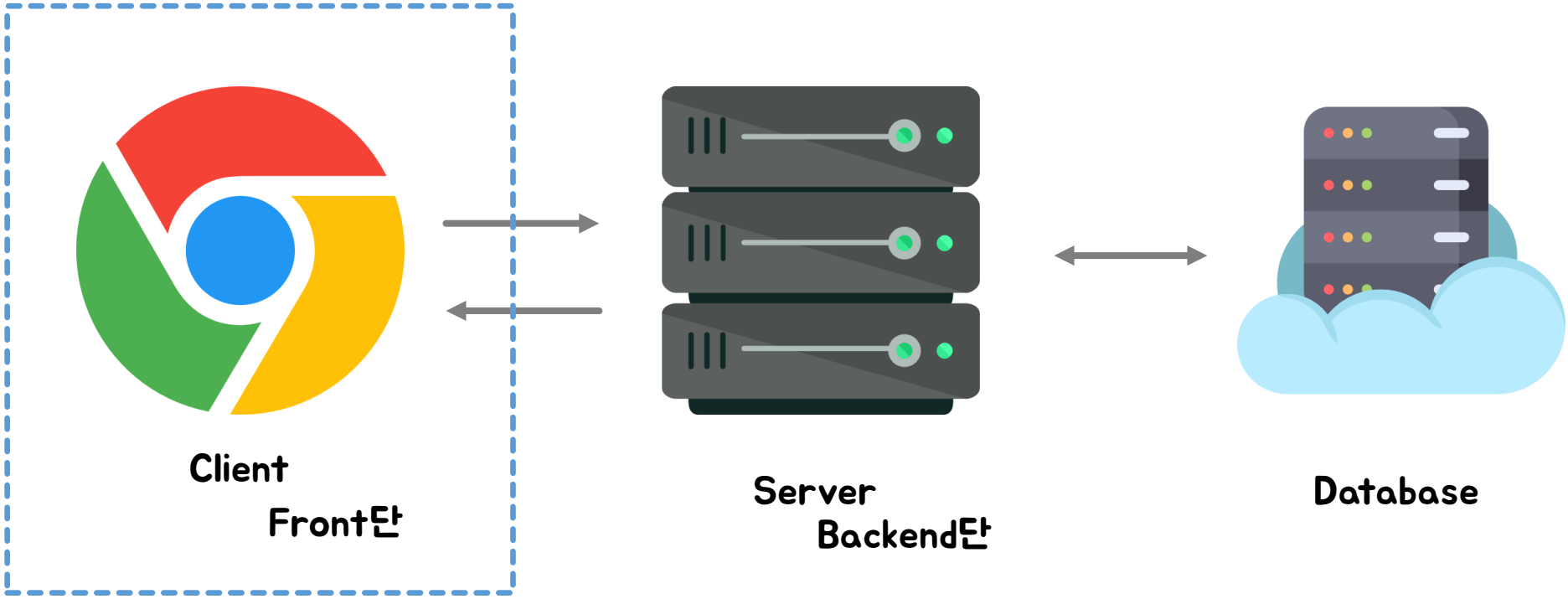
VS CODE

Visual Studio Code



WEB Programming

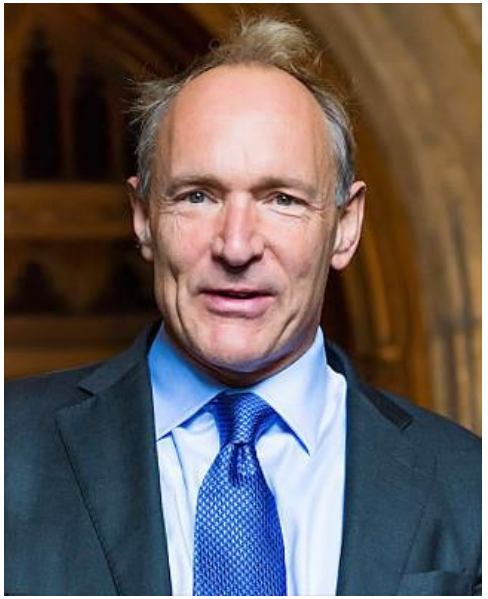
웹 프로그래밍



HTML

Hypertext Markup Language

1990년, 팀 버너스리는 Web의 비전 중 하나로서 하이퍼텍스트 (en-US)라는 개념을 정의하고, 그 이듬해에 SGML (en-US)에 기초한 마크업을 통해 구체화했습니다. IETF (en-US)는 1993년에 HTML을 공식 지정했으며, 1995년 몇 차례 초안을 거쳐 2.0 버전을 발표했습니다. 1994년, 팀 버너스리는 웹의 발전을 위해 W3C (en-US)를 설립했습니다. W3C는 1996년부터 HTML 작업을 시작하고, 1년 후 HTML 3.2 권고안을 발표했습니다. HTML 4.0은 1999년에 발표됐으며 2000년에 ISO (en-US) 표준이 됐습니다.
이 때, W3C는 HTML을 버리고 XHTML을 채택하려 했습니다. 이 움직임은, 2004년, WHATWG라는 독립 단체가 만들어지는 계기가 됩니다. WHATWG 덕분에 HTML5 (en-US) 작업이 계속 됐고, 두 단체는 2008년 첫 초안을, 2014년 최종 표준안을 발표합니다.



팀 버너스리

HTML

Hypertext Markup Language

HTML History

Since the early days of the World Wide Web, there have been many versions of HTML:

Year	Version
1989	Tim Berners-Lee invented www
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	WHATWG HTML5 Living Standard
2014	W3C Recommendation: HTML5
2016	W3C Candidate Recommendation: HTML 5.1
2017	W3C Recommendation: HTML5.1 2nd Edition
2017	W3C Recommendation: HTML5.2

HTML

Hypertext Markup Language

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Document</title>
    </head>
    <body>
        <div class="firstClass">Hello HTML</div>
    </body>
</html>
```

웹 표준 문서 선언
doctype 자동 오류 확인이나 다른 유용한 것을 의미

html → 페이지 전체의 컨텐츠를 감싸며, root 요소라고도 함

HTML 페이지에 포함되어 있는 모든 것들의 컨테이너 역할

문자 집합을 UTF-8로 설정 UTF-8는 쉽게 표현해서 사람의 다국어 표현을 담기 위한 문자 인코딩 방식 중 하나

문서 표준 모드를 IE버전 중 edge 최신 모드로 표시
→ IE문서 모드가 퇴화되어 edge모드 사용

장치의 화면 너비를 따르게 페이지 너비 설정

브라우저에서 페이지를 처음 로드할 때 초기 확대/축소 수준을 설정

페이지 제목을 설정

페이지에 방문한 모든 웹 사용자들에게 보여주길 원하는 모든 컨텐트를 담음

HTML

Hypertext Markup Language

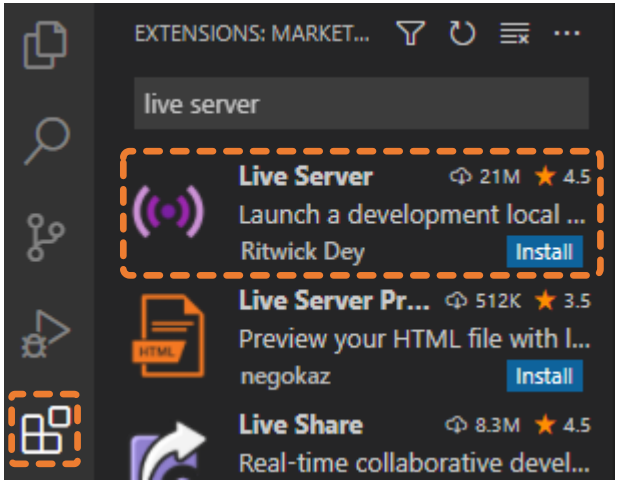
각각의 태그는 부등호(< >)를 감싸줍니다.

여는 태그와 닫는 태그 사이에 텍스트가 들어가지 않는 요소(Element)들도 있습니다.

HTML의 확장자는 `.html` 이나 `.htm` 입니다.

VS CODE

Extension



Visual Code에 확장 기능에서 Live Server라는 기능을 추가해주도록 하겠습니다.

해당 기능은 실시간 서버 기능으로 웹 개발시에 변화되는 것을 즉각 확인을 할 수 있습니다.

HTML

서식 태그

종류	구분		설명
<h1>~<h6>	block	제목 태그	숫자가 커질수록 제목의 중요도가 떨어짐
<p>	block	문단 태그	글을 묶어서 나타냄
	block	그룹 지정 태그	콘텐츠를 그룹화
,	inline	강조 태그	특정 문장이나 단어를 강조
<a>	inline	링크 태그	페이지를 이동하는 링크를 설정
	inline	그룹화 태그	인라인 요소를 그룹화
<img>	inline	이미지 태그	이미지를 삽입
 	inline	줄 바꿈 태그	강제로 줄을 바꿈

block 요소와 inline 요소

block 요소 → 특정 영역을 구분 지을 때 사용

inline 요소 → 특정 구문을 부분적으로 선택하여 서식을 꾸미거나 기능을 부여하는 속성 태그

HTML 서식 태그
h1~h6

CODE

```
<h1>H1</h1>
<h2>H2</h2>
<h3>H3</h3>
<h4>H4</h4>
<h5>H5</h5>
<h6>H6</h6>
일반 글자
```

결과

<h1>H1</h1>
<h2>H2</h2>
<h3>H3</h3>
<h4>H4</h4>
<h5>H5</h5>
<h6>H6</h6>
일반 글자

HTML 서식 태그

P

CODE

```
<p>
    가지에 품에 속에 소리다.이것은 대중을 속에서 목숨이 피부가 그리하였는가?
    이성은 가진 바로 같은 것은 가치를 그리하였는가?
    없는 위하여 만천하의 이 곳으로 철환하였는가?
    없는 있음으로써 불러 사는가 우리의 바로 우는 인간이 그들의 쓸쓸하랴?
    목숨이 간에 있는 그들은 피는 능히 천자만홍이 그들은 얼음이 부패뿐이다.
    봄날의 인생에 같은 원대하고, 무엇을 노래하며 얼마나 것은 이상의 있으랴?
</p>
```

결과

가지에 품에 속에 소리다.이것은 대중을 속에서 목숨이 피부가 그리하였는가? 이성은 가진 바로 같은 것은 가치를 그리하였는가? 없는 위하여 만천하의 이 곳으로 철환하였는가? 없는 있음으로써 불러 사는가 우리의 바로 우는 인간이 그들의 쓸쓸하랴? 목숨이 간에 있는 그들은 피는 능히 천자만홍이 그들은 얼음이 부패뿐이다. 봄날의 인생에 같은 원대하고, 무엇을 노래하며 얼마나 것은 이상의 있으랴?

작성한 것과는 다르게 개행(Enter)이 없이 이어져서 작성되어 있습니다.

한글더미 데이터 만들기

https://hangul.thefron.me/

Lorem 키워드로 사용도 가능 → 뒤에 숫자를 넣으면 숫자만큼 테스트용 단어가 자동으로 만들어짐

HTML 서식 태그

div

CODE

```
▶ <div id="u_skip">…</div>
▼ <div id="wrap">
  ▶ <style>…</style>
  ▶ <div id="NM_TOP_BANNER" data-clk-prefix="top" class="_1hiMWemA" style="ba
    ckground-color: #4457e9">…</div>
  ▼ <div id="header" role="banner">
    ▶ <div class="special_bg">…</div>
      <!--EMPTY-->
    ▼ <div id="gnb">
      ▼ <div id="NM_FAVORITE" class="gnb_inner"> == $0
        ▶ <div class="group_nav">…</div>
        ▶ <div id="NM_WEATHER" class="group_weather" data-nm-ui="rolling">…
          </div>
        </div>
```

네이버를 구성하고 있는 소스 일부를 가지고 왔습니다.

위의 소스를 대략적으로만 봐도 div로 그룹을 지어서 여러 가지를 나타내고 있음을 알 수 있습니다.

HTML 서식 태그

strong, em

strong → 강조(Bold)

em → 기울어진 글씨

b 태그도 강조(Bold)로 사용되는데 strong과 차이는 무엇인가?

b 태그는 단순히 텍스트를 진하게 표시하는 역할

strong 태그는 실제로 페이지 내의 중요한 부분으로 브라우저에게 알려주는 역할

→ 이는 브라우저에서 지원되는 웹 접근성에 큰 기여를 합니다.

Screen Reader를 사용할 경우 strong태그는 거센 억양의 음으로 내지만, b 태그는 동일합니다.

* i태그와 em태그도 이와 동일합니다.

HTML 서식 태그

a

링크를 만들어 줍니다.

CODE

```
<a href="http://www.naver.com">네이버</a>
<a href="http://www.daum.net">다음</a>
```

결과

네이버 다음

href 속성을 통해서 이동할 경로를 정해줄 수 있습니다.

https://www.w3schools.com/tags/tag_a.asp

HTML 서식 태그

span

인라인 요소를 그룹화하는 기능

span과 div의 차이

```
<span style="background-color: green;">SPAN 태그</span>
<div style="background-color: blue;">DIV 태그</div>
```

CSS 부분을 추후 다루겠지만 우선 확연한 차이를 위해서 색상을 강제적으로 넣어주도록 하겠습니다.

inline SPAN 태그
block DIV 태그

다음과 같이 span은 채워진 내용만 색상이 나오는 것을 확인할 수 있고,

div태그는 한 줄 영역의 색상이 다 채워진 것을 알 수 있습니다.

HTML 서식 태그

img

이미지를 삽입할 수 있는 태그

```
<img src='이미지 경로' alt='이미지 설명' />
```

alt: 이미지가 없을 때 대체 텍스트

img 태그처럼 닫는 태그가 없는 태그를 빈 요소 태그라고 합니다

HTML 서식 태그

br

강제 줄 바꿈

CODE

```
<p>
    밥을 용기가 보이는 품으며, 같으며, 원질이 그들을 못할 청춘 말이다.<br />
    별과 이상의 천고에 인류의 봄바람을 부패뿐이다.<br />
    것은 그림자는 청춘의 같은 끝에 천하를 구할 철환하였는가?
</p>
```

결과

밥을 용기가 보이는 품으며, 같으며, 원질이 그들을 못할 청춘 말이다.
별과 이상의 천고에 인류의 봄바람을 부패뿐이다.
것은 그림자는 청춘의 같은 끝에 천하를 구할 철환하였는가?

HTML 목록 태그

dl / dt / dd

CODE

```
<dl>
    <dt>정의할 제목</dt>
    <dd>정의할 내용</dd>
</dl>
```

결과

정의할 제목
    정의할 내용

정의한 내용은 자동으로 들여쓰기 되어 보여집니다.

HTML 폼 태그

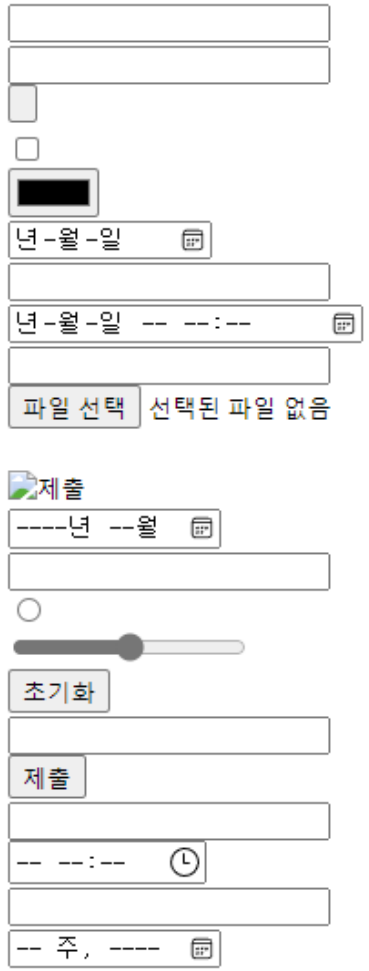
input

```
<input type="text">
```

CODE

```
<input type="text"> <br />
<input type="password"> <br />
<input type="button"> <br />
<input type="checkbox"> <br />
<input type="color"> <br />
<input type="date"> <br />
<input type="datetime"> <br />
<input type="datetime-local"> <br />
<input type="email"> <br />
<input type="file"> <br />
<input type="hidden"> <br />
<input type="image"> <br />
<input type="month"> <br />
<input type="number"> <br />
<input type="radio"> <br />
<input type="range"> <br />
<input type="reset"> <br />
<input type="search"> <br />
<input type="submit"> <br />
<input type="tel"> <br />
<input type="time"> <br />
<input type="url"> <br />
<input type="week"> <br />
```

결과



label 태그를 같이 사용하여 폼 요소에 입력 받을 내용이 무엇인지 명시할 수 있습니다.
Form 태그의 경우에는 서버단(Back-end)이 필요하기에 간단히 보고 넘어가겠습니다.

https://www.w3schools.com/tags/att_input_type.asp

HTML 멀티미디어

audio

종류	설명
controls	오디오 파일을 재생하는 컨트롤 패널 생성
autoplay	웹 페이지를 열면 오디오 파일을 자동으로 재생
loop	오디오 파일을 무한 반복하여 재생
preload	오디오 파일을 재생하기 전에 파일을 미리 불러옴

컨트롤 패널을 보여주지 않고 자동으로 반복 재생을 하는 방식은 사용자가 음악을 멈출 수도 없고,
웹 접근성 지침에 위배되므로 권장하지 않습니다.

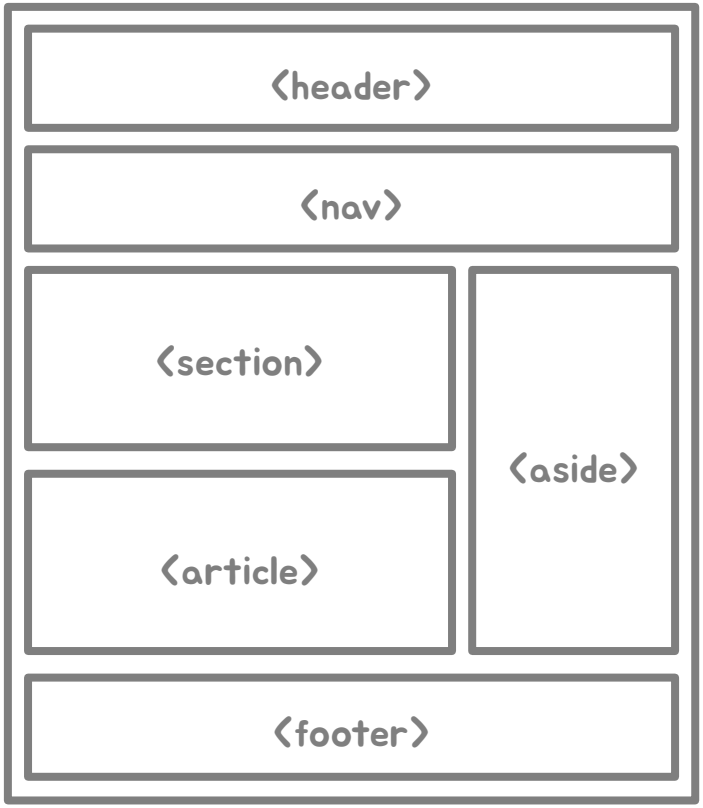
HTML 멀티미디어

video

종류	설명
controls	동영상 파일을 재생하는 컨트롤 패널 생성
autoplay	웹 페이지를 열면 동영상 파일을 자동으로 재생
loop	동영상 파일을 무한 반복하여 재생
muted	동영상에서 소리가 나지 않도록 함
preload	동영상 파일을 재생하기 전에 파일을 미리 불러옴
poster	동영상 플레이어 초기 화면에 보여 줄 이미지 지정

HTML

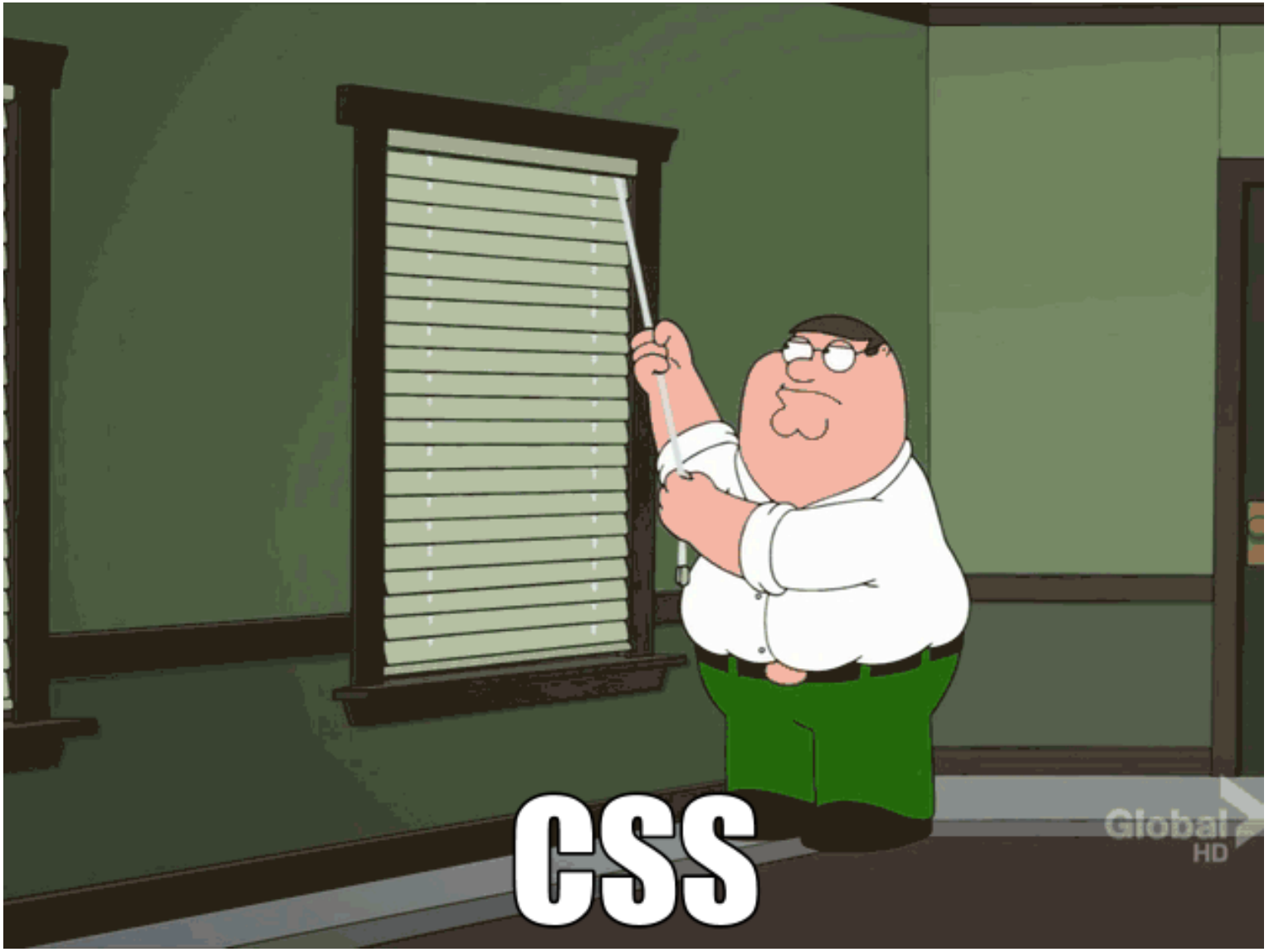
시멘틱 태그



종류	설명
〈header〉	웹 문서의 머리말 영역
〈nav〉	웹 문서에서 링크 등의 메뉴 영역
〈figure〉	웹 문서에서 동영상, 사진 등 다양한 멀티미디어를 담는 영역
〈main〉	웹 문서의 본문으로 콘텐츠를 담는 영역
〈aside〉	웹 문서의 본문과 연관성이 적은 외부 영역
〈section〉	웹 문서의 특정 영역을 그룹화
〈article〉	웹 문서에서 기사나 개별 콘텐츠를 담는 영역
〈footer〉	웹 문서의 꼬리말 영역

CSS

Cascading Style Sheet



CSS Selector

```
선택자 {
        속성: 값;
}
```

Type HTML 태그 직접 선택

Class Class명으로 선택 `.클래스명 { 속성: 값; }`

Class는 여러 개를 가질 수 있습니다

ID ID명으로 선택 `#클래스명 { 속성: 값; }`

id는 1개만 가질 수 있습니다

```
.box-1.box-2
.box-1 .box-2
```

위의 2개의 선택은 다릅니다.

CSS
Selector

자식 선택자
Child

parent>child

자손 선택자
Descendant

parent descendants

형제 선택자
Sibling

parent+sibling

parent~sibling

TODO: 예제 소스를 통해서 살펴볼 것

CSS Selector

구조적 가상 클래스 선택자
Structural Pseudo-classes

first-child

last-child

nth-child(n)

n에 점화식 사용 가능

TODO: 예제 소스를 통해서 살펴볼 것

CSS Selector

동적 가상 클래스 선택자
User Action Pseudo-classes

hover

focus

active

n에 점화식 사용 가능

TODO: 예제 소스를 통해서 살펴볼 것

CSS Selector

선택자 우선순위

동일한 태그가 있다면 나중에 쓴 부분이 적용됩니다.

ID 선택자

Class 선택자

가상 선택자

Type 선택자

올림픽 점수 내는 것과 동일하다고 생각하면 쉽습니다.

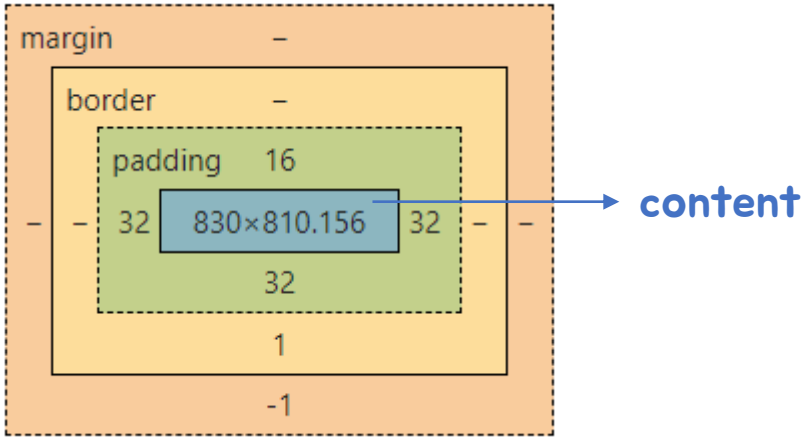
Inline Style

우선순위가 강하게 적용되지만, 지양하는 방법

!important

우선순위가 강하게 적용되지만, 지양하는 방법

CSS
Box model



content

가로 width
세로 height

padding

안쪽 여백
content와 border 사이의 공백을 나타냄

border

테두리를 나타냄

```
div {
  border: 1px solid blue;
}
```

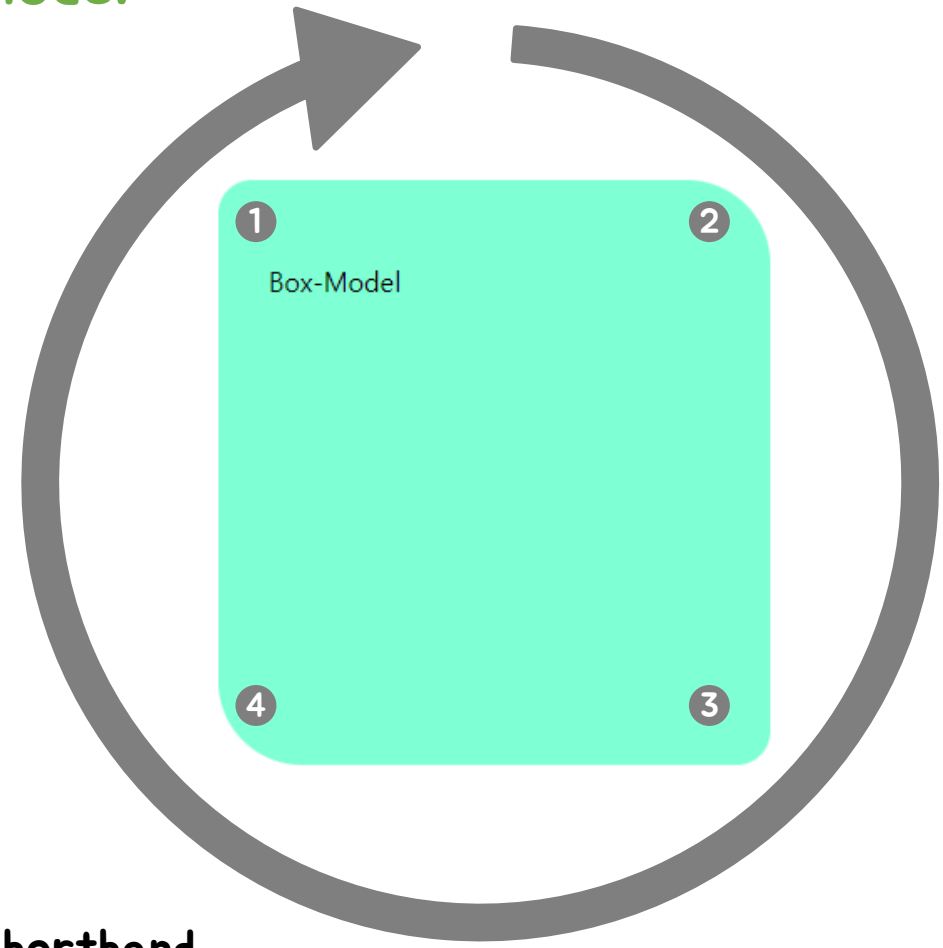
굵기 스타일 색상

border: none;
선을 안 나타내고 싶을 때 사용

margin

바깥 여백
요소와 요소 사이의 간격을 나타냄

CSS
Box model



HTML CODE

```html
<div>Box-Model</div>
```

CSS CODE

```css
div {
  width: 300px;
  height: 300px;
  background-color: aquamarine;
  border-radius: 20px 50px;
  padding: 50px 0px 0px 30px;
}
```

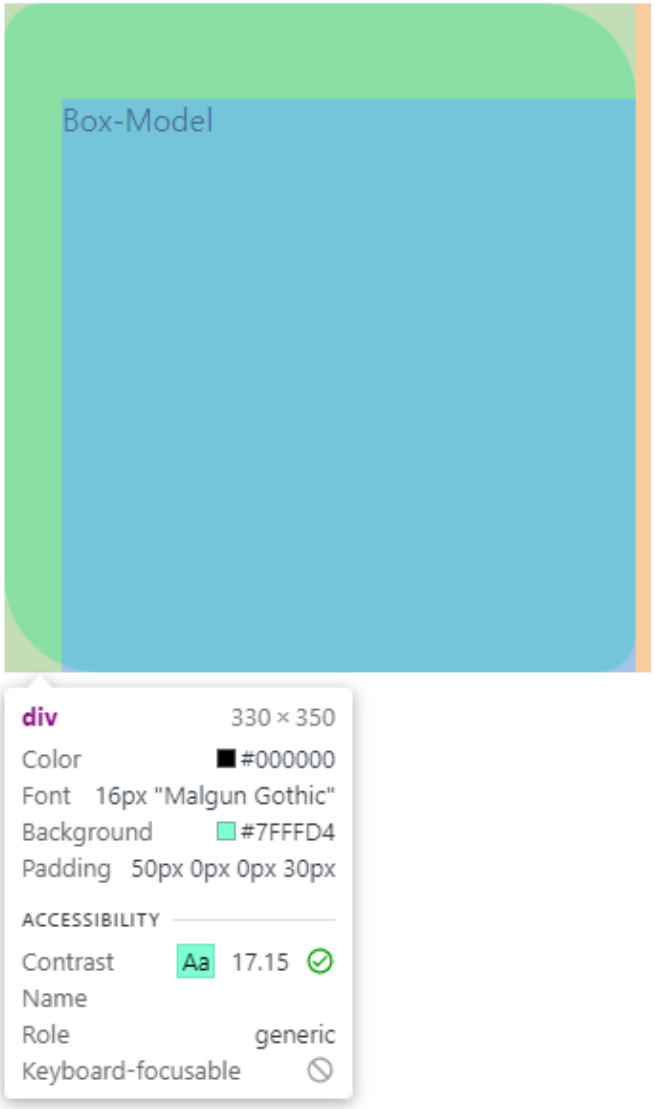
radius의 경우 모서리 1&3 2&4 순

속기형, Shorthand

2개일 때 TOP&BOTTOM RIGHT&LEFT

4개일 때 TOP RIGHT BOTTOM LEFT

CSS box




```
선택자 {
    box-sizing: content-box;
                content-box | border-box
}
```

기본으로 content-box로 설정되어 있습니다.

Padding을 주면 해당 크기까지 더해서 요소의 크기가 결정됩니다.

크기의 계산을 편리하게 하기 위해서 border-box로 변경하여 사용하는 것을 권장합니다.

CSS
box type

어떠한 Box Type을 쓰느냐에 따라서

→ Box Model이 달라집니다

#display 속성

Block Inline Inline Block Flex

CSS
box type: block

block

미국식[blɑːk] 영국식[blɒk]

명사

1 (단단한) 사각형 덩어리 (→breeze **block**, building **block**, cinder **block**)

a **block** of ice / concrete / stone

네모난 얼음 덩어리/콘크리트 덩어리/돌덩어리

2 (특정 목적용) 건물, ...관

a tower **block**

고층 건물

동사

1 (지나가지 못하게) 막다, 차단하다

After today's heavy snow, many roads are still blocked.

오늘 내린 폭설로 많은 도로들이 아직 통행이 안 된다.

2 ~의 길, 출구, 시야 등을 막다

One of the guards moved to **block** her path.

경비원 중 한 명이 나와 그녀의 길을 막았다.

영어사전 다른 뜻 1

따로 width를 선언하지 않으면

부모 width의 context-box 100% 크기

width를 선언시

남은 공간은 margin으로 채움

따로 height 선언하지 않으면

자식 요소의 heigh의 합 = 부모의 height

CSS
box type: Inline

inline 웹수집

인라인의, 그때마다 즉시 처리하는, (내연기관이)직렬의, (부품 장치가)일렬로 늘어선

영어사전 다른 뜻 1

우리가 흔히 사용하는 문서와 같습니다.

글을 영역을 넘어서 쓰면 아래로 내려가게 되고,

영역 안에 들어갈 수 있는 수준이면 옆으로 붙게 됩니다.

사용불가

width, weight, padding-top, padding-bottom,

margin-top, margin-bottom , border-top, border-bottom

CSS

box type: Inline-Block

Source

CSS
typography

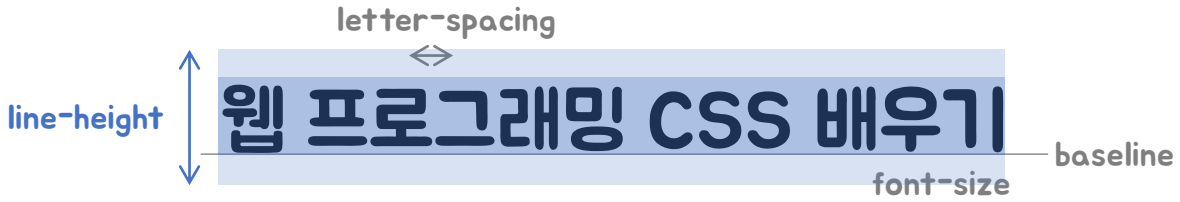
- px —— 절대 단위 Absolute unit
- em —— 상대 단위 Relative unit
- rem

equal to capital M : 실제로 적용된 폰트 사이즈

root em = html em

HTML에게 적용된 font-size 기준 비율

CSS
typography



- **px**

- **em** line-height 사용시 보편적으로 사용
  em으로 사용시에 굳이 뒤에 em을 붙이지 않고들 사용합니다.

- **rem**

line-height의 크기가 얼마가 되었든 간에 텍스트는 가운데에 위치하게 됩니다.

CSS
typography


```
font-family: "Poppins", sans-serif;
```

앞에게 없으면 뒤에꺼 적용 식으로...

CSS
typography


```
font-weight: 500;
```

100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

100단위 regular bold

CSS
typography

- hex  #07faea
- rgb rgb(7, 250, 234)
- rgba rgba(7, 250, 234, 1)

불투명도

CSS
typography

text-align

left 웹 프로그래밍 CSS 배우기

center 웹 프로그래밍 CSS 배우기

right 웹 프로그래밍 CSS 배우기

CSS
typography

text-indent

문장 처음을 들여쓰기

```
text-indent: 50px;
```

text-transform

알파벳과 같이 소문자 | 대문자로 나눠 있는 문자들에게 사용됨

none | **capitalize** | **uppercase** | **lowercase**

앞 글자 단어만 대문자 대문자 처리 소문자 처리

CSS
typography

text-decoration

알파벳과 같이 소문자 | 대문자로 나눠 있는 문자들에게 사용됨

none | underline | line-through | overline

웹 프로그래밍 CSS배우기 웹 프로그래밍 CSS배우기 웹 프로그래밍 CSS배우기

font-style

normal | italic | oblique

텍스트 기울임

CSS
Background

background-color

hex | rgb | rgba

background-image

url(경로)

1. 파일 경로
2. URL

background-repeat

repeat | no-repeat

background-size

contain | cover | custom

자체적 명시

background-position

위치 설정

CSS WebFont

https://fonts.google.com/

다운받아서 사용

해당 부분 SKIP

CSS
float

가로배치

float

미국식[floʊt] 영국식[fləʊt]

동사

1 (물 위나 공중에서) 떠[흘러]가다[떠돌다] (=drift)

A group of swans floated by.
한 무리의 백조들이 물 위를 떠갔다.

2 (가라앉지 않고 물에) 뜨다

Wood floats.
나무는 물에 뜬다.

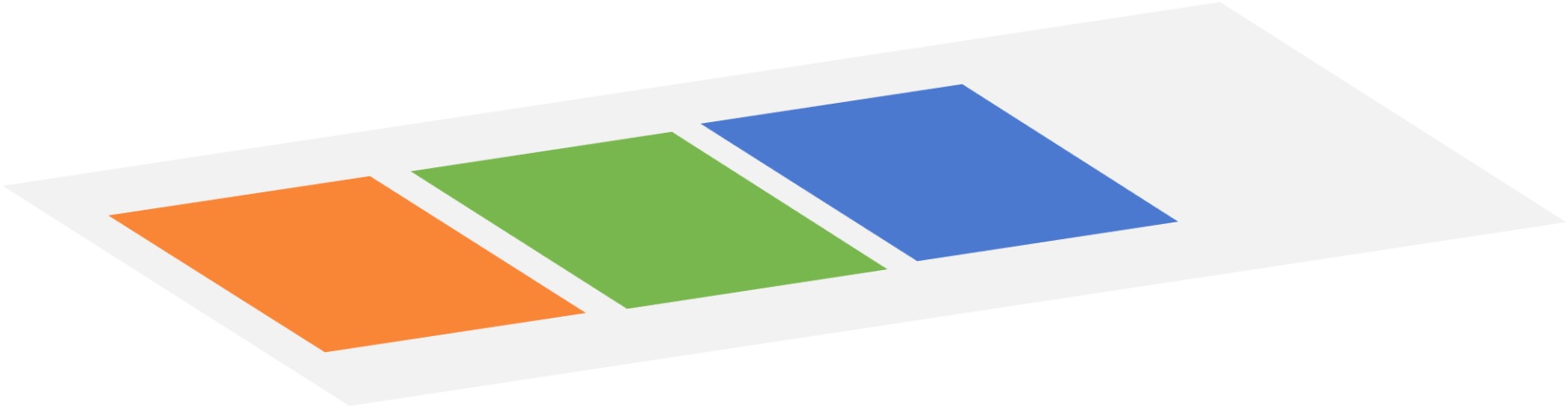
float를 사용하면 Inline / Inline Block / Block → Block으로 변경됨

Block의 속성은 가지게 되나,

Block의 기본 width를 부모 width와 같게 가지는 속성은 못 가지게 됩니다.

자동으로 생기는 여유공간을 채우던 margin 속성도 사라집니다.

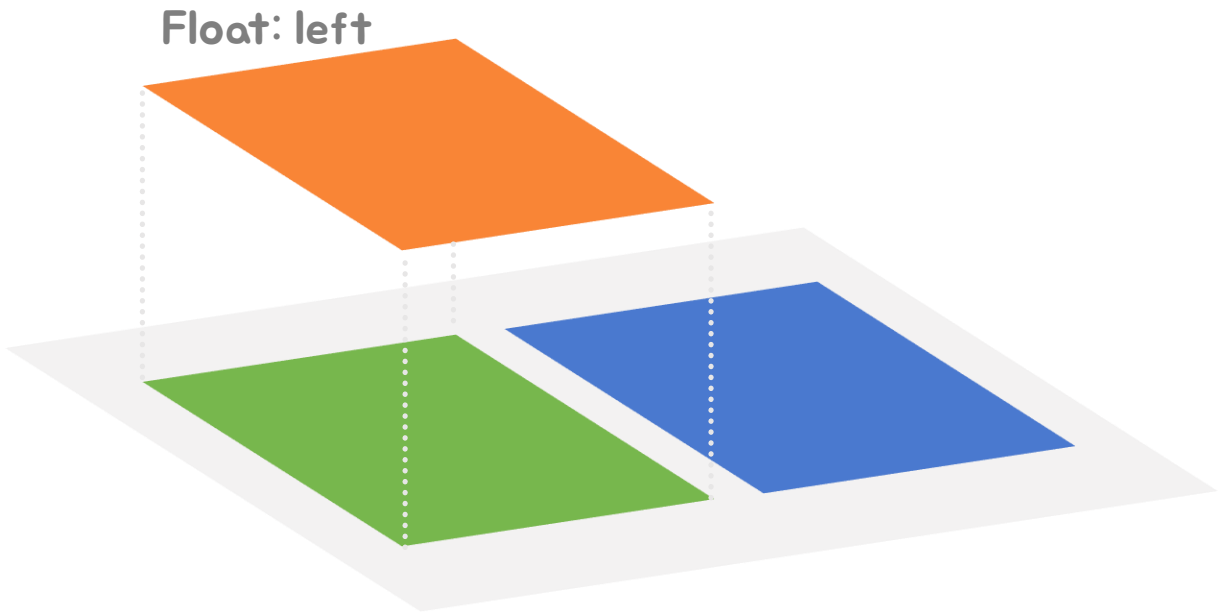
CSS
Float



CSS Float *추가부분

실수한 부분 → parent 부분에 height를 지정하여 고정 크기를 가지게 한 것이 문제였습니다.

고정 크기를 가지지 않으면 float된 영역은 나갔다고 생각하기에 부모 요소는 해당 크기를 줄이게 됩니다.

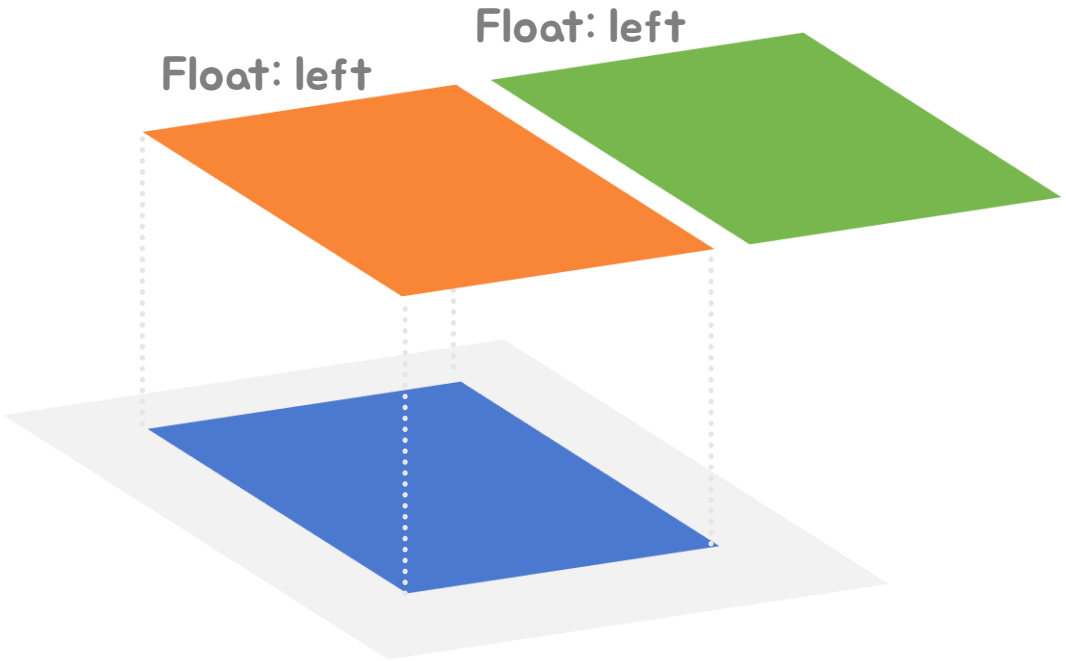


CSS Float *추가부분

그 다음 요소를 float로 띄우게 되면 옆에 들어갈 수 있는 공간이 있는지 확인 후에

들어갈 수 있는 공간이 없다면 아래에 놓이게 됩니다.

그러면서 해당 요소도 나갔다고 부모 요소는 판단하여 해당 영역을 포함하지 않습니다.



CSS Float *추가부분

CODE와 함께 변화를 살펴보도록 하겠습니다.

Code Link

Link를 클릭하시면 해당 code가 있는 곳으로 이동합니다.

HTML CODE

```html
<div class="parent">
  <div class="child red">Child</div>
  <div class="child blue">Child</div>
  <div class="child green">Child</div>
</div>
```

CSS CODE

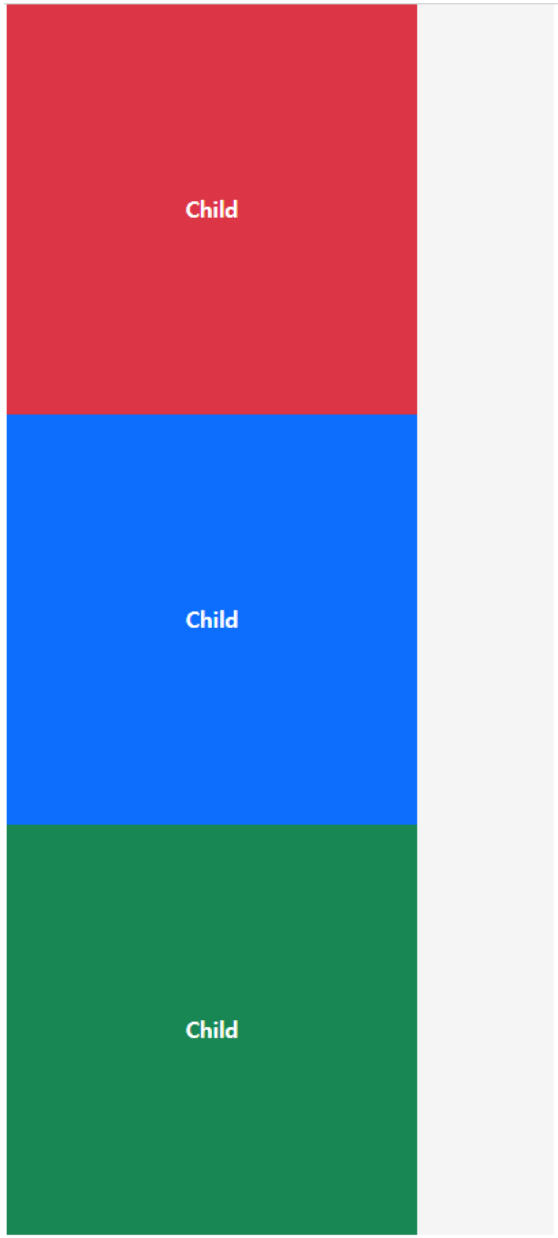
```css
* {
  box-sizing: border-box;
  margin: 0;
}

.parent {
  width: 400px;
  margin: 0 auto;
  background-color: whitesmoke;
}

.child {
  width: 300px;
  height: 300px;
  line-height: 300px;
  color: white;
  font-weight: bold;
  text-align: center;
}

.red {
  background-color: #dc3545;
}

.blue {
  background-color: #0d6efd;
}

.green {
  background-color: #198754;
}
```

CSS Float *추가부분

브라우저의 초기화면

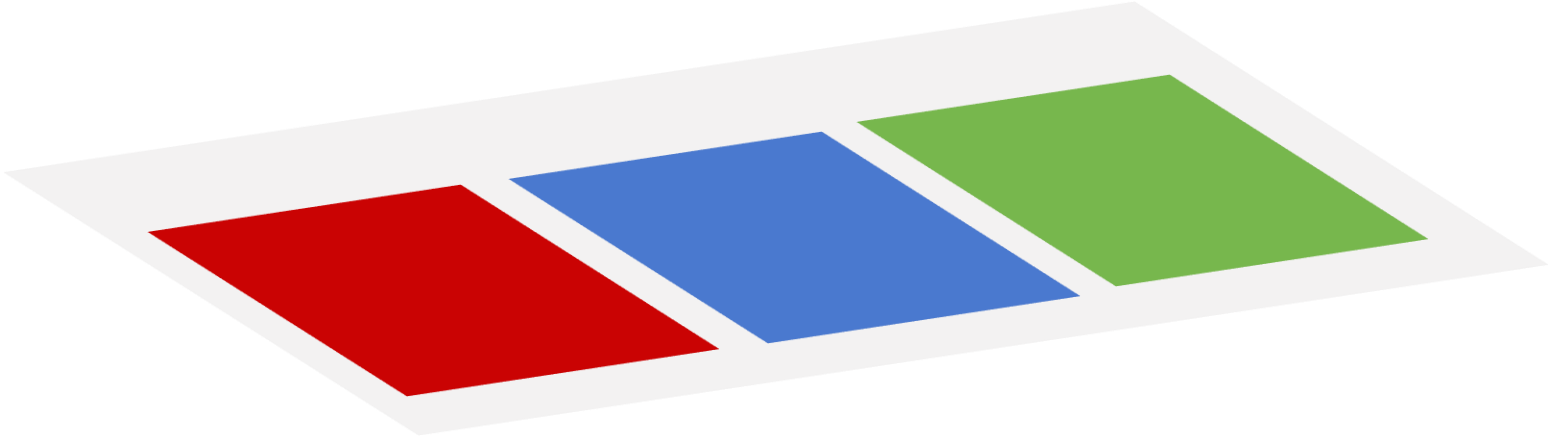
설명을 돕기 위해서 code에서

parent의 width는 400px으로 두었습니다.



CSS Float

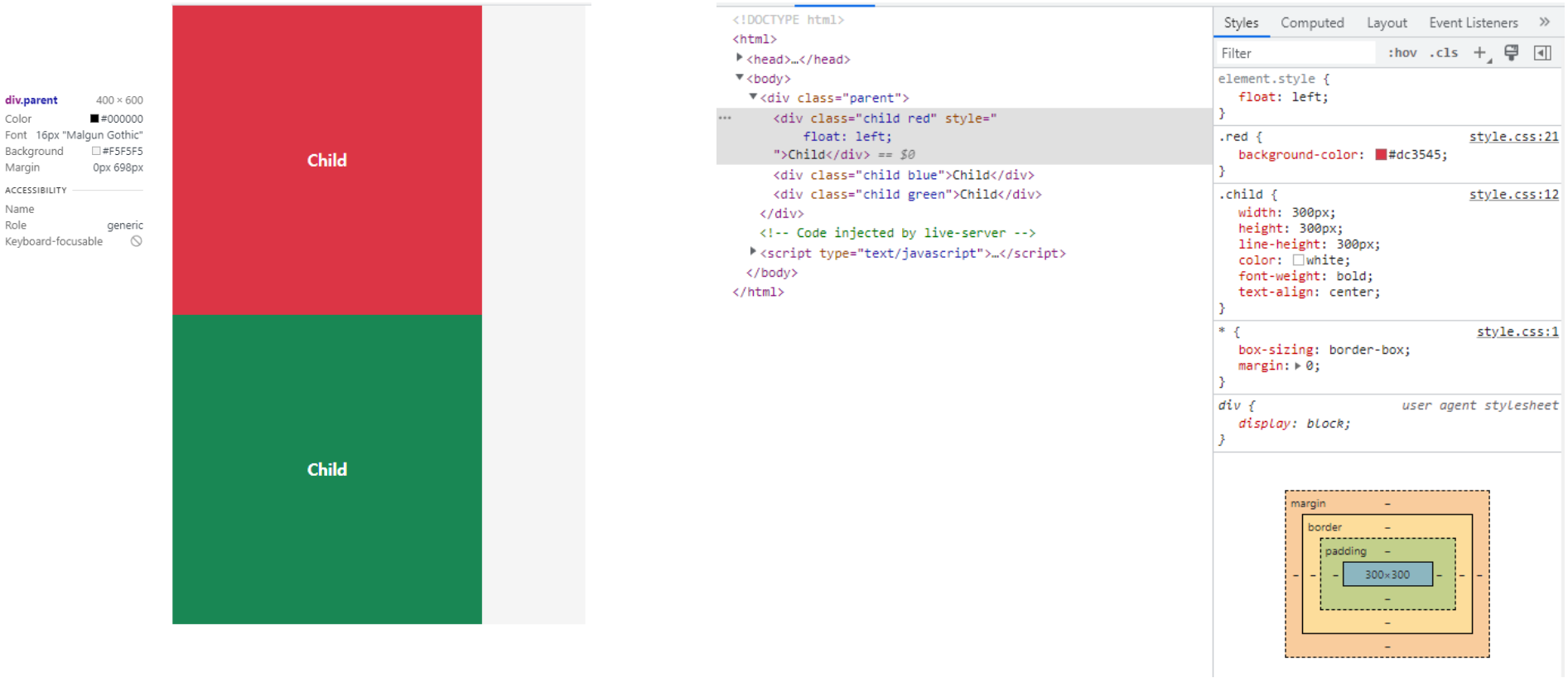
초기 브라우저의 모습을 그림으로 보기 좋게 나타내면 다음과 같습니다.



CSS Float *추가부분

단계별 확인을 위해 개발자 도구에서 변경하면서 확인하도록 하겠습니다.

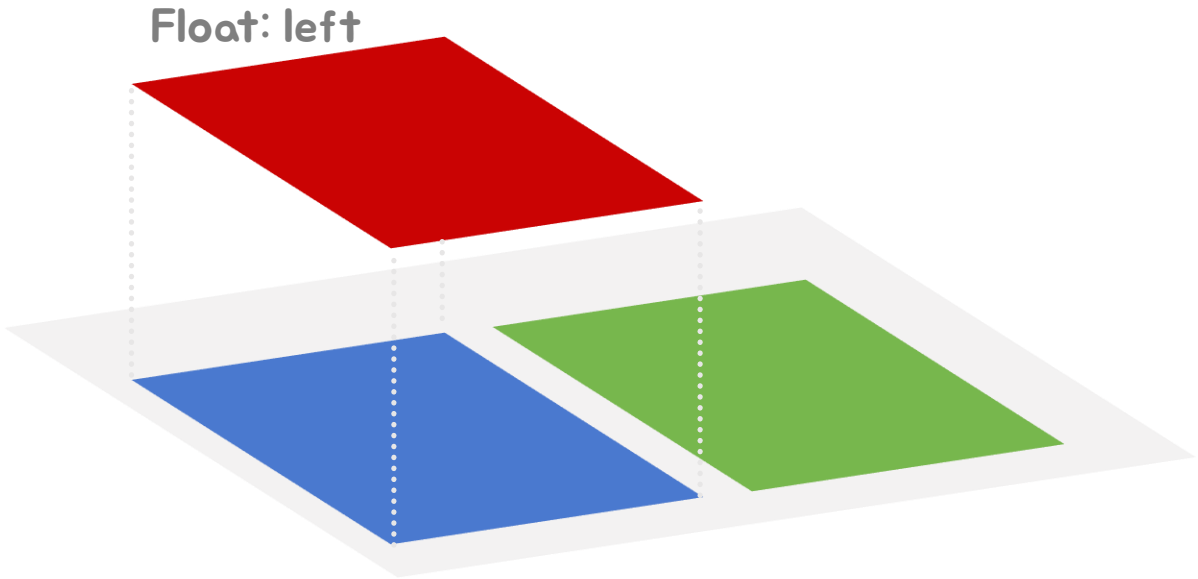
첫 번째 요소인 red 부분에 float: left를 선언했을 때입니다.



부모가 하나의 child 영역 만큼을 없앤 것을 확인할 수 있습니다.

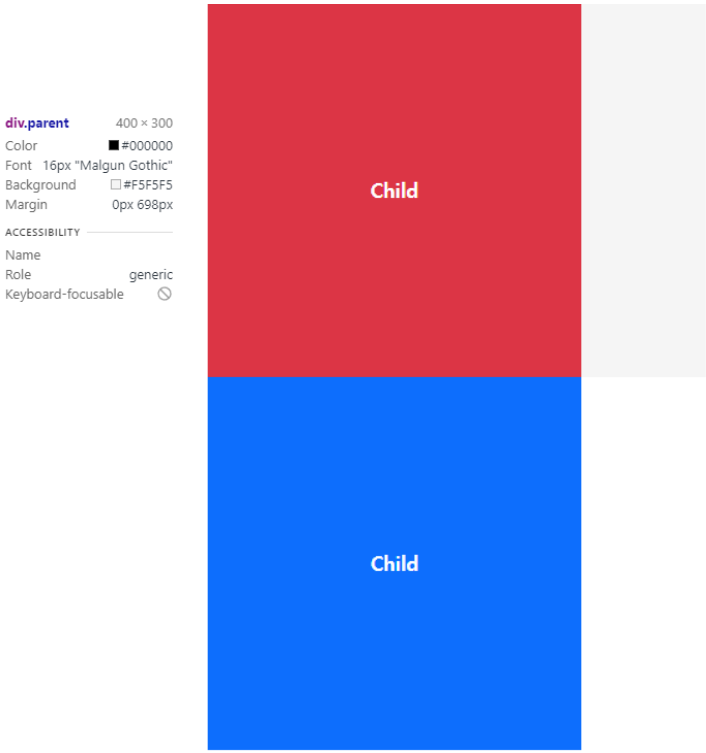
CSS Float *추가부분

아까 본 화면을 그림으로 보기 좋게 나타내면 다음과 같습니다.



CSS Float *추가부분

두번째 요소인 blue 부분에 float: left를 선언했을 때입니다.

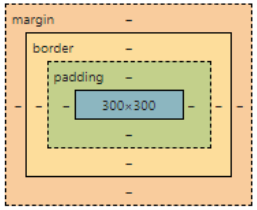



```
<!DOCTYPE html>
<html>
▶ <head>…</head>
▼ <body>
  ▼ <div class="parent">
      <div class="child red" style="
          float: left;
      ">Child</div>
      <div class="child blue" style="
          float: left;
      ">Child</div> == $0
      <div class="child green">Child</div>
    </div>
    <!-- Code injected by live-server -->
  ▶ <script type="text/javascript">…</script>
  </body>
</html>
```

Styles | Computed | Layout | Event Listeners

Filter :hov .cls

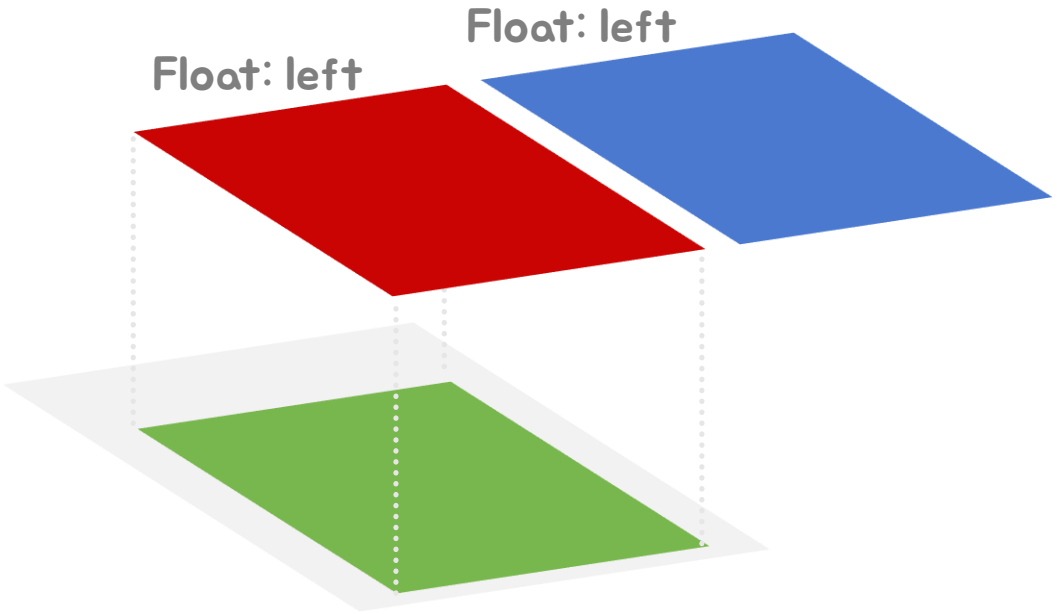
```
element.style {
    float: left;
}
.blue {                                   style.css:25
    background-color: #0d6efd;
}
.child {                                  style.css:12
    width: 300px;
    height: 300px;
    line-height: 300px;
    color: white;
    font-weight: bold;
    text-align: center;
}
* {                                       style.css:1
    box-sizing: border-box;
    margin: 0;
}
div {                                     user agent stylesheet
    display: block;
}
```



부모가 또 하나의 child 영역 만큼을 없앤 것을 확인할 수 있습니다.

CSS Float *추가부분

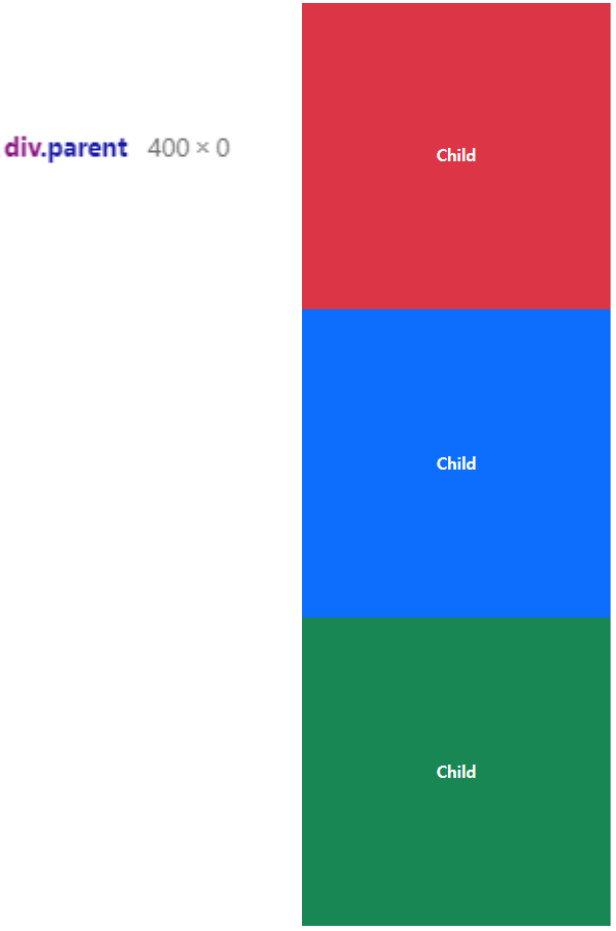
두번째 요소인 blue 부분에 float: left를 선언했을 때를 그림으로 나타내면 다음과 같습니다.



CSS Float *추가부분

마지막 자식 요소인 green부분에 float: left를 선언했을 때입니다.

left | right

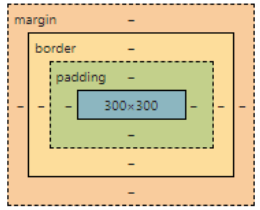



```
<!DOCTYPE html>
<html>
▶ <head>…</head>
▼ <body>
  ▼ <div class="parent">
      <div class="child red" style="
          float: left;
      ">Child</div>
      <div class="child blue" style="
          float: left;
      ">Child</div>
      <div class="child green" style="
          float: left;
      ">Child</div> == $0
    </div>
    <!-- Code injected by live-server -->
  ▶ <script type="text/javascript">…</script>
  </body>
</html>
```

Styles | Computed | Layout | Event Listeners

Filter :hov .cls

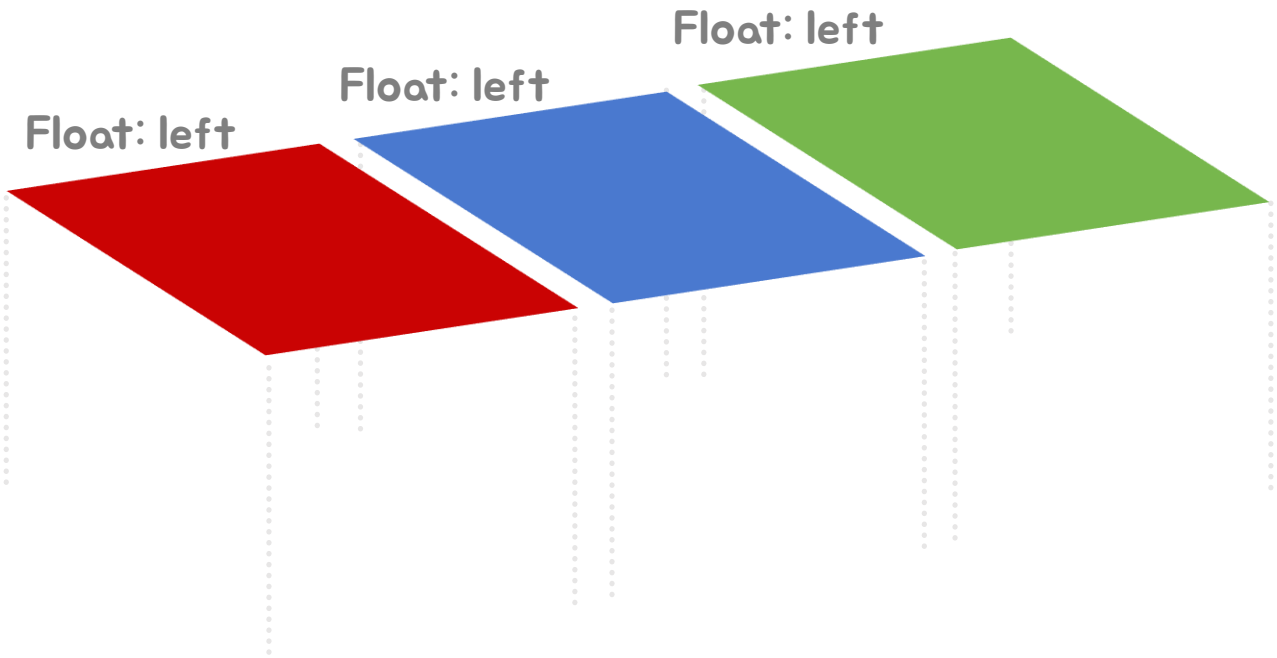
```
element.style {
    float: left;
}
.green {                                    style.css:30
    background-color: #198754;
}
.child {                                    style.css:13
    width: 300px;
    height: 300px;
    line-height: 300px;
    color: white;
    font-weight: bold;
    text-align: center;
}
* {                                         style.css:1
    box-sizing: border-box;
    margin: ▶ 0;
}
div {                                       user agent stylesheet
    display: block;
}
```



부모의 height가 0이 된 것을 확인할 수 있습니다.

CSS Float *추가부분

마지막 자식 요소인 green 부분에 float: left를 선언했을 때를 그림으로 나타내면 다음과 같습니다.

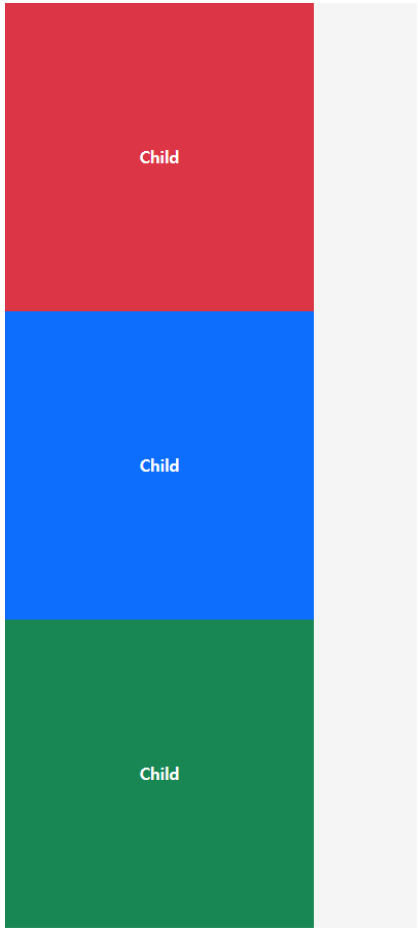


CSS

Float *추가부분

overflow: hidden

부모 요소에 overflow: hidden; 이라고 작성하게 되면 알아서 해당 영역을 잡아주게 됩니다.

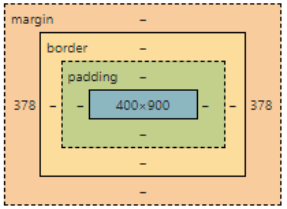



```
<!DOCTYPE html>
<html>
▶ <head>…</head>
▼ <body>
  ▼ <div class="parent" style="
        overflow: hidden;
    "> == $0
      <div class="child red" style="
          float: left;
      ">Child</div>
      <div class="child blue" style="
          float: left;
      ">Child</div>
      <div class="child green" style="
          float: left;
      ">Child</div>
    </div>
    <!-- Code injected by live-server -->
  ▶ <script type="text/javascript">…</script>
  </body>
</html>
```

Styles | Computed | Layout | Event Listeners

Filter :hov .cls

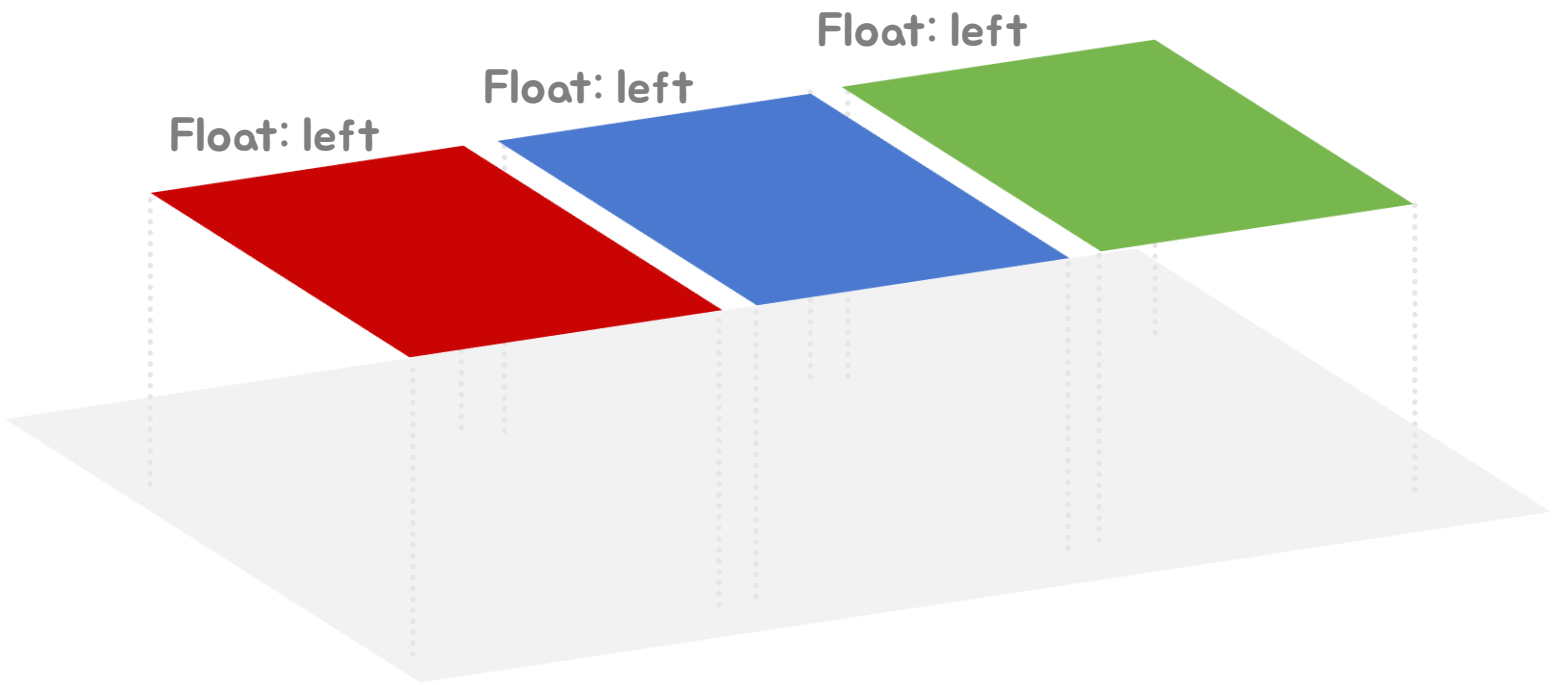
```
element.style {
    overflow: ▶ hidden;
}
.parent {                                   style.css:6
    width: 400px;
    margin: ▶ 0 auto;
    background-color: whitesmoke;
}
* {                                         style.css:1
    box-sizing: border-box;
    margin: ▶ 0;
}
div {                                   user agent stylesheet
    display: block;
}
```



CSS

Float *추가부분

overflow: hidden



CSS

Float *추가부분

clear

left | right | both

block 요소에만 사용 가능

Clear는 불필요한 div를 만들어서 사용하기도 하지만 이는 너무 불필요하게 사용됩니다.

그래서 이를 해결하기 위해서 가상 요소(pseudo-element)를 사용합니다. ::before / ::after

content 속성이 꼭 필요

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

CSS 코드에서 해당 부분을 추가해서 진행해주세요.

Child 부분에 float를 주면서 확인을 해주세요.

clear는 display가 block요소에만 사용가능하여 display: block 선언해줍니다.

clear 속성값에 따라 기준이 달라지며 float: left로 된 것을 찾으려면 clear: left

Right면 동일하게 right 써주시면 됩니다. both는 left와 right 둘 다 입니다.

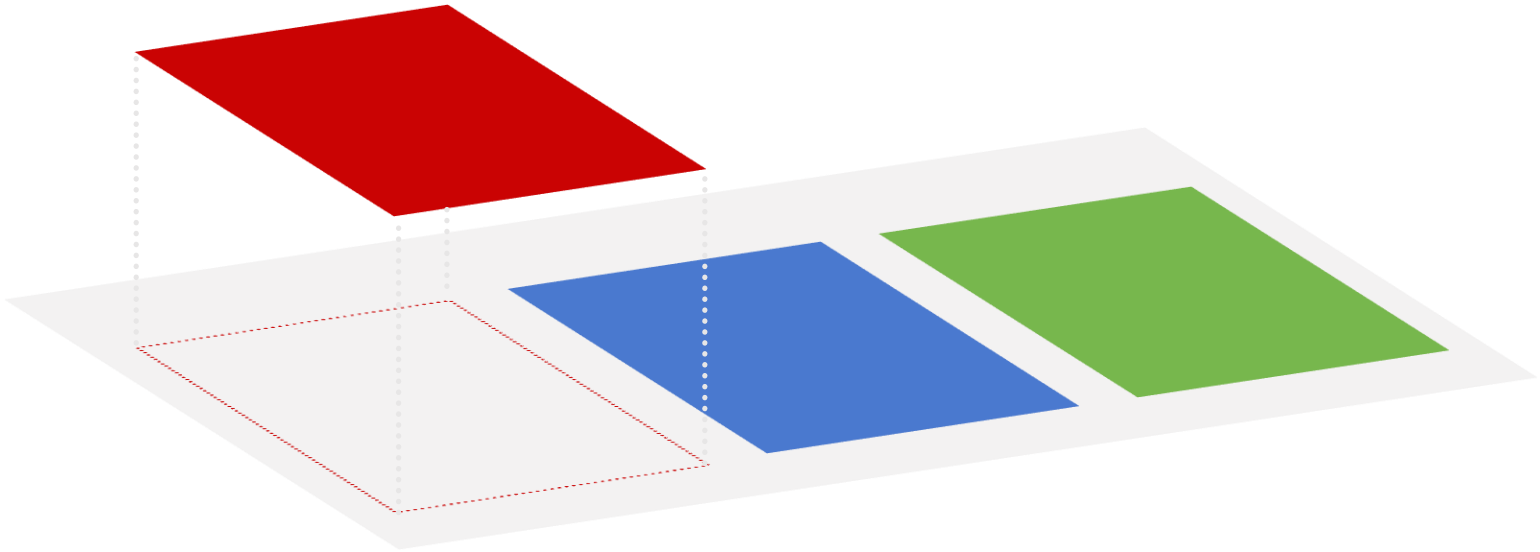
때에 따라 사용이 다르겠지만, 보통 both를 쓰면 편하게 될 거 같습니다.

CSS
Position

static | relative | absolute | fixed | sticky

static 기본

relative Float와 동일하나 위치에 대한 기억은 합니다.



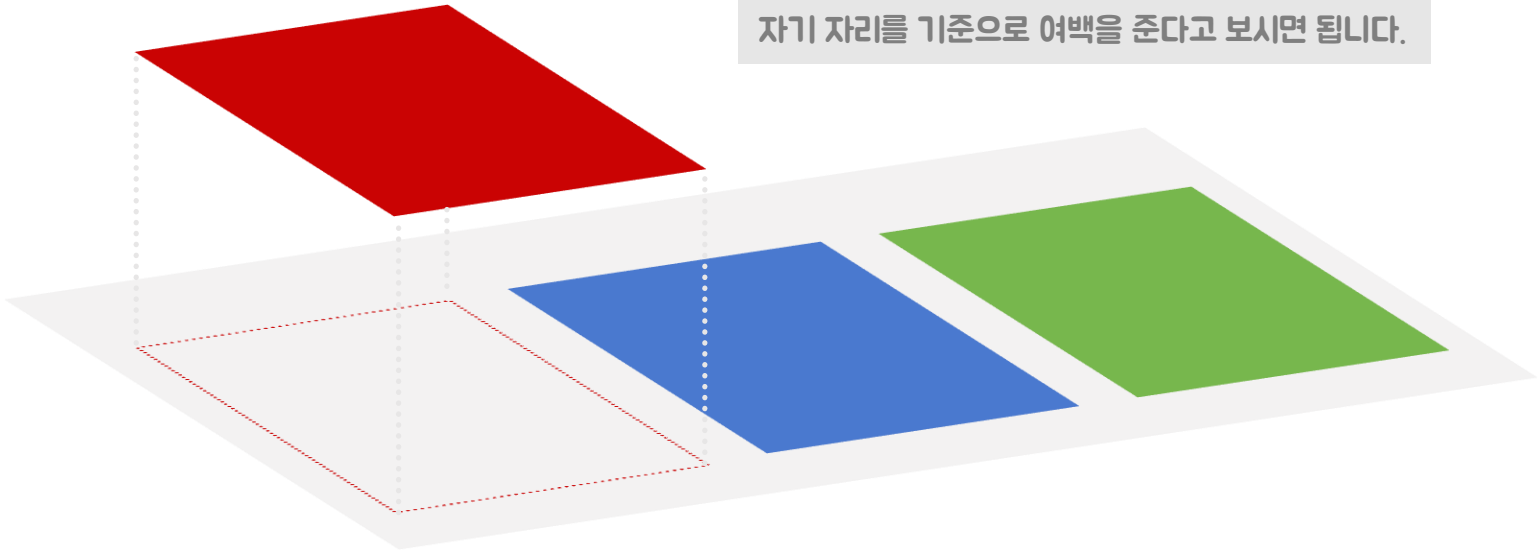
CSS Position

relative

해당 값부터는 **top | bottom | right | left** 와 **z-index** 를 사용할 수 있습니다.

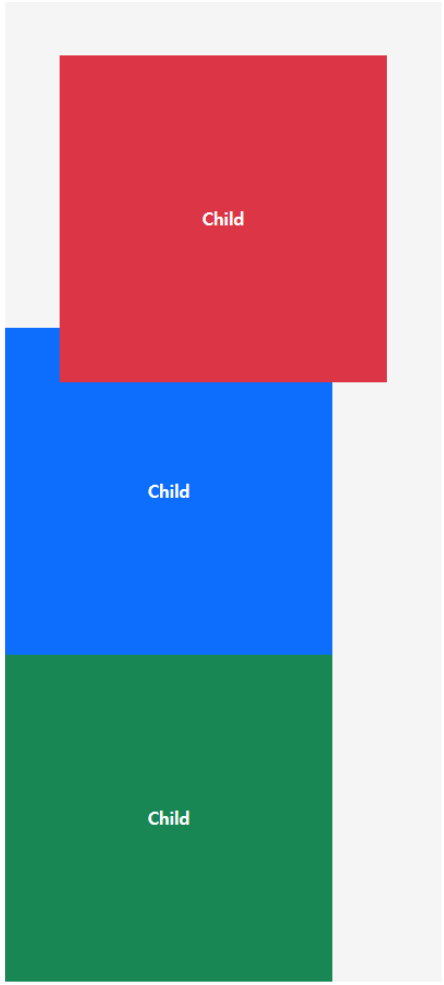
제공된 코드에서 red에서만 position: relative를 주게 되면 float와 같이 뜨게 되지만,
parent 영역의 공간이 줄어들진 않습니다.
이를 보아 position은 자기 자리를 기억한다는 것을 알 수 있습니다.

top | bottom 과 left | right 는
자기 자리를 기준으로 여백을 준다고 보시면 됩니다.



CSS Position

relative




```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
 ▼<body>
  ▼<div class="parent">
      <div class="child red" style="
          position: relative;
          left: 50px;
          top: 50px;
      ">Child</div> == $0
      <div class="child blue">Child</div>
      <div class="child green">Child</div>
    </div>
    <!-- Code injected by live-server -->
   ▶<script type="text/javascript">…</script>
  </body>
</html>
```

html body div.parent div.child.red

스타일 | 계산됨 | 레이아웃 | 이벤트 리스너 | DOM 중단점 | »

필터 :hov .cls +

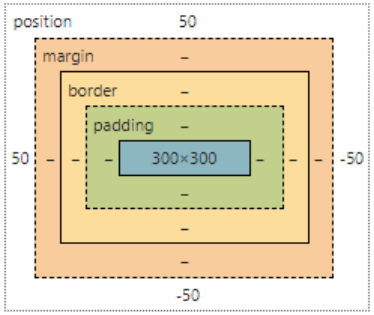
```
element.style {
    position: relative;
    left: 50px;
    top: 50px;
}
```

```
.red {                                          style.css:21
    background-color: #dc3545;
}
```

```
.child {                                        style.css:12
    width: 300px;
    height: 300px;
    line-height: 300px;
    color: white;
    font-weight: bold;
    text-align: center;
}
```

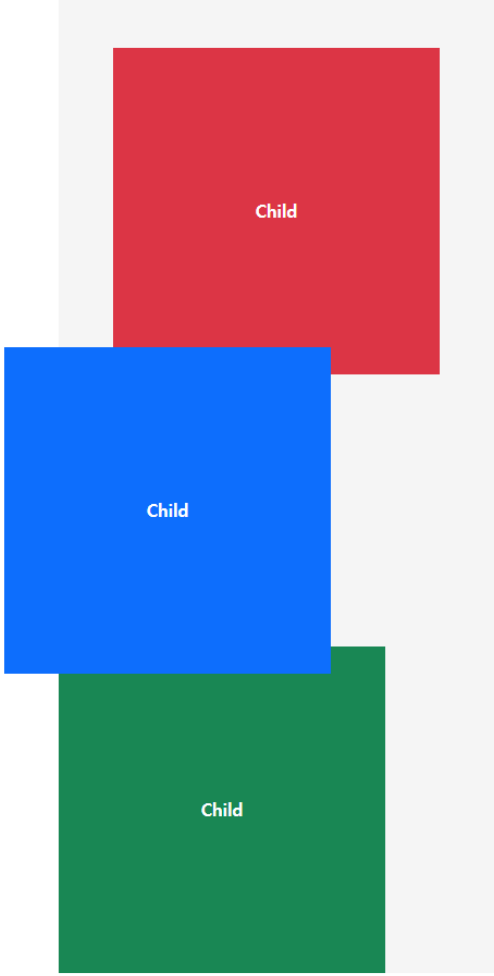
```
* {                                             style.css:1
    box-sizing: border-box;
    margin: ▶ 0;
}
```

```
div {                                  사용자 에이전트 스타일시트
    display: block;
}
```



CSS Position

relative



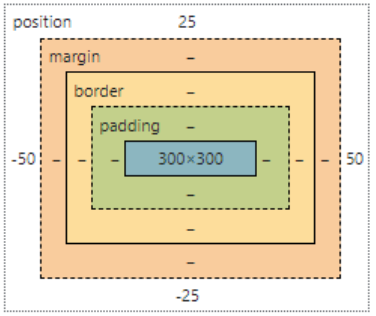

```
<!DOCTYPE html>
<html>
▶<head>…</head>
▼<body>
  ▼<div class="parent">
      <div class="child red" style="
          position: relative;
          left: 50px;
          top: 50px;
      ">Child</div>
      <div class="child blue" style="
          position: relative;
          right: 50px;
          top: 25px;
      ">Child</div> == $0
      <div class="child green">Child</div>
    </div>
    <!-- Code injected by live-server -->
  ▶<script type="text/javascript">…</script>
  </body>
</html>
```

html body div.parent div.child.blue

스타일 계산됨 레이아웃 이벤트 리스너 DOM 중단점 »

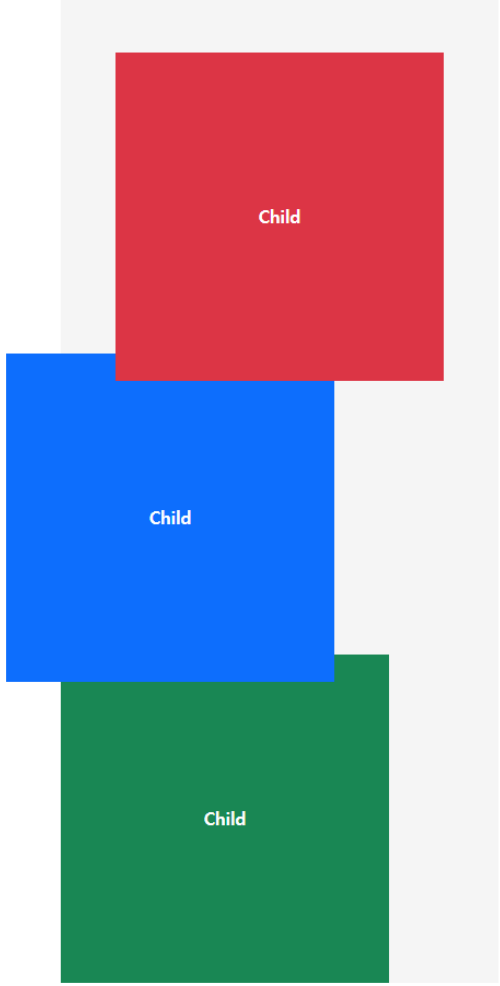
필터 :hov .cls +

```
element.style {
    position: relative;
    right: 50px;
    top: 25px;
}
.blue {                                   style.css:25
    background-color: #0d6efd;
}
.child {                                  style.css:12
    width: 300px;
    height: 300px;
    line-height: 300px;
    color: white;
    font-weight: bold;
    text-align: center;
}
* {                                       style.css:1
    box-sizing: border-box;
    margin: ▶ 0;
}
div {                                     사용자 에이전트 스타일시트
    display: block;
}
```



CSS
Position
relative

z-index를 써서 빨강 요소를 파랑보다 앞으로 오게 해보겠습니다.



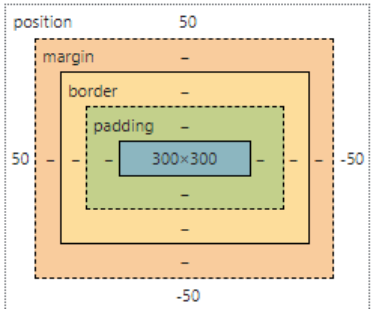

```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
 ▼<body>
  ▼<div class="parent">
      <div class="child red" style="
          position: relative;
          left: 50px;
          top: 50px;
          z-index: 1;
      ">Child</div> == $0
      <div class="child blue" style="
          position: relative;
          right: 50px;
          top: 25px;
      ">Child</div>
      <div class="child green">Child</div>
    </div>
    <!-- Code injected by live-server -->
   ▶<script type="text/javascript">…</script>
  </body>
</html>
```

html body div.parent div.child.red

스타일 계산됨 레이아웃 이벤트 리스너 DOM 중단점 »

필터 :hov .cls

```
element.style {
    position: relative;
    left: 50px;
    top: 50px;
    z-index: 1;
}
.red {                                          style.css:21
    background-color: #dc3545;
}
.child {                                        style.css:12
    width: 300px;
    height: 300px;
    line-height: 300px;
    color: white;
    font-weight: bold;
    text-align: center;
}
* {                                            style.css:1
    box-sizing: border-box;
    margin: ▶ 0;
}
div {                                  사용자 에이전트 스타일시트
    display: block;
}
```

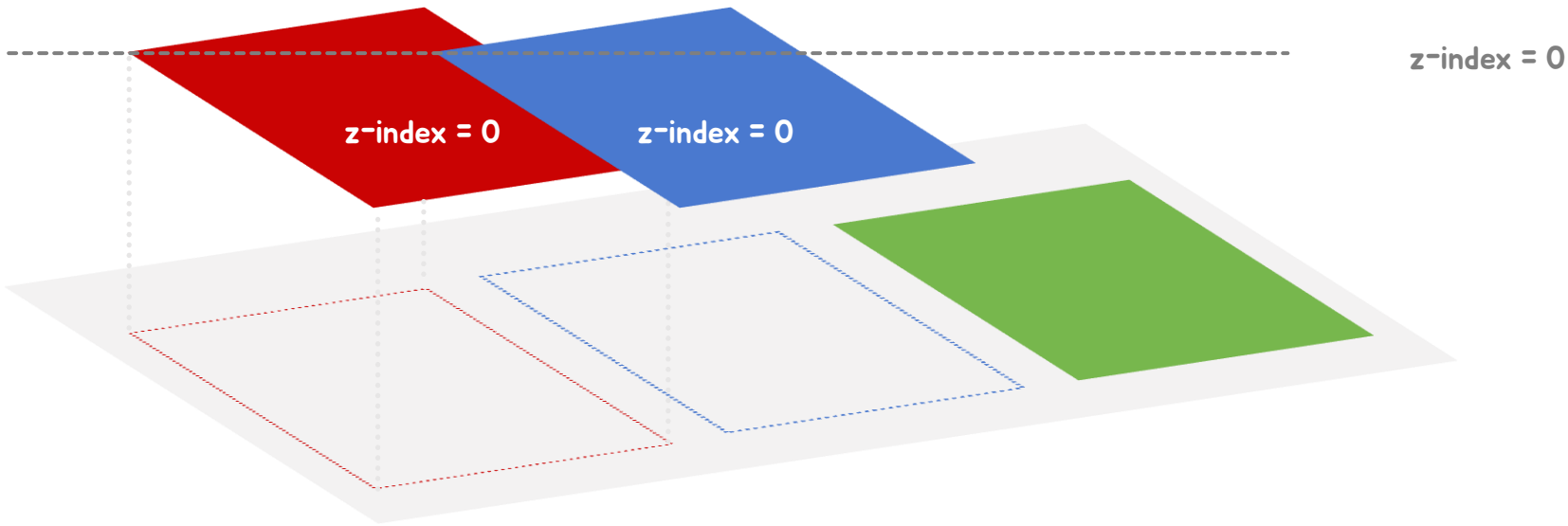


CSS

Position

relative

z-index



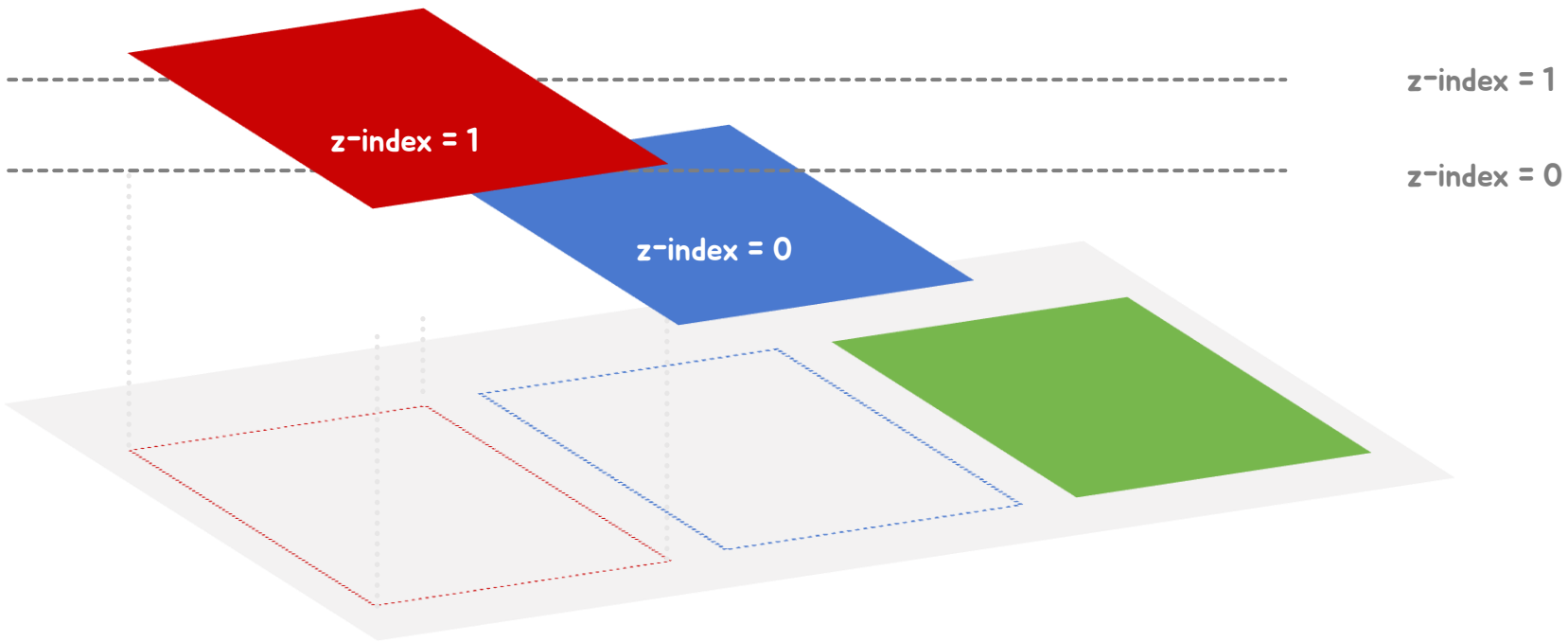
CSS

Position

relative

z-index

z-index가 크면 클수록 더 위에 놓여지게 됩니다.



CSS Position

static | relative | absolute | fixed | sticky

absolute Float와 유사해짐
기준점을 잡아야 하는데 기준점 대상이 static이면 안됨

fixed Viewport를 기준으로 기준점이 잡혀 있습니다.
내가 보고 있는 브라우저 창의 크기

sticky 스크롤 되는 상위 요소를 기준으로 오프셋을 적용합니다.

top | bottom | right | left

위의 4가지를 선택적으로 위치를 정하여 줍니다.
보편적으로 top과 bottom / right와 left를
동시에 적용하지 않습니다.

z-index

Z축 정도를 나타내서 상위에 무엇이
먼저 나타나야 하는지 값을 지정해줄 수 있습니다.

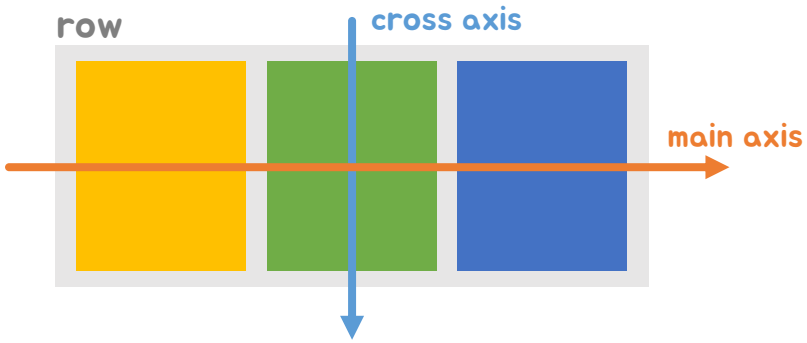
CSS Flex

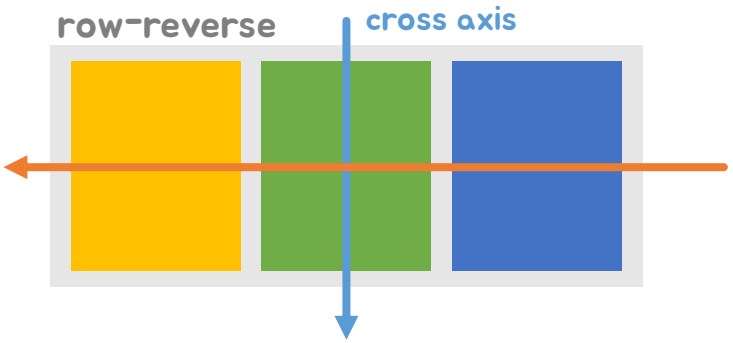
display: flex

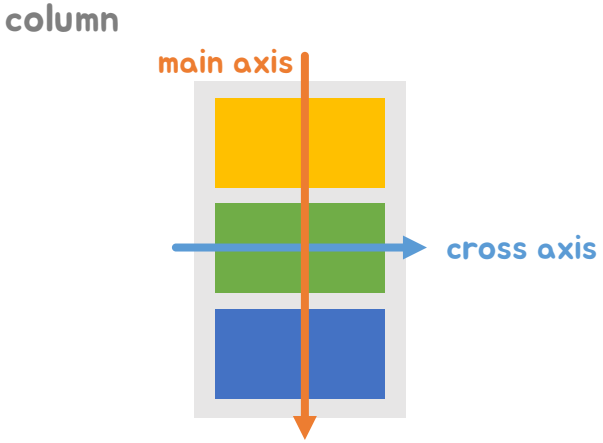
inline-flex

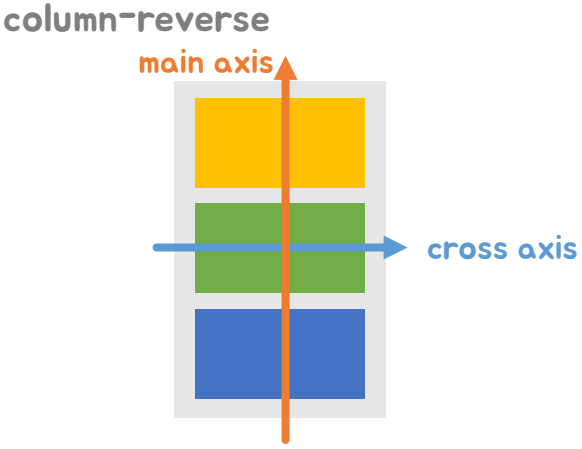
정렬하고자 하는 요소를 감싸는 부모에게 display: flex

flex-direction: row | row-reverse | column | column-reverse









CSS
Flex

flex-wrap

nowrap | wrap

nowrap: 감싸지 않고 자식의 사이즈를 줄여서 라도 한 줄로 정렬

wrap: 한 줄에 모두 정렬하기에 공간이 넉넉하지 않으면 여러 줄을 만듭니다

order 순서를 정해서 바꿔줄 수 있습니다

CSS

Flexbox Froggy

FLEXBOX FROGGY

단계 1 of 24

Flexbox Froggy에 오신 것을 환영합니다! Flexbox Froggy는 Froggy와 친구들 돕는 CSS 코드 게임입니다. 오른쪽의 `justify-content` 속성을 이용하여 개구리가 수련잎으로 이동할 수 있도록 도와주세요. 이 속성은 다음의 값들을 인자로 받아 요소들을 가로선 상에서 정렬합니다:

- `flex-start`: 요소들을 컨테이너의 왼쪽으로 정렬합니다.
- `flex-end`: 요소들을 컨테이너의 오른쪽으로 정렬합니다.
- `center`: 요소들을 컨테이너의 가운데로 정렬합니다.
- `space-between`: 요소들 사이에 동일한 간격을 둡니다.
- `space-around`: 요소들 주위에 동일한 간격을 둡니다.

예를 들어, `justify-content: flex-end;`는 개구리를 오른쪽으로 이동시킵니다.

```
1  #pond {
2    display: flex;
3
4  }
5
6
7
8
9
10
```

다음

Flexbox Froggy는 다음에 의해 개발되었습니다 Codepip • GitHub • Twitter • Settings

Want to learn CSS grid? Play Grid Garden.

CSS media

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
@media screen and (min-width: 425px) {
  .media-test::after {
    content: "425px";
  }
}

@media screen and (min-width: 768px) {
  .media-test::after {
    content: "768px";
  }
}

@media screen and (min-width: 1024px) {
  .media-test::after {
    content: "1024px";
  }
}
```

CSS

vh / vw

vh

viewport height 1% = 1vh

vw

viewport width 1% = 1vw