Hello,

KDT 웹 개발자 양성 프로젝트

5기!

with

사용자 입력 보내기

```
<body>
    <h1>강아지 vs 고양이</h1>
    <form action="http://3.34.177.57:4000" method="GET">
      <input type="text" name="name" />
      <br />
      <select name="choice">
        <option value="dog">강아지</option>
        <option value="cat">고양이</option>
        <option value="whatever">상관 없음</option>
      </select>
      <input type="submit" value="제출" />
    </form>
  </body>
```

```
데이터 통신 서버가 4000에서 작동 중입니다!
{ name: '이효석', choice: 'dog' }
```

git

Linus Benedict Torvalds

VCS(Version Control System)

Git이 하는 일은!

Commit?

```
$ git init
# 현재 프로젝트에서 변경사항 추적(버전 관리)을 시작.
```

git

사용자
(local, 로컬)

master

index.html

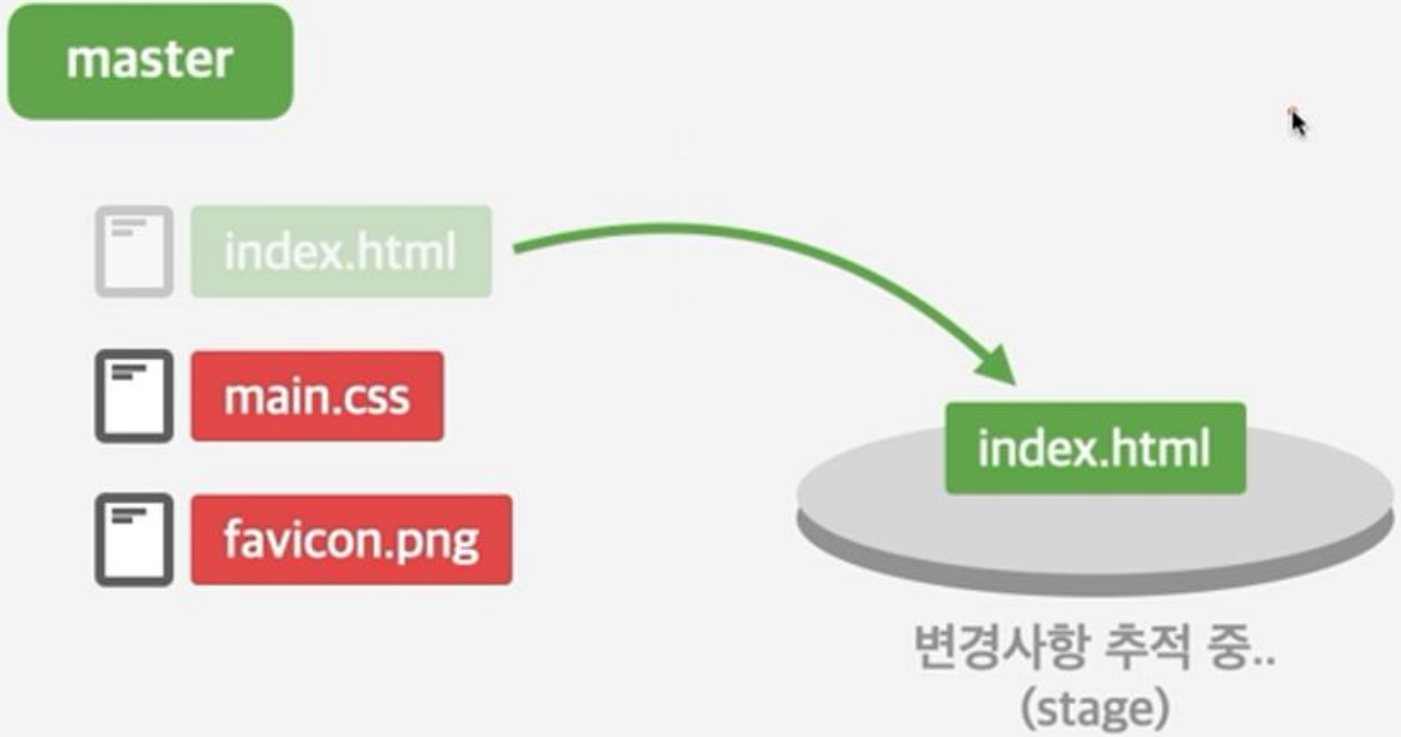
main.css

favicon.png

변경사항 추적 중..
(stage)

```
$ git add index.html
# 변경사항을 추적할 특정 파일(index.html)을 지정.
```

git 사용자 (local, 로컬)



```
$ git add .
# 모든 파일의 변경사항을 추적하도록 지정.
```

git 사용자 (local, 로컬)

master

```
$ git commit -m '프로젝트 생성'
# 메시지(-m)와 함께 버전을 생성.
```

git

사용자
(local, 로컬)

master

- index.html
- main.css
- favicon.png

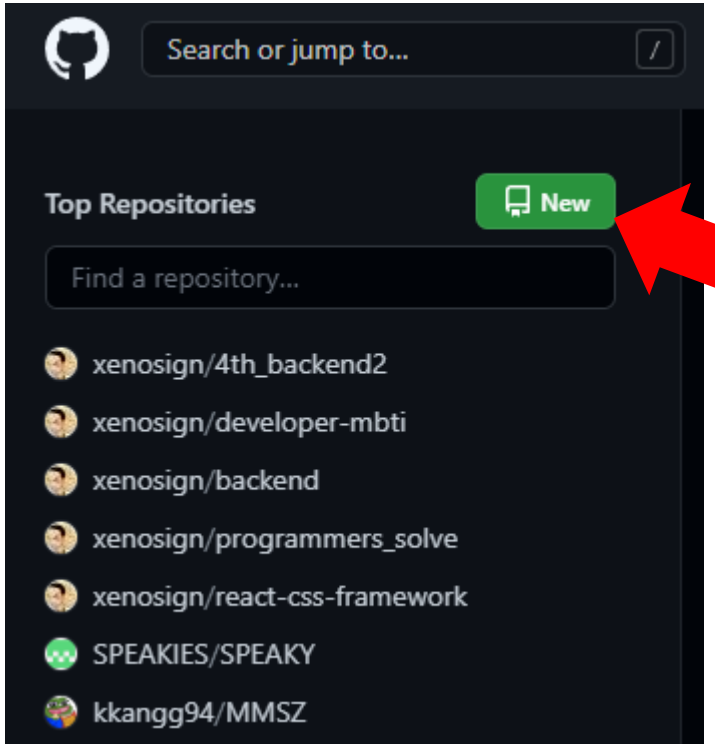
● 프로젝트 생성

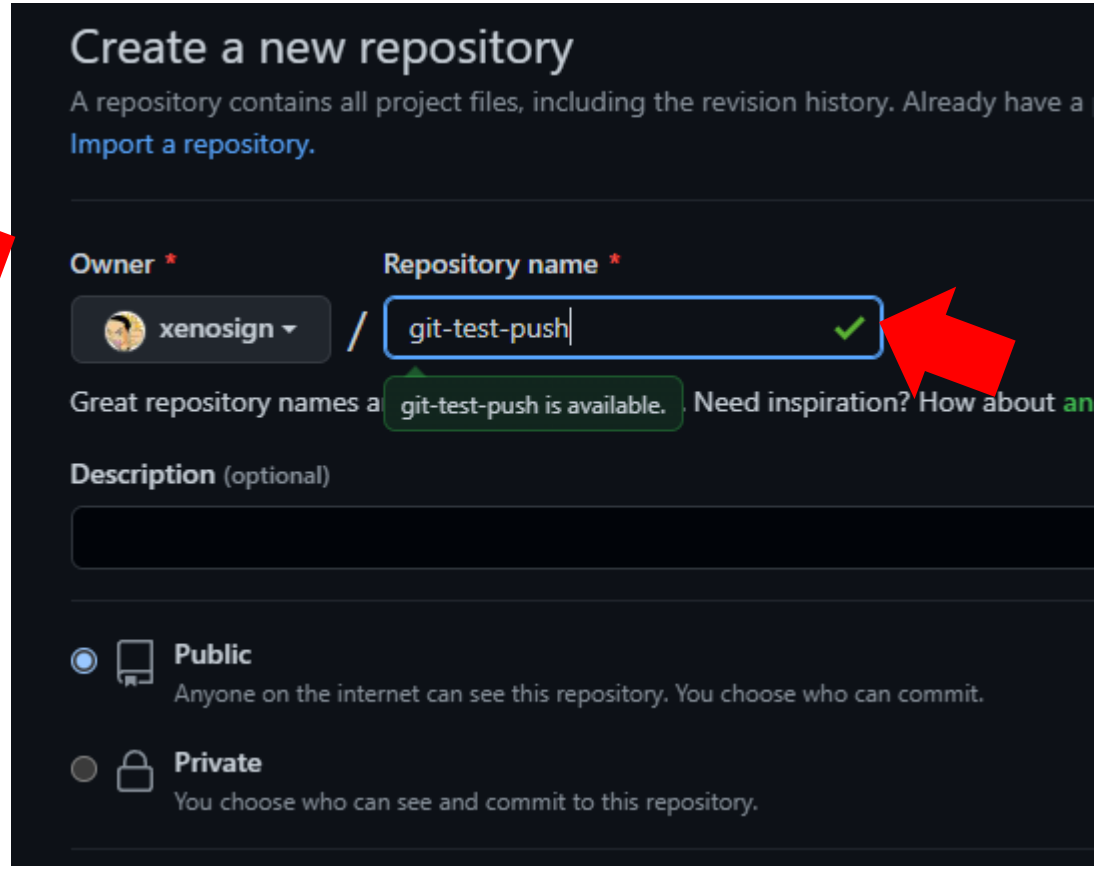
Push!?

GitHub
(remote, 원격)

원격 저장소
(Repository)

원격 저장소 생성하기


Search or jump to...
/
Top Repositories
New
Find a repository...
xenosign/4th_backend2
xenosign/developer-mbti
xenosign/backend
xenosign/programmers_solve
xenosign/react-css-framework
SPEAKIES/SPEAKY
kkangg94/MMSZ


Create a new repository
A repository contains all project files, including the revision history. Already have a
Import a repository.
Owner *
Repository name *
xenosign
/
git-test-push
Great repository names a
git-test-push is available.
Need inspiration? How about an
Description (optional)
Public
Anyone on the internet can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.

git 사용자 (local, 로컬)

GitHub (remote, 원격)

```
$ git remote add origin https://github.c...
# origin이란 별칭으로 원격 저장소를 연결.
```

master

- index.html
- main.css
- favicon.png
- main.js

원격 저장소 (Repository)

git 사용자 (local, 로컬)

GitHub (remote, 원격)

```
$ git push origin master
# origin이란 별칭의 원격 저장소로 버전 내역 전송.
```

master

- index.html
- main.css
- favicon.png
- main.js

원격 저장소 (Repository)

CSS

신문검색 | 국외검색 NAVER 서치센터 | 광고안내

맞춤정보 서비스, MyNAVER
매일매일 핫&쿨, WebToday

삼성생명 오픈 이벤트

네이버를 나의 홈페이지로,,,
네오넷의 부동산 급매물 서비스!

NAVER 검색

분류에서 검색

제목에서 검색 Quick Help 확장검색

NAVER 소식

네이버 포탈서비스 오픈! MyNAVER
네이버 웹문서수 3,141,838 건

NAVER 분류

- 건강/의학 — 건강관리, 병원, 의학
- 교육 — 대학, 시험,자격증, 유학
- 뉴스/미디어 — 신문, 잡지, 텔레비전
- 레크리에이션 — 스포츠, 게임, 여행, 레저
- 사업/경제 — 기업, 취업, 온라인쇼핑
- 사회과학 — 경제학, 사회학, 언어학
- 사회/문화 — 결혼, 기관/단체
- 연예/오락 — 연예인, 영화, 음악, 유머
- 인문/예술 — 디자인, 인문과학, 박물관
- 자연과학 — 공학, 컴퓨터과학, 생물학
- 정부/공공기관 — 정치, 한국정부, 국제기구
- 지역정보 — 대한민국, 서울, 국가
- 참고자료 — 도서관, 사람찾기, 사전
- 컴퓨터 — 인터넷, S/W, O/S, 통신

NAVER 추천

New! 크리스마스
행사,이벤트 여행 온라인쇼핑
영화,비디오 취업,채용 투자,재테크

AT & T 전화접속서비스

도움말 | 새소식 | 홈페이지등록 | 다른검색사이트

Microsoft Internet Explorer

VS

NAVER whale 눈부심 없는 편안~함 내가 다크 모드 쓰는 이유 다운로드 3일 동안 보지 않기

네이버를 시작페이지로

NAVER

메일 카페 블로그 지식iN 쇼핑 쇼핑LIVE Pay TV 사전 뉴스 증권 부동산 지도 VIBE 책 웹툰 더보기 미세 좋음 초미세 좋음

4월 11일 부터 자동차보험료를 내렸습니다!
개인용 자동차 평균 1.2% ↓, 4/11 보험시작일 기준
바로확인

네이버를 더 안전하고 편리하게 이용하세요
NAVER 로그인
아이디 · 비밀번호찾기 회원가입

연합뉴스 › '오자범위내 접전' 인천 계양을...이재명-윤형선 총력전
뉴스홈 · 연예 스포츠 지방선거

증시 | 다우 32,893.24 ▲256.05 +0.78%

뉴스스탠드 › 구독한 언론사 · 전체언론사

연합뉴스TV KBS 한국경제TV 매일경제 헤럴드경제 뉴스1
동아일보 노컷뉴스 KBS WORLD 이데일리 프레시안 BLOTER
Net Korea sportalkorea NewDaily 디지털정글 DAILY NK YTN 사이언스
CEO스코어데일리 Media오늘 Byline Network 비즈한국 더쿠쿠프 뉴스핌

제8회 지방선거 사전투표2일차
투표 안내 › 투표소 찾기 › 후보자 정보 › 투표시간 오전 6시 ~ 오후 6시

오늘 읽을만한 글 주제별로 분류된 다양한 글 모음
681 개의 글 | 관심주제 설정

N 인플루언서
번아웃 증후군을 이겨내는 법
도서 블로거 @힐링의숲 추천
직장인 마음 치유 도서

트렌드쇼핑 › 상품 쇼핑몰 MEN
쿠팡 · G마켓 · 옥션 · 11번가 · 티몬 · 올리브영
위메프 · SSG닷컴 · GS샵 · 롯데홈 · CJ온스타일 · 패션플러스

1 / 18

CSS를 적용하는 방식

- 인라인 스타일
- 내장 Style
- 파일 링크

인라인(inline) 방식은 불편해요!!

인라인 방식

요소의 style 속성에 직접 스타일을 작성하는 방식
(선택자 없음)

```
<div style="color: red; margin: 20px;"></div>
```

- 각각 태그마다 전부 스타일을 적어줘야 함
- 아래에서 같은 스타일을 가진 태그를 사용하려고 해도 코드를 복붙 필요
- 즉, 재사용이 전혀 불가능!

그래서 탄생! 내장 Style

```
<style>
  div {
    color: red;
    margin: 20px;
  }
</style>
```

내장 방식

<style></style>의 내용(Contents)으로
스타일을 작성하는 방식

그래서 탄생! 링크 방식

```
<link rel="stylesheet" href="./css/main.css">
```

링크 방식

<link /> 로 외부 CSS 문서를
가져와서 연결하는 방식

main.css

```
div {
  color: red;
  margin: 20px;
}
```

그래서 탄생! @import 방식

```
<link rel="stylesheet" href="./css/main.css">
```

main.css

```
@import url("./box.css");

div {
  color: red;
  margin: 20px;
}
```

box.css

```
.box {
  background-color: red;
  padding: 20px;
}
```

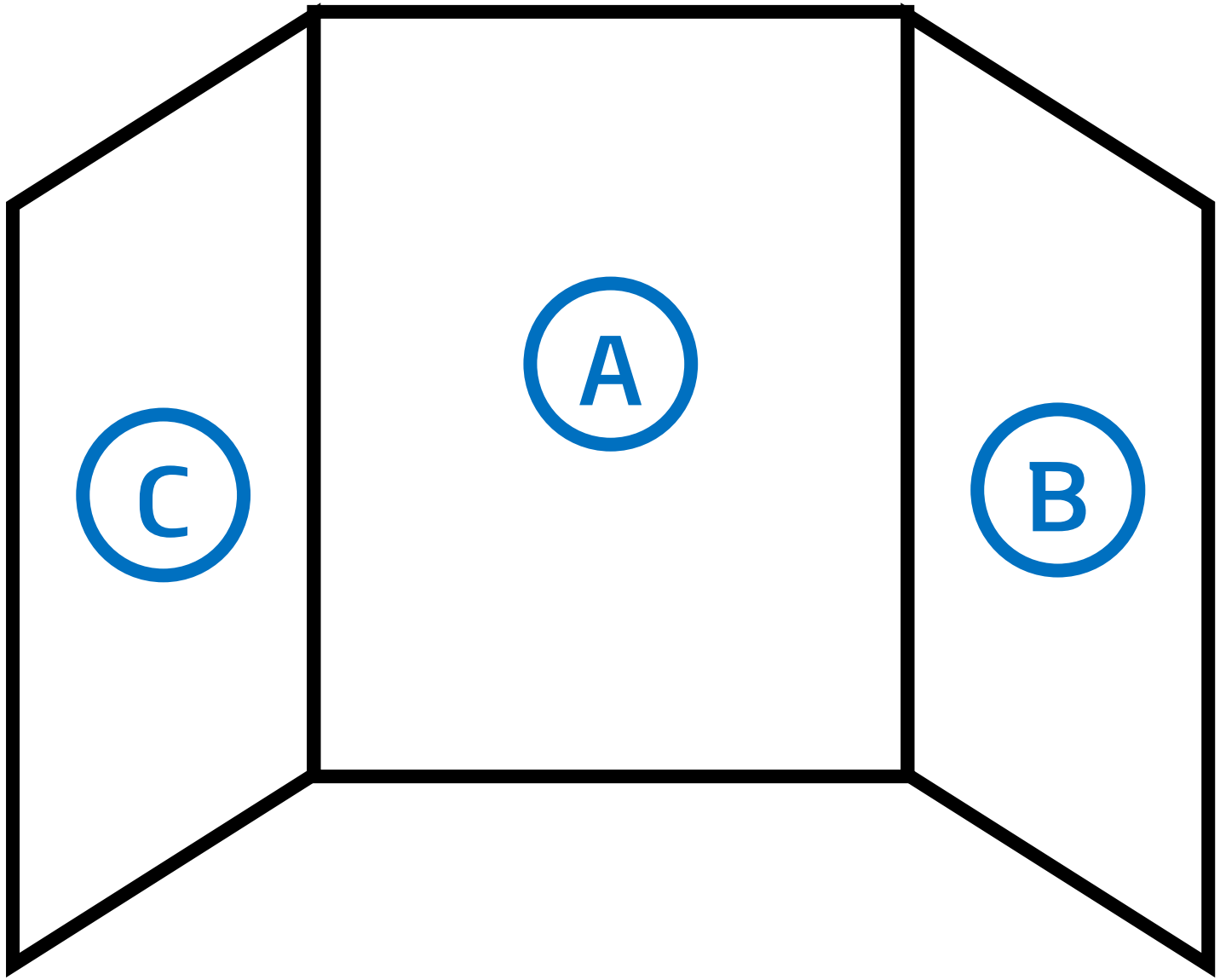
@import 방식

CSS의 @import 규칙으로 CSS 문서 안에서
또 다른 CSS 문서를 가져와 연결하는 방식


지금 시작합니다

한번

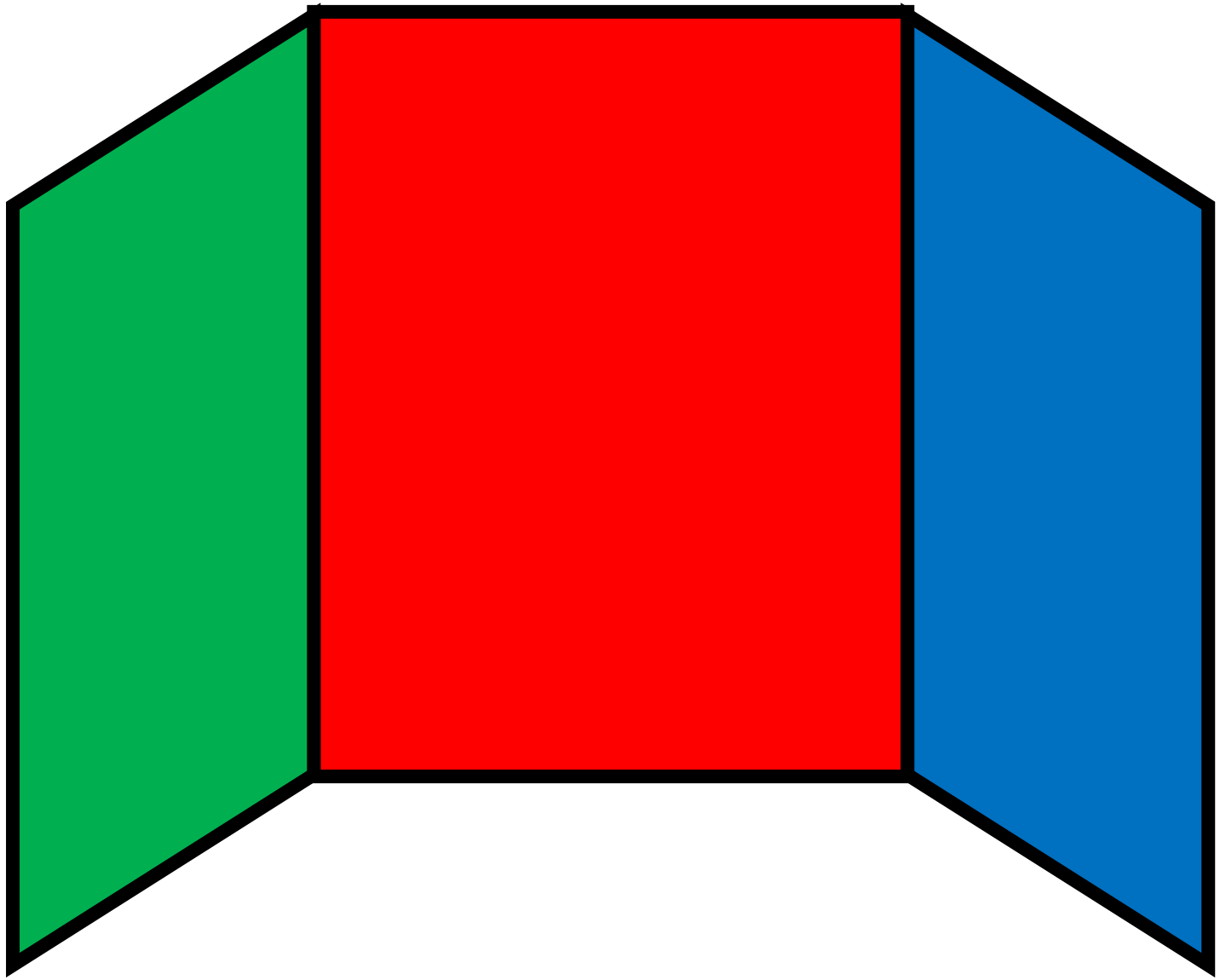
생각해 보세요!


A
C
B


2"
SERIES P-200-L

"빨간, 파란, 초록색으로 예쁘게 칠해주세요"

"A는 빨간색
B는 파란색
C는 초록색으로
예쁘게 칠해주세요!"



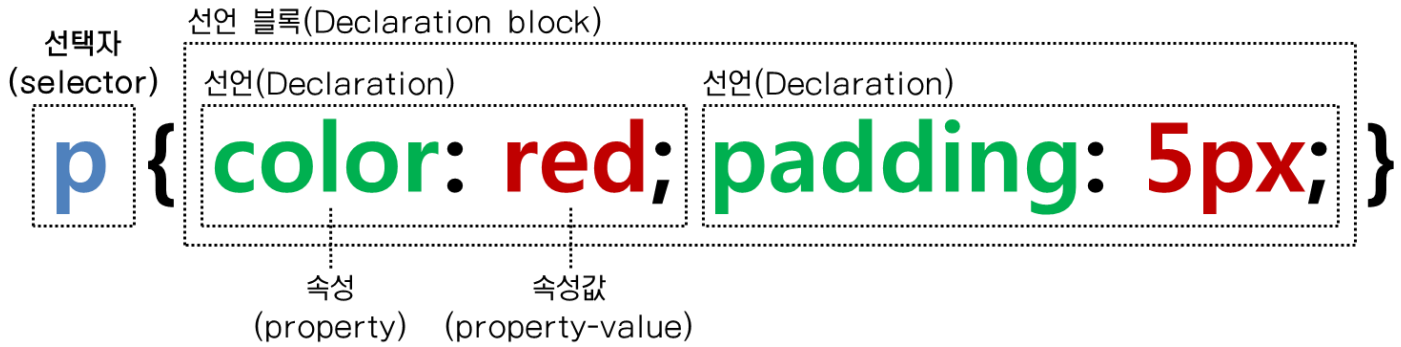

2"
SERIES P-200-L

CSS를 적용 할 때도 어떤 곳에 적용할 지 정확하게 불러주는 것이 중요합니다~!

CSS

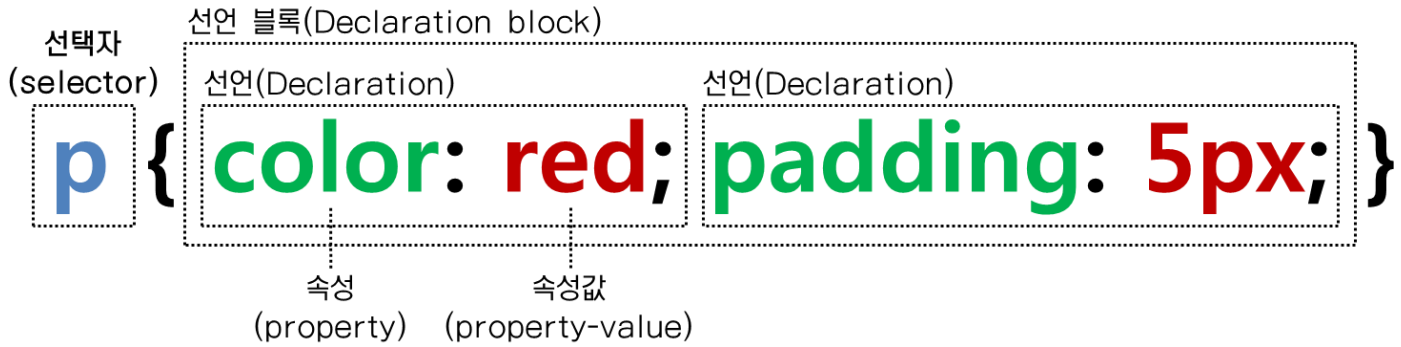
선택자!

CSS, 선택자(Selector)



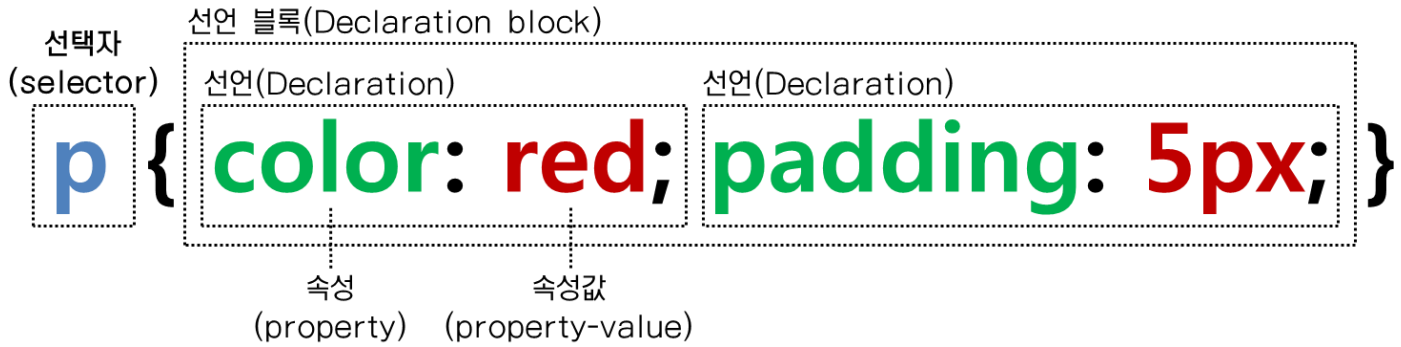
- CSS는 HTML 요소의 style(design, layout etc)을 정의 하여야 하므로 CSS 사용을 위해서는 스타일을 적용하고자 하는 HTML 요소를 선택할 수 있어야 한다
- 선택자는 스타일을 적용하고자 하는 HTML 요소를 선택하기 위해 CSS에서 제공하는 수단

CSS, 속성(Property)



- 선택자로 HTML 요소를 선택하고 { } 내에 속성 값을 지정하여 다양한 style 을 정의
- 속성은 표준 스펙으로 이미 지정되어 있는 것을 사용, 사용자가 임의로 정의 할 수 없다
- 여러 개의 프로퍼티를 연속해서 지정할 수 있으며 세미콜론(;)으로 구분

CSS, 값(Value)



- 값은 해당 속성에 사용할 수 있는 값을 키워드나 크기 단위 또는 색상 단위 등의 특정 단위로 지정

CSS 선택자

기본 | 복합 | 가상 클래스 | 가상 요소 | 속성

기본 선택자!

- 기본 선택자니까 기본적인 선택자겠죠?
- 별도의 테크닉 없이, 순수하게 무엇인가를 호출 할 때 사용합니다!
- 종류
  - 전체 선택자
  - 태그 선택자
  - Class 선택자
  - ID 선택자

* 기본 전체 선택자 (Universal Selector)

모든 요소를 선택.

```html
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li>오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span>오렌지</span>
</div>
```

선택

```css
* {
  color: red;
}
```

ABC

기본 | 태그 선택자 (Type Selector)

태그 이름이 ABC인 요소 선택.

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li>오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span>오렌지</span>
</div>
```

선택

```
li {
  color: red;
}
```

.ABC

기본 | 클래스 선택자 (Class Selector)

HTML class 속성의 값이 ABC인 요소 선택.

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li class="orange">오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span class="orange">오렌지</span>
</div>
```

선택

```
.orange {
  color: red;
}
```

#ABC

기본 | 아이디 선택자 (ID Selector)

HTML id 속성의 값이 ABC인 요소 선택.

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li id="orange" class="orange">오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span class="orange">오렌지</span>
</div>
```

선택

```
#orange {
  color: red;
}
```

실습, 기본 선택자 활용

```
<h1>KDT 선택자 예제</h1>
<p>선택자 사용 연습을 해봅시다</p>
<span>기본 선택자를 위한 예제입니다</span>
<ul>
  선택자로 선택하기
  <li id="first">1</li>
  <li class="second">2</li>
  <li>3</li>
  <li>4</li>
</ul>
```

- 전체 태그에 대한 글자색 → 하얀색
- 태그 배경색 → 빨간색
- id 가 first 인 태그 배경색 → 파란색
- class 가 second 인 태그 배경색 → 초록색

CSS 선택자

기본 | 복합 | 가상 클래스 | 가상 요소 | 속성

복합 선택자!

- 특수한 요소를 호출하고 싶을 때, 기본 선택자만으로는 선택이 불가능한 경우에 사용
- 종류
  - 일치 선택자
  - 자식 선택자
  - 후손 선택자
  - 인접 형제 선택자
  - 일반 형제 선택자

ABCXYZ

복합 | 일치 선택자 (Basic Combinator)

선택자 ABC와 XYZ를 **동시에 만족**하는 요소 선택.

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li class="orange">오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span class="orange">오렌지</span>
</div>
```

선택

```
span.orange {
  color: red;
}
```

ABC > XYZ

복합 | 자식 선택자 (Child Combinator)

선택자 ABC의 자식 요소 XYZ 선택.

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li class="orange">오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span class="orange">오렌지</span>
</div>
```

선택

```
ul > .orange {
  color: red;
}
```

ABC XYZ

복합 **하위(후손) 선택자 (Descendant Combinator)**

선택자 ABC의 **하위** 요소 XYZ 선택.
'띄어쓰기'가 선택자의 기호!

```
<div>
  <ul>
    <li>사과</li>
    <li>딸기</li>
    <li class="orange">오렌지</li>
  </ul>
  <div>당근</div>
  <p>토마토</p>
  <span class="orange">오렌지</span>
</div>
<span class="orange">오렌지</span>
```

선택

```
div .orange {
  color: red;
}
```

ABC + XYZ

복합 | 인접 형제 선택자 (Adjacent Sibling Combinator)

선택자 ABC의 다음 형제 요소 XYZ 하나를 선택.

```
.orange + li {
  color: red;
}
```

선택

```
<ul>
  <li>딸기</li>
  <li>수박</li>
  <li class="orange">오렌지</li>
  <li>망고</li>
  <li>사과</li>
</ul>
```

ABC ~ XYZ

복합 | 일반 형제 선택자 (General Sibling Combinator)

선택자 ABC의 다음 형제 요소 XYZ 모두를 선택.

```css
.orange ~ li {
  color: red;
}
```

선택

```html
<ul>
  <li>딸기</li>
  <li>수박</li>
  <li class="orange">오렌지</li>
  <li>망고</li>
  <li>사과</li>
</ul>
```

실습, 일치 선택자 활용

```
<h1>복합 선택자 사용 연습</h1>
<p>동물원에 왔어요</p>
<div class="zoo">
  <ul>
    여긴 사파리!
    <li>곰</li>
    <li id="tiger">호랑이</li>
    <li>팬더</li>
    <li class="lion">사자</li>
    <li>사육사</li>
    <li>사육사2</li>
  </ul>
  <span class="lion">사파리 밖의 사자</span>
</div>
<p class="lion">동물원 밖의 사자</span>
```

- 동물원 밖의 사자만 선택, 배경을 빨간색
- 사파리 안의 사자만 선택, 배경을 초록색
- 호랑이만 선택, 배경을 노란색
- 사자를 전부 선택, 글자를 하얀색
- 동물원 안의 사자만 선택, 폰트를 두껍게
- 호랑이 뒤에 있는 사육사를 선택, 배경을 하늘색
- 사파리 사자 뒤에 있는 사육사를 모두 선택, 배경을 오렌지색

CSS 선택자

기본 복합 가상 클래스 가상 요소 속성

가상 클래스 선택자!

- 사용자의 행동에 따라 변화하는 가상 상황에 따라서 요소 선택 시
- 각 요소의 상황에 따라 사용자가 원하는 요소를 선택 할 때 사용
- 특정 요소를 부정 할 때 사용
- 종류
  - 가상 클래스 선택자
  - 사용자의 행동에 따라 변화 : Hover, Active, Focus
  - 요소의 상황 : first-child, last-child, nth-child
  - 부정 선택 : not

ABC:hover

가상 클래스 선택자 (Pseudo-Classes) HOVER

선택자 ABC 요소에 마우스 커서가 올라가 있는 동안 선택.

화면에 출력!

NAVER

NAVER

선택

```
a:hover {
    color:  red;
}
```

```
<a href="https://www.naver.com">NAVER</a>
```

ABC:active

가상 클래스 선택자 (Pseudo-Classes) ACTIVE

선택자 ABC 요소에 마우스를 클릭하고 있는 동안 선택.

화면에 출력!

NAVER

NAVER

선택

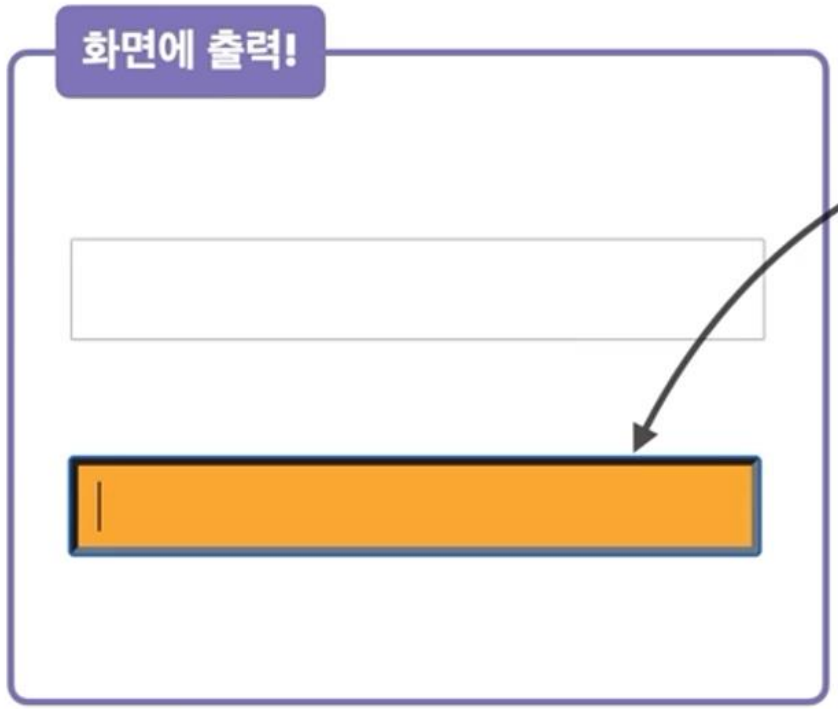
```
a:active {
    color:  red;
}
```

```
<a href="https://www.naver.com">NAVER</a>
```

ABC:focus

가상 클래스 선택자 (Pseudo-Classes) FOCUS

선택자 ABC 요소가 포커스되면 선택.




```css
input:focus {
  background-color: orange;
}
```

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

Focus가 안되는 요소에 강제로 추가하기!

- 속성으로 tabindex="-1" 값을 주어서 포커스 효과를 줄 수 있음
- 속성 이름을 보고 유추 할 수 있듯, Tab 키를 눌러서 포커스를 하는 순서를 지정하는 속성
- "-1" 이외의 값을 넣을 경우 기존 페이지의 포커스 흐름이 깨지기 때문에 비추천

실습, 가상 클래스 선택자 1

- <h1>, <a> 은 1개, <input type="text"> 태그는 2개 만들기
- <h1> 태그에 마우스가 올라가면 글자가 빨간색
- <a> 태그를 클릭 하고 있으면 배경이 파란색
- <input> 태그 중 하나의 <input> 태그에 포커스가 가면 입력창이 오렌지색으로 변경

ABC:first-child

가상 클래스 선택자 (Pseudo-Classes) | FIRST CHILD

선택자 ABC가 형제 요소 중 첫째라면 선택.

```
.fruits span:first-child {
  color: red;
}
```

선택

```
<div class="fruits">
  <span>딸기</span>
  <span>수박</span>
  <div>오렌지</div>
  <p>망고</p>
  <h3>사과</h3>
</div>
```

?

```
.fruits div:first-child {
  color: red;
}
```

ABC:last-child

가상 클래스 선택자 (Pseudo-Classes) | LAST CHILD

선택자 ABC가 형제 요소 중 막내라면 선택.

```
.fruits h3:last-child {
  color: red;
}
```

선택

```
<div class="fruits">
  <span>딸기</span>
  <span>수박</span>
  <div>오렌지</div>
  <p>망고</p>
  <h3>사과</h3>
</div>
```

ABC:nth-child(n)

가상 클래스 선택자 (Pseudo-Classes) NTH CHILD

선택자 ABC가 형제 요소 중 (n)째라면 선택.

```
.fruits *:nth-child(2) {
  color: red;
}
```

선택

```
<div class="fruits">
  <span>딸기</span>
  <span>수박</span>
  <div>오렌지</div>
  <p>망고</p>
  <h3>사과</h3>
</div>
```

ABC:nth-child(n)

가상 클래스 선택자 (Pseudo-Classes) | NTH CHILD

선택자 ABC가 형제 요소 중 (n)째라면 선택.

```
<div class="fruits">
  <span>딸기</span>
  <span>수박</span>
  <div>오렌지</div>
  <p>망고</p>
  <h3>사과</h3>
</div>
```

선택

```
.fruits *:nth-child(2n) {
  color: red;
}
```

n은 0부터 시작!
(Zero-Based Numbering)

```
.fruits *:nth-child(n+2) {
  color: red;
}
```

n은 0부터 시작!
(Zero-Based Numbering)

ABC:not(XYZ)

부정 선택자 (Negation) NOT

선택자 XYZ가 아닌 ABC 요소 선택.

```
.fruits *:not(span) {
  color: red;
}
```

선택

```
<div class="fruits">
  <span>딸기</span>
  <span>수박</span>
  <div>오렌지</div>
  <p>망고</p>
  <h3>사과</h3>
</div>
```

실습, 가상 클래스 선택자 2

```
<body>
  <div class="zoo">
    <p class="predator">곰</p>
    <p class="predator">사자</p>
    <p class="predator">호랑이</p>
    <p class="predator">재규어</p>
    <div>기린</div>
    <p class="predator">치타</p>
    <h3>얼룩말</h3>
    <span class="predator">하이에나</span>
  </div>
</body>
```

- 사자, 재규어, 치타, 하이에나 선택 → 배경 오렌지
- 곰만 선택 → 배경 보라색
- 초식 동물만 선택 → 배경 초록색
- 하이에나만 선택 → 배경 빨간색 / 마우스가 올라가면 배경
- 호랑이만 선택 → 배경 파란색

CSS 선택자

기본 복합 가상 클래스 가상 요소 속성

가상 요소 선택자!

- 선택 된 요소의 앞, 뒤에 별도의 Content 를 삽입하는 선택자
- 반드시 content 라는 속성을 사용
- 빈 값(“”) 이라도 넣어 주어야 적용이 됨
- 종류
  - After : 요소의 뒤에 내용 삽입
  - Before : 요소의 앞에 내용 삽입

ABC::before

인라인(글자) 요소

가상 요소 선택자 (Pseudo-Elements) BEFORE

선택자 ABC 요소의 내부 앞에 내용(Content)을 삽입.

```
.box::before {
  content: "앞!";
}
```

화면에 출력!

앞! Content!

::before

```
<div class="box">
  
  Content!

</div>
```

ABC::after

가상 요소 선택자 (Pseudo-Elements) AFTER

선택자 ABC 요소의 내부 뒤에 내용(Content)을 삽입.

인라인(글자) 요소

```
.box::after {
  content: "뒤!";
}
```

화면에 출력!

Content! 뒤!

::after

```
<div class="box">

  Content!

</div>
```

실습, 가상 요소 선택자

- 여기요! 만들기
- 여기요 앞에, "택시" 넣기
- 겨이요 뒤에, "빨리" 넣기

가상 요소 선택자!

- 실제로 의미 없는 HTML 태그를 만들지 않고 요소 삽입이 가능하여 매우 자주 사용!
- 예를 들어 쇼핑몰 페이지에 메뉴에 Hot, 추천 등을 넣기 위해 별도의 태그를 삽입 하는 것이 아니라 가상 요소 선택자를 활용하여 처리하면 편리함!

CSS 선택자

기본 복합 가상 클래스 가상 요소 속성

속성 선택자!

- 지정한 특정 속성을 가지고 있는 태그를 선택하는 선택자
- 종류
  - 특정 속성만 지정
  - 속성과 속성의 값을 지정

[ABC]

속성 선택자 (Attribute) ATTR

속성 ABC을 포함한 요소 선택

```
[disabled] {
  color: red;
}
```

선택

```
<input type="text" value="HEROPY">
<input type="password" value="1234">
<input type="text" value="ABCD" disabled>
```

[ABC="XYZ"]

속성 선택자 (Attribute) ATTR=VALUE

속성 ABC을 포함하고 값이 XYZ인 요소 선택.

```
[type="password"] {
  color: red;
}
```

선택

```
<input type="text" value="HEROPY">
<input type="password" value="1234">
<input type="text" value="ABCD" disabled>
```

실습, 가상 요소 선택자

```
<input type="text" placeholder="이름" />
<input type="password" value="pw" />
<input type="text" value="000-0000-0000" />
<input type="text" placeholder="핸드폰" />
<input type="text" placeholder="주민번호" disabled />
```

- Disabled 속성을 가진 태그 선택 → 배경 빨간색
- placeholder 속성이 “이름” 값을 가지는 태그 선택 → 배경 오렌지
- placeholder 속성을 가지지 않는 태그 선택 → 배경 파란색

스타일

상속!

```css
.animal {
  color: red;
}
```

선택

```html
<div class="ecosystem">생태계
  <div class="animal">동물
    <div class="tiger">호랑이</div>
    <div class="lion">사자</div>
    <div class="elephant">코끼리</div>
  </div>
  <div class="plant">식물</div>
</div>
```

?

스타일 상속!

- 자식 요소가 별도의 Style 속성이 없을 경우, 부모의 속성 값을 그대로 상속하는 CSS 특성!
- 단, 상속이 되는 속성과 상속이 안되는 속성으로 구분

상속되는 CSS 속성들..

모두 글자/문자 관련 속성들!

(모든 글자/문자 속성은 아님 주의!)

font-style : 글자 기울기
font-weight : 글자 두께
font-size : 글자 크기
line-height : 줄 높이
font-family : 폰트(서체)
color : 글자 색상
text-align : 정렬
...

강제 상속!

- 자동 상속 되지 않는 속성의 값을 “inherit”을 주어 부모의 CSS 속성 값을 그대로 상속하게 하는 방법

```css
.parent {
  width: 400px;
  height: 200px;
  background-color: red;
  border: 1px solid #000;
}
.child {
  width: 100px;
  height: inherit;
  top: 100px;
  right: 100px;
  position: fixed;
  border: 1px solid #000;
}
```

선택자 우선 순위!

- CSS 참조 방식 우선 순위와는 다른 개념입니다!
- 같은 HTML 요소가 여러 선택자에 의해서 선택 되었을 경우 어떤 선택자의 CSS 속성을 우선 적용할지 결정하는 방법
- 1 순위 : 점수가 높은 선택자가 우선!
- 2 순위 : 점수가 같으면, 가장 마지막에 읽힌 선택자가 우선!

```css
div              { color: red !important; }
#color_yellow    { color: yellow; }
.color_green     { color: green; }
div              { color: blue; }
*                { color: darkblue; }
body             { color: violet; }
```

선택

```html
<div
  id="color_yellow"
  class="color_green"
  style="color: orange;">
  Hello world!
</div>
```

과연 글자색은??

!important - 9999999999점

ID 선택자 - 100점

Class 선택자 - 10점

태그 선택자 - 1점

전체 선택자 - 0점

상속 - X

```
div                { color: red !important; }
#color_yellow      { color: yellow; }
.color_green       { color: green; }
div                { color: blue; }
*                  { color: darkblue; }
body               { color: violet; }
```

인라인 선언 - 1000점

```
<div
  id="color_yellow"
  class="color_green"
  style="color: orange;">
  Hello world!
</div>
```

선택자, 우선 순위 점수

- !important : 9999999999 점
- 인라인 방식 : 1000 점
- #ID 선택자 : 100 점
- .Class 선택자 : 10 점
- 태그 선택자 : 1 점
- 전체 선택자(*) : 0 점

21점 `.list li.item { color: red; }`

21점 `.list li:hover { color: red; }`

11점 `.box::before { content: "Good "; color: red; }`

101점 `#submit span { color: red; }`

22점 `header .menu li:nth-child(2) { color: red; }`

1점 `h1 { color: red; }`

10점 `:not(.box) { color: red; }`

CSS의 속성

CSS의

목적이 뭐죠?

박스 모델

글꼴, 문자

배경

배치

플렉스(정렬)

전환

변환

띄움

애니메이션

그리드

다단

필터

박스 모델

글꼴, 문자

배경

배치

플렉스(정렬)

전환

변환

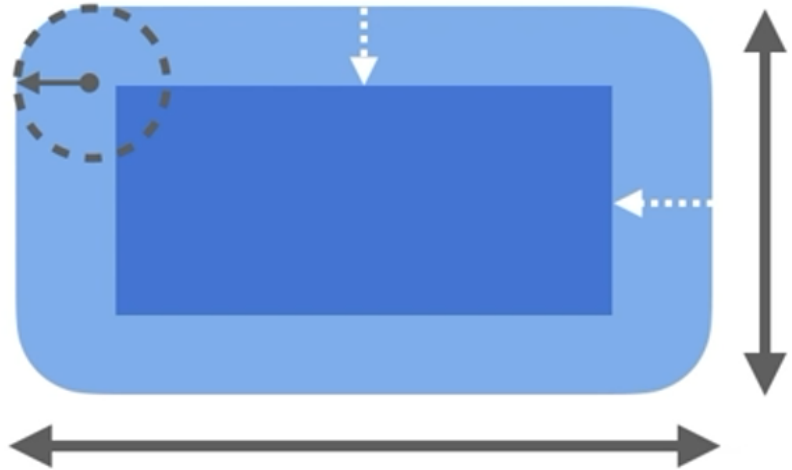
띄움

애니메이션

그리드

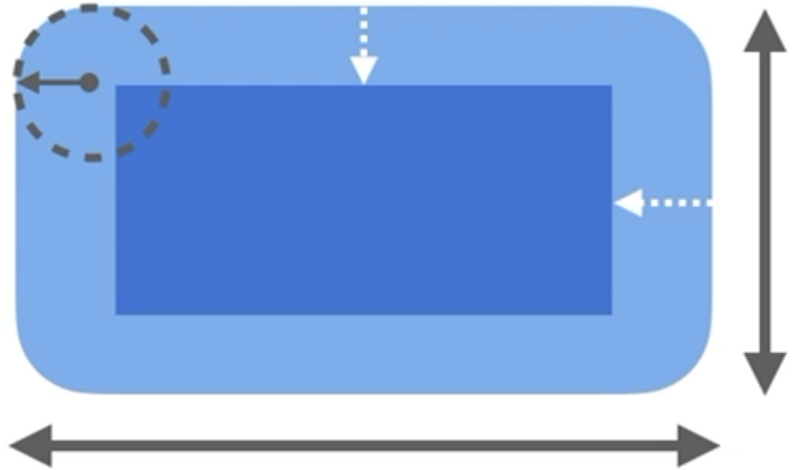
다단

필터



박스 모델

글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터



박스 모델

글꼴, 문자

배경

배치

플렉스(정렬)

전환

변환

띄움

애니메이션

그리드

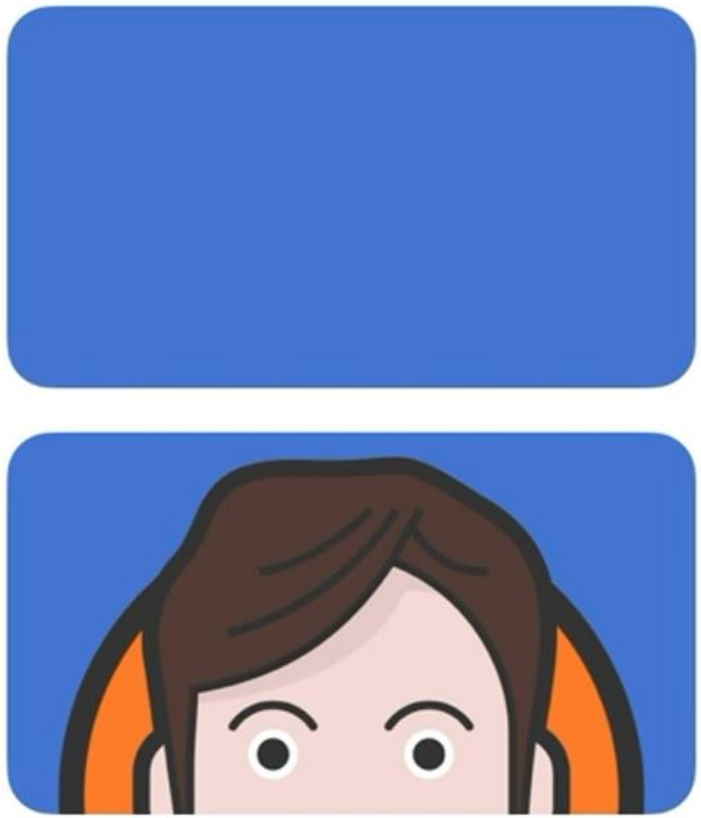
다단

필터

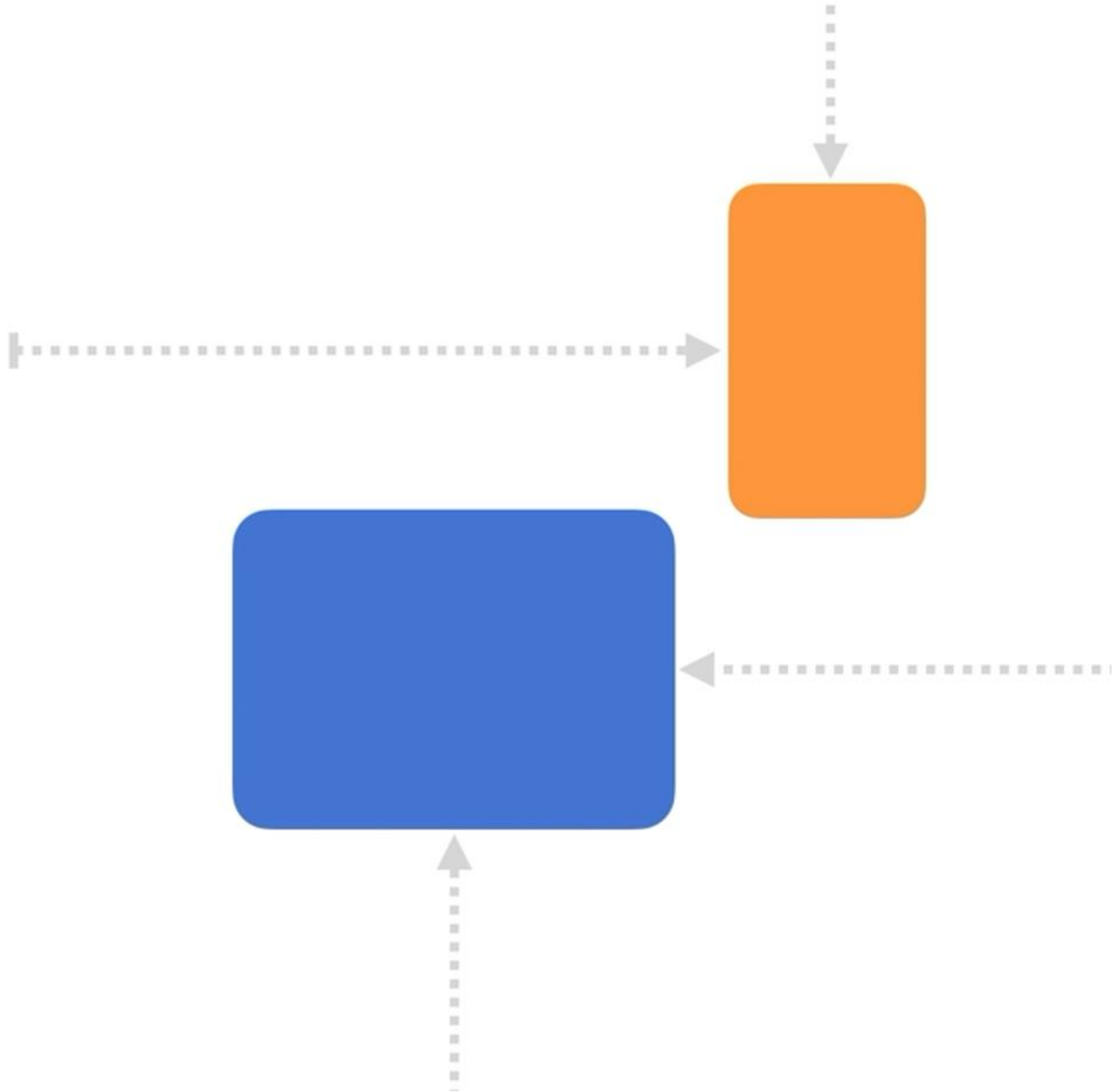
Hello **world**

Good morning~

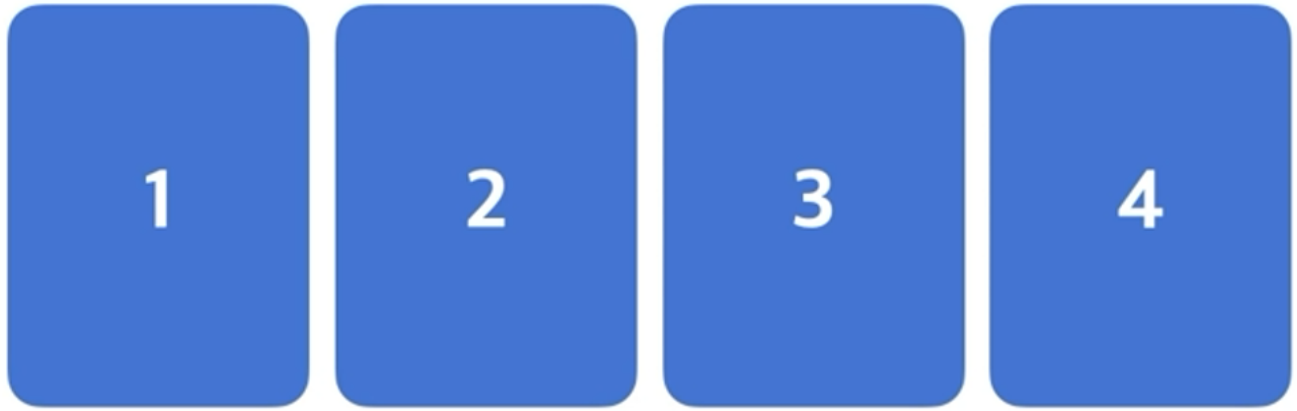
박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터



박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

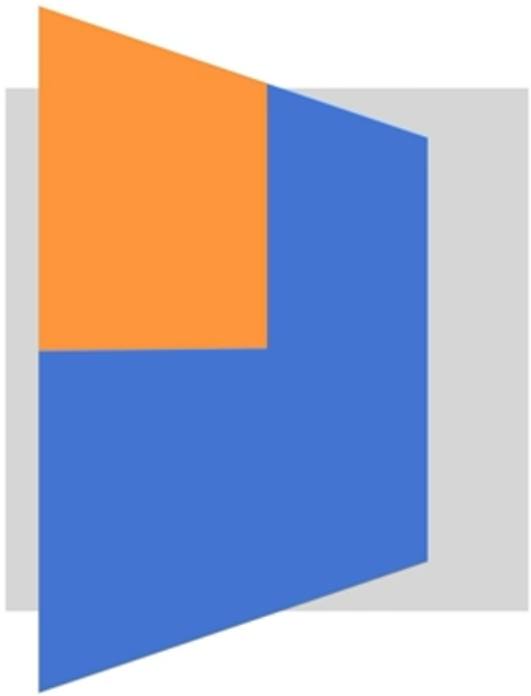


박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

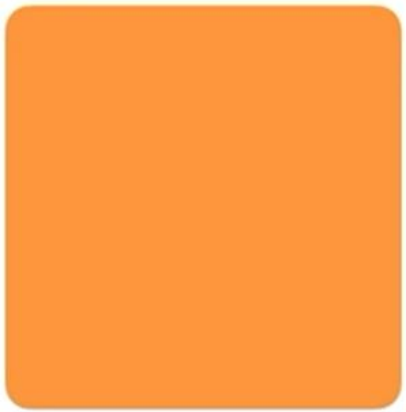


박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

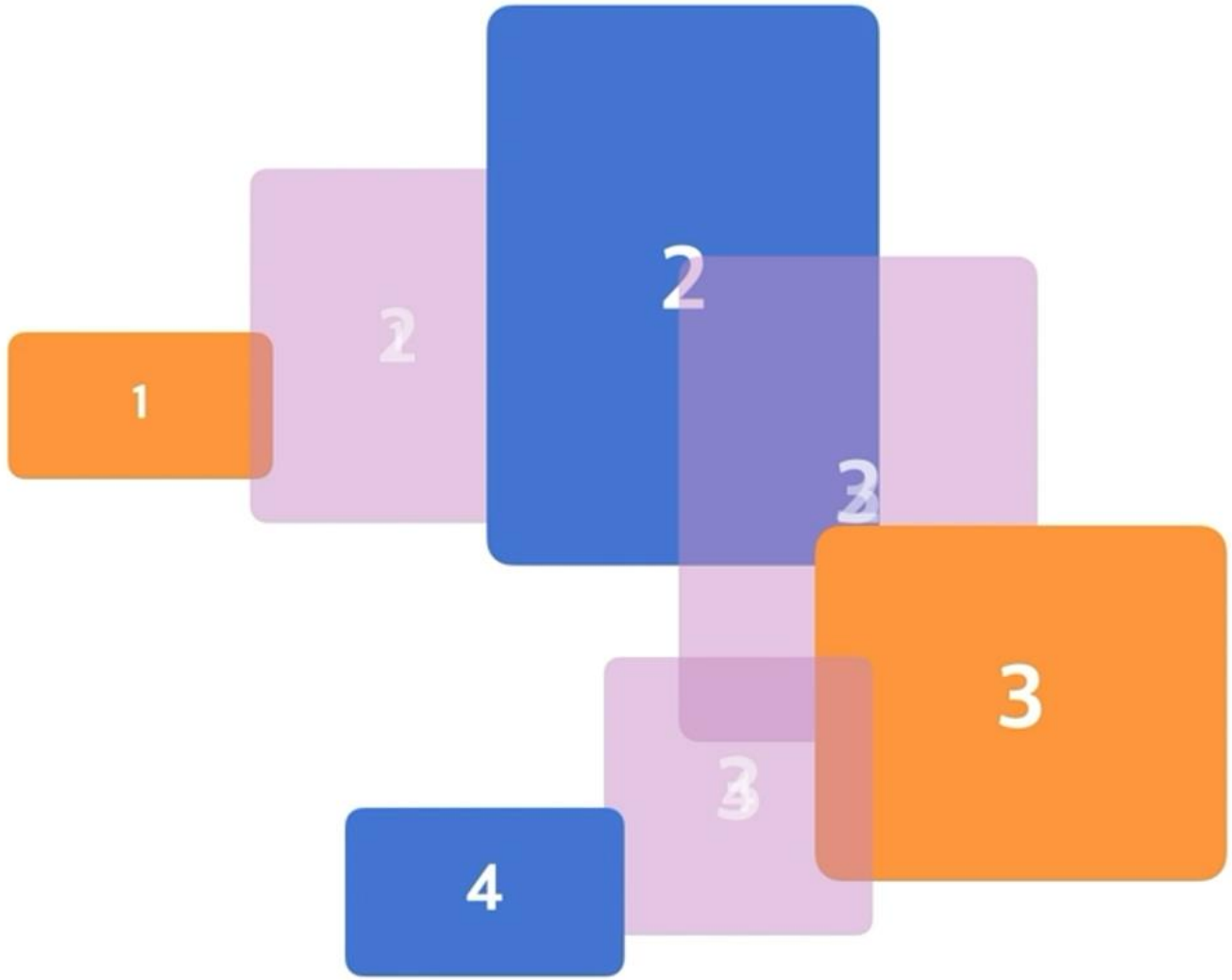


박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

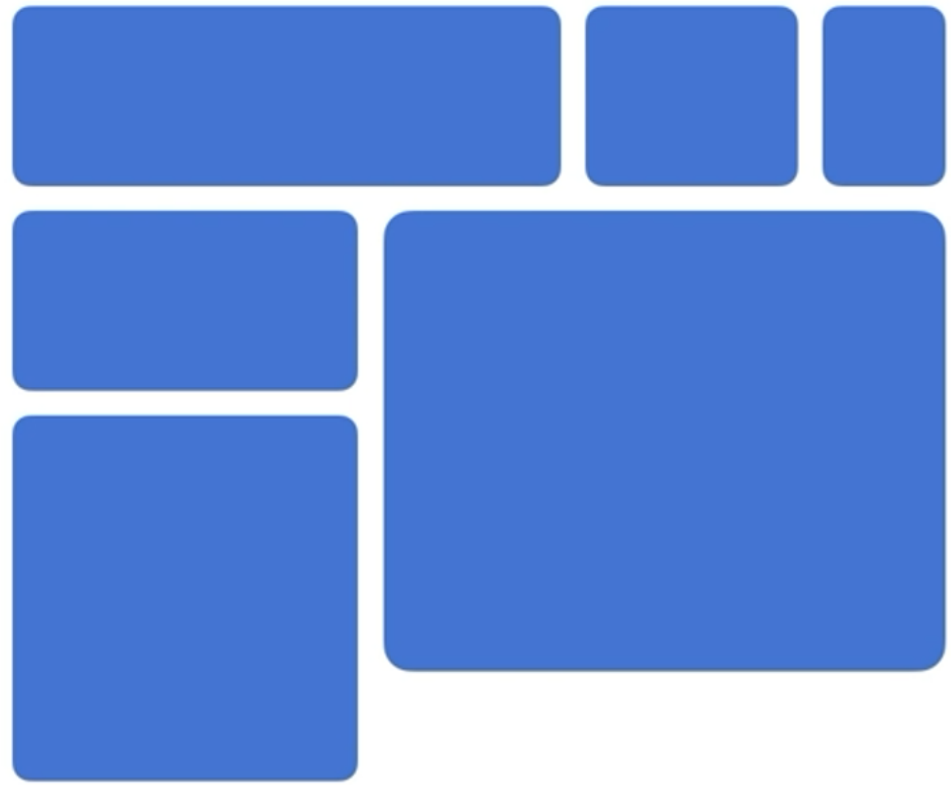


Lorem Ipsum is simply dummy text of the printing and Ipsum has been the industry's standard dummy text ever since the 1500s, when an

박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터



박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터



박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type

박스 모델
글꼴, 문자
배경
배치
플렉스(정렬)
전환
변환
띄움
애니메이션
그리드
다단
필터

박스 모델

글자와 상자

요소가 화면에 출력되는 특성, 크게 2가지로 구분.

인라인(Inline) 요소 : 글자를 만들기 위한 요소들.
블록(Block) 요소 : 상자(레이아웃)를 만들기 위한 요소들.

Inline 요소

```
<span>Hello</span>
<span>World</span>
```

****

대표적인 인라인 요소!
본질적으로 아무것도 나타내지 않는,
콘텐츠 영역을 설정하는 용도.

Hello World

요소가 수평으로 쌓임

```
<span>Hello</span>
<span>World</span>
```

****
대표적인 **인라인** 요소!
본질적으로 아무것도 나타내지 않는,
콘텐츠 영역을 설정하는 용도.

auto
포함한 콘텐츠 크기만큼 자동으로 **줄어듦!**

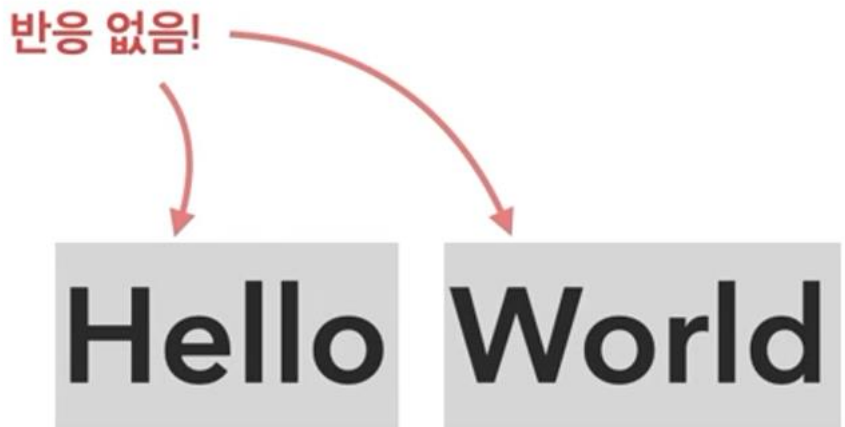
포함한 콘텐츠 크기만큼
자동으로 **줄어듦!**
auto

Hello World

요소의 가로 너비를 지정하는 CSS 속성

```
<span style="width: 100px;">Hello</span>
<span style="height: 100px;">World</span>
```

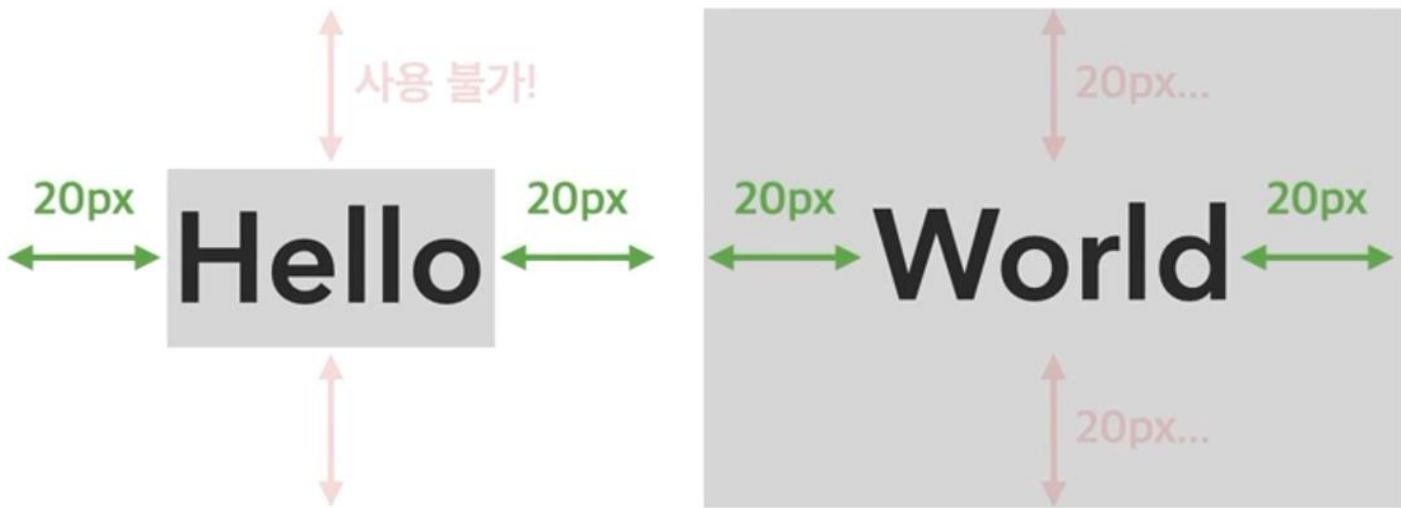
요소의 세로 너비를 지정하는 CSS 속성



요소의 외부 여백을 지정하는 CSS 속성

```
<span style="margin: 20px 20px;">Hello</span>
<span style="padding: 20px 20px;">World</span>
```

요소의 내부 여백를 지정하는 CSS 속성



인라인 요소

블록 요소

불가!

```
<span><div></div></span>
```

```
<span><span></span></span>
```

가능!

색을 사용하는 모든 속성에 적용 가능한 색상 표현

색상 표현

색상 이름	브라우저에서 제공하는 색상 이름	red, tomato, royalblue
Hex 색상코드	16진수 색상(Hexadecimal Colors)	#000, #FFFFFF
RGB	빛의 삼원색	rgb(255, 255, 255)
RGBA	빛의 삼원색 + 투명도	rgba(0, 0, 0, 0.5)
HSL	색상, 채도, 명도	hsl(120, 100%, 50%)
HSLA	색상, 채도, 명도 + 투명도	hsla(120, 100%, 50%, 0.3)

Block 요소

```
<div>Hello</div>
<div>World</div>
```

** **

대표적인 블록 요소!
본질적으로 아무것도 나타내지 않는,
콘텐츠 영역을 설정하는 용도.

요소가
수직으로 쌓임

Hello

World

```
<div>Hello</div>
<div>World</div>
```

** **

대표적인 블록 요소!
본질적으로 아무것도 나타내지 않는,
콘텐츠 영역을 설정하는 용도.

auto

부모 요소의 크기만큼 자동으로 **늘어남!**

Hello

World

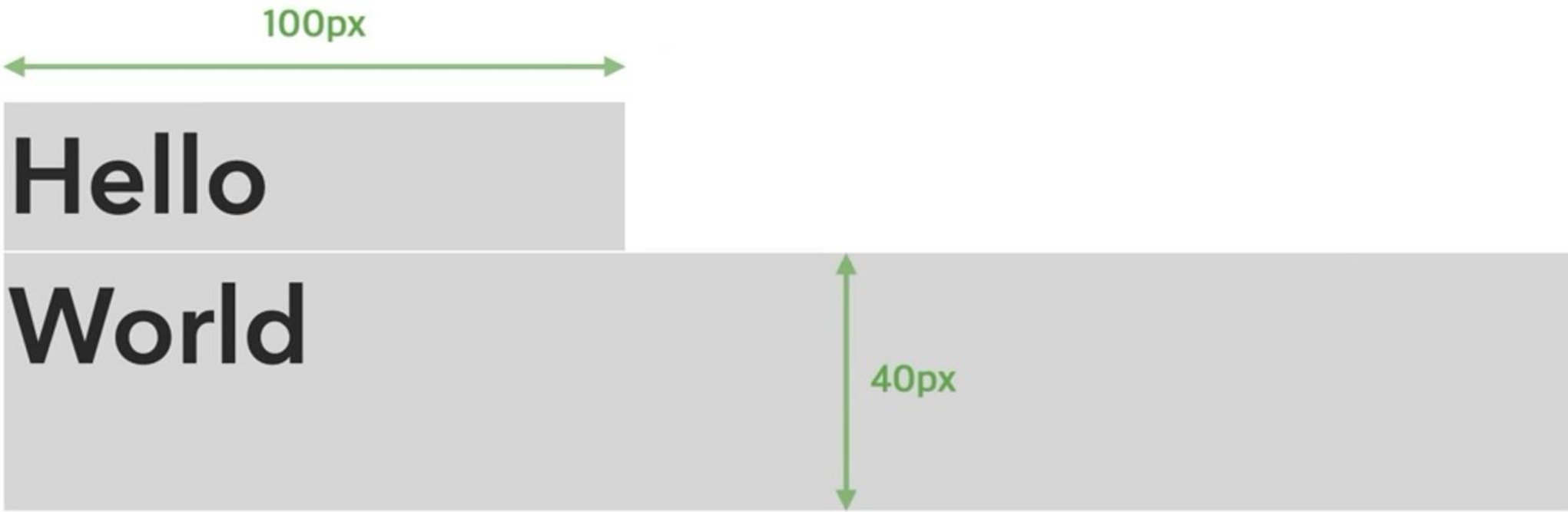
포함한 콘텐츠 크기만큼
자동으로 **줄어듬!**

auto

요소의 가로 너비를 지정하는 CSS 속성

```
<div style="width: 100px;">Hello</div>
<div style="height: 40px;">World</div>
```

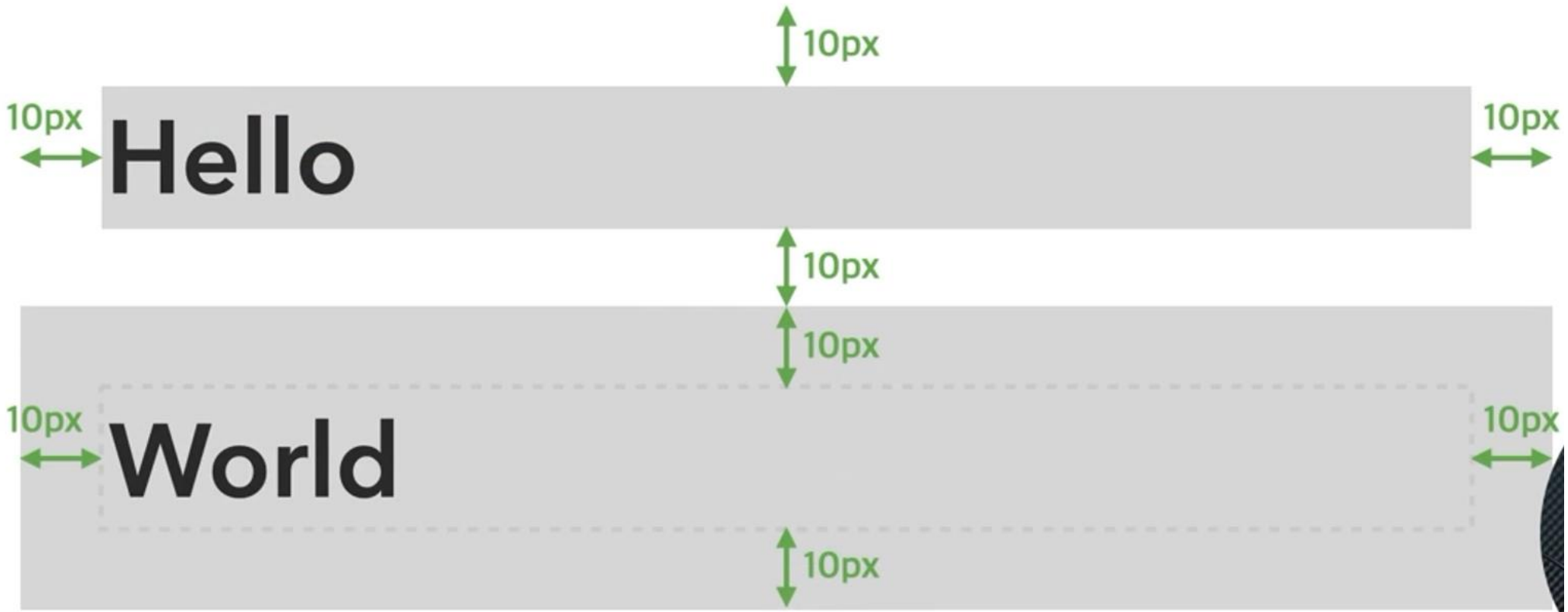
요소의 세로 너비를 지정하는 CSS 속성



요소의 외부 여백을 지정하는 CSS 속성

```
<div style="margin: 10px;">Hello</div>
<div style="padding: 10px;">World</div>
```

요소의 내부 여백를 지정하는 CSS 속성



블록 요소

가능!

```
<div><div></div></div>
<div><span></span></div>
```

가능!

인라인 요소

inline, block

inline

block

Inline-block?

- Block 요소가 수직으로만 쌓이는 문제점을 해결하고자 나온 개념
- 디자인이 중요시 되면서 가로로 블록을 나열 하는 경우가 많이 생겼는데 span 태그는 inline 요소라서 해당 구역을 꾸미기가 어려움!
- 따라서, 둘의 특징을 합쳐서 만든 inline-block!

```
<style>
  span {
    display: inline-block;
    width: 300px;
    height: 300px;
    color: white;
    background-color: blue;
  }
</style>

<body>
    <span>Inline-block</span>
    <span>Inline-block</span>
</body>
```

Inline-block Inline-block

inline, block, inline-block

	inline	block	inline-block
기본 넓이	컨텐츠 만큼	부모의 넓이	컨텐츠 만큼
width, height	무시	적용	적용
가로 공간 차지	공유	독점	공유
margin	가로만	가로, 세로 전부 *세로 상쇄	가로, 세로 전부
padding	가로만, *세로는 배경색만	가로, 세로 전부	가로, 세로 전부

```
<style>
  html {
    font-size: 20px;
    font-weight: bold;
  }
  span {
    color: white;
    background-color: blue;
  }
  div {
    color: white;
    background-color: red;
  }
</style>
```

```
<body>
  <span>span 요소</span>
  <div>div 요소</div>
  <span>span 요소</span>
  <span>span 요소</span>
  <div>div 요소</div>
  <div>div 요소</div>
  <span>span 요소</span>
  <span>span 요소</span>
  <span>span 요소</span>
  <div>div 요소</div>
  <div>div 요소</div>
  <div>div 요소</div>
  <br />
  HTML 컨텐츠 사이에 들어있는 <span>span 요소</span> 입니다! <br /><br />
  <div>div 요소 안에 들어있는 <span>span 요소</span> 입니다!</div>
  <br />
  <span
    >span 요소 안에 들어있는
    <div>div 요소</div>
    입니다!</span
  >
</body>
```

inline, block, inline-block

span 요소
div 요소
span 요소 span 요소
div 요소
div 요소
span 요소 span 요소 span 요소
div 요소
div 요소
div 요소

HTML 컨텐츠 사이에 들어있는 span 요소 입니다!

div 요소 안에 들어있는 span 요소 입니다!

span 요소 안에 들어있는
div 요소
입니다!

inline, block, inline-block

	inline	block	inline-block
기본 넓이	컨텐츠 만큼	부모의 넓이	컨텐츠 만큼
width, height	무시	적용	적용
가로 공간 차지	공유	독점	공유
margin	가로만	가로, 세로 전부 *세로 상쇄	가로, 세로 전부
padding	가로만, *세로는 배경색만	가로, 세로 전부	가로, 세로 전부

inline padding, 세로는 배경색만?

```
<style>
  html {
    font-size: 20px;
    font-weight: bold;
  }
  span {
    color: white;
    background-color: blue;
    padding: 10px;
  }
  div {
    color: white;
    background-color: red;
  }
</style>
```

block margin, 세로 상쇄!?

block margin, 세로 상쇄!?

```
<style>
  html {
    font-size: 20px;
    font-weight: bold;
  }
  span {
    display: inline-block;
    margin: 10px;
    padding: 10px;
    color: white;
    background-color: blue;
  }
  div {
    margin: 10px;
    padding: 10px;
    color: white;
    background-color: red;
  }
</style>
```

```
<body>
  <div>세로 상쇄 테스트</div>
  <div>세로 상쇄 테스트</div>
  <br /><br />
  <span>세로 상쇄 테스트</span><br />
  <span>세로 상쇄 테스트</span>
</body>
```

Width, Height

요소의 가로/세로 너비

width, height

기본값
(요소에 이미 들어있는 속성의 값)

auto 브라우저가 너비를 계산

단위 px, em, vw 등 단위로 지정

CSS 크기 단위

표현 단위

단위

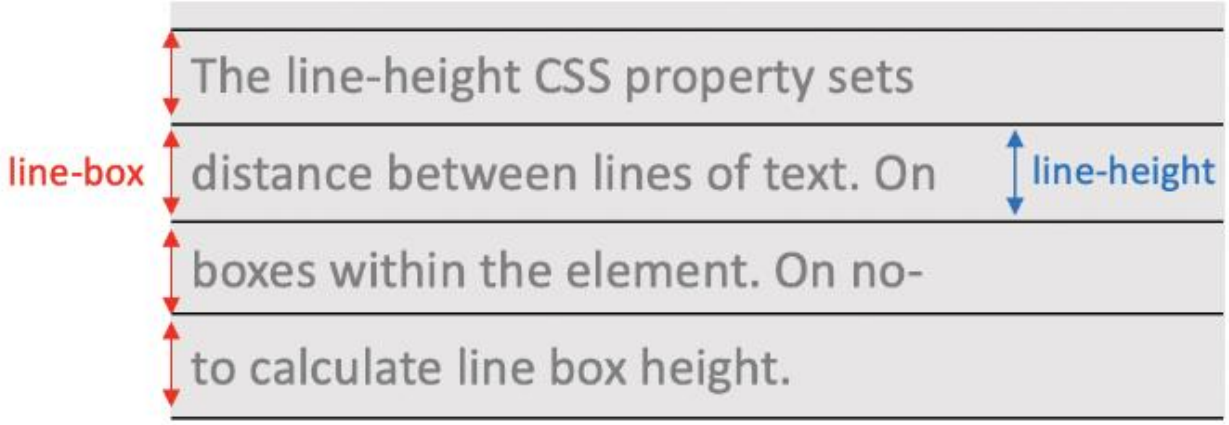
px	픽셀
%	상대적 백분율
em	요소의 글꼴 크기
rem	루트 요소(html)의 글꼴 크기
vw	뷰포트 가로 너비의 백분율
vh	뷰포트 세로 너비의 백분율

Calc()

- 사용자가 원하는 크기 값을 계산하여 적용
- Ex
  - Calc(100vh – 20vw)
  - Calc(1920px – 10vw)

line-height

- 영역 요소 내부 컨텐츠 글자의 줄 높이
- Box model의 크기 단위 적용 가능(font 에도 적용 가능!)
- 컨텐츠가 1줄인 경우 box height 와 line height 를 동일하게 하면 세로 중앙 정렬 효과!



요소가 커질 수 있는 최대 가로/세로 너비

max-width, max-height

none	최대 너비 제한 없음
auto	브라우저가 너비를 계산
단위	px, em, vw 등 단위로 지정

```
<style>
  div {
    width: 50vw;
    max-width: 500px;
    background-color: orange;
  }
</style>

<body>
  <div>Max-width</div>
</body>
```

요소가 작아질 수 있는 최소 가로/세로 너비

min-width, min-height

0	최소 너비 제한 없음
auto	브라우저가 너비를 계산
단위	px, em, vw 등 단위로 지정

```
<style>
  div {
    width: 50vw;
    min-width: 300px;
    background-color: orange;
  }
</style>

<body>
  <div>Max-width</div>
</body>
```

실습, 크기 요소 사용하기

- div 를 현재 화면 넓이의 1/5 + 화면 높이의 1/5 인 정사각형으로 만들기, 배경색은 예쁘게!
- 정사각형의 화면 크기에 따라 변화하지만 한 변의 최대 길이가 500px 이상이 될 수 없도록 설정하기
- 정사각형의 화면 크기에 따라 변화하지만 한 변의 최소 길이가 300px 이하가 될 수 없도록 설정하기
- div 안에는 KDT 라는 컨텐츠를 삽입하고 세로 중앙 정렬 하기

Margin

요소의 외부 여백(공간)을 지정하는 단축 속성

margin

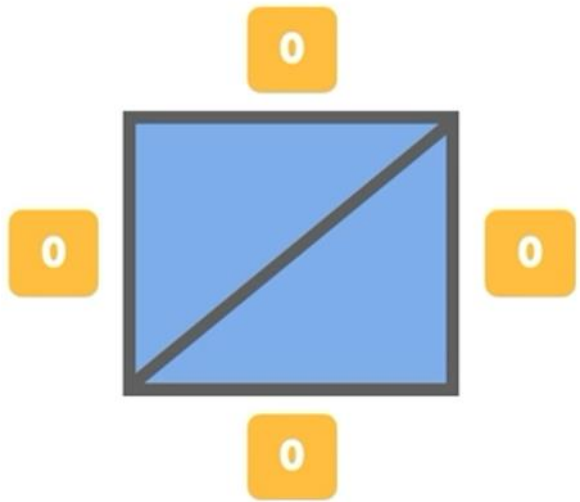
가로(세로) 너비가 있는 요소의
가운데 정렬에 활용해요!

음수를 사용할 수 있어요!

0	외부 여백 없음
auto	브라우저가 여백을 계산
단위	px, em, vw 등 단위로 지정
%	부모 요소의 가로 너비에 대한 비율로 지정

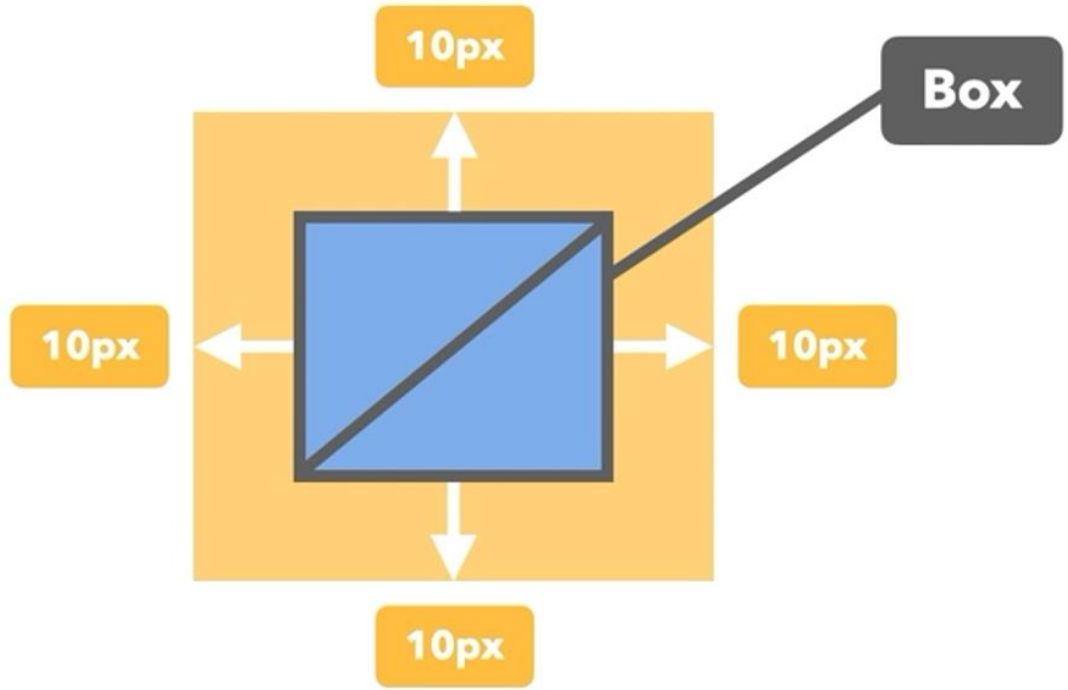
```
margin: 0;
```

top, right, bottom, left



```
margin: 10px;
```

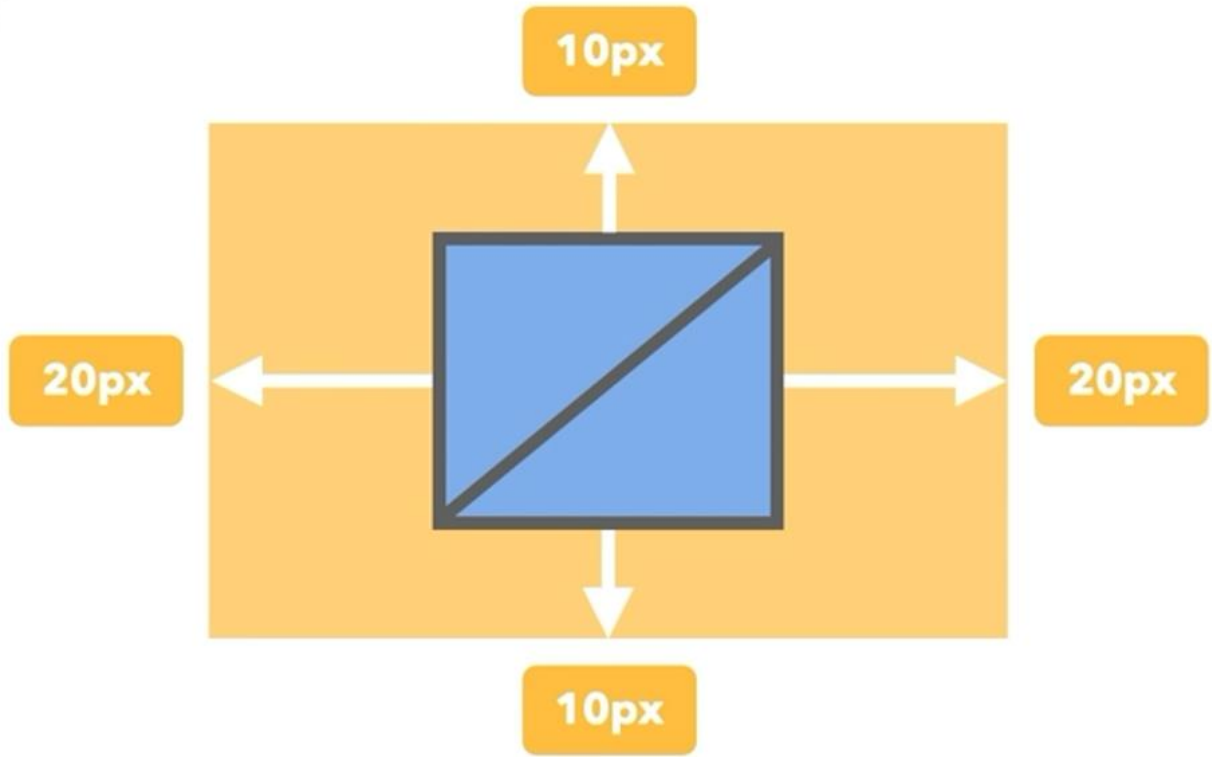
top, right, bottom, left



```
margin: 10px 20px;
```

top, bottom　left, right

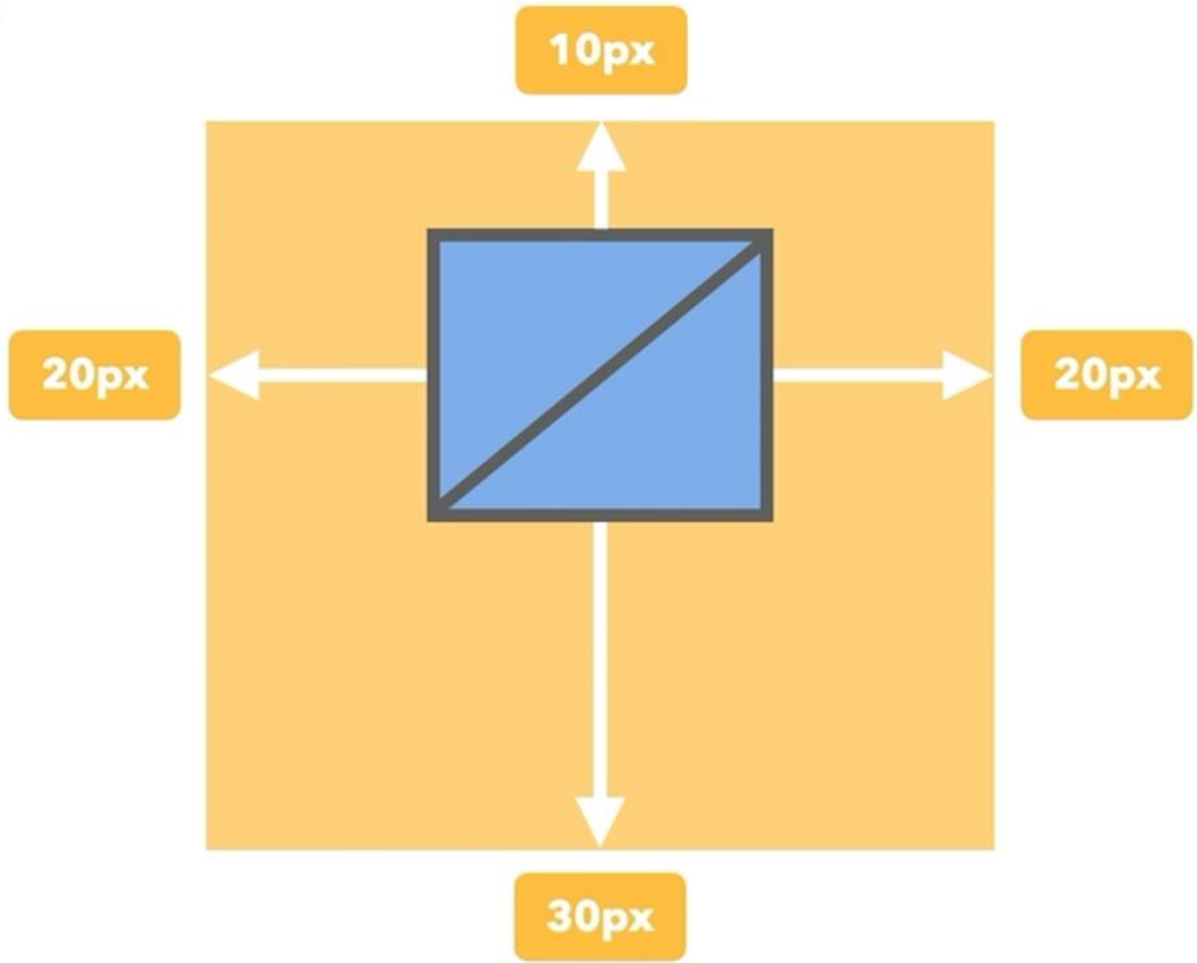
상하.좌우.



```
margin: 10px 20px 30px;
```

top | left, right | bottom

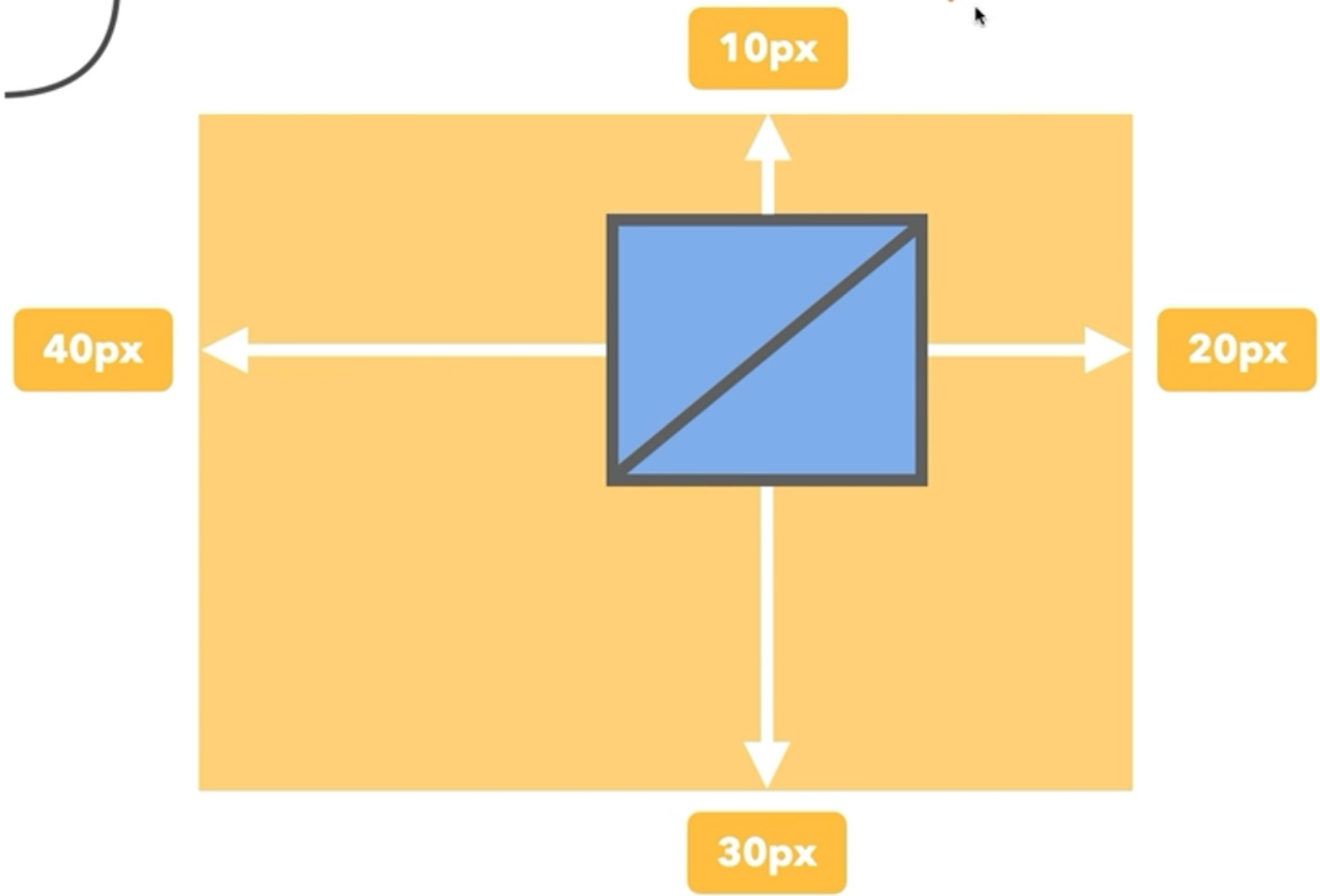
상.중.하



```
margin: 10px 20px 30px 40px;
```

top right bottom left

상.시계 방향



Margin, 단축 속성 정리

margin: top, right, bottom, left ;

margin: top, bottom left, right ;

margin: top left, right bottom ;

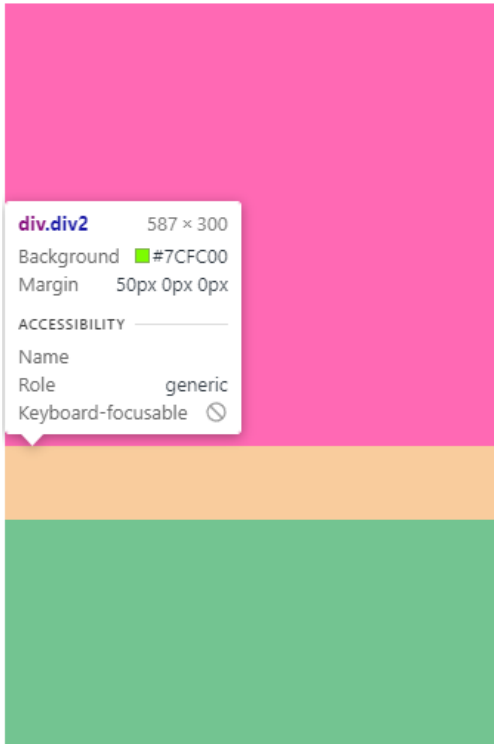
margin: top right bottom left ;

Margin, 개별 속성 정리

margin-top
margin-bottom
margin-left
margin-right

margin은 크게 남기는 쪽으로 기운다?

- 두 블록 요소의 margin 이 겹칠 경우 margin의 값이 큰 쪽만 반영




```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Inline & Block & Inlineblock</title>
    <style>
      .div1 {
        background-color: hotpink;
        height: 300px;
        margin-bottom: 20px;
      }
      .div2 {
        background-color: lawngreen;
        height: 300px;
        margin-top: 50px;
      }
    </style>
  </head>
  <body>
    <div class="div1"></div>
    <div class="div2"></div>
  </body>
</html>
```

실습, margin 단축 속성 전부 사용하기

- Div를 100px 100px 로 만들기, 배경은 예쁘게! x 5
- 전체 margin 을 10px
- 상하 20px, 좌우 30px
- 상 10px, 좌우 50px, 하 20px
- 상 10px, 우 20px, 하 30px, 좌 40px
- 상 40px, 우 30px, 하 20px, 좌 10px 을 개별 속성으로

Padding

요소의 내부 여백(공간)을 지정하는 단축 속성

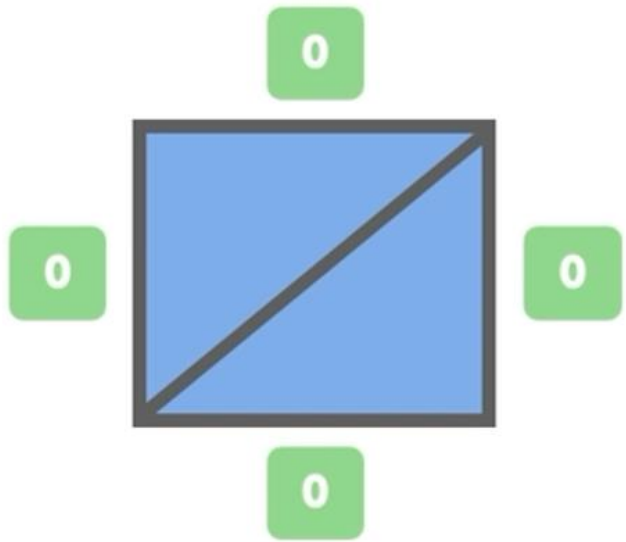
padding

요소의 크기가 커져요!

0	내부 여백 없음
단위	px, em, vw 등 단위로 지정
%	부모 요소의 가로 너비에 대한 비율로 지정

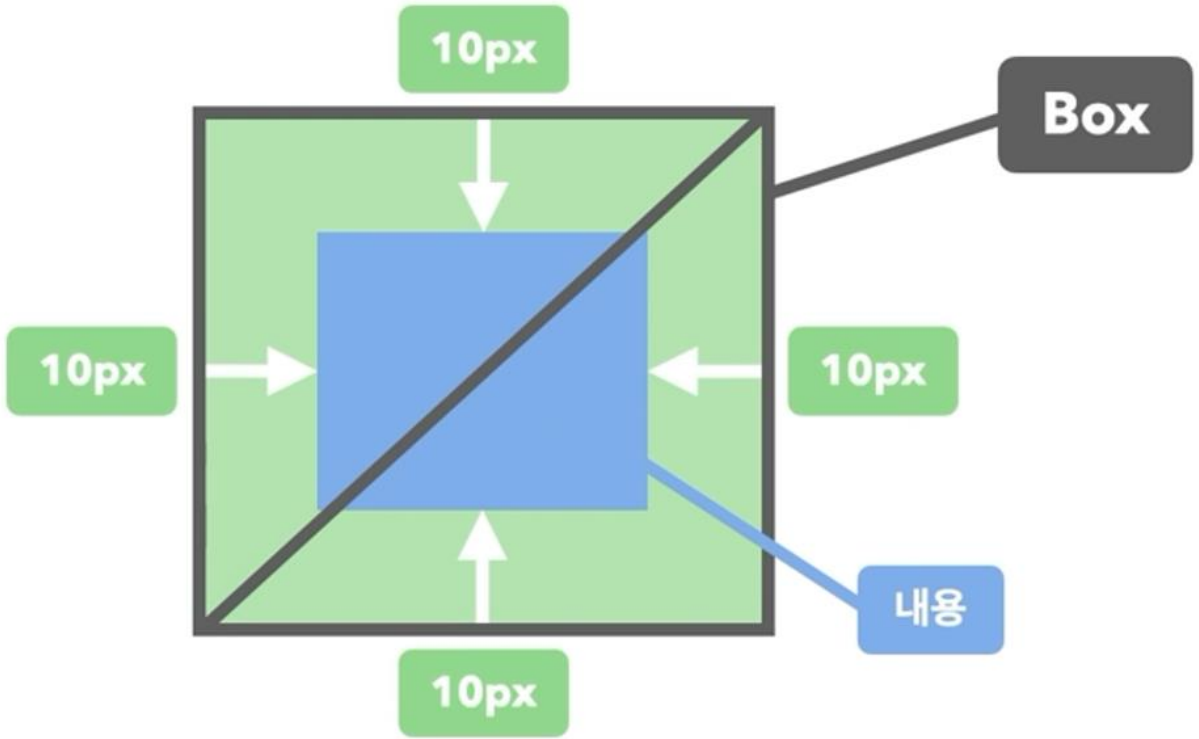
```
padding: 0;
```

top, right, bottom, left



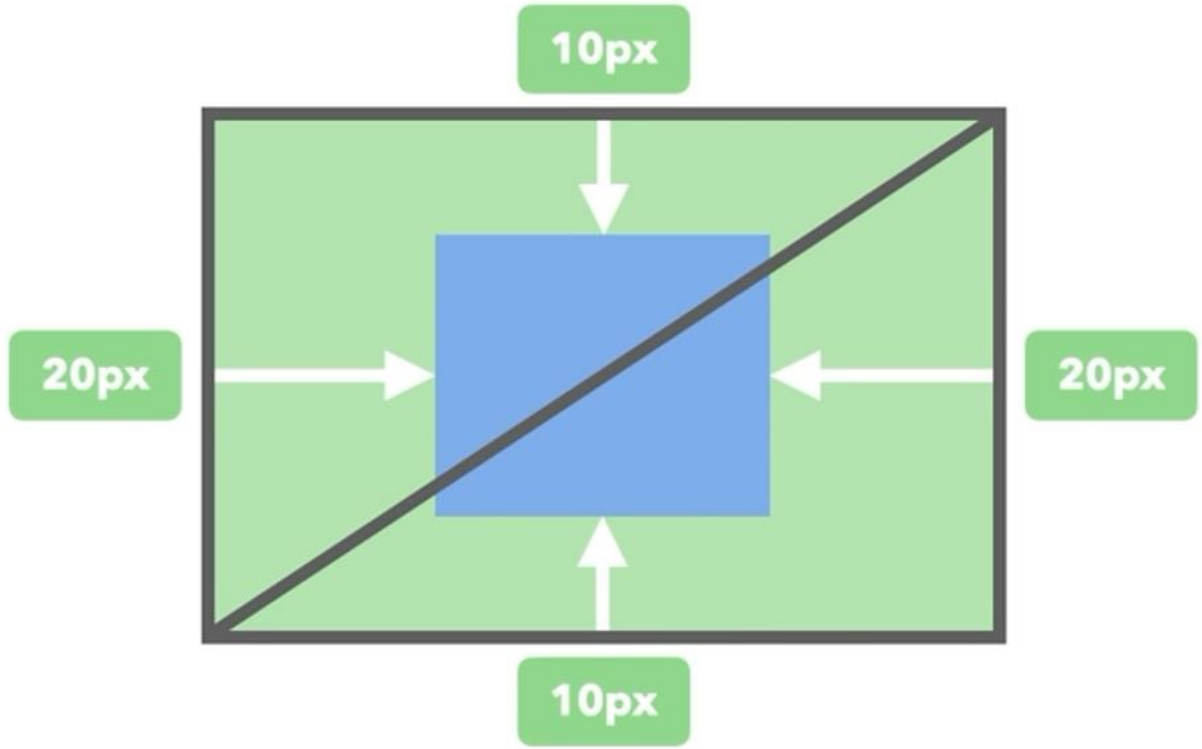
```
padding: 10px;
```

top, right, bottom, left



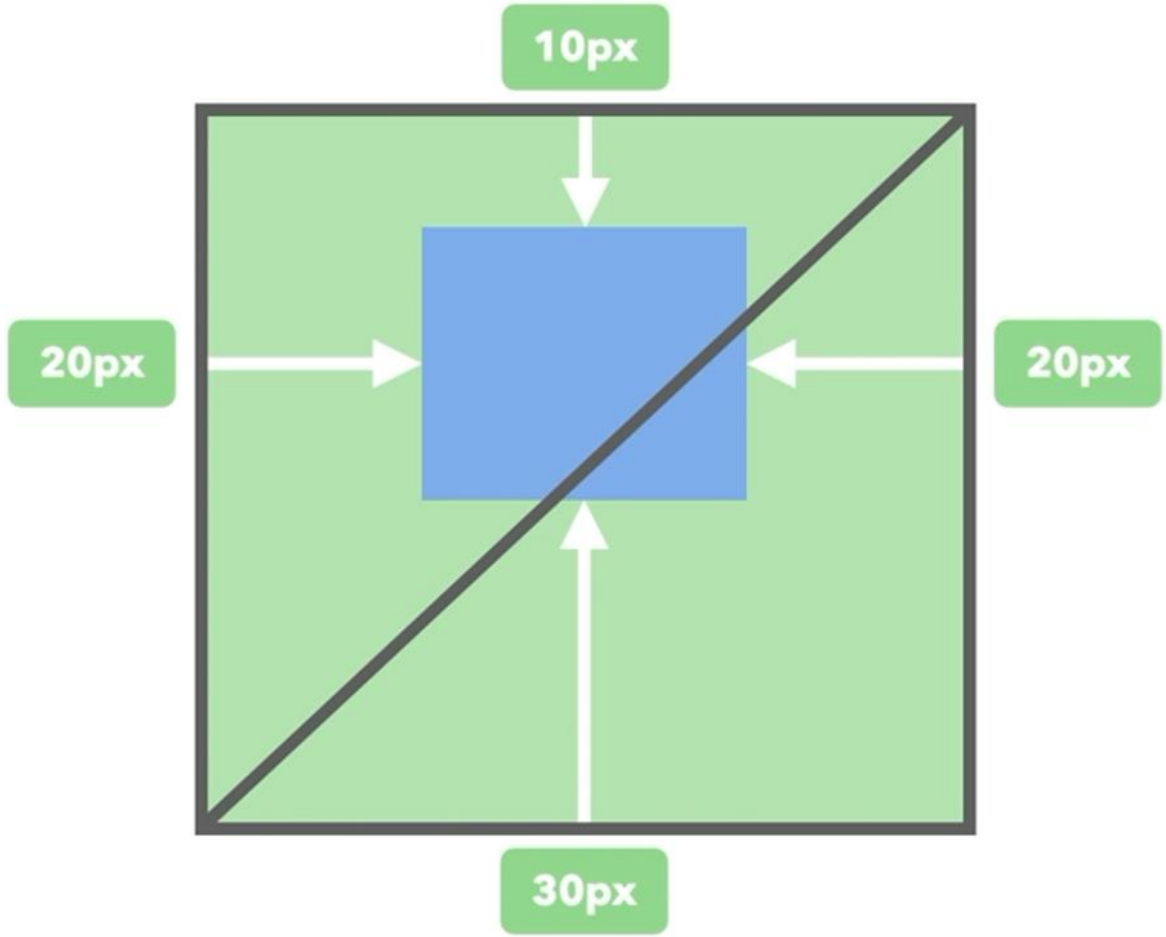
```
padding: 10px 20px;
```

top, bottom | left, right



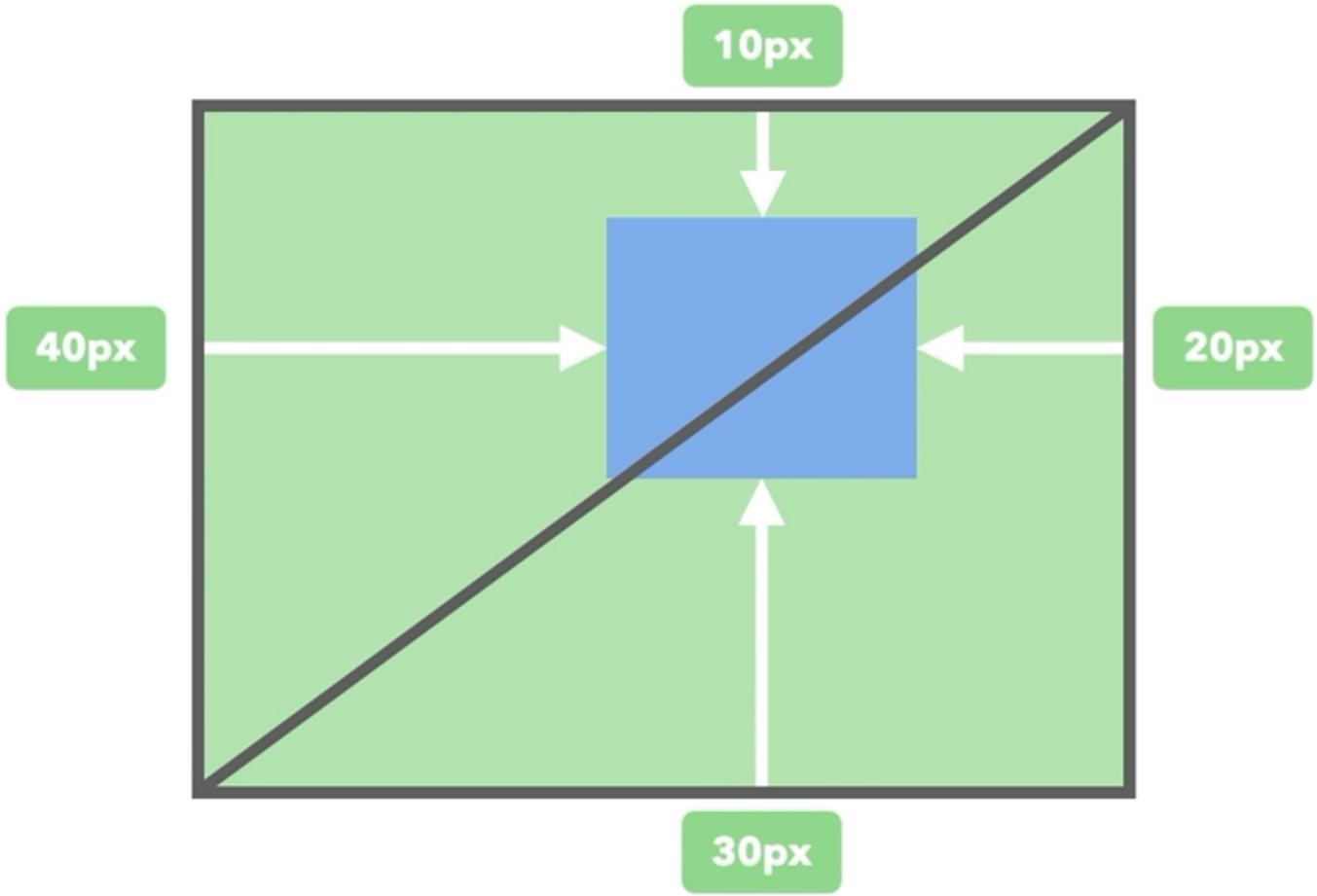
```
padding: 10px 20px 30px;
```

top | left, right | bottom



```
padding: 10px 20px 30px 40px;
```

top right bottom left



Padding, 단축 속성 정리

padding: top, right, bottom, left ;

padding: top, bottom left, right ;

padding: top left, right bottom ;

padding: top right bottom left ;

Padding, 개별 속성 정리

padding-top
padding-bottom
padding-left
padding-right

inline padding, 세로는 배경색만?

실습, padding 단축 속성 전부 사용하기

- Div를 100px 100px 로 만들기, 배경은 예쁘게! x 5
- 전체 padding 을 10px
- 상하 20px, 좌우 30px
- 상 10px, 좌우 50px, 하 20px
- 상 10px, 우 20px, 하 30px, 좌 40px
- 상 40px, 우 30px, 하 20px, 좌 10px 을 개별 속성으로

Border

요소의 크기가 커져요!

요소의 테두리 선을 지정하는 단축 속성

border-color

border: 선-두께 선-종류 선-색상;

border-width

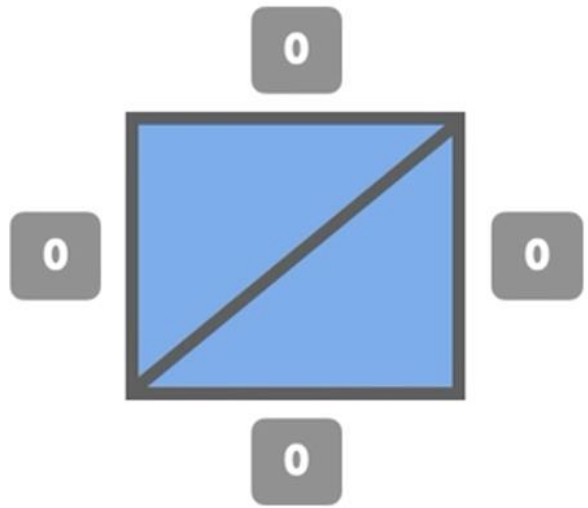
border-style

```
border: medium none ■black;
```

border-width | border-style | border-color

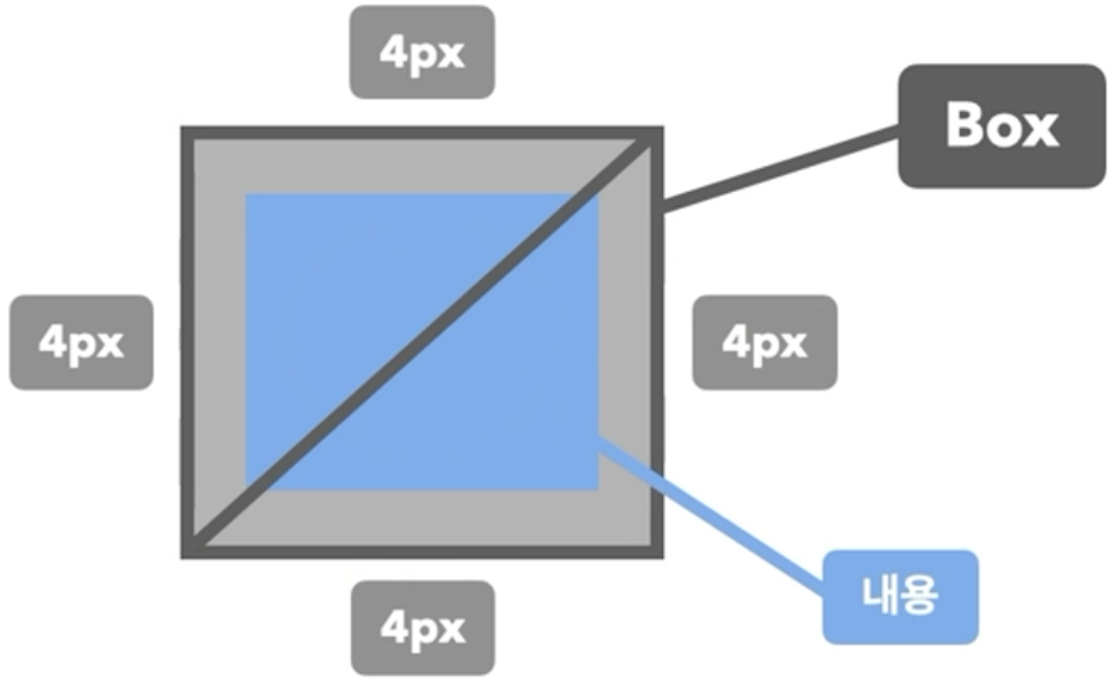
기본값
(요소에 이미 들어있는 속성의 값)

선의 종류가 없어서(none)
출력되지 않아요!



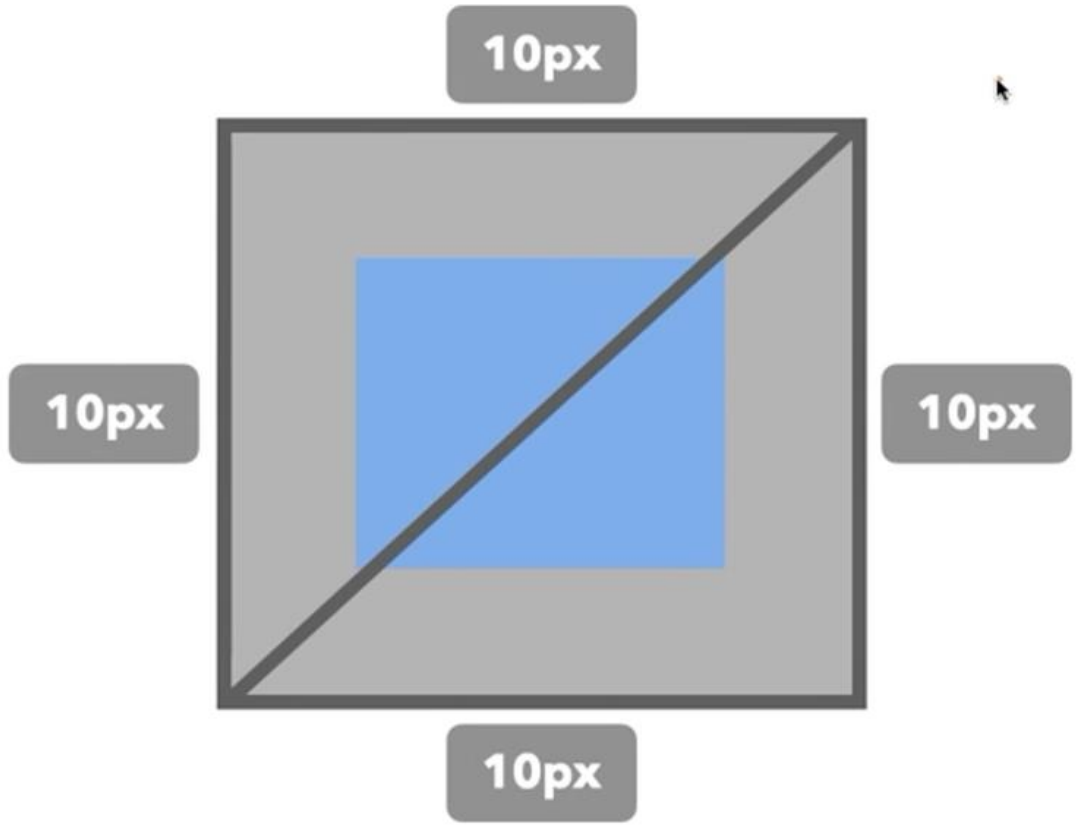
```
border: 4px solid ■black;
```

border-width | border-style | border-color



```
border: 10px solid black;
```

border-width **border-style** **border-color**



요소 테두리 선의 두께

border-width

medium	중간 두께
thin	얇은 두께
thick	두꺼운 두께
단위	px, em, % 등 단위로 지정

Border-width, 단축 속성 정리

```
border-width: top, right, bottom, left ;
border-width: top, bottom  left, right ;
border-width: top  left, right  bottom ;
border-width: top  right  bottom  left ;
```

요소 테두리 선의 종류

border-style

- **none** 선 없음
- **solid** 실선 (일반 선)
- **dotted** 점선
- **dashed** 파선
- **double** 두 줄 선
- **groove** 홈이 파여있는 모양
- **ridge** 솟은 모양 (groove의 반대)
- **inset** 요소 전체가 들어간 모양
- **outset** 요소 전체가 나온 모양

Border-style, 단축 속성 정리

```
border-style: top, right, bottom, left ;
border-style: top, bottom  left, right ;
border-style: top  left, right  bottom ;
border-style: top  right  bottom  left ;
```

요소 테두리 선의 색상을 지정하는 단축 속성

border-color

black	검정색
색상	선의 색상
transparent	투명

Border-color, 단축 속성 정리

border-color: top, right, bottom, left ;

border-color: top, bottom left, right ;

border-color: top left, right bottom ;

border-color: top right bottom left ;

요소의 테두리 선을 지정하는 기타 속성들

border-방향
border-방향-속성

border-top: 두께 종류 색상;
border-top-width: 두께;
border-top-style: 종류;
border-top-color: 색상;

border-right: 두께 종류 색상;
border-right-width: 두께;
border-right-style: 종류;
border-right-color: 색상;

실습, 테두리 그리기

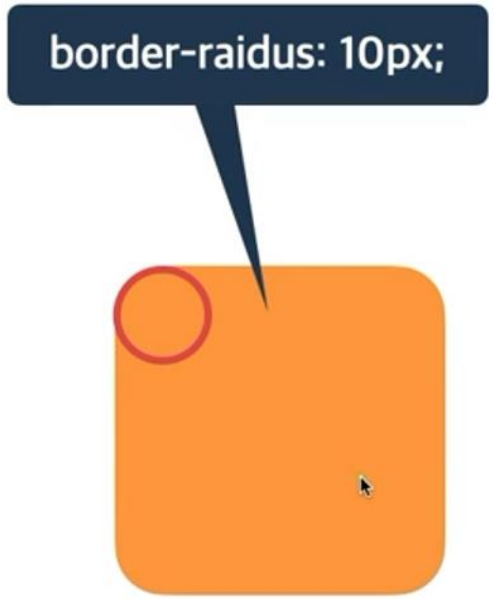
- Border 속성으로 테두리 적용하기 x 5 개(각각 클래스 부여)
- Border-width 속성으로 4 방향 테두리 두께 변경하기
- Border-style 속성으로 4 방향 테두리 스타일 변경하기
- Border-color 속성으로 4 방향 테두리 색상 변경하기
- Border-방향 속성으로 모든 방향의 테두리 스타일이 다른 div 완성하기

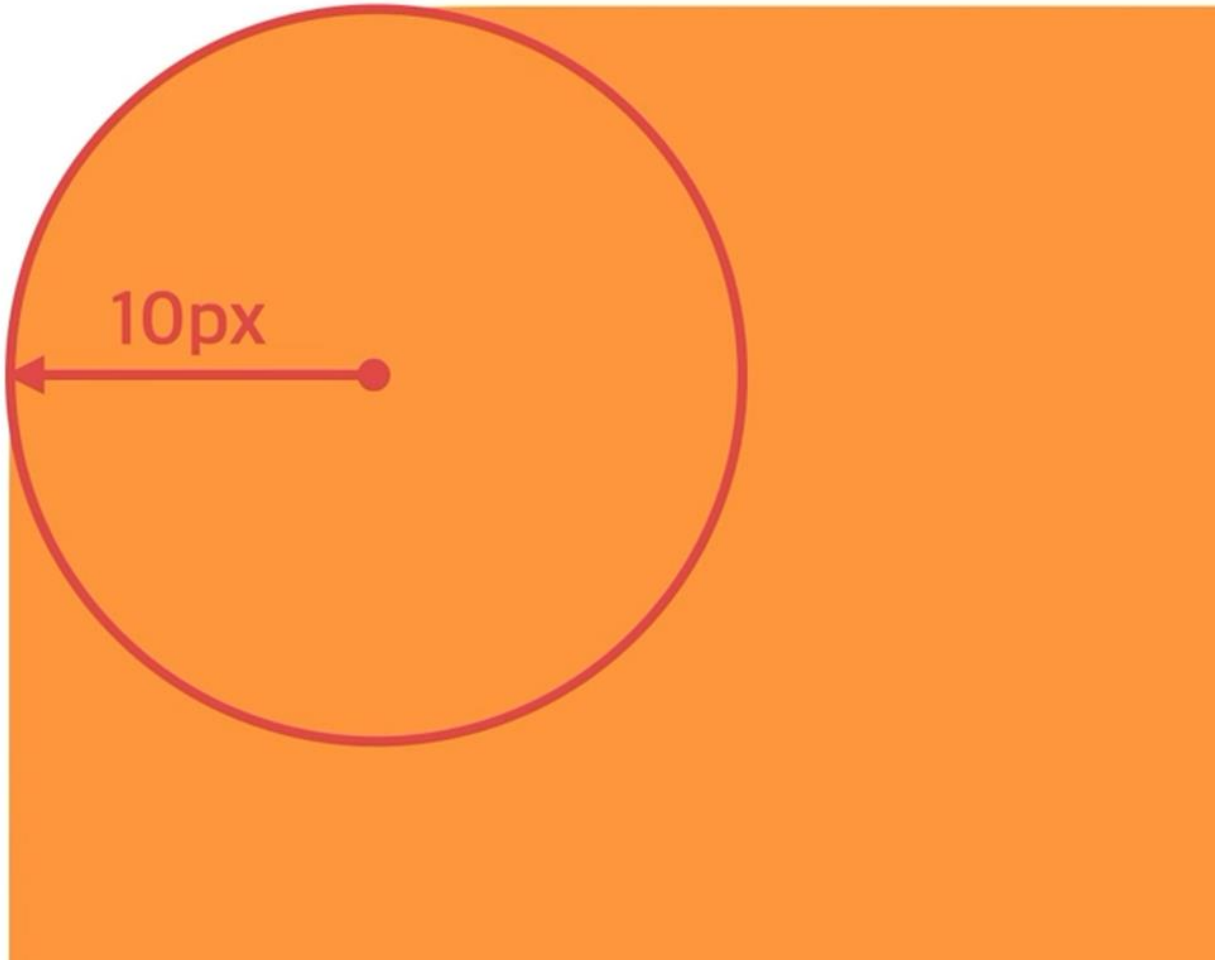
Border-radius

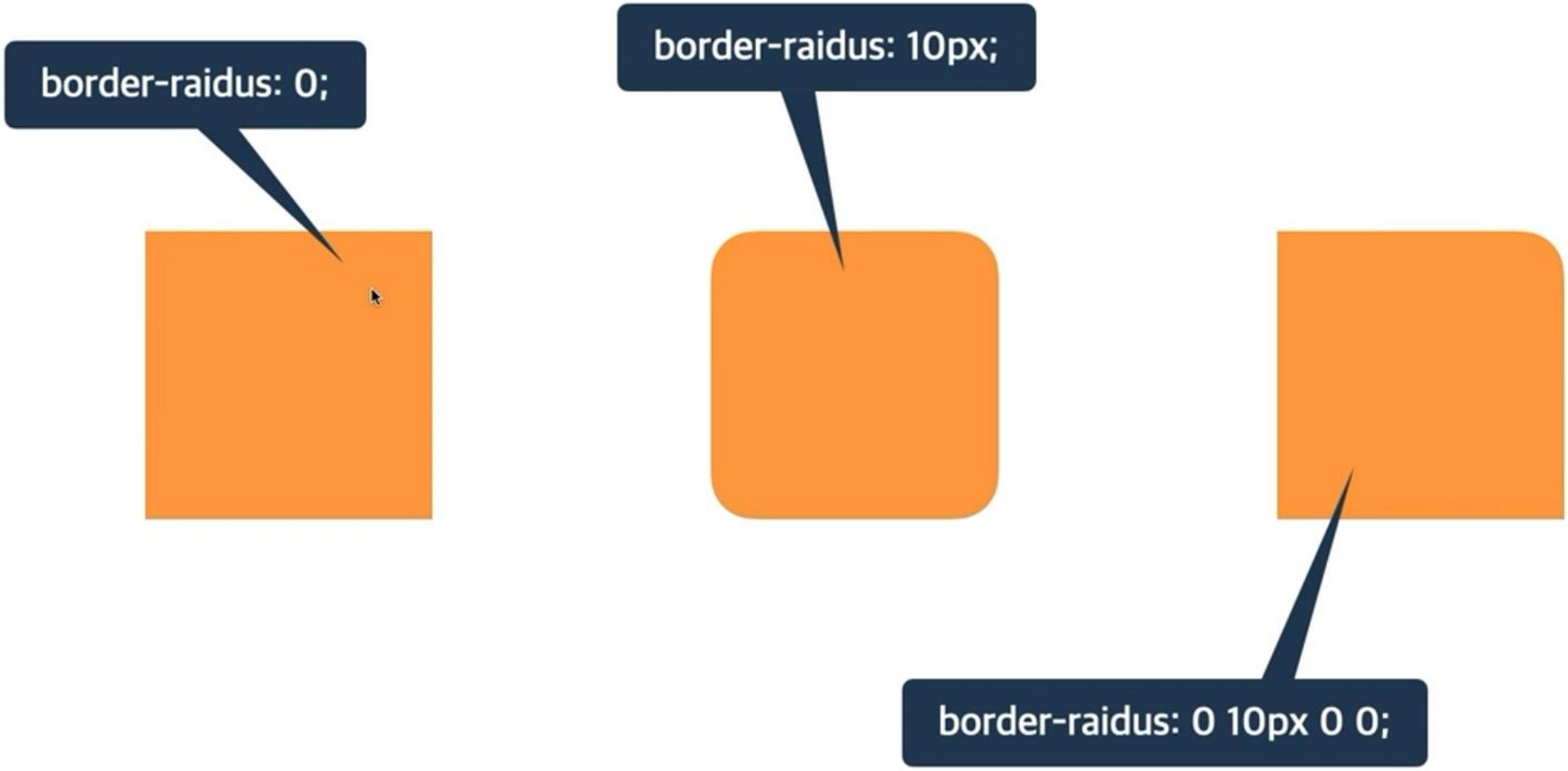
요소의 모서리를 둥글게 깎음

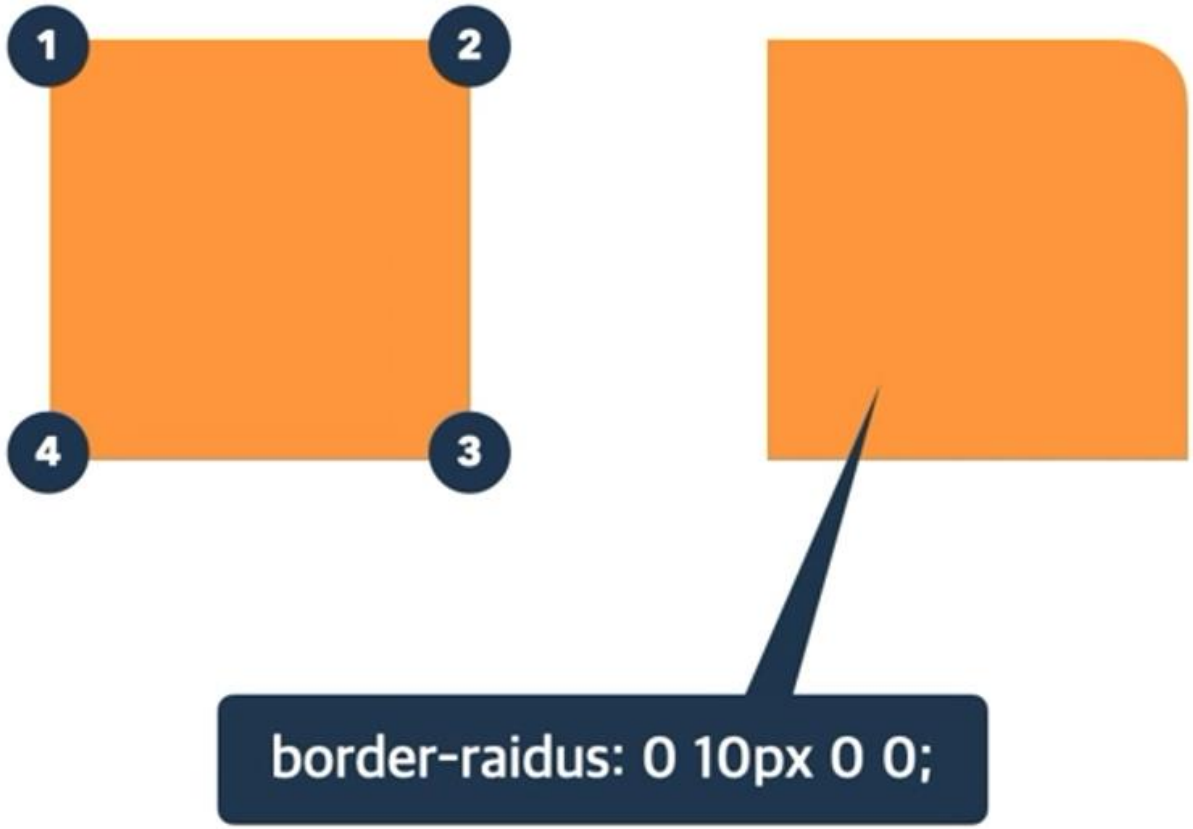
border-radius

0	둥글게 없음
단위	px, em, vw 등 단위로 지정


border-raidus: 10px;


10px


border-raidus: 0;
border-raidus: 10px;
border-raidus: 0 10px 0 0;


1
2
4
3
border-raidus: 0 10px 0 0;

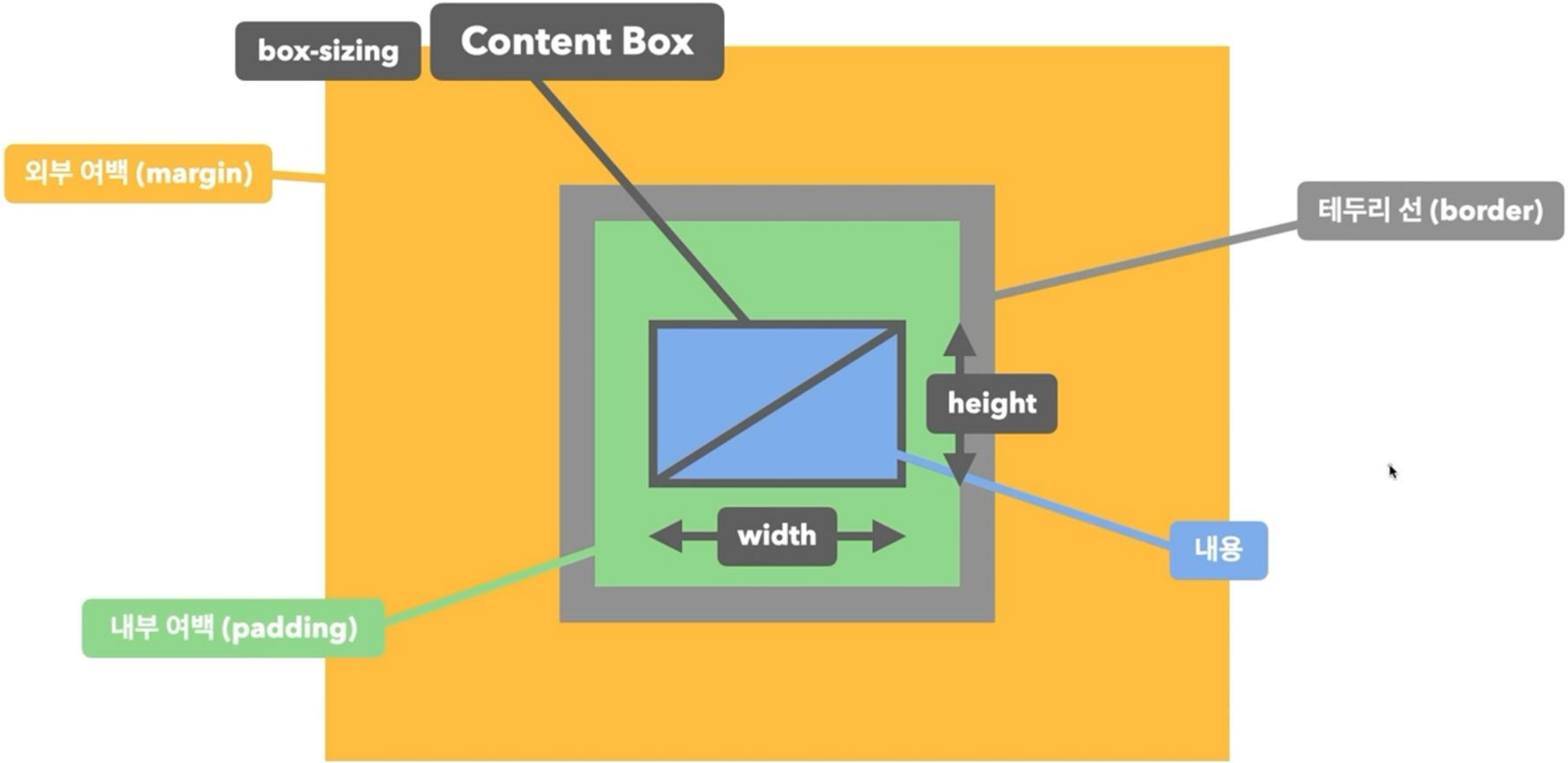
Box-sizing

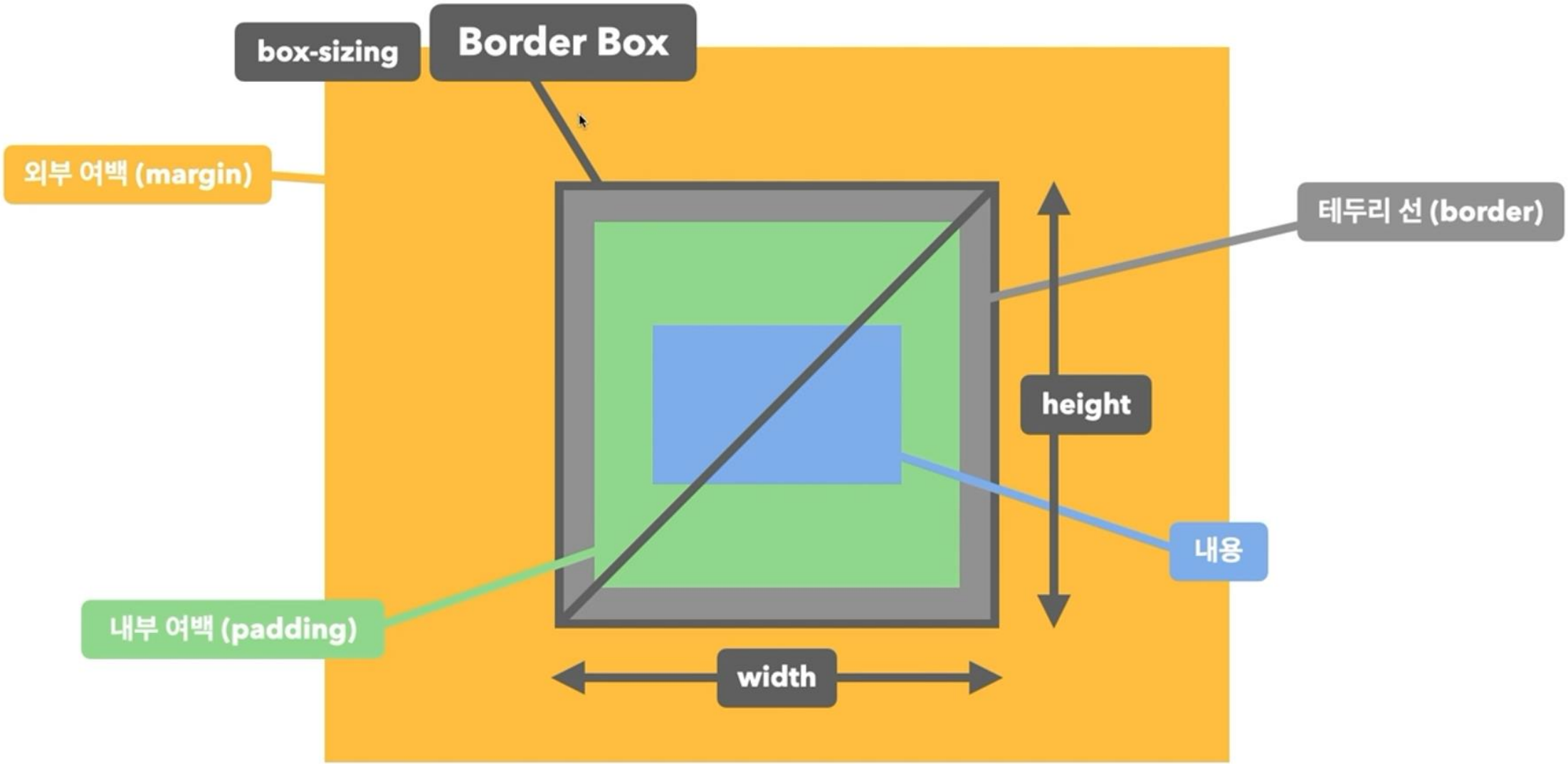
요소의 크기 계산 기준을 지정

box-sizing

content-box 요소의 내용(content)으로 크기 계산

border-box 요소의 내용 + padding + border로 크기 계산


box-sizing
Content Box
외부 여백 (margin)
테두리 선 (border)
height
width
내용
내부 여백 (padding)


box-sizing
Border Box
외부 여백 (margin)
테두리 선 (border)
height
내용
내부 여백 (padding)
width

실습, 크기 비교하기

- Width 100px, height 100px, padding 20px,border 10px solid red, 배경 오렌지인 div 2개 선언
- 하나는 content-box, 하나는 border-box 로 속성을 주고 둘의 크기를 비교

Overflow

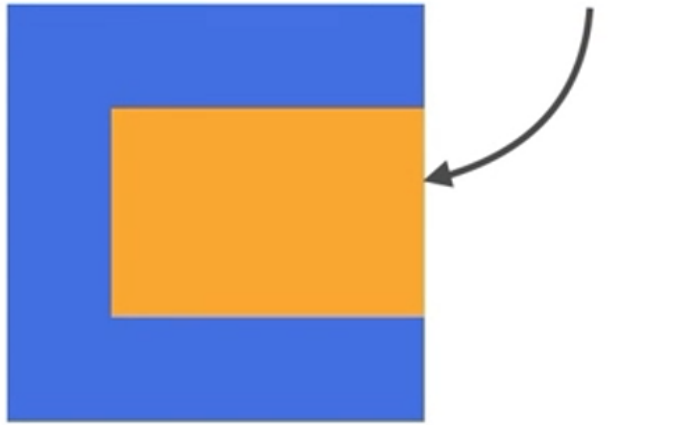
요소의 크기 이상으로 내용이 넘쳤을 때, 보여짐을 제어하는 단축 속성

overflow

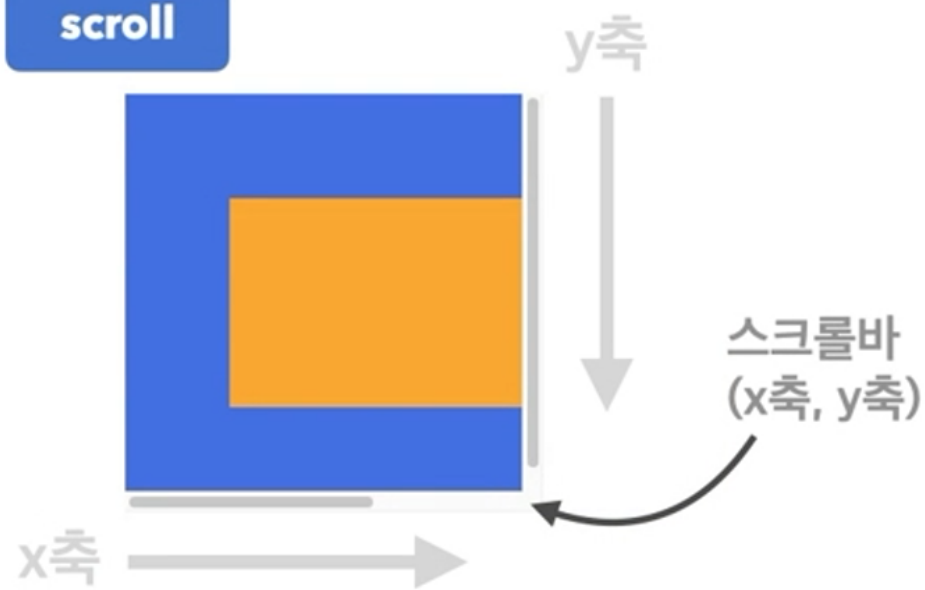
visible	넘친 내용을 그대로 보여줌
hidden	넘친 내용을 잘라냄
scroll	넘친 내용을 잘라냄, 스크롤바 생성
auto	넘친 내용이 있는 경우에만 잘라내고 스크롤바 생성

visible

hidden



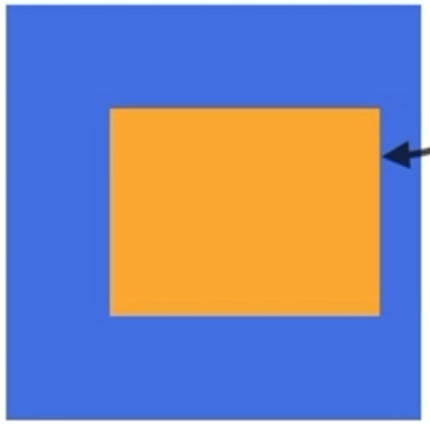
scroll



auto

visible

내용 넘치지 않음!



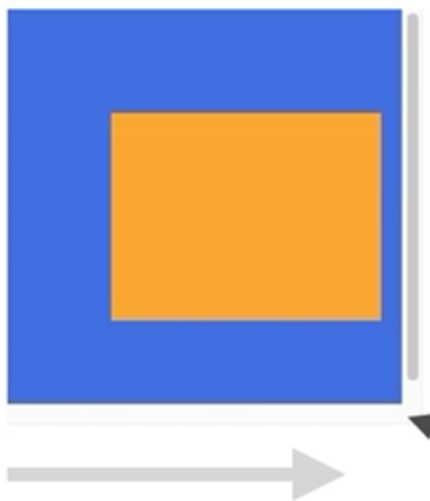
hidden

scroll

y축

스크롤바
(x축, y축)

x축



auto

요소의 크기 이상으로 내용이 넘쳤을 때, 보여짐을 제어하는 개별 속성들

overflow-x
overflow-y

실습, 오버 플로우 사용하기

- 부모 div 요소의 크기를 200px 200px 배경 파란색으로 설정
- 자식 div 요소의 크기를 400px 100px 배경 오랜지로 설정
- 위와 같은 케이스를 4개 생성 후, overflow 속성을 각각 적용
- 자식 요소의 크기를 100px 400px 로 변경 후, overflow 속성을 각각 적용

Display

요소의 화면 출력(보여짐) 특성

display

각 요소에 이미 지정되어 있는 값

- **block** 상자(레이아웃) 요소
- **inline** 글자 요소
- **inline-block** 글자 + 상자 요소

따로 지정해서 사용하는 값

- **flex** 플렉스 박스 (1차원 레이아웃)
- **grid** 그리드 (2차원 레이아웃)
- **none** 보여짐 특성 없음, 화면에서 사라짐
- **기타** table, table-row, table-cell 등..

실습, Display: none

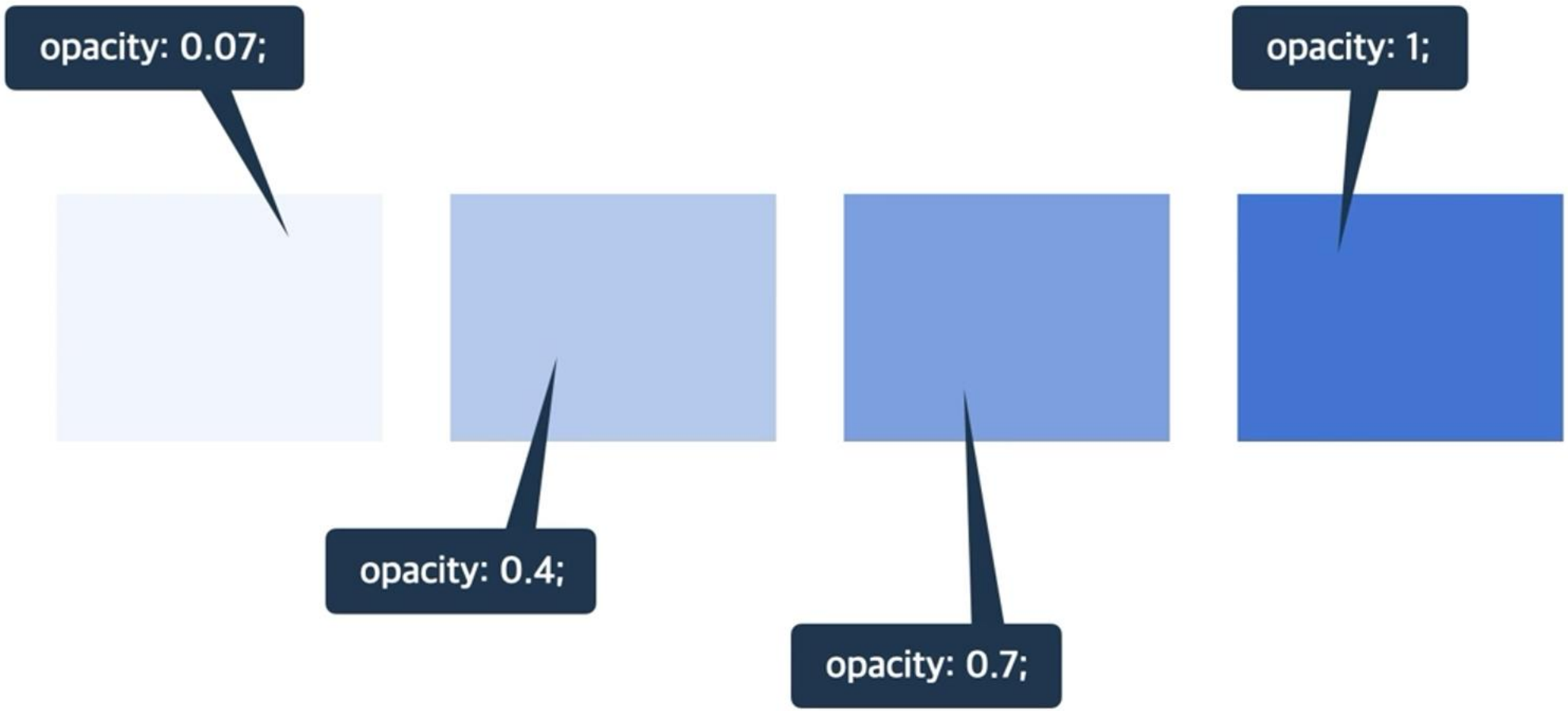
- div 요소의 크기를 200px 200px 배경 파란색으로 설정
- 해당 div에 마우스가 올라가면 요소가 사라지도록 만들기

Opacity

요소 투명도

opacity

1	불투명
0~1	0부터 1 사이의 소수점 숫자


opacity: 0.07;
opacity: 1;
opacity: 0.4;
opacity: 0.7;

실습, 투명도 적용하기

- div 요소의 크기를 200px 200px 배경 파란색으로 설정 x 4
- 각각 원하는 opacity 속성을 적용하기!

Visibility

요소를 숨기는 3가지 방법!

- opacity(투명하게 만들기)
  - 모습만 숨기는 방법 / 속성 남음 / 자리 차지
- visibility : hidden
  - 모습과 속성을 숨기는 방법 / 자리 차지
- display : none
  - 그냥 없애 버리는 방법 / 자리도 사라짐

oppacity visibility display last

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Document</title>
    <style>
      div {
        width: 200px;
        height: 200px;
        cursor: pointer;
        display: inline-block;
      }
      .oppacity {
        background-color: aqua;
        opacity: 1;
      }
      .visibility {
        background-color: greenyellow;
        visibility: visible;
      }
      .display {
        background-color: dodgerblue;
      }
      .last {
        background-color: violet;
      }
    </style>
  </head>
  <body>
    <div class="oppacity">oppacity</div>
    <div class="visibility">visibility</div>
    <div class="display">display</div>
    <div class="last">last</div>
  </body>
</html>
```

Font

글자의 기울기

font-style

normal	기울기 없음
italic	이텔릭체
oblique	기울어진 글자

글자의 두께(가중치)

font-weight

normal, 400	기본 두께
bold, 700	두껍게
bolder	상위(부모) 요소보다 더 두껍게
lighter	상위(부모) 요소보다 더 얇기
100 ~ 900	100단위의 숫자 9개, normal과 bold 이외 두께

글자의 크기

font-size

16px	기본 크기
단위	px, em, rem 등 단위로 지정
%	부모 요소의 폰트 크기에 대한 비율
smaller	상위(부모) 요소보다 작은 크기
larger	상위(부모) 요소보다 큰 크기
xx-small ~ xx-large	가장 작은 크기 ~ 가장 큰 크기까지, 7단계의 크기를 지정

한 줄의 높이, 행간과 유사

line-height

normal	브라우저의 기본 정의를 사용
숫자	요소의 글꼴 크기의 배수로 지정
단위	px, em, rem 등의 단위로 지정
%	요소의 글꼴 크기의 비율로 지정

동해물과 백두산이 마르고 닳도록

하느님이 보우하사 우리나라 만세

```
font-size: 16px;
line-height: 32px;
/* line-height: 2; */
/* line-height: 200%; */
```

32px 16px 동해물과 백두산이 마르고 닳도록

32px 16px 하느님이 보우하사 우리나라 만세

```
font-size: 16px;
line-height: 32px;
/* line-height: 2; */
/* line-height: 200%; */
```

2배 차이

필수로 작성!

글꼴(서체) 지정

```
font-family: 글꼴1, "글꼴2", ... 글꼴계열;
```

띄어쓰기 등 특수문자가 포함된
글꼴 이름은 큰 따옴표로 묶어야 합니다~

Hello World! **serif** 바탕체 계열

Hello World! **sans-serif** 고딕체 계열

Hello World! **monospace** 고정너비(가로폭이 동등) 글꼴 계열

Hello World! **cursive** 필기체 계열

EMPIRE **fantasy** 장식 글꼴 계열

문자에 대한 속성

글자의 색상

color

rgb(0,0,0) 검정색

색상 기타 지정 가능한 색상

문자의 정렬 방식

text-align

- left 왼쪽 정렬
- right 오른쪽 정렬
- center 가운데 정렬
- justify 양쪽 정렬

문자의 장식(선)

text-decoration

화면에 출력!

동해물과 백두산이 마르고 닳도록 — none 장식 없음

동해물과 백두산이 마르고 닳도록 — underline 밑줄

동해물과 백두산이 마르고 닳도록 — overline 윗줄

~~동해물과 백두산이 마르고 닳도록~~ — line-through 중앙 선

동해물과 백두산이 마르고 닳도록 하느님이 보우하사 우리나라 만세 무궁화 삼천리 화려 강산 대한 사람 대한으로 길이 보전하세

들여쓰기(50px)

음수를 사용할 수 있어요!
반대는 내어쓰기(outdent)입니다.

문자 첫 줄의 들여쓰기

text-indent

0	들여쓰기 없음
단위	px, em, rem 등 단위로 지정
%	요소의 가로 너비에 대한 비율

배경

요소의 배경 색상

background-color

transparent 투명함

색상 지정 가능한 색상

요소의 배경 이미지 삽입

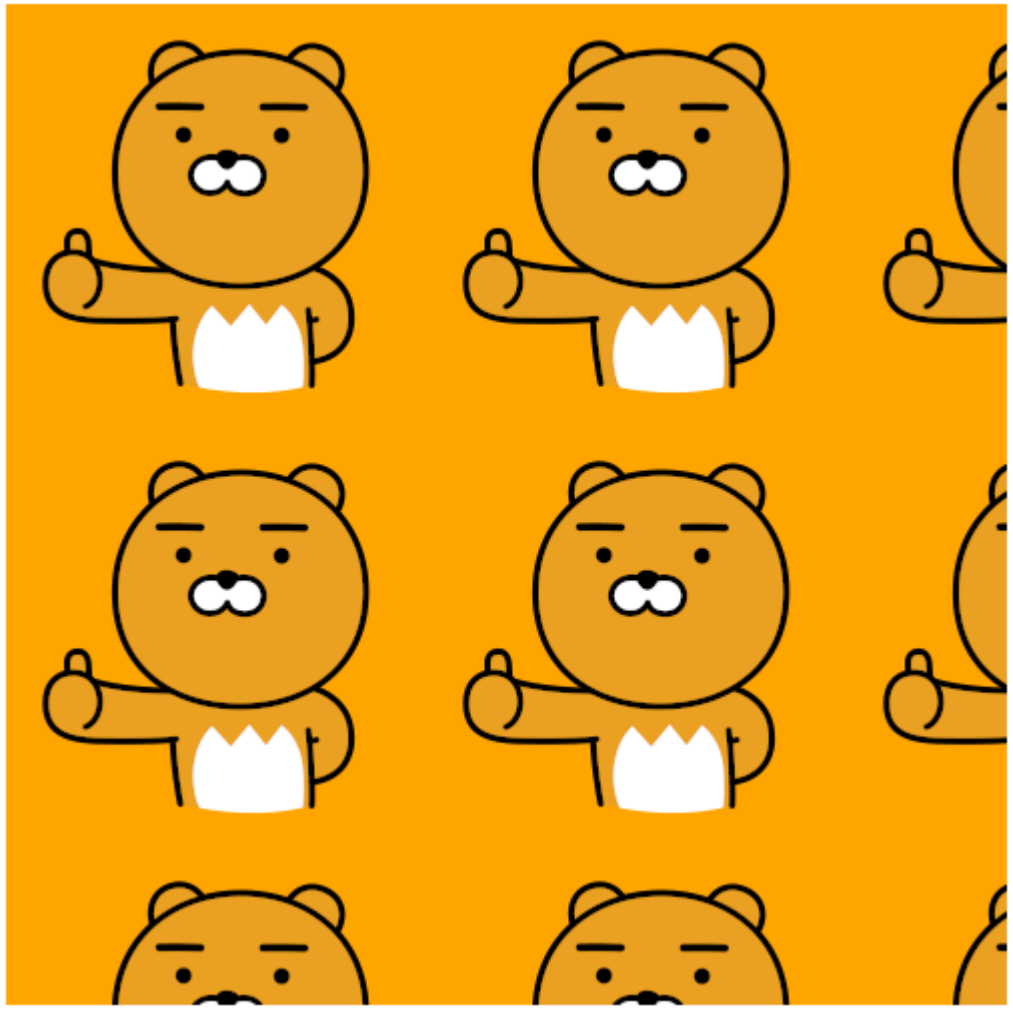
background-image

`none` 이미지 없음

`url("경로")` 이미지 경로

백그라운드 컬러 위에 이미지!

```
<style>
    div {
        background-color: orange;
        background-image: url("");
        width: 500px;
        height: 500px;
    }
</style>
```



요소의 배경 이미지 반복

background-repeat

- **repeat** 이미지를 수직, 수평 반복
- **repeat-x** 이미지를 수평 반복
- **repeat-y** 이미지를 수직 반복
- **no-repeat** 반복 없음

요소의 배경 이미지 위치

background-position

방향1 방향2

0% 0% 0% ~ 100% 사이 값

방향 top, bottom, left, right, center 방향

단위 px, em, rem 등 단위로 지정

x축 y축

Background-position: top right;

Background-position: center;

Background-position: 100px 30px;

30px

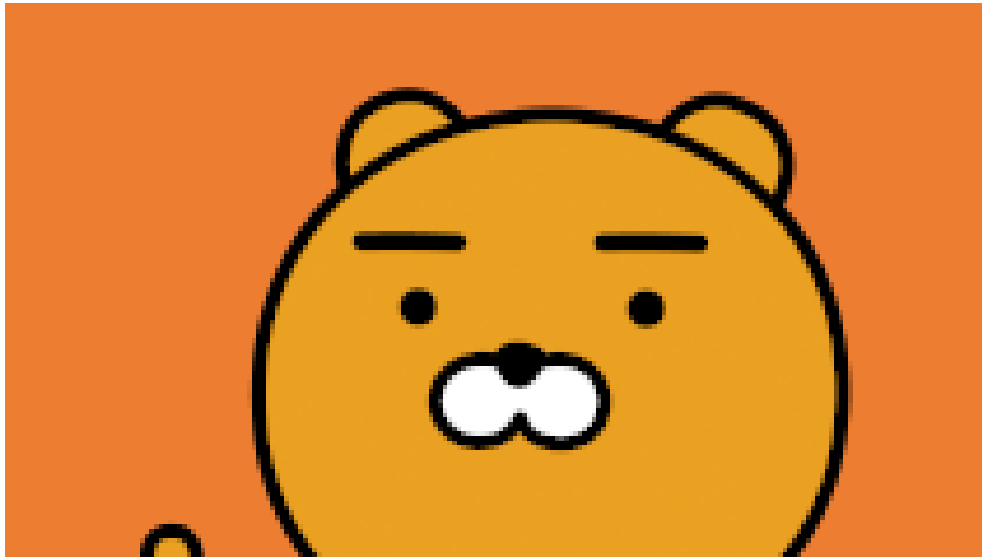
100px

요소의 배경 이미지 크기

background-size

- **auto** 이미지의 실제 크기
- **단위** px, em, rem 등 단위로 지정
- **cover** 비율을 유지, 요소의 더 넓은 너비에 맞춤
- **contain** 비율을 유지, 요소의 더 짧은 너비에 맞춤

Background-size: cover;



Background-size: contain;

요소의 배경 이미지 스크롤 특성

background-attachment

- **scroll** 이미지가 요소를 따라서 같이 스크롤
- **fixed** 이미지가 뷰포트에 고정, 스크롤 X
- **local** 요소 내 스크롤 시 이미지가 같이 스크롤

```
<style>
        body {
            height: 3000px;
        }

        div {
            background-color: orange;
            background-image:
url("https://item.kakaocdn.net/do/d84248170c2c52303db27306a00fb861f604e7b0e6900f9ac5
3a43965300eb9a");
            height: 900px;
            background-size: cover;
            background-attachment: scroll;
        }
    </style>
```

배치1

position과 같이 사용하는 CSS 속성들!
모두 음수를 사용할 수 있어요!

top
bottom
left
right
z-index

요소의 위치 지정 기준

position

위치 상 부모 요소를
꼭 확인해야 해요!

- **static** 기준 없음
- **relative** 요소 자신을 기준
- **absolute** 위치 상 부모 요소를 기준
- **fixed** 뷰포트(브라우저)를 기준
- **sticky** 스크롤 영역 기준

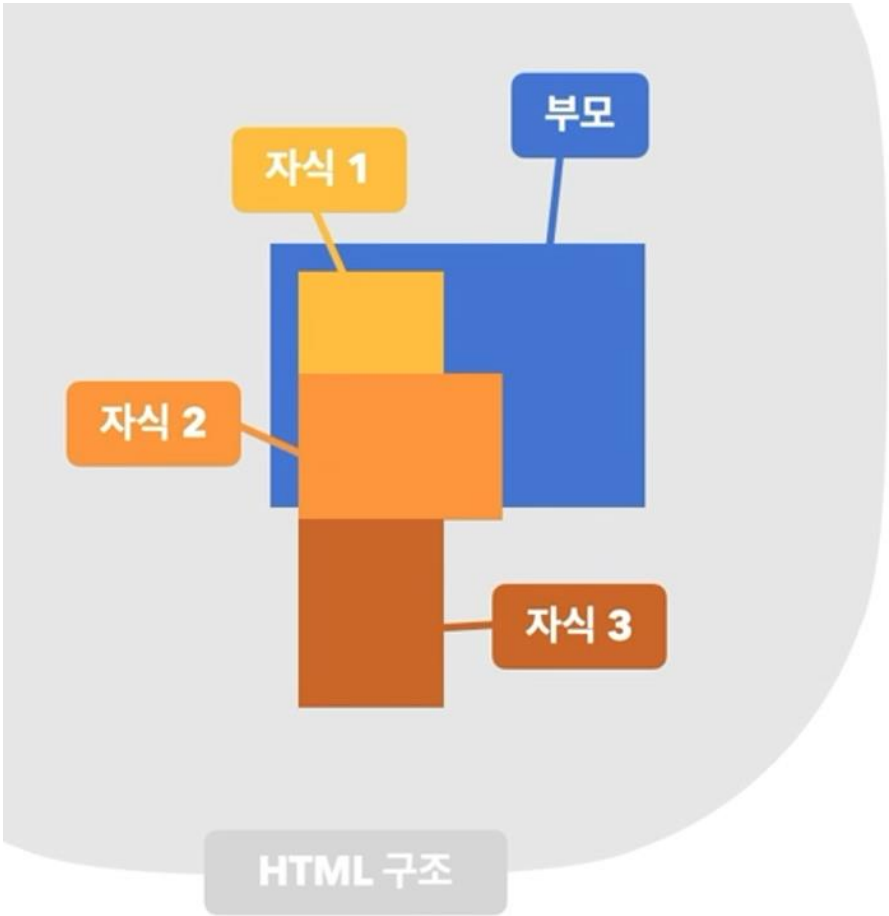
요소의 각 방향별 거리 지정

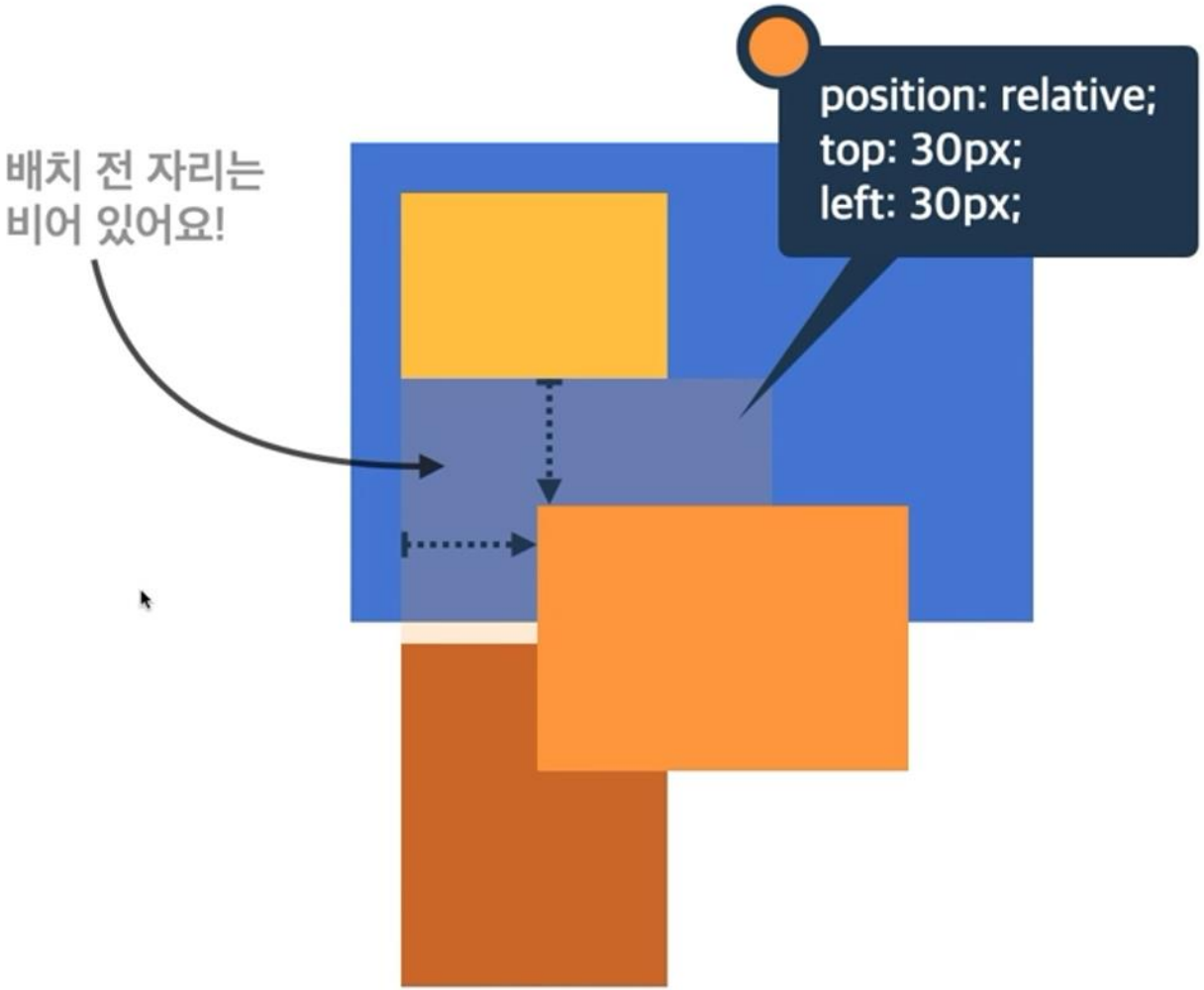
top, bottom, left, right

auto 브라우저가 계산

단위 px, em, rem 등 단위로 지정

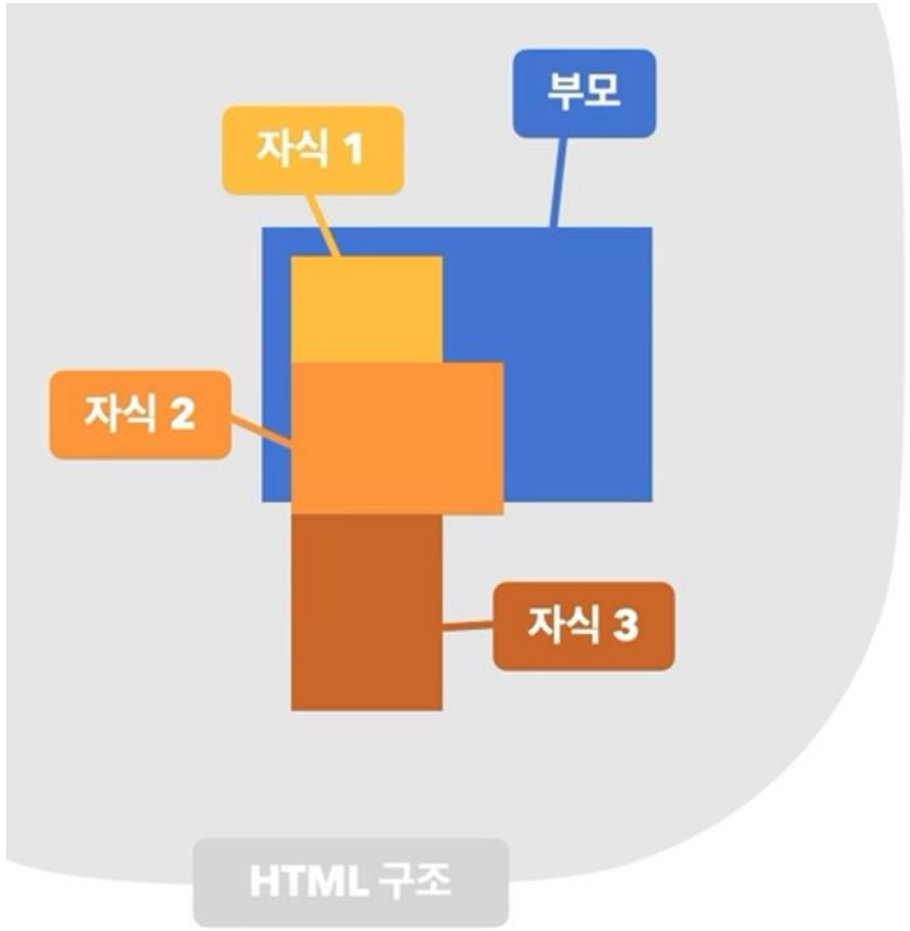
relative

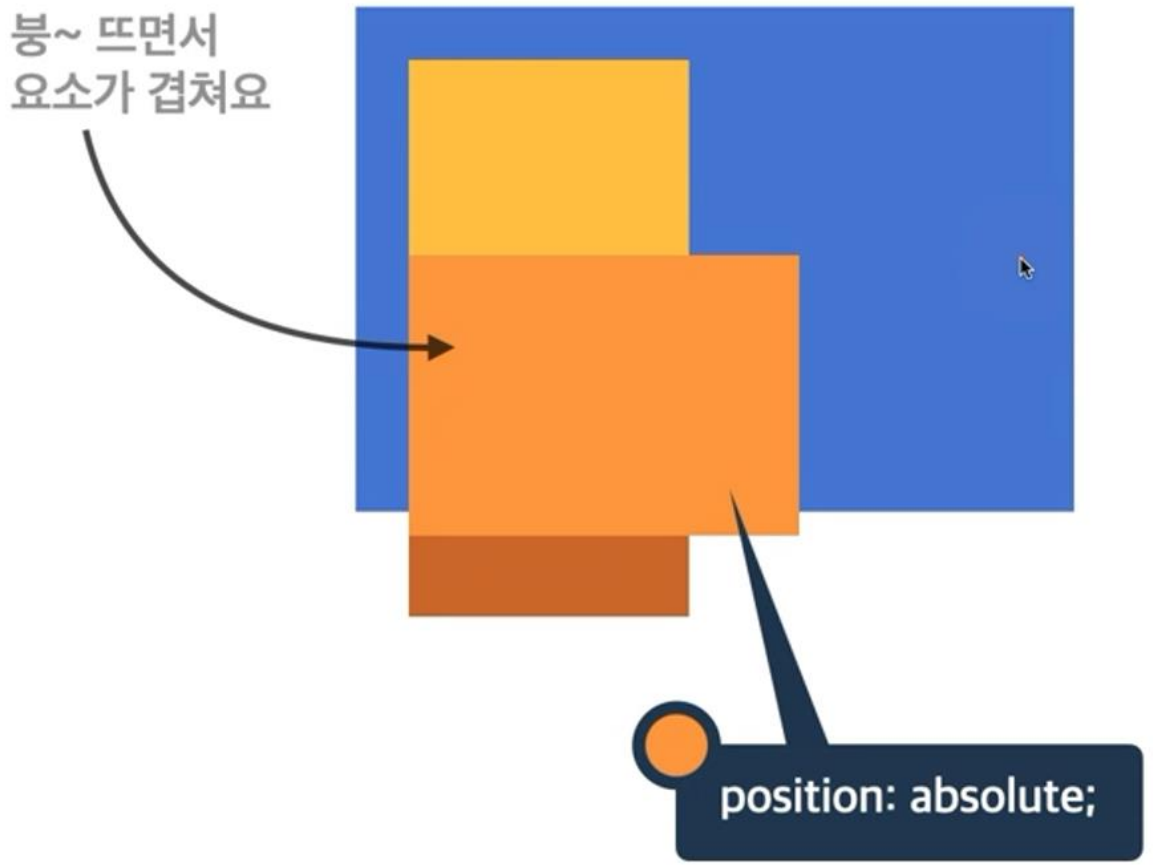




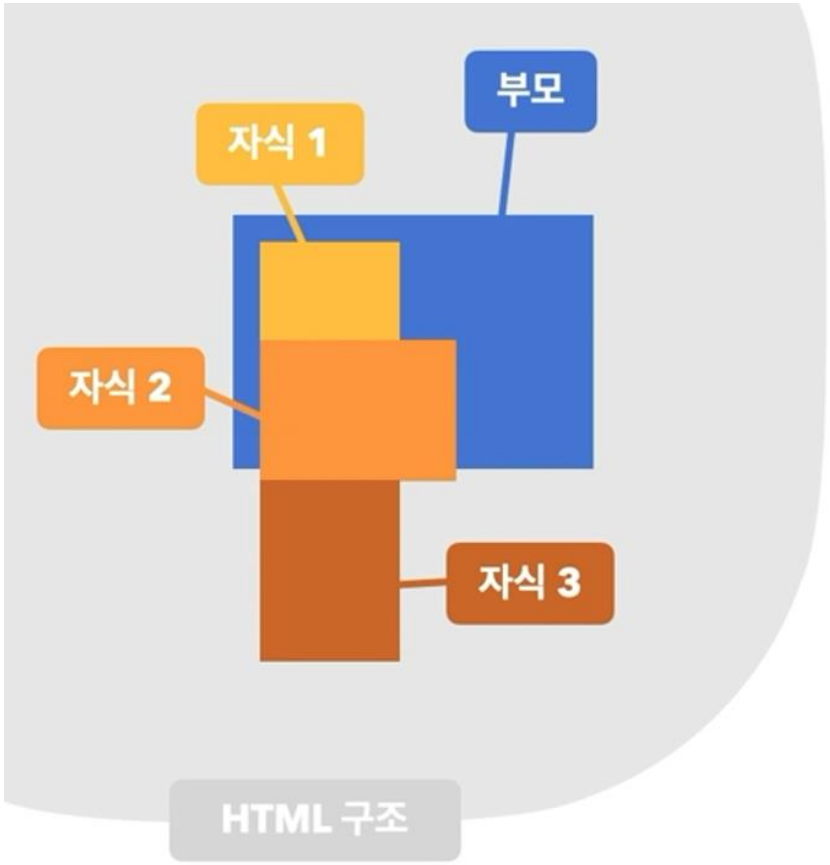
relative 요소 자신을 기준으로 배치!

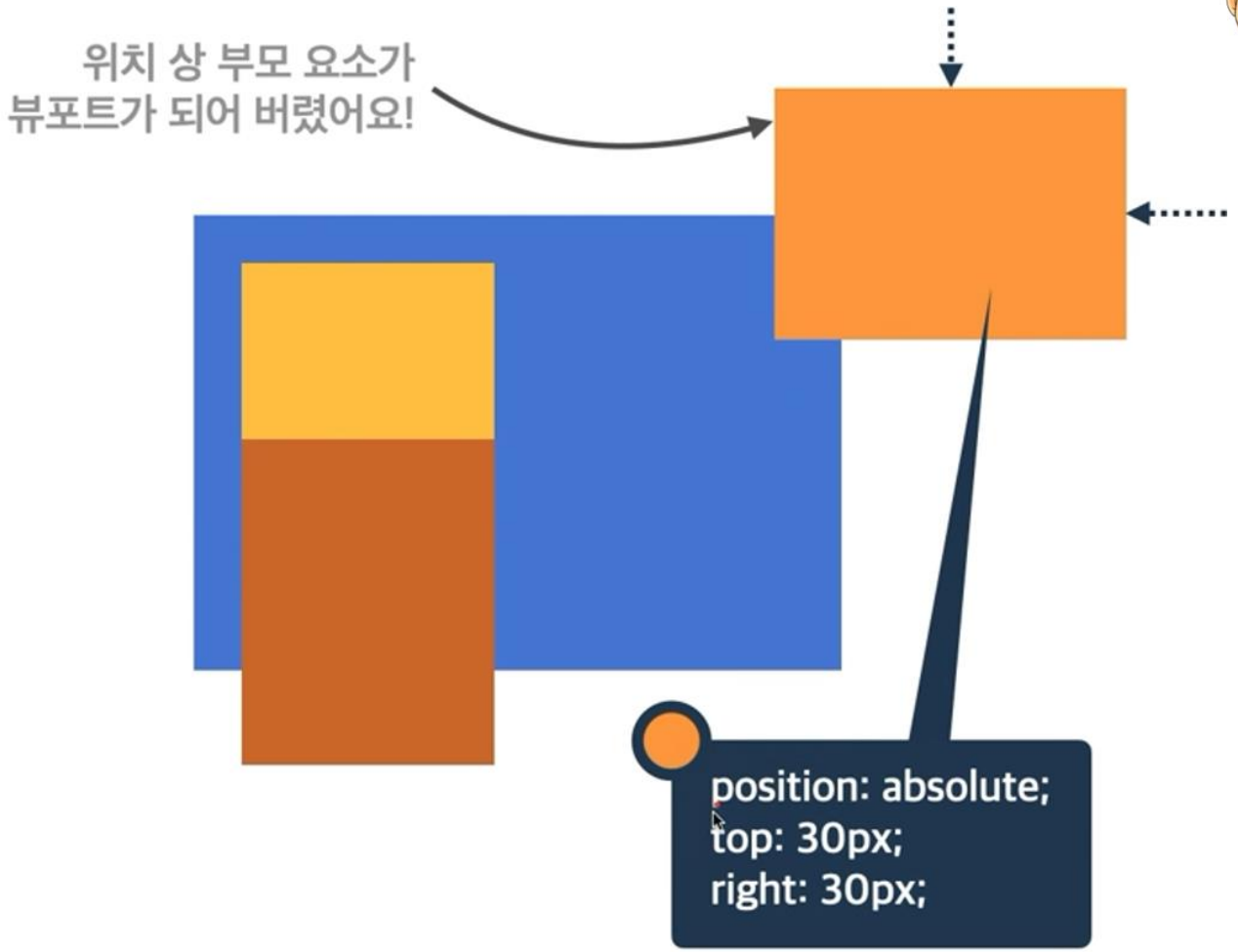
absolute



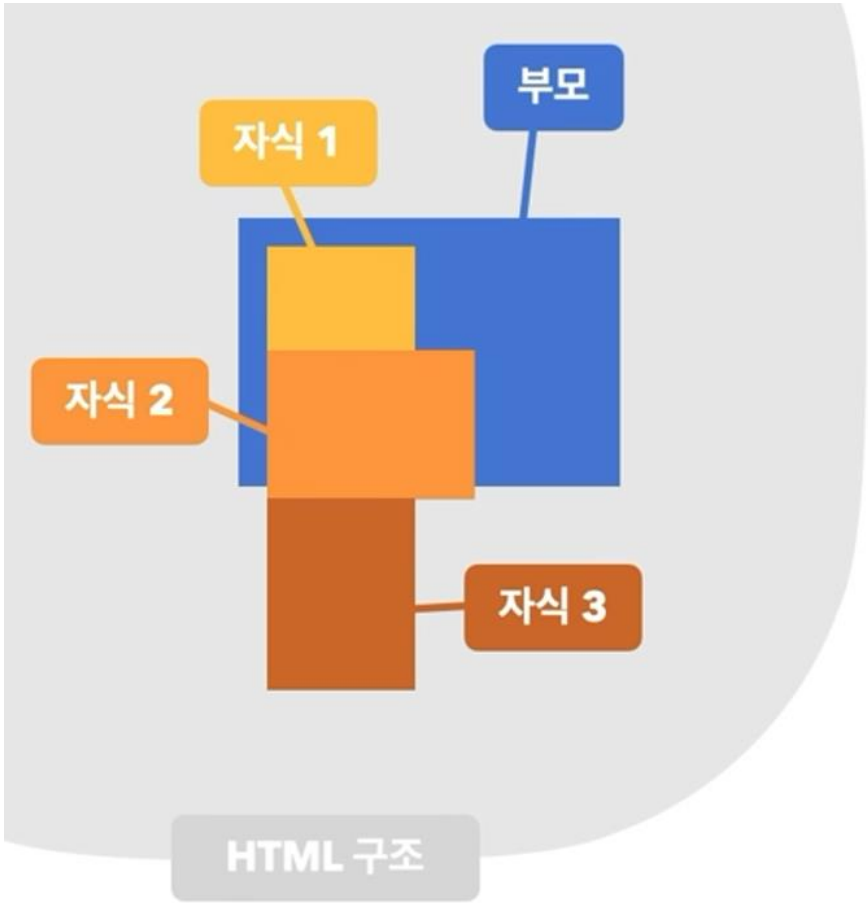


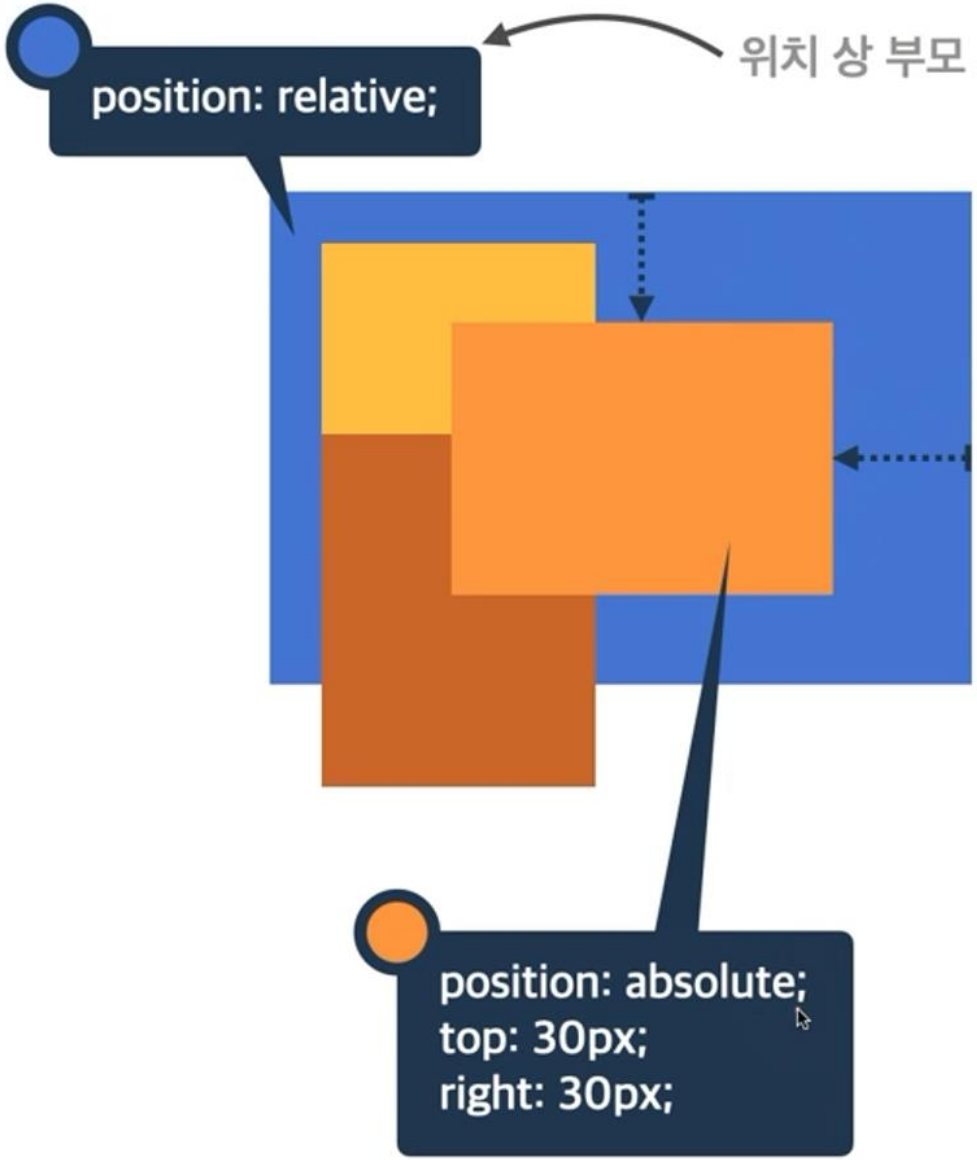
absolute 위치 상 부모 요소를 기준으로 배치!



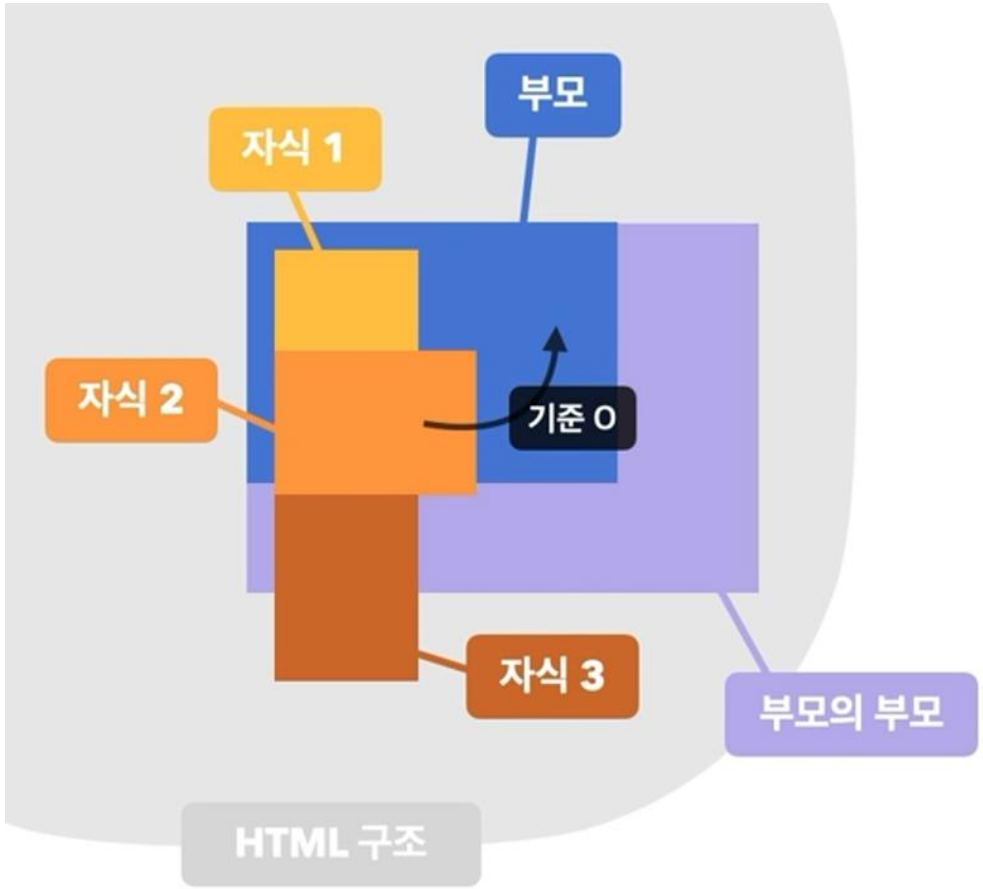


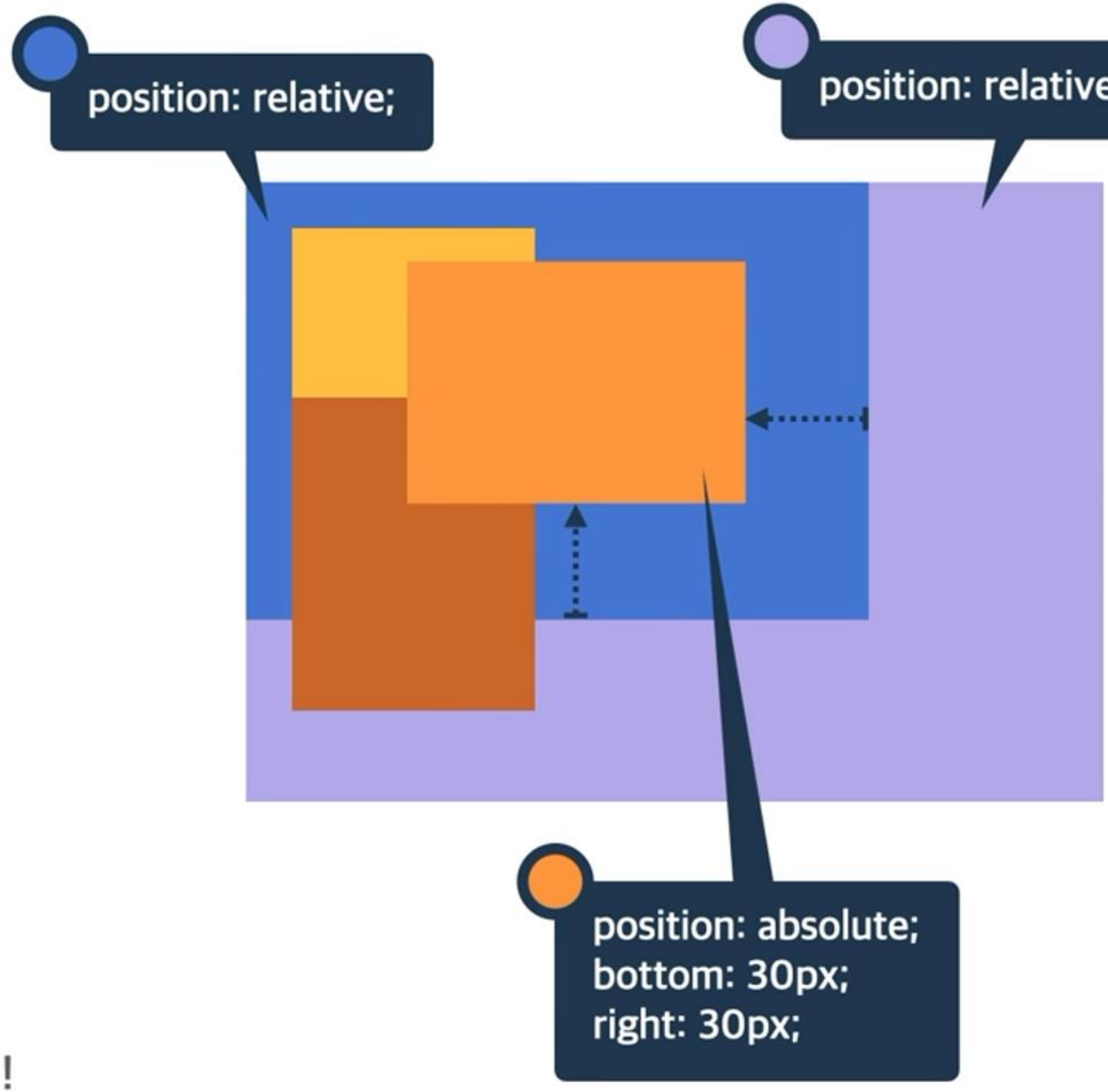
absolute 위치 상 부모 요소를 기준으로 배치!



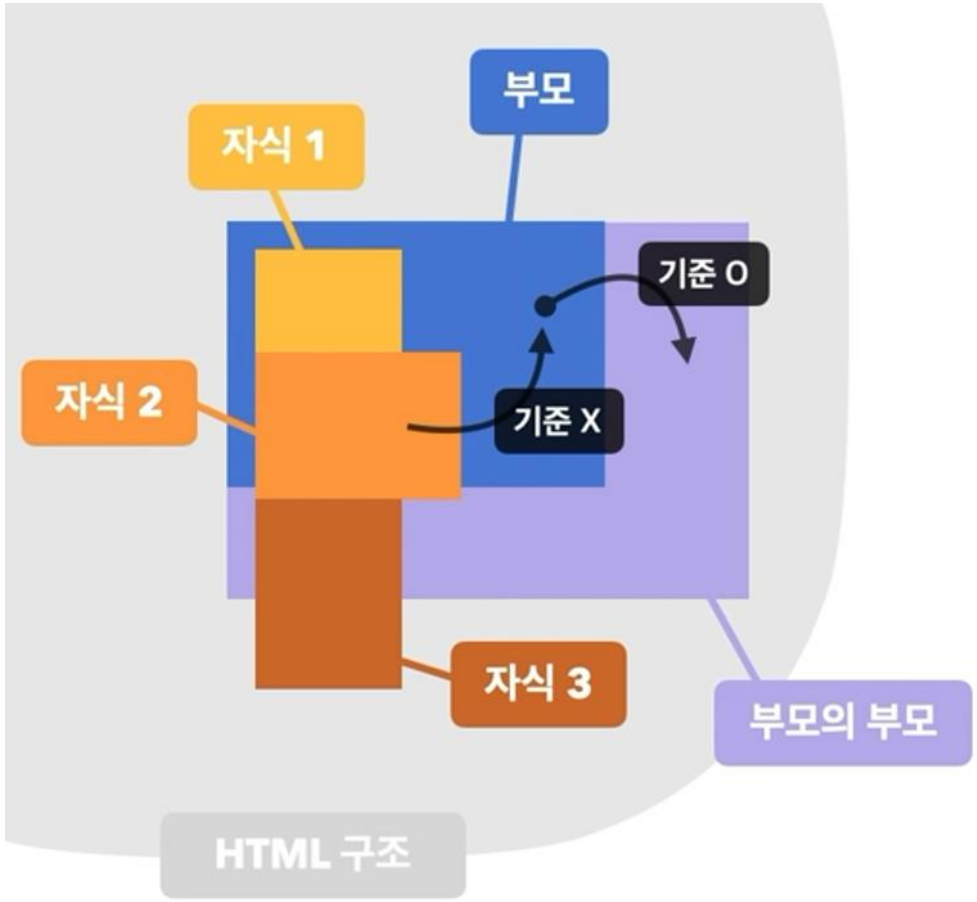


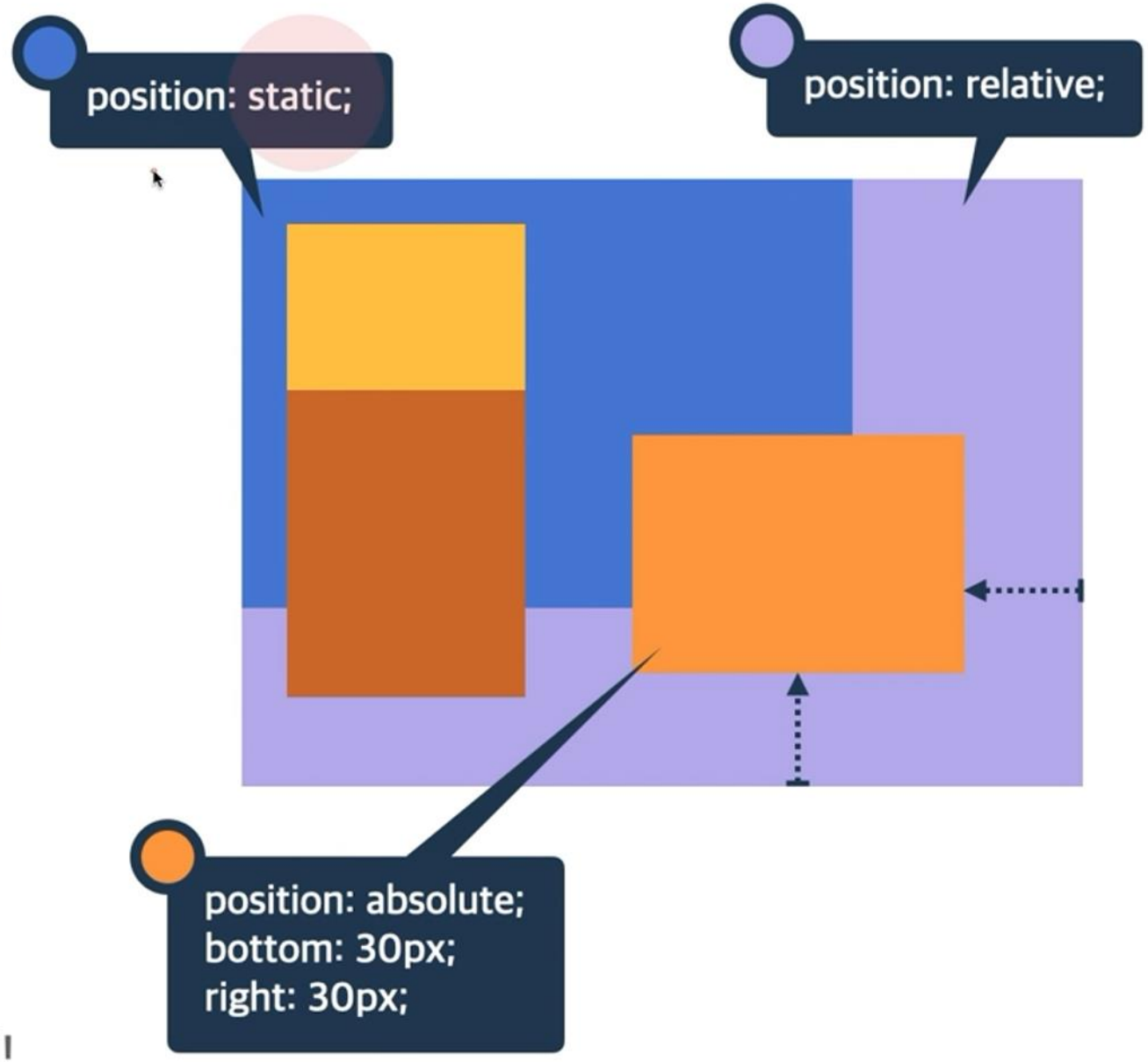
absolute 위치 상 부모 요소를 기준으로 배치!



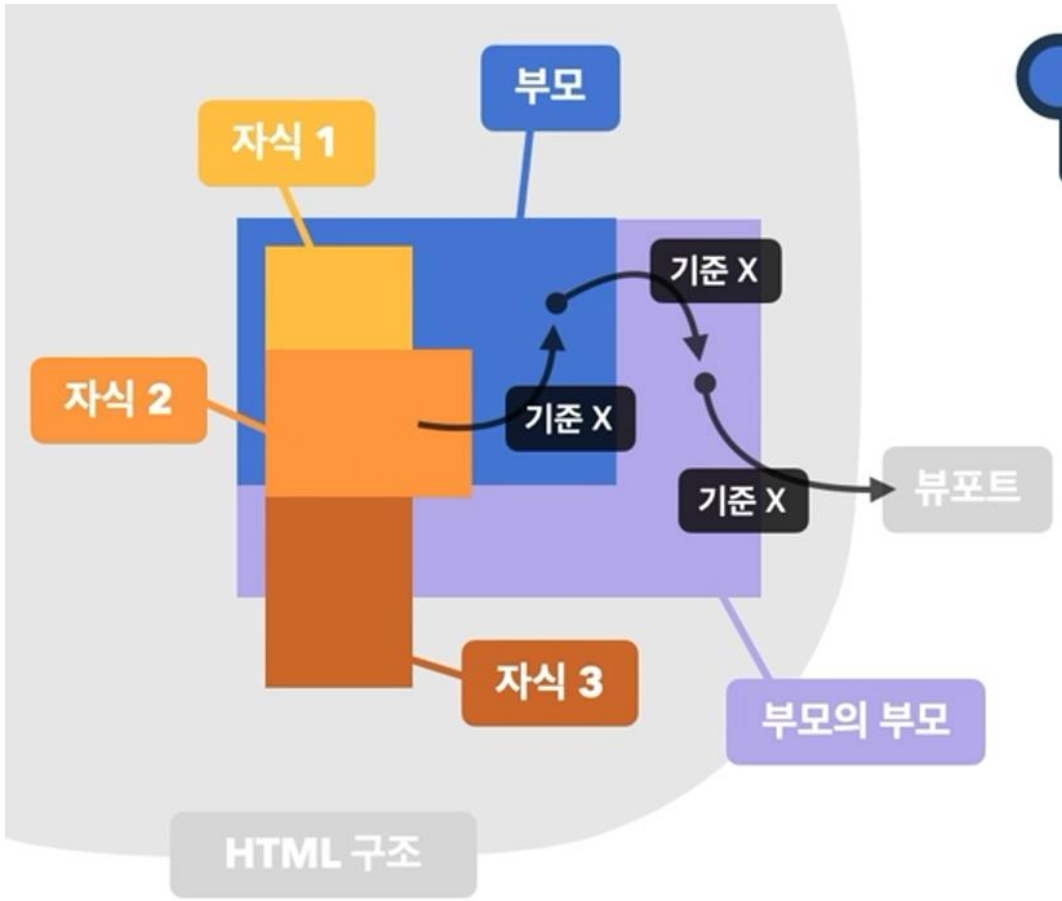


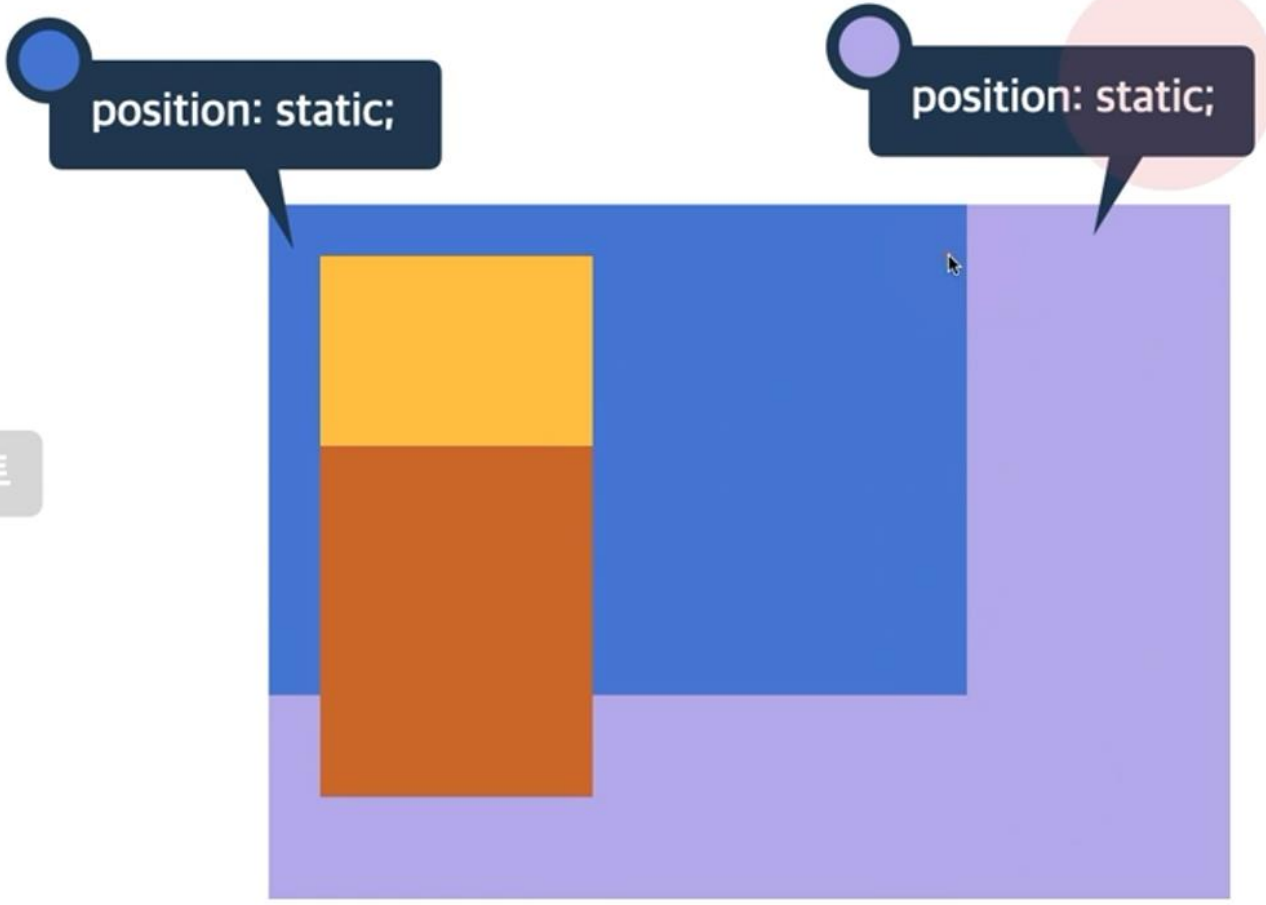
absolute 위치 상 부모 요소를 기준으로 배치!





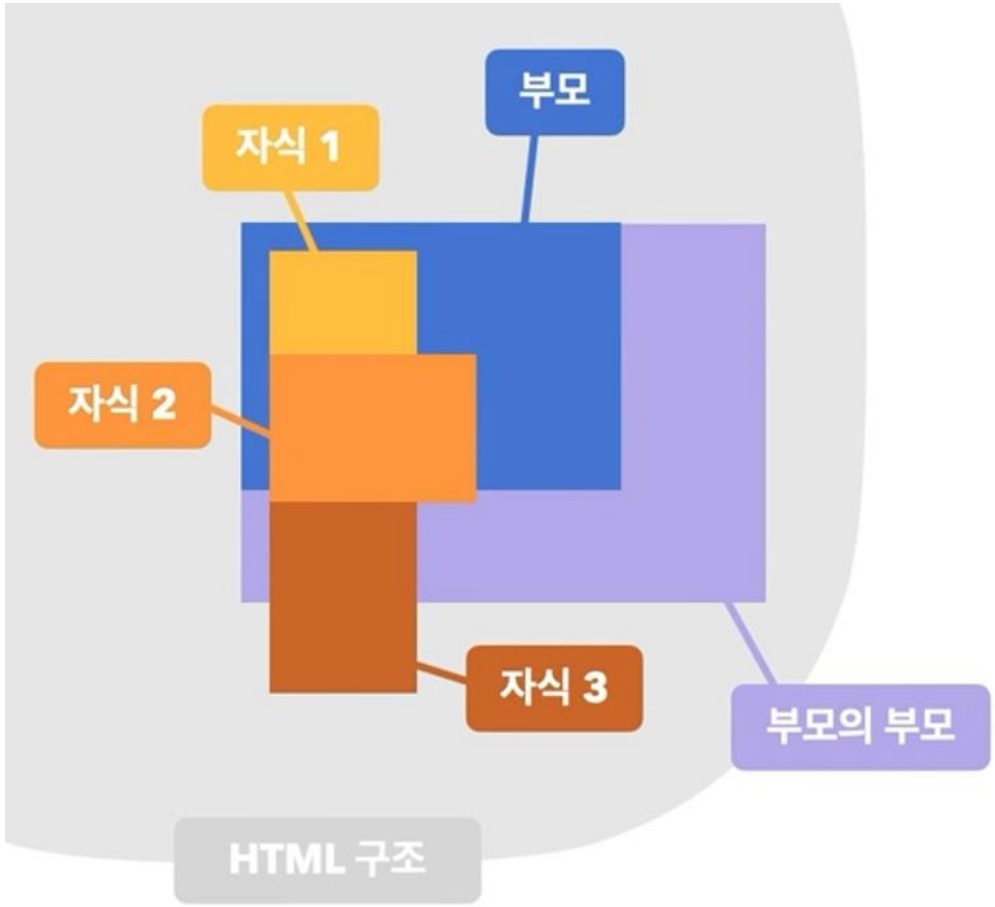
absolute 위치 상 부모 요소를 기준으로 배치!

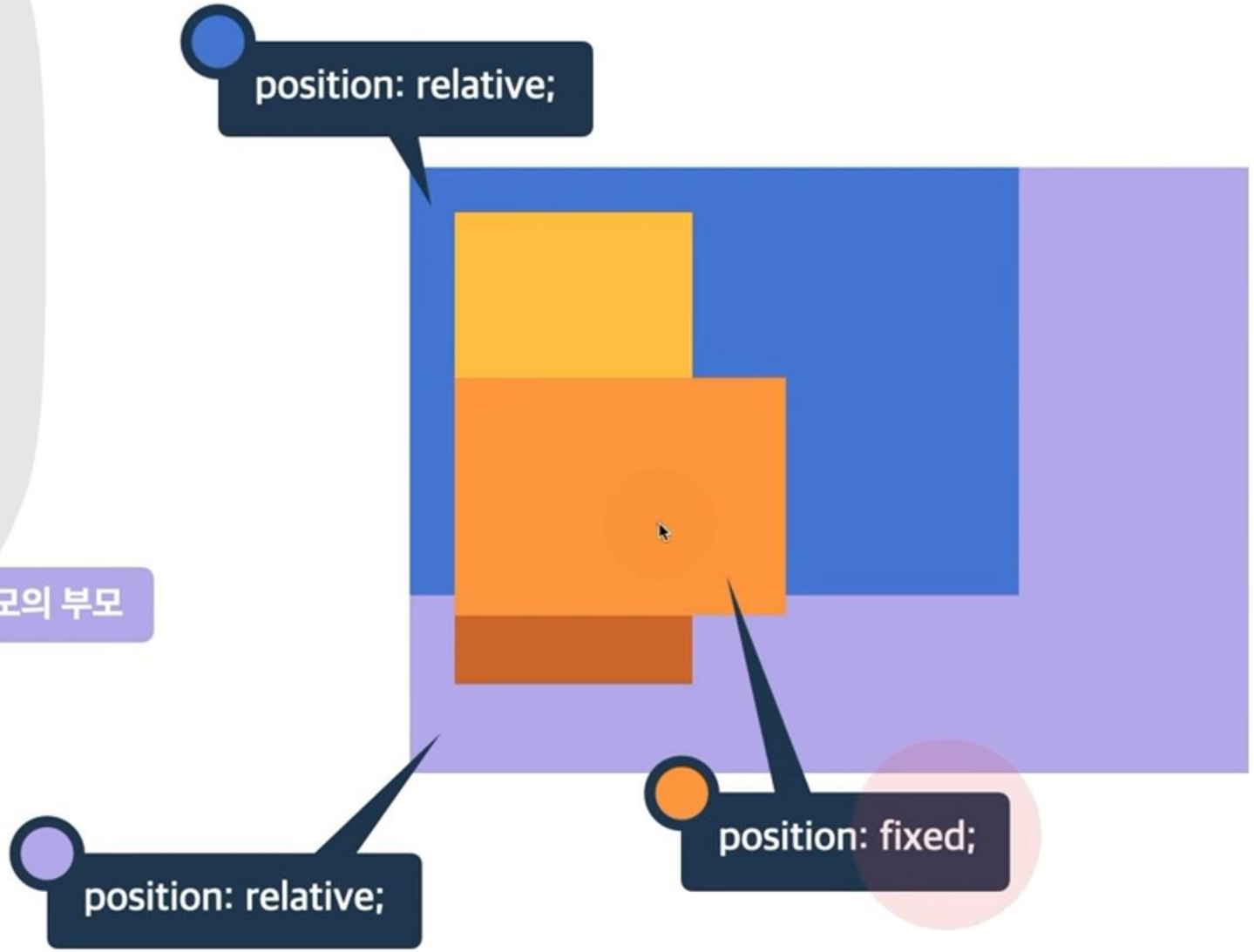




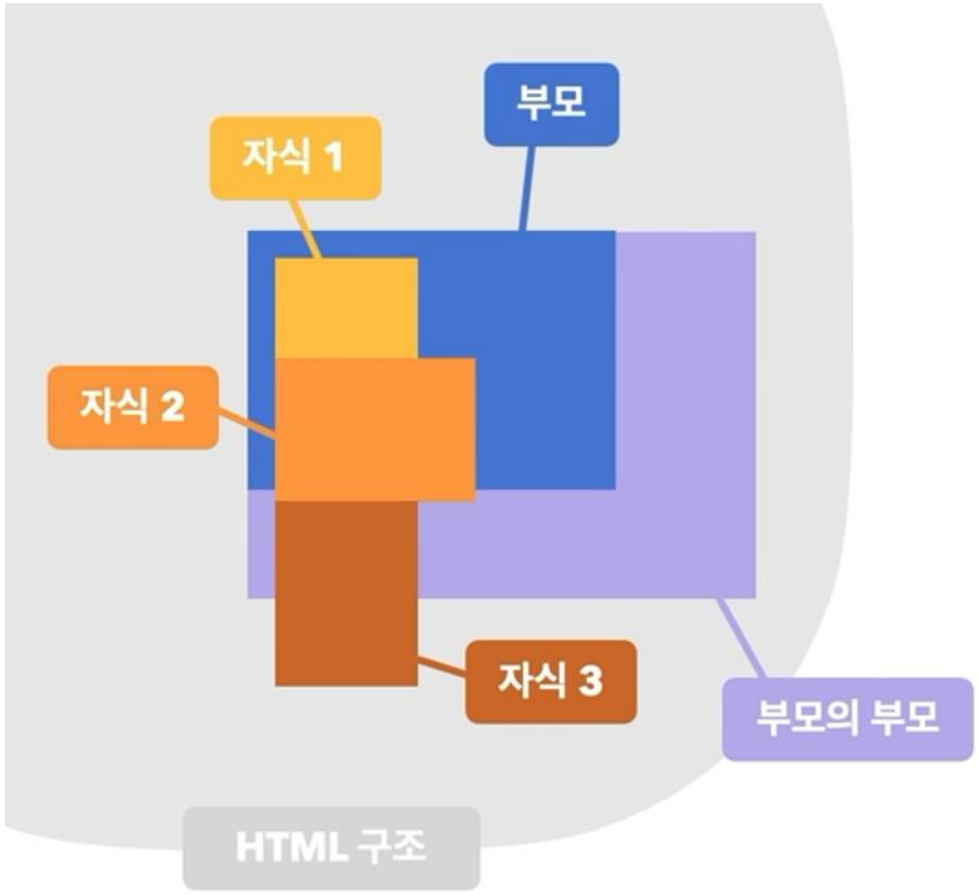
absolute 위치 상 부모 요소를 기준으로 배치!

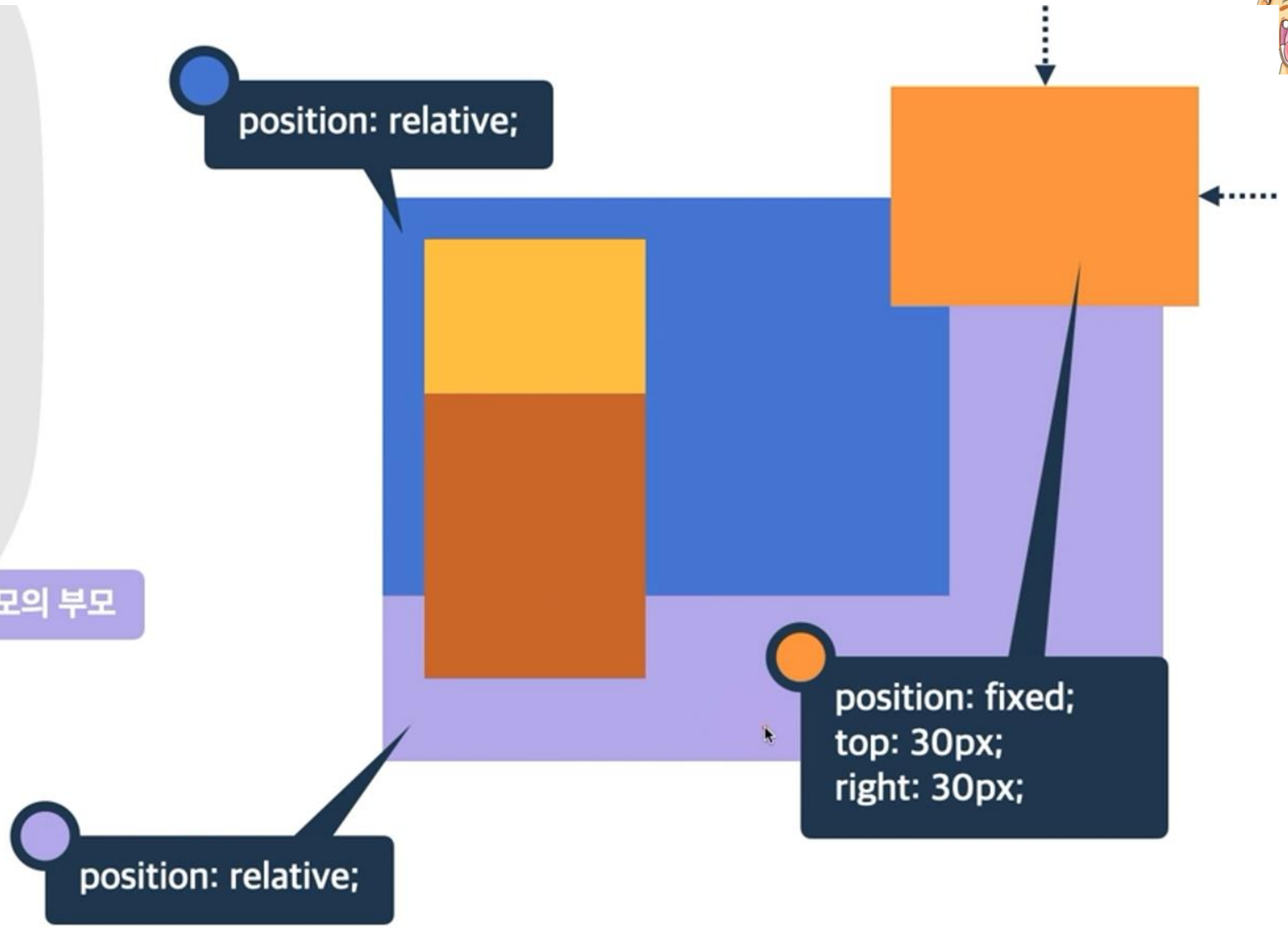
fixed





fixed 뷰포트(브라우저)를 기준으로 배치!





fixed 뷰포트(브라우저)를 기준으로 배치!

요소 쌓임 순서(Stack order)

어떤 요소가 사용자와 더 가깝게 있는지(위에 쌓이는지) 결정

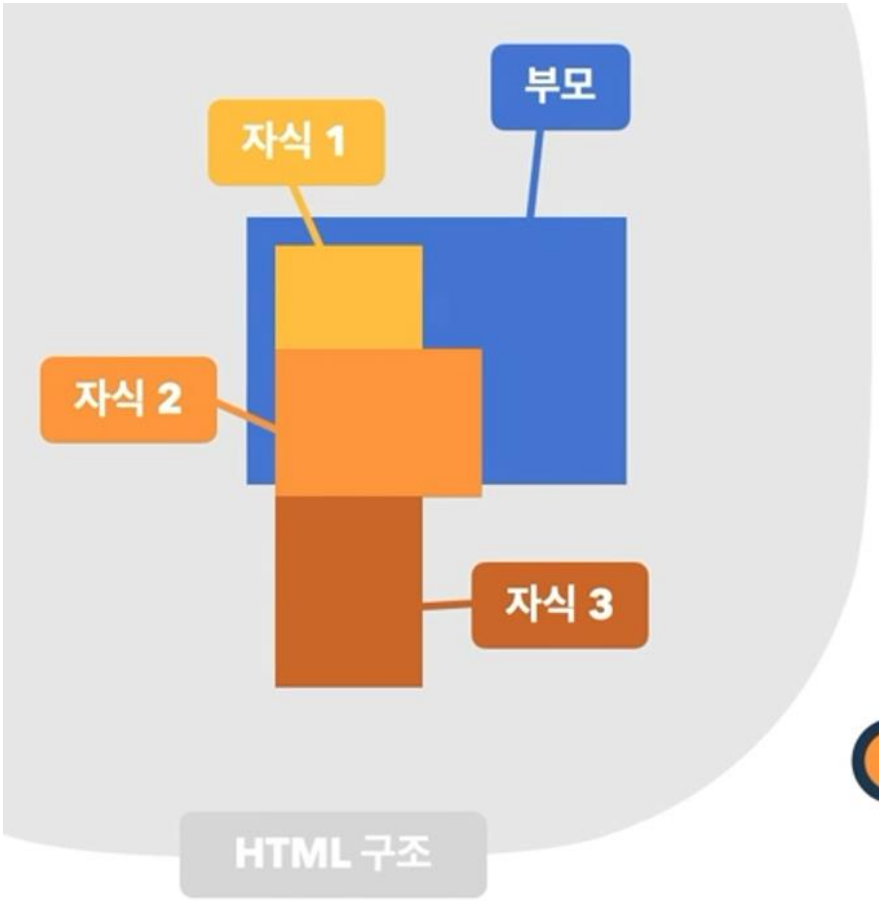
1. 요소에 position 속성의 값이 있는 경우 위에 쌓임.(기본값 static 제외)
2. 1번 조건이 같은 경우, z-index 속성의 숫자 값이 높을 수록 위에 쌓임.
3. 1번과 2번 조건까지 같은 경우, HTML의 다음 구조일 수록 위에 쌓임.

요소의 쌓임 정도를 지정

z-index

auto 부모 요소와 동일한 쌓임 정도

숫자 숫자가 높을 수록 위에 쌓임



position: relative;

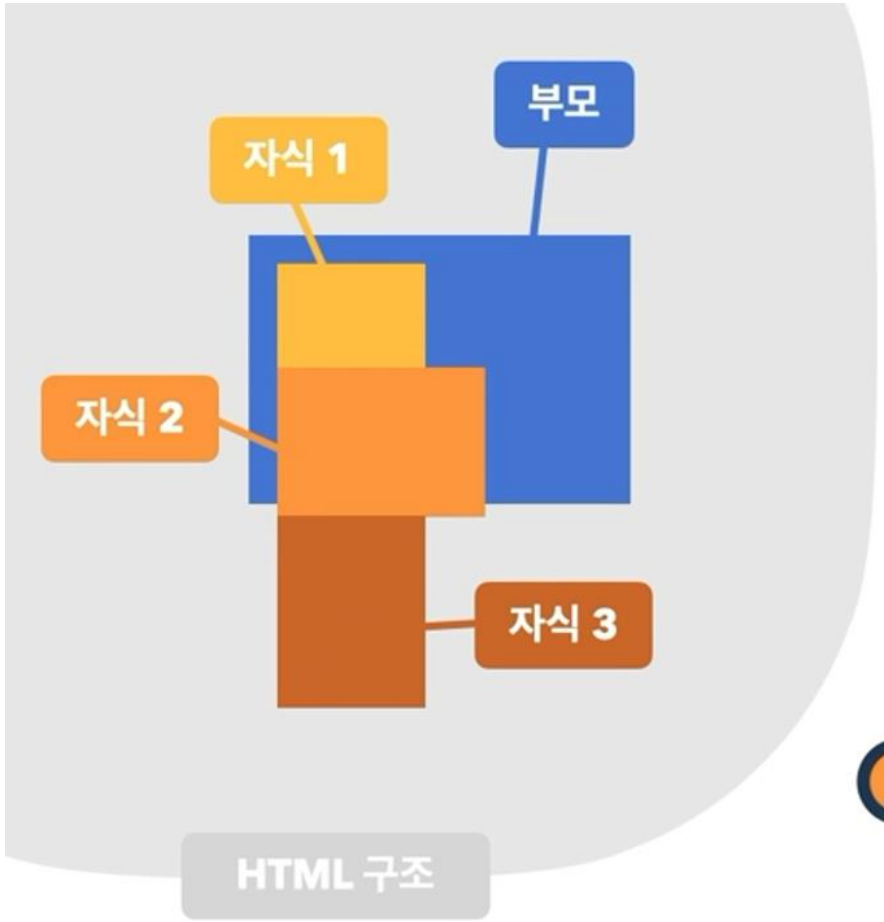
position: static;
z-index: 999;

1

2

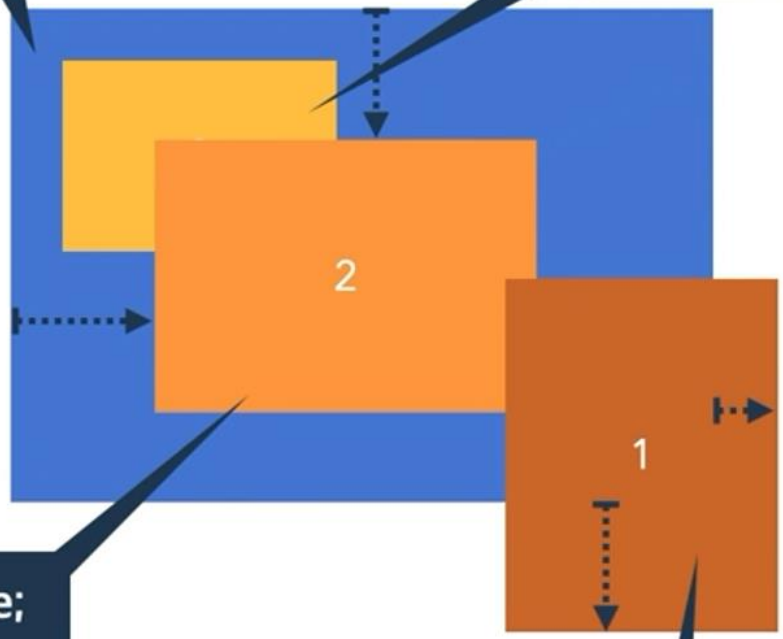
position: absolute;
top: 30px;
left: 30px;
z-index: 1;

position: absolute;
bottom: -30px;
right: -10px;



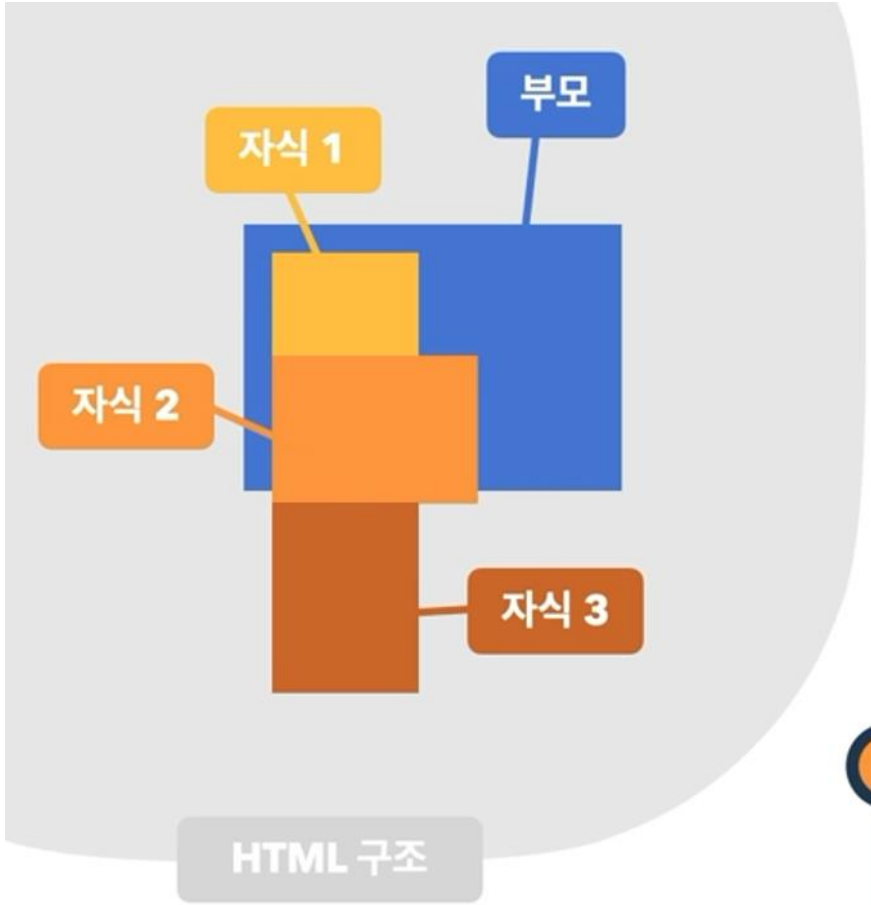
position: relative;

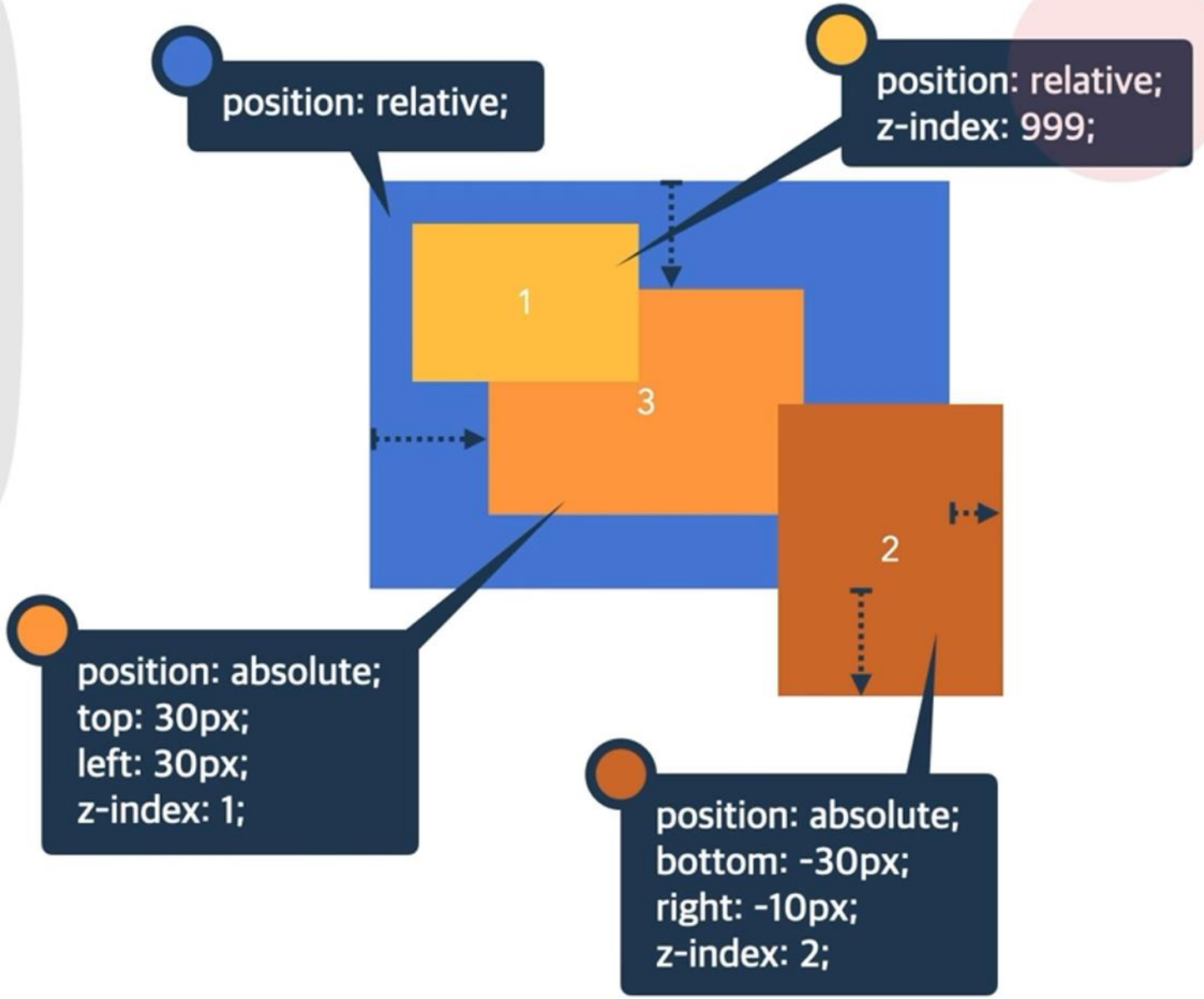
position: static;
z-index: 999;



position: absolute;
top: 30px;
left: 30px;
z-index: 1;

position: absolute;
bottom: -30px;
right: -10px;
z-index: 2;


부모
자식 1
자식 2
자식 3
HTML 구조


position: relative;
position: relative;
z-index: 999;
1
3
2
position: absolute;
top: 30px;
left: 30px;
z-index: 1;
position: absolute;
bottom: -30px;
right: -10px;
z-index: 2;

요소의 display가 변경됨

position 속성의 값으로 absolute, fixed가 지정된 요소는,

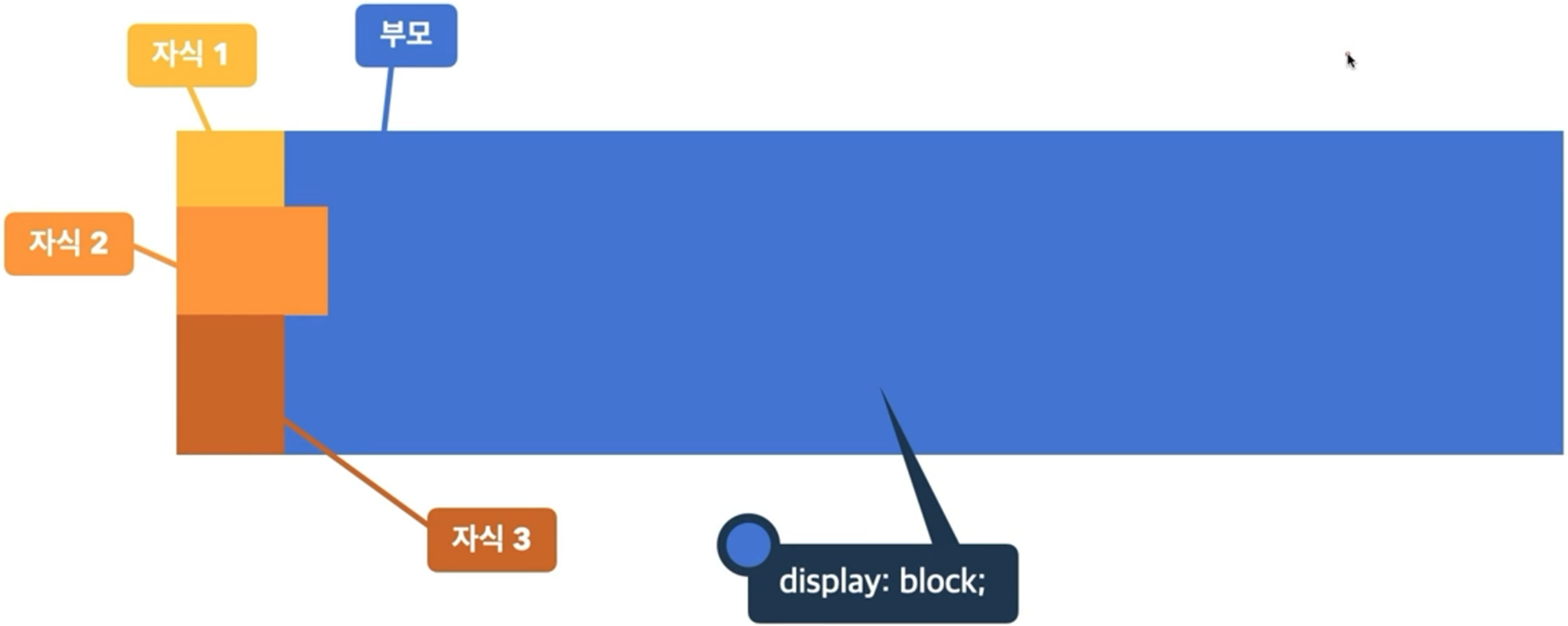
display 속성이 block으로 변경됨.

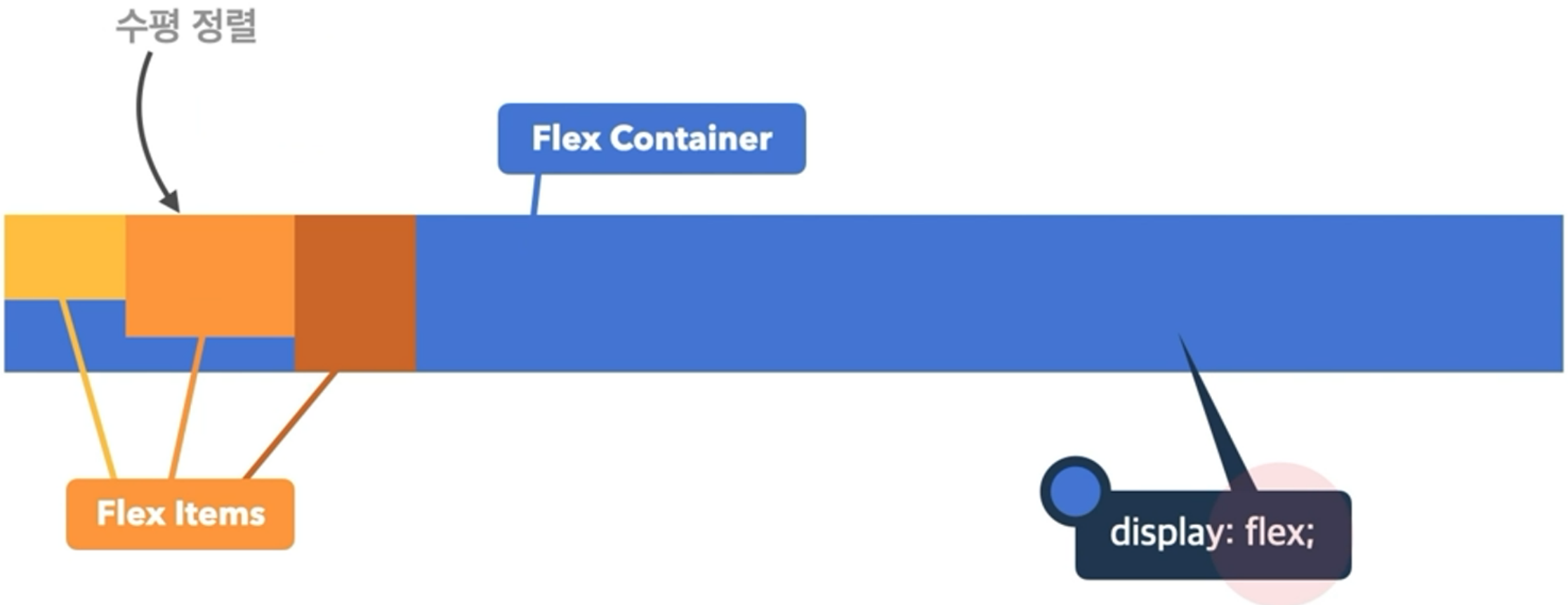
```
display: block;
position: absolute;
top: 30px;
left: 30px;
z-index: 1;
```

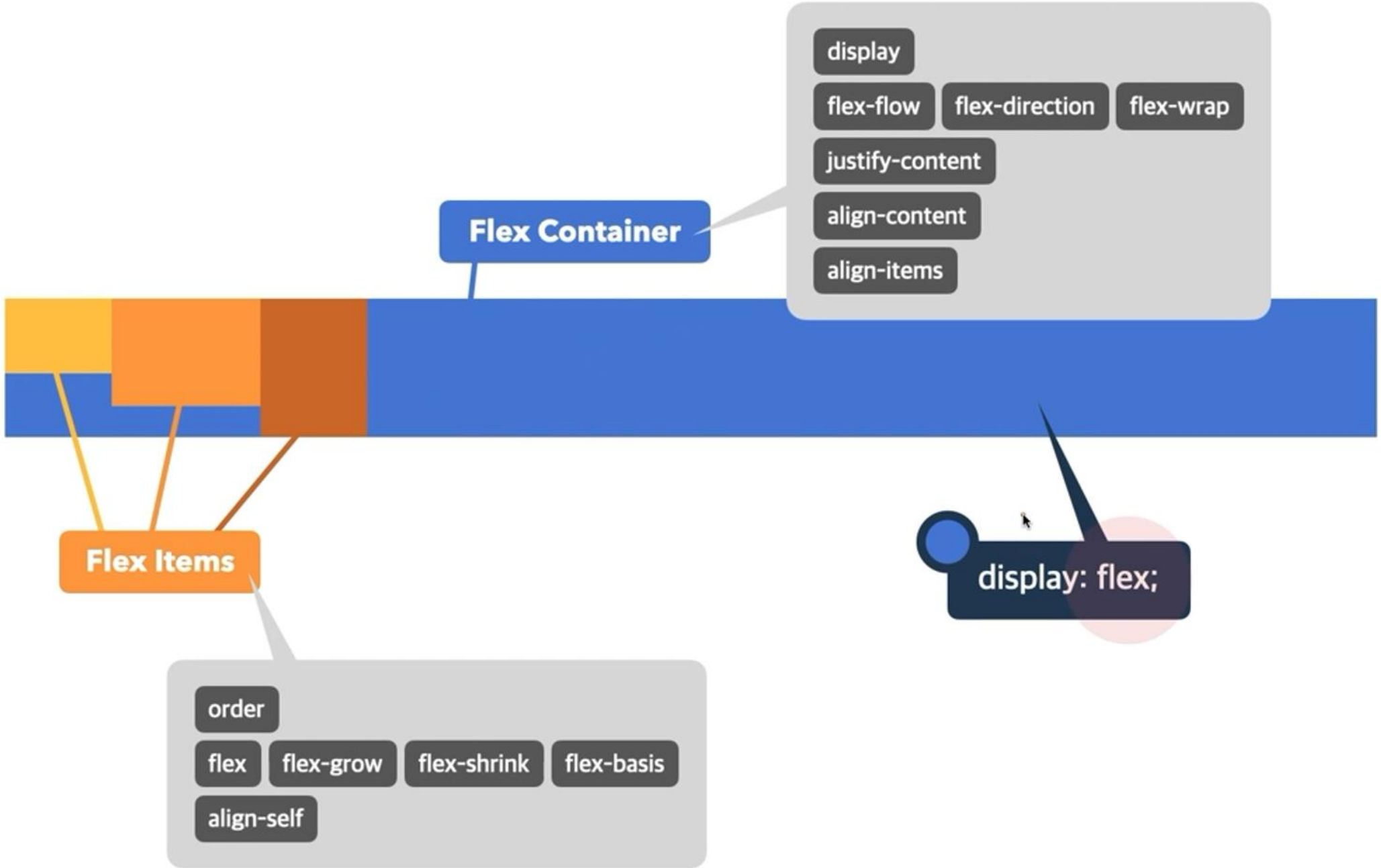
=

```
position: absolute;
top: 30px;
left: 30px;
z-index: 1;
```

Flex


부모
자식 1
자식 2
자식 3
display: block;


수평 정렬
Flex Container
Flex Items
display: flex;


display
flex-flow
flex-direction
flex-wrap
justify-content
align-content
align-items
Flex Container
Flex Items
display: flex;
order
flex
flex-grow
flex-shrink
flex-basis
align-self

Flex

Container 속성

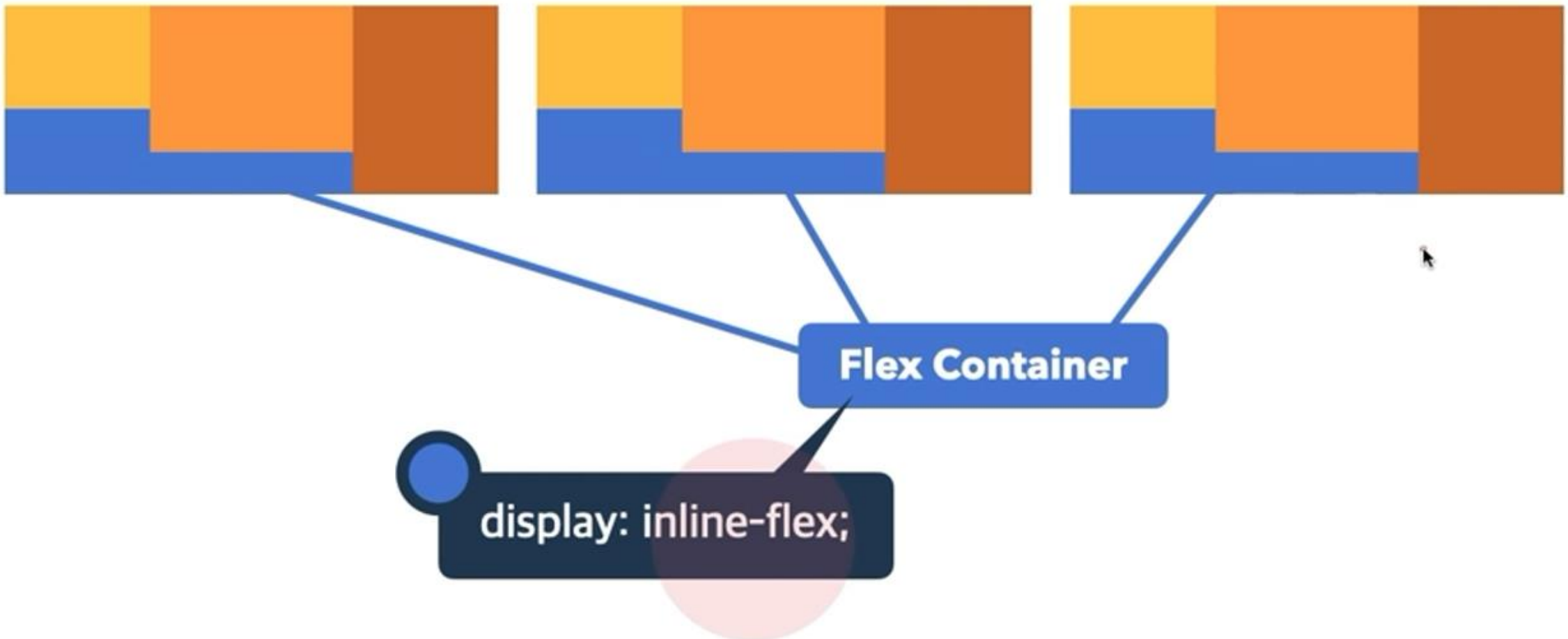
Flex Container의 화면 출력(보여짐) 특성

display

flex 블록 요소와 같이 Flex Container 정의

inline-flex 인라인 요소와 같이 Flex Container 정의


display: flex;
Flex Container


Flex Container
display: inline-flex;

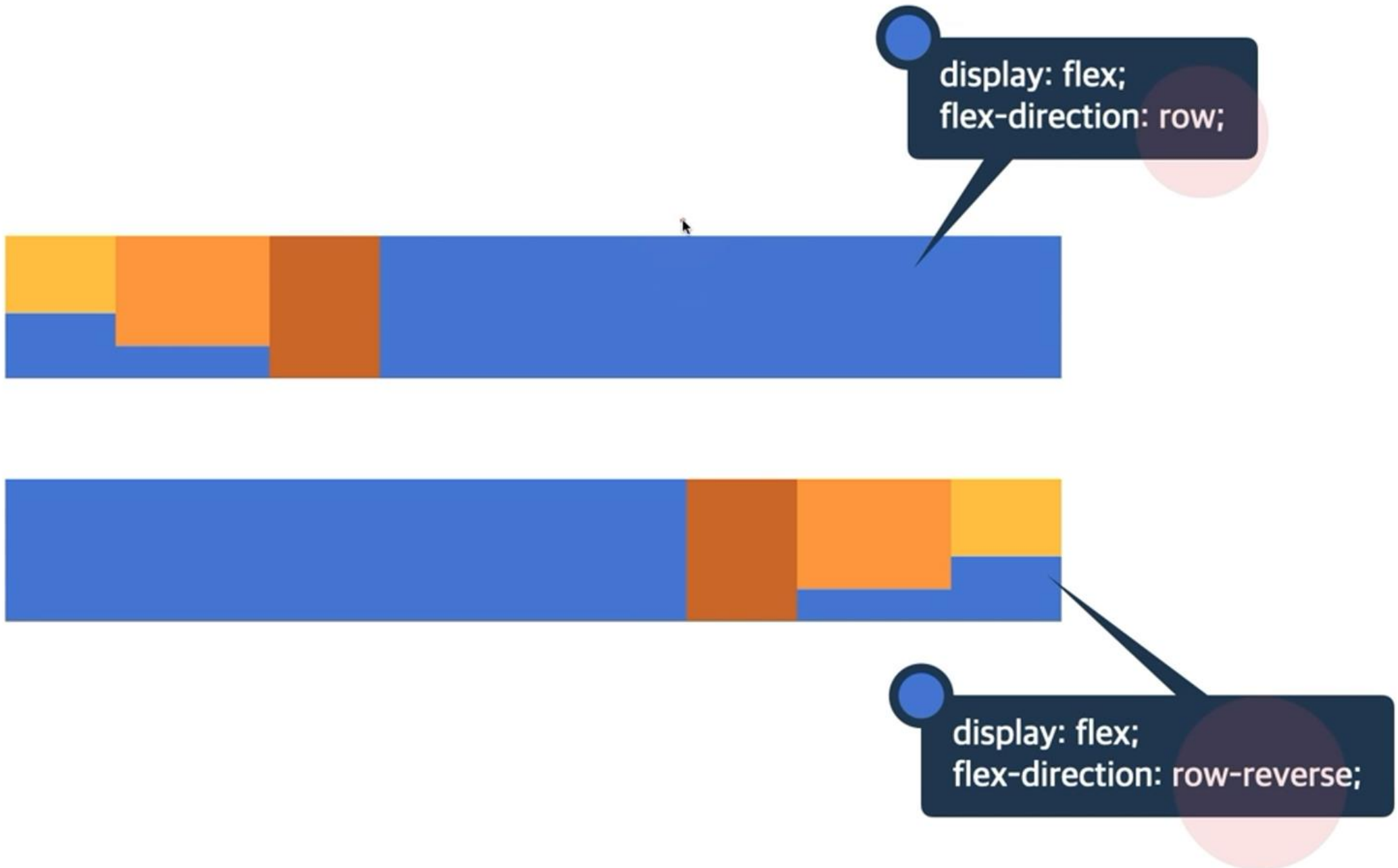
주 축을 설정

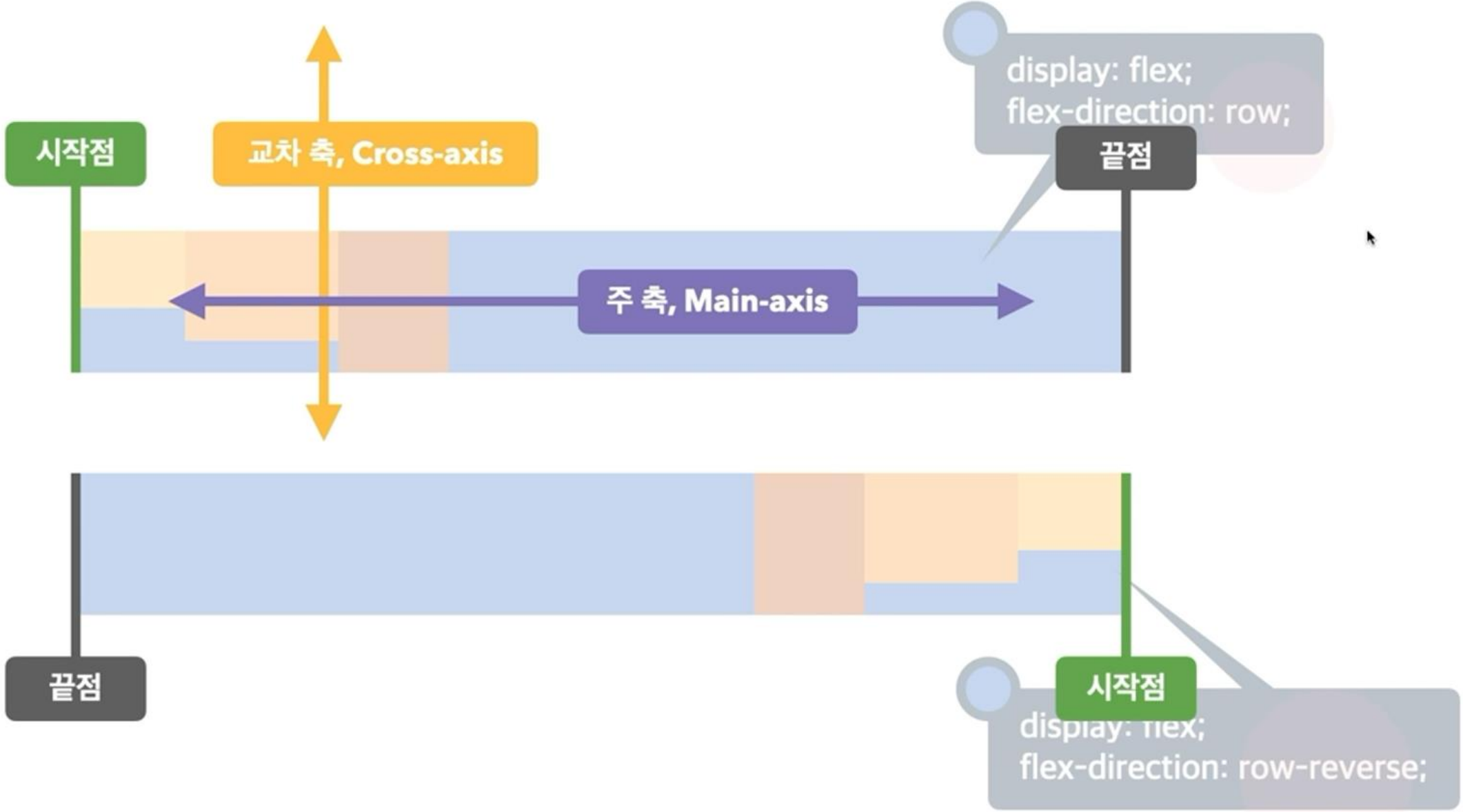
flex-direction

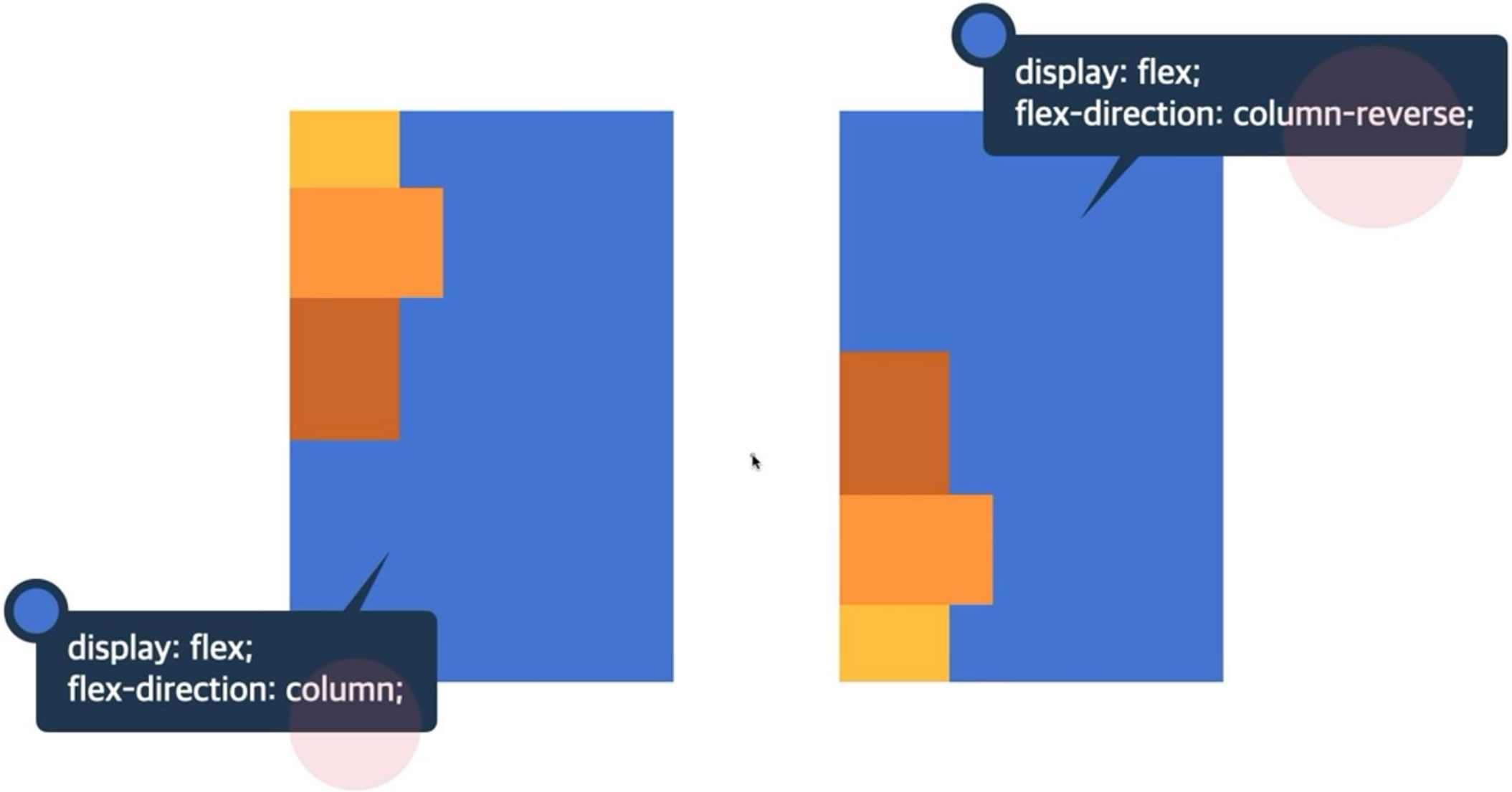
- **row** 행 축 (좌 => 우)
- **row-reverse** 행 축 (우 => 좌)
- **column** 열 축 (위 => 아래)
- **column-reverse** 열 축 (아래 => 위)

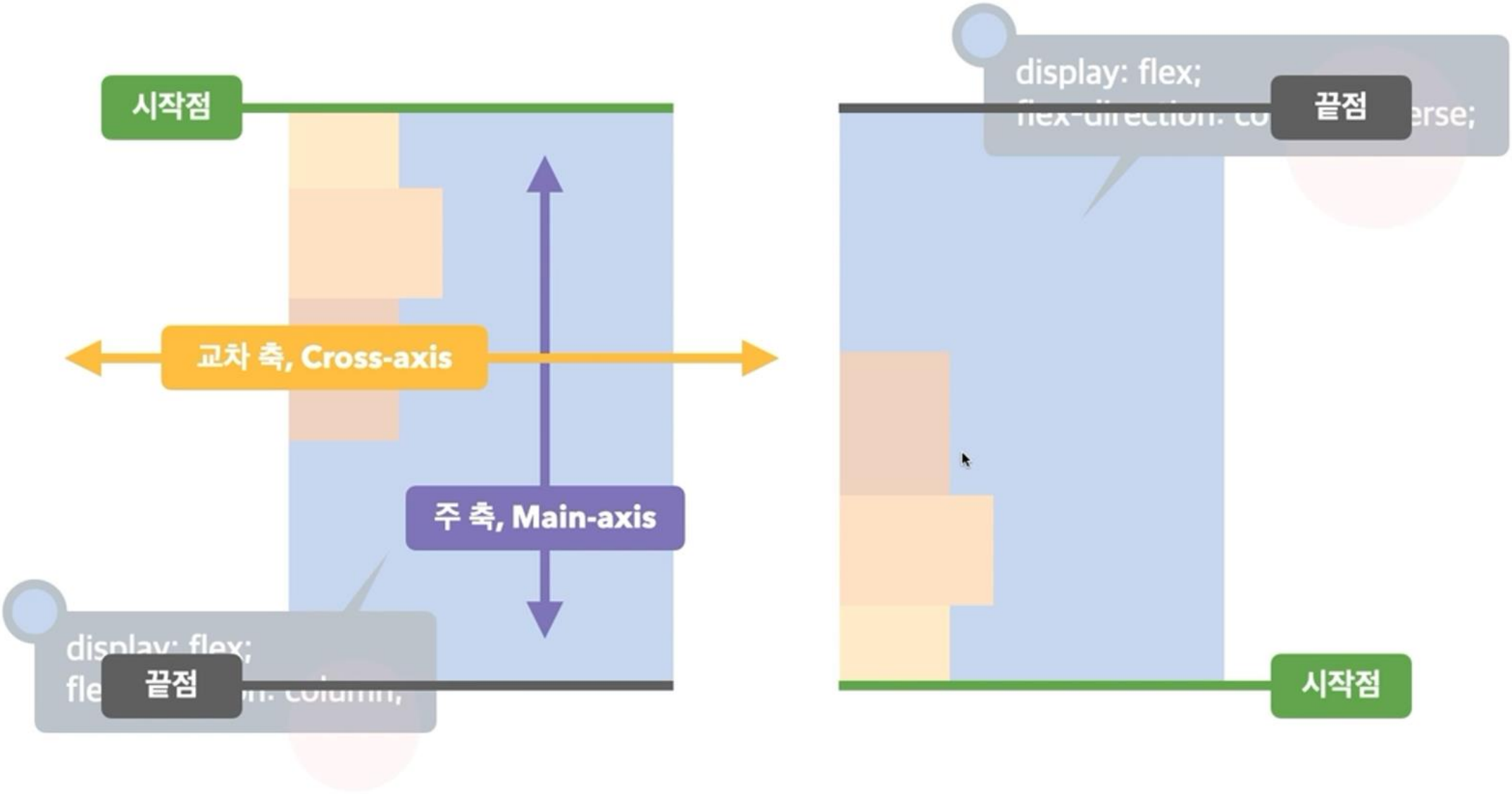
행, Row

열, Column


display: flex;
flex-direction: row;
display: flex;
flex-direction: row-reverse;


display: flex;
flex-direction: row;
시작점
교차 축, Cross-axis
끝점
주 축, Main-axis
끝점
시작점
display: flex;
flex-direction: row-reverse;


display: flex;
flex-direction: column-reverse;
display: flex;
flex-direction: column;


시작점
교차 축, Cross-axis
주 축, Main-axis
끝점
display: flex;
끝점
시작점

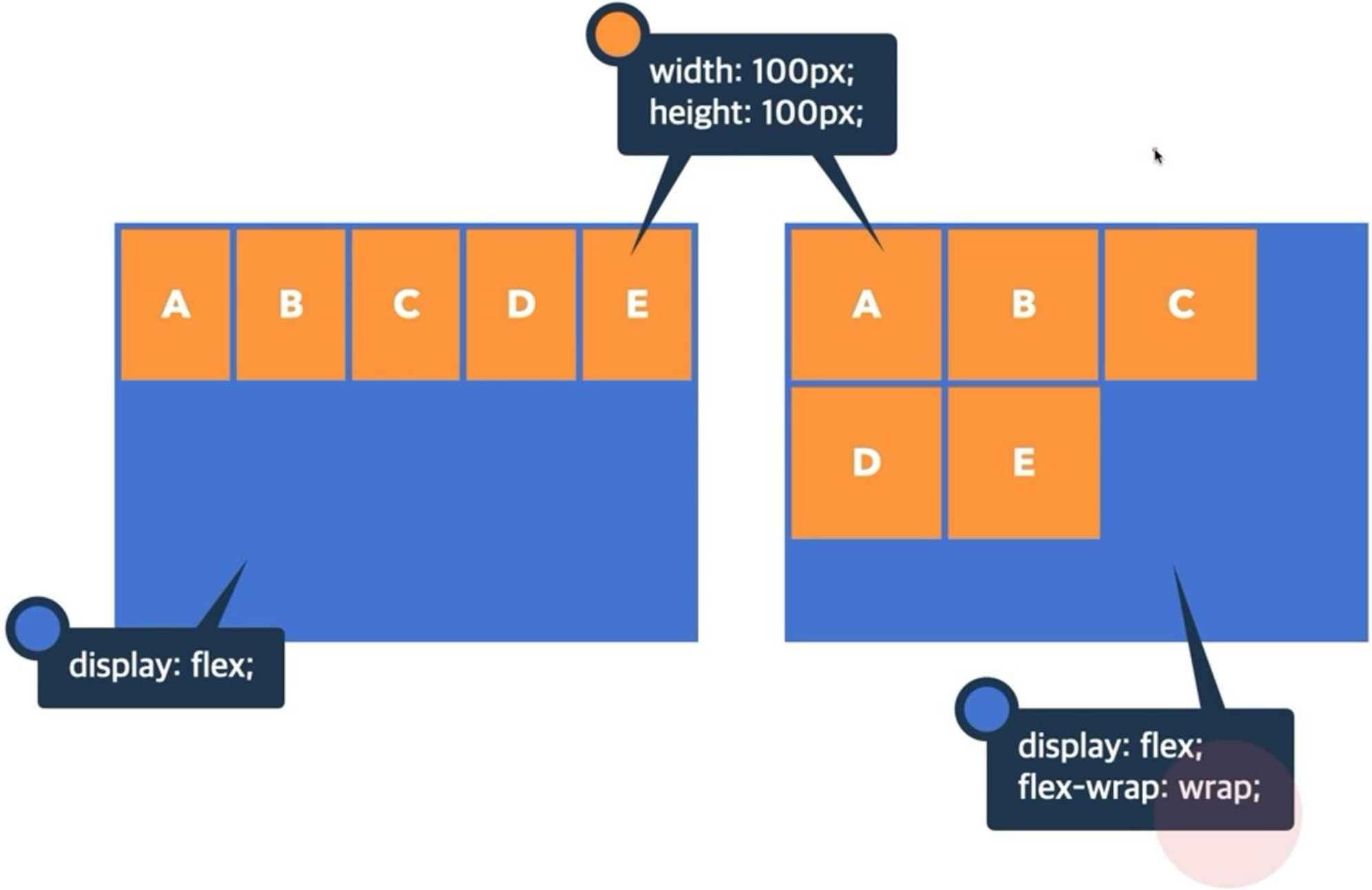
Flex Items 묶음(줄 바꿈) 여부

flex-wrap

nowrap 묶음(줄 바꿈) 없음

wrap 여러 줄로 묶음

wrap-reverse wrap의 반대 방향으로 묶음


width: 100px;
height: 100px;
A
B
C
D
E
A
B
C
D
E
display: flex;
display: flex;
flex-wrap: wrap;

주 축의 정렬 방법

justify-content

flex-start Flex Items를 시작점으로 정렬

flex-end Flex Items를 끝점으로 정렬

center Flex Items를 가운데 정렬

space-between 각 Flex Item 사이를 균등하게 정렬

space-around 각 Flex Item의 외부 여백을 균등하게 정렬

display: flex;

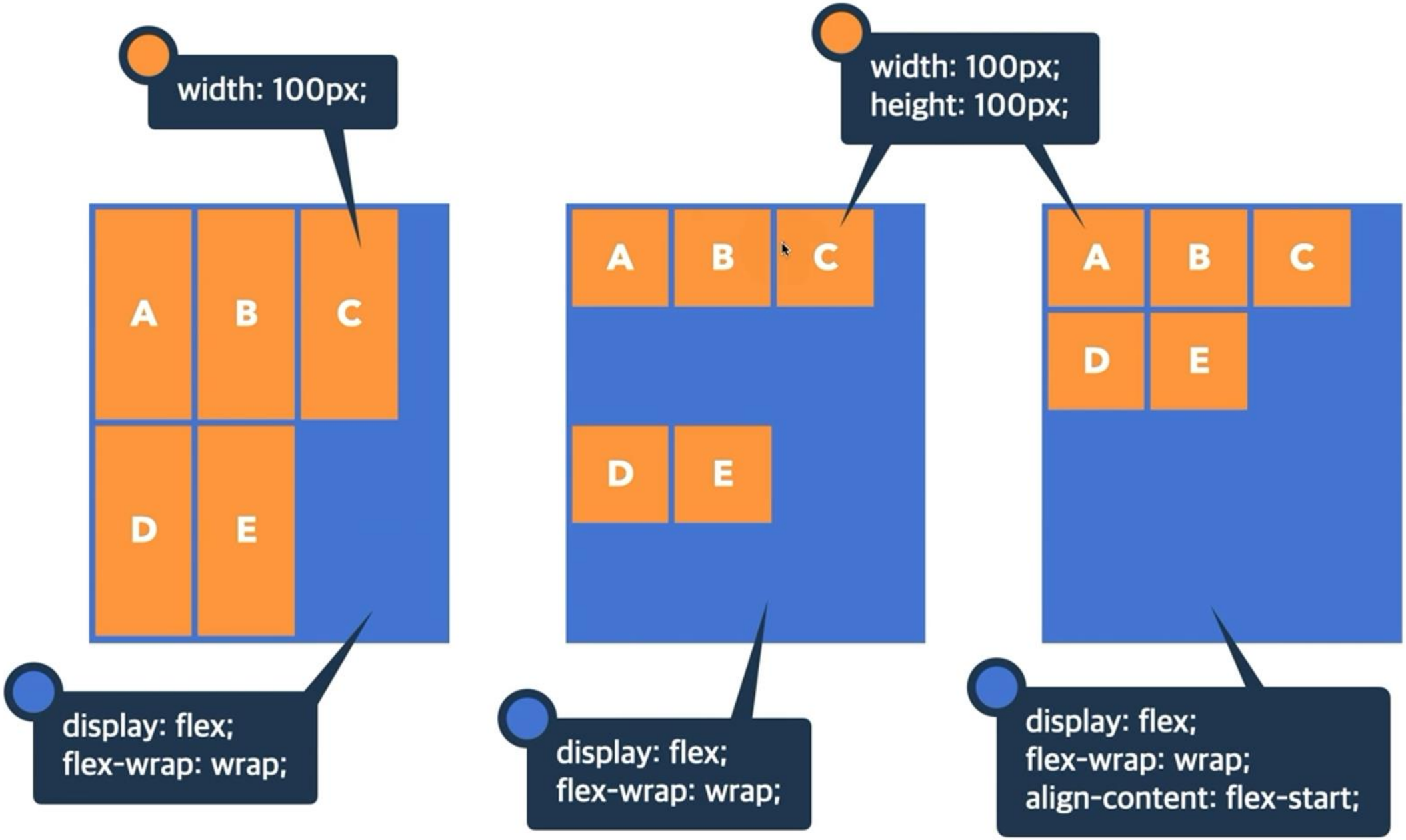
display: flex;
justify-content: flex-end;

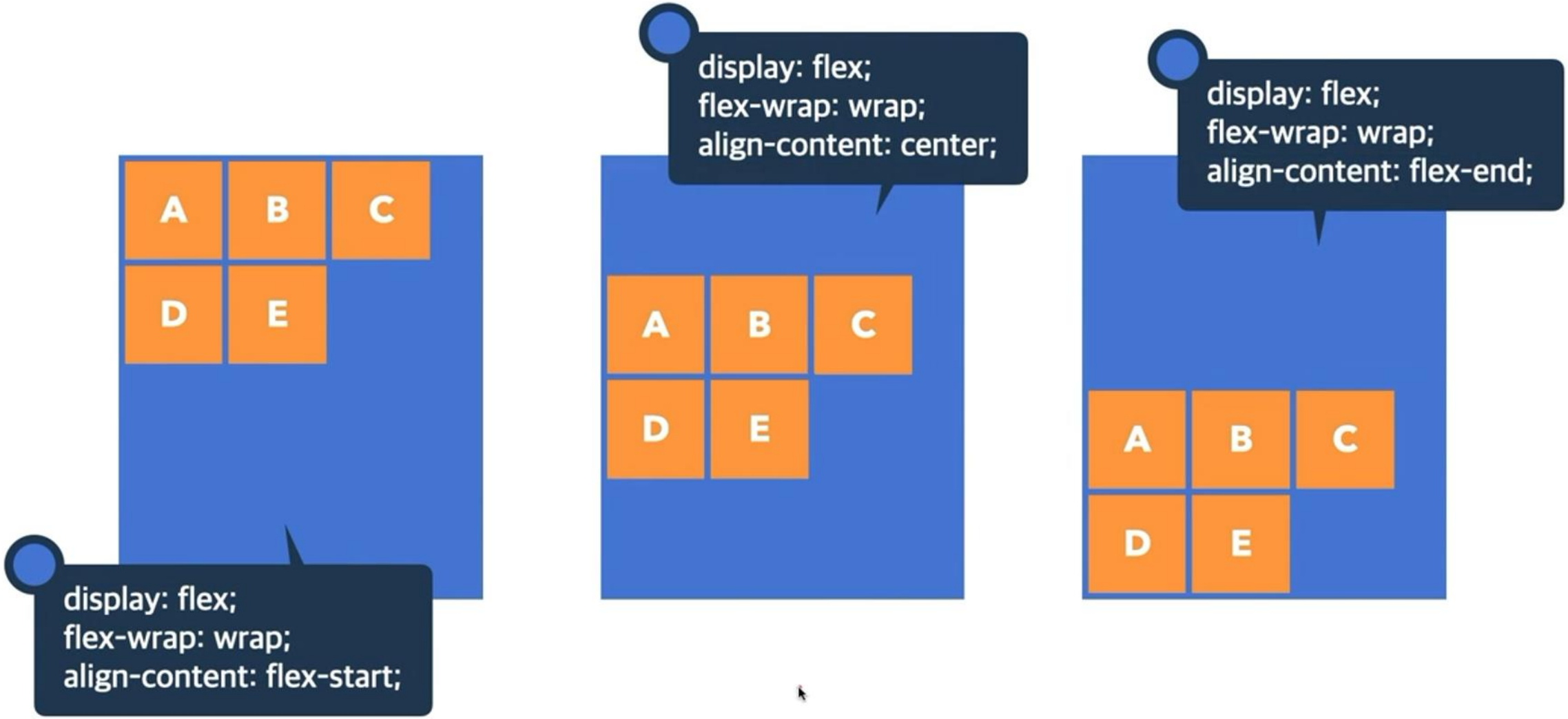
display: flex;
justify-content: center;

교차 축의 여러 줄 정렬 방법

align-content

stretch	Flex Items를 시작점으로 정렬
flex-start	Flex Items를 시작점으로 정렬
flex-end	Flex Items를 끝점으로 정렬
center	Flex Items를 가운데 정렬
space-between	각 Flex Item 사이를 균등하게 정렬
space-around	각 Flex Item의 외부 여백을 균등하게 정렬

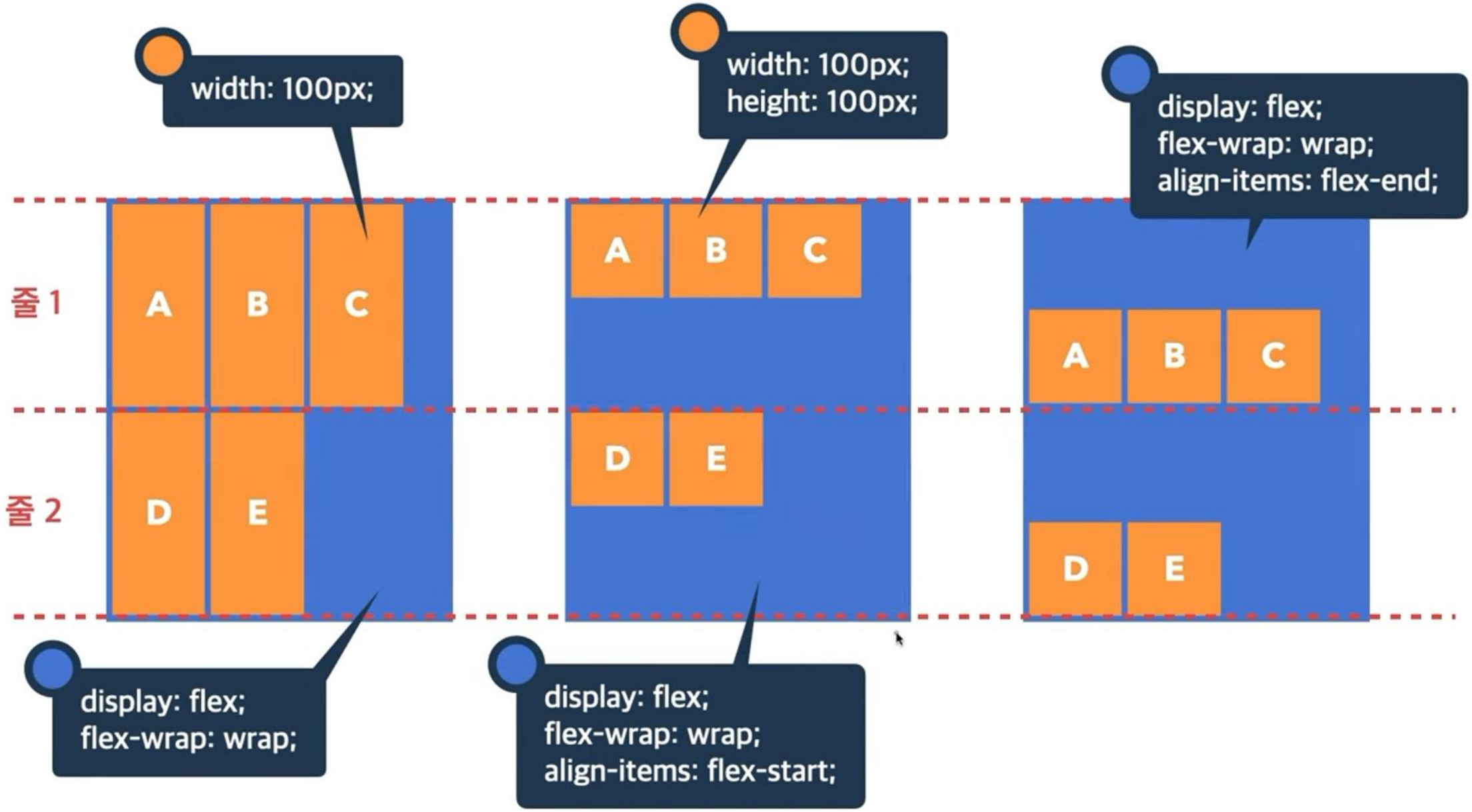

width: 100px;
A
B
C
D
E
display: flex;
flex-wrap: wrap;
width: 100px;
height: 100px;
A
B
C
D
E
display: flex;
flex-wrap: wrap;
A
B
C
D
E
display: flex;
flex-wrap: wrap;
align-content: flex-start;

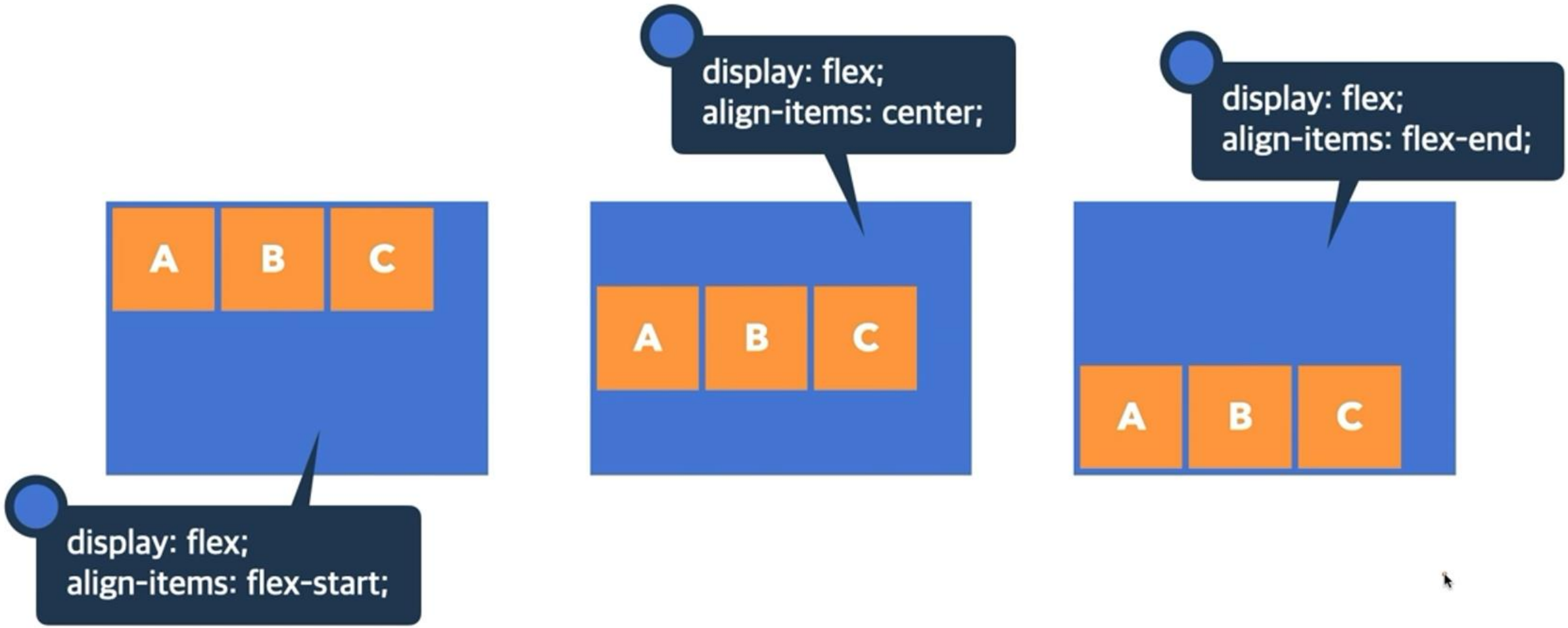

A
B
C
D
E
display: flex;
flex-wrap: wrap;
align-content: flex-start;
display: flex;
flex-wrap: wrap;
align-content: center;
A
B
C
D
E
display: flex;
flex-wrap: wrap;
align-content: flex-end;
A
B
C
D
E

교차 축의 한 줄 정렬 방법

align-items

stretch	Flex Items를 교차 축으로 늘림
flex-start	Flex Items를 각 줄의 시작점으로 정렬
flex-end	Flex Items를 각 줄의 끝점으로 정렬
center	Flex Items를 각 줄의 가운데 정렬
baseline	Flex Items를 각 줄의 문자 기준선에 정렬


width: 100px;
width: 100px;
height: 100px;
display: flex;
flex-wrap: wrap;
align-items: flex-end;
줄 1
줄 2
A
B
C
D
E
display: flex;
flex-wrap: wrap;
display: flex;
flex-wrap: wrap;
align-items: flex-start;


display: flex;
align-items: center;
display: flex;
align-items: flex-end;
A
B
C
A
B
C
A
B
C
display: flex;
align-items: flex-start;

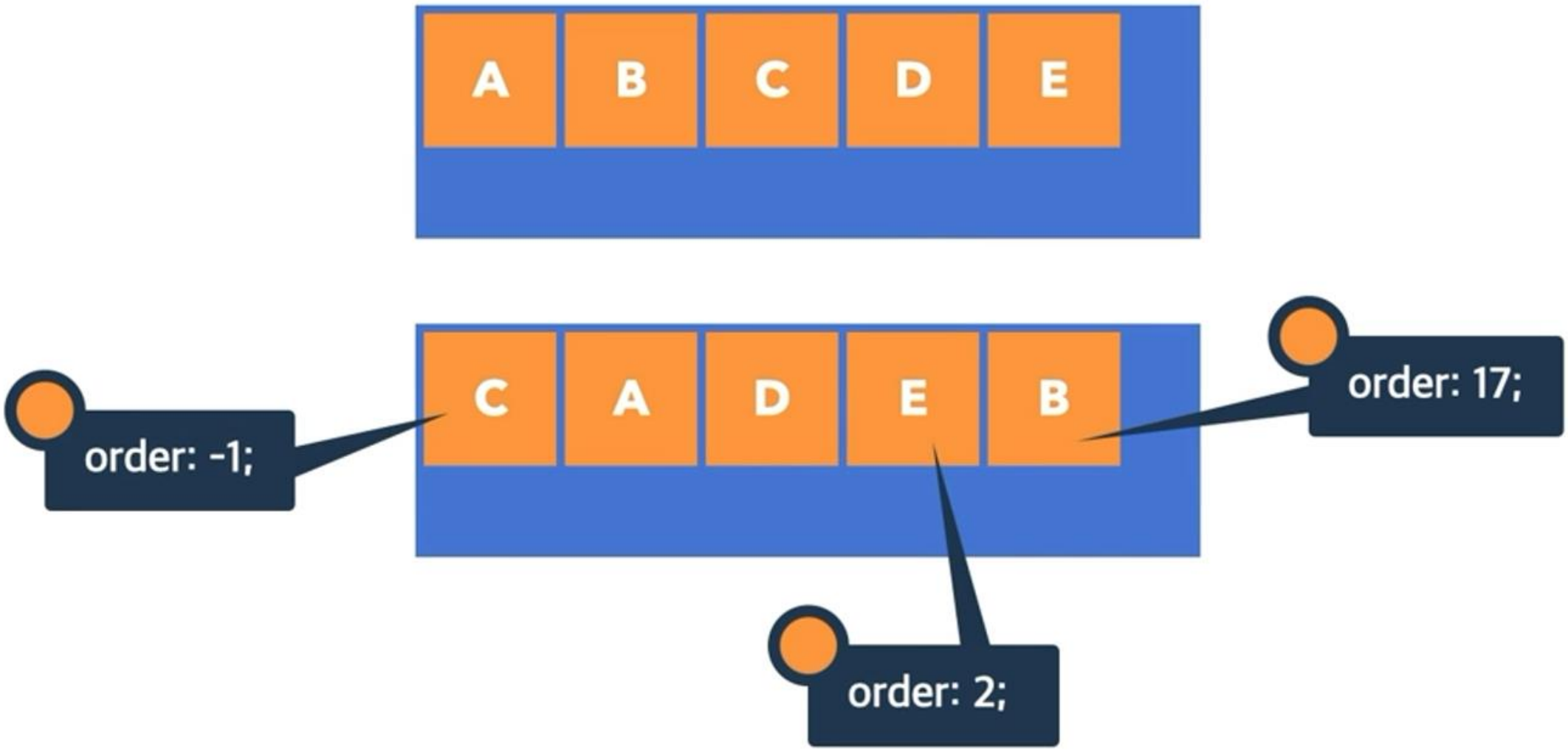
Flex

Item 속성

Flex Item의 순서

order

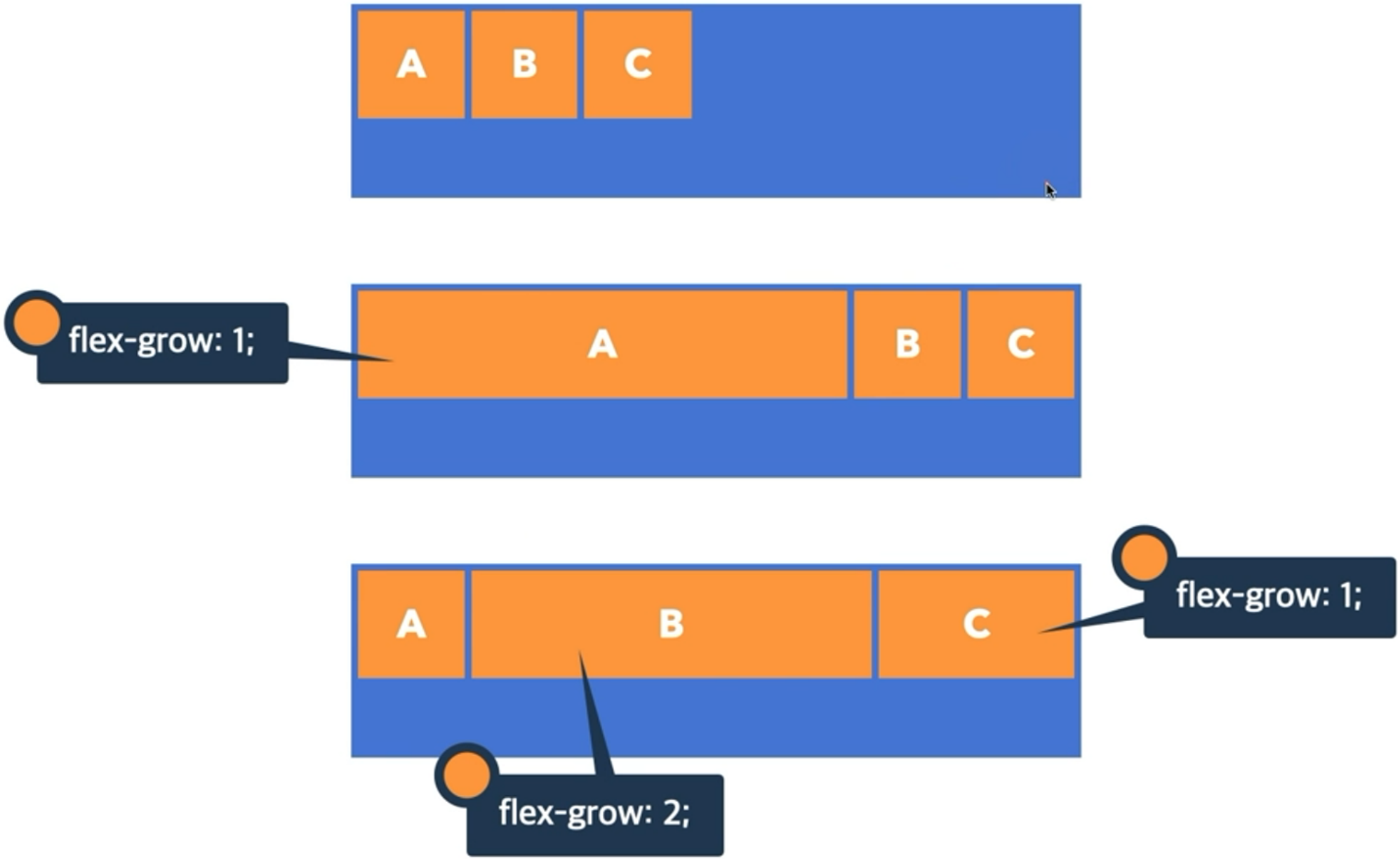
0	순서 없음
숫자	숫자가 작을 수록 먼저


A
B
C
D
E
C
A
D
E
B
order: -1;
order: 17;
order: 2;

Flex Item의 증가 너비 비율

flex-grow

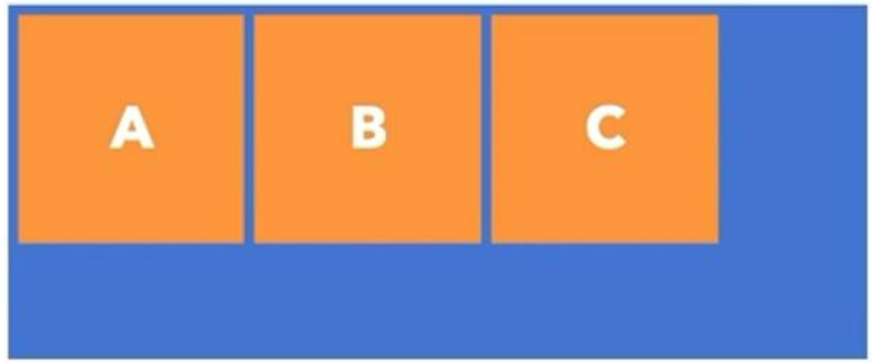
0	증가 비율 없음
숫자	증가 비율

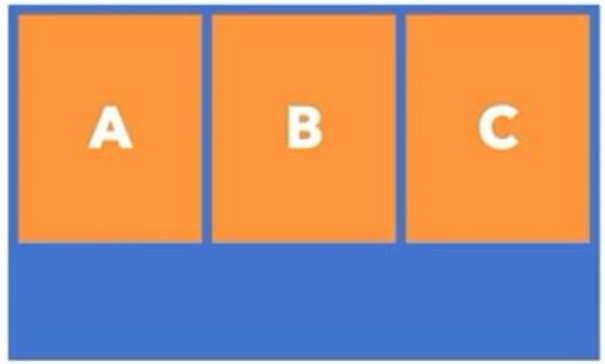

A
B
C
flex-grow: 1;
A
B
C
A
B
C
flex-grow: 1;
flex-grow: 2;

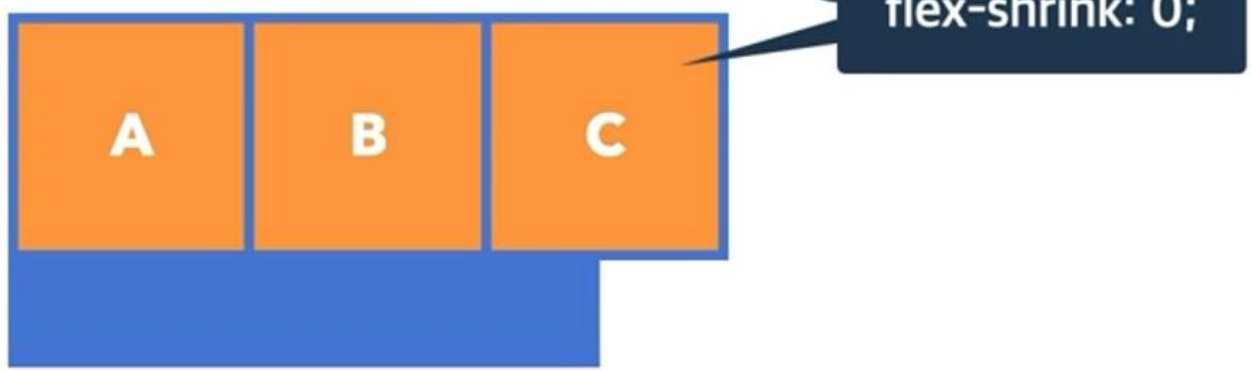
Flex Item의 감소 너비 비율

flex-shrink

1	Flex Container 너비에 따라 감소 비율 적용
숫자	감소 비율


A
B
C


A
B
C

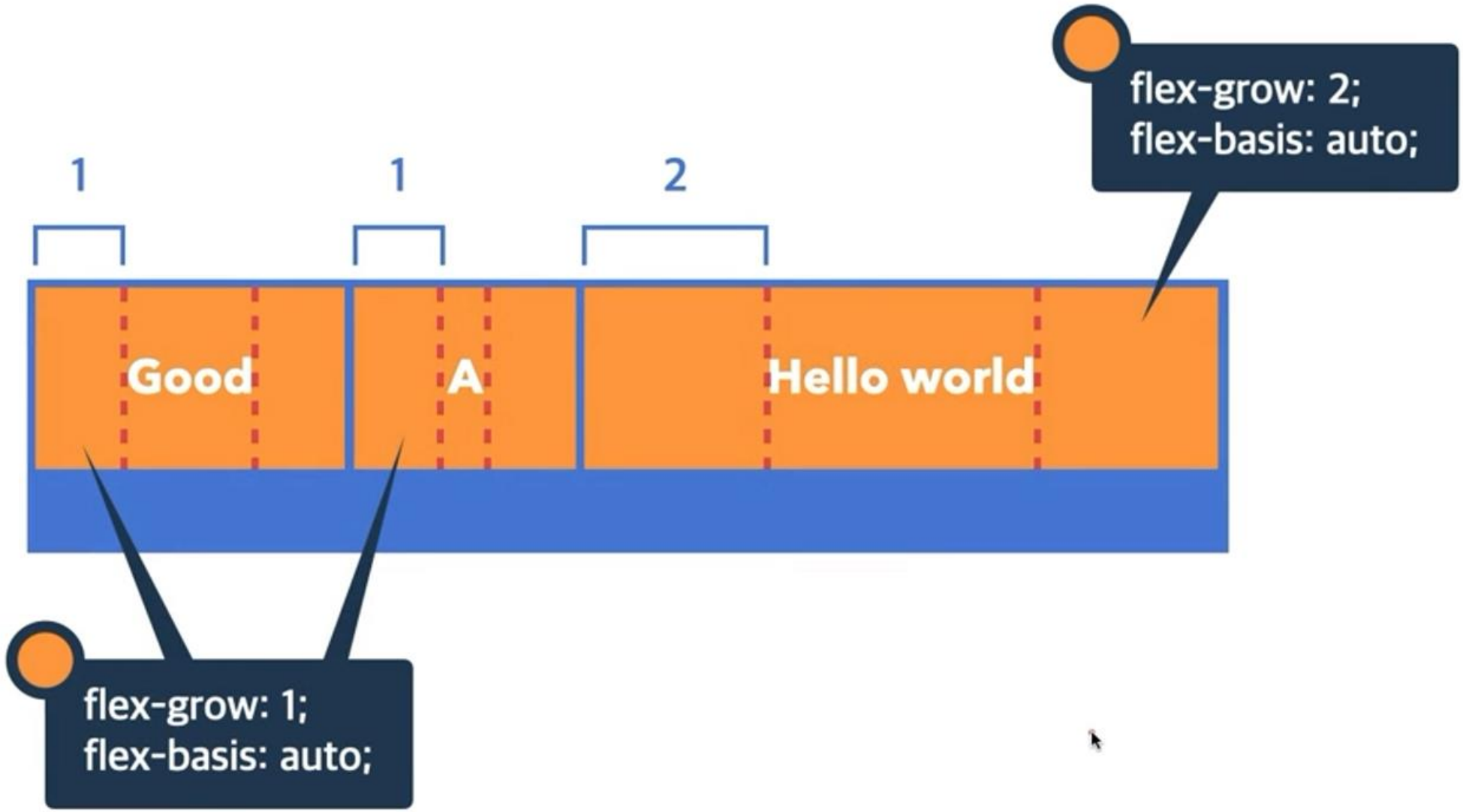

A
B
C
flex-shrink: 0;

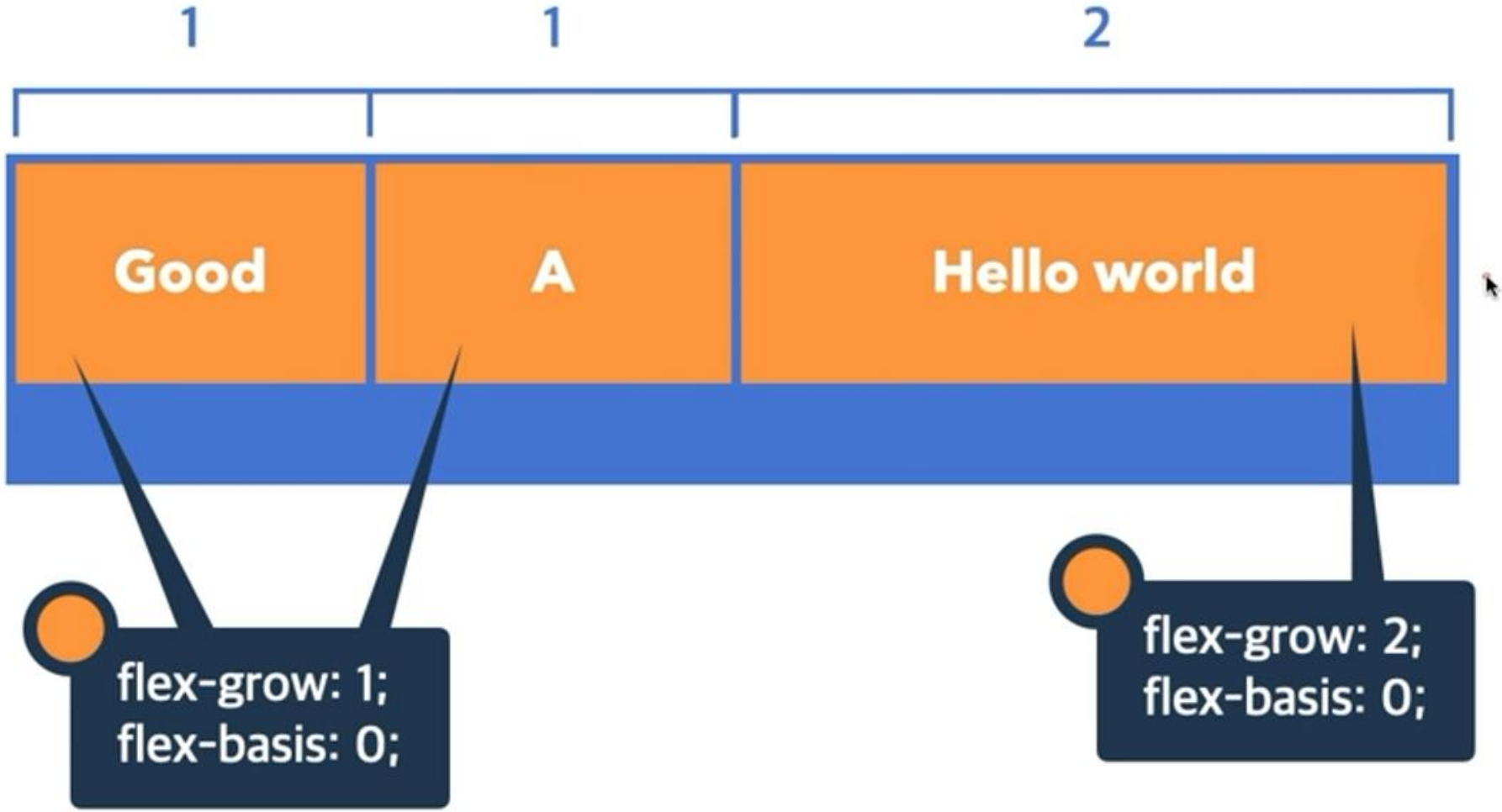
Flex Item의 공간 배분 전 기본 너비

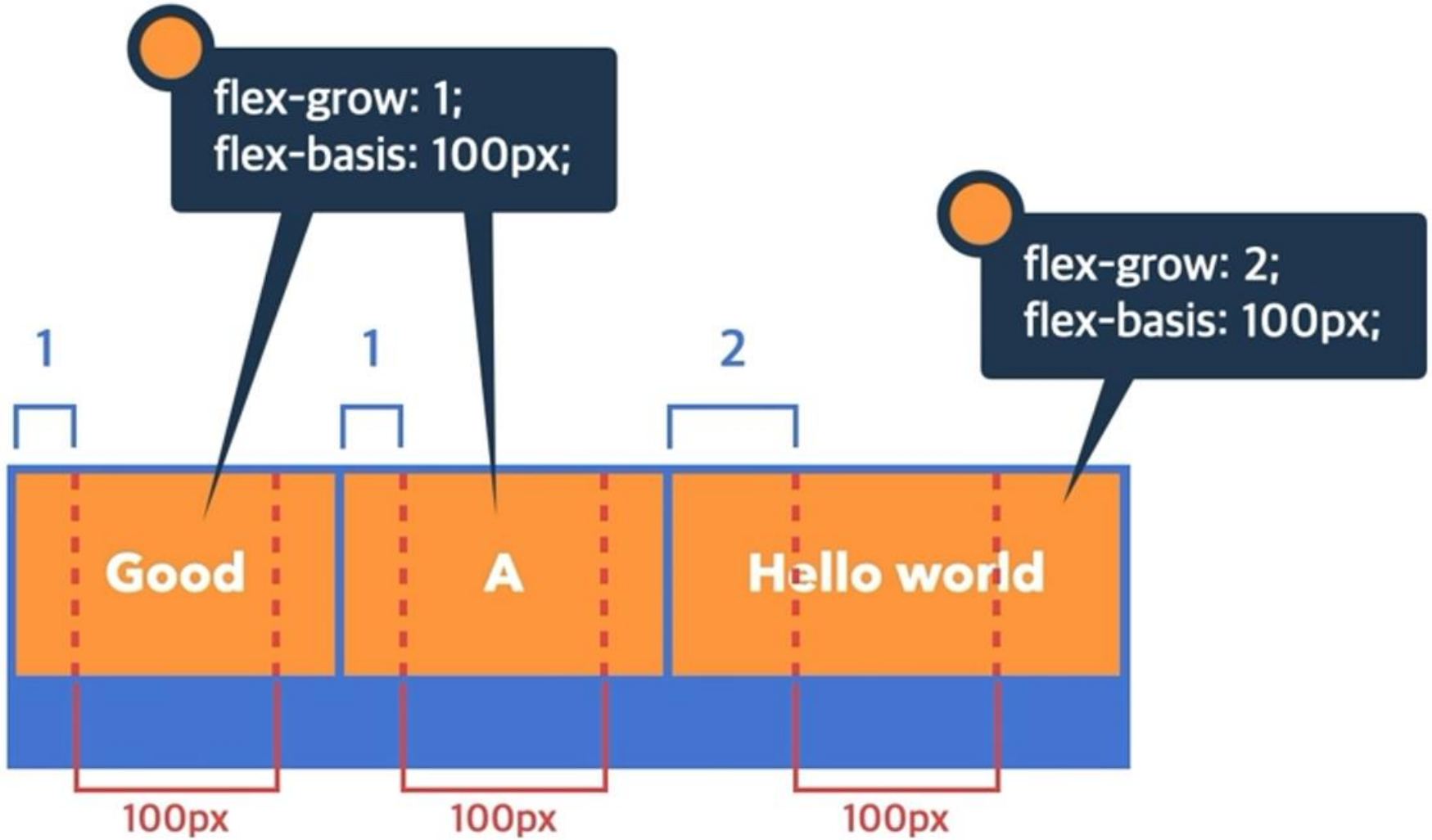
flex-basis

auto 요소의 Content 너비

단위 px, em, rem 등 단위로 지정


flex-grow: 2;
flex-basis: auto;
1
1
2
Good
A
Hello world
flex-grow: 1;
flex-basis: auto;


1
1
2
Good
A
Hello world
flex-grow: 1;
flex-basis: 0;
flex-grow: 2;
flex-basis: 0;


flex-grow: 1;
flex-basis: 100px;
flex-grow: 2;
flex-basis: 100px;
1
1
2
Good
A
Hello world
100px
100px
100px


수고하셨습니다!