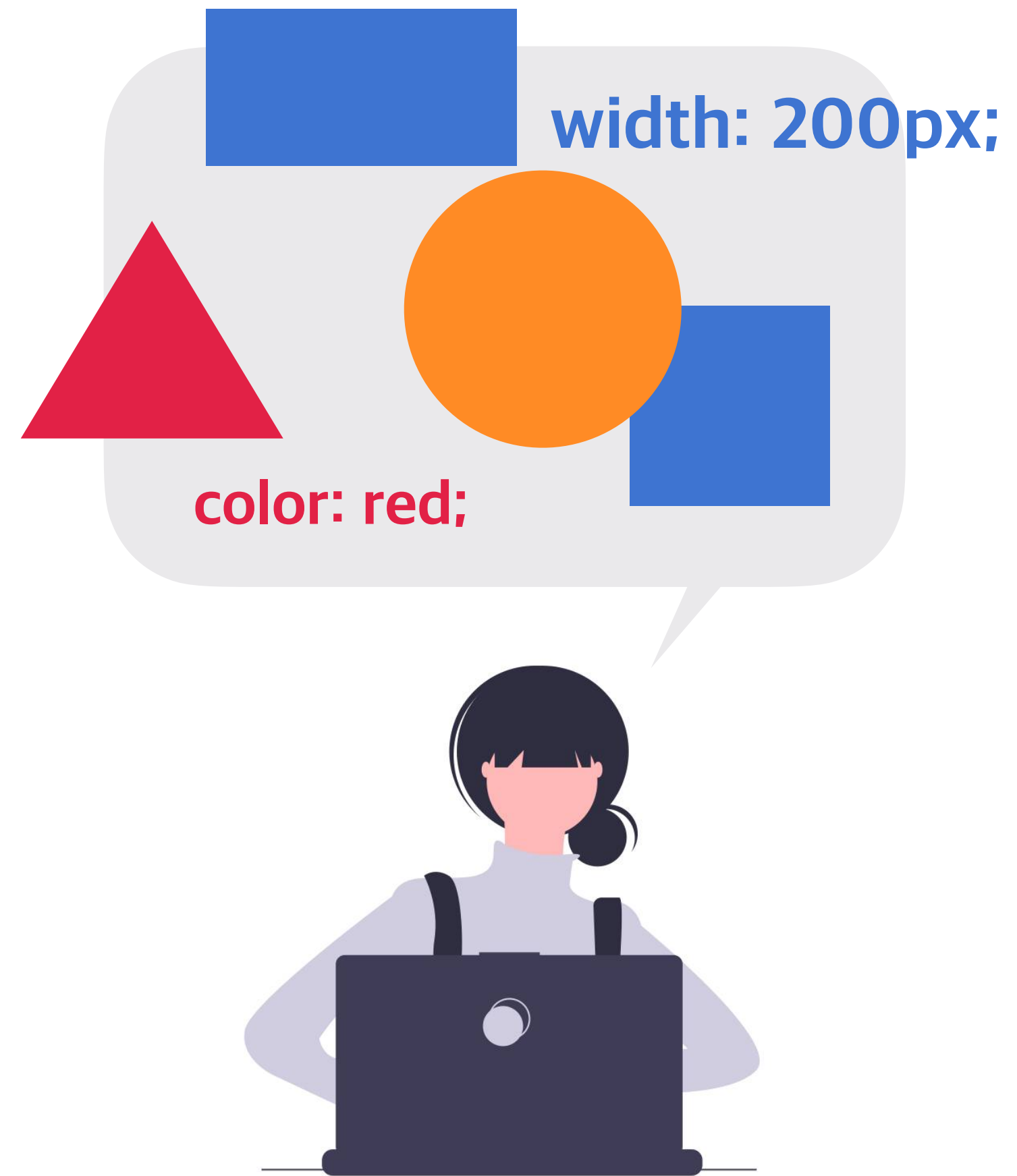


<https://heropy.blog>



박영웅

**속성(Properties)**



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

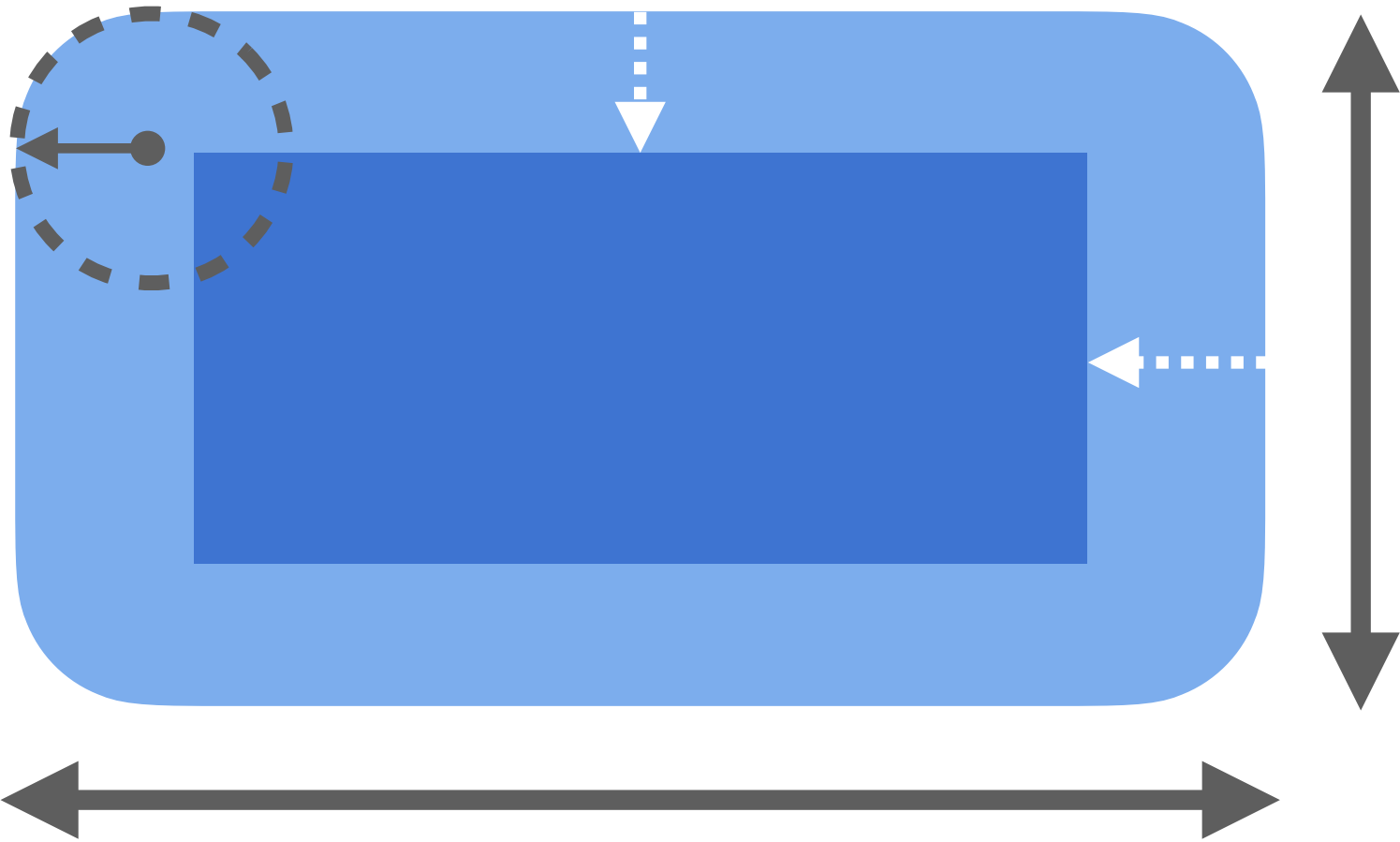
띄움

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

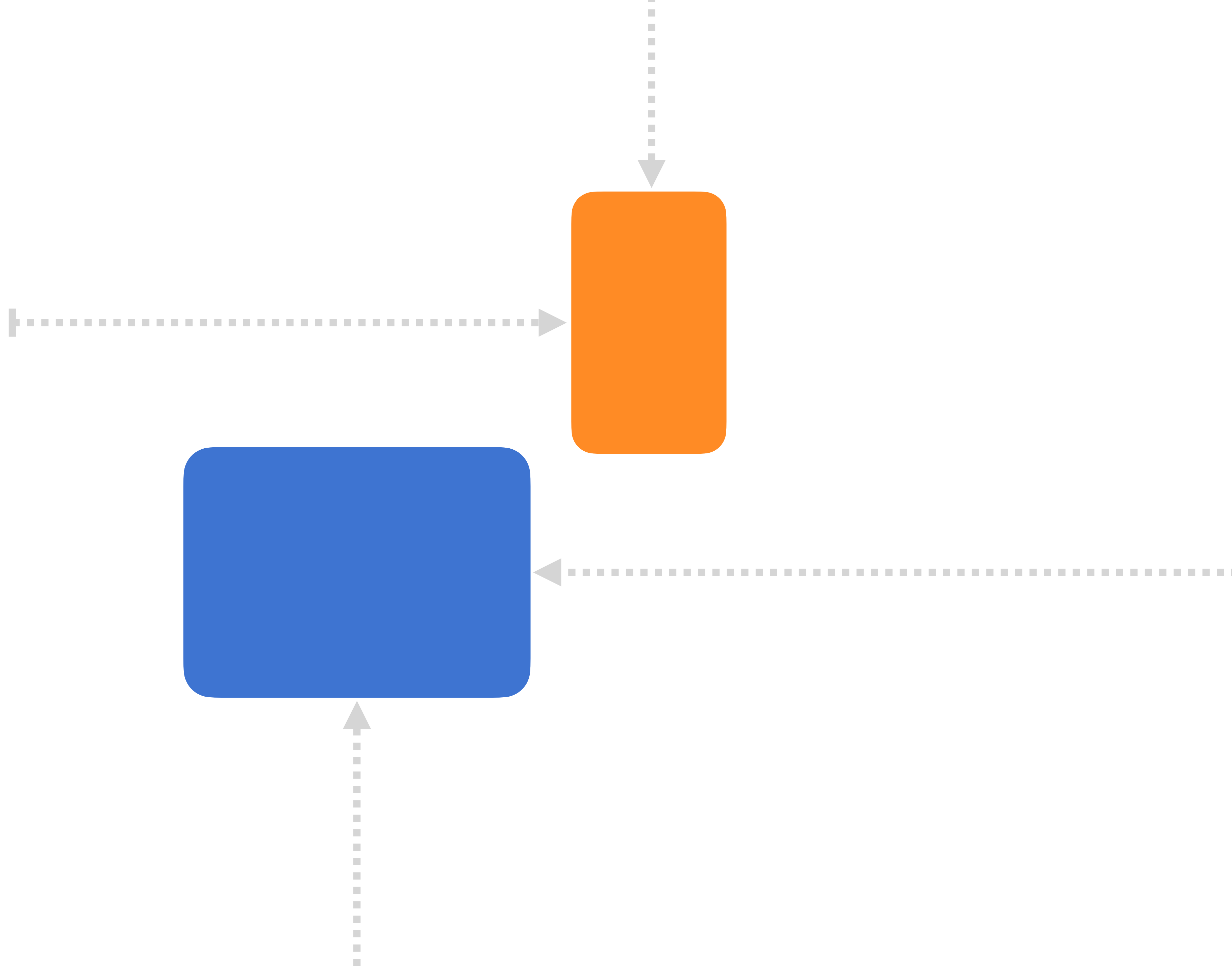
띄움

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

**변화**

띄움

애니메이션

그리드

다단

필터



박스 모델  
글꼴, 문자

배경

배치

플렉스(정렬)

전환

변환

**띠움**

애니메이션

그리드

다단

필터

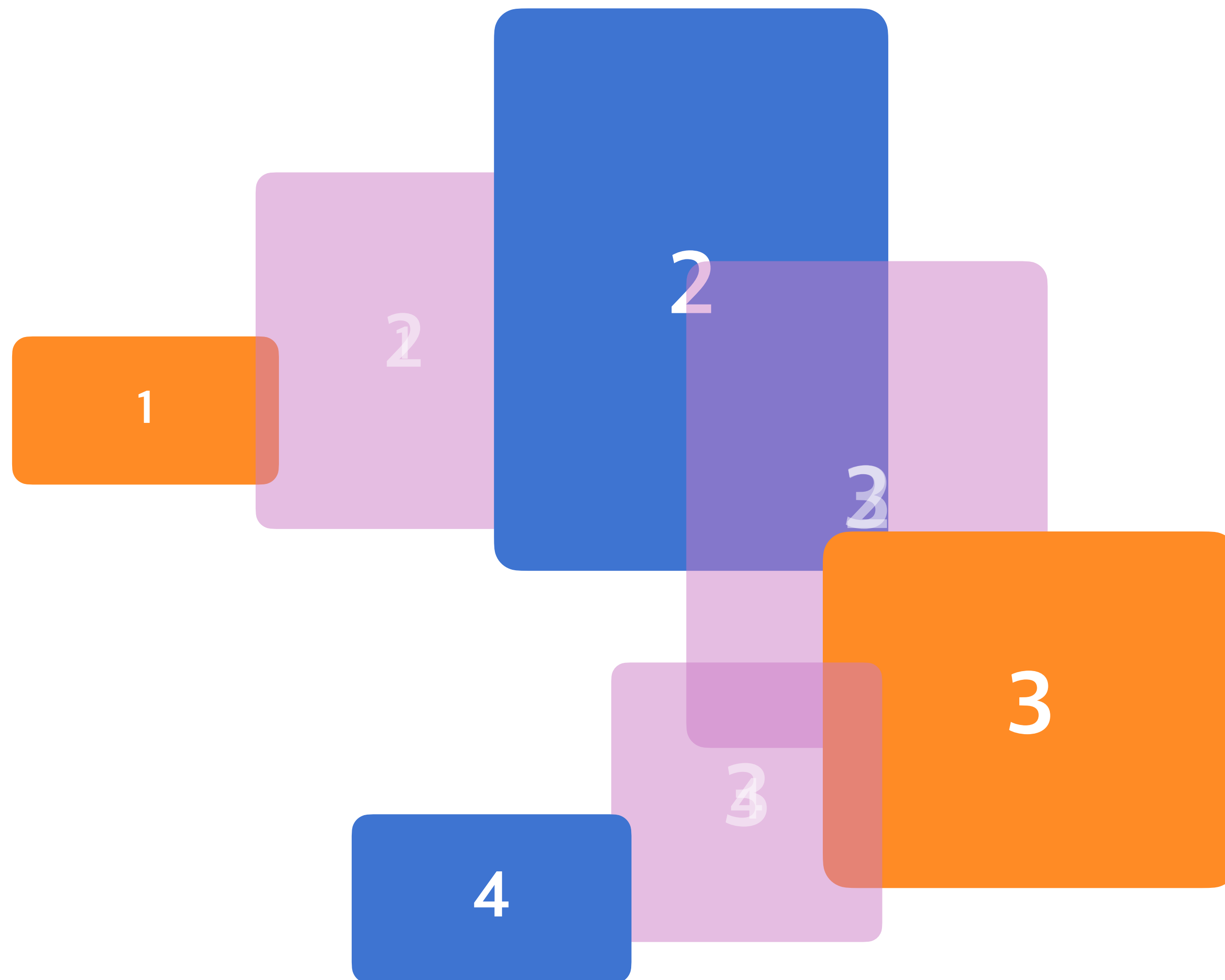


Lorem

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

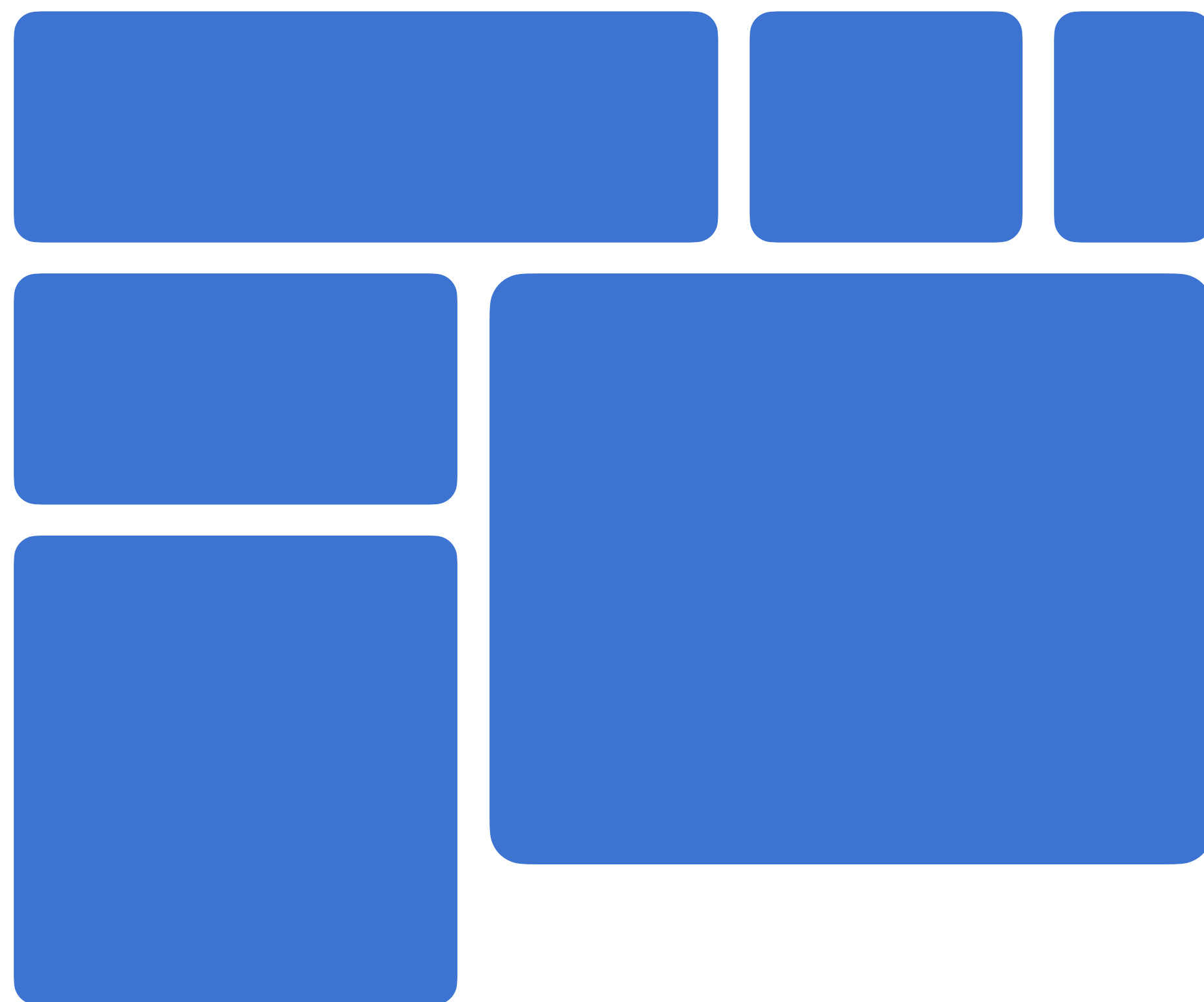
띄움

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

요소의 가로/세로 너비

# width, height

기본값

(요소에 이미 들어있는 속성의 값)



**auto**

브라우저가 너비를 계산

**단위**

px, em, vw 등 단위로 지정

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

`<span></span>`

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



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

<div></div>

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

auto

부모 요소의 크기만큼 자동으로 늘어남!

Hello

포함한 콘텐츠 크기만큼  
자동으로 줄어듦!

World

auto

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

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

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

요소의 외부 여백(공간)을 지정하는 단축 속성

가로(세로) 너비가 있는 요소의  
가운데 정렬에 활용해요!

# margin

음수를 사용할 수 있어요!

0

외부 여백 없음

auto

브라우저가 여백을 계산

단위

px, em, vw 등 단위로 지정

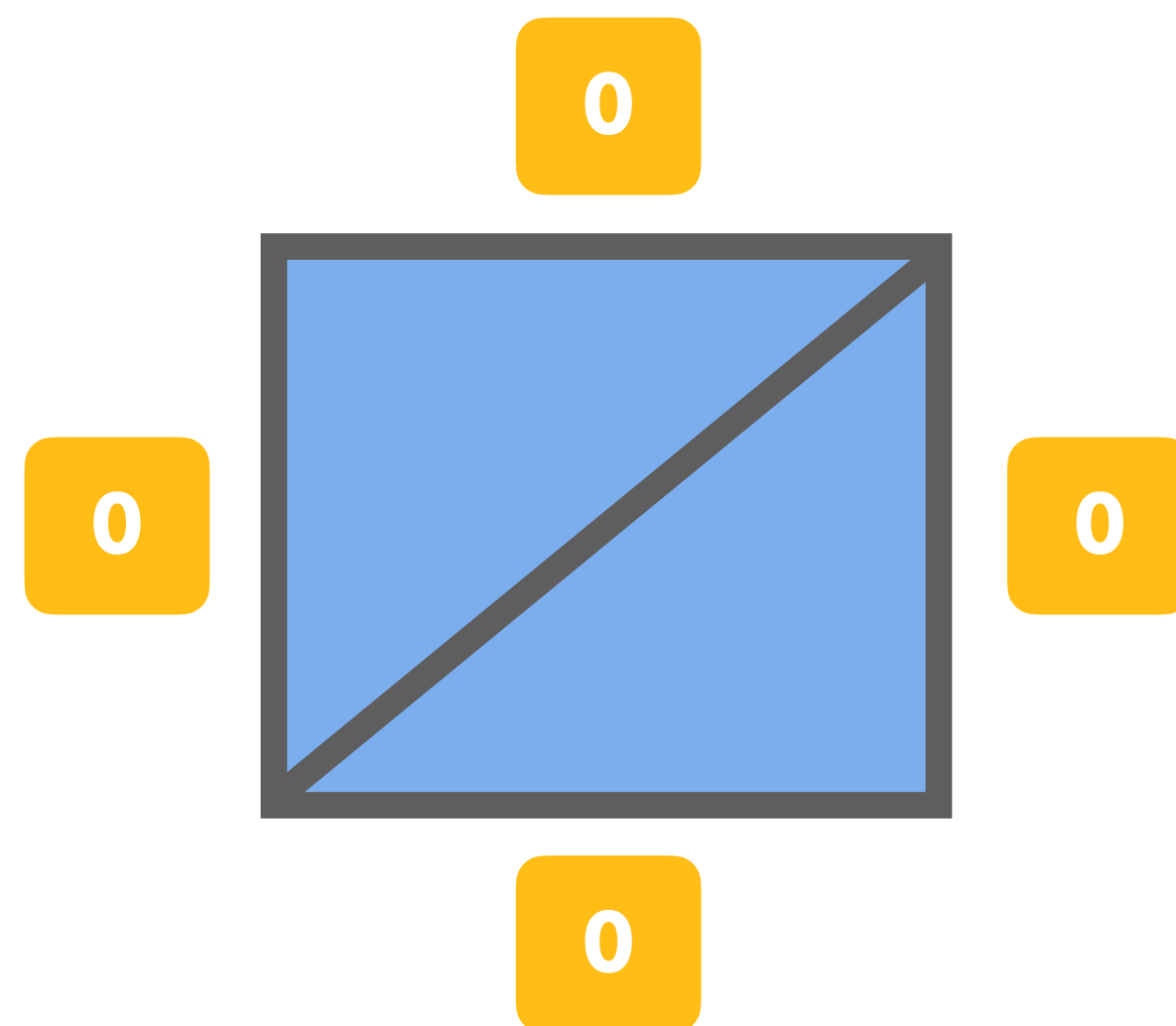
%

부모 요소의 가로 너비에 대한 비율로 지정



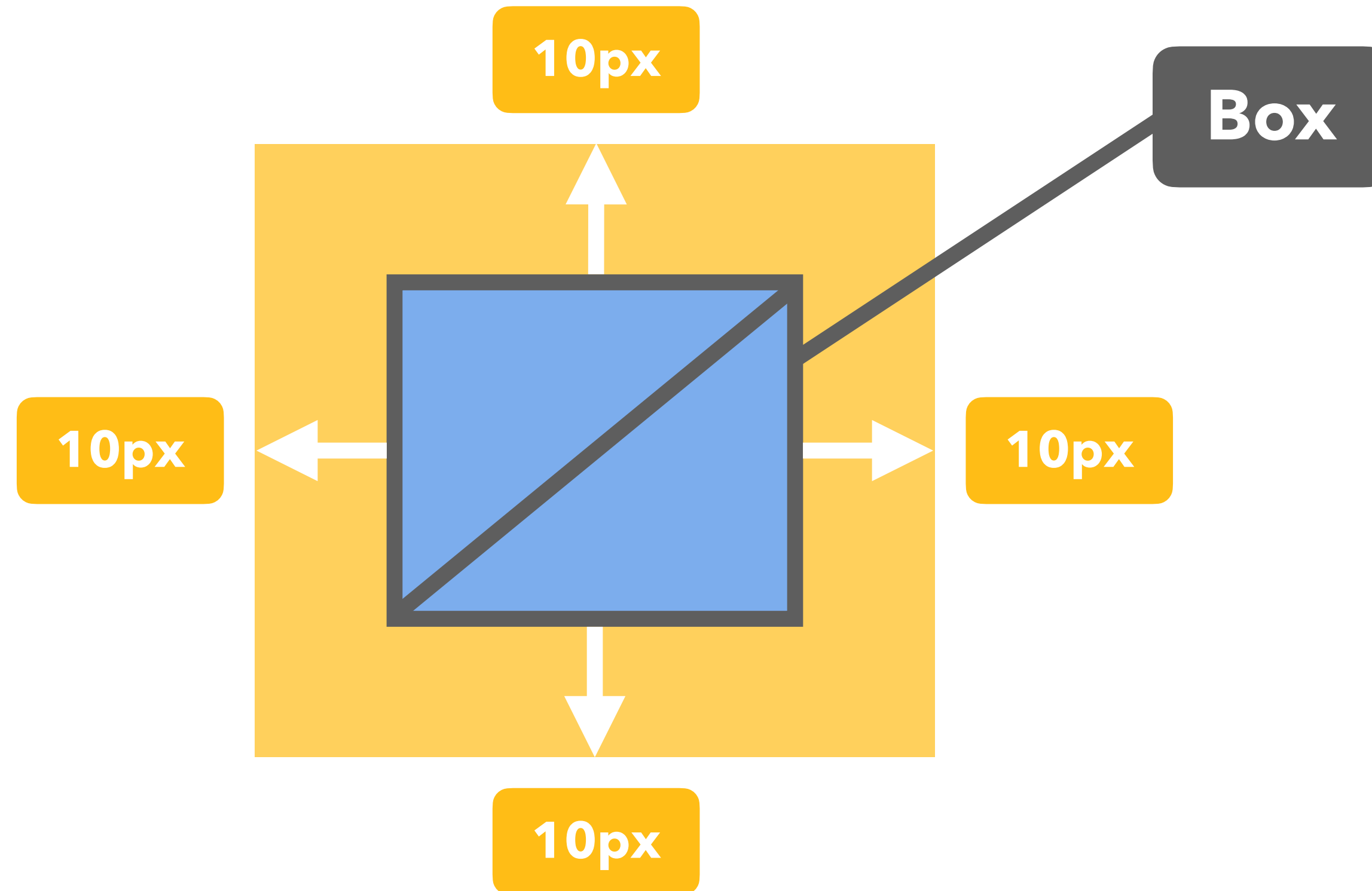
margin: 0;

top, right, bottom, left



margin: 10px;

top, right, bottom, left

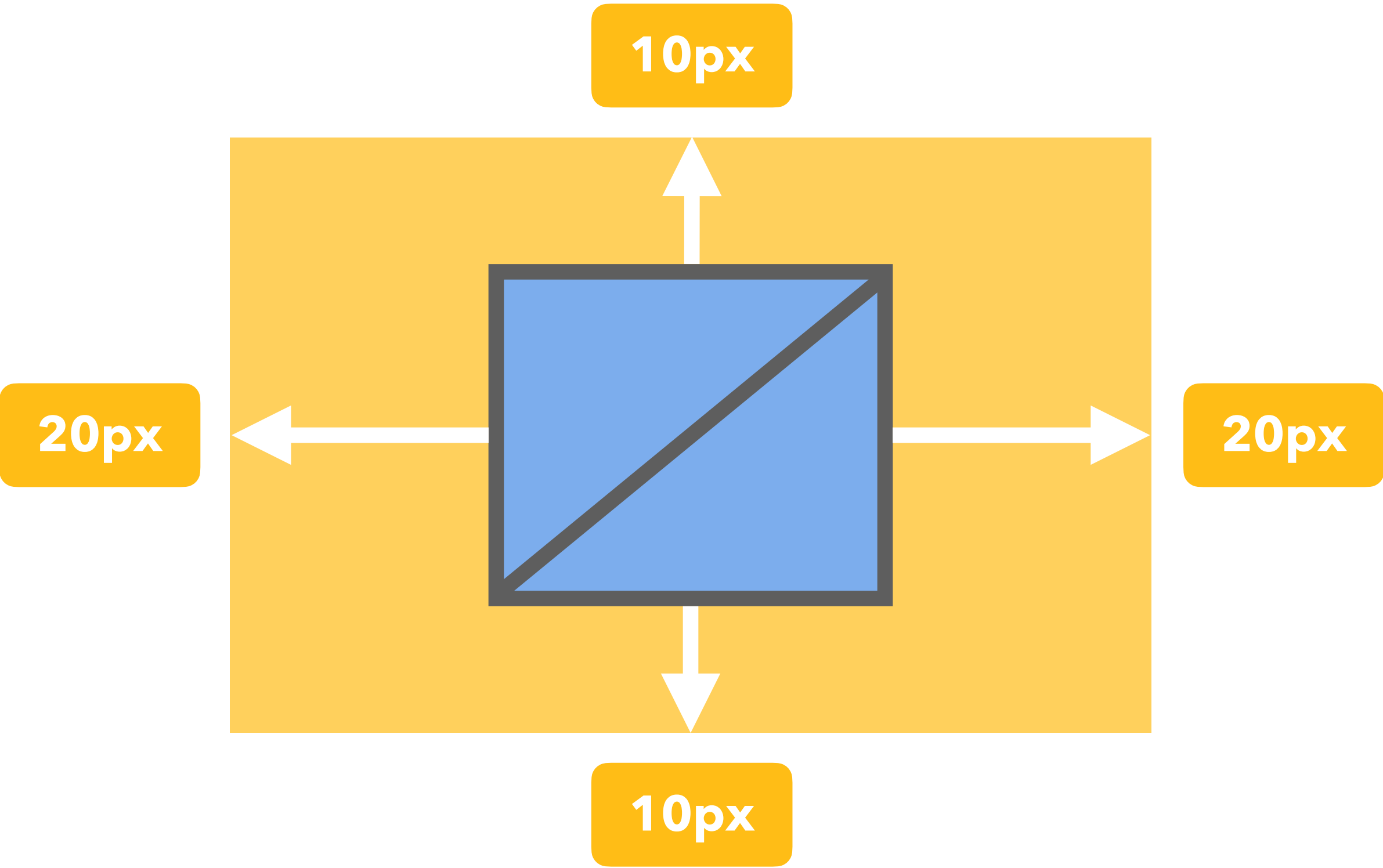


margin: 10px 20px;

top, bottom

left, right

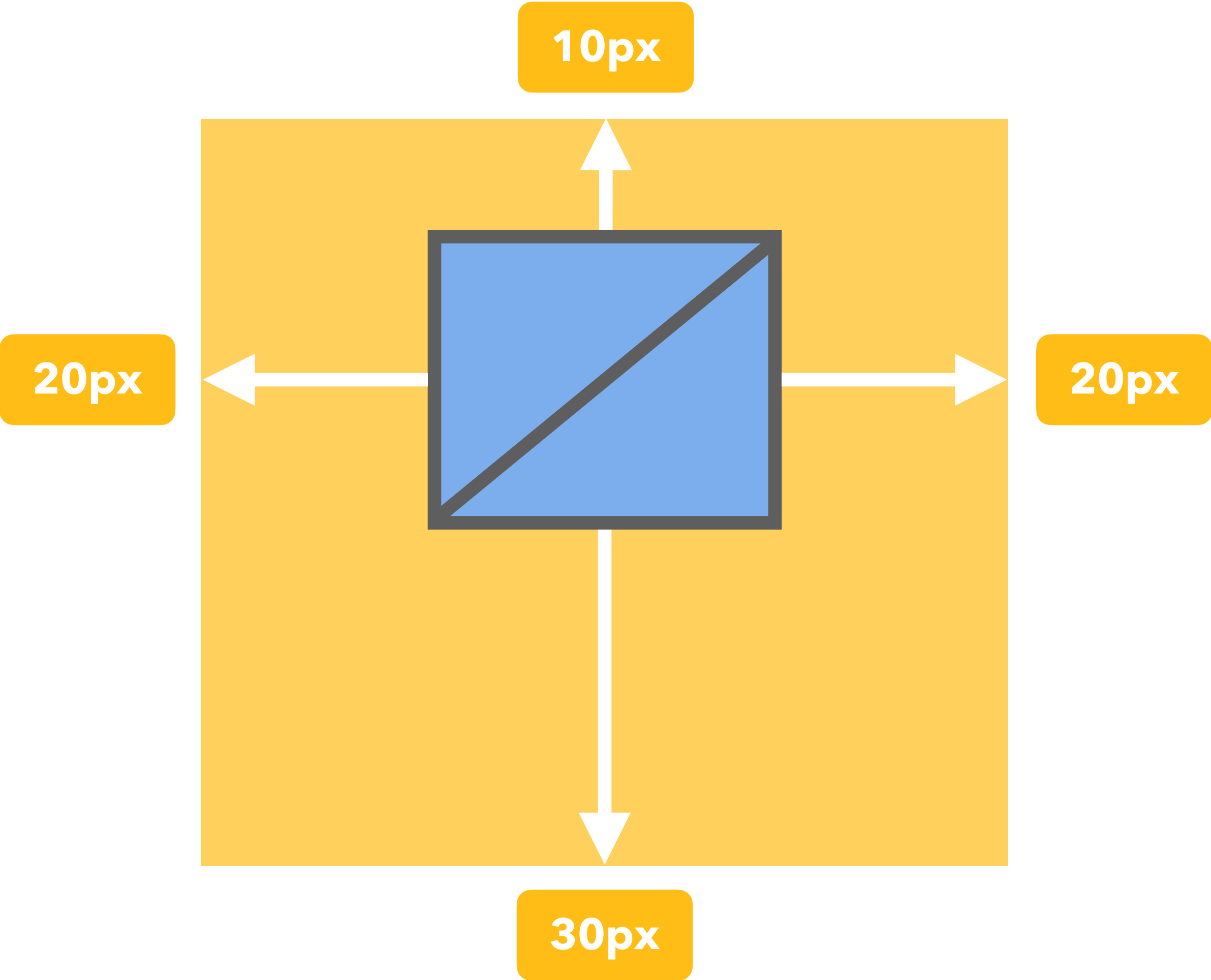
상하.좌우.



margin: 10px 20px 30px;

top left, right bottom

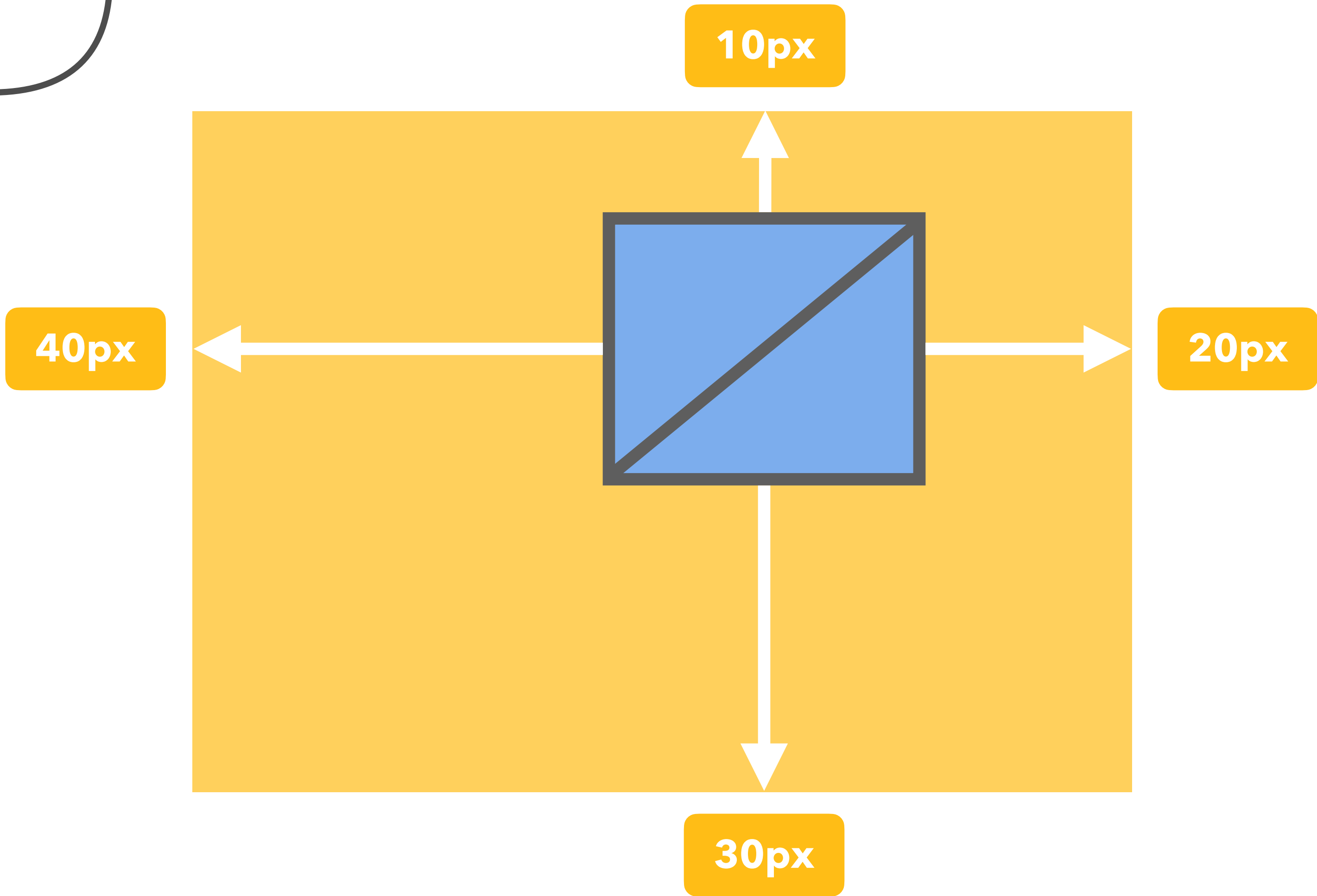
상.중.하



margin: 10px 20px 30px 40px;

top right bottom left

상.시계 방향



margin: top, right, bottom, left ;

margin: top, bottom left, right ;

margin: top left, right bottom ;

margin: top right bottom left ;

요소의 외부 여백(공간)을 지정하는 기타 개별 속성들

# margin-방향

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



요소의 내부 여백(공간)을 지정하는 단축 속성

# padding

요소의 크기가 커져요!



0

내부 여백 없음

단위

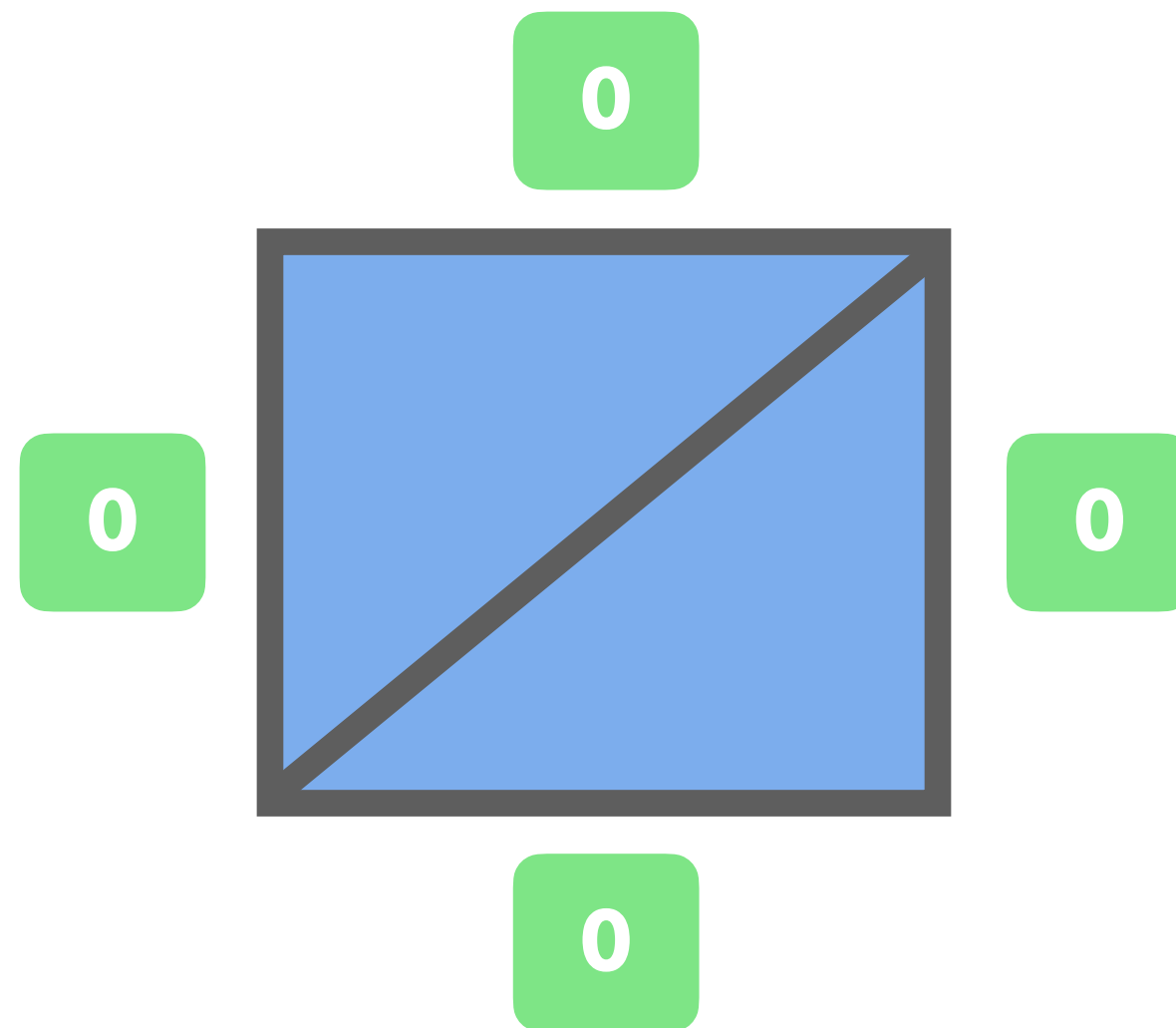
px, em, vw 등 단위로 지정

%

부모 요소의 가로 너비에 대한 비율로 지정

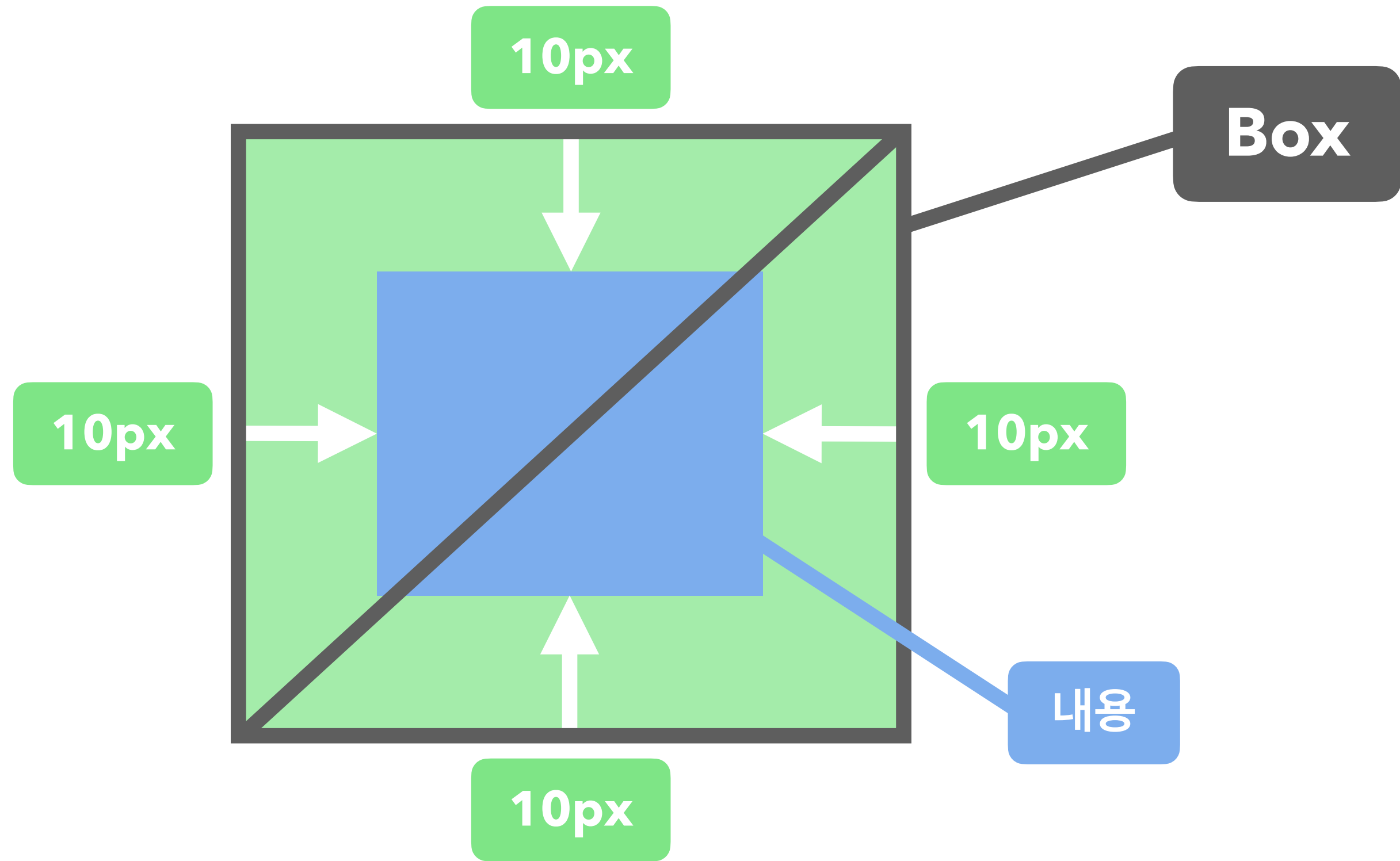
padding: 0;

top, right, bottom, left



padding: 10px;

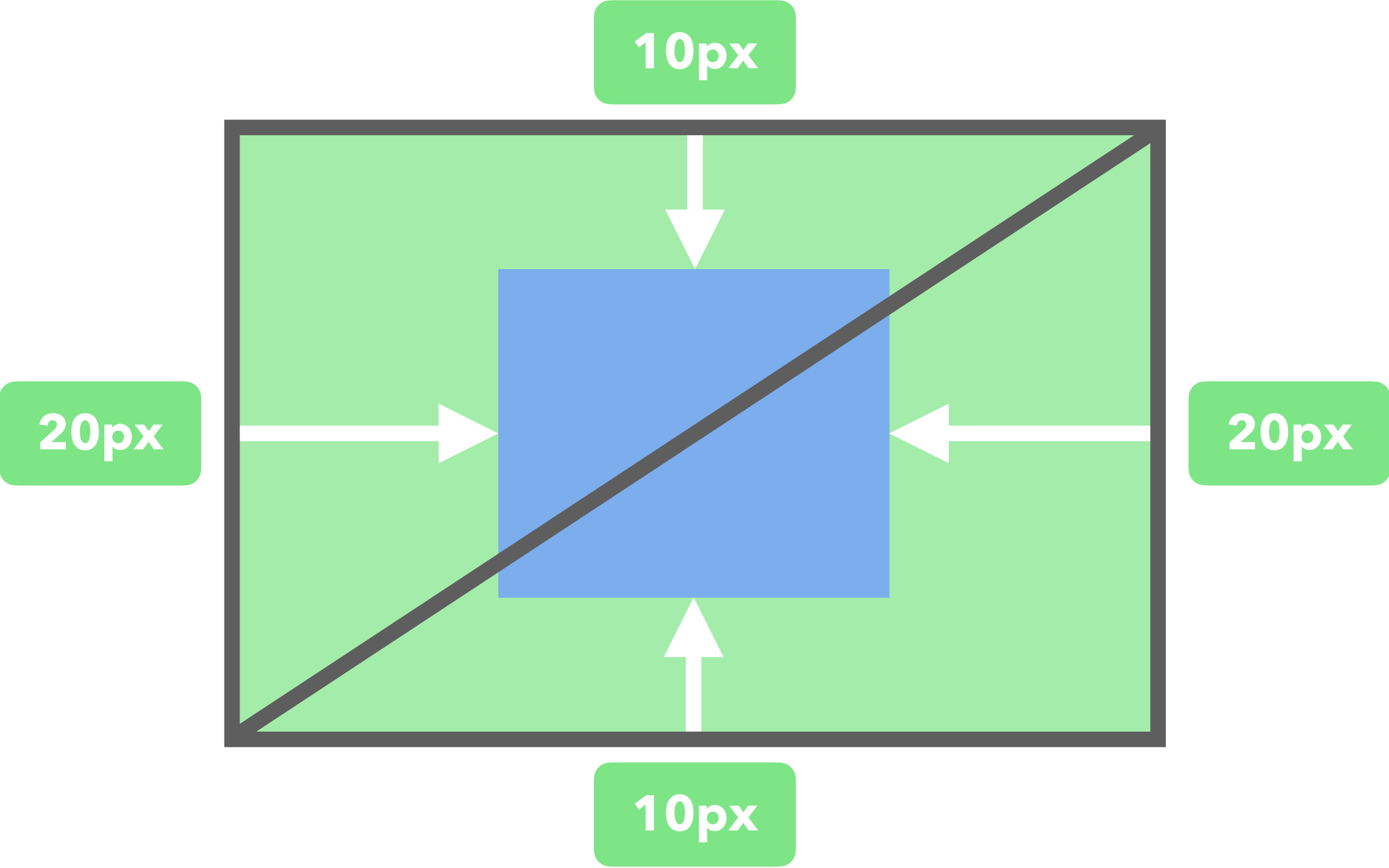
top, right, bottom, left



padding: 10px 20px;

top, bottom

left, right

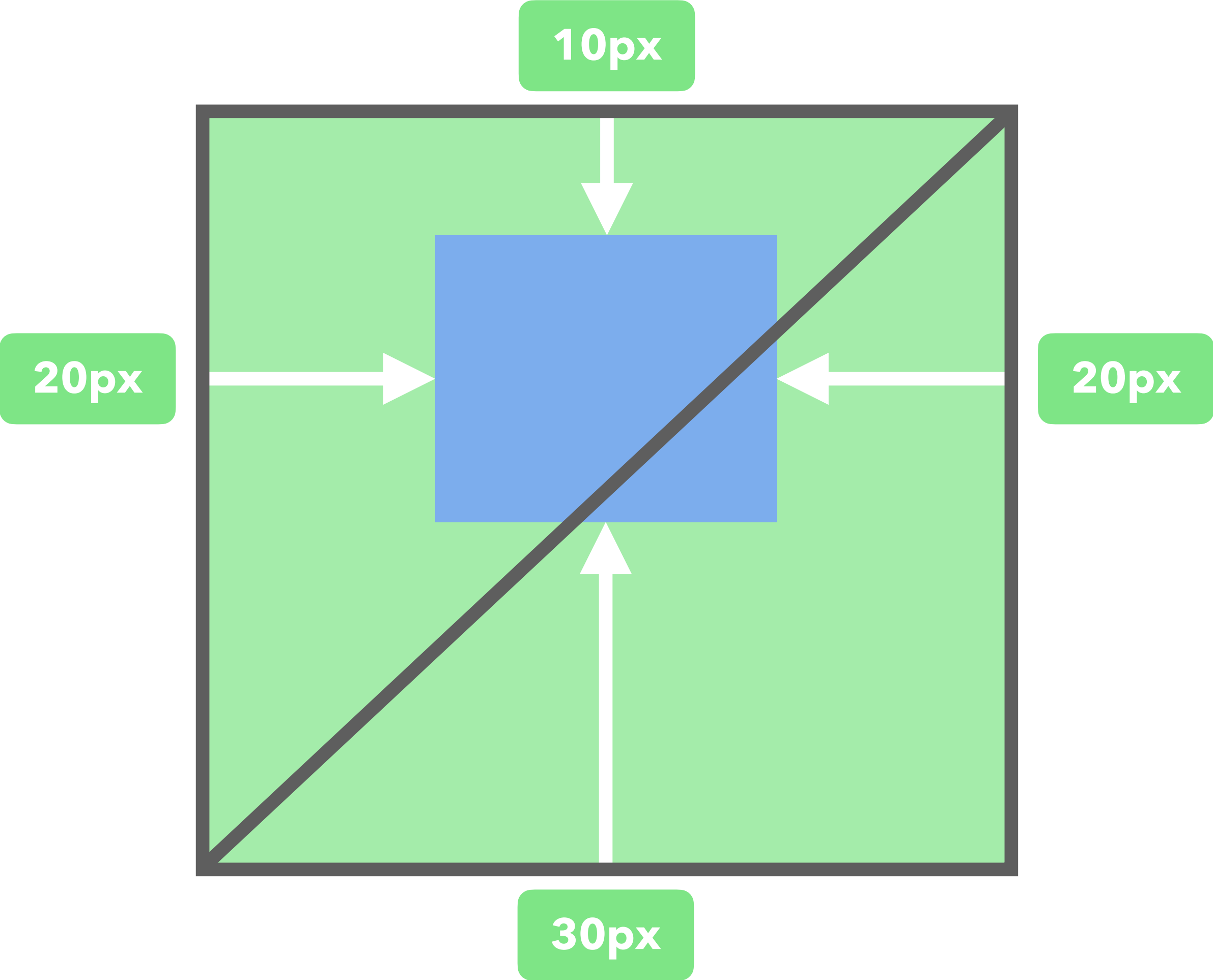


padding: 10px 20px 30px;

top

left, right

bottom



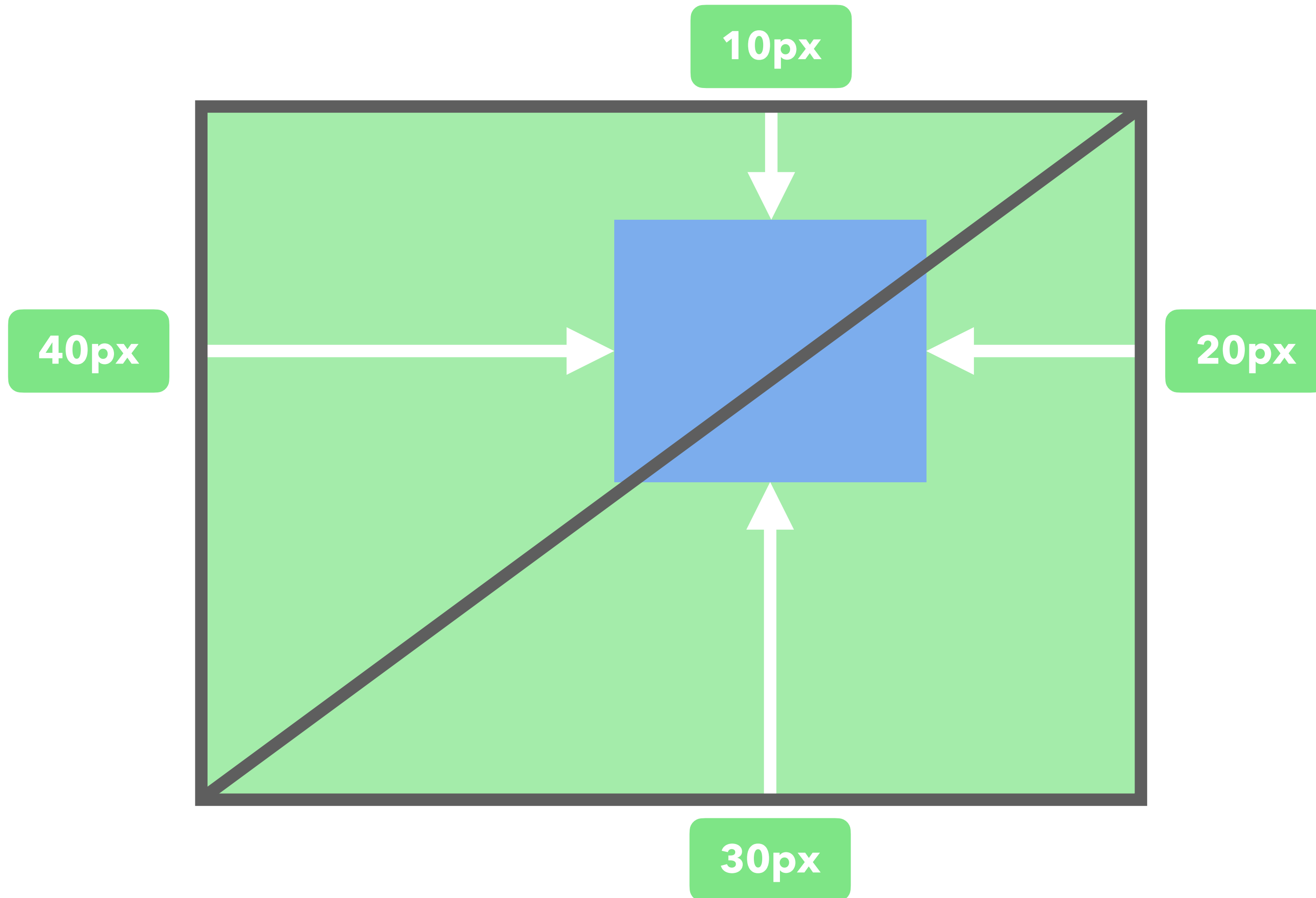
padding: 10px 20px 30px 40px;

top

right

bottom

left



padding: top, right, bottom, left ;

padding: top, bottom left, right ;

padding: top left, right bottom ;

padding: top right bottom left ;

요소의 내부 여백(공간)을 지정하는 기타 개별 속성들

# padding-방향



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

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;

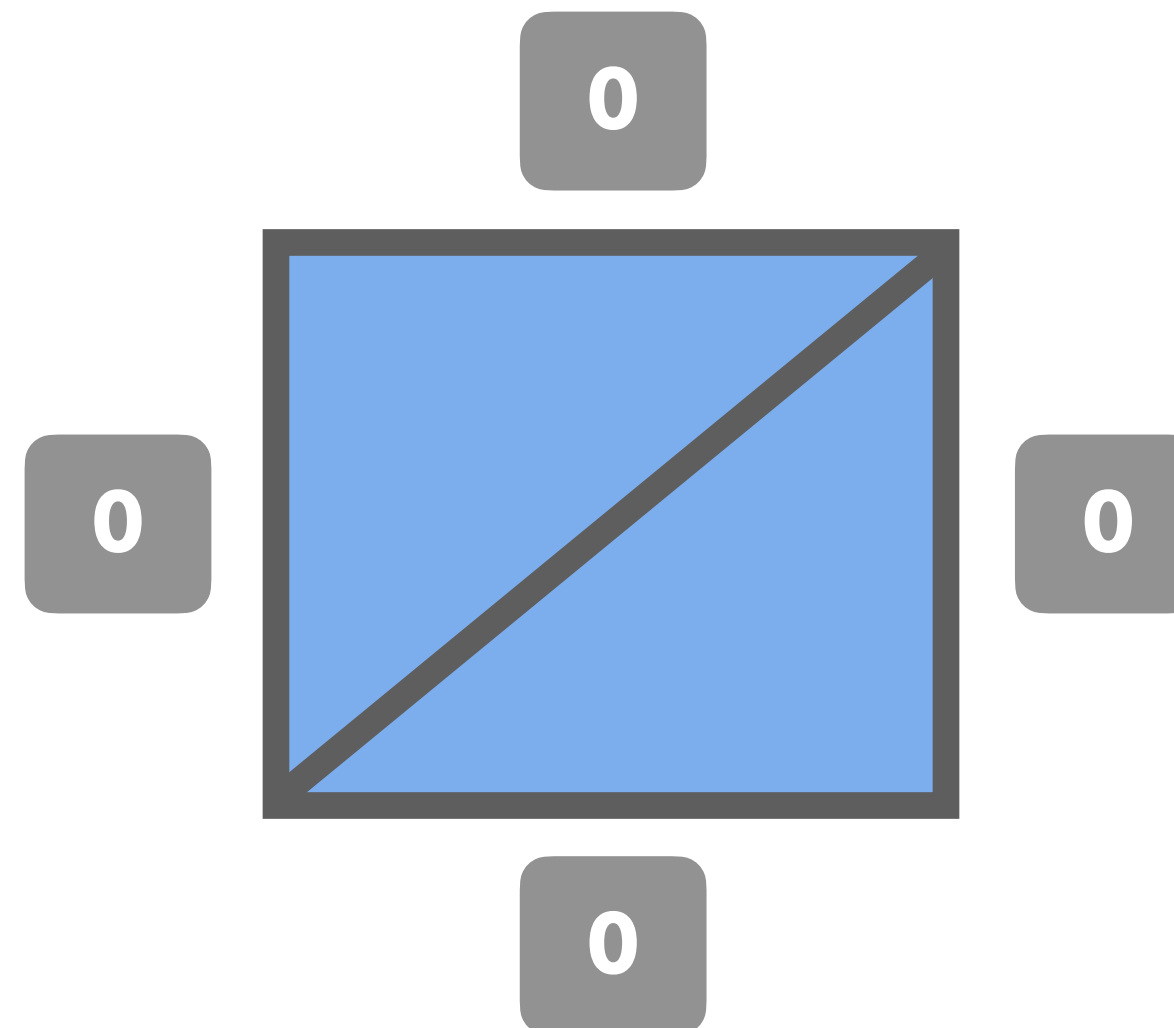
border-width

border-style

border-color

기본값  
(요소에 이미 들어있는 속성의 값)

선의 종류가 없어서(none)  
출력되지 않아요!

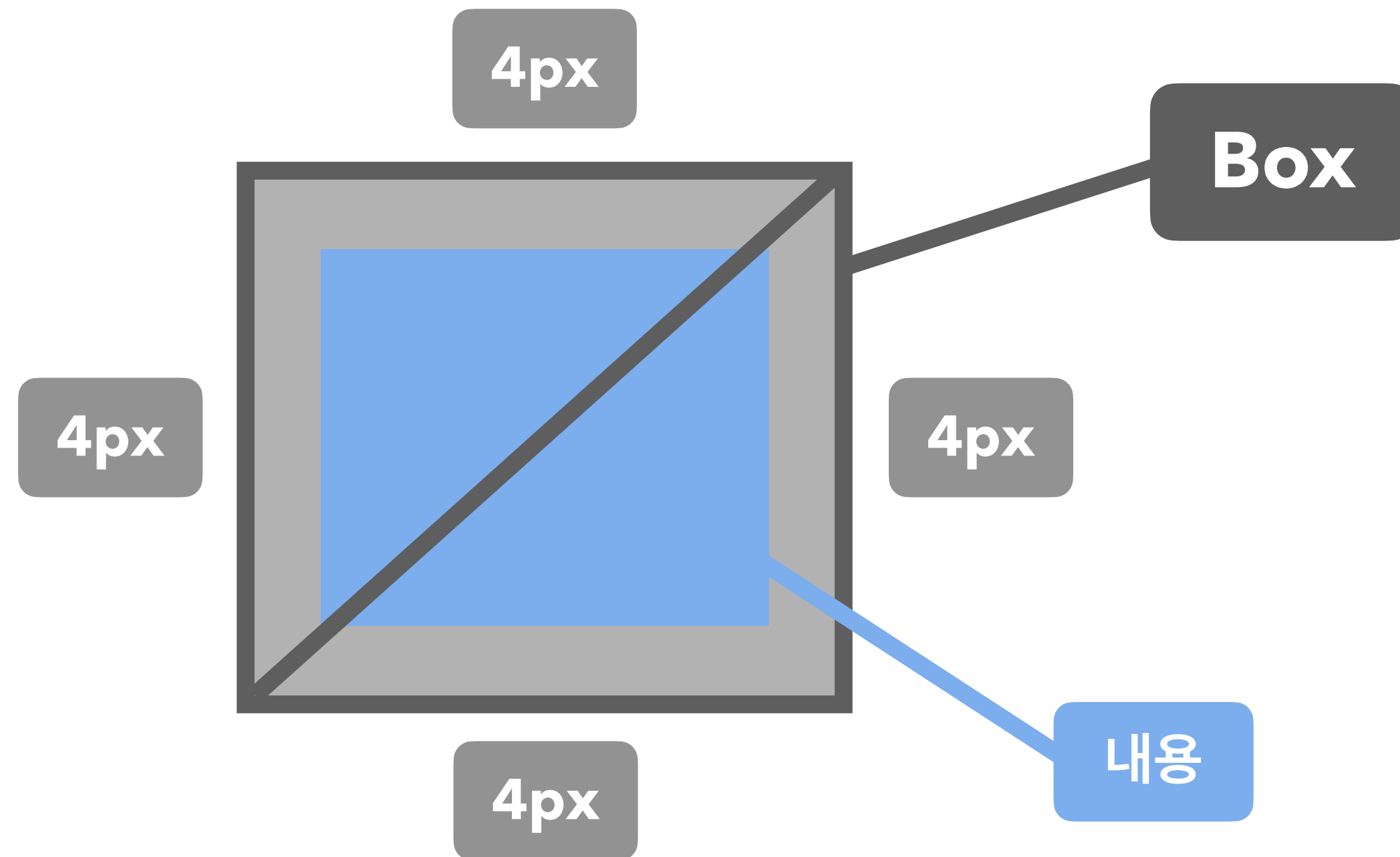


**border:** 4px solid ■ black;

border-width

border-style

border-color



**border:** 10px solid ■ black;

border-width

border-style

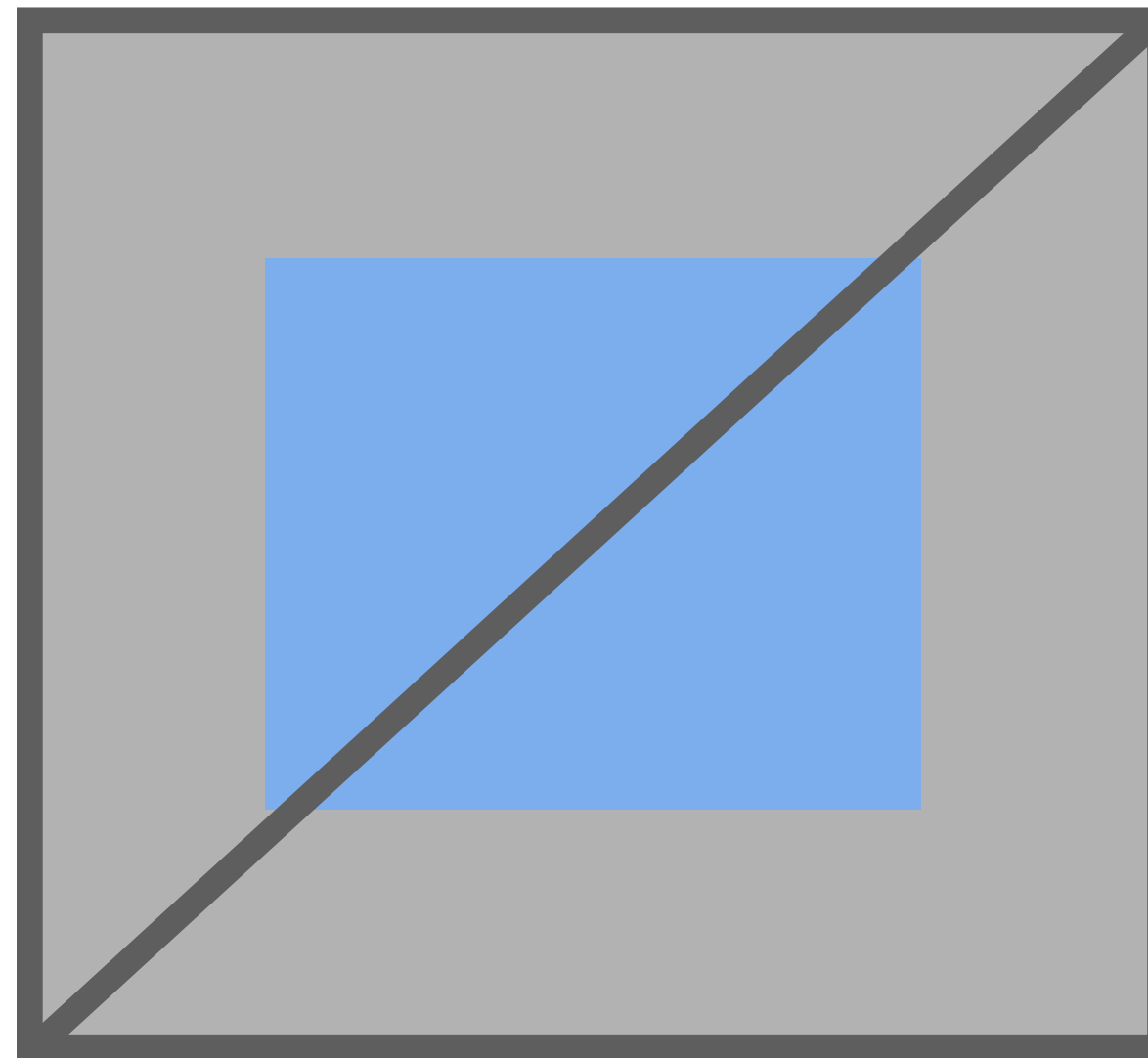
border-color

10px

10px

10px

10px



요소 테두리 선의 두께

# border-width

medium

중간 두께

thin

얇은 두께

thick

두꺼운 두께

단위

px, em, % 등 단위로 지정

**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

숫은 모양 (groove의 반대)

inset

요소 전체가 들어간 모양

outset

요소 전체가 나온 모양



**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: top, right, bottom, left ;**

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

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

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

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, #FFFFFFF
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)

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-bottom:** 두께 종류 색상;

**border-bottom-width:** 두께;

**border-bottom-style:** 종류;

**border-bottom-color:** 색상;

**border-left:** 두께 종류 색상;

**border-left-width:** 두께;

**border-left-style:** 종류;

**border-left-color:** 색상;



**border-right:** 두께 종류 색상;

**border-right-width:** 두께;

**border-right-style:** 종류;

**border-right-color:** 색상;

요소의 모서리를 둥글게 깎음

# border-radius

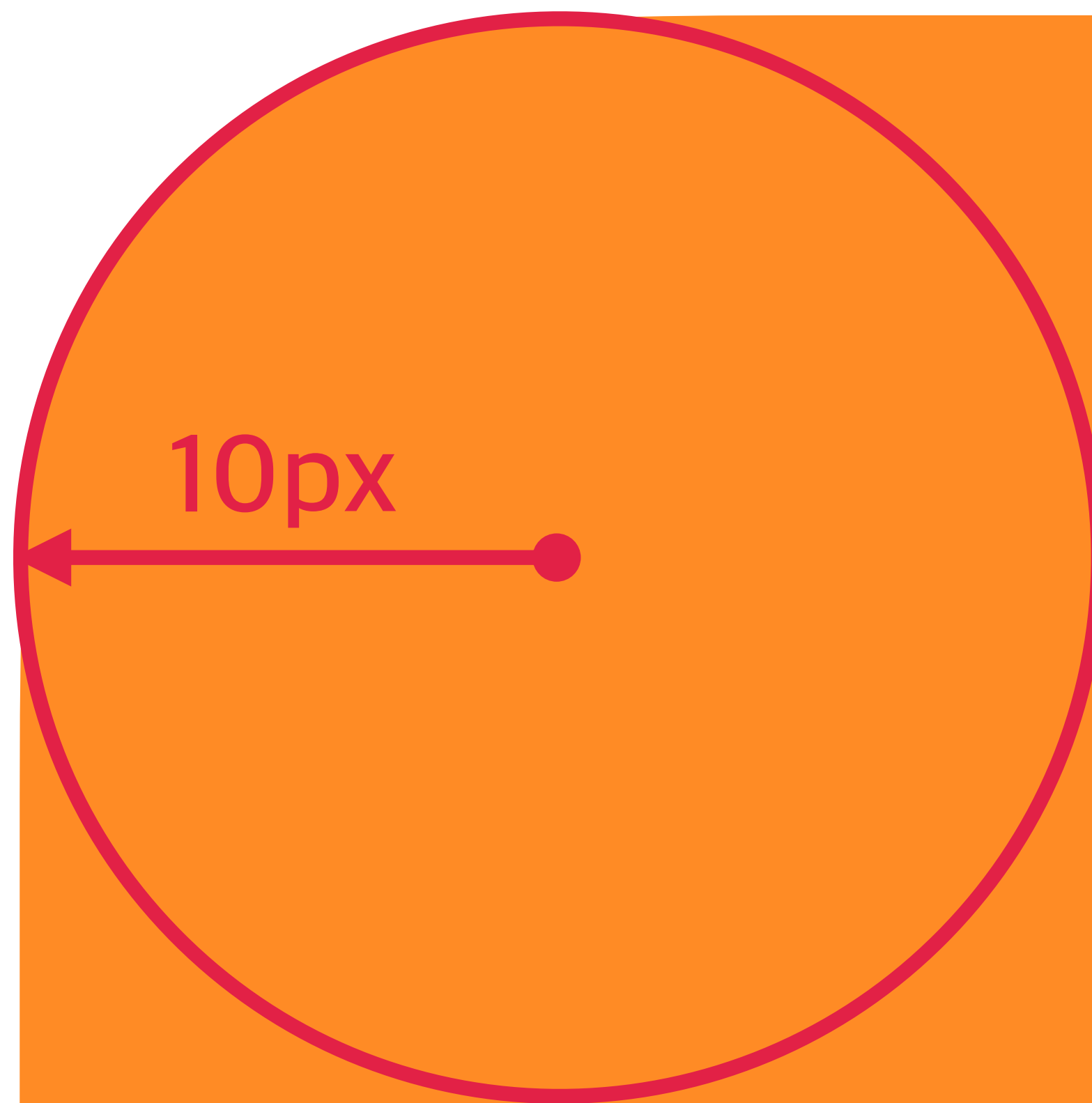
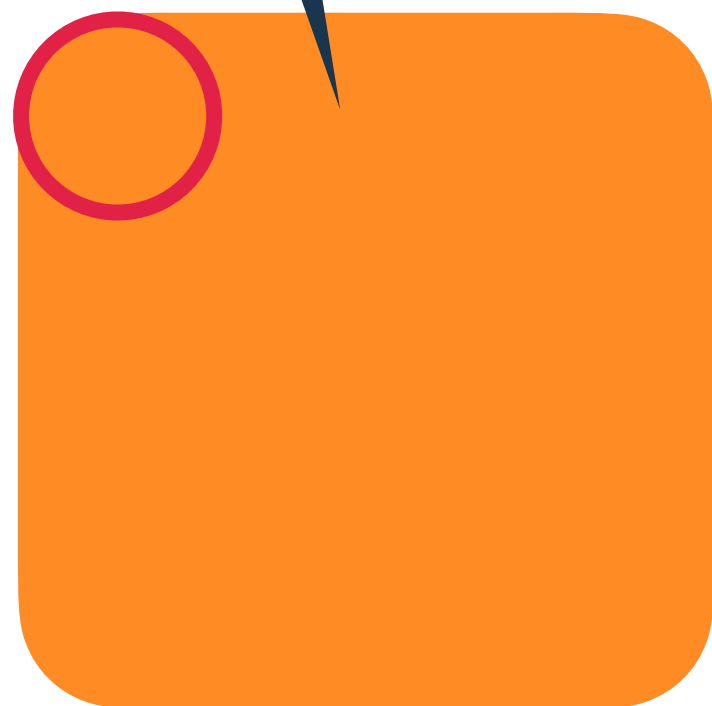
0

둥글게 없음

단위

px, em, vw 등 단위로 지정

border-raidus: 10px;



`border-radius: 0;`

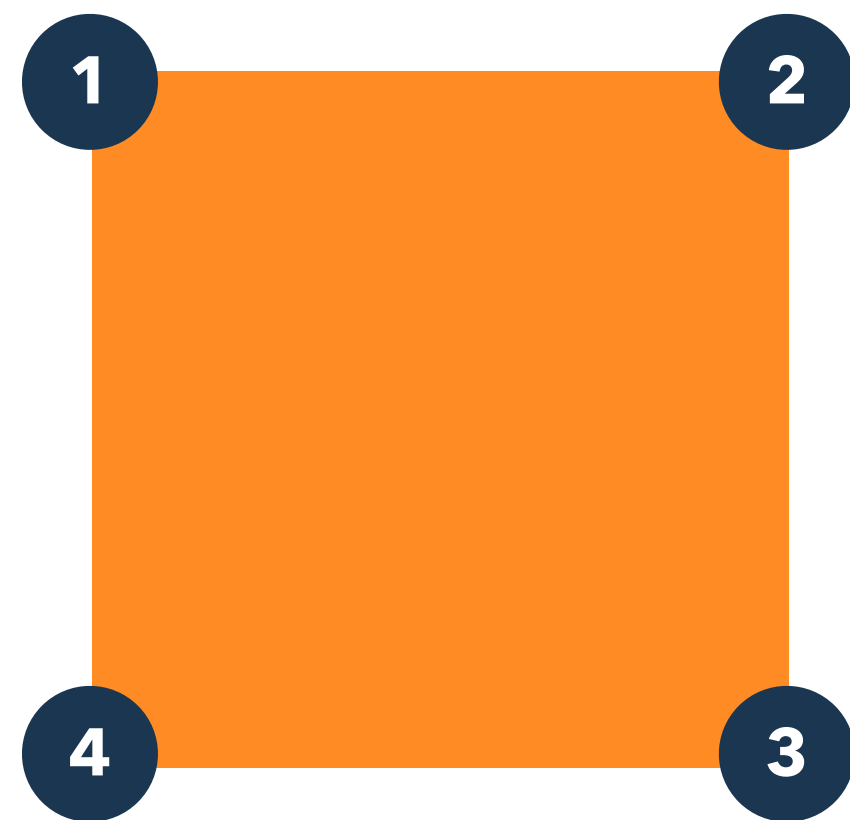


`border-radius: 10px;`



`border-radius: 0 10px 0 0;`





`border-radius: 0 10px 0 0;`

요소의 크기 계산 기준을 지정

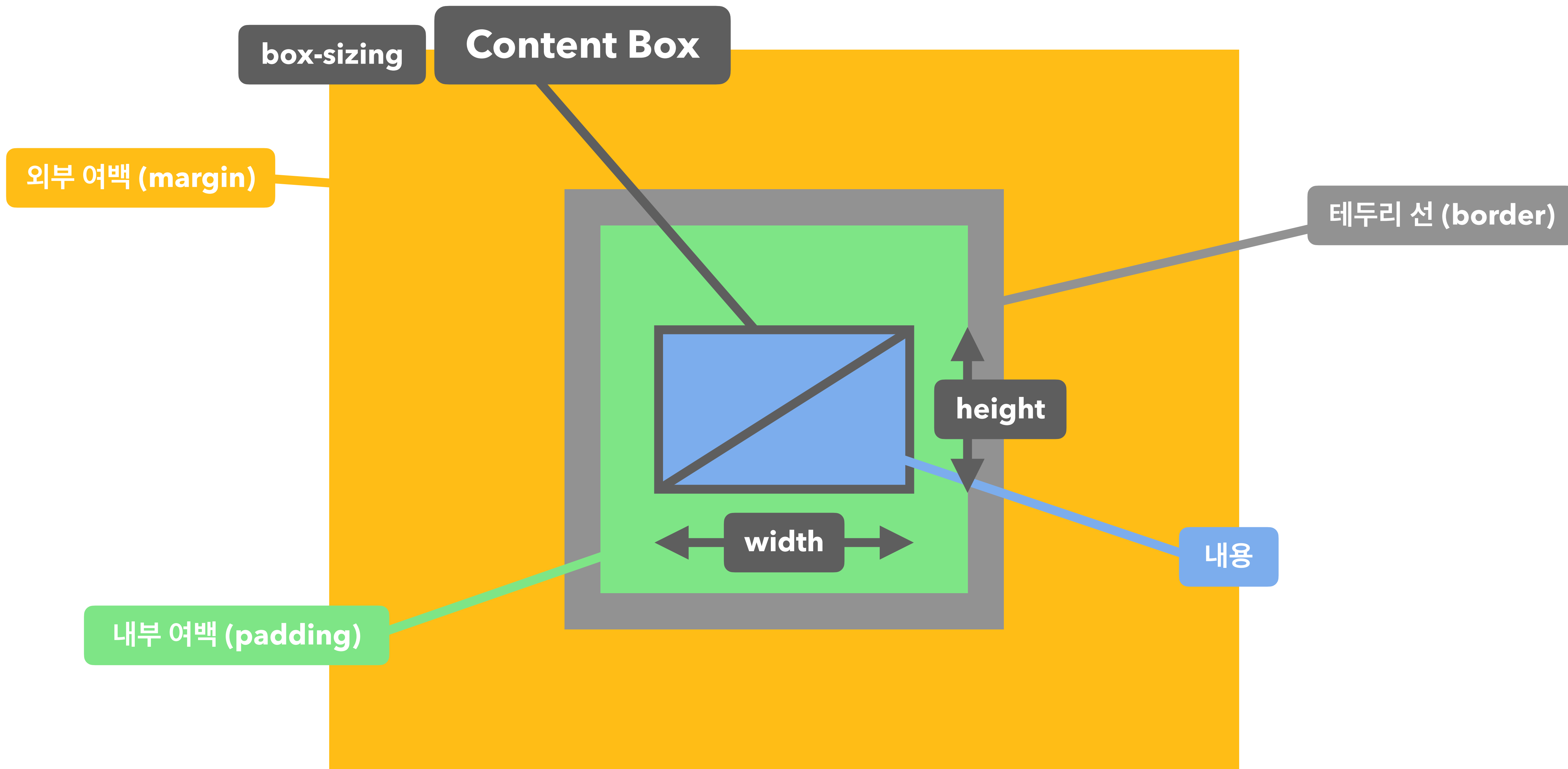
# box-sizing

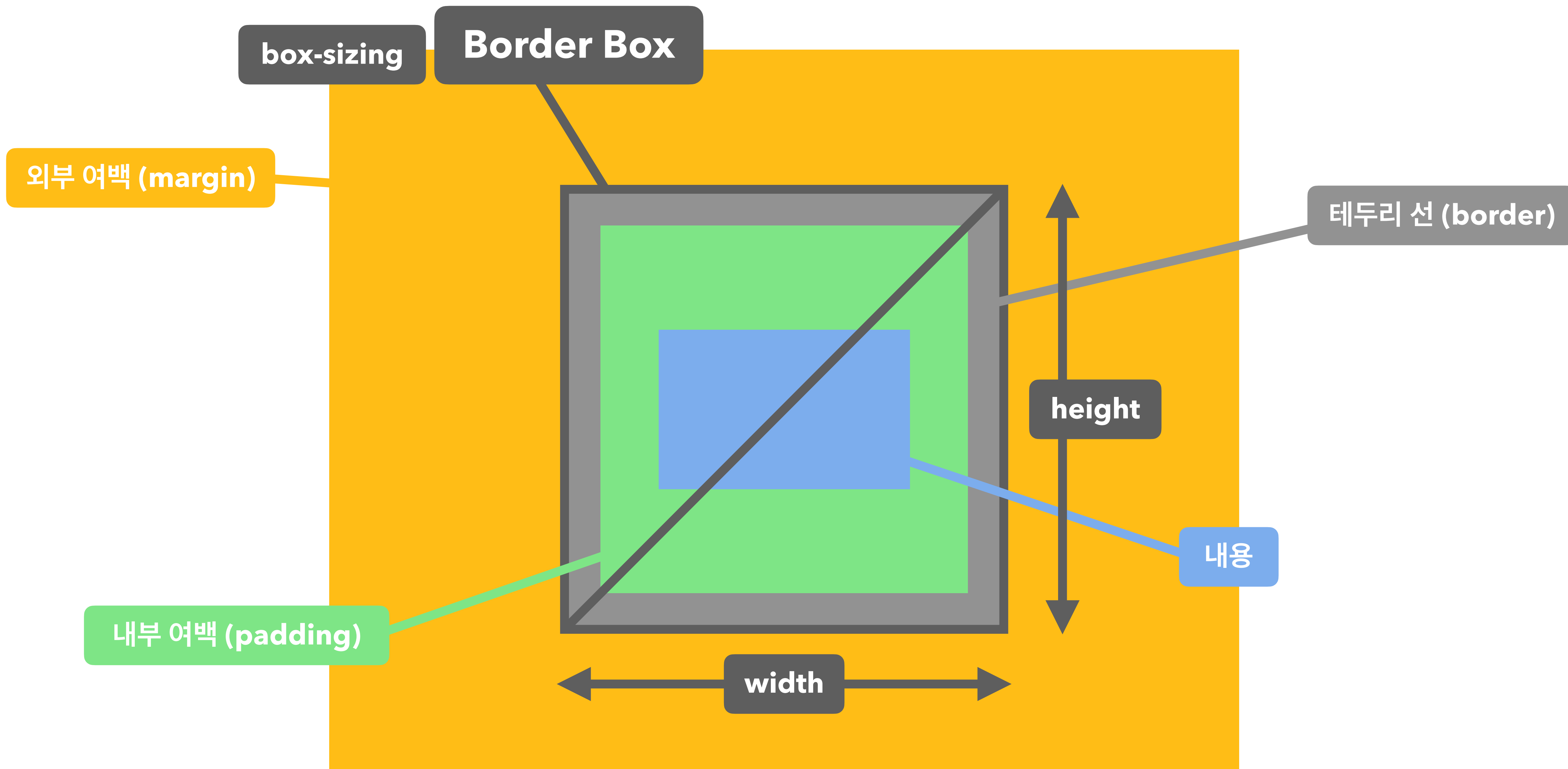
**content-box**

요소의 내용(content)으로 크기 계산

**border-box**

요소의 내용 + padding + border로 크기 계산







요소의 크기 이상으로 내용이 넘쳤을 때, 보여짐을 제어하는 단축 속성

# overflow

**visible**

넘친 내용을 그대로 보여줌

**hidden**

넘친 내용을 잘라냄

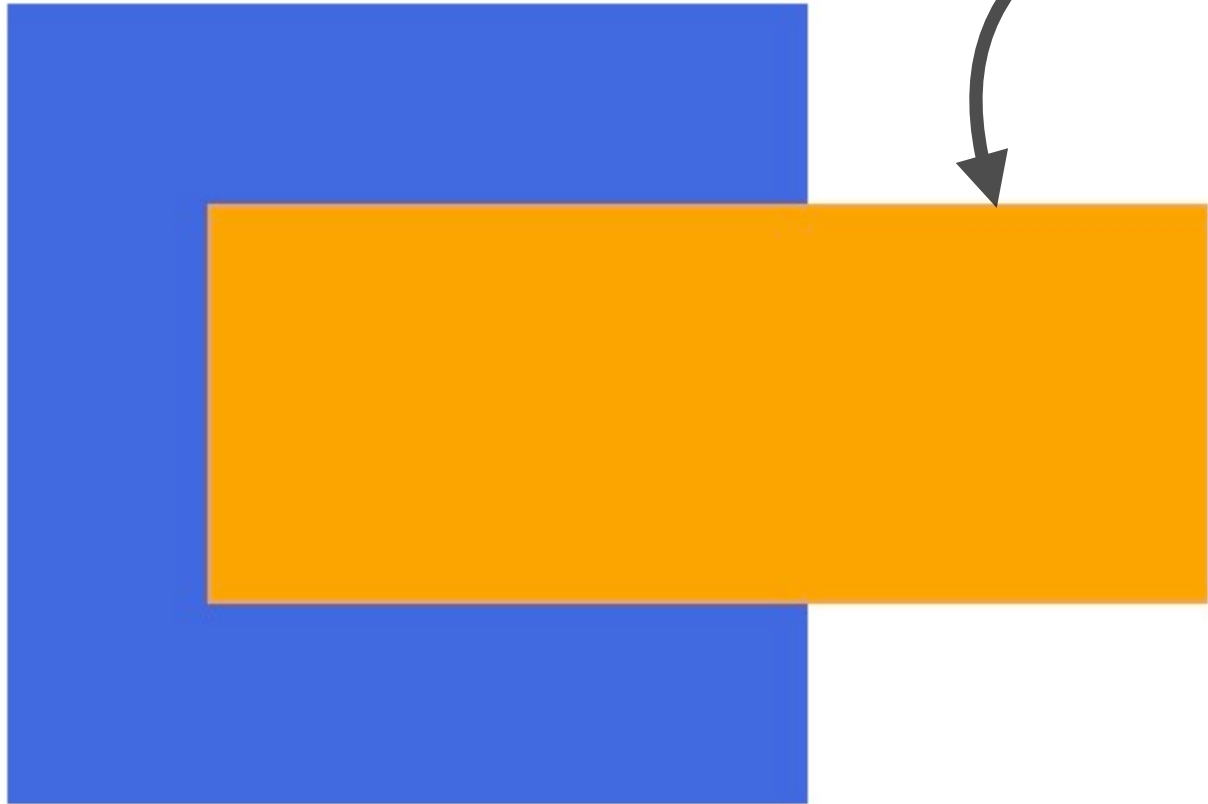
**scroll**

넘친 내용을 잘라냄, 스크롤바 생성

**auto**

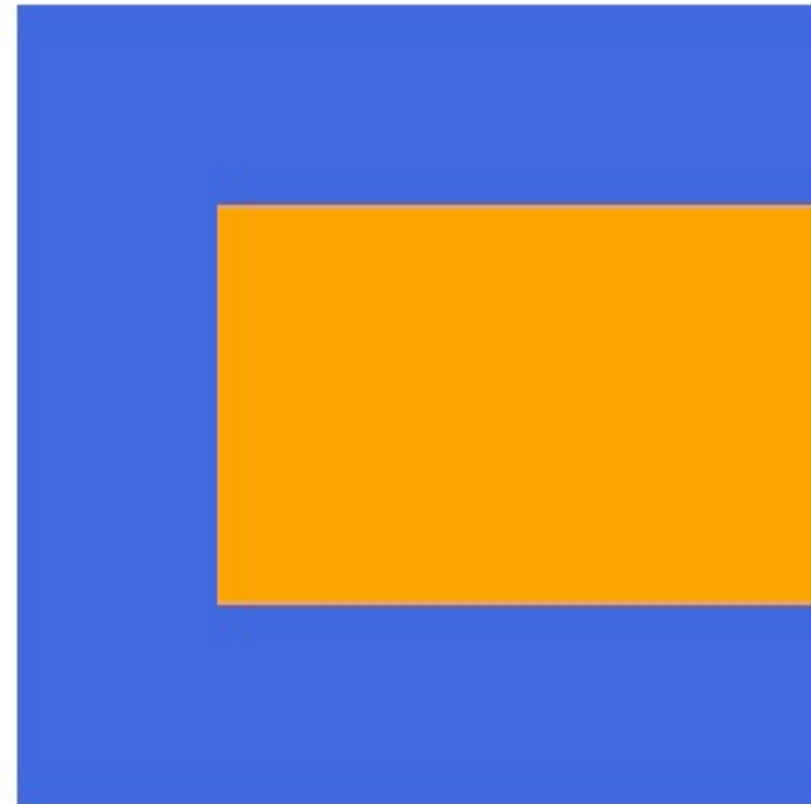
넘친 내용이 있는 경우에만 잘라내고 스크롤바 생성

visible



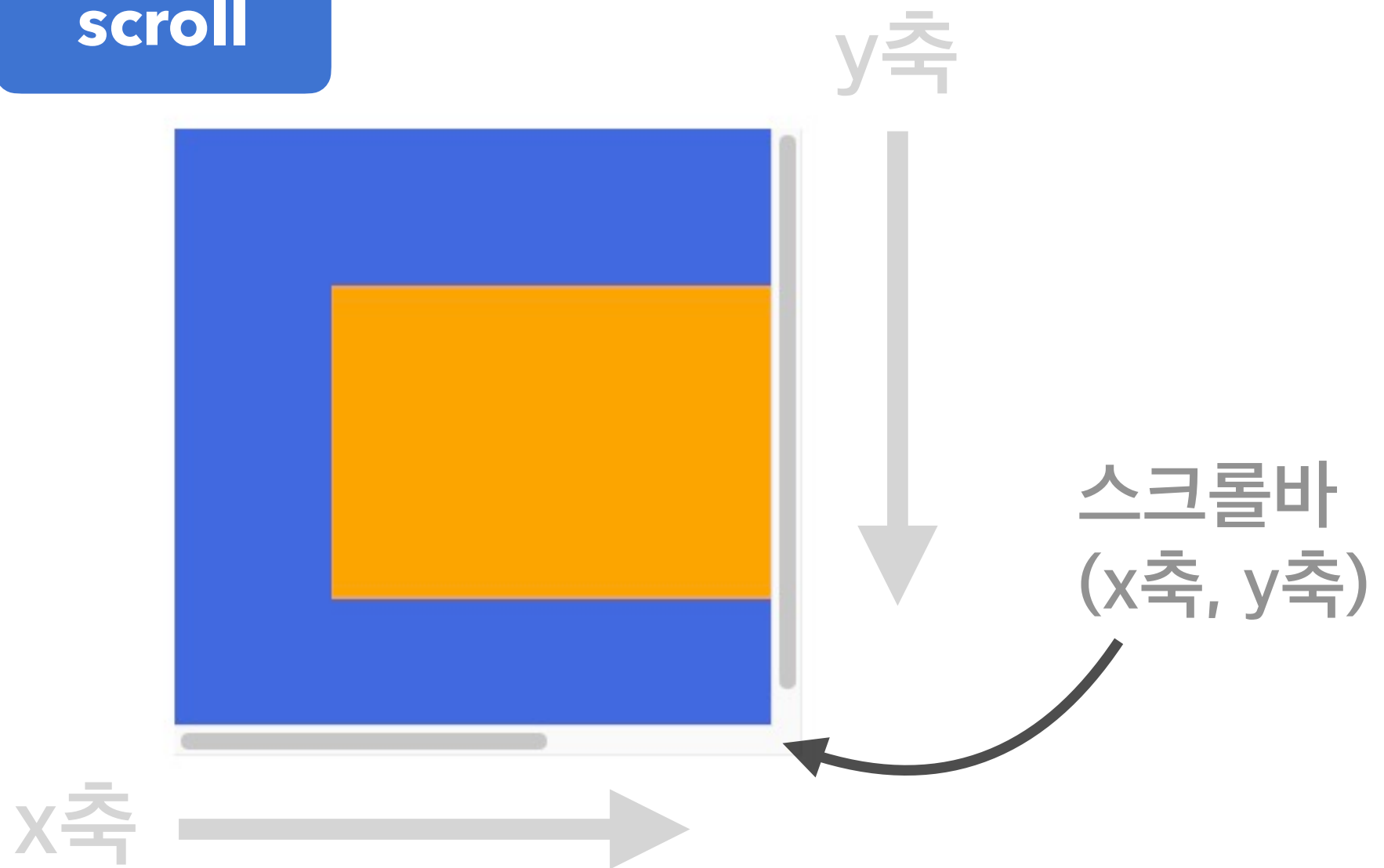
내용 넘침

hidden



잘라냄

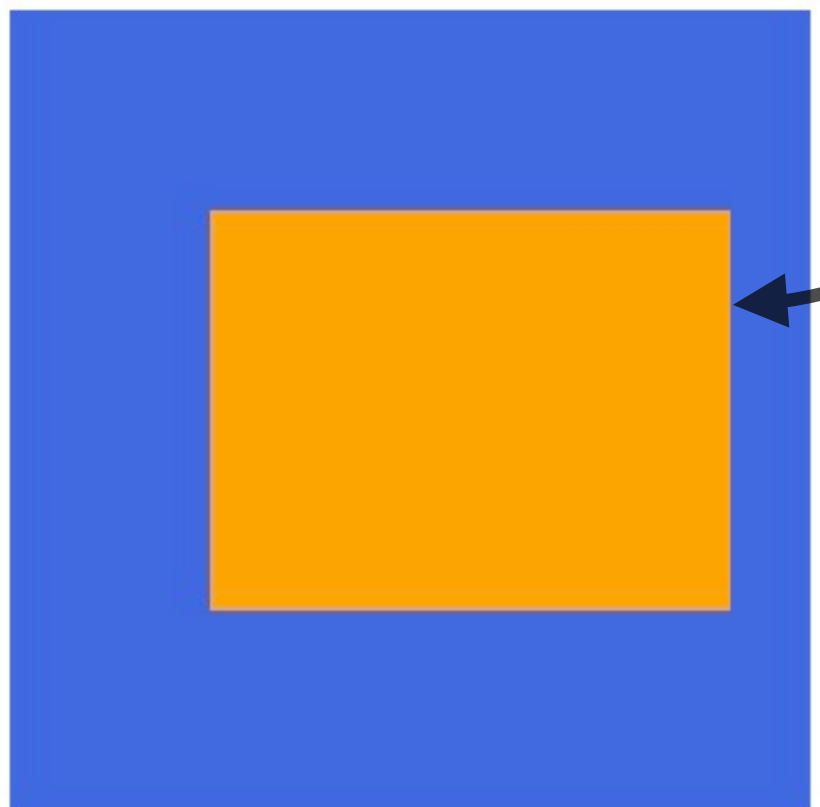
scroll



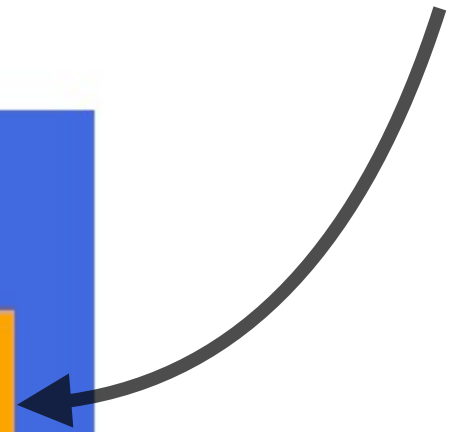
auto



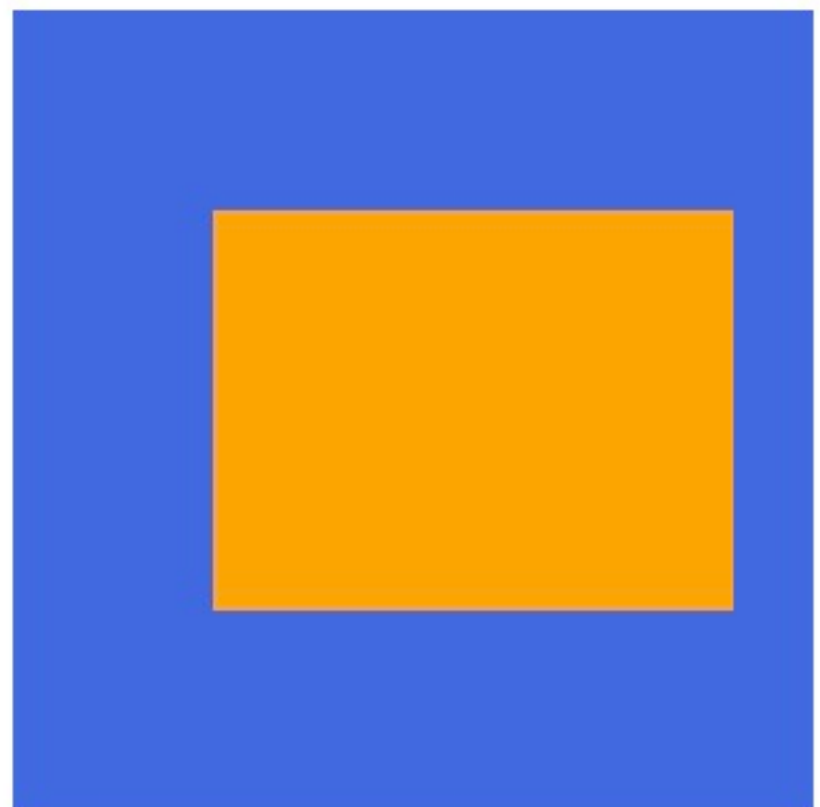
visible



내용 넘치지 않음!



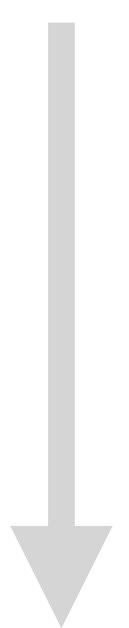
hidden



scroll



y축



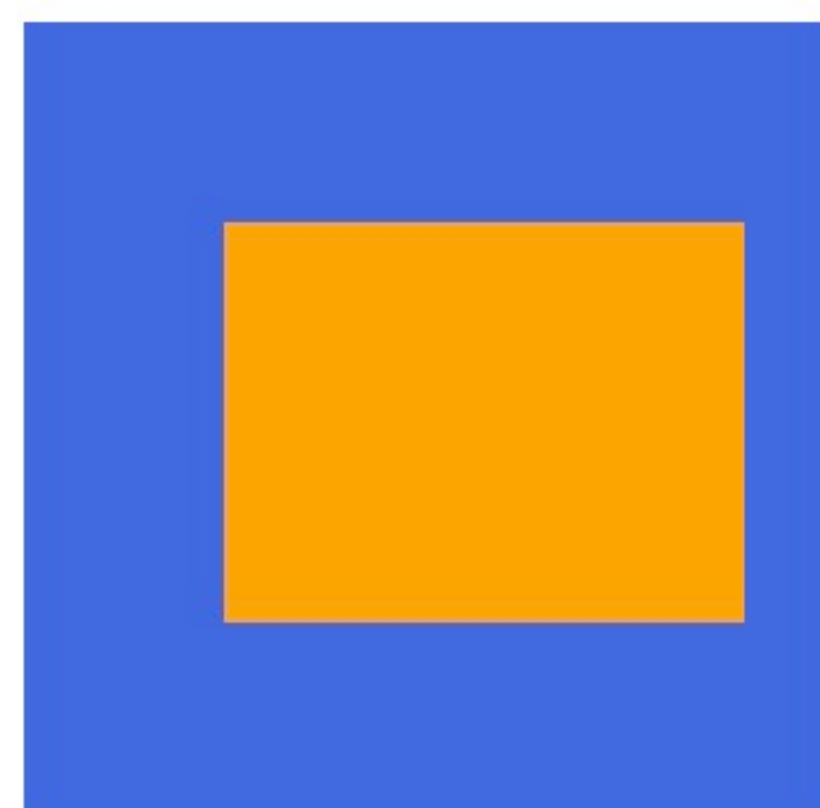
스크롤바  
(x축, y축)



x축



auto



요소의 크기 이상으로 내용이 넘쳤을 때, 보여짐을 제어하는 개별 속성들

**overflow-x**

**overflow-y**

## 요소의 화면 출력(보여짐) 특성

# display

각 요소에 이미 지정되어 있는 값

**block**

상자(레이아웃) 요소

**inline**

글자 요소

**inline-block**

글자 + 상자 요소

**flex**

플렉스 박스 (1차원 레이아웃)

**grid**

그리드 (2차원 레이아웃)

**none**

보여짐 특성 없음, 화면에서 사라짐

**기타**

table, table-row, table-cell 등..

따로 지정해서 사용하는 값

요소 투명도

# opacity

1

불투명

0~1

0부터 1 사이의 소수점 숫자

opacity: 0.07;



opacity: 0.4;



opacity: 0.7;



opacity: 1;



글꼴



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

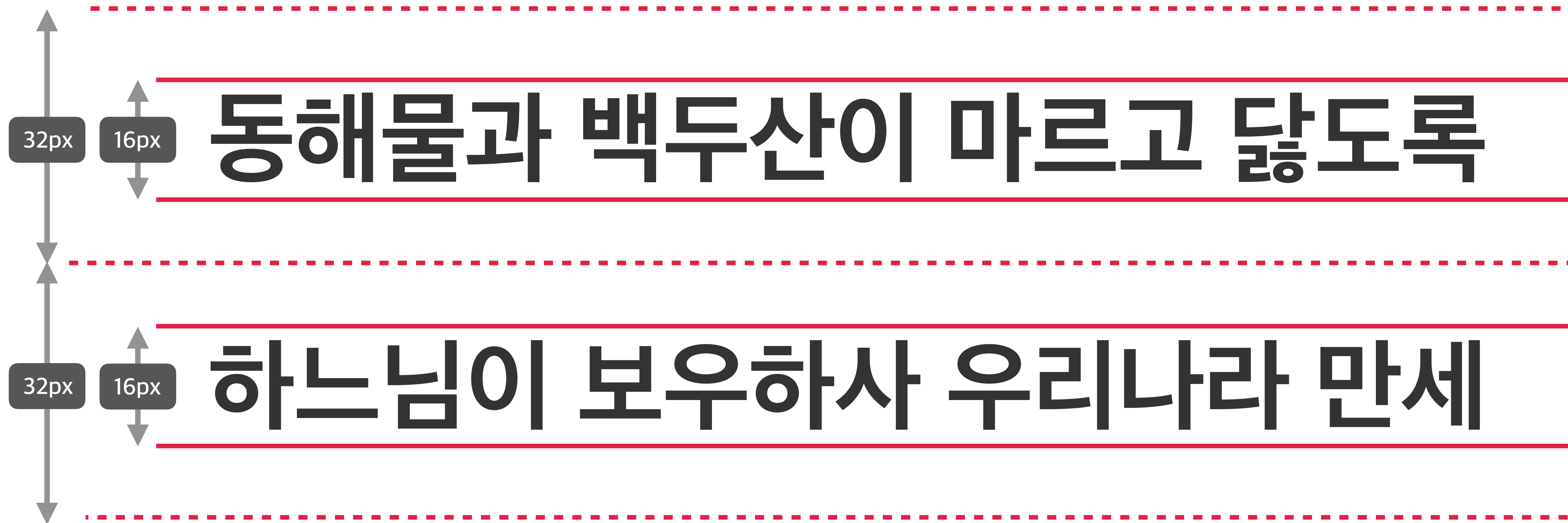
%

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%; */
```



```
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

He l l o Wo r l d !

monospace

고정너비(가로폭이 동등) 글꼴 계열

*Hello World!*

cursive

필기체 계열

**EMPIRE**

fantasy

장식 글꼴 계열



문자

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 하느  
님이 보우하사 우리나라 만세 무궁화 삼천리 화  
려 강산 대한 사람 대한으로 길이 보전하세

들여쓰기(50px)

문자 첫 줄의 들여쓰기

음수를 사용할 수 있어요!  
반대는 내어쓰기(outdent)입니다.

# 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

background-color: orange;





요소의 배경 이미지 삽입

# background-image

**none**

이미지 없음

**url("경로")**

이미지 경로

```
background-color: orange;  
background-image: url("../images/heropy.png");
```



배경 색상은  
이미지 뒤에 나와요!



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

```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: repeat-x;
```



```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: repeat-y;
```

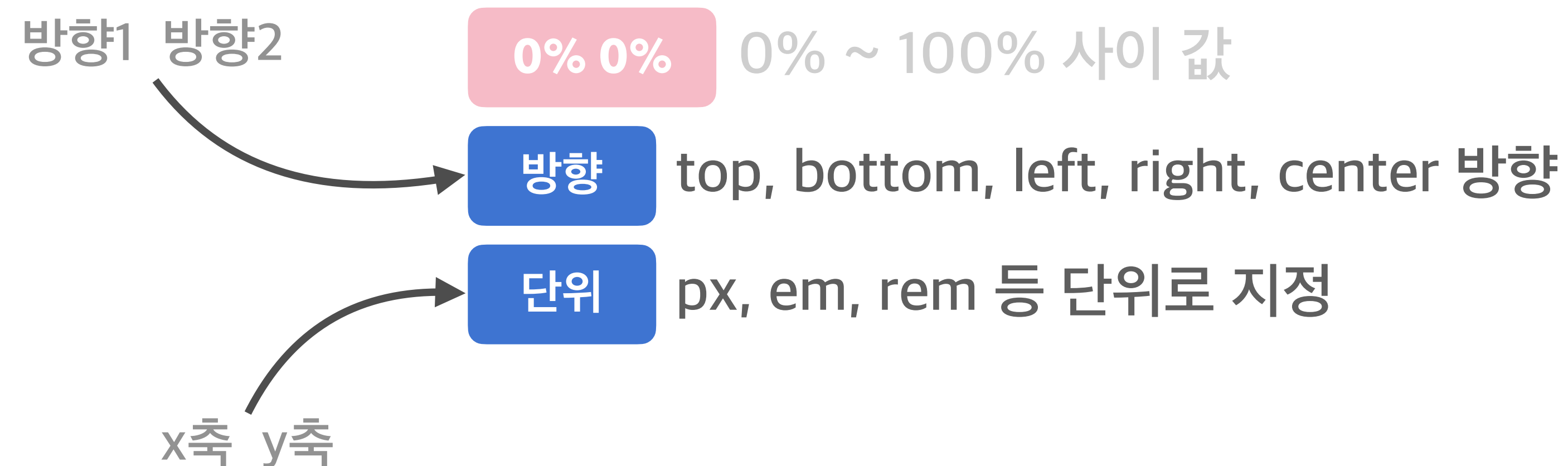


```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;
```



요소의 배경 이미지 위치

# background-position





```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-position: top right;
```





```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-position: center;
```



```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-position: 100px 30px;
```

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-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-size: cover;
```



```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-size: contain;
```



요소의 배경 이미지 스크롤 특성

# background-attachment

**scroll**

이미지가 요소를 따라서 같이 스크롤

**fixed**

이미지가 뷰포트에 고정, 스크롤 X

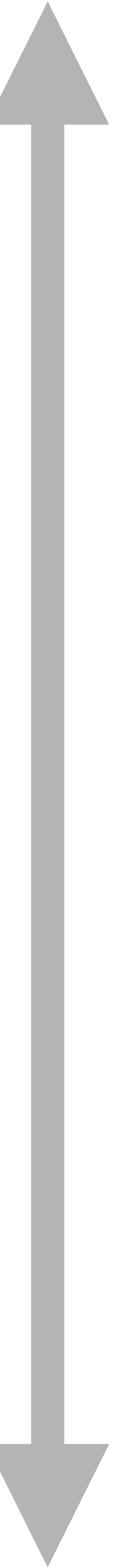
**local**

요소 내 스크롤 시 이미지가 같이 스크롤

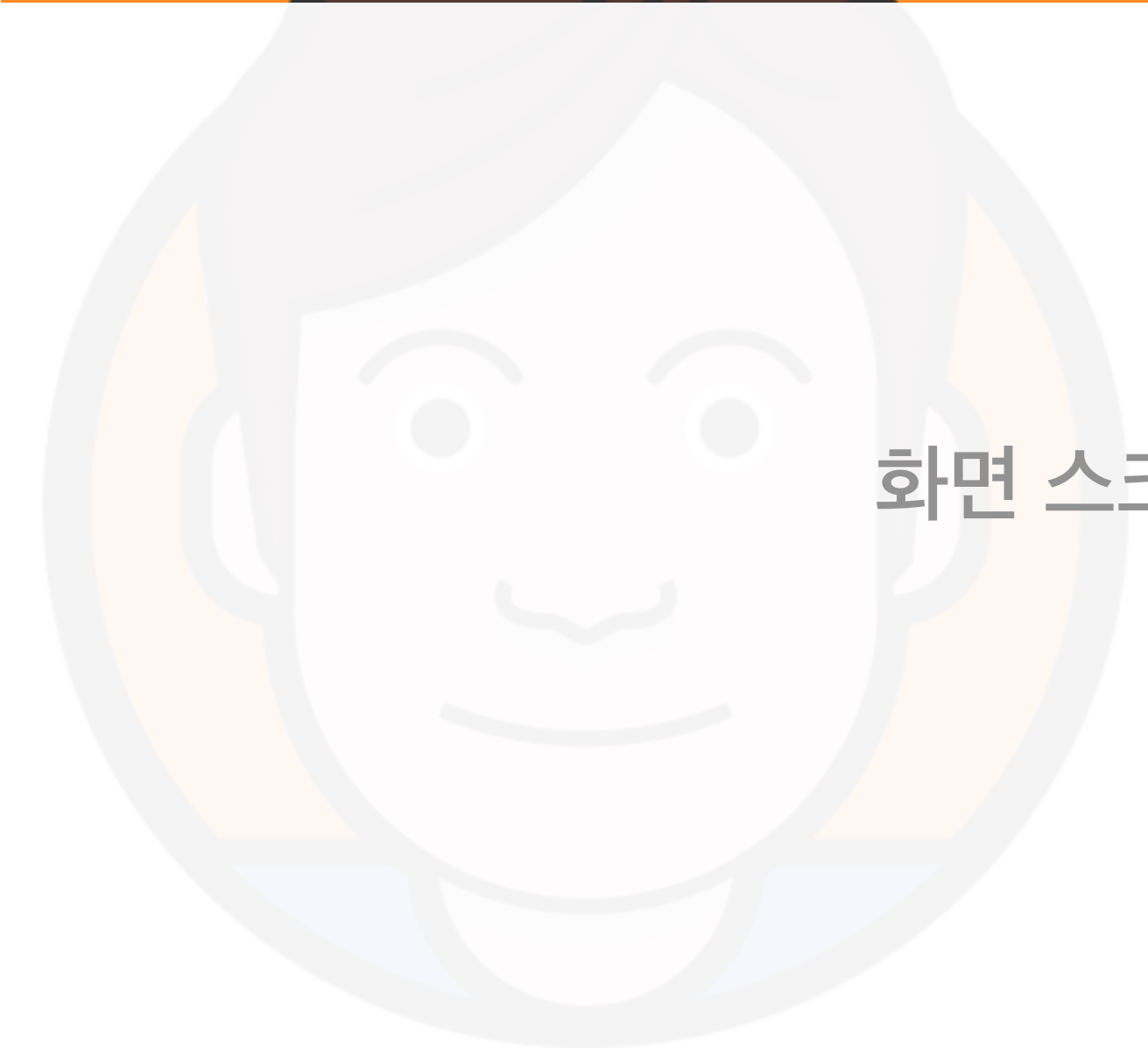
```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-size: cover;  
background-attachment: scroll;
```



화면 스크롤



```
background-color: orange;  
background-image: url("../images/heropy.png");  
background-repeat: no-repeat;  
background-size: cover;  
background-attachment: fixed;
```



화면 스크롤



배치



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

top  
bottom  
left  
right  
z-index

요소의 위치 지정 기준

# position

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

위치 상 부모 요소를  
꼭 확인해야 해요!

요소의 각 방향별 거리 지정

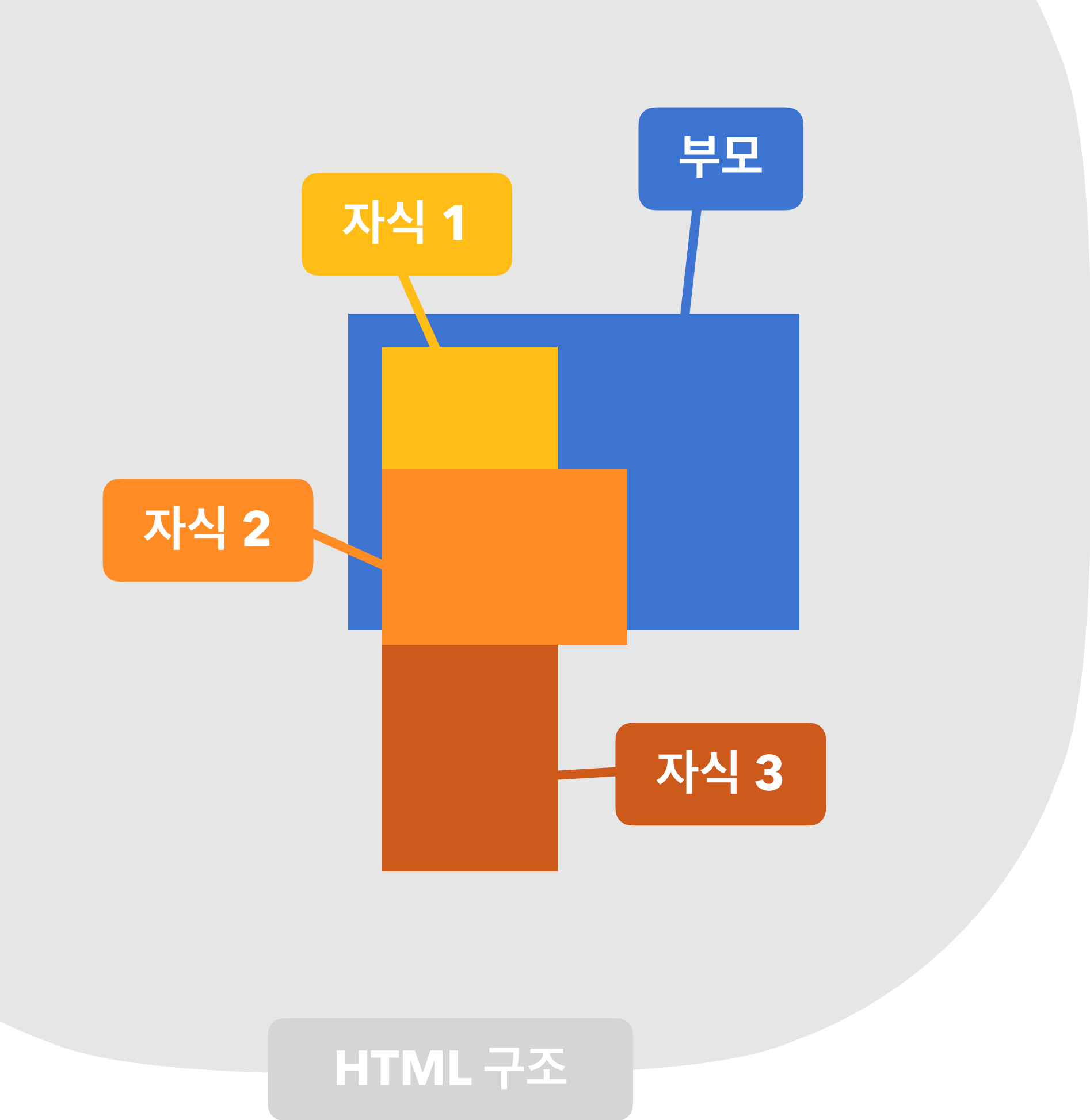
# top, bottom, left, right

**auto**

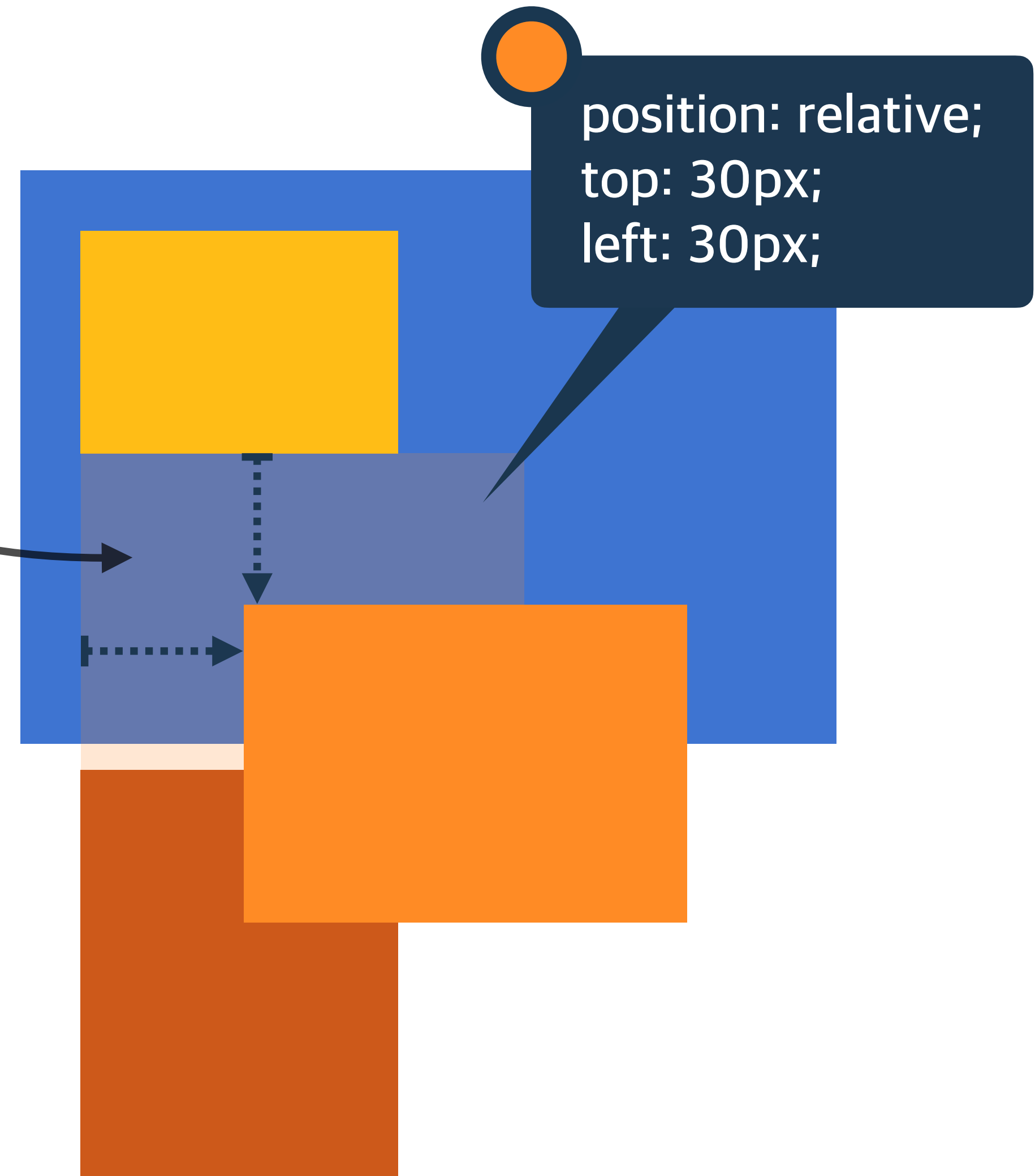
브라우저가 계산

**단위**

px, em, rem 등 단위로 지정

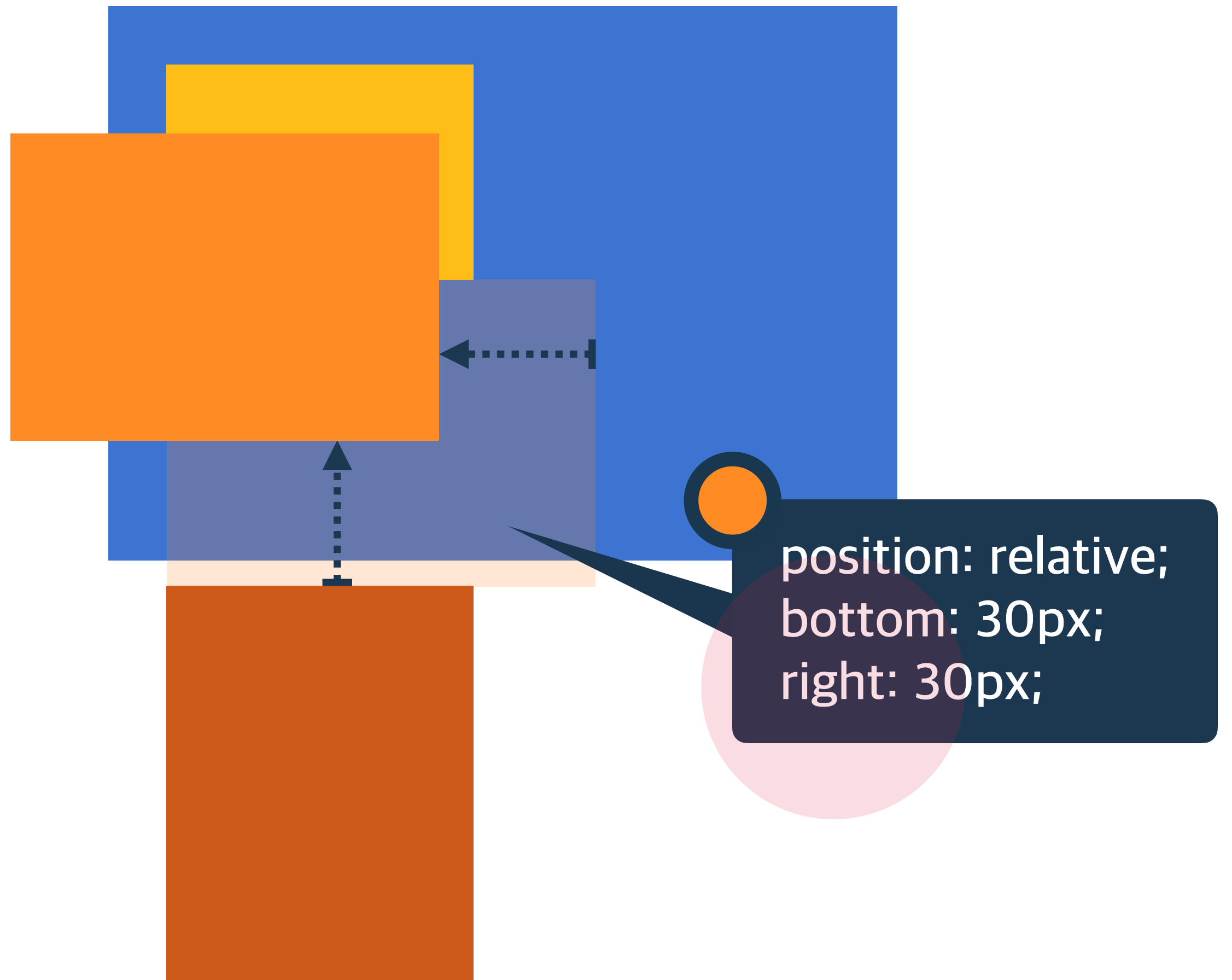
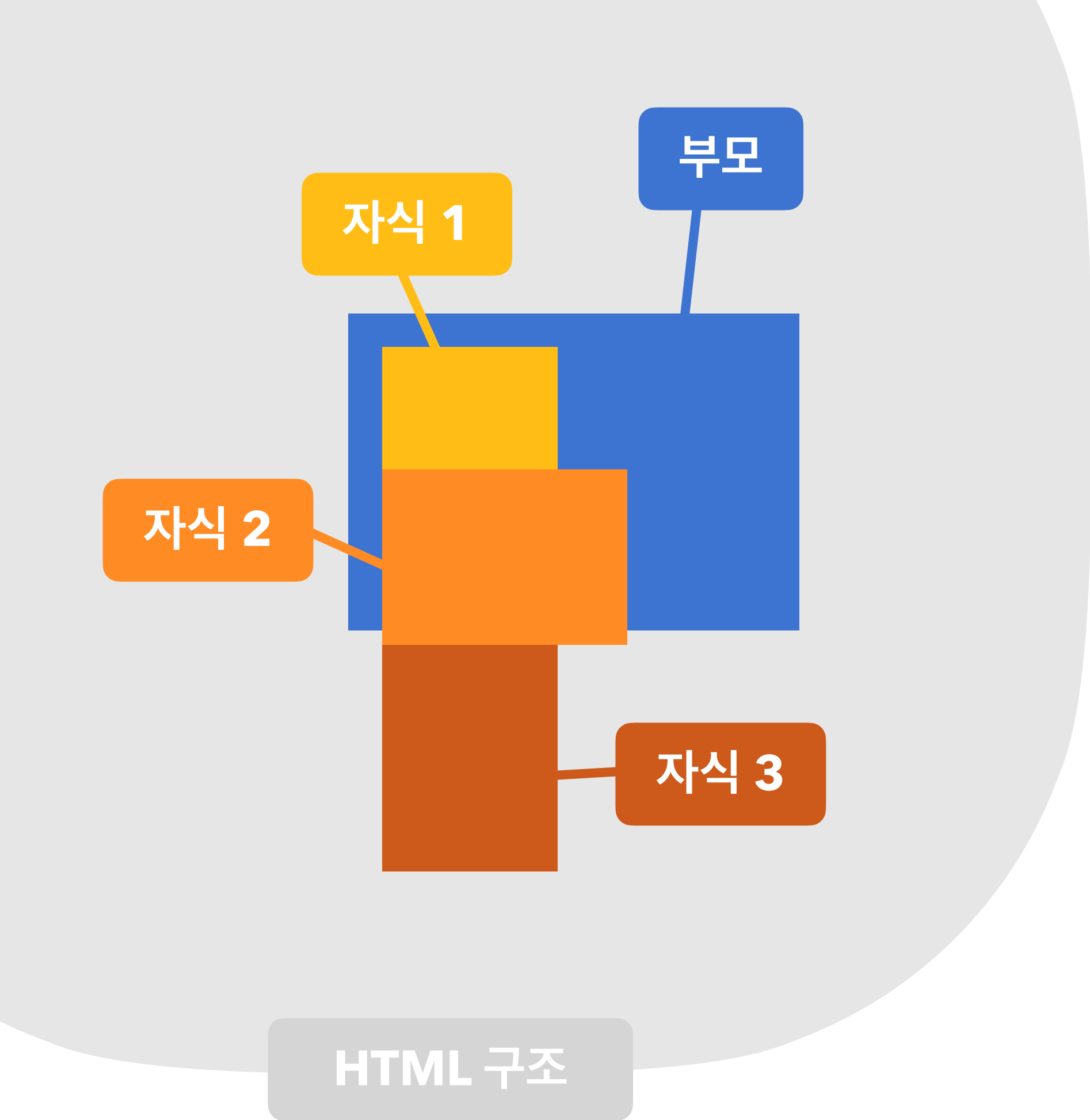


배치 전 자리는  
비어 있어요!

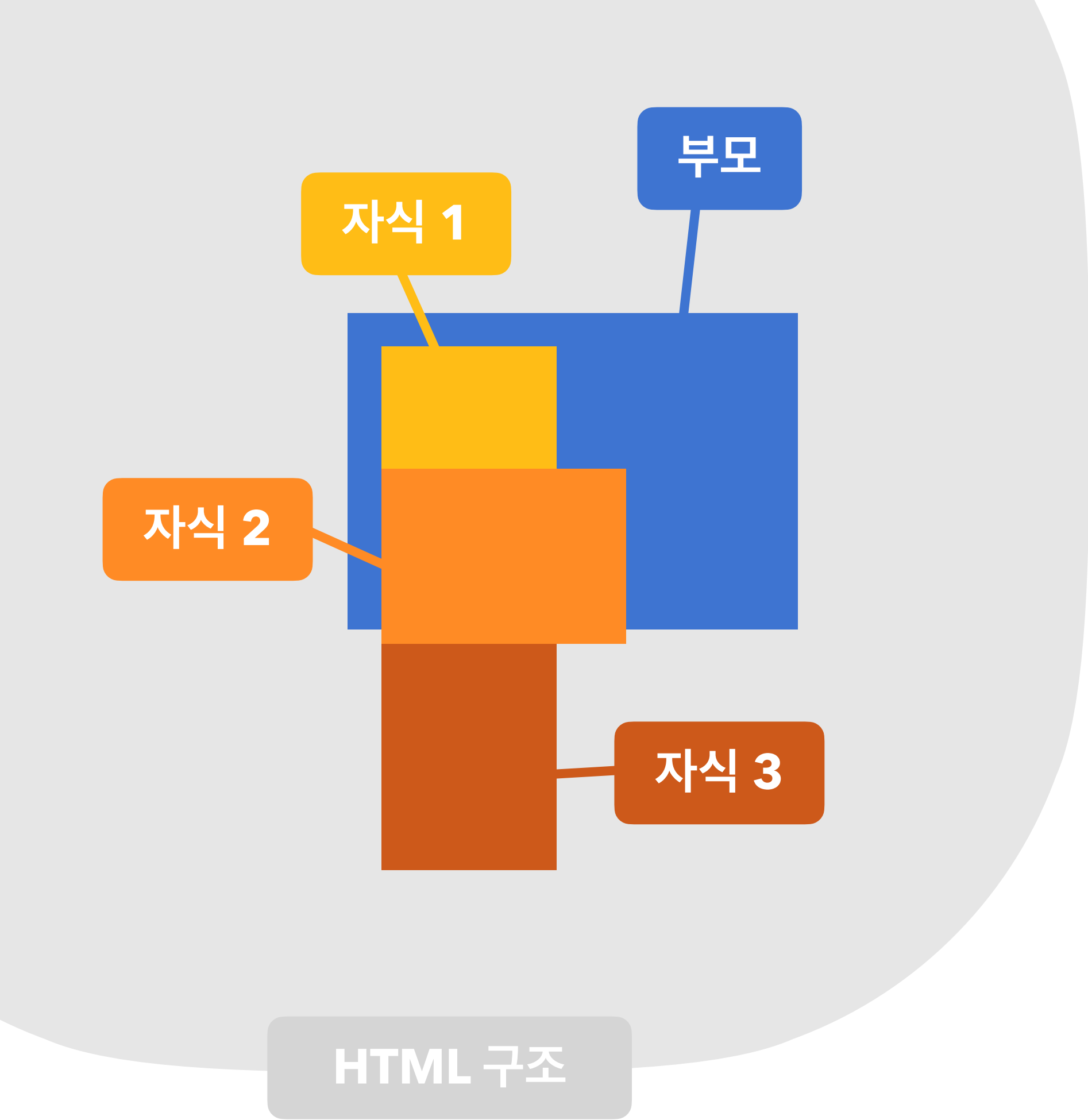


**relative**

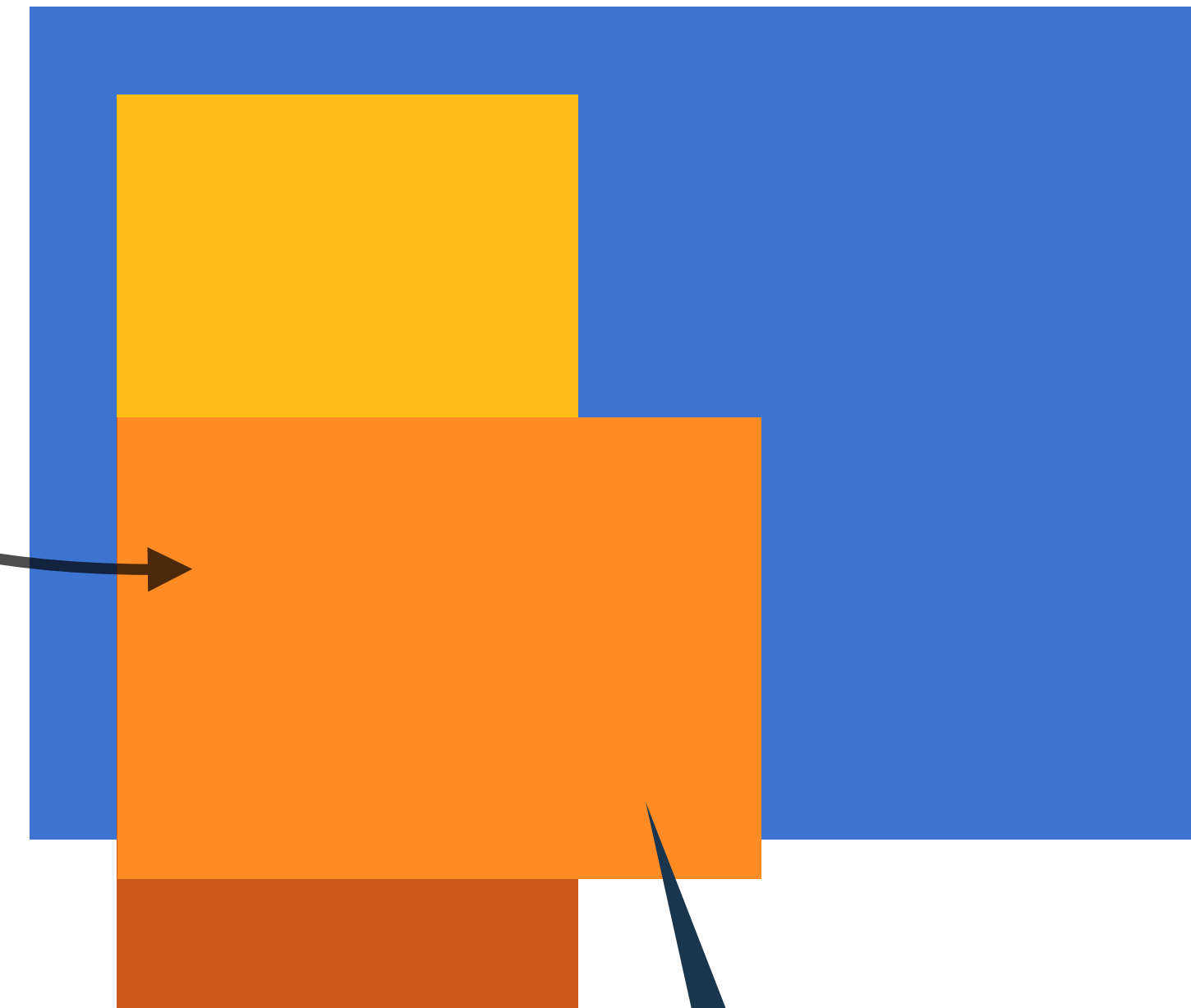
요소 자신을 기준으로 배치!



**relative** 요소 자신을 기준으로 배치!

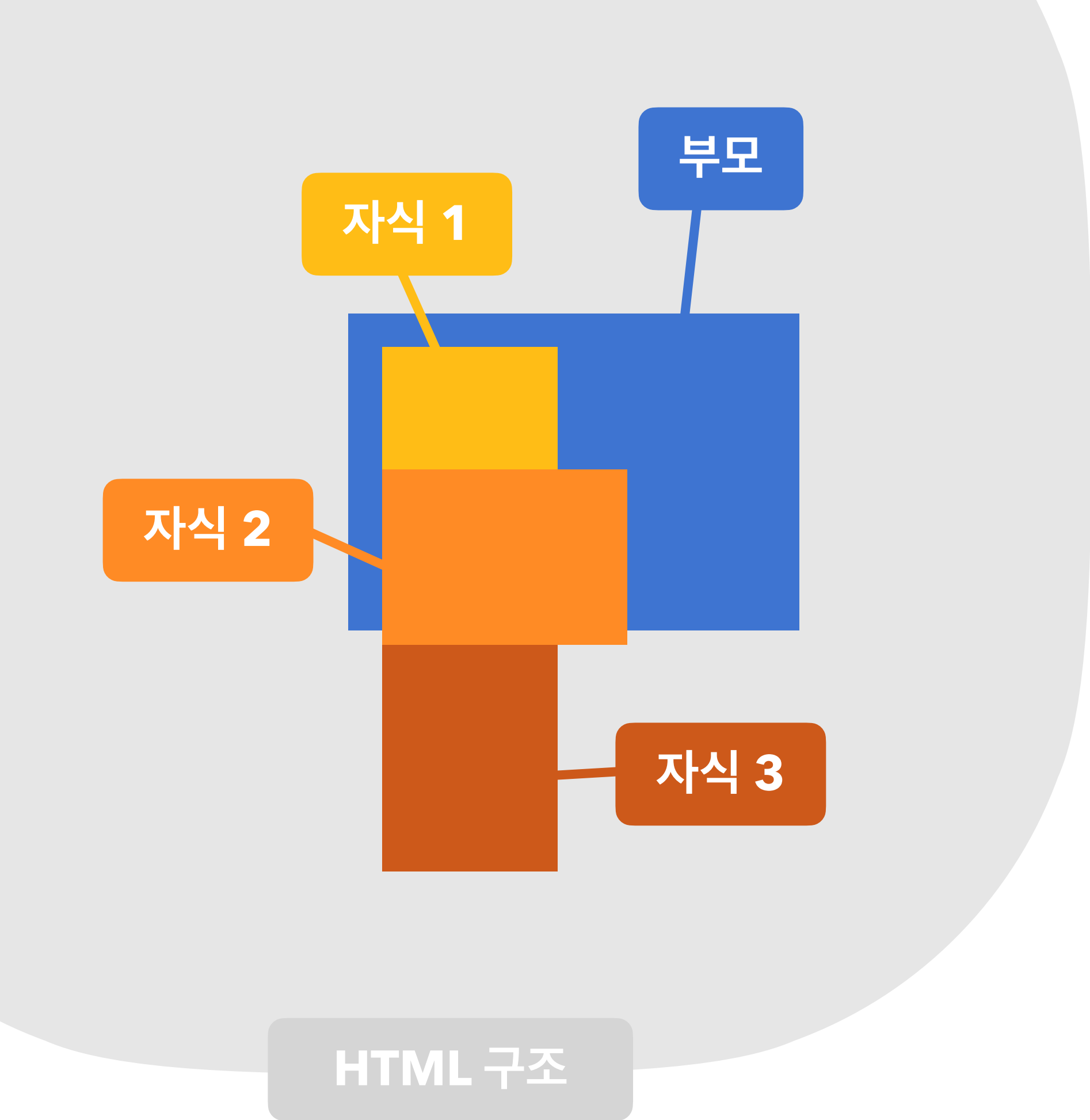


붕~ 뜨면서  
요소가 겹쳐요

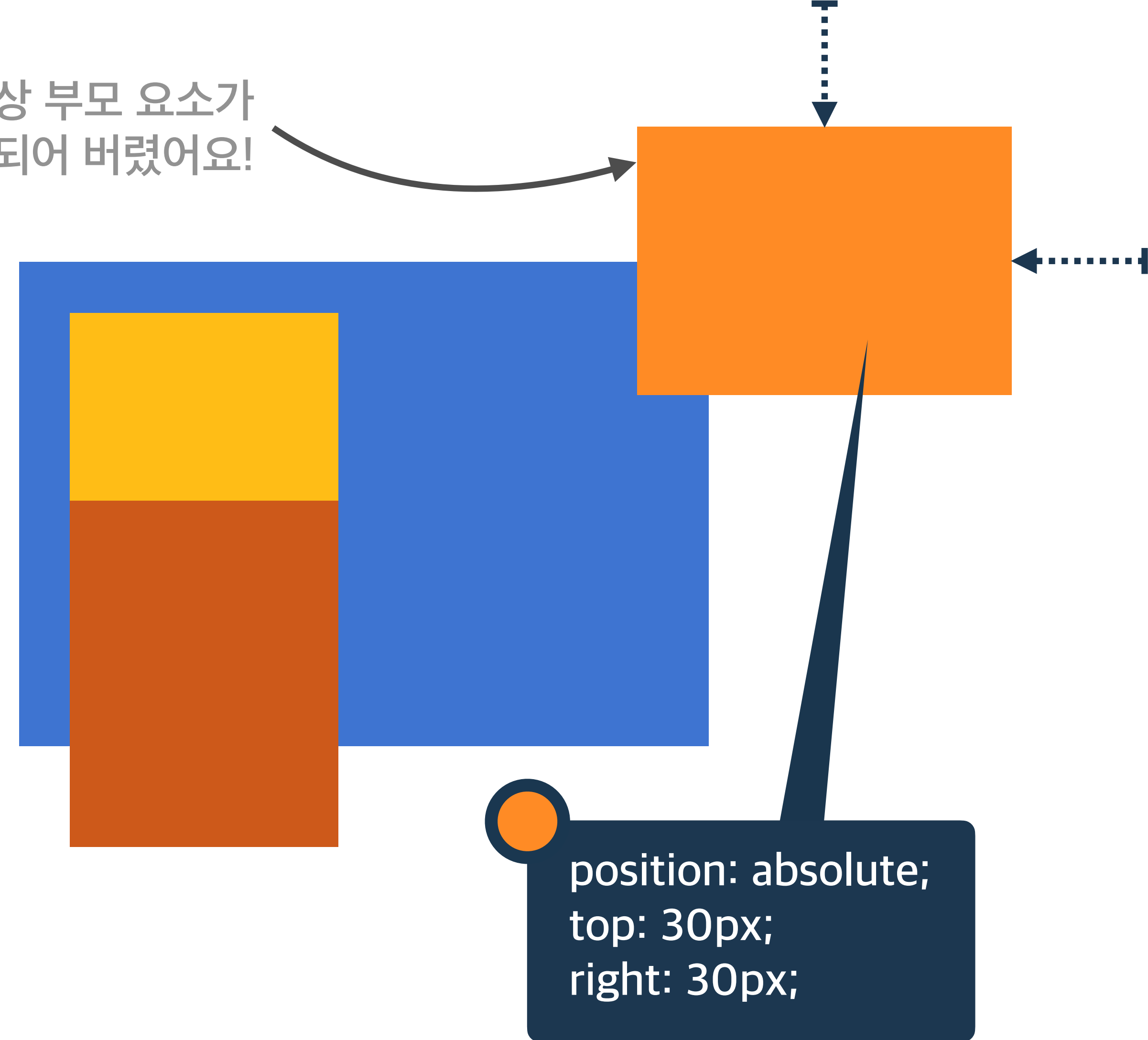


position: absolute;

**absolute** 위치 상 부모 요소를 기준으로 배치!

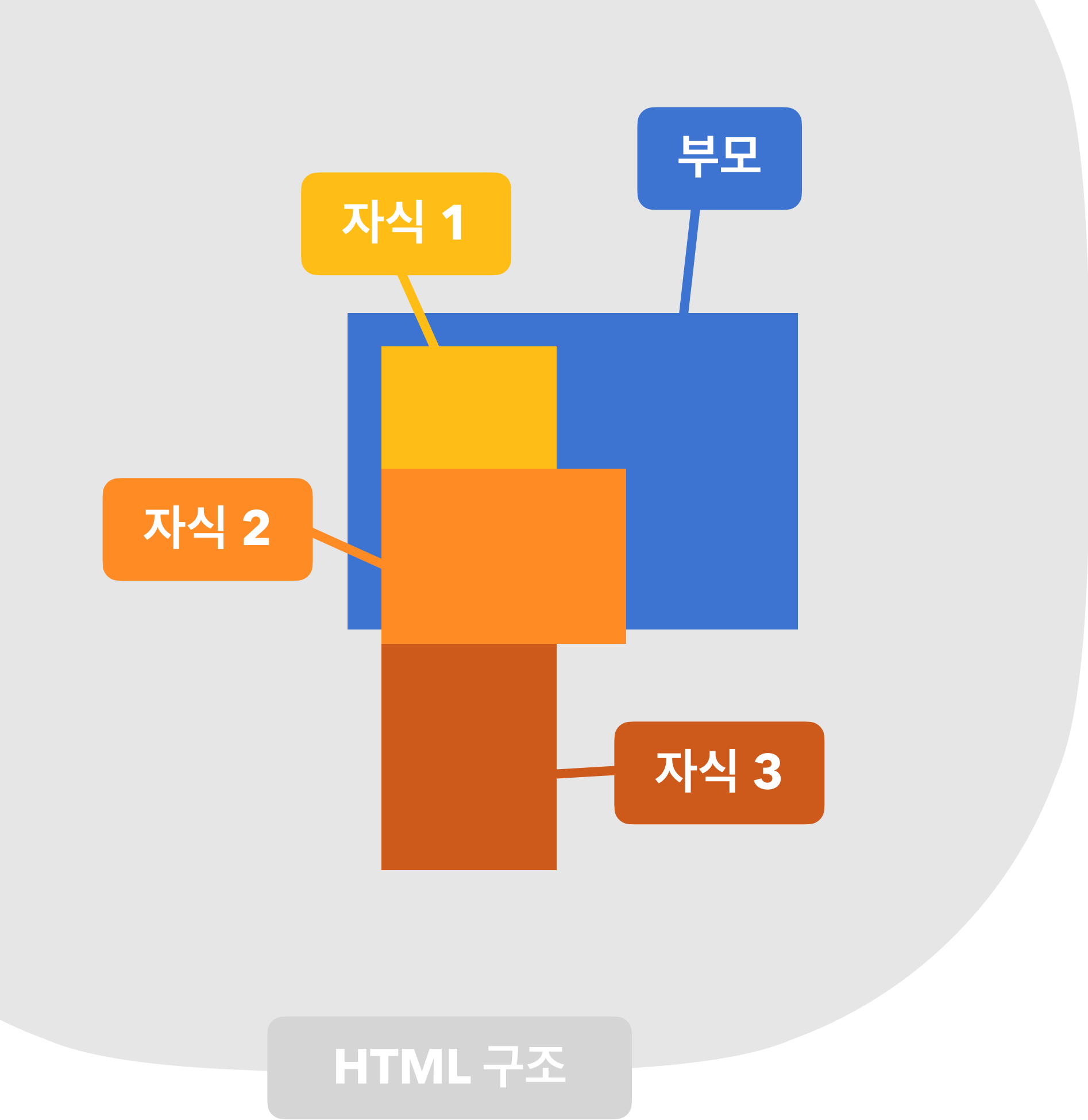


위치 상 부모 요소가  
뷰포트가 되어 버렸어요!

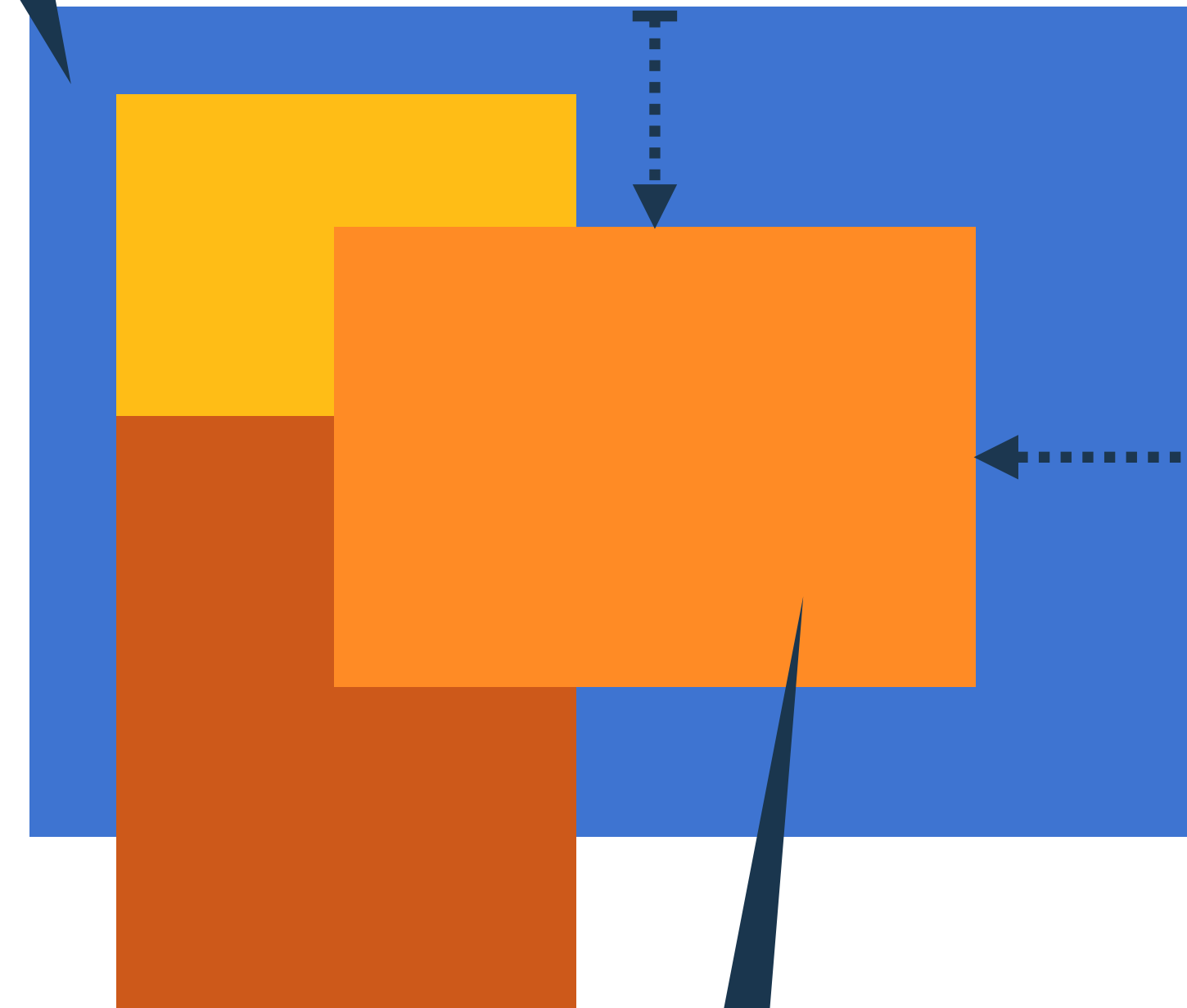


**absolute**

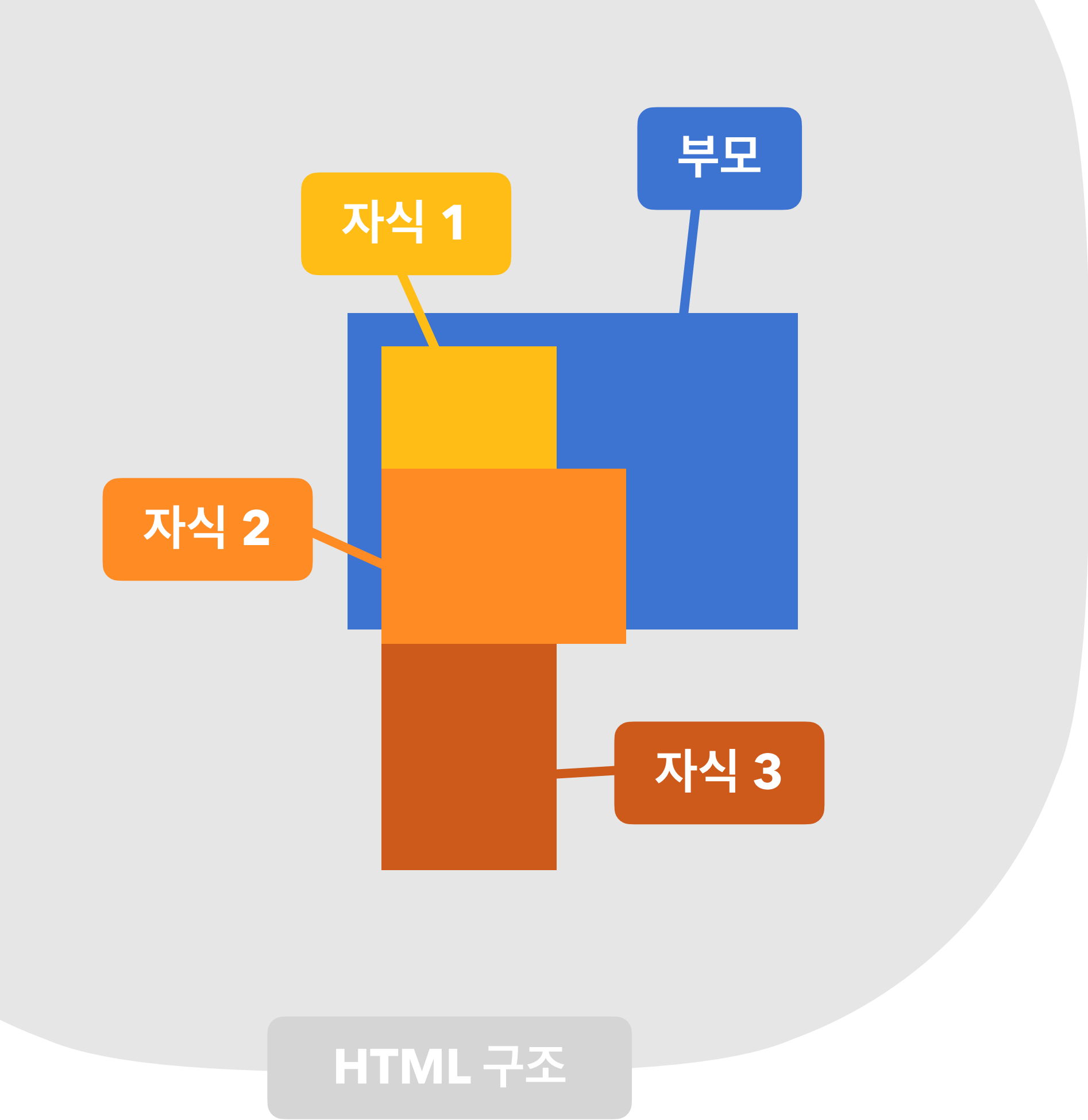
위치 상 부모 요소를 기준으로 배치!



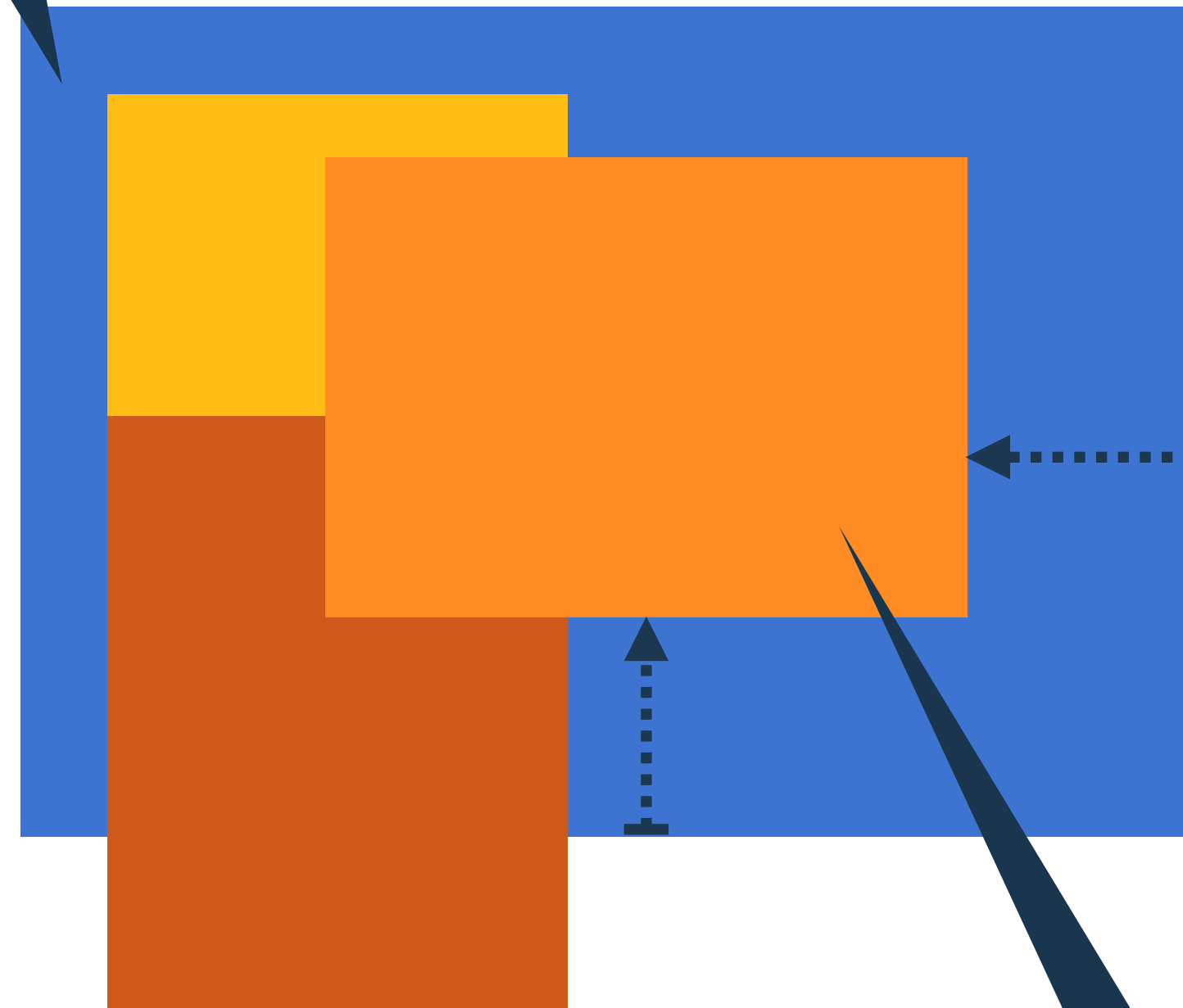
`position: relative;` 위치 상 부모 요소로 지정!



**absolute** 위치 상 부모 요소를 기준으로 배치!



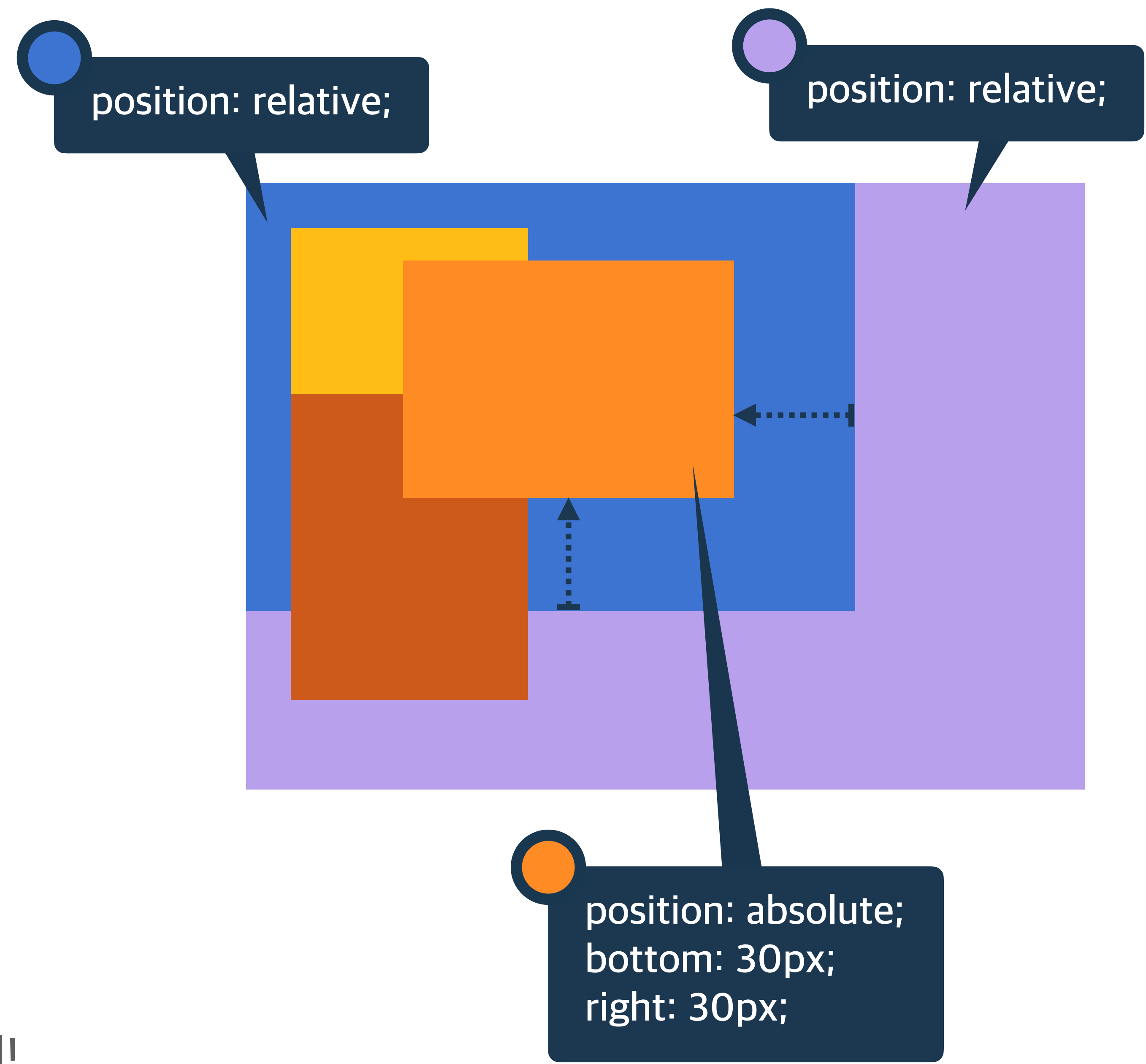
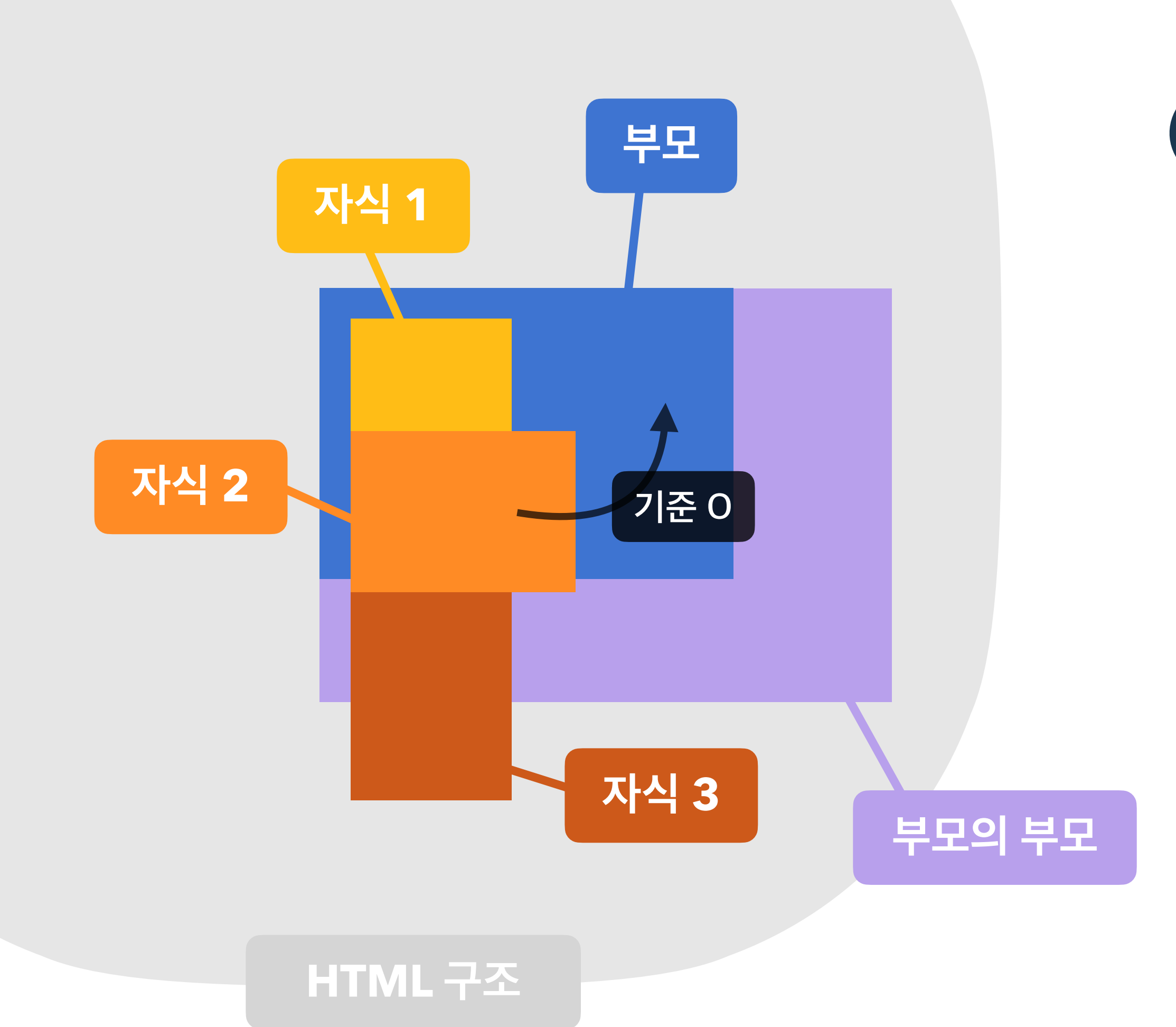
position: relative;



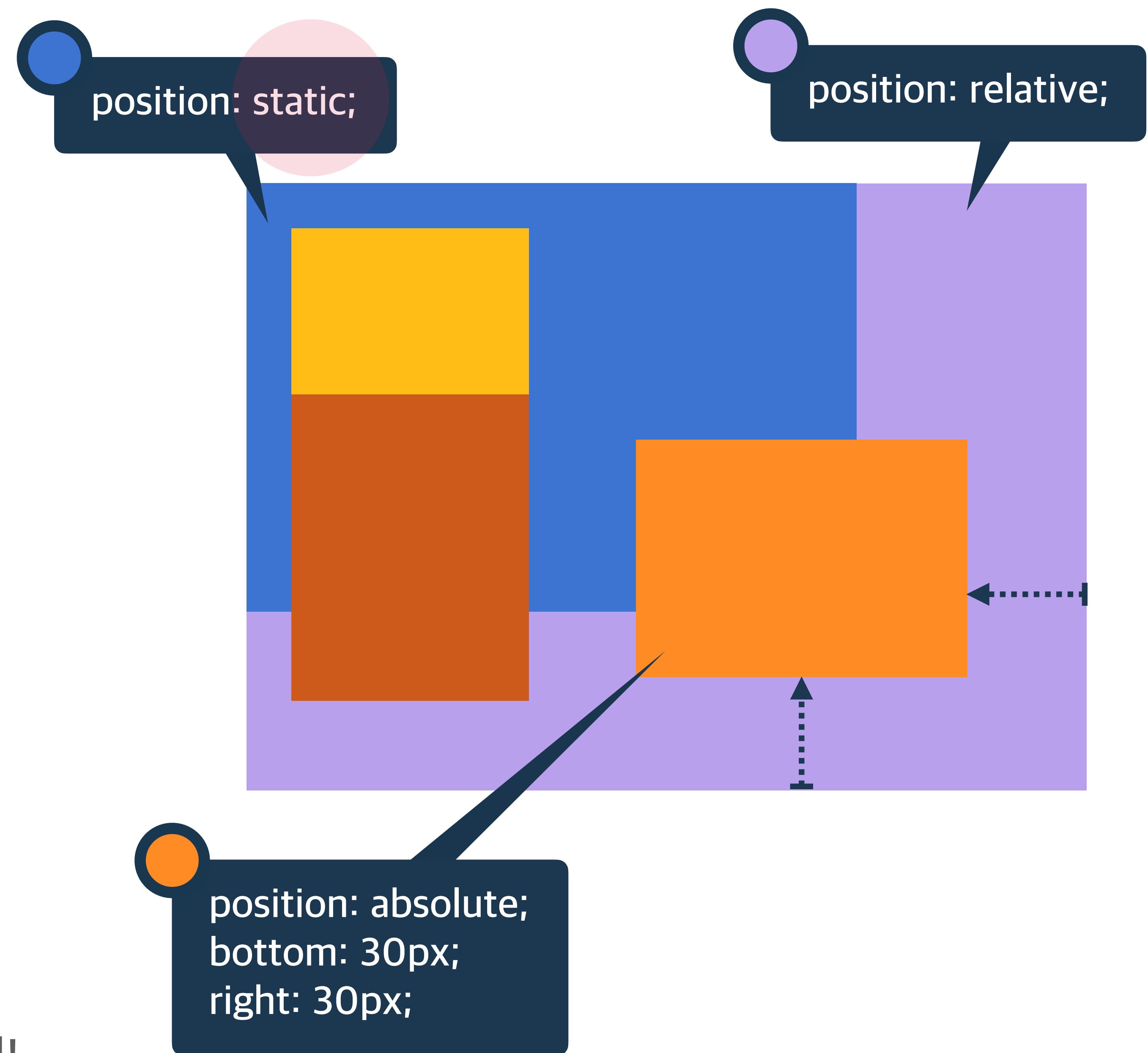
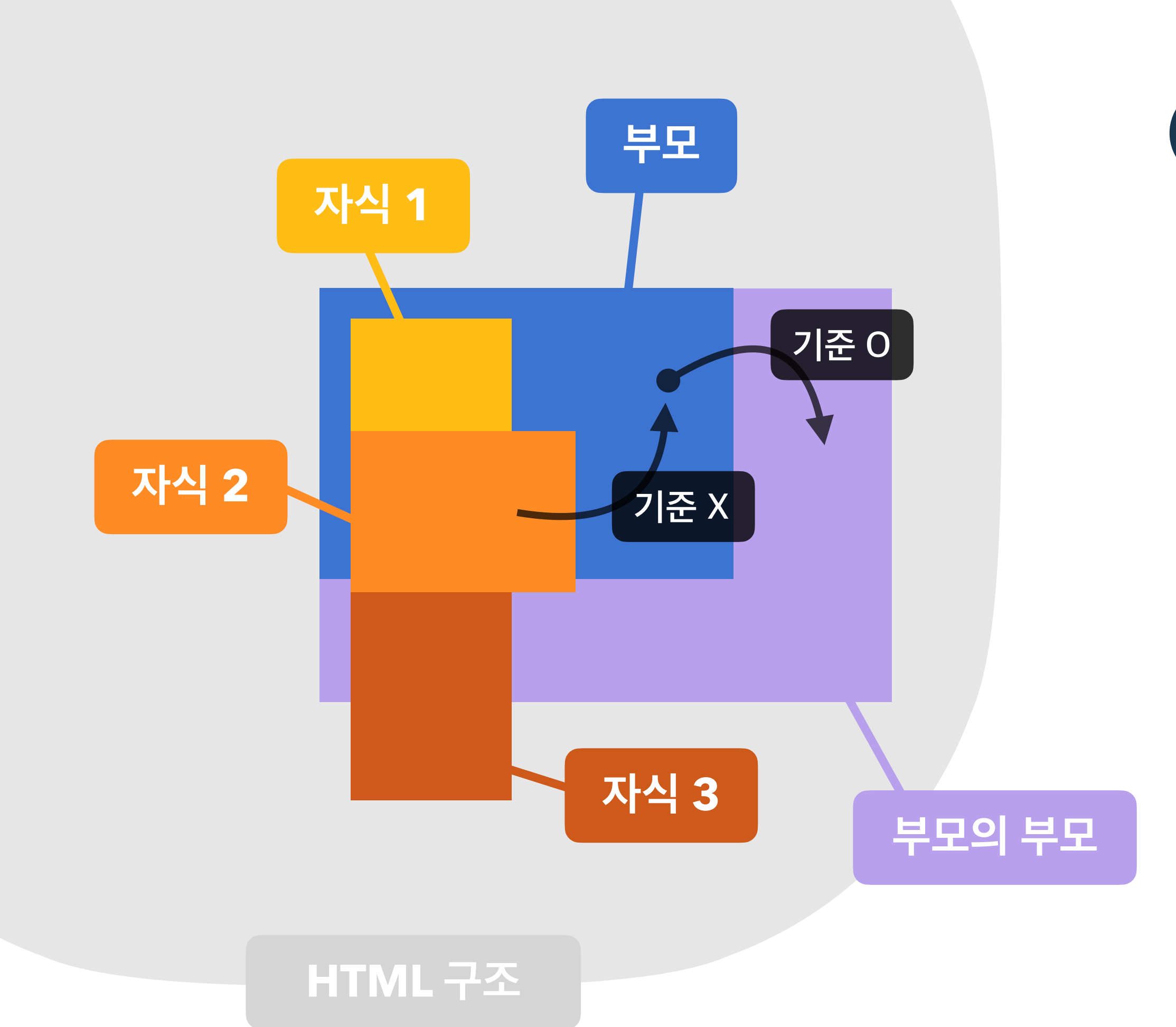
position: absolute;  
bottom: 30px;  
right: 30px;

**absolute** 위치 상 부모 요소를 기준으로 배치!

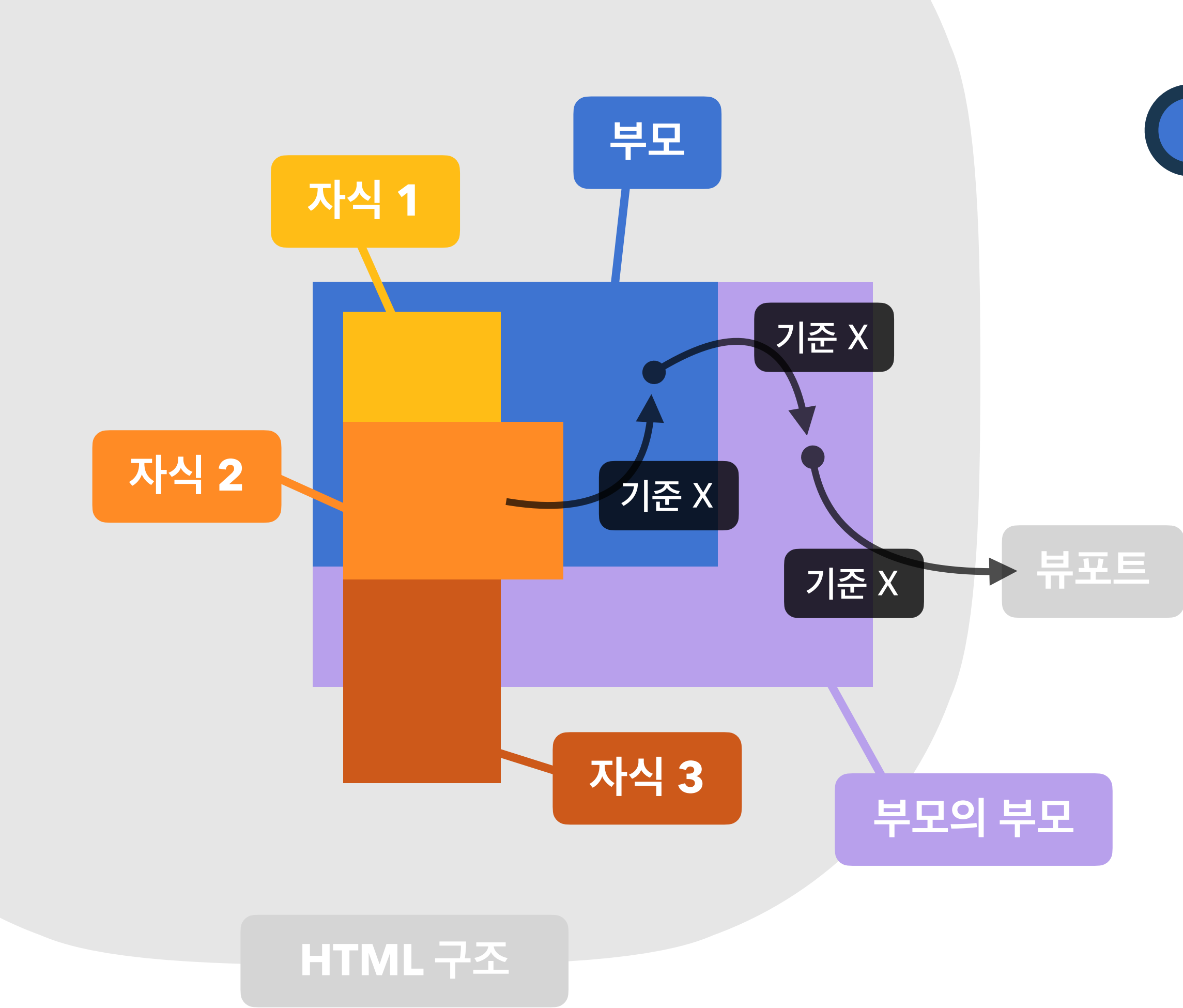




**absolute** 위치 상 부모 요소를 기준으로 배치!



**absolute** 위치 상 부모 요소를 기준으로 배치!

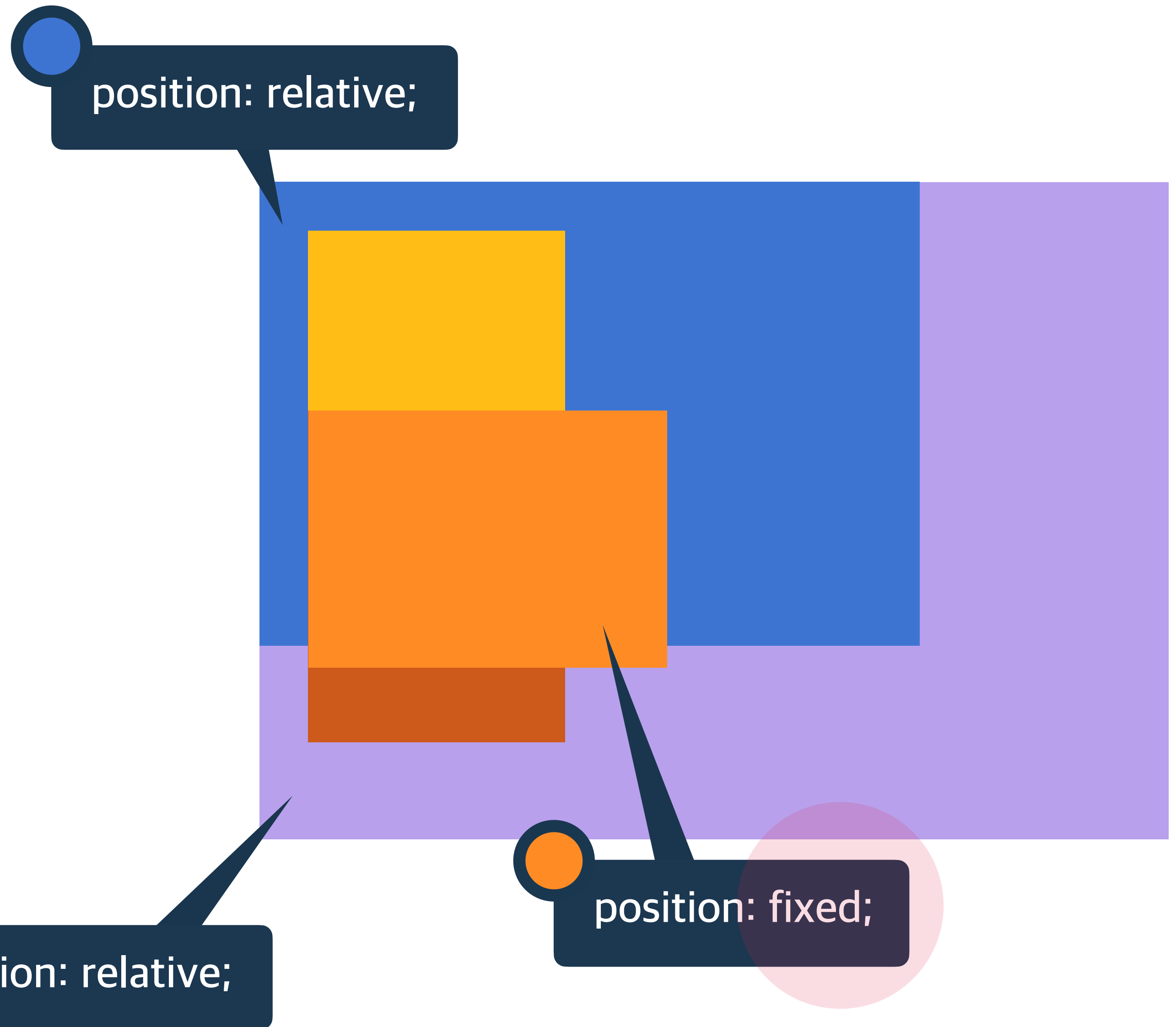
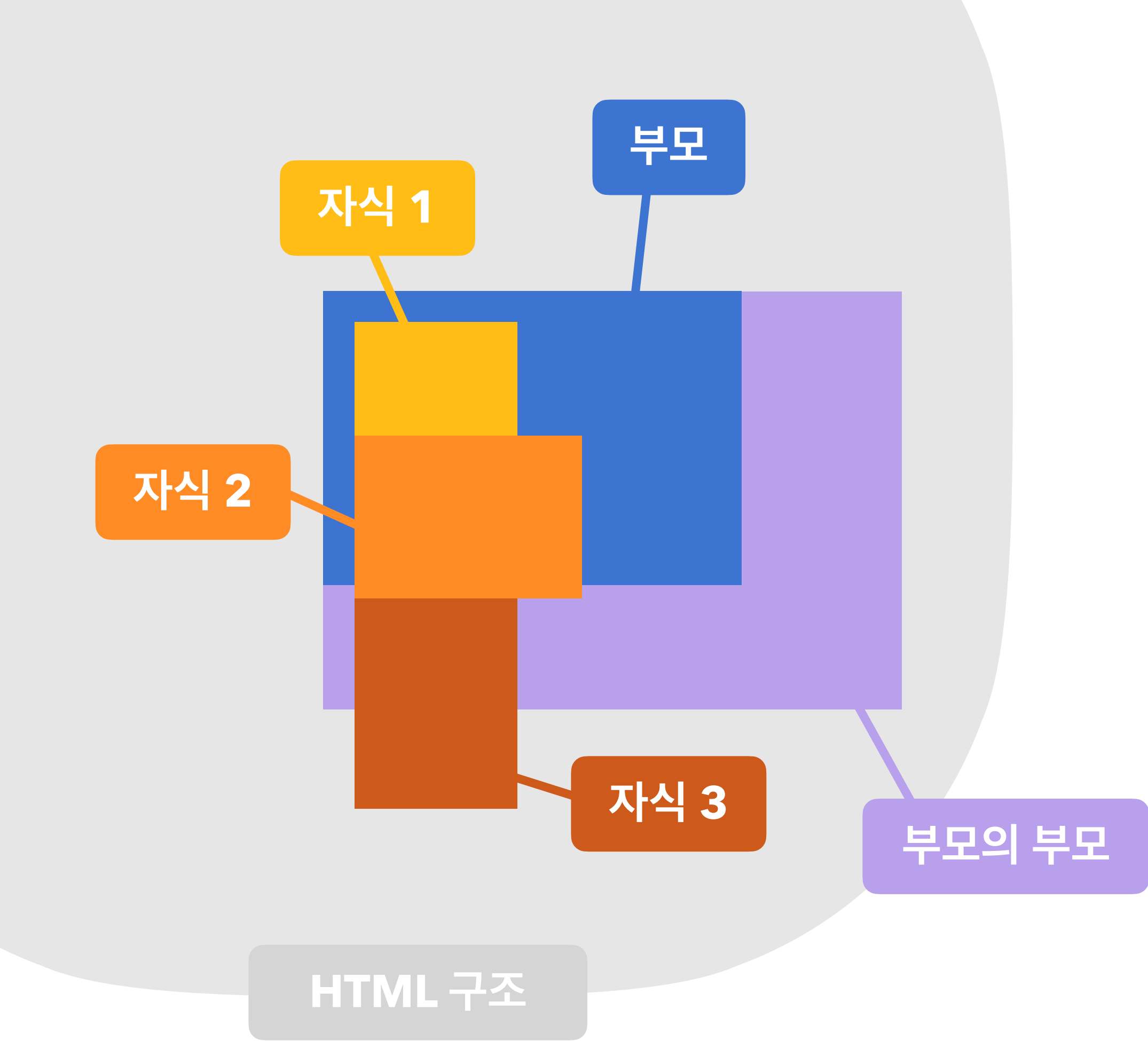


position: static;

position: static;

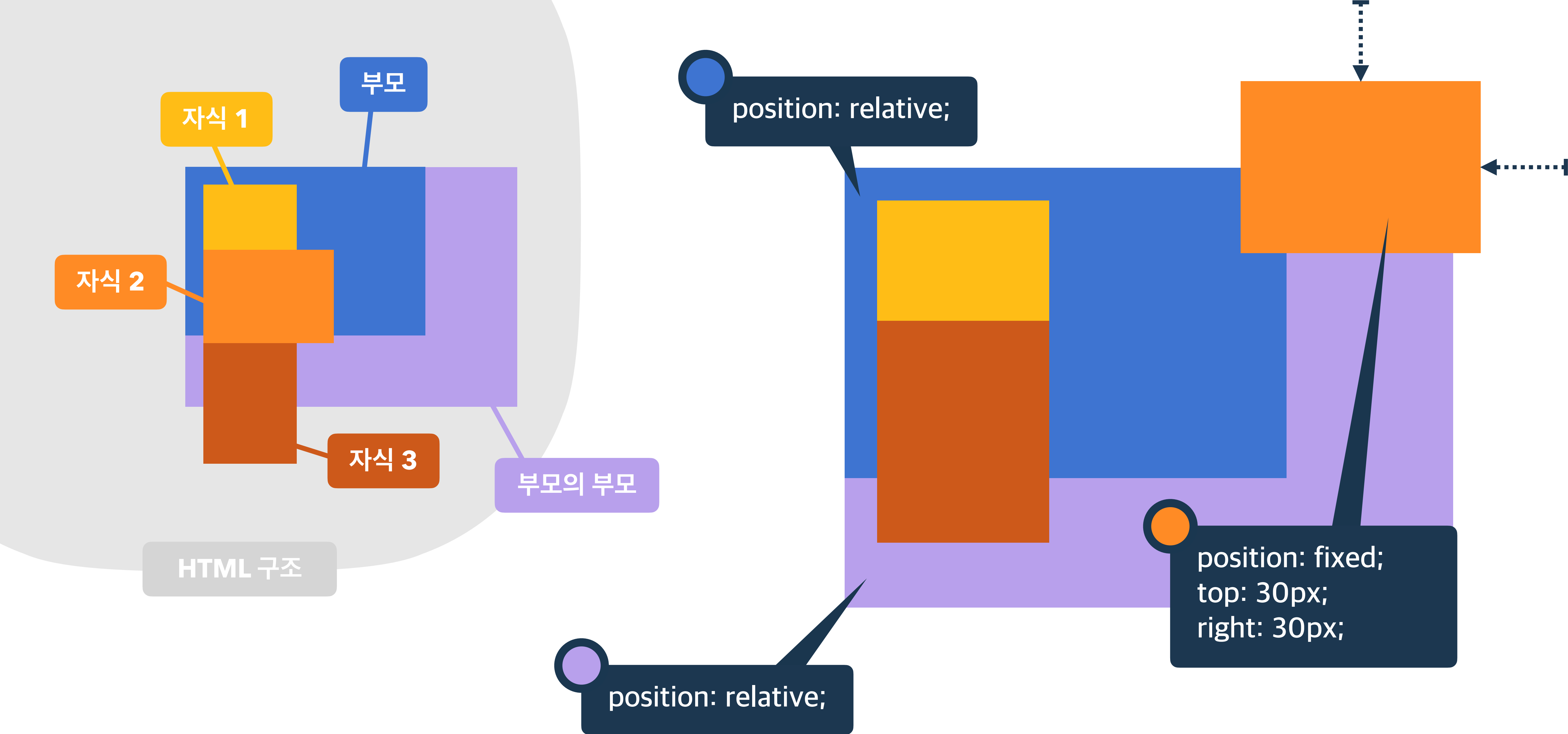
position: absolute;  
bottom: 30px;  
right: 30px;

**absolute** 위치 상 부모 요소를 기준으로 배치!



**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

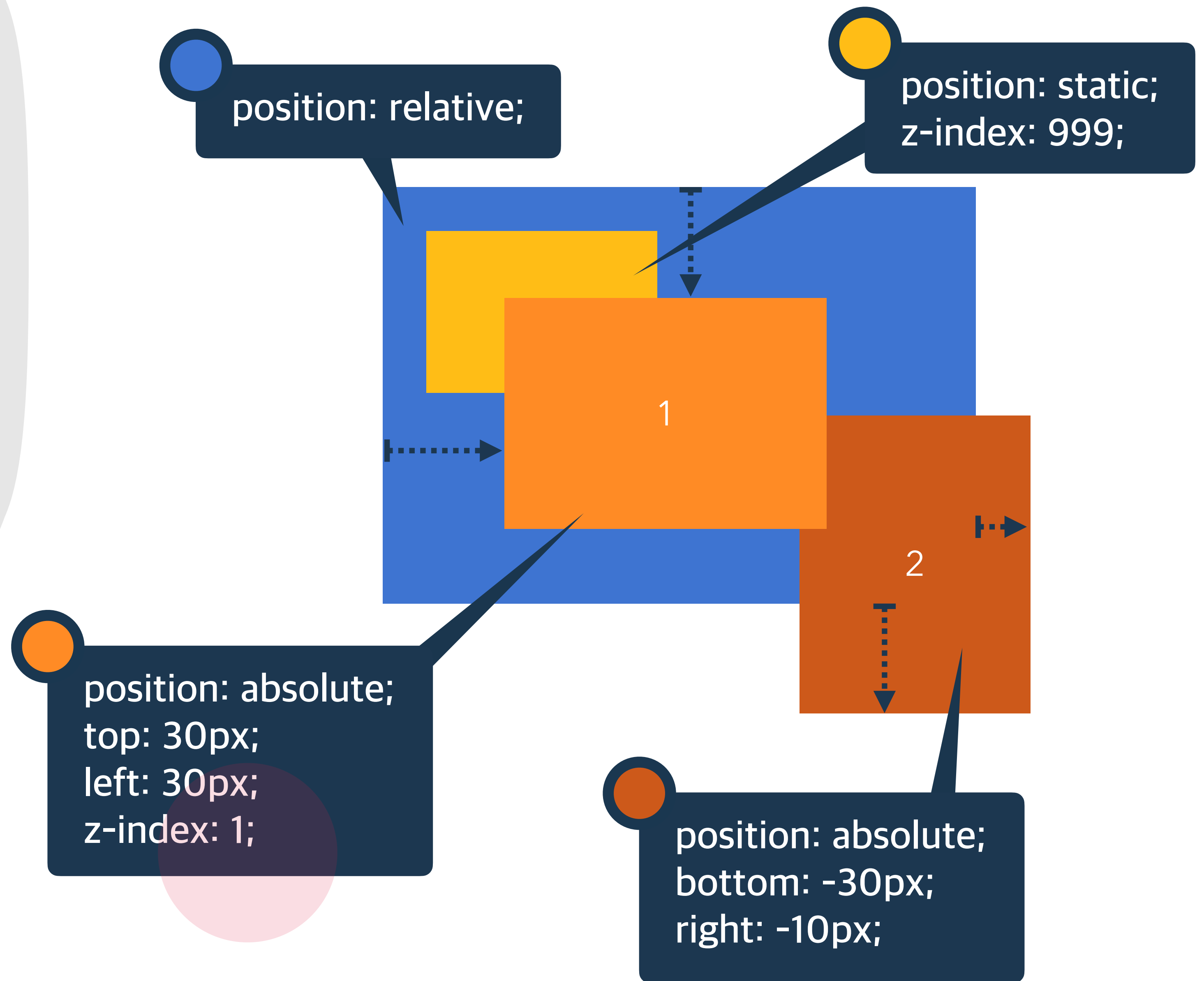
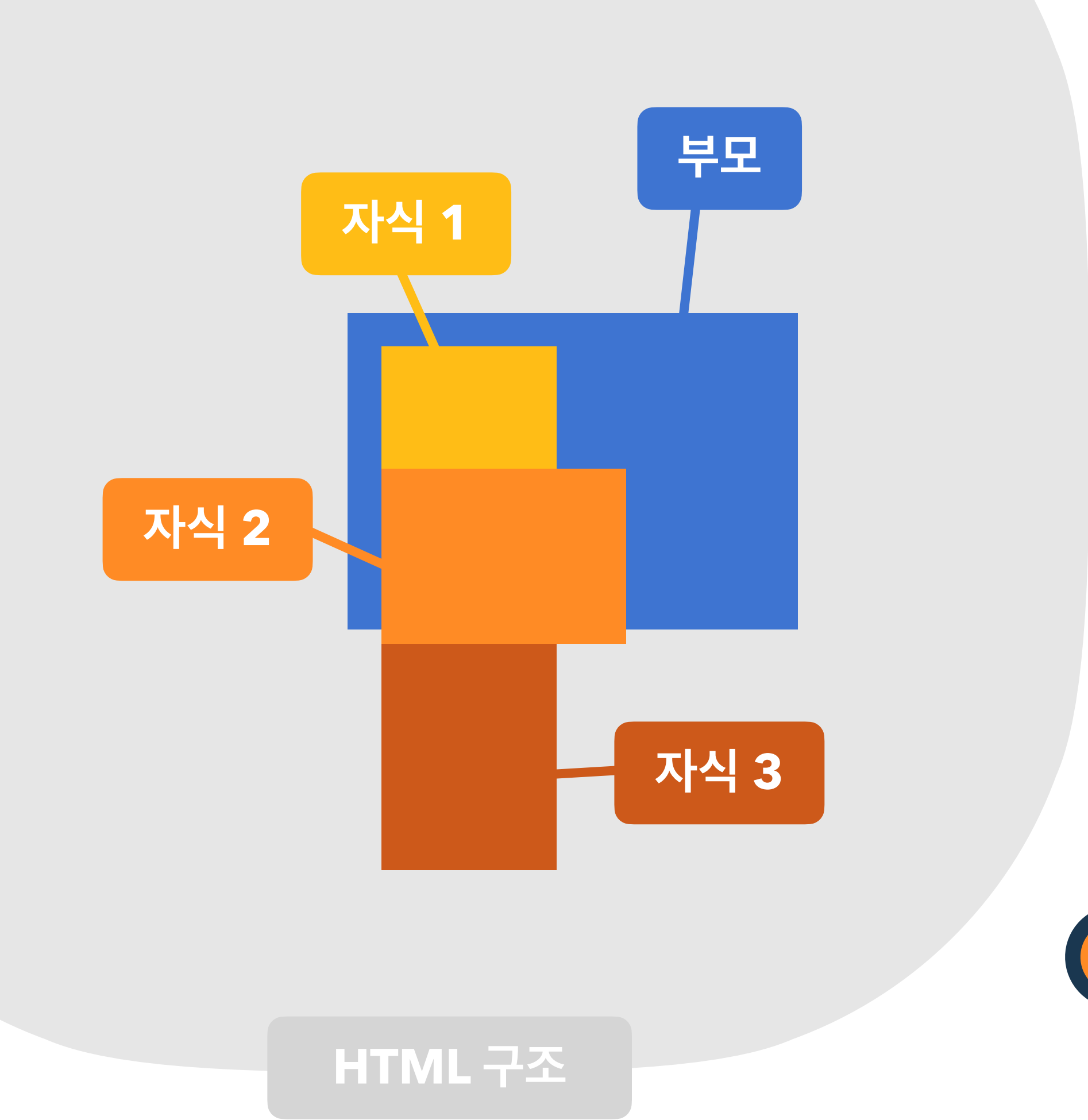
# z-index

**auto**

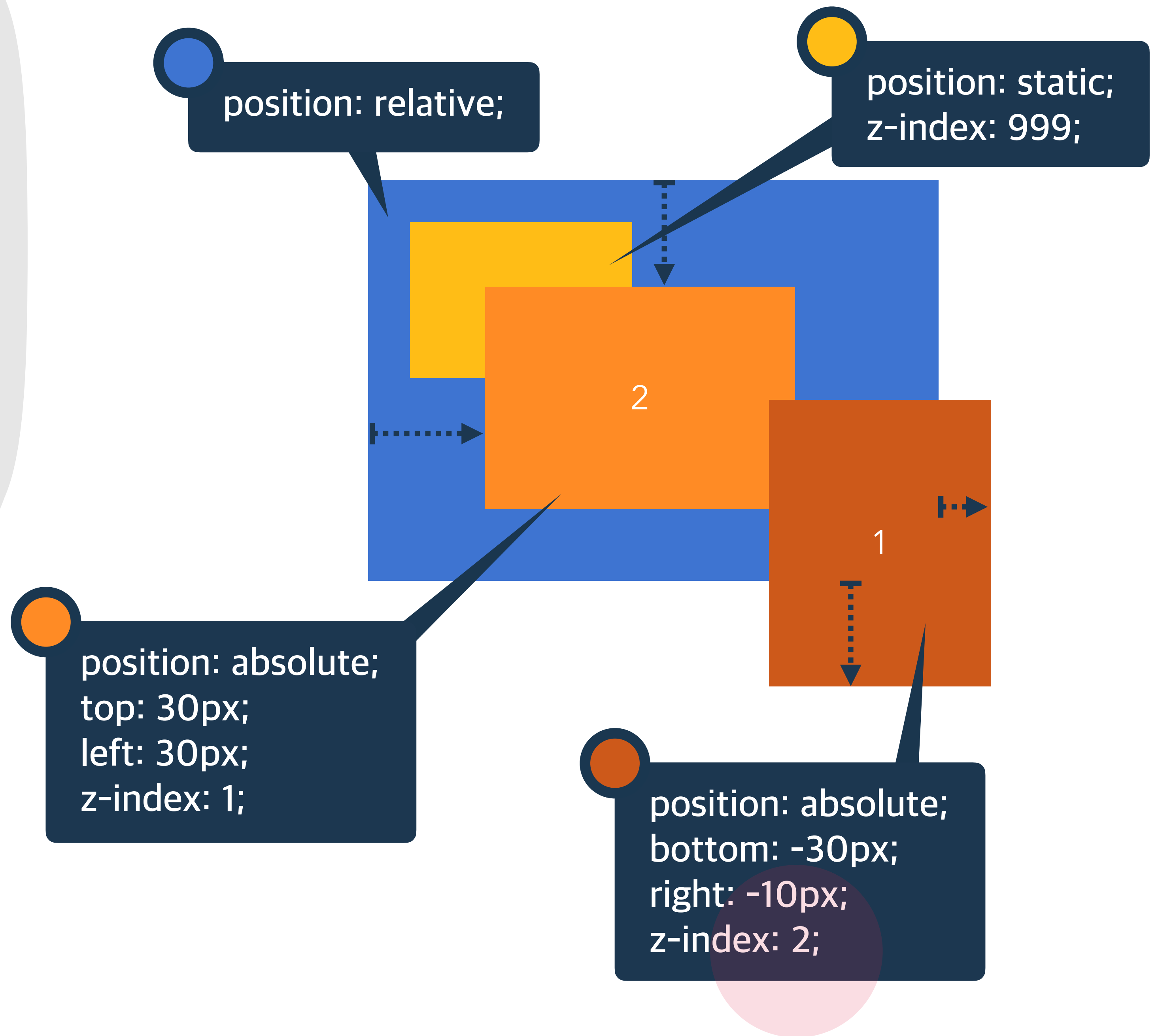
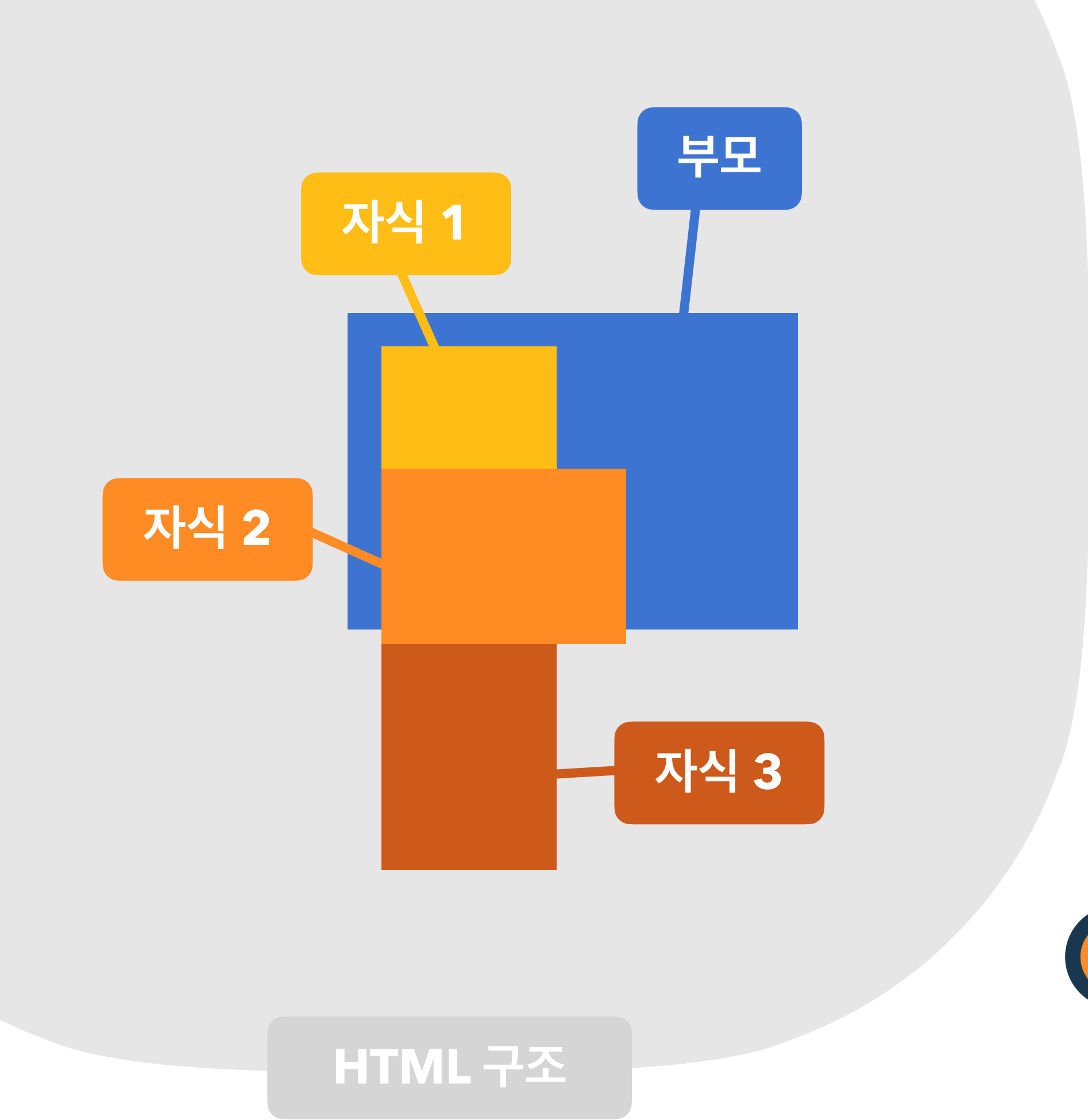
부모 요소와 동일한 쌓임 정도

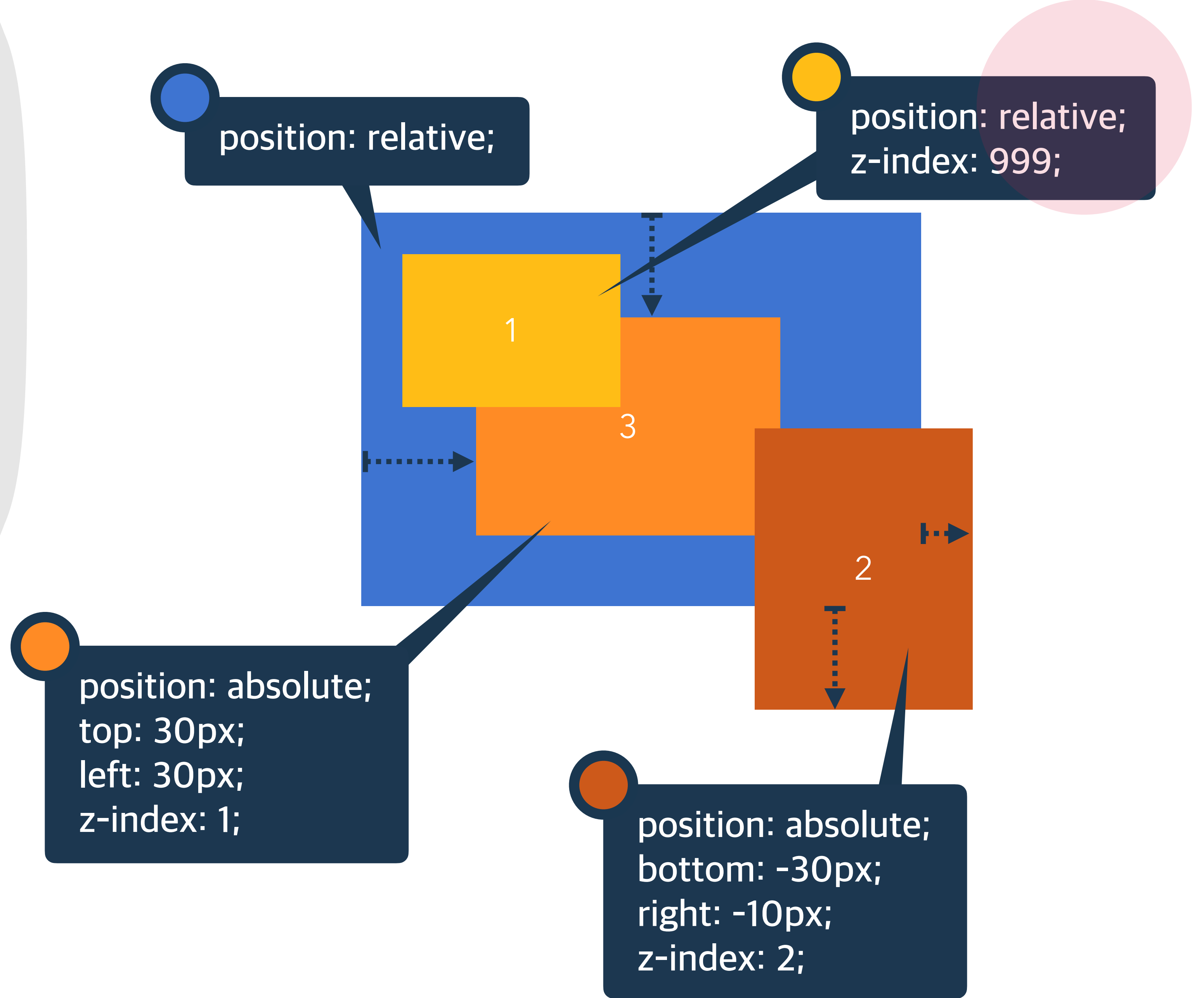
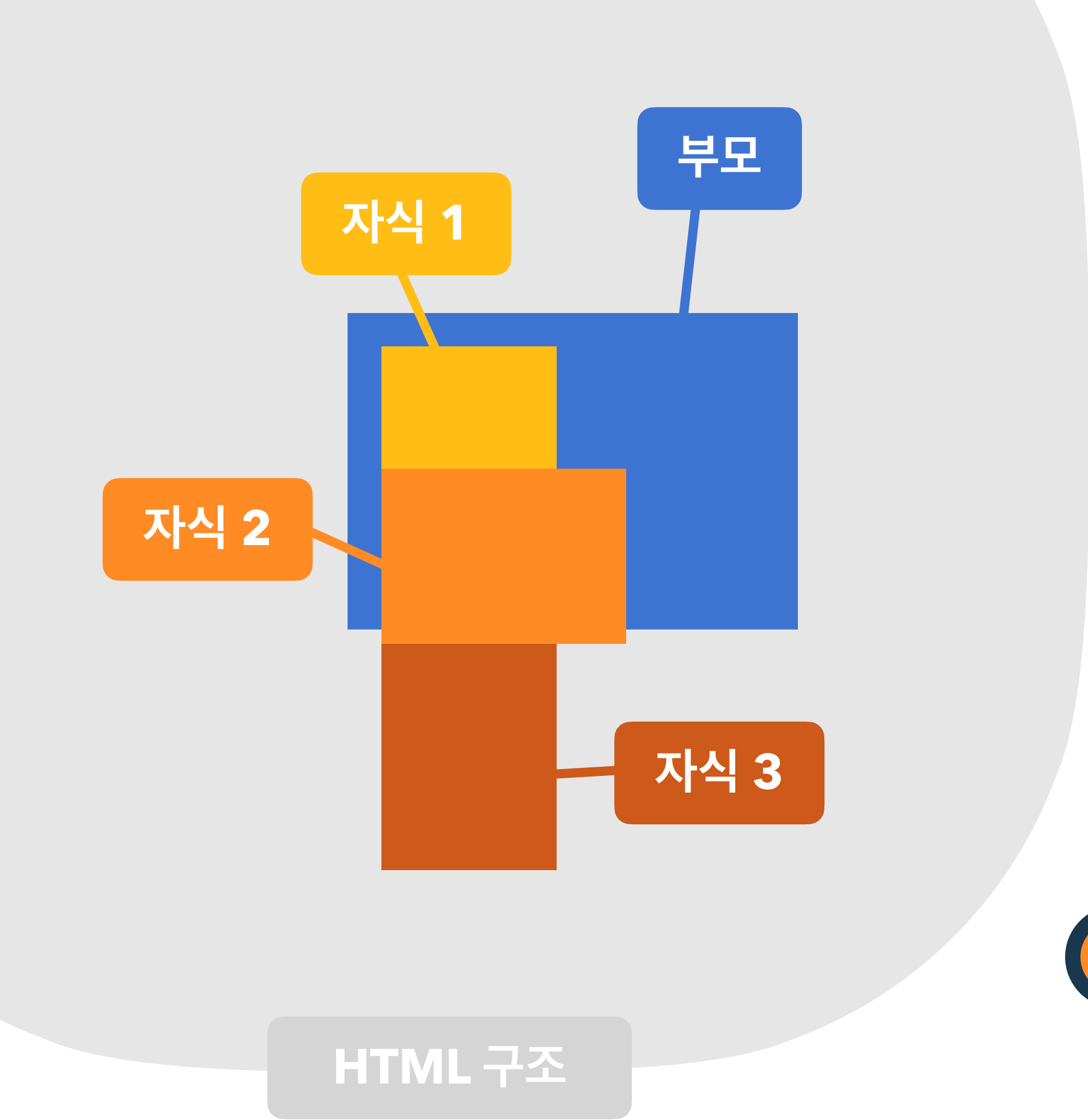
**숫자**

숫자가 높을 수록 위에 쌓임



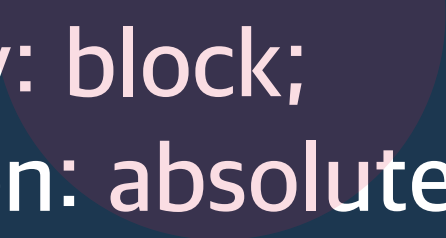






# 요소의 display가 변경됨

position 속성의 값으로 absolute, fixed가 지정된 요소는,  
display 속성이 block으로 변경됨.

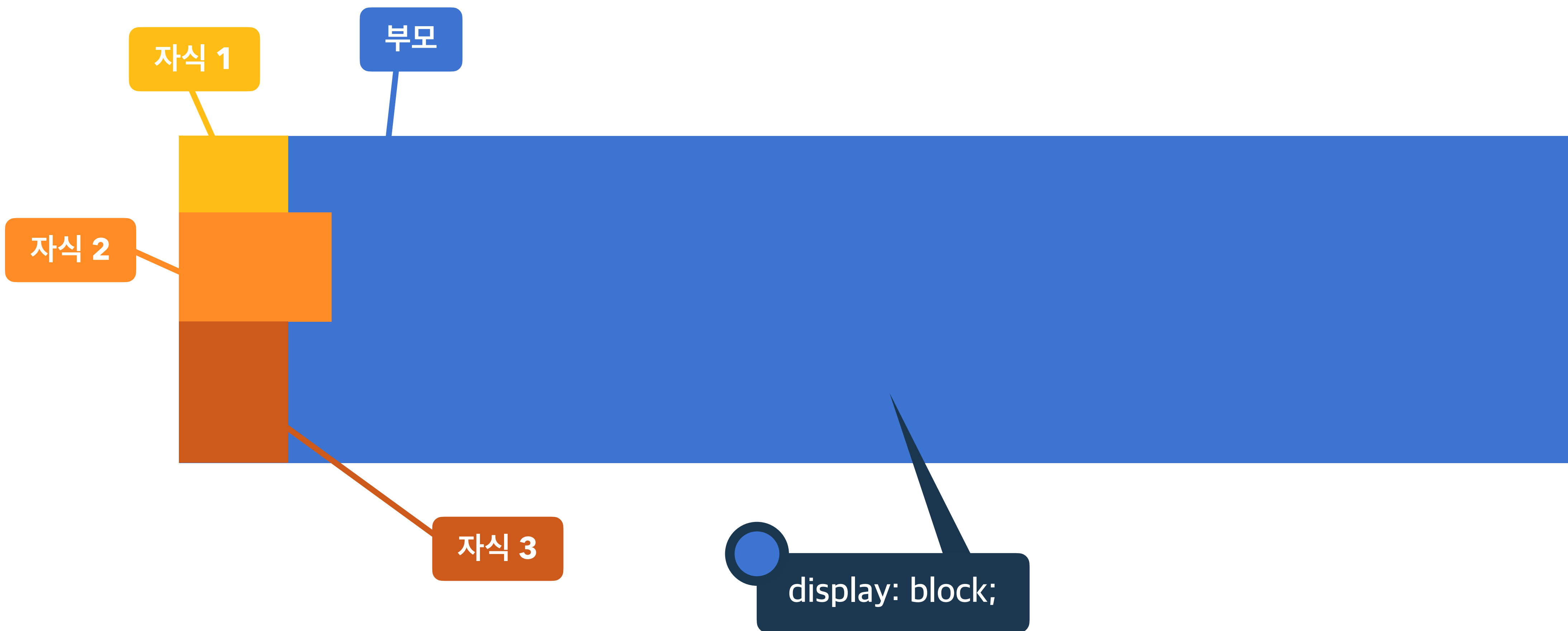


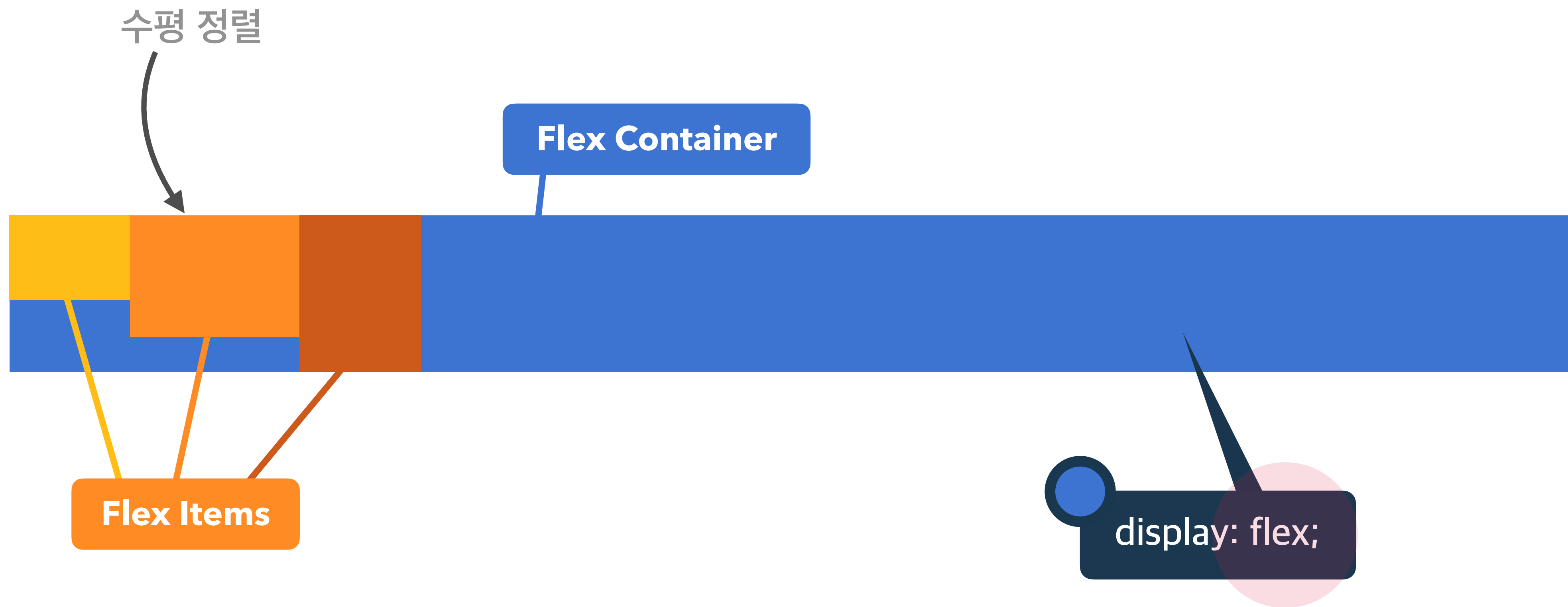
```
display: block;  
position: absolute;  
top: 30px;  
left: 30px;  
z-index: 1;
```

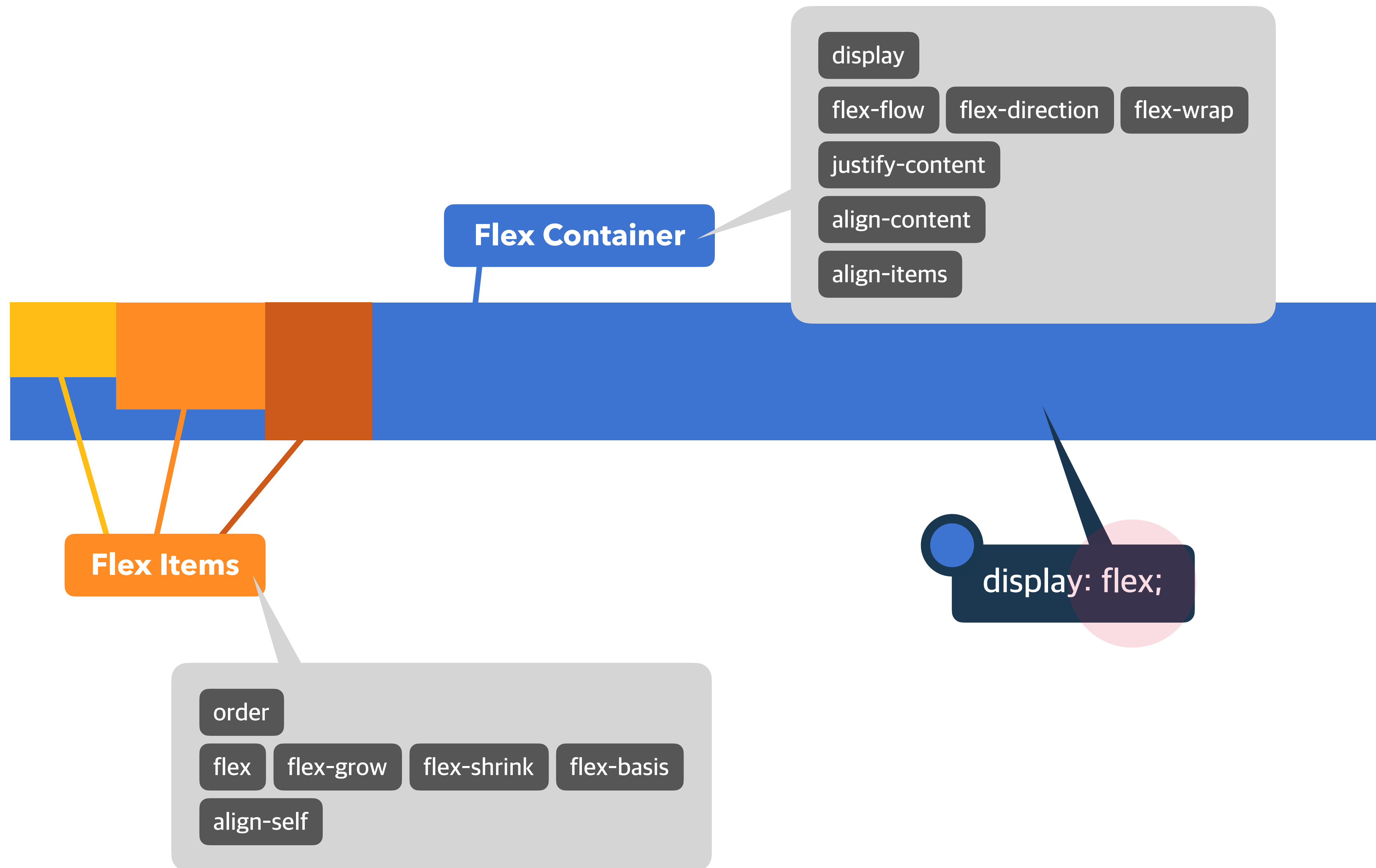
=

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

플렉스(정렬)









Flex Container의 화면 출력(보여짐) 특성

# display

**flex**

블록 요소와 같이 Flex Container 정의

**inline-flex**

인라인 요소와 같이 Flex Container 정의





**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

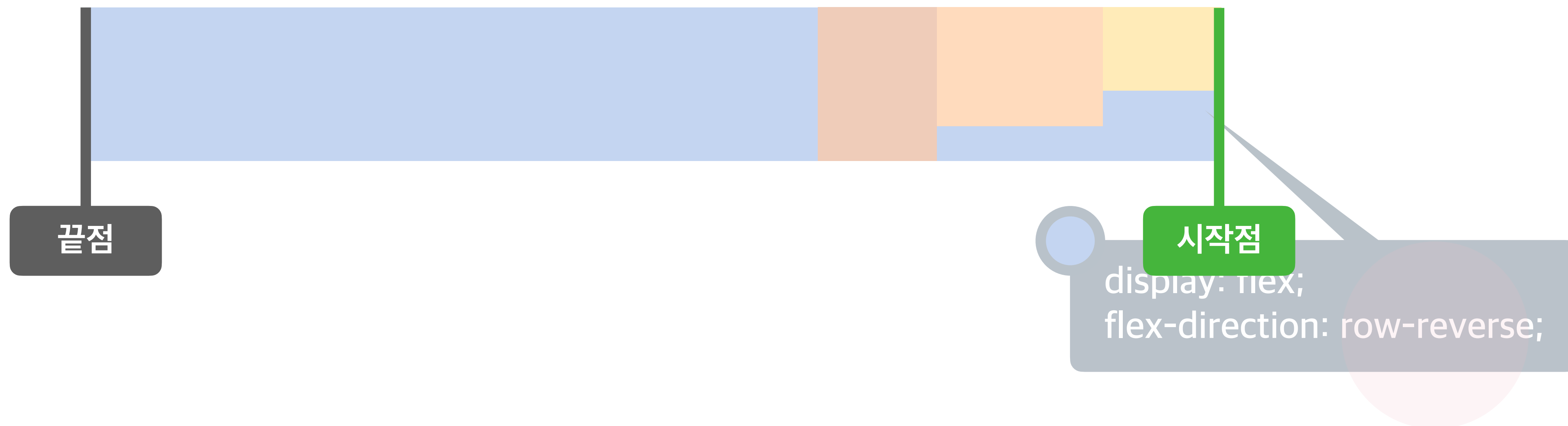
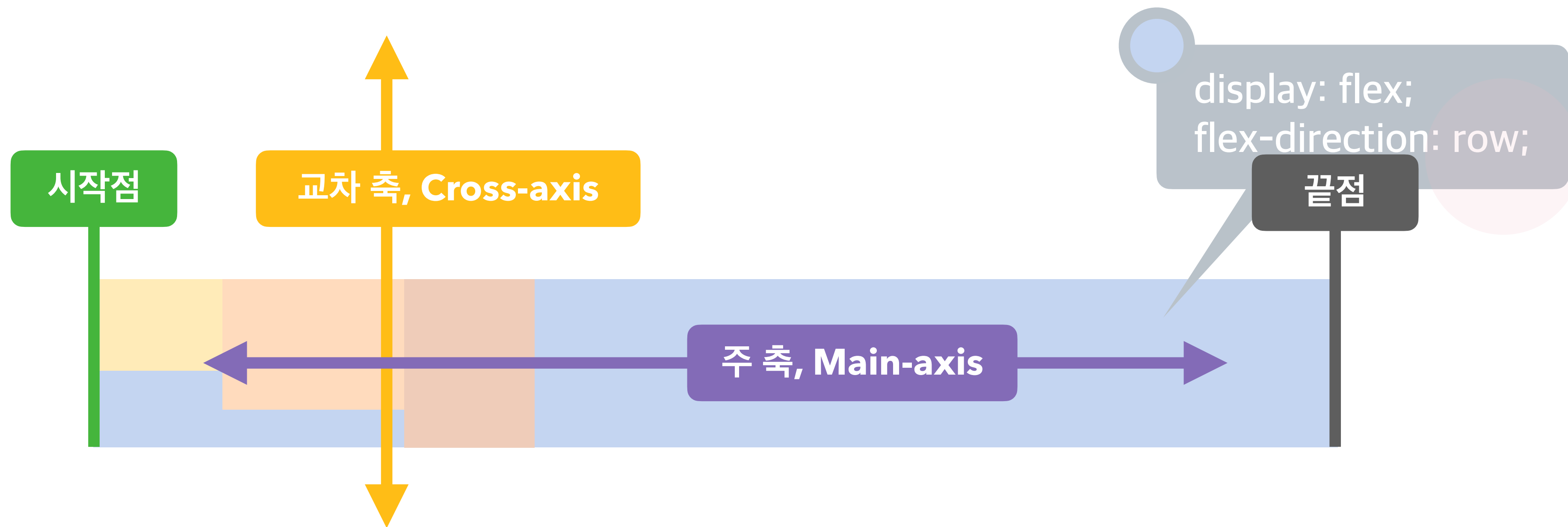
**열, Column**




display: flex;  
flex-direction: row;



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







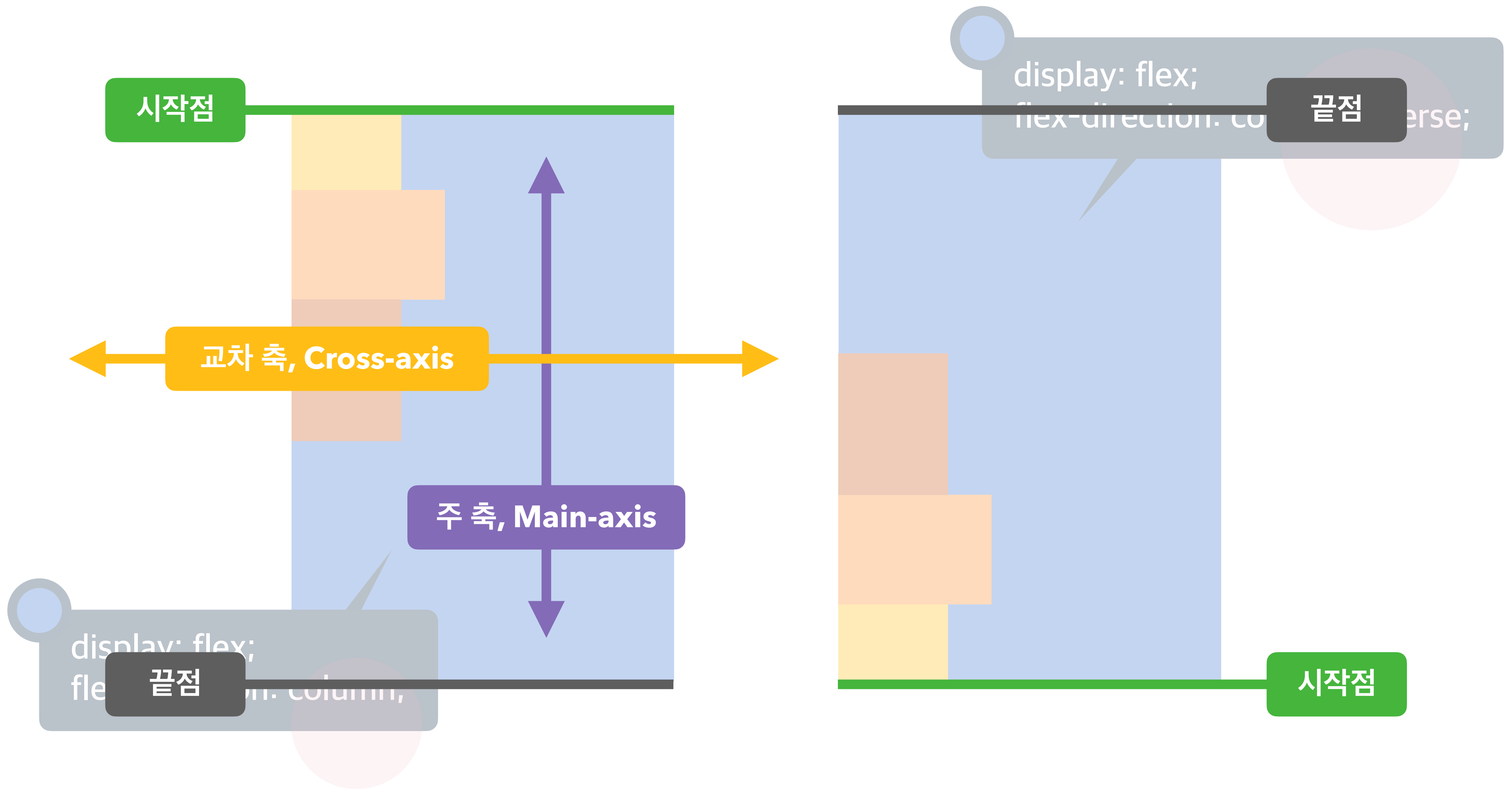
A large blue square represents a flex container. On the left side, three smaller squares are stacked vertically: a yellow square at the top, an orange square in the middle, and a brown square at the bottom.

`display: flex;`  
`flex-direction: column;`



A large blue square represents a flex container. On the left side, three smaller squares are stacked vertically: a brown square at the top, an orange square in the middle, and a yellow square at the bottom.

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



Flex Items 묶음(줄 바꿈) 여부

# flex-wrap

**nowrap**

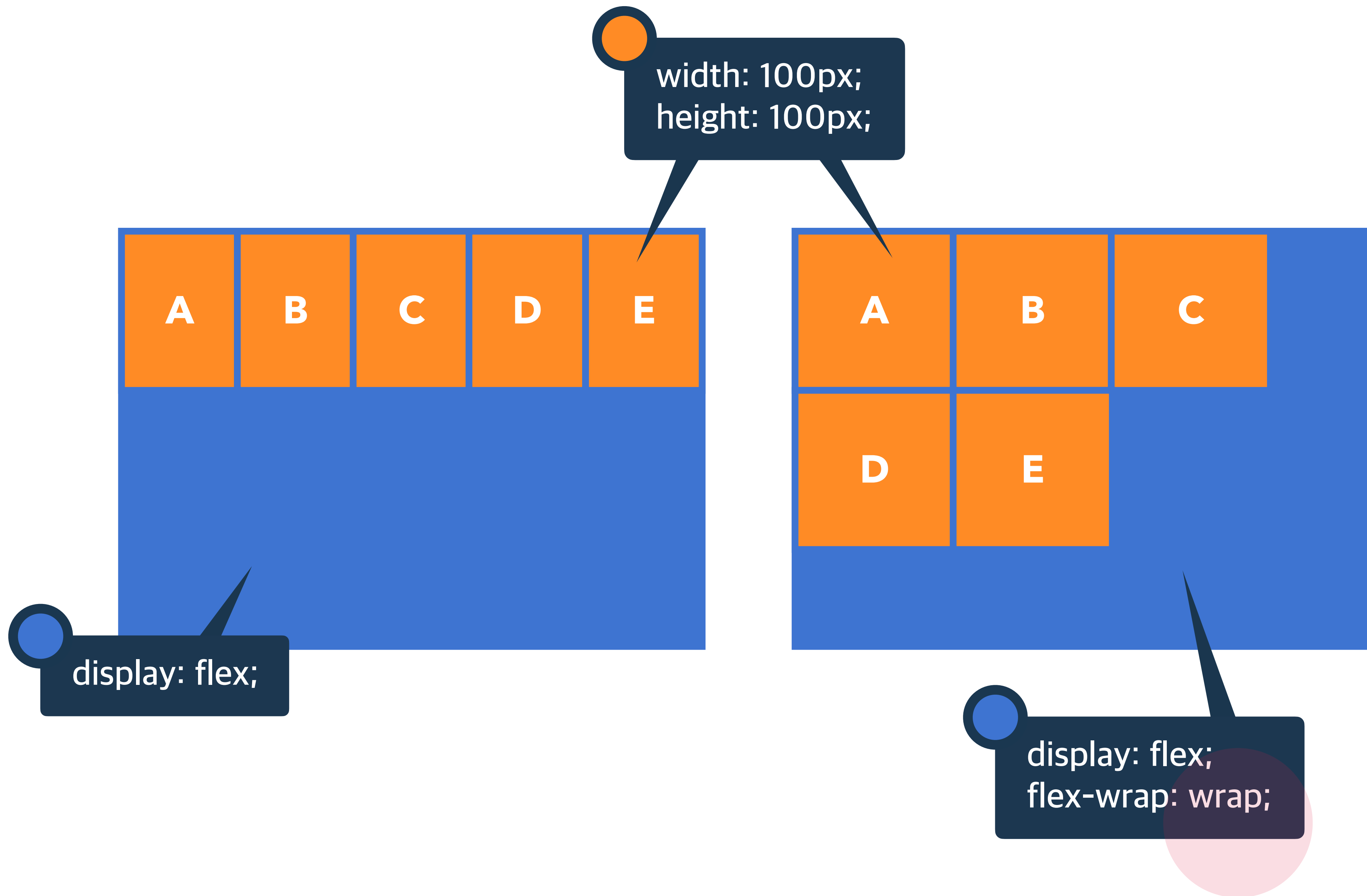
묶음(줄 바꿈) 없음

**wrap**

여러 줄로 묶음

**wrap-reverse**

wrap의 반대 방향으로 묶음



주 축의 정렬 방법

# justify-content

**flex-start**

Flex Items를 시작점으로 정렬

**flex-end**

Flex Items를 끝점으로 정렬

**center**

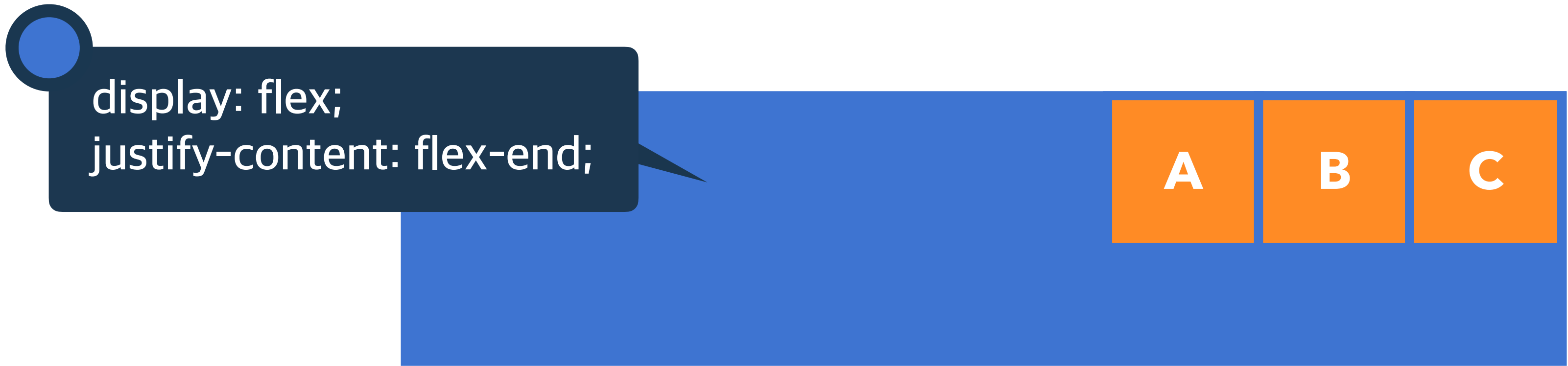
Flex Items를 가운데 정렬

**space-between**

각 Flex Item 사이를 균등하게 정렬

**space-around**

각 Flex Item의 외부 여백을 균등하게 정렬



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

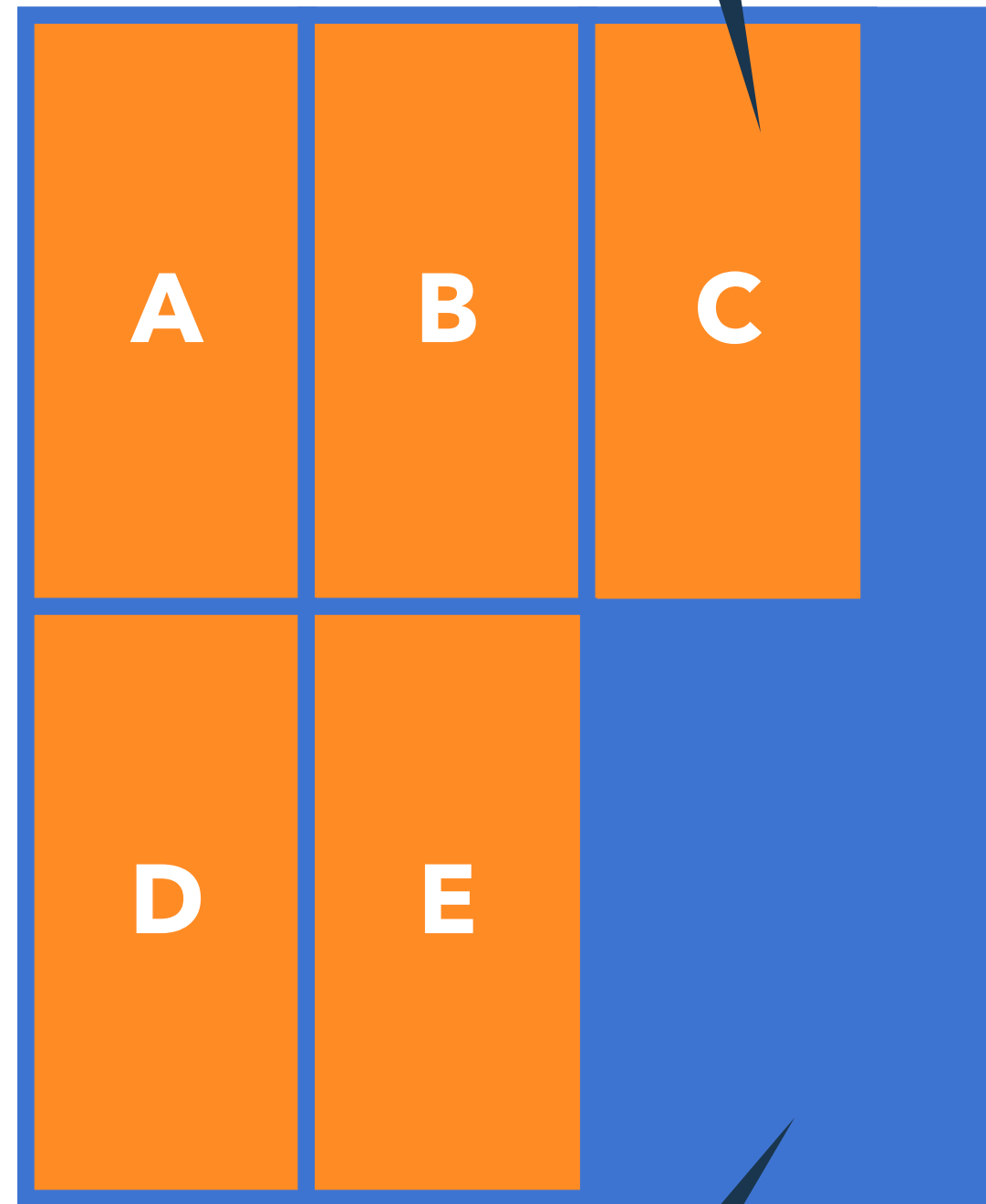
**space-between**

각 Flex Item 사이를 균등하게 정렬

**space-around**

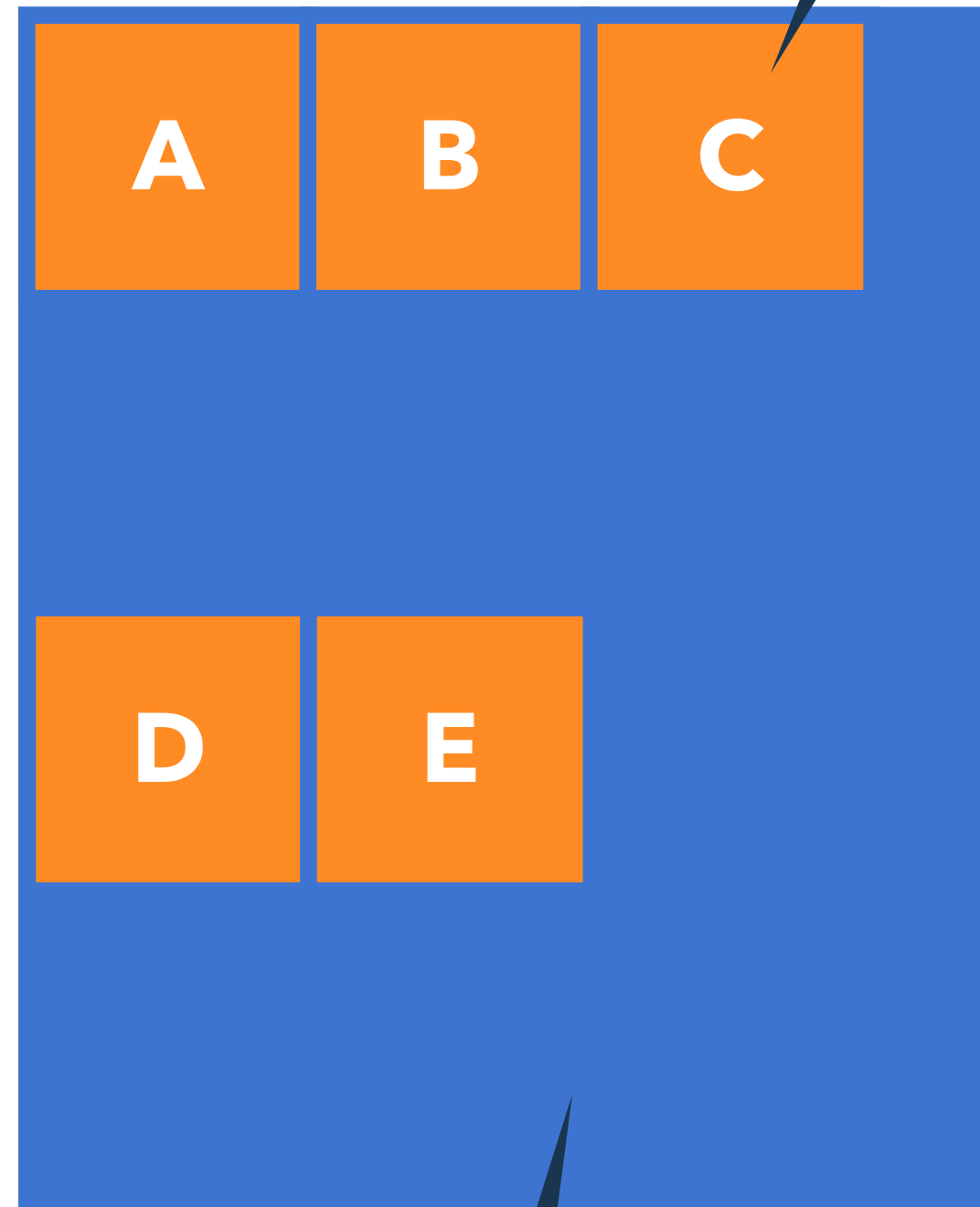
각 Flex Item의 외부 여백을 균등하게 정렬

width: 100px;



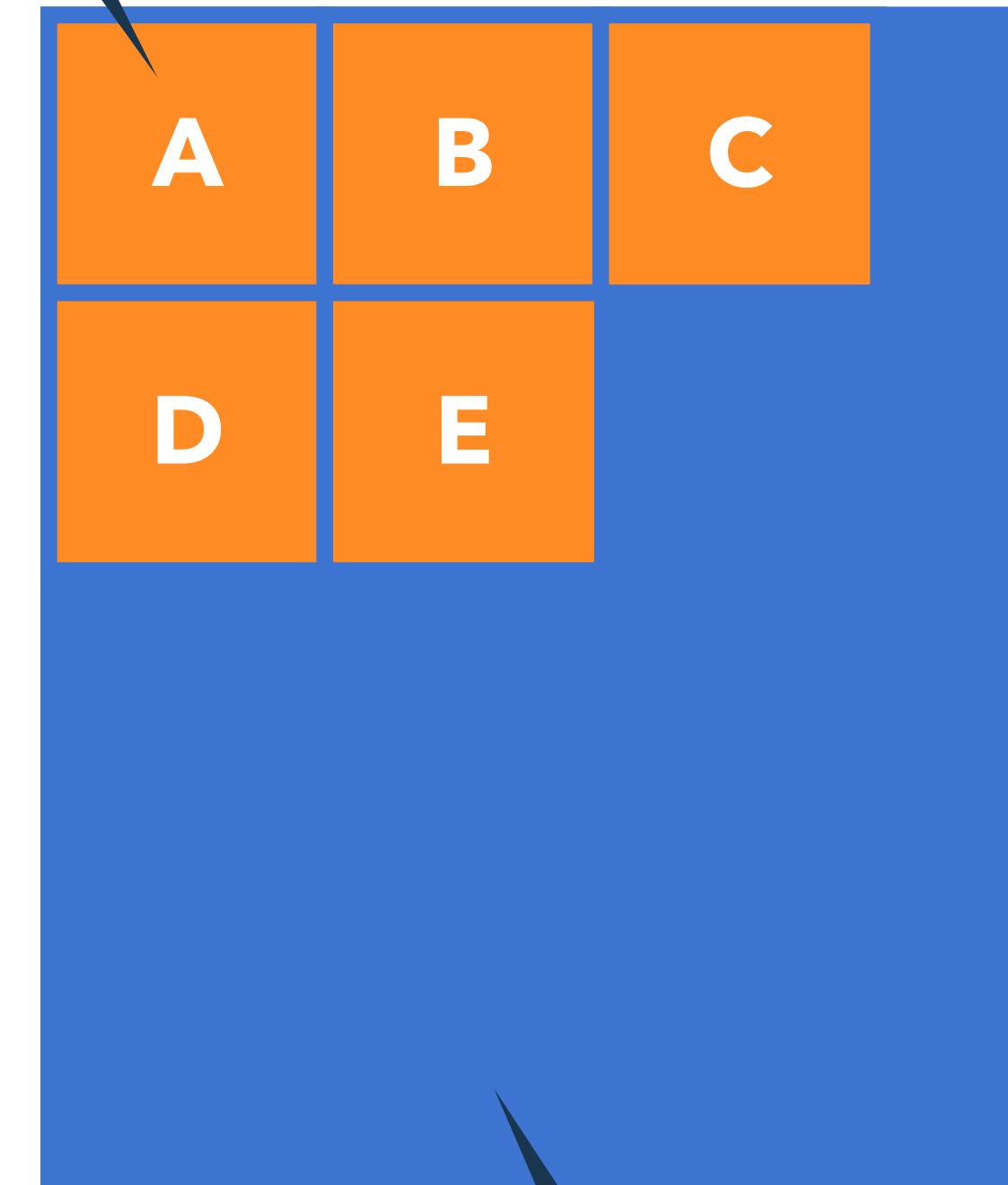
display: flex;  
flex-wrap: wrap;

width: 100px;  
height: 100px;

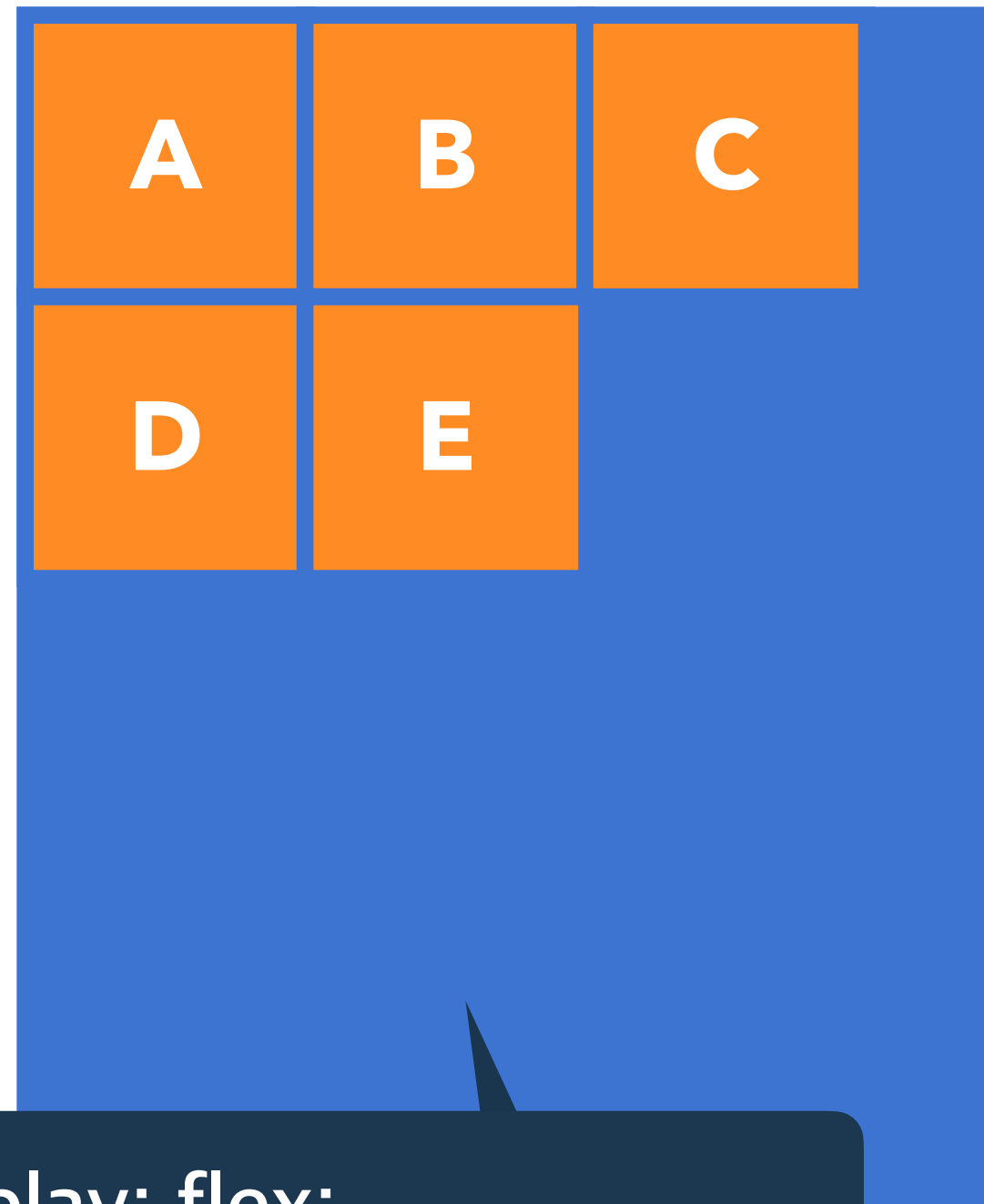


display: flex;  
flex-wrap: wrap;

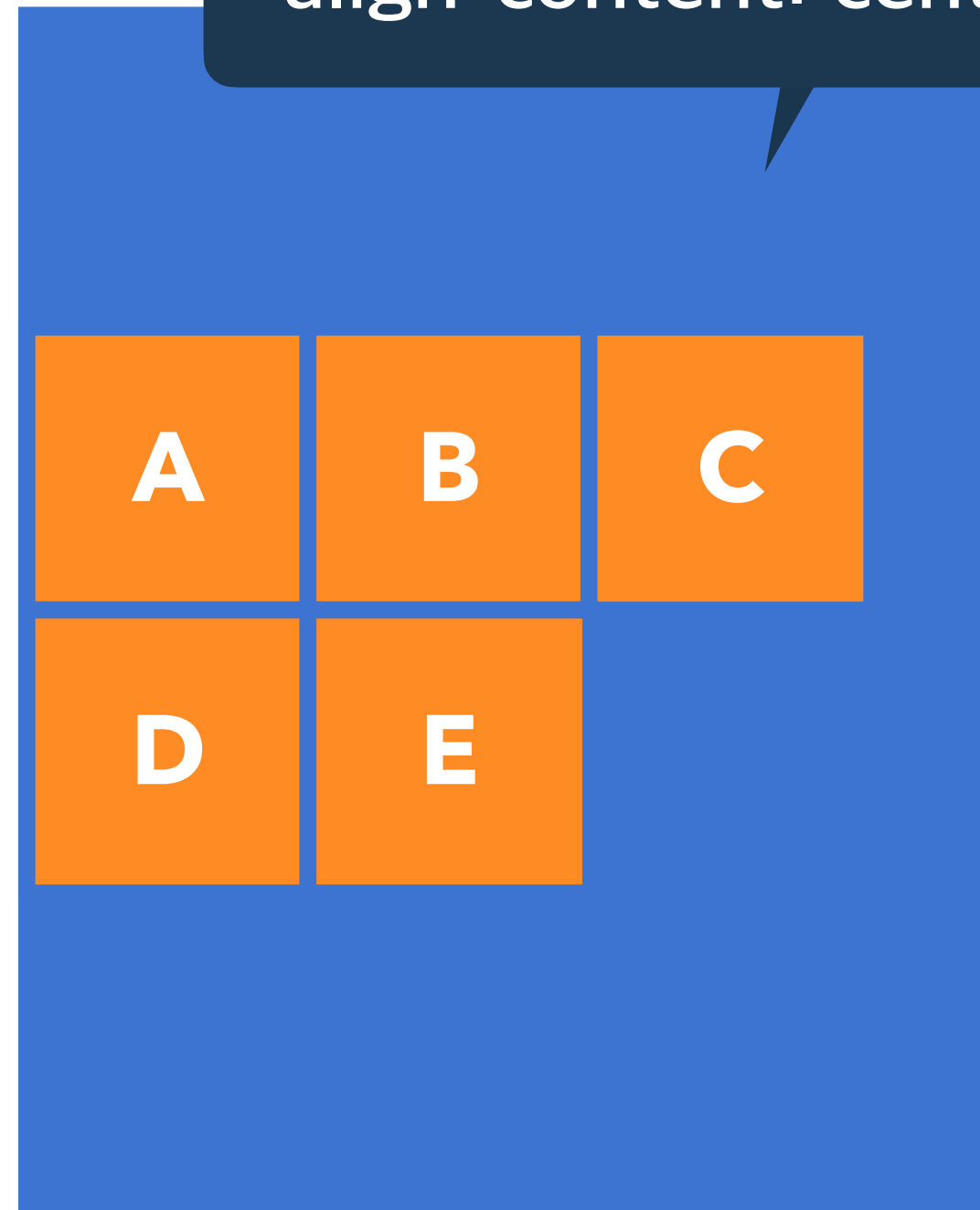
display: flex;  
flex-wrap: wrap;  
align-content: flex-start;



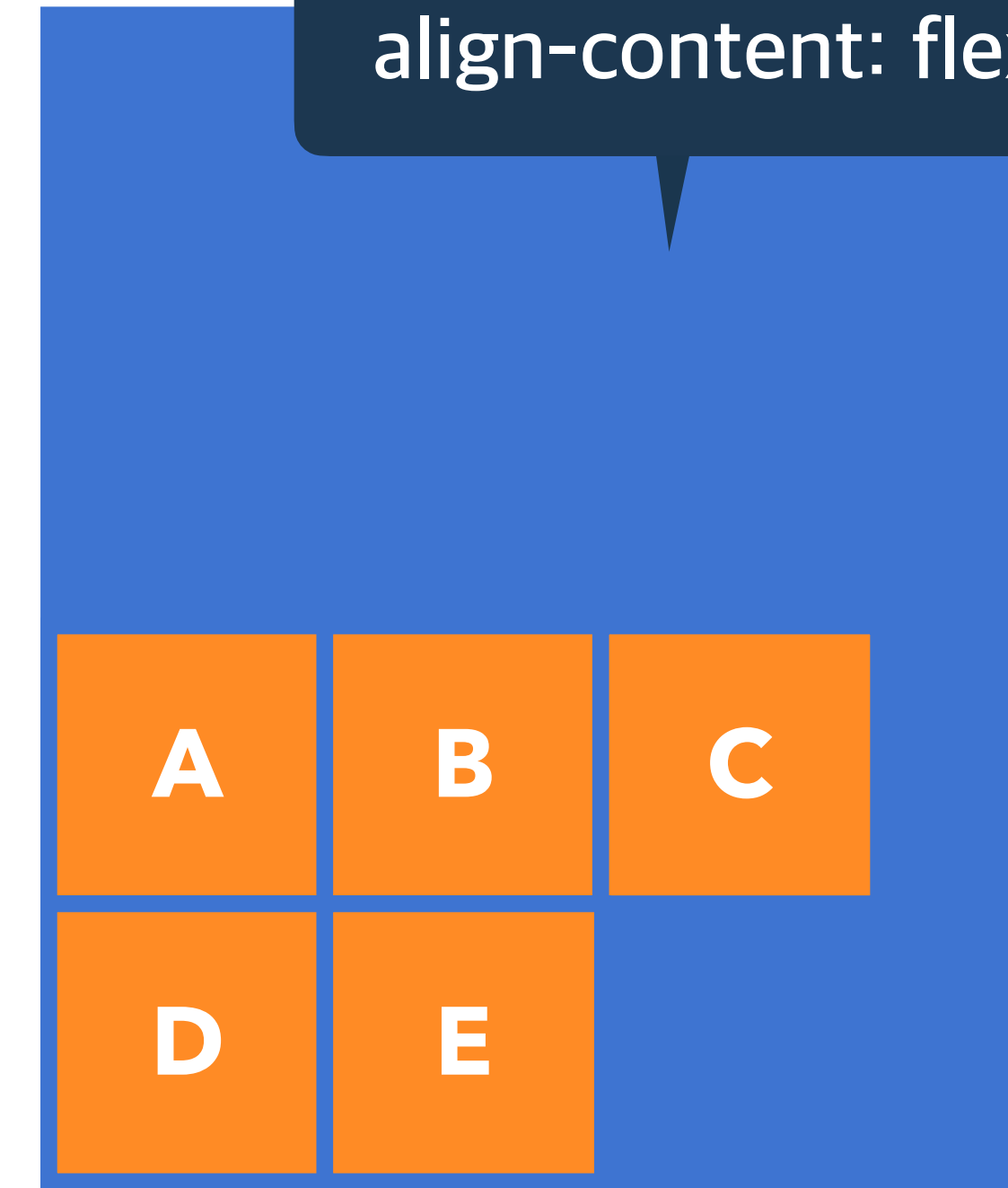




`display: flex;`  
`flex-wrap: wrap;`  
`align-content: flex-start;`



`display: flex;`  
`flex-wrap: wrap;`  
`align-content: center;`



`display: flex;`  
`flex-wrap: wrap;`  
`align-content: flex-end;`

교차 축의 한 줄 정렬 방법

# align-items

**stretch**

Flex Items를 교차 축으로 늘림

**flex-start**

Flex Items를 각 줄의 시작점으로 정렬

**flex-end**

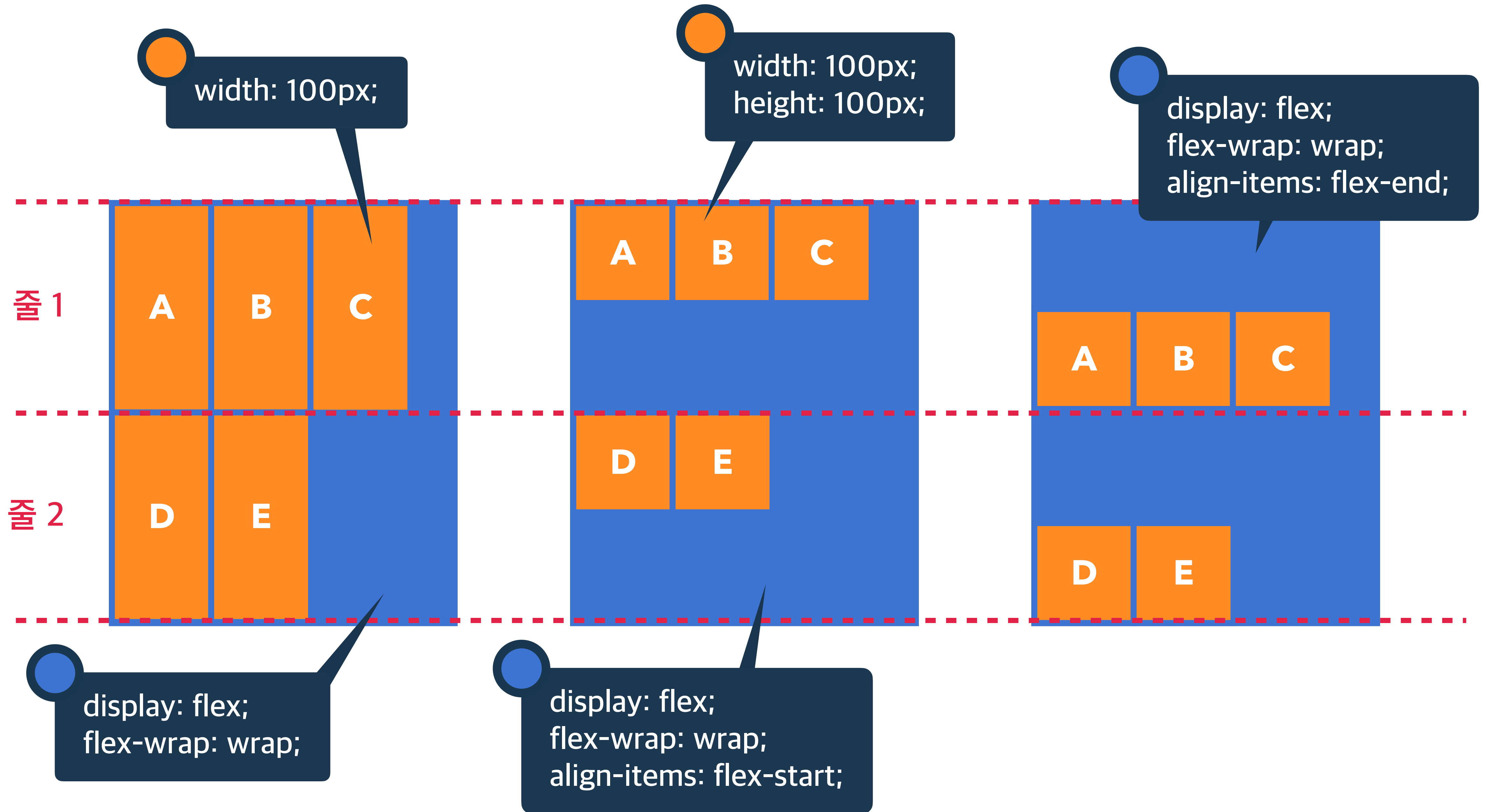
Flex Items를 각 줄의 끝점으로 정렬

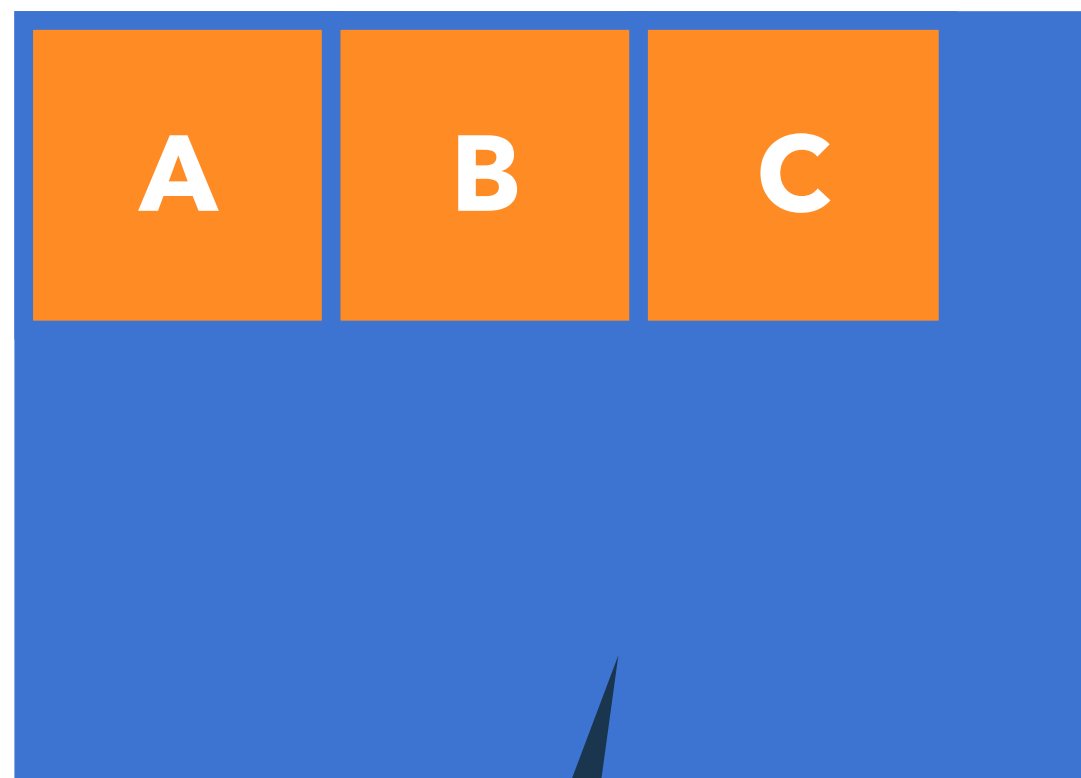
**center**

Flex Items를 각 줄의 가운데 정렬

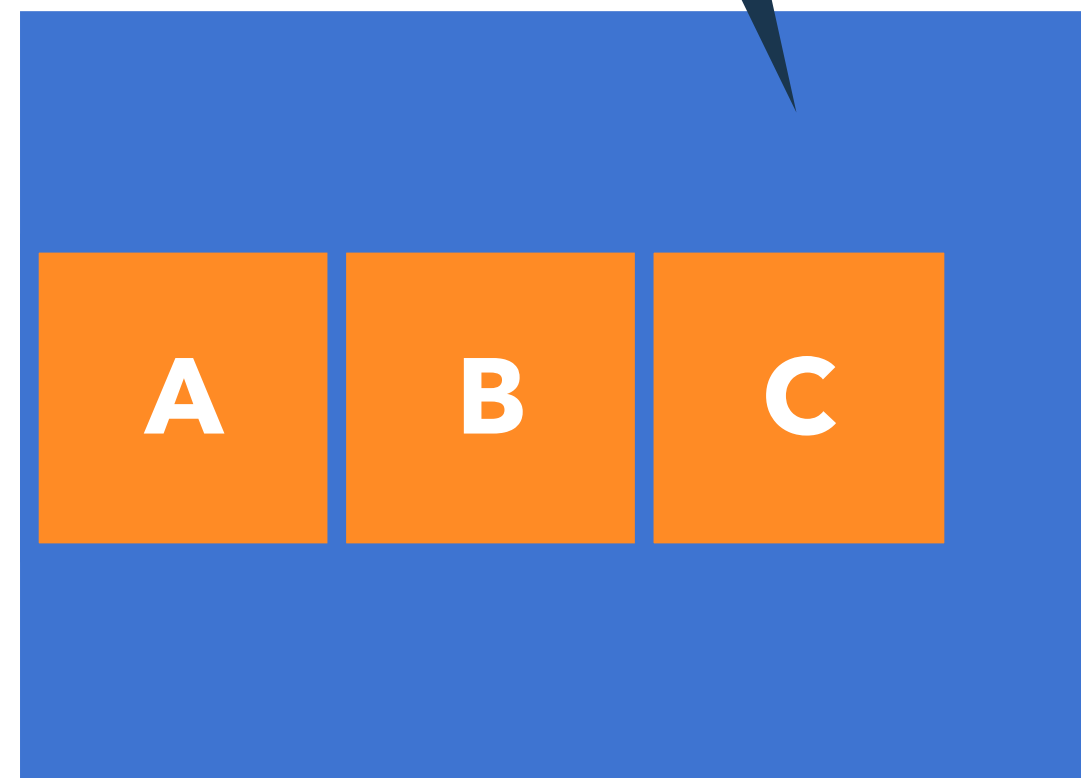
**baseline**

Flex Items를 각 줄의 문자 기준선에 정렬

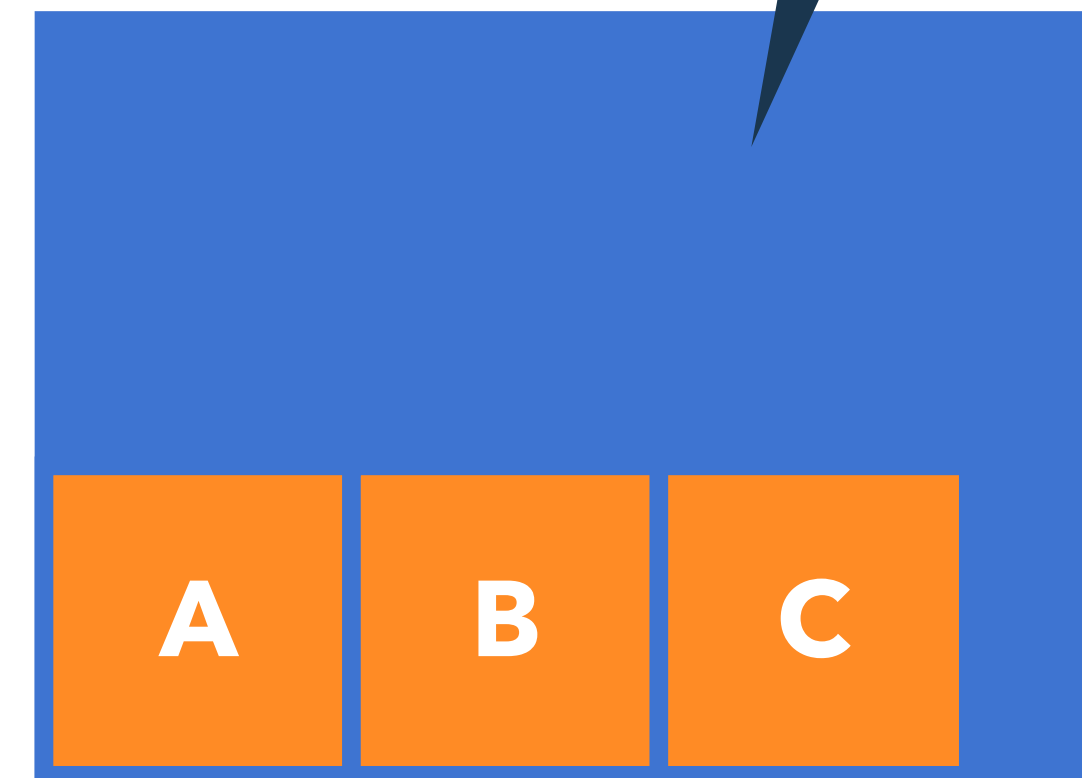




`display: flex;`  
`align-items: flex-start;`



`display: flex;`  
`align-items: center;`



`display: flex;`  
`align-items: flex-end;`

Flex Item의 순서

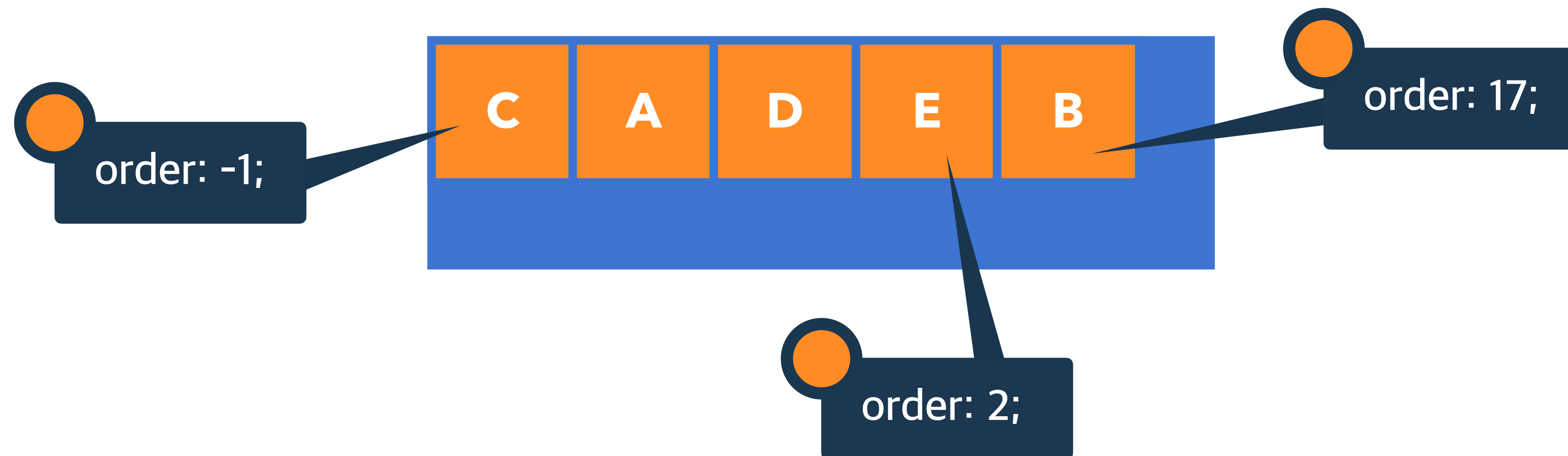
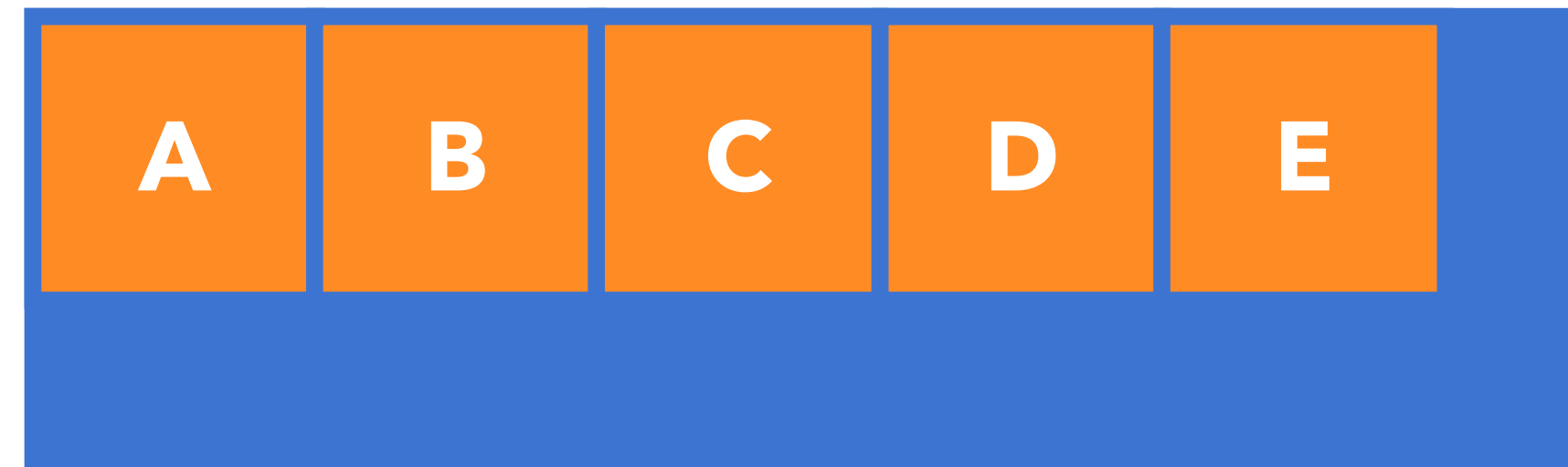
# order

0

순서 없음

숫자

숫자가 작을 수록 먼저



Flex Item의 증가 너비 비율

# flex-grow

0

증가 비율 없음

숫자

증가 비율



flex-grow: 1;



flex-grow: 1;

flex-grow: 2;



Flex Item의 감소 너비 비율

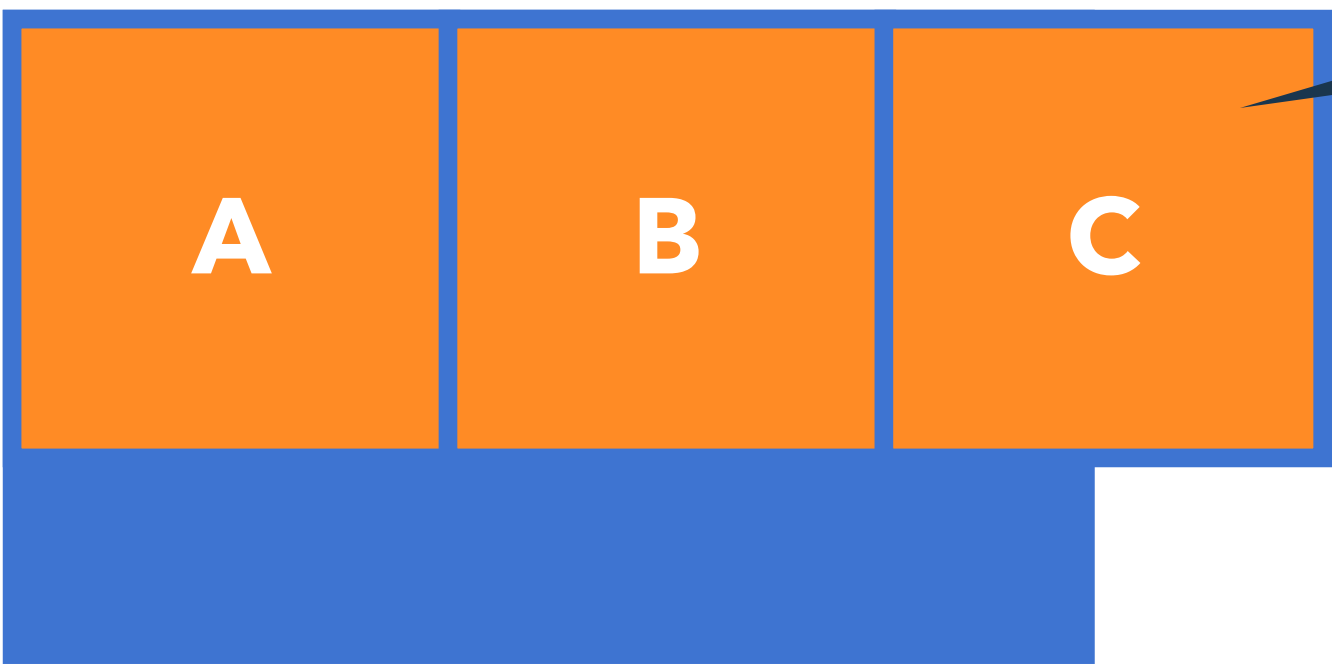
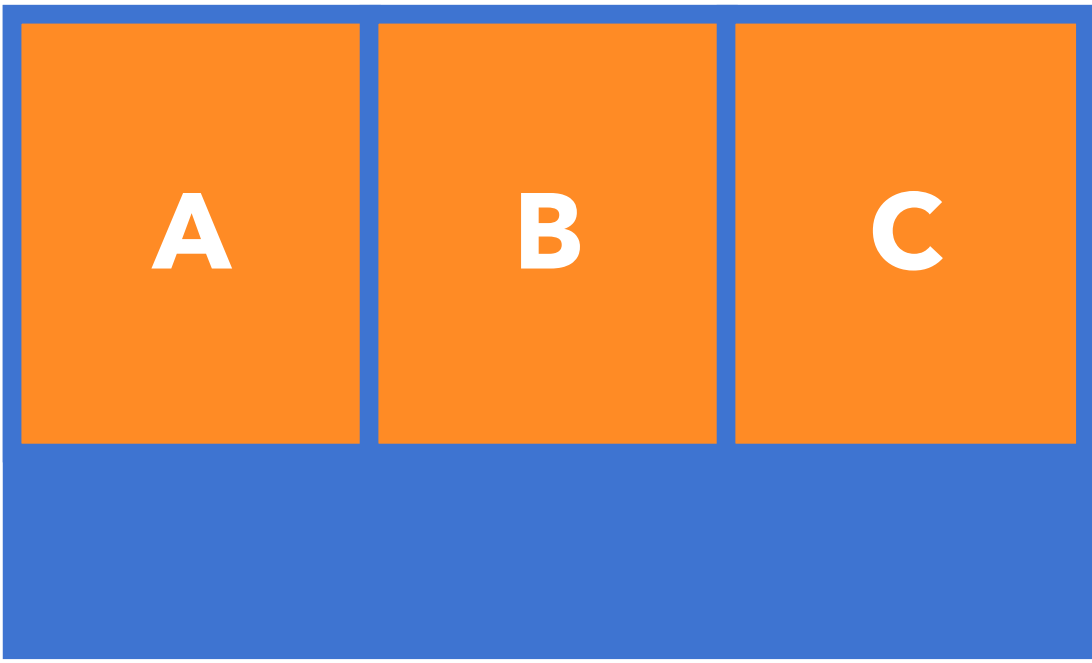
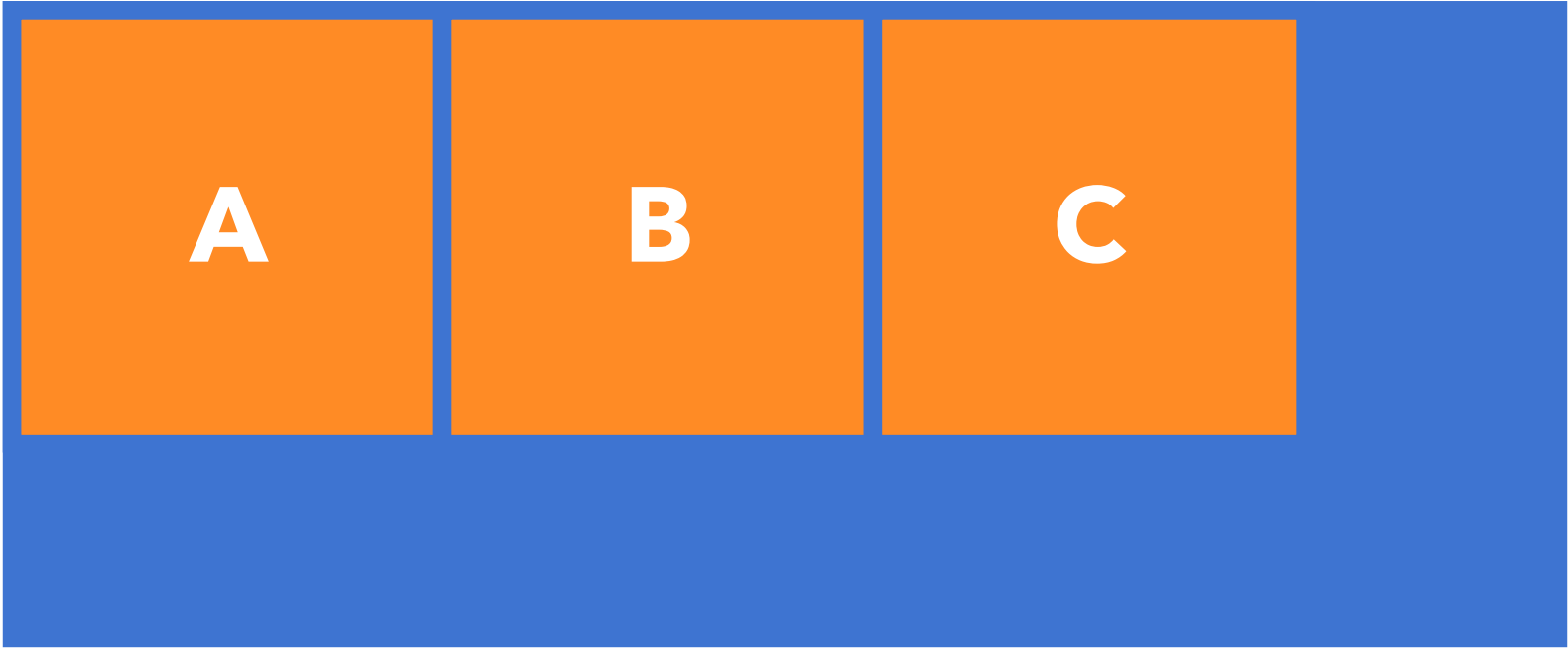
# flex-shrink

1

Flex Container 너비에 따라 감소 비율 적용

숫자

감소 비율



flex-shrink: 0;

Flex Item의 공간 배분 전 기본 너비

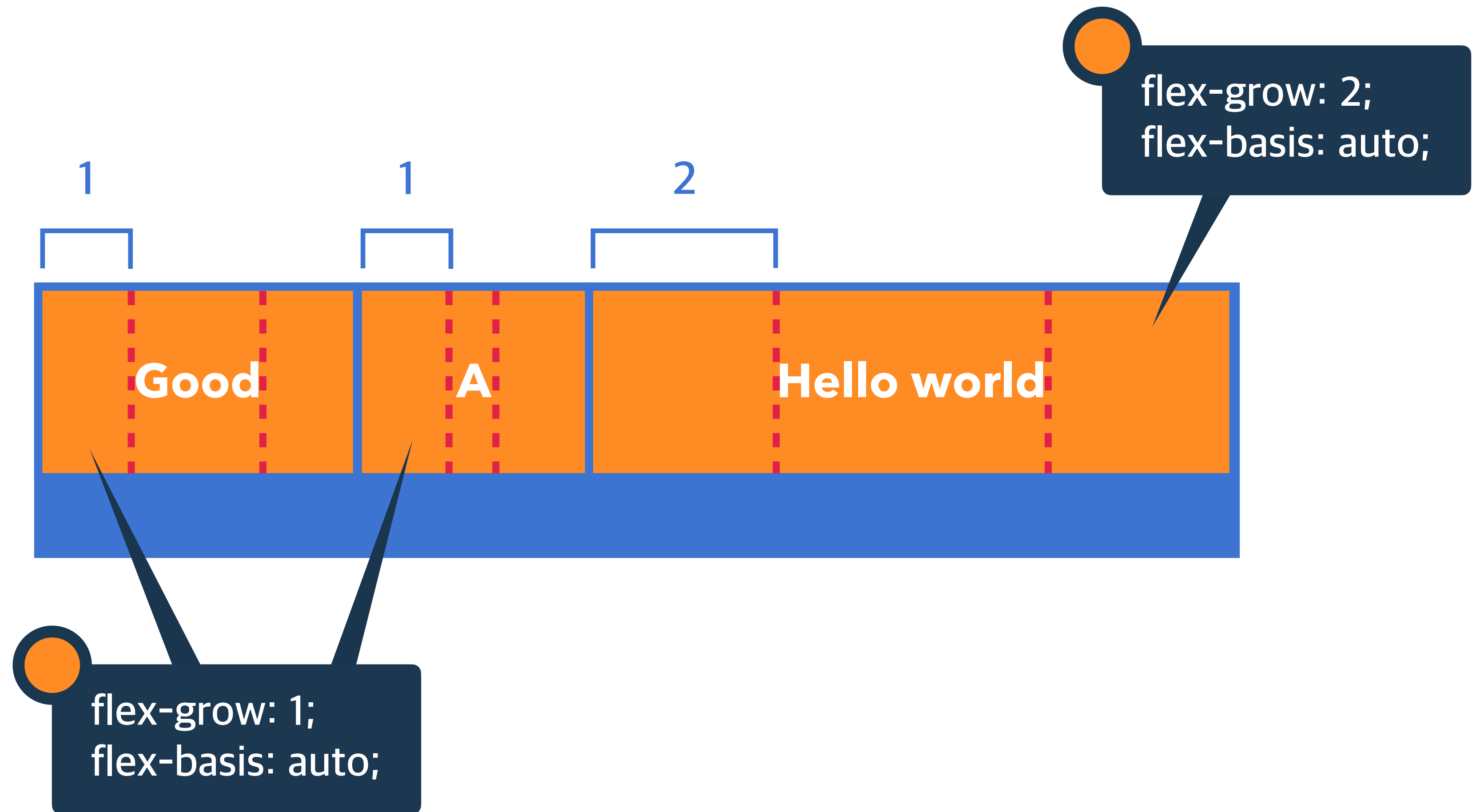
# flex-basis

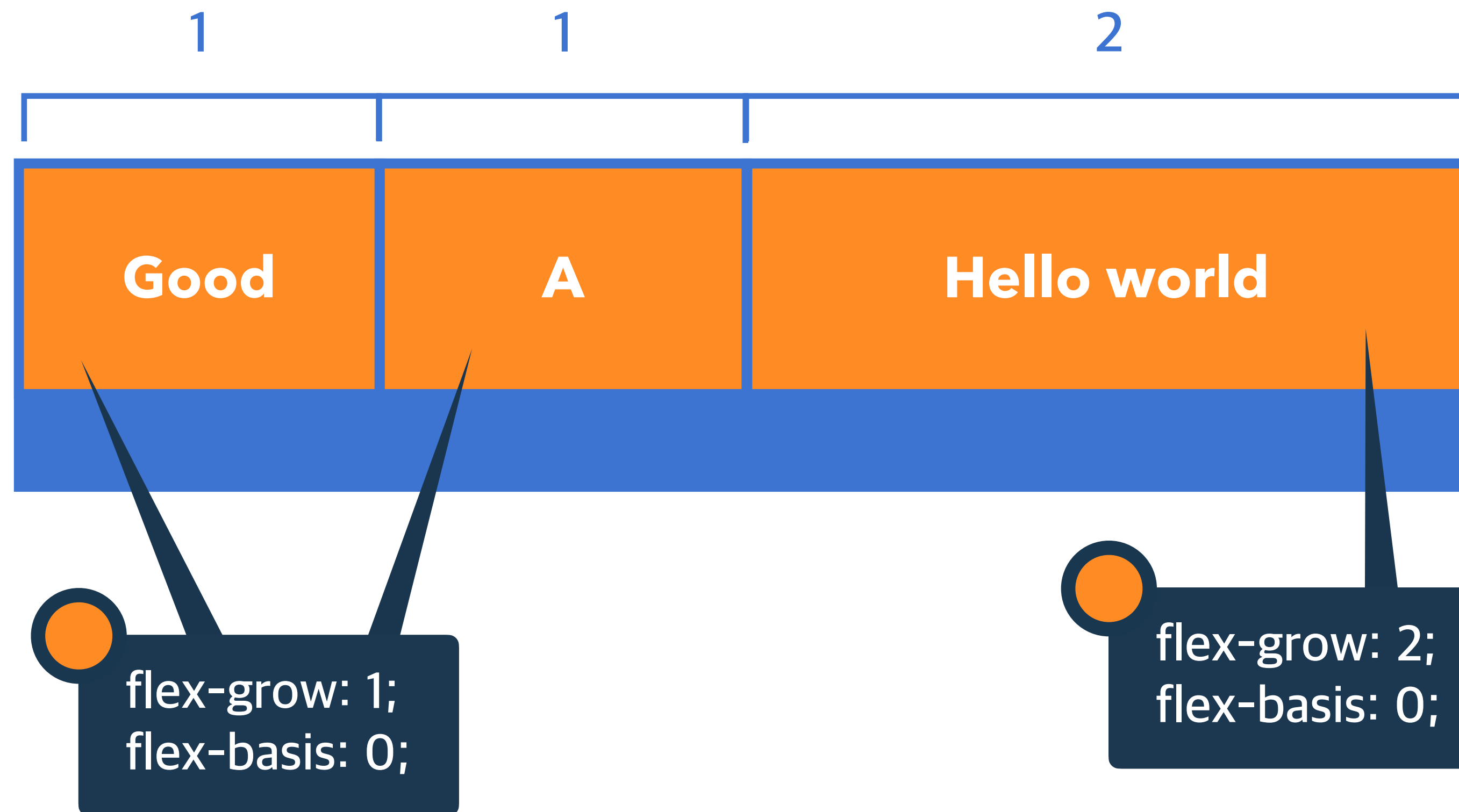
**auto**

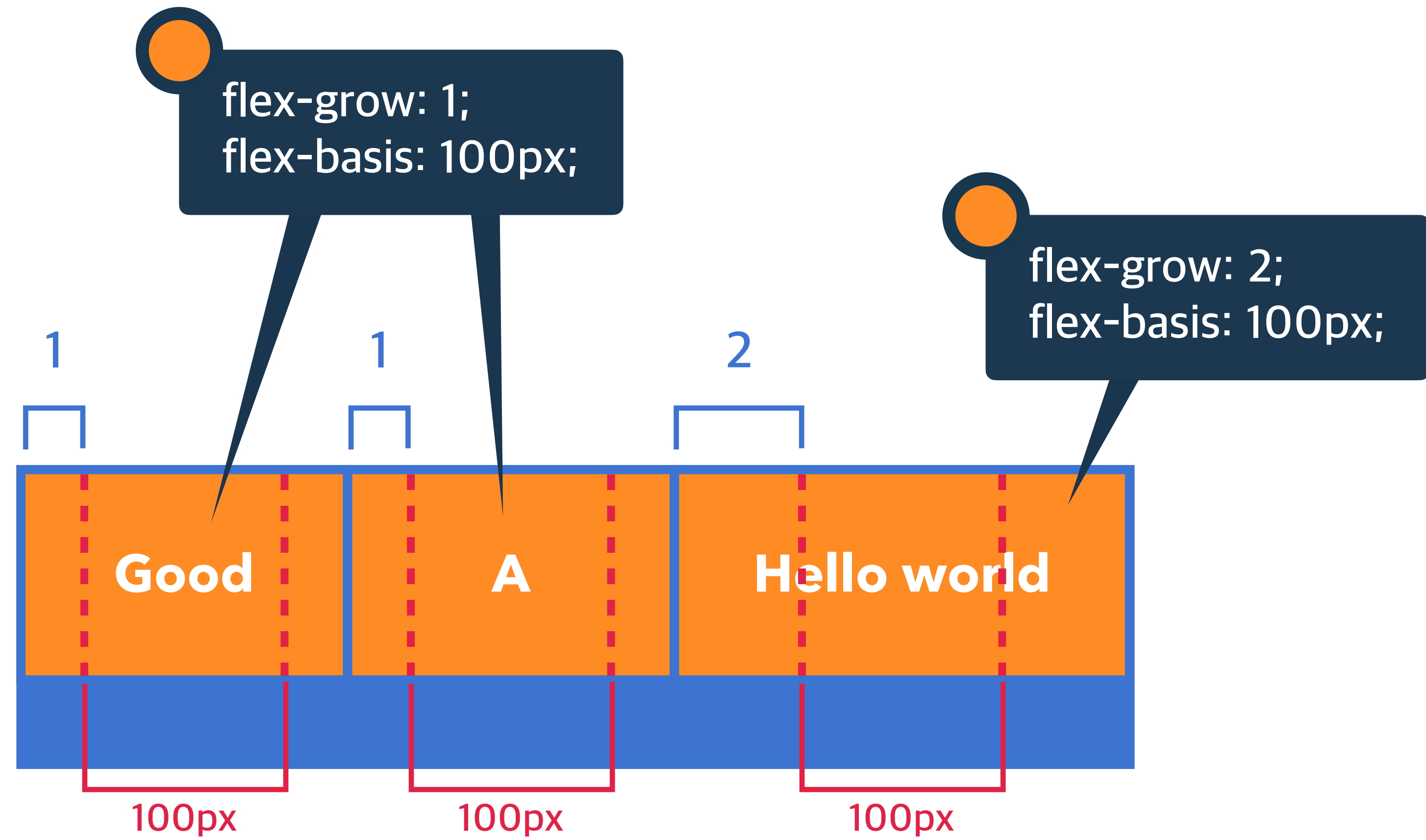
요소의 Content 너비

**단위**

px, em, rem 등 단위로 지정







전화

단축형으로 작성할 때,  
필수 포함 속성!

요소의 전환(시작과 끝) 효과를 지정하는 단축 속성

transition: 속성명 **지속시간** 타이밍함수 대기시간;

transition-property

transition-duration

transition-timing-function

transition-delay



전환 효과를 사용할 속성 이름을 지정

# transition-property

all

모든 속성에 적용

속성이름

전환 효과를 사용할 속성 이름 명시

전환 효과의 지속시간을 지정

# transition-duration

0s

전환 효과 없음

시간

지속시간(s)을 지정

전환 효과의 타이밍(Easing) 함수를 지정

# transition-timing-function



전환 효과가 몇 초 뒤에 시작할지 대기시간을 지정

# transition-delay

0s

대기시간 없음

시간

대기시간(s)을 지정

변화

요소의 변환 효과

**transform: 변환함수1 변환함수2 변환함수3 ... ;**

**transform: 원근법 이동 크기 회전 기울임;**

# 2D 변환 함수

px

**translate(x, y)** 이동(x축, y축)

**translateX(x)** 이동(x축)

**translateY(y)** 이동(y축)

**scale(x, y)** 크기(x축, y축)

**scaleX(x)** 크기(x축)

**scaleY(y)** 크기(y축)

없음(배수)

deg

**rotate(degree)** 회전(각도)

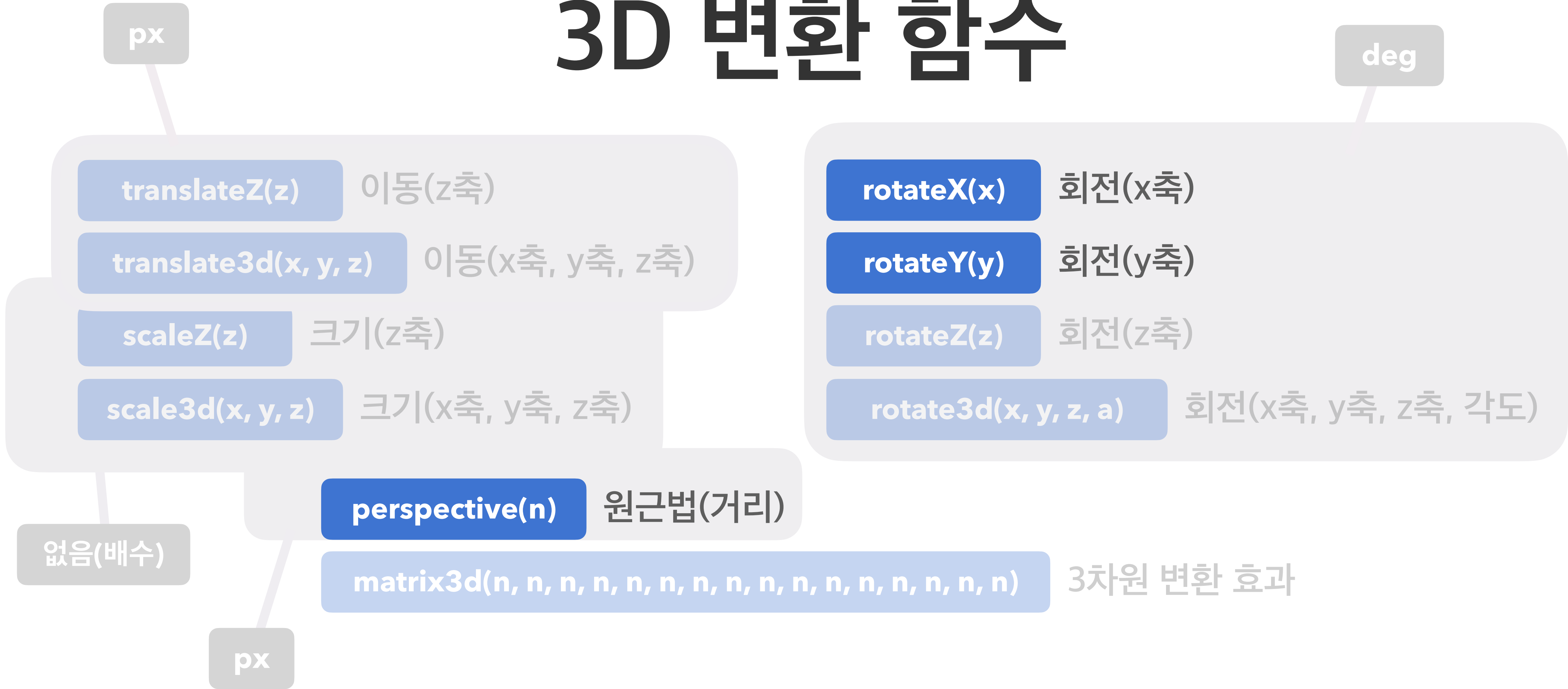
**skew(x, y)** 기울임(x축, y축)

**skewX(x)** 기울임(x축)

**skewY(y)** 기울임(y축)

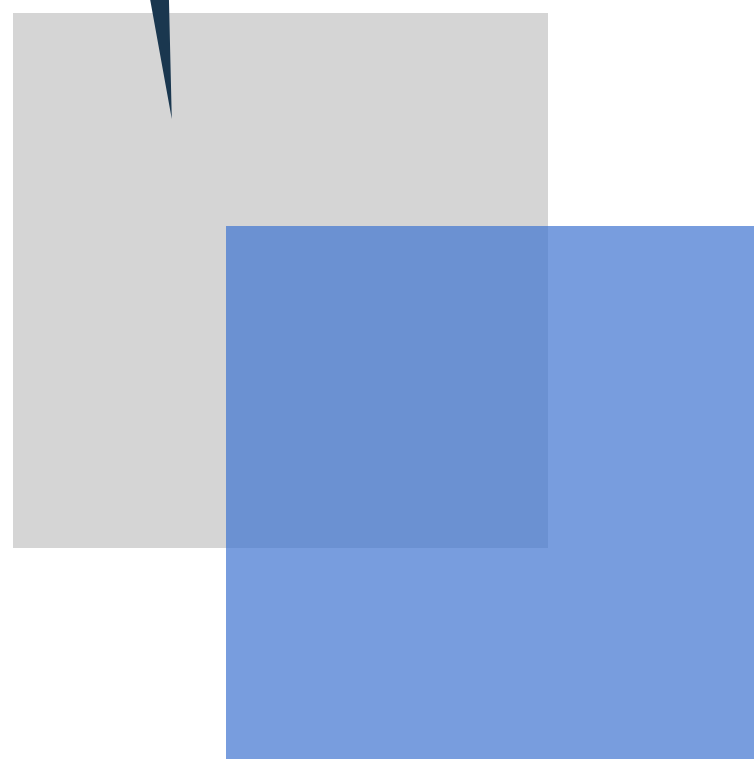
**matrix(n,n,n,n,n,n)** 2차원 변환 효과

# 3D 변환 함수





transform: translate(40px, 40px);



transform: translateY(40px);



transform: translateX(40px);



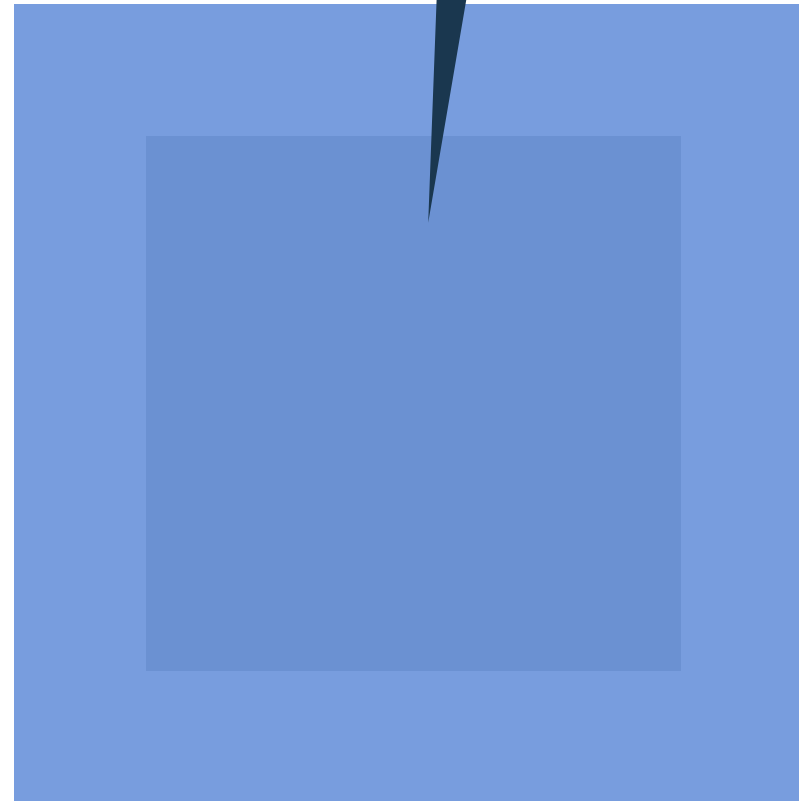
transform: translate(40px, 0);

또는

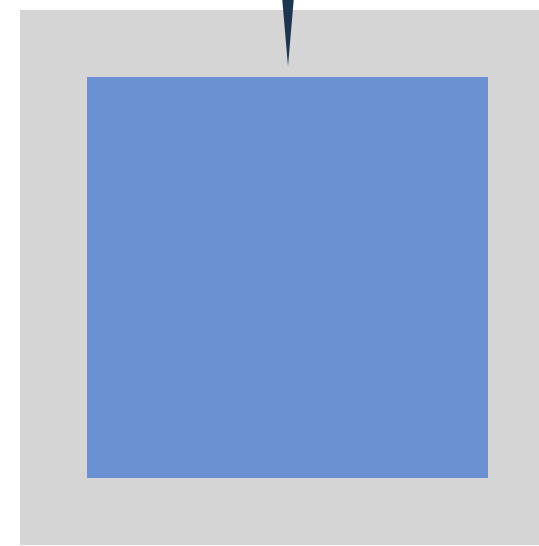
```
transform: scale(1.5, 1.5);
```

또는

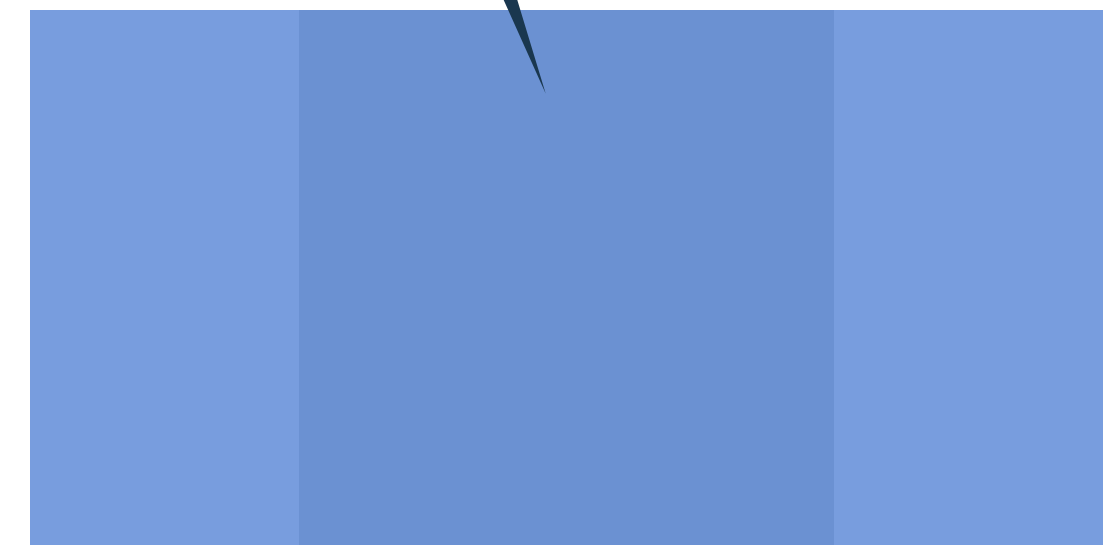
```
transform: scale(1.5);
```



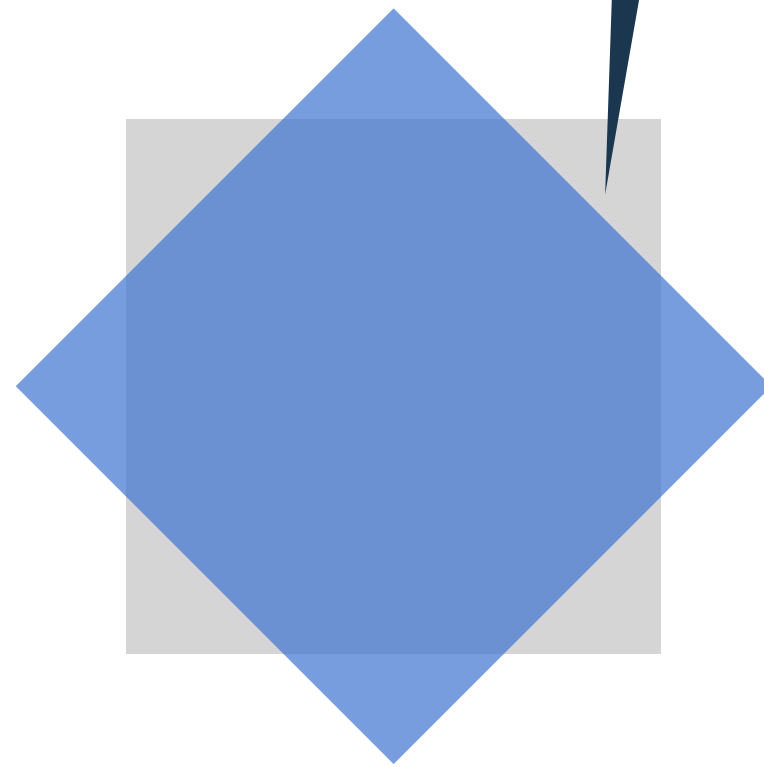
```
transform: scale(0.7);
```



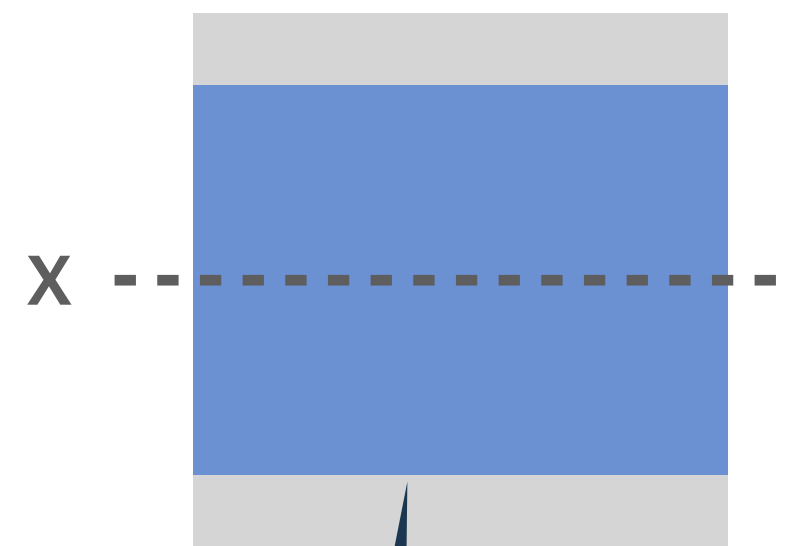
```
transform: scaleX(2);
```



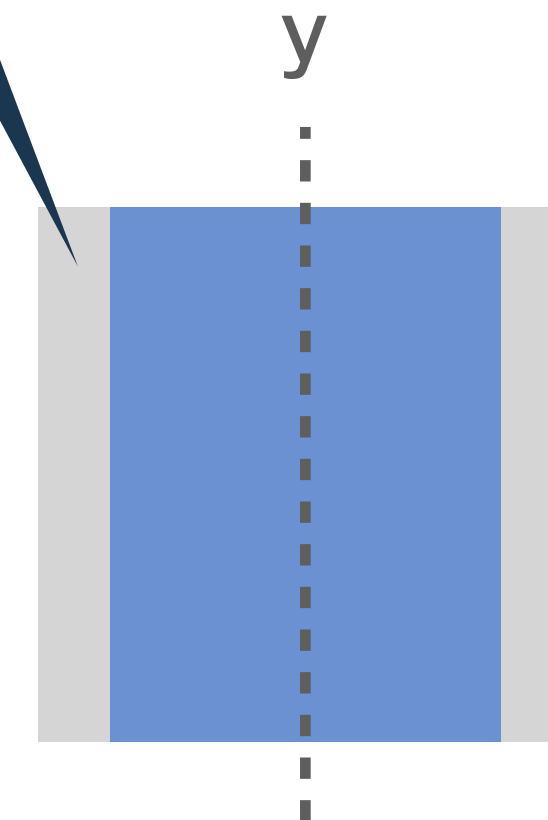
`transform: rotate(45deg);`



`transform: rotateX(45deg);`



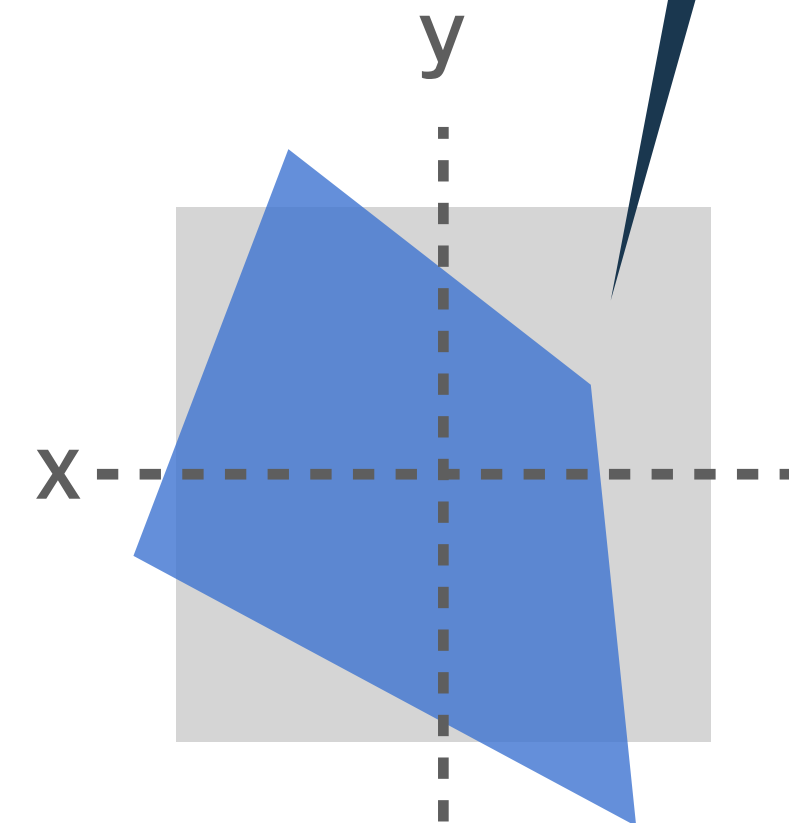
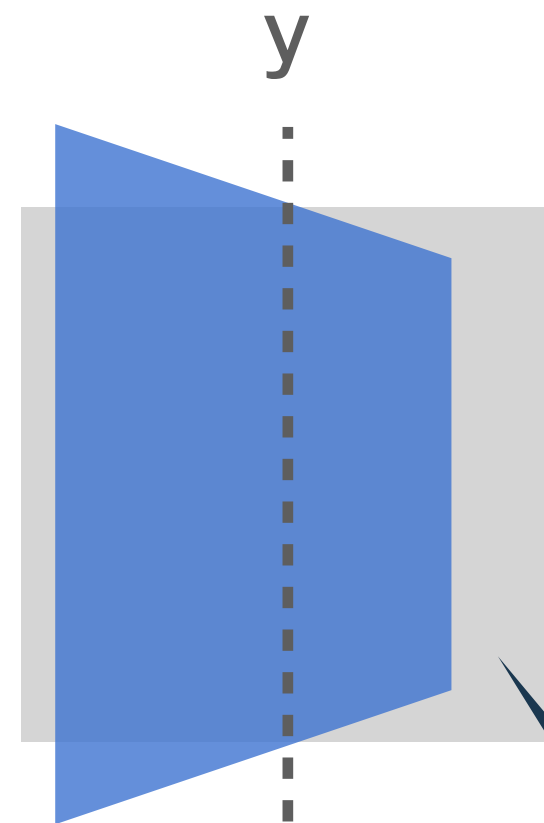
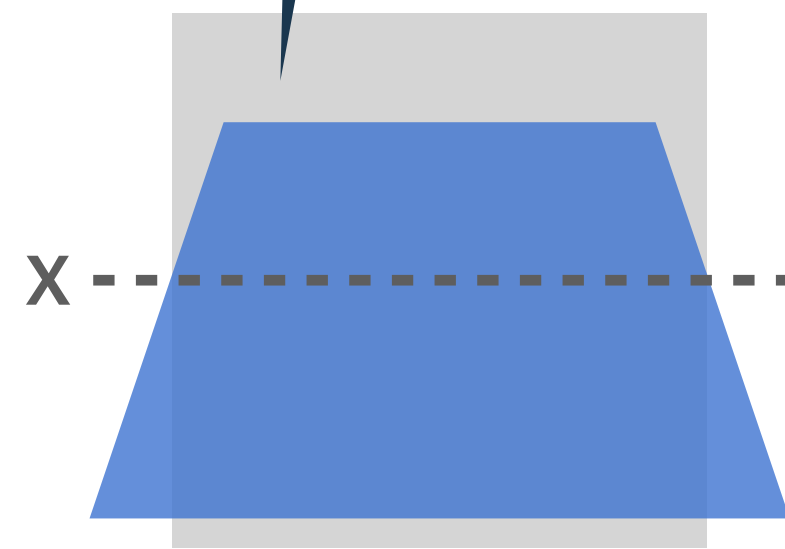
`transform: rotateY(45deg);`



원근법 함수는  
제일 앞에 작성해야 합니다!!

```
transform: perspective(500px) rotateX(45deg) rotateY(45deg);
```

```
transform: perspective(500px) rotateX(45deg);
```



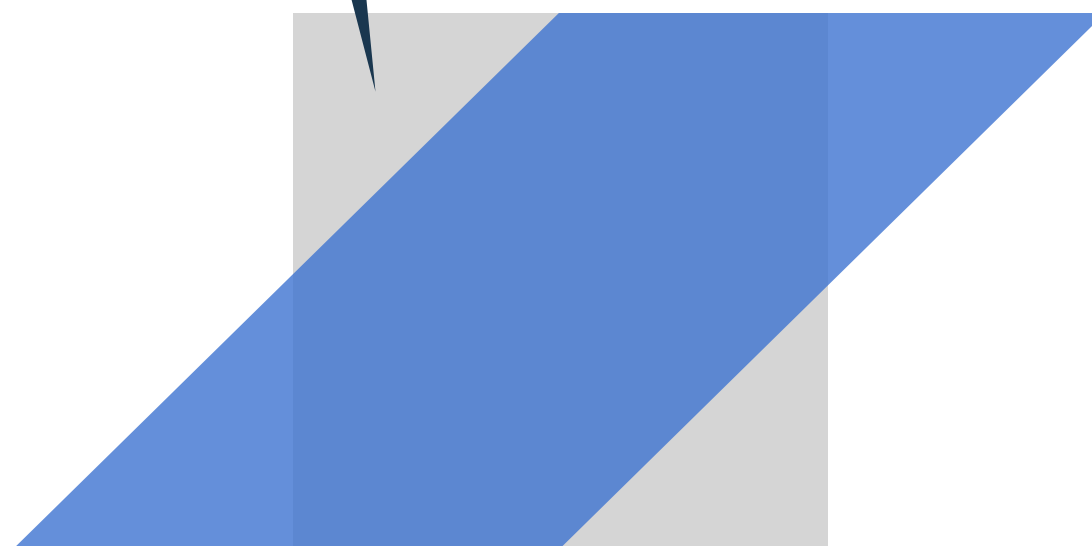
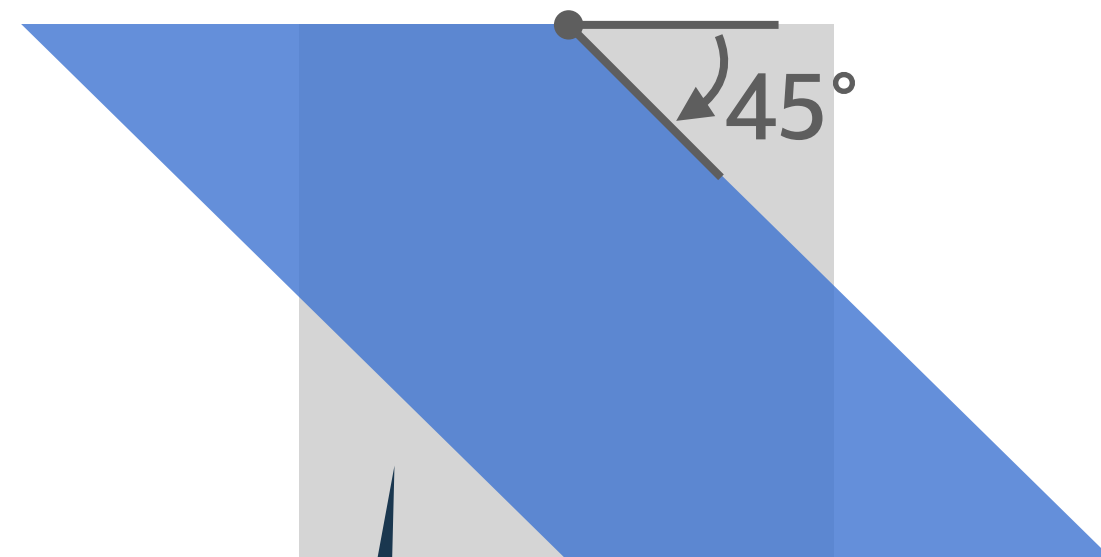
```
transform: perspective(500px) rotateY(45deg);
```

`transform: skewY(45deg);`

`transform: skewX(-45deg);`

`transform: skewX(45deg);`

또는  
`transform: skew(45deg, 0);`



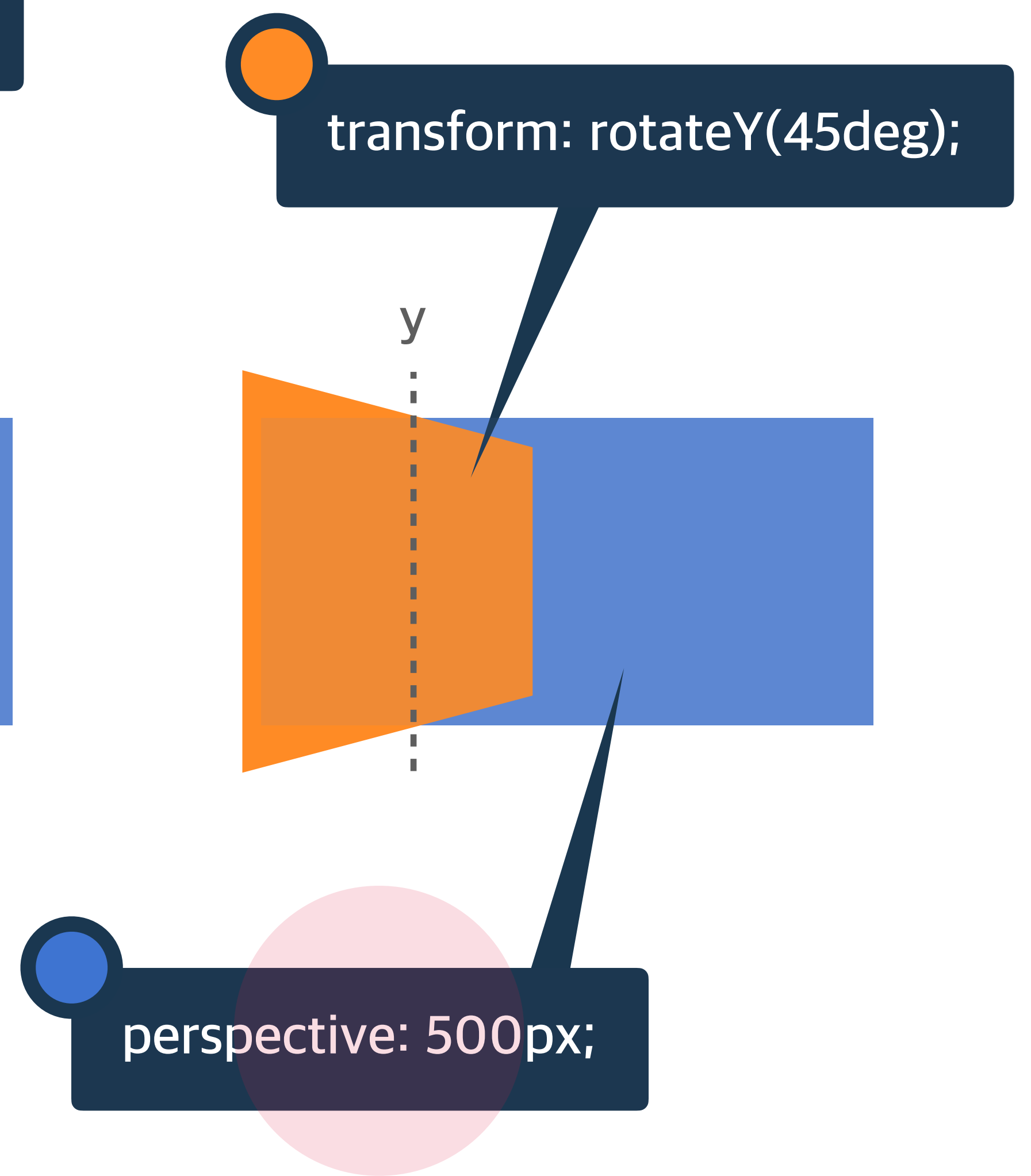
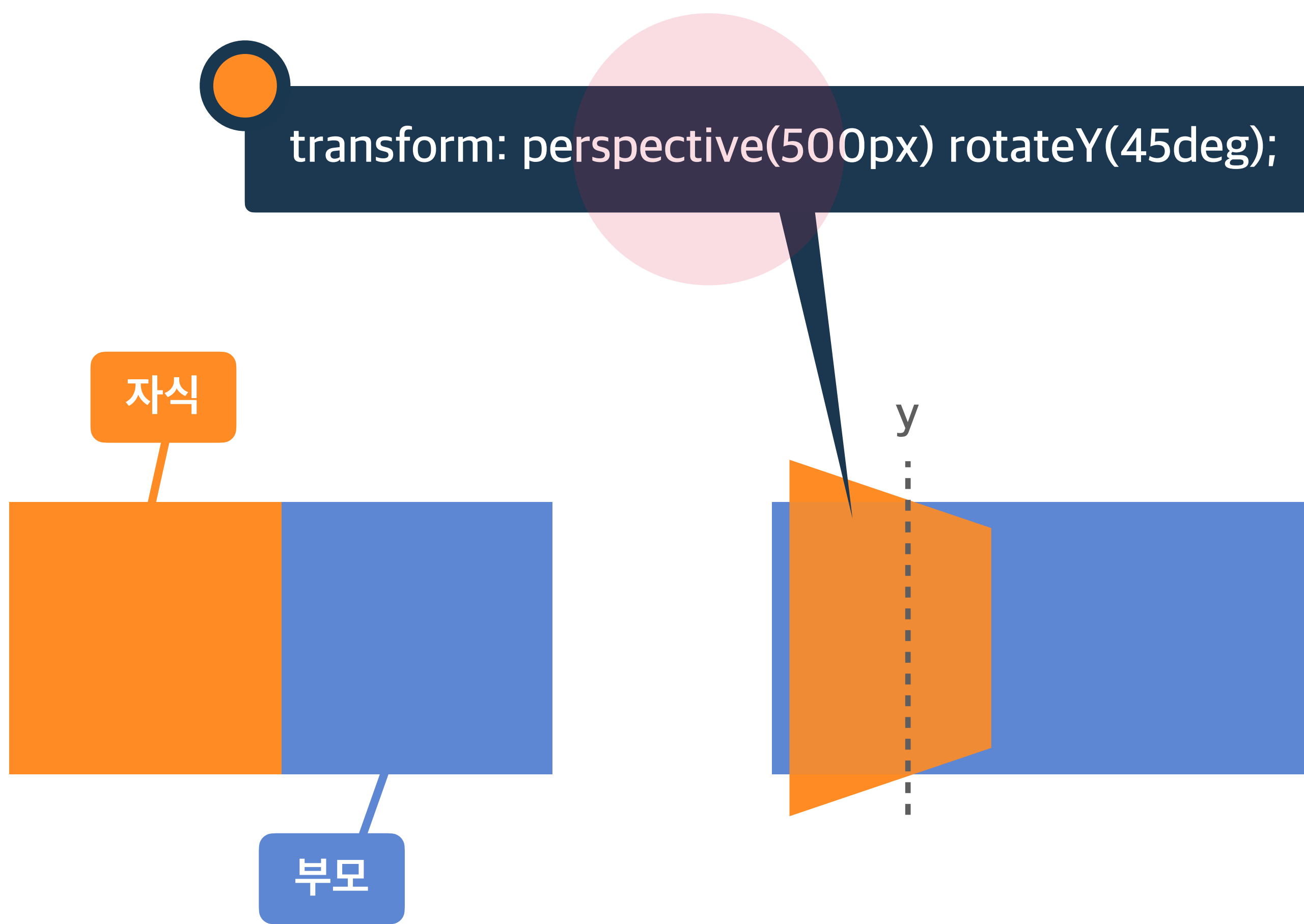
하위 요소를 관찰하는 원근 거리를 지정

# perspective

**단위** px 등 단위로 지정

# perspective 속성과 함수 차이점

속성 / 함수	적용 대상	기준점 설정
<code>perspective: 600px;</code>	관찰 대상의 부모	<code>perspective-origin</code>
<code>transform: perspective(600px)</code>	관찰 대상	<code>transform-origin</code>





3D 변환으로 회전된 요소의 뒷면 숨김 여부

# backface-visibility

**visible**

뒷면 보임

**hidden**

뒷면 숨김

