

# 13. CSS flex

---

IE를 지원하지 않아도 된다면 가장 쓸 모 있는.



# CSS flex

플렉스는  
박스의

**크기, 방향, 순서, 정렬, 간격을**

제어하는

새로운 박스 모델.

<https://www.w3.org/TR/css-flexbox-1/>

# CSS flex 'IE' bugs 🤖

1. Minimum content sizing of flex items not honored
2. Column flex items set to align-items: center overflow their container
3. **min-height** on a flex container won't apply to its flex items ⚠️
4. flex shorthand declarations with unitless **flex-basis values are ignored** ⚠️
5. Column flex items don't always preserve intrinsic aspect ratios
6. The **default flex value** has changed ⚠️
7. flex-basis doesn't account for **box-sizing: border-box** ⚠️
8. flex-basis doesn't support **calc()** ⚠️
9. Some HTML elements can't be flex containers
10. align-items: baseline doesn't work with nested flex containers
11. Min and max size declarations are ignored when wrapping flex items
12. **Inline elements** are not treated as flex-items ⚠️
13. Importance is ignored on flex-basis when using flex shorthand
14. Shrink-to-fit containers with flex-flow: column wrap do not contain their items
15. Column flex items **ignore margin: auto** on the cross axis ⚠️
16. flex-basis cannot be animated
17. Flex items are not correctly justified when **max-width** is used ⚠️

Underline = IE bug(21/17), ⚠️ = 치명적(8/17). flexbug - <https://t.ly/cIJU>

“

# Simple Demo



# 아이템 크기 자동 분배 (flex-grow/-shrink/-basis)

The diagram illustrates the concept of flex-grow/-shrink/-basis using two examples of a horizontal menu. A blue arrow at the top points to the right, indicating the direction of growth.

**Example 1: Default Flex-grow**

- UI Element:** A horizontal menu with five items: 전체, 분류, 유아식, 커피, 우유/두유. The entire menu is highlighted with a blue background.
- HTML Structure:**

```
<h2 class="skip">제품 카테고리</h2>
<ul class="order-tab-main">
  <li class="on">...</li>
  <li>...</li>
  <li>...</li>
  <li>...</li>
  <li>...</li>
</ul>
```
- CSS:**

```
.order-tab-main {
  display: -webkit-box;
  display: flex;
  height: 44px;
}
```

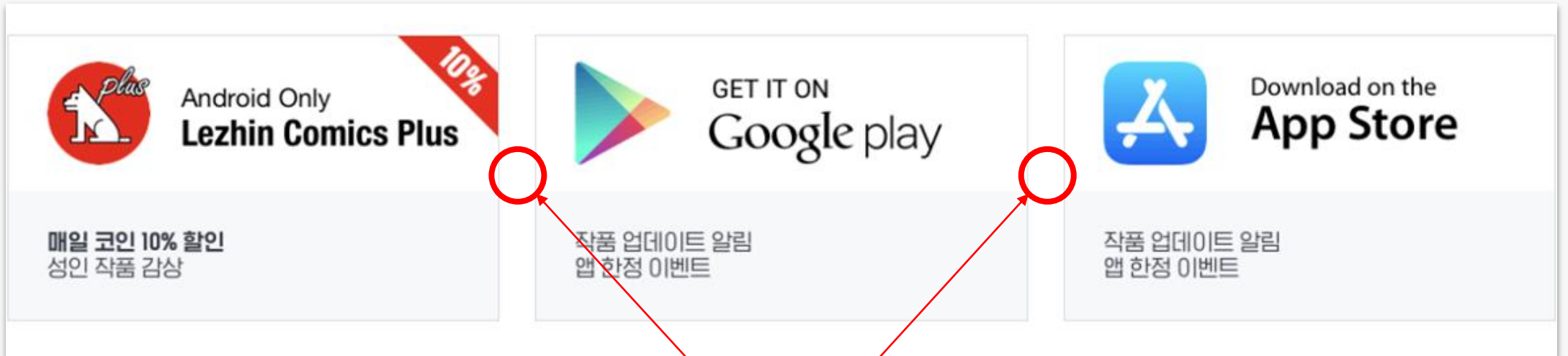
**Example 2: Explicit Flex-grow**

- UI Element:** The same horizontal menu, but the '분류' item is highlighted with a blue background, while the other items have a white background.
- HTML Structure:**

```
<h2 class="skip">제품 카테고리</h2>
<ul class="order-tab-main">
  <li class="on">...</li>
  <li>...</li>
  <li>...</li>
  <li>...</li>
  <li>...</li>
</ul>
```
- CSS:**

```
.order-tab-main>li {
  -webkit-box-flex: 1;
  flex-grow: 1;
}
```

# 진행 측정렬 (공간 자동 분배, justify-content)



Auto margin(X) => Free space(O)

# 진행 속 정렬 (공간 자동 분배, justify-content)

NOW오더 재설정  
a | 296 × 68

Wi-Fi 연결 정보 관리  
SKP-OA

NOW오더 서비스 해지  
남양 NOW오더

결제정보 / 배송지 설정

>

>

>

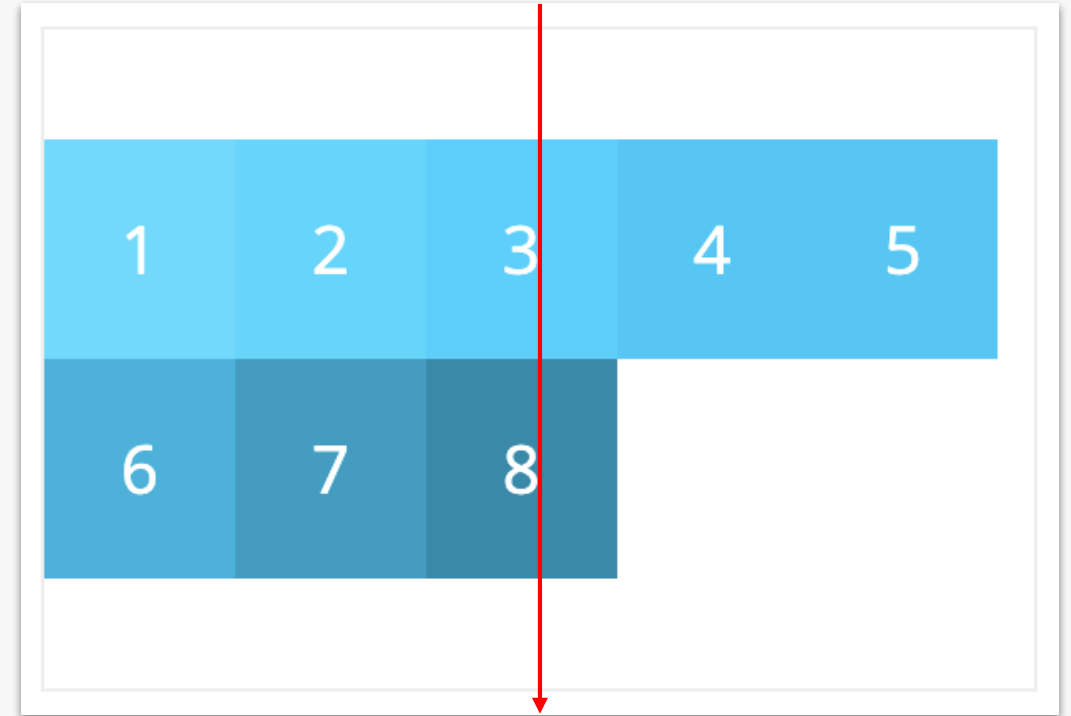
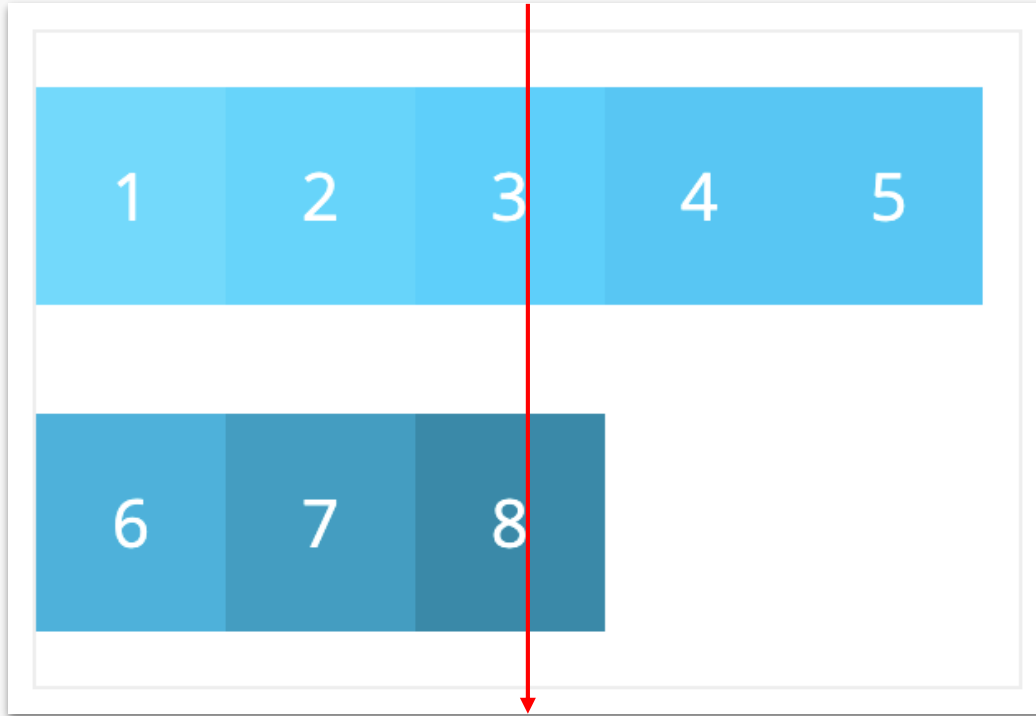
>

```
▼<section class="setup_info">
  <h2 class="skip">결제정보 관리
  </h2>
  ▼<ul>
    ▶<li>...</li>
    ▼<li>
      ... ▼<a href="#"> == $0
        "Wi-Fi 연결 정보 관리 "
        <output>SKP-OA
        </output>
        ::after
        </a>
      </li>
    ▶<li>...</li>
    ▶<li>...</li>
```

```
.setup_info a {
  display: -webkit-box;
  display: flex;
  -webkit-box-orient: vertical;
  flex-direction: column;
  -webkit-box-pack: center;
  justify-content: center;
  position: relative;
  font-size: 16px;
  border-top: 1px solid #ddd;
  height: 68px;
  color: #333;
}

body * {
  overflow-wrap: break-word;
```

# 교차 축 정렬 (공간 자동 분배, align-items/-self/-content)



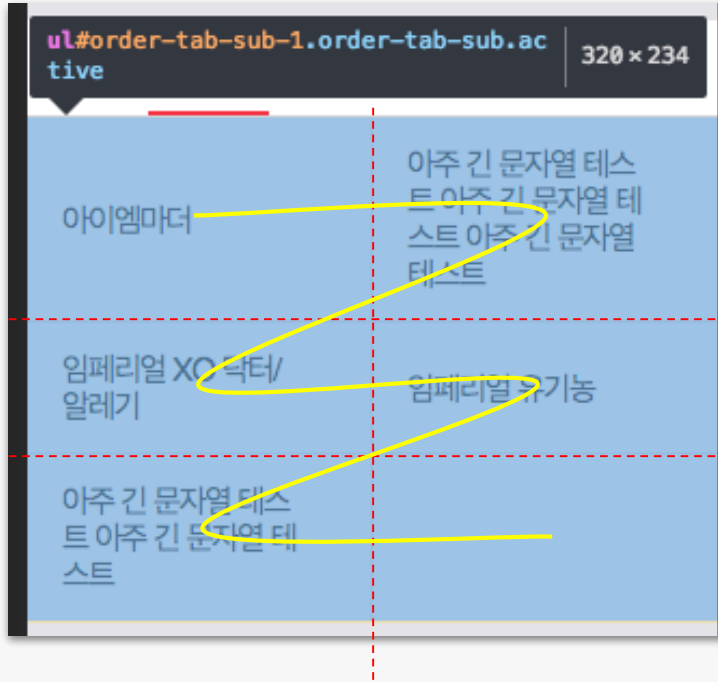


## 배치 순서 변경 (order)



```
▼<section class="curationHot">
  ▼<h2 class="curationHot__heading">
    <i class="curationHot__headingIcon">HOT</i>
    <span class="curationHot__headingText">100만명이 선택한 만화</span> == $0
    <span class="curationHot__headingDesc">첫번째로 보셔야 할 작품~!</span>
  </h2>
```

# 감싸기 (flex-direction/-wrap/-flow)



ul#order-tab-sub-1.order-tab-sub.active 320 x 234

아이엠마더

아주 긴 문자열 테스트 아주 긴 문자열 테스트 아주 긴 문자열 테스트

임페리얼 XQ 닥터/알레기

임페리얼 유가농

아주 긴 문자열 테스트 아주 긴 문자열 테스트

```
</nav>
...
<ul class="order-tab-sub active" id="order-tab-sub-1"> =
  ><li>...</li> =
  ><li>...</li> $0
  ><li>...</li>
  ><li>...</li>
  ><li>...</li>
  </ul>
  ><section class="order-list">...
  </section>
</main>
<!-- footer -->
<script src="/MW/html/inc/cm_footer.js"></script>
<footer>...</footer>
<!-- //footer -->
```

```
.order-tab-sub.active {
  display: block;
  display: -webkit-box;
  display: flex;
  -webkit-box-lines: multiple;
  flex-wrap: wrap;
}

.order-tab-sub {
  display: none;
  margin-top: -8px;
  background: #fafafa;
  border-bottom: 1px solid #d1d1d6;
  font-size: 0;
}

body * {
```

“

# Flex 용어

flex container

flex item

free space

main axis

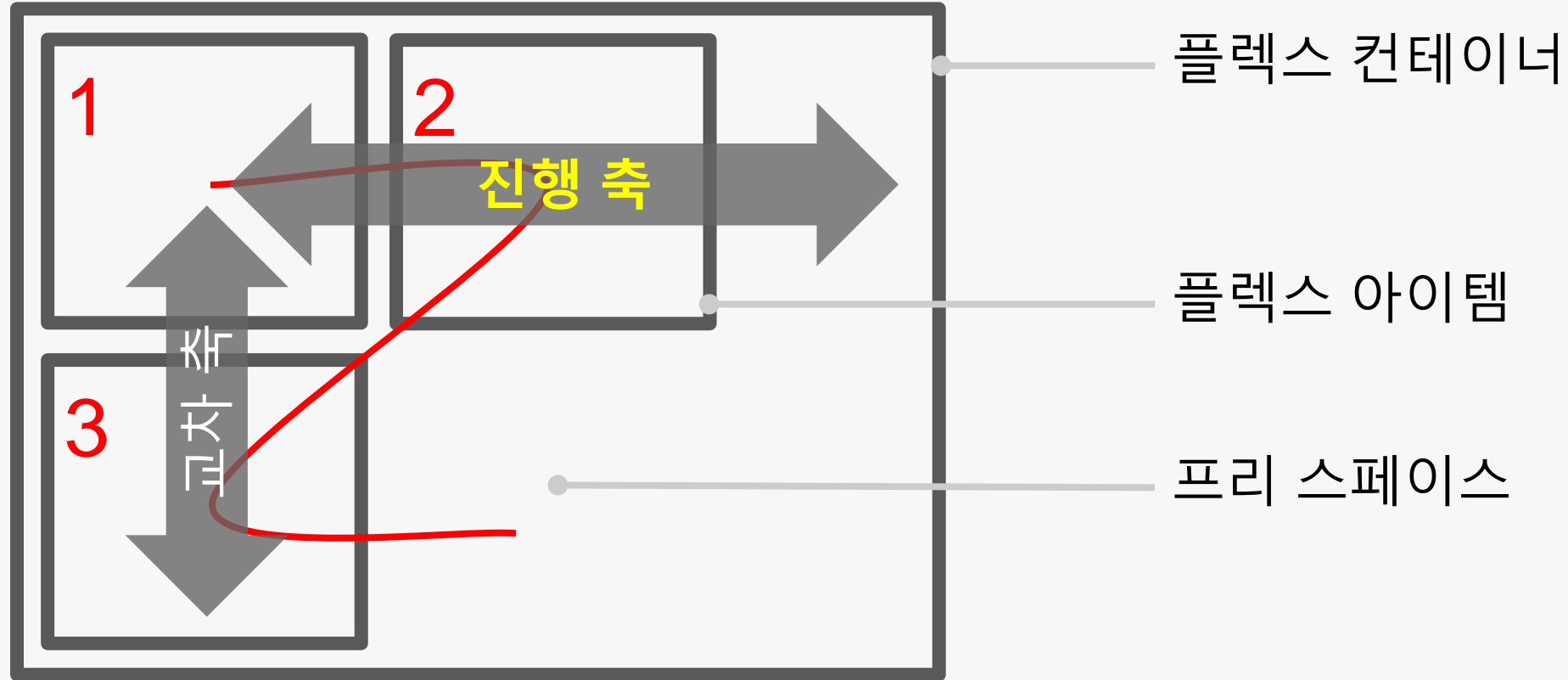
cross axis



# Flex 용어

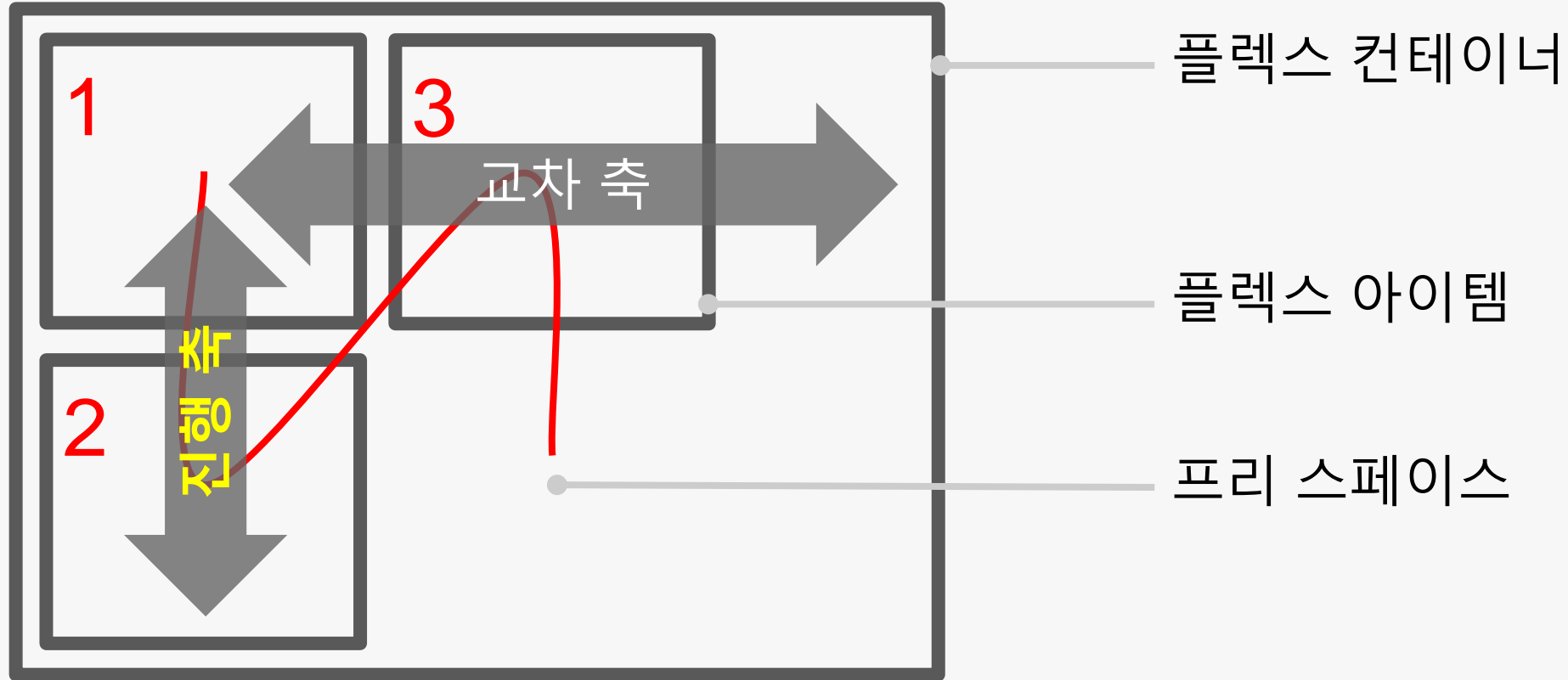
1. flex container(플렉스 컨테이너)
2. flex item(플렉스 아이템)
3. free space(빈 공간)
4. main axis(진행 축)
5. cross axis(교차 축)

# Flexible box model



진행 축이 **수평**인 경우 아이템이 흐르는 방향.

# Flexible box model



진행 축이 수직인 경우 아이템이 흐르는 방향.

# Flex container

'display' 속성 값이  
'flex' 또는 'inline-flex'인 박스.

내부에 흐르는 자식 콘텐츠(엘리먼트, 텍스트 노드)  
는  
저절로 플렉스 아이템이 된다.

# Flex item

플렉스 컨테이너 내부에서 흐르는 자식 박스 (엘리먼트, 텍스트 노드)를 플렉스 아이템이라고 부른다.

플렉스 박스 모델에 따라 배치된다.



# Flex container

플렉스 컨테이너에만 적용 가능한 속성이 있다.

```
.flex-container {  
    display: flex;                                /* 컨테이너 선언 */  
    flex-flow: row nowrap;                       /* 흐름 방향 + 줄 바꿈 */  
    flex-direction: row;                          /* 흐름 방향 */  
    flex-wrap: nowrap;                            /* 줄 바꿈 */  
    justify-content: flex-start;                  /* 진행 축 정렬 */  
    align-items: stretch;                        /* 교차 축 정렬 */  
    align-content: stretch;                      /* 여러 줄 교차 축 정렬 */  
}
```

# Flex item

플렉스 아이টে만 적용 가능한 속성이 있다.

```
.flex-item {  
    flex: 0 1 auto;           /* 팽창지수 + 수축지수 + 기준 크기 */  
    flex-grow: 0;             /* 팽창지수 */  
    flex-shrink: 1;           /* 수축지수 */  
    flex-basis: auto;         /* 기준 크기 */  
    align-self: auto;         /* 독립적 교차 축 정렬 */  
    order: 0;                 /* 배치 순서 */  
}
```

## Free space ★

플렉스 아이템이 점유하는 영역  
(flex-basis, width, height, padding, border, margin)을  
제외하고 남은 공간을 프리 스페이스라고 부른다.

'0, 양수, 음수' 프리 스페이스가 발생할 수 있다.

프리 스페이스는 플렉스 아이템의 팽창 지수(flex-grow)와  
수축 지수(flex-shrink)를 이용하여 플렉스 아이템으로 분배할 수 있다.

이 문서에서 'FS'는 '프리 스페이스'를 의미한다.

## Main axis (진행 축) ☆

플렉스 아이템 배치 방향으로써 플렉스 컨테이너에 flex-direction 속성을 적용하여 설정할 수 있다.

진행 축(flex-direction)의 초기 값은 row이며 row-reverse, column, column-reverse 값을 적용할 수 있다.

## Cross axis (교차 축) ☆

플렉스 아이템 배치 방향과 직각으로 교차하는 방향으로써 flex-direction 값의 직각 방향.

진행 축의 방향에 따라 교차 축이 달라지기 때문에 교차 축은 행이 될 수도 있고 열이 될 수도 있다.

align-\* 속성의 기준이 되는 축이다.

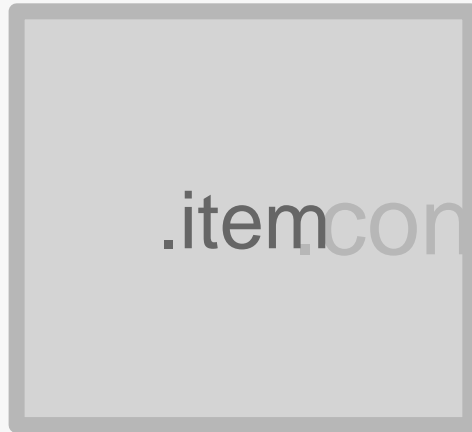
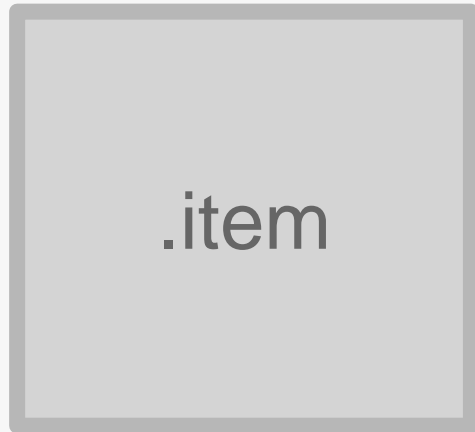
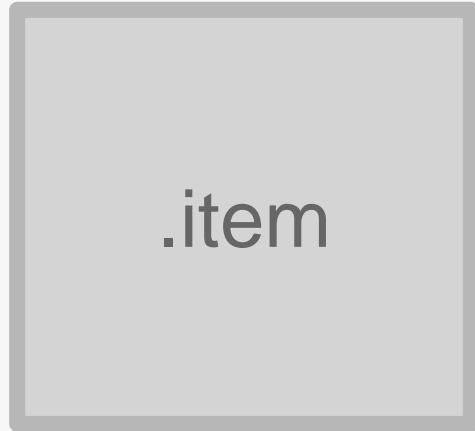
“

# Flex Container

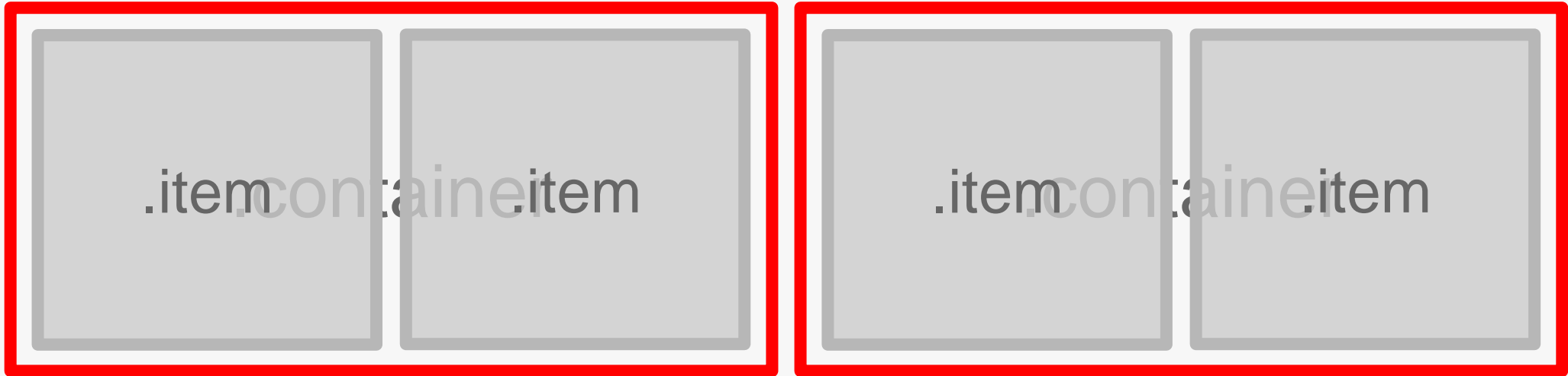
display: flex | inline-flex



```
.container { display : flex }
```



```
.container { display : inline-flex }
```





“

## 플렉스 아이템의 팽창과 수축

flex-grow:  
flex-shrink:  
flex-basis:  
☆ flex:



## 플렉스 아이템의 팽창과 수축

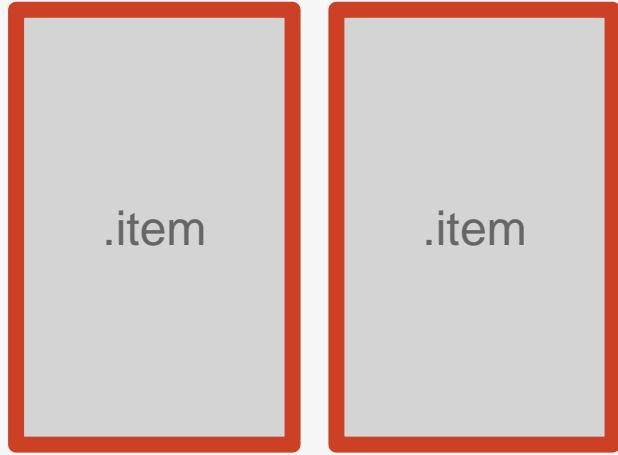
```
flex-grow      : 0;  
flex-shrink    : 1;  
flex-basis     : auto;  
flex         : 0 1 auto; 👍
```

## flex-grow:

양의 FS 발생 시 플렉스 아이템의 팽창을 제어.

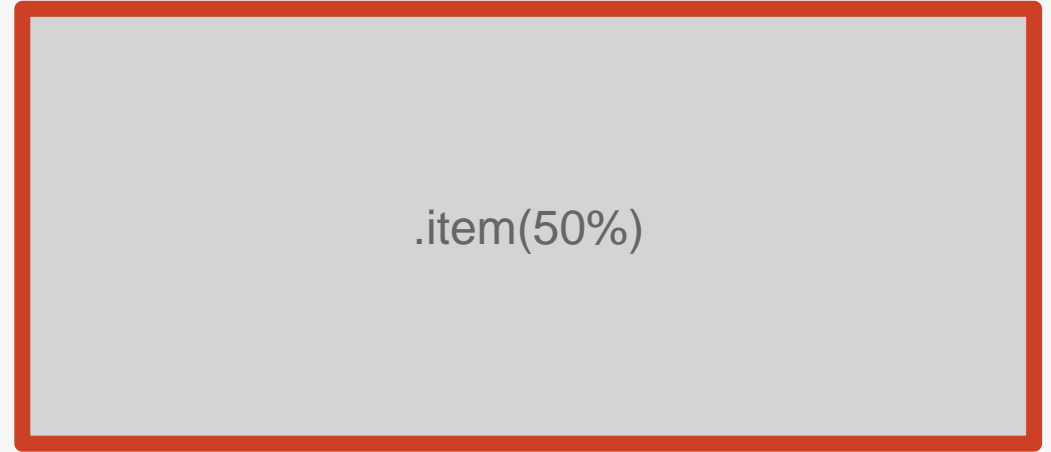
- 값: **<number>**  
// 음수 사용 불가. 보통 '0' 또는 '1' 사용.
- 초기 값: 0  
// 단축 속성에서 생략하면 '1'이 됨.
- 적용: 플렉스 아이TEM

```
.item { flex-grow: 0 }
```



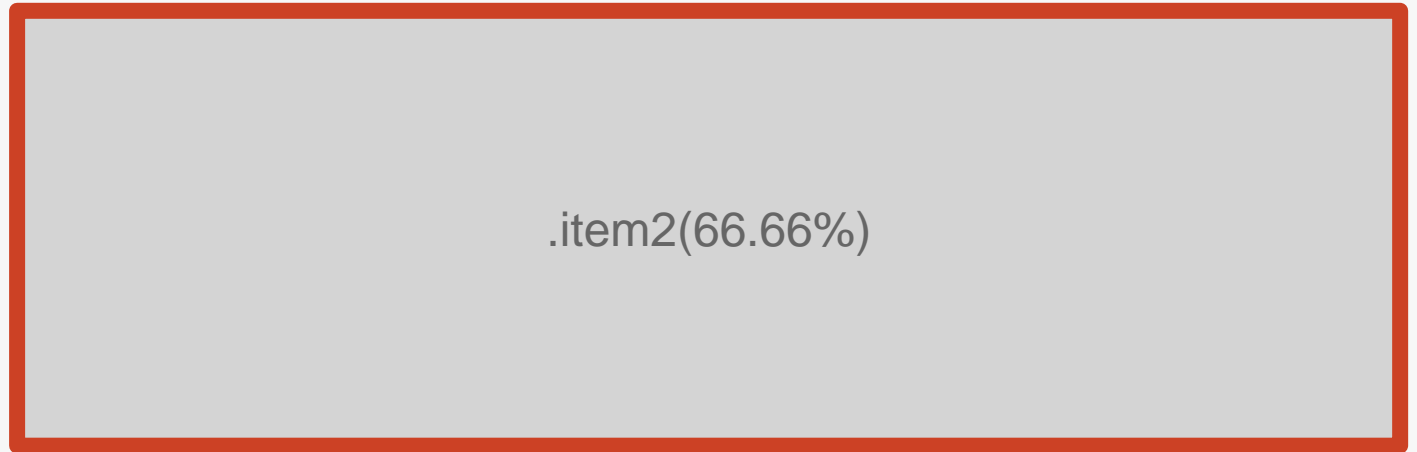
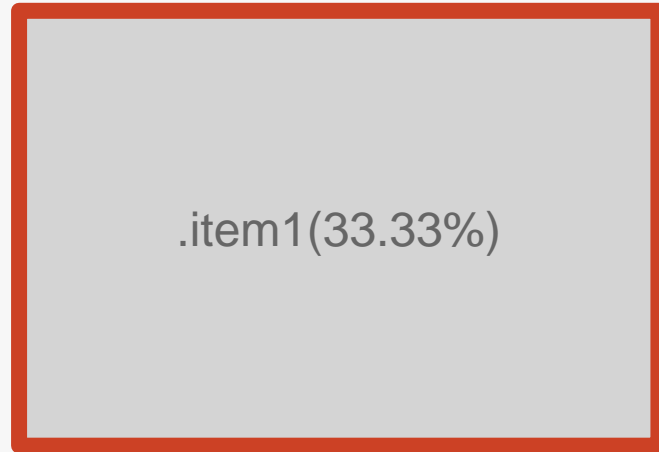
플렉스 컨테이너에 양의 FS가 발생했지만  
초기 값은 '0' 으로서 **팽창하지 않는다.**

```
.item { flex-grow: 1 }
```



양의 FS 발생 시 'flex-grow:1'을 적용하면  
플렉스 아이탬은 1:1 비율로 균등 팽창.

```
.item1 { flex-grow: 1 } .item2 { flex-grow: 2 }
```



플렉스 아이템마다 팽창 비율을 다르게 처리할 수 있다.

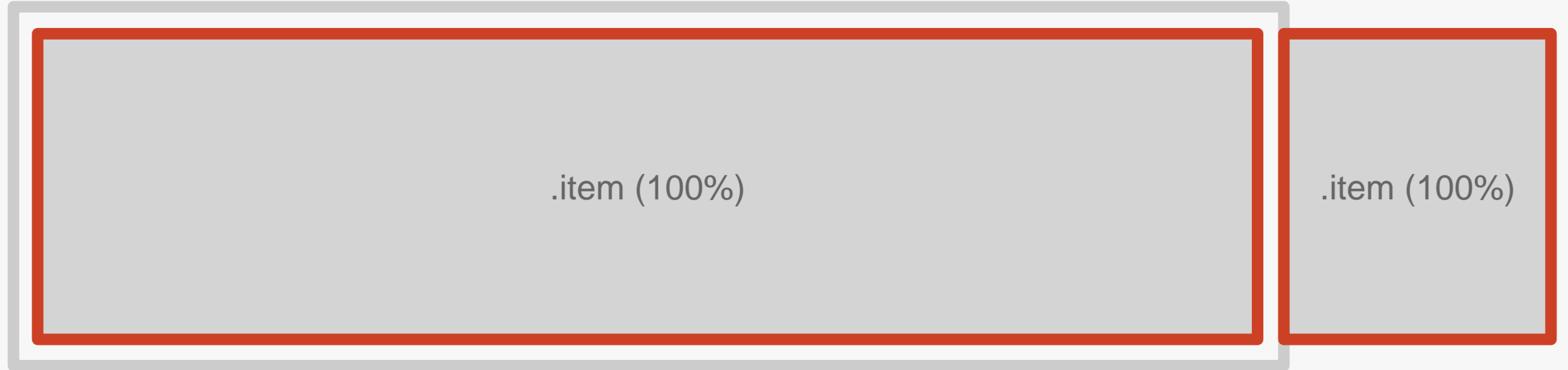
1:2 비율로 팽창.

## flex-shrink:

음의 FS 발생 시 플렉스 아이템의 수축을 제어.

- 값: <number>  
// 음수 사용 불가. 보통 '0' 또는 '1' 사용.
- 초기 값: 1  
// 단축 속성에서 생략해도 '1'이 됨.
- 적용: 플렉스 아이템

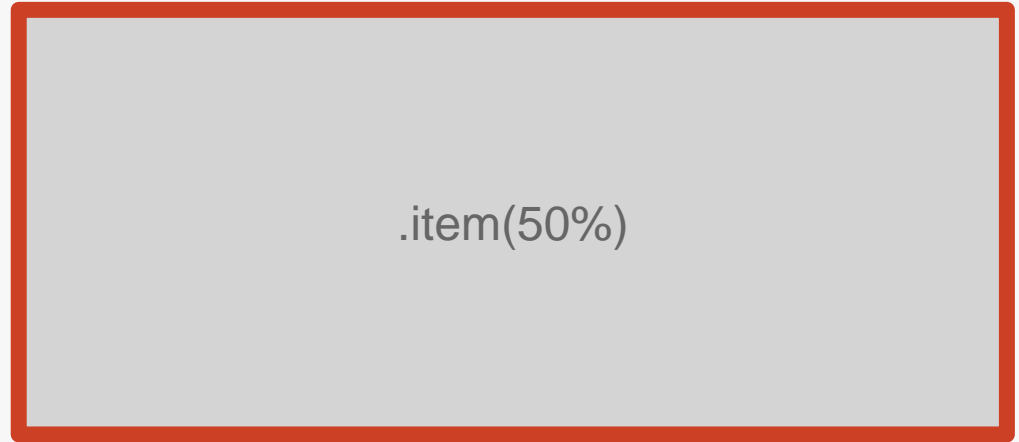
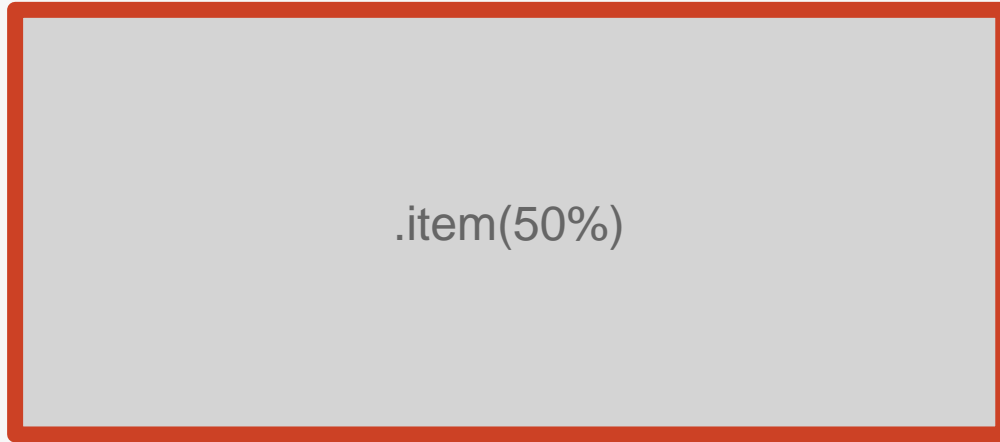
```
.item { flex-shrink: 0 }
```



음의 FS가 발생했다. 'flex-shrink:0'을 적용하면  
음의 FS가 발생 하더라도 수축하지 않음.

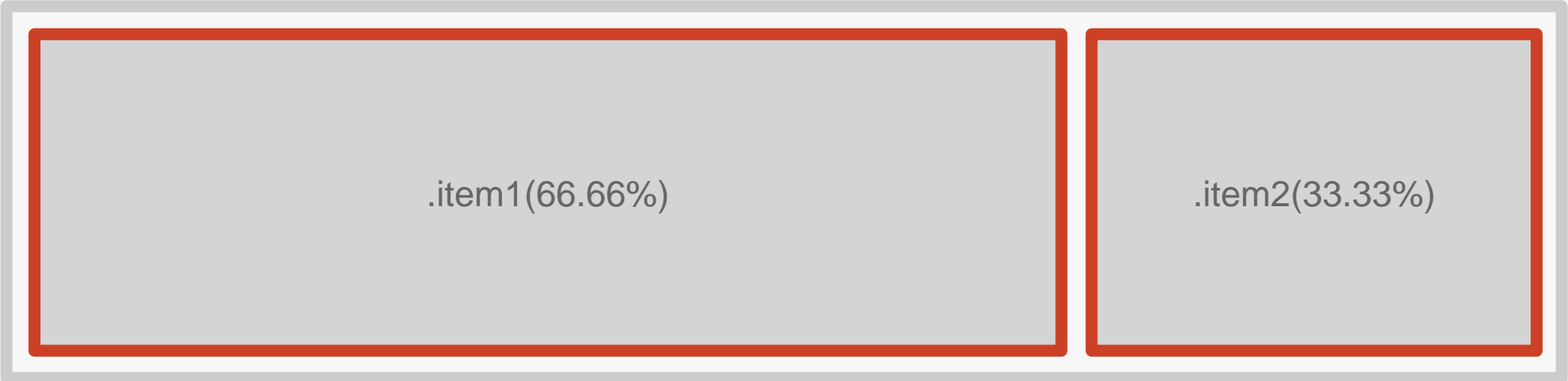


```
.item { flex-shrink: 1 }
```



초기 값은 '1' 으로써 음의 FS 발생 시 **균등 수축**.  
1:1 비율로 수축한 결과.

```
.item1 { flex-shrink: 1 } .item2 { flex-shrink: 2 }
```



플렉스 아이템마다 수축 비율을 다르게 처리할 수 있다.

1:2 비율로 수축한 결과.

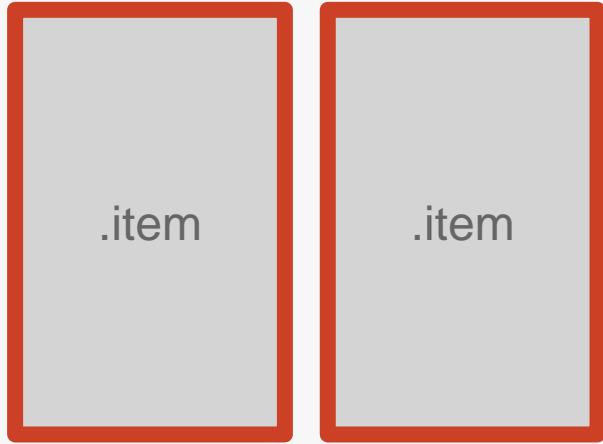
## flex-basis:

플렉스 아이템의 진행 방향 기본 크기를 설정함으로써

FS 초기 값에 영향을 준다.

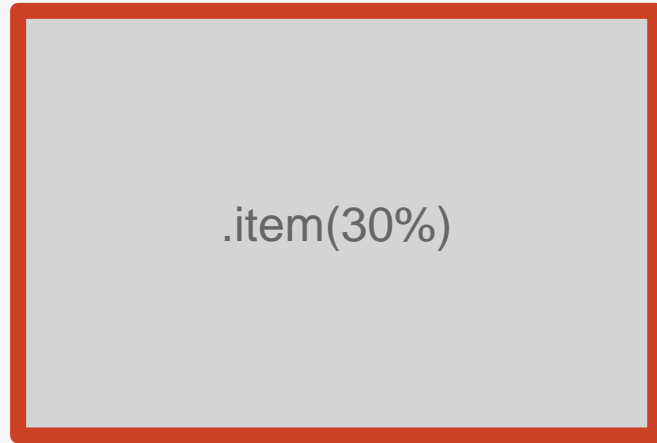
- 값: **content**⚠ | **<width>**  
// 팽창/수축하기 이전의 기본 크기.
- 초기 값: **auto**  
// content 또는 <width> 값이 적용됨.  
// 단축 속성에서 생략하면 초기 값이 '0' 이 된다.
- 적용: **플렉스 아이템**

```
.item { flex-basis: auto }
```



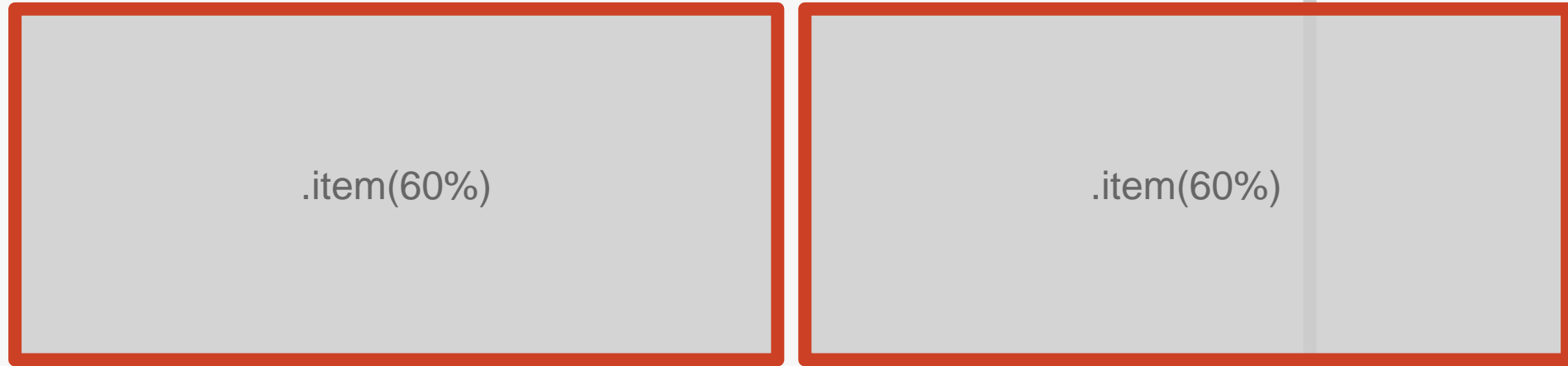
플렉스 아이টে姆에 적용된 기본 크기.  
컨테이너에 남은 공간이 양의 FS가 된다.

```
.item { flex-basis: 30% }
```



진행 축으로 **30%** 크기가 아이템의 기본 크기.  
컨테이너에 남은 공간(40%)이 양의 FS가 된다.

```
.item { flex-basis: 60% }
```



진행 축으로 **60%** 크기가 아이템의 기본 크기.  
컨테이너를 초과하는 공간(-20%)이 음의 FS가 된다

.

.item { flex-basis: 0 } ☆



FS 100%

아이템의 기본 크기가 '0' 이므로 컨테이너의 100%가 양의 FS가 되었다.

## flex: ★

플렉스 아이템의 '팽창, 수축, 기본 크기'를 제어하는 단축 속성.

- 값: none | [<'flex-grow'> <'flex-shrink'>? || <'flex-basis'>]  
// | 반드시 하나, ? 생략하거나 한 번, || 하나 또는 그 이상.
- 초기 값: 0 1 auto  
// 생략한 속성의 값은 재설정(변경) 된다.
- 적용: 플렉스 아이템



flex: 1 ☆ 실무에서 사용할 확률 99%

```
.item { flex: 1 } ✨
```

==

```
.item { flex: 1 1 }
```

==

```
.item { flex: 1 1 0 }
```

flex-grow 또는 flex-shrink 값을 선언하면  
flex-basis 값은 'auto' 에서 '0' 으로 변경된다.

**flex: 0%** (사용 빈도 낮음.👉)

```
.item { flex: 0% }
```

==

```
.item { flex: 1 0% }
```

==

```
.item { flex: 1 1 0% }
```

flex-grow 초기 값이 '0' 에서 '1' 으로 변경된다.

flex: initial (사용 빈도 낮음. 🗨️)

```
.item { flex: initial }
```

==

```
.item { flex: 0 1 auto }
```

플렉스 아이টে에 아무것도 선언하지 않았을 때의 초기 값. 굳이 이렇게 선언할 필요는 없다.

## flex: none (사용 빈도 낮음.👉)

```
.item { flex: none }
```

==

```
.item { flex: 0 0 auto }
```

플렉스 박스 모델을 사용하지 않기 위해  
값을 덮어 쓸 필요가 있을 때 이렇게 쓸 수 있다.  
flex-shrink 값이 변경된다.

flex: auto (사용 빈도 낮음. 🗨️)

```
.item { flex: auto }
```

==

```
.item { flex: 1 auto }
```

==

```
.item { flex: 1 1 auto }
```

flex-grow 초기 값이 '0' 에서 '1' 으로 변경된다.

“

## 플렉스 아이템의 방향과 순서

flex-direction:

flex-wrap:

☆ flex-flow:

☆ order:



# 플렉스 아이템의 방향과 순서

flex-direction

: row | row-reverse | column | column-

reverse;  
flex-wrap

flex-flow

: nowrap | wrap | wrap-reverse;

order

: <'flex-direction'> || <'flex-wrap'>; ☆

order

: <integer>; // 0, 1, -1...

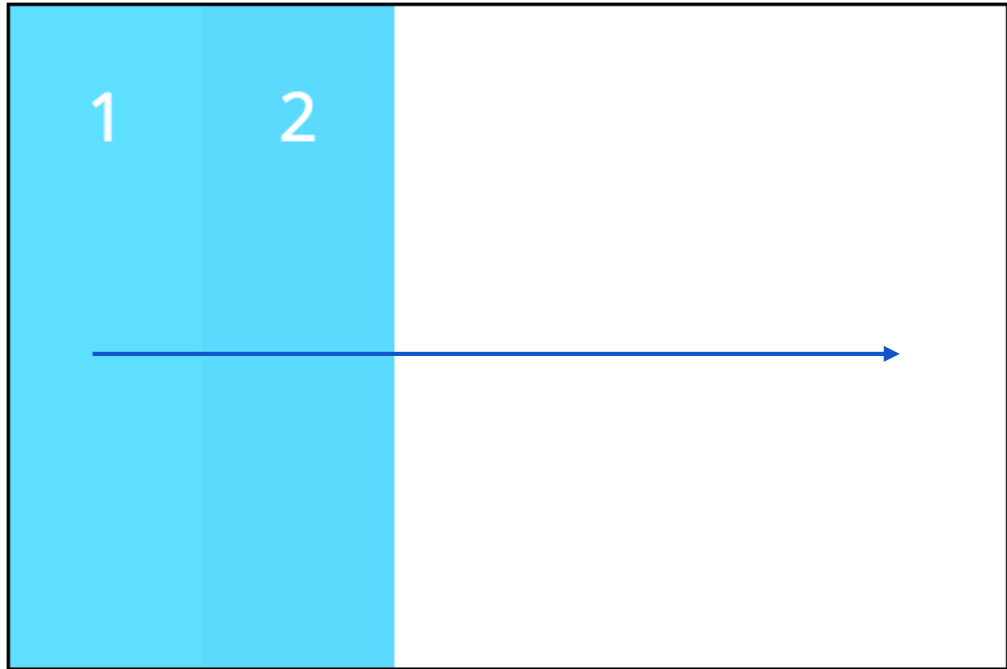
## flex-direction:

플렉스 아이템의 진행 방향.

- 값: row | row-reverse | column | column-reverse
- 초기 값: row
- 적용: 플렉스 컨테이너

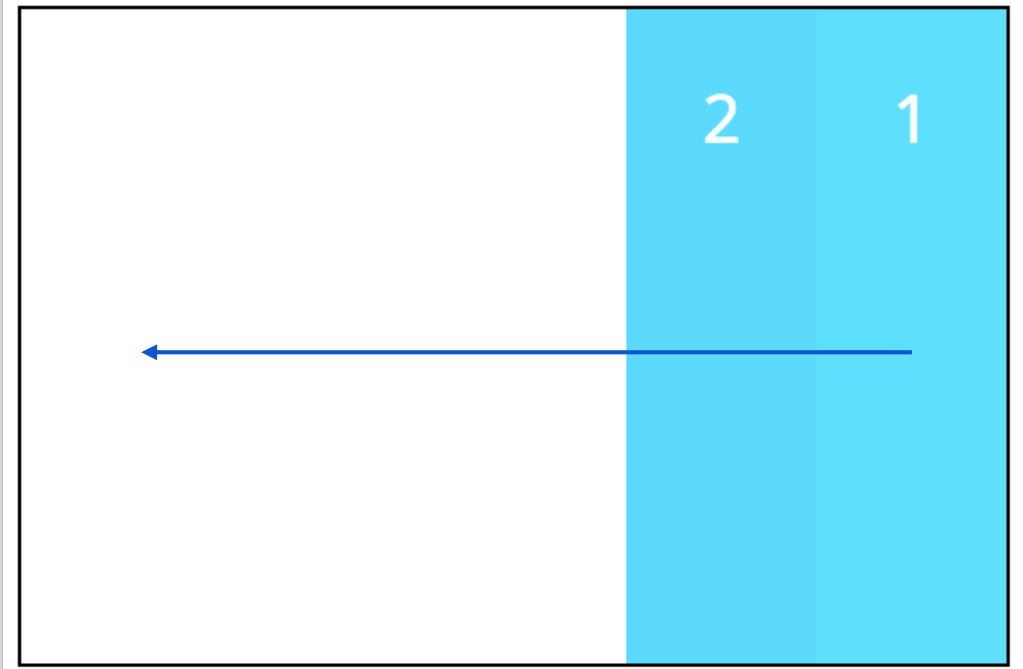


# flex-direction: row | row-reverse 🗨



## FLEX-DIRECTION

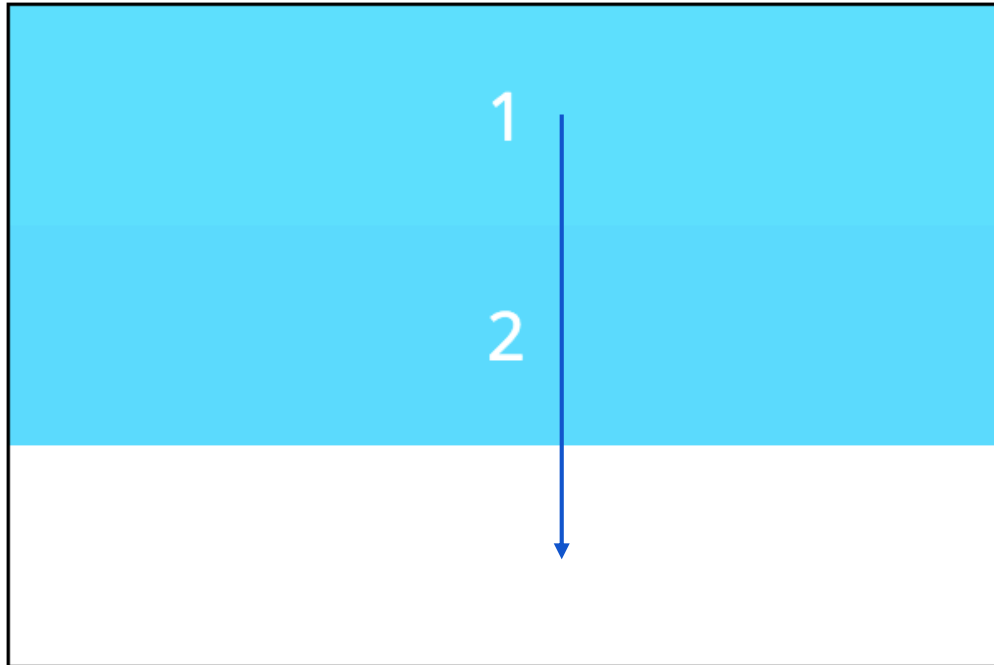
- ☒ row (default)
- ☐ row-reverse
- ☐ column
- ☐ column-reverse



## FLEX-DIRECTION

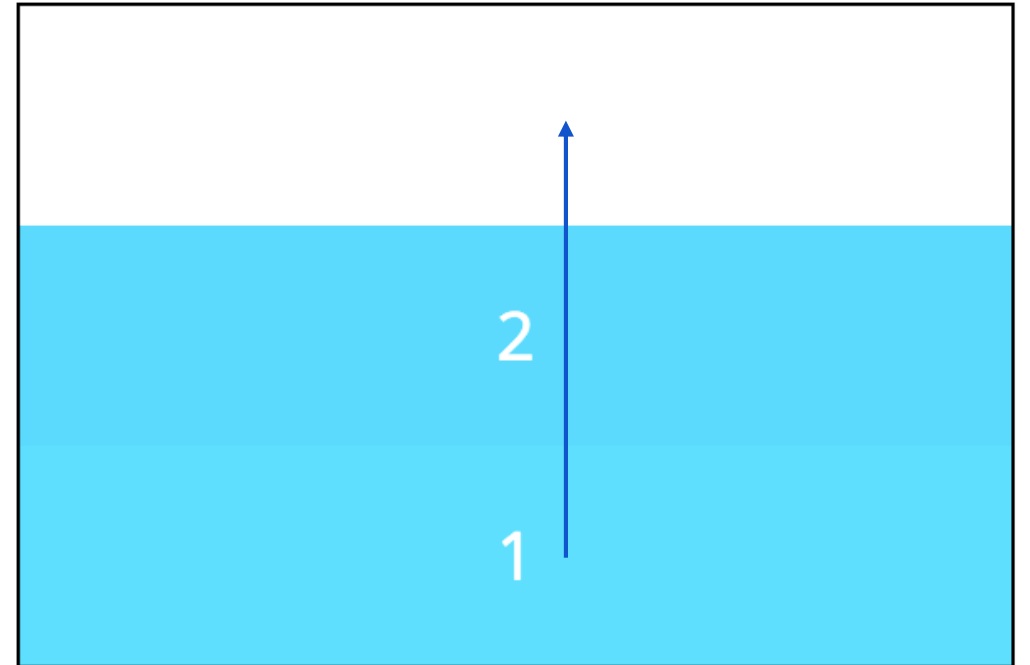
- ☐ row (default)
- ☒ row-reverse
- ☐ column
- ☐ column-reverse

# flex-direction: column | column-reverse 🗨️



## FLEX-DIRECTION

- ☐ row (default)
- ☐ row-reverse
- ☒ column
- ☐ column-reverse



## FLEX-DIRECTION

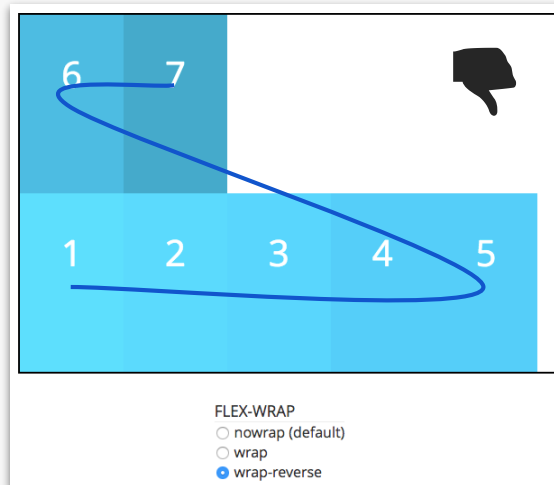
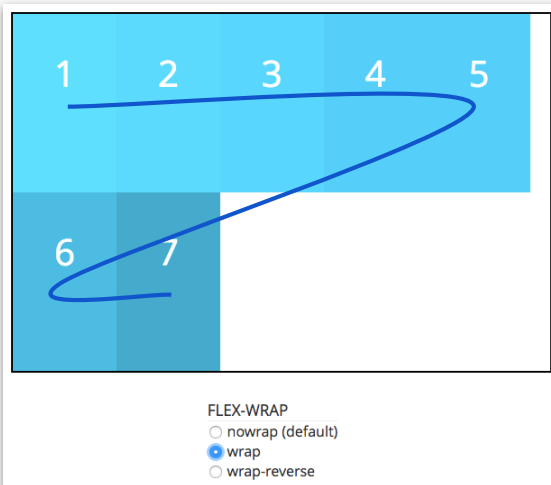
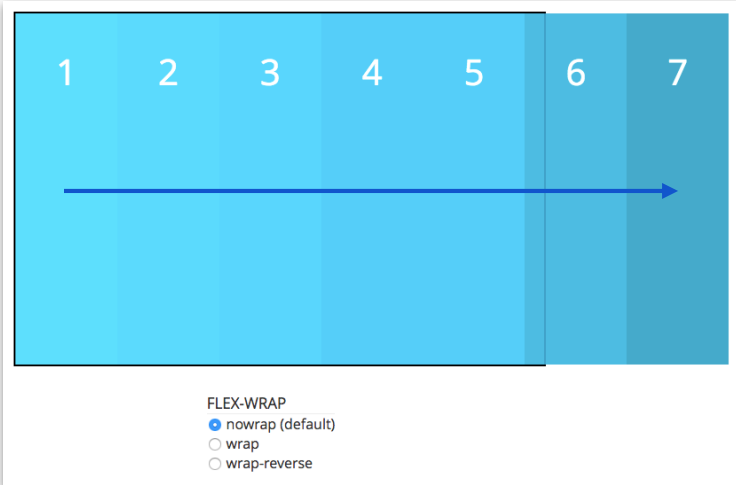
- ☐ row (default)
- ☐ row-reverse
- ☐ column
- ☒ column-reverse

# flex-wrap:

플렉스 아이템의 줄 바꿈.

- 값: nowrap | wrap | wrap-reverse
- 초기 값: nowrap
- 적용: 플렉스 컨테이너

# flex-wrap: nowrap | wrap | wrap-reverse

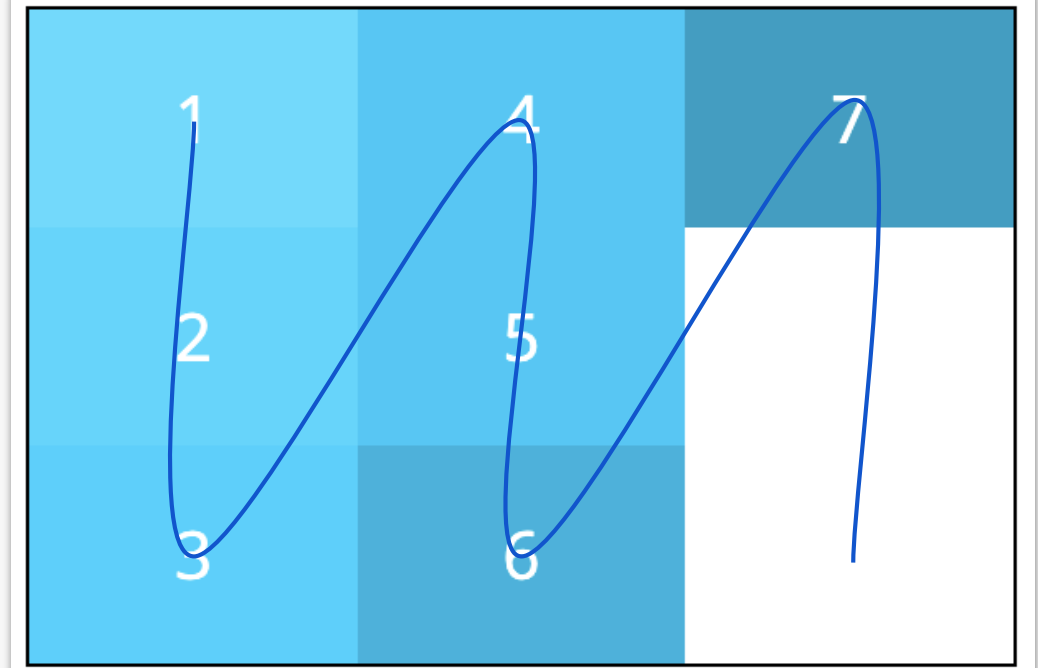
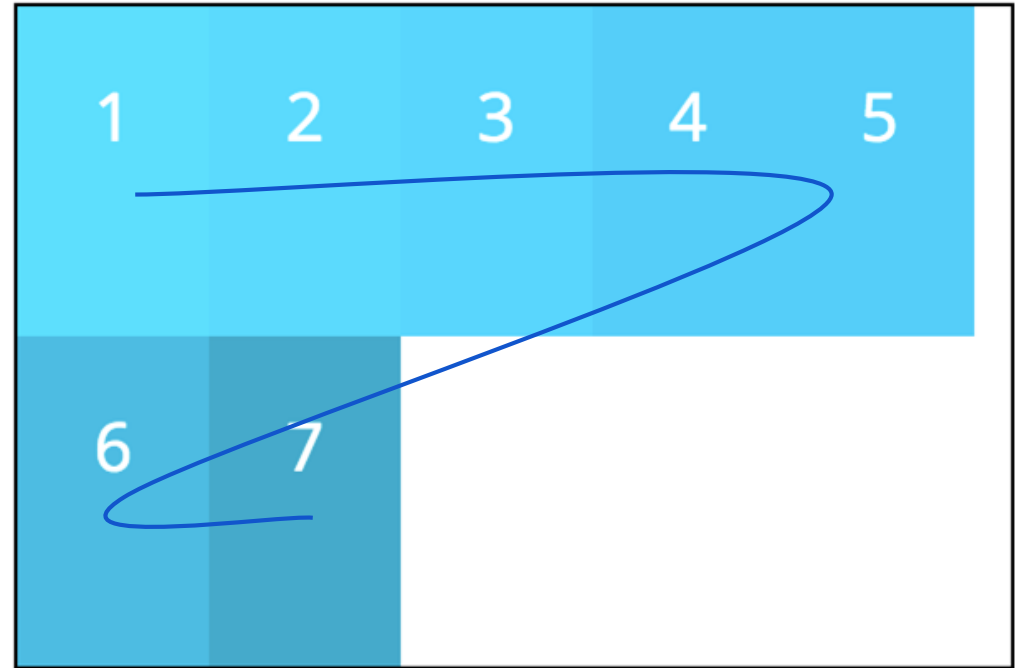


## flex-flow: ☆

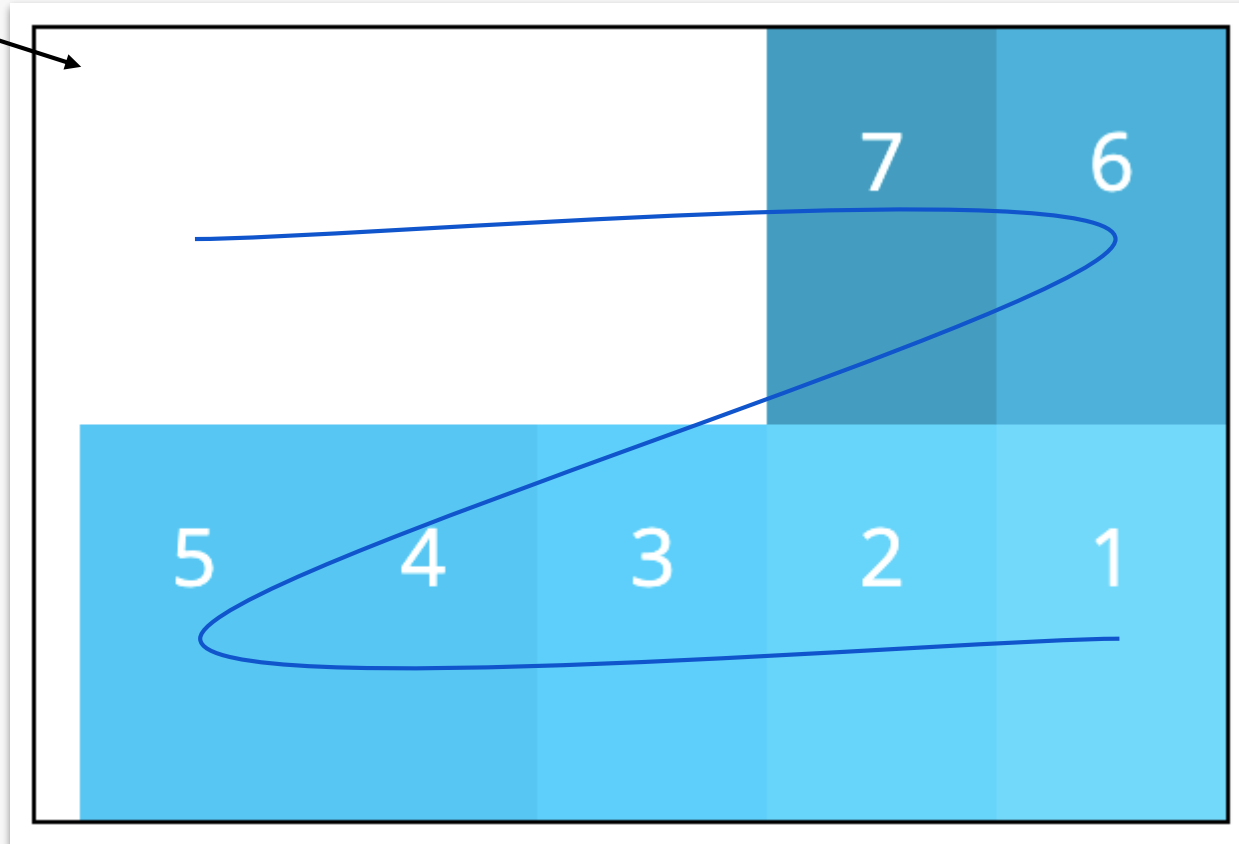
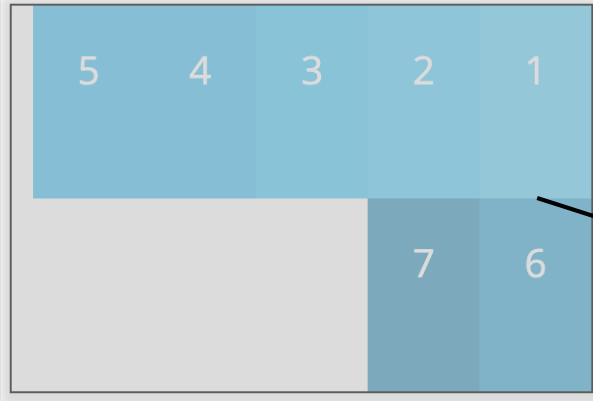
플렉스 아이템의 진행 방향과 줄 바꿈 단축.

- 값: <'flex-direction'> || <'flex-wrap'>
- 초기 값: row nowrap
- 적용: 플렉스 컨테이너

# flex-flow: row wrap | column wrap ★



# flex-flow: row-reverse wrap-reverse 🗨



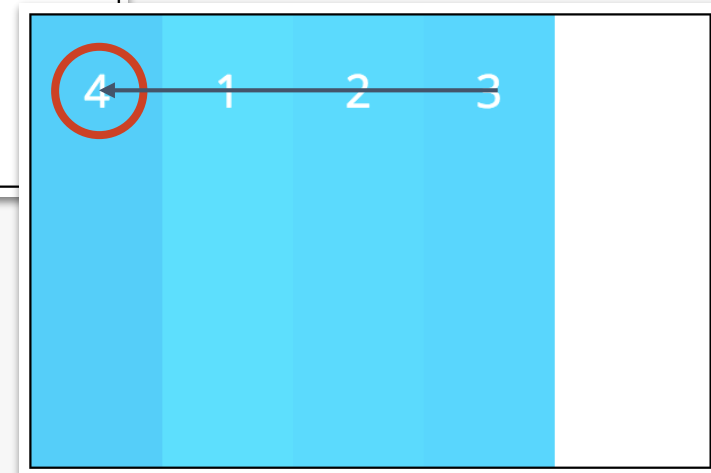
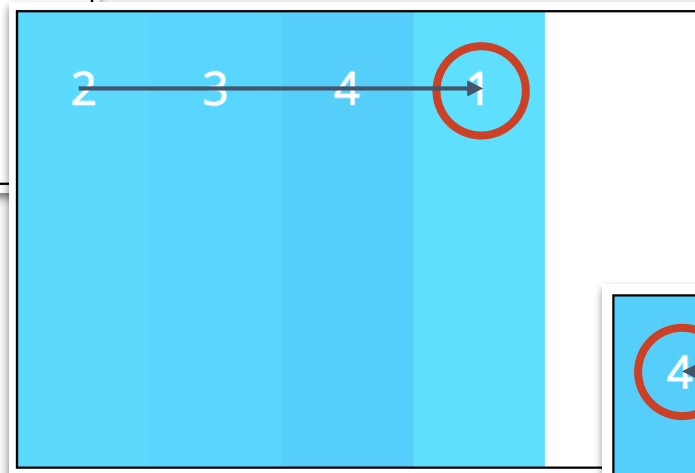
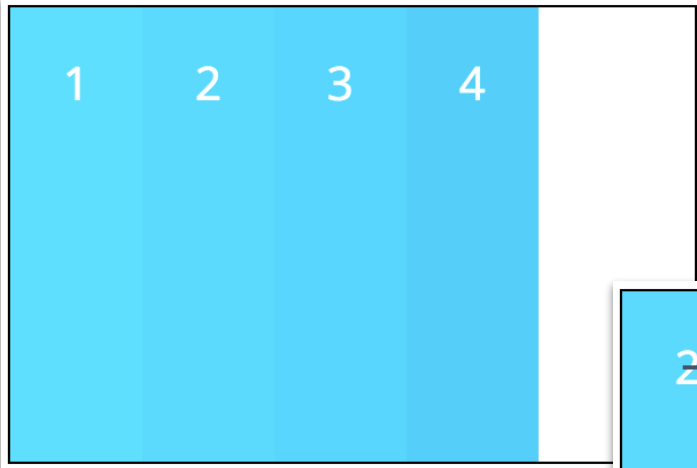
order: ☆

플렉스 아이템의 배치 순서.

- 값: <integer> // '0, 양의 정수, 음의 정수'
- 초기 값: 0
- 적용: 플렉스 아이템



order: 0 | 1 | -1



“

## 플렉스 아이템의 정렬과 간격

☆ justify-content:

☆ align-items:

align-self:

align-content:

☆ gap:

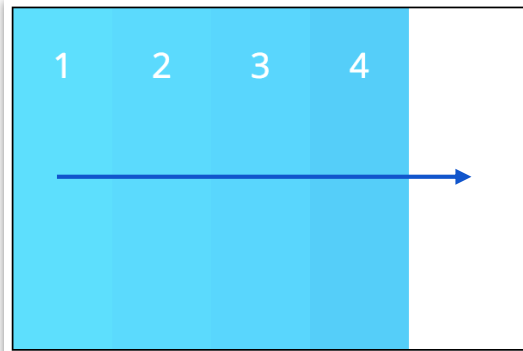


## justify-content: ☆

플렉스 아이템의 진행 축 정렬.

- 값: flex-start | flex-end | center | space-between | space-around | space-evenly
- 초기 값: flex-start
- 적용: 플렉스 컨테이너

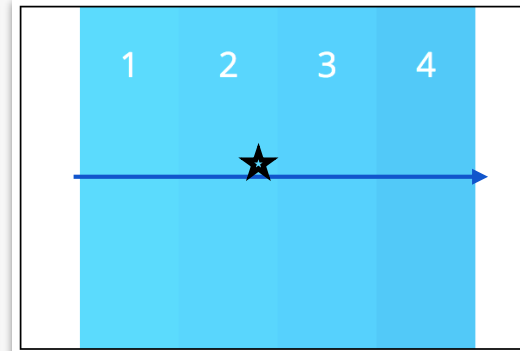
# justify-content: flex-start | flex-end | center | space-around | space-between | space-evenly



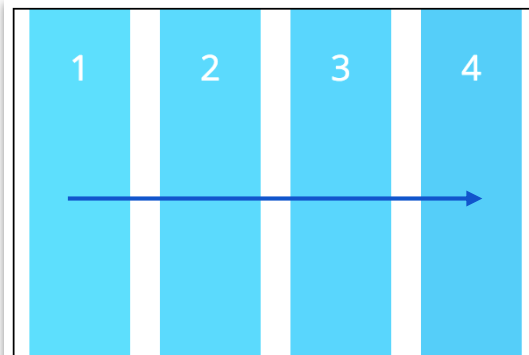
JUSTIFY-CONTENT  
☒ flex-start (default)  
☐ flex-end  
☐ center  
☐ space-around  
☐ space-between



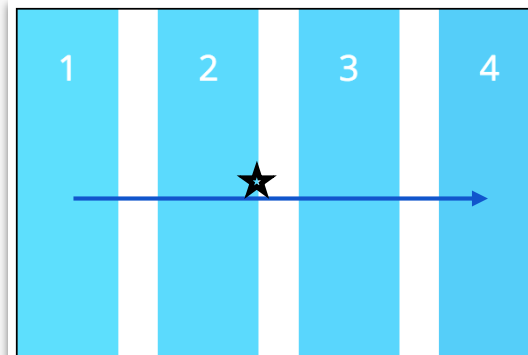
JUSTIFY-CONTENT  
☐ flex-start (default)  
☒ flex-end  
☐ center  
☐ space-around  
☐ space-between



JUSTIFY-CONTENT  
☐ flex-start (default)  
☐ flex-end  
☒ center  
☐ space-around  
☐ space-between



JUSTIFY-CONTENT  
☐ flex-start (default)  
☐ flex-end  
☐ center  
☒ space-around  
☐ space-between



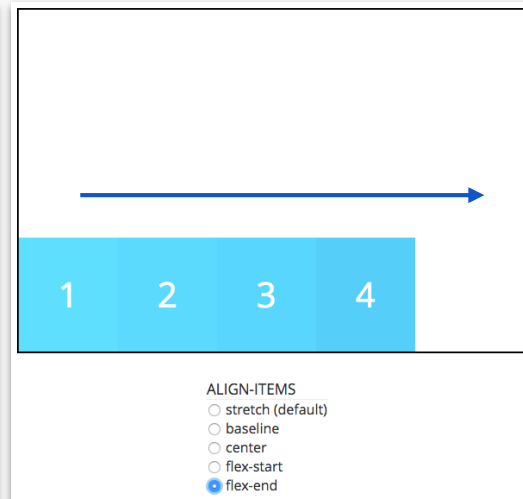
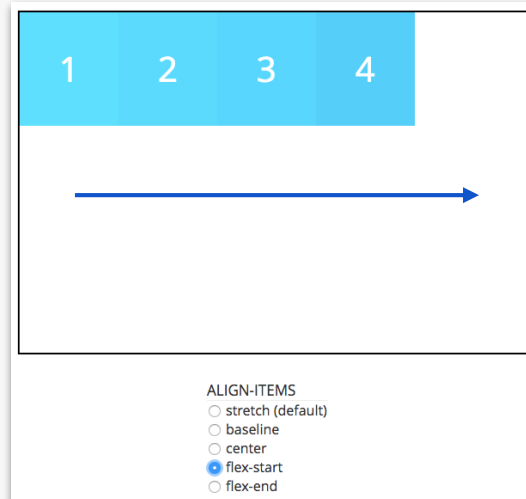
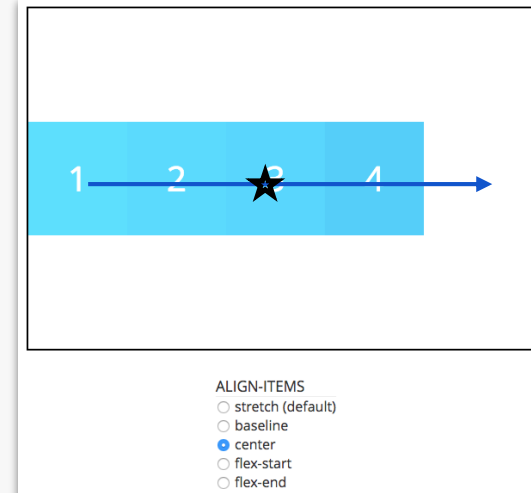
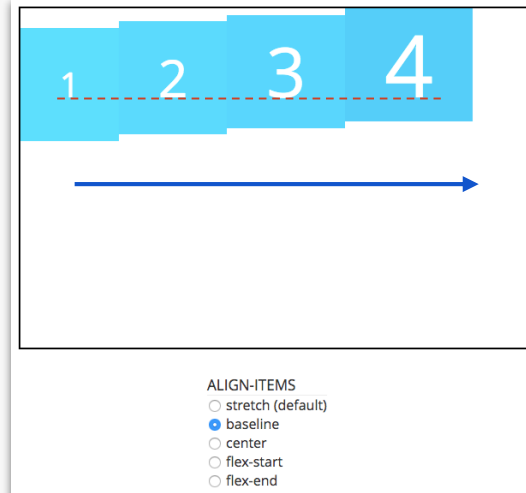
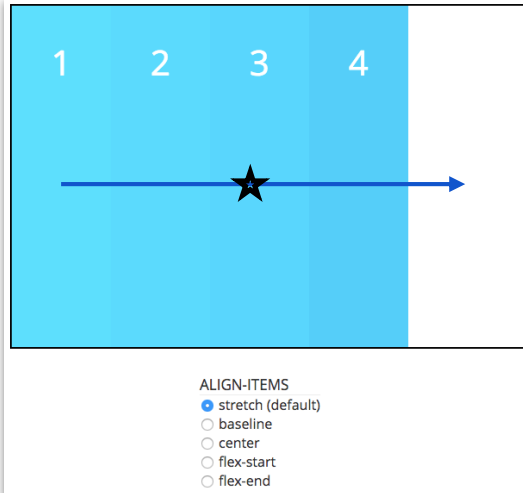
JUSTIFY-CONTENT  
☐ flex-start (default)  
☐ flex-end  
☐ center  
☐ space-around  
☒ space-between

## align-items: ★

플렉스 아이템이 한 줄일 때 교차 축 정렬.

- 값: flex-start | flex-end | center | baseline | stretch
- 초기 값: **stretch** // 아이템을 교차 축으로 잡아 늘림.
- 적용: 플렉스 컨테이너

# align-items: stretch | baseline | center | flex-start | flex-end

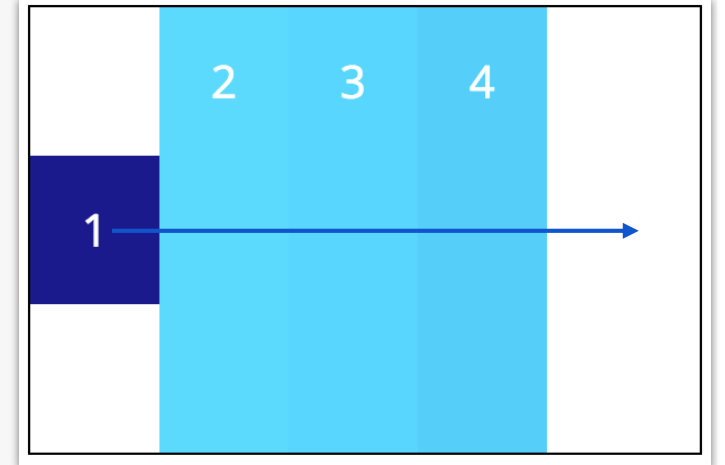
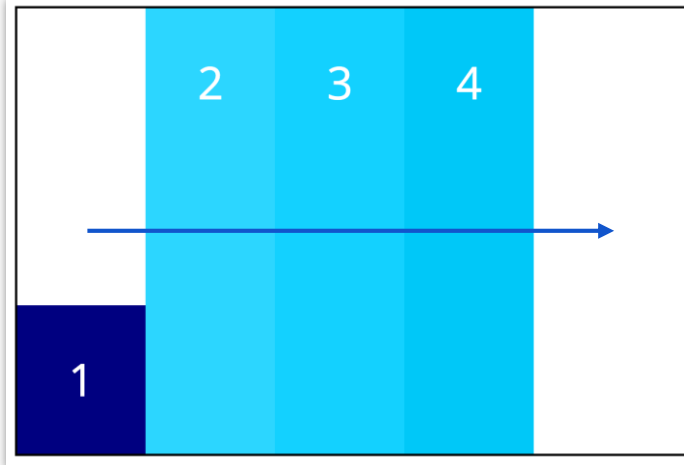
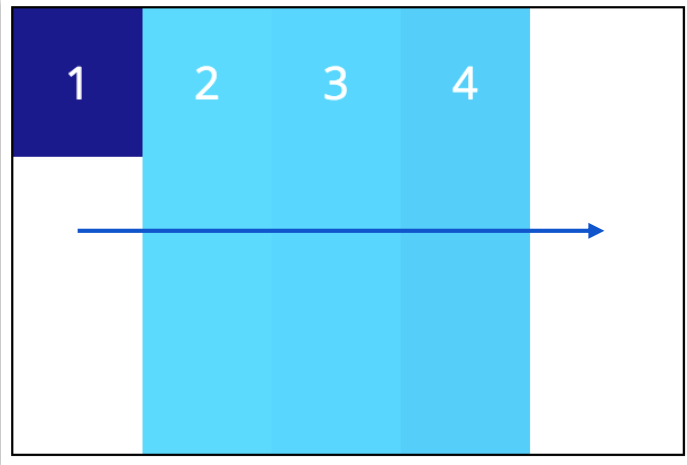


## align-self:

플렉스 아이템의 독립적 교차 축 정렬.

- 값: auto | flex-start | flex-end | center | baseline | stretch
- 초기 값: auto // 컨테이너의 align-items 값을 상속 받음.
- 적용: 플렉스 아이템

# align-self: auto | flex-start | flex-end | center | baseline | stretch



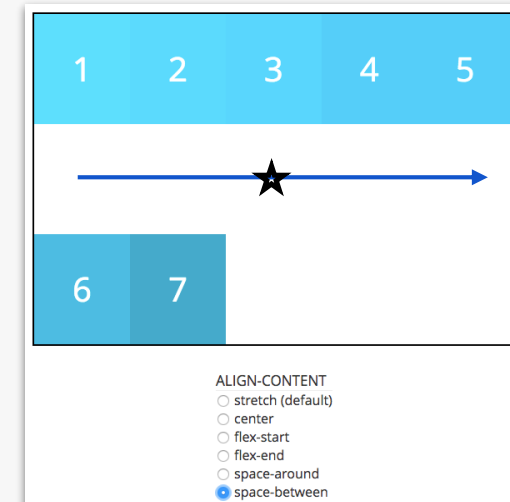
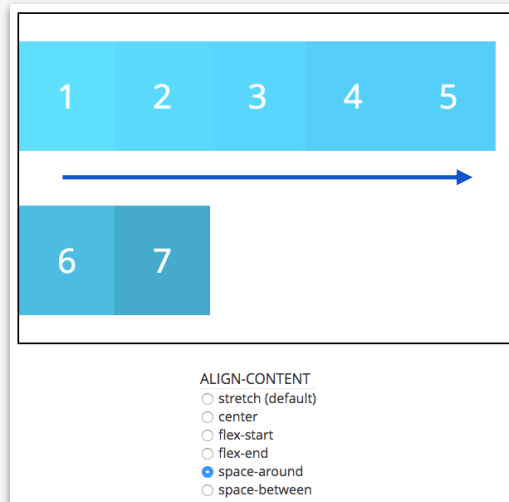
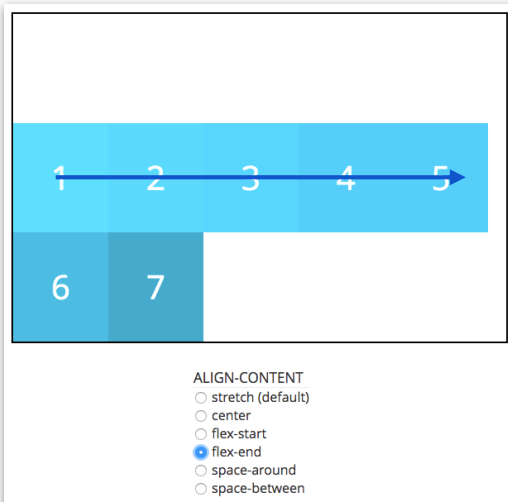
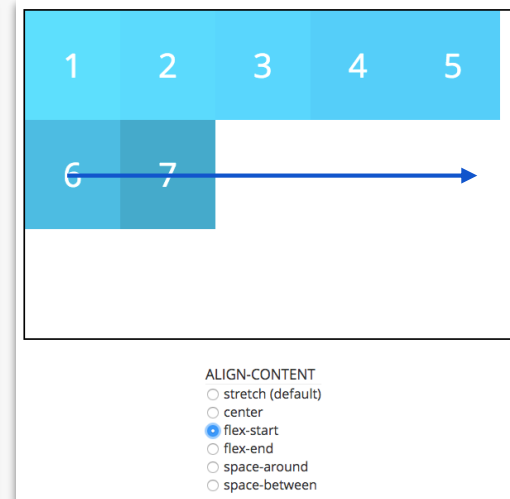
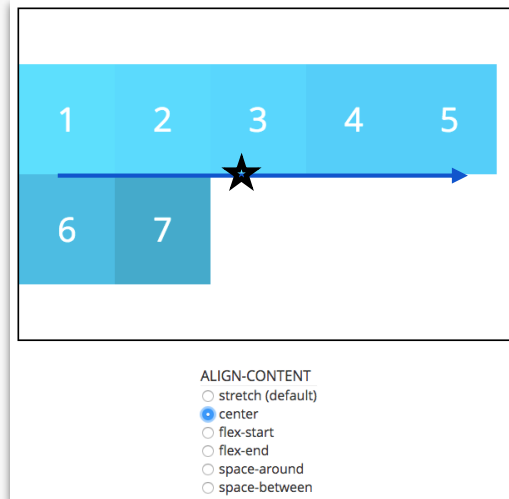
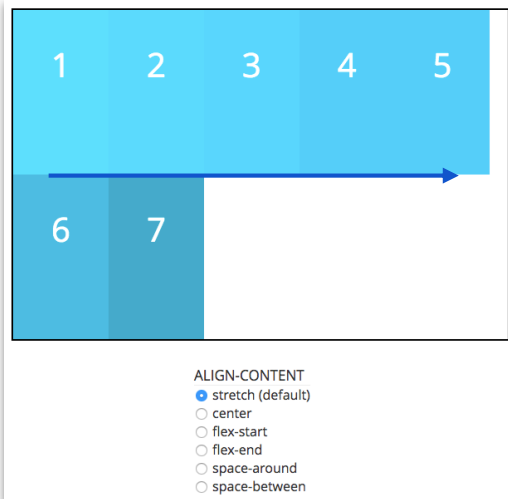


## align-content:

플렉스 아이템의 여러 줄 교차 축 정렬과 간격.

- 값: stretch | center | flex-start | flex-end | space-around | space-between | space-evenly
- 초기 값: stretch // 줄 간격을 교차 축으로 균등하게 늘림.
- 적용: '여러 줄' 플렉스 컨테이너

# align-content: stretch | center | flex-start | flex-end | space-around | space-between | space-evenly

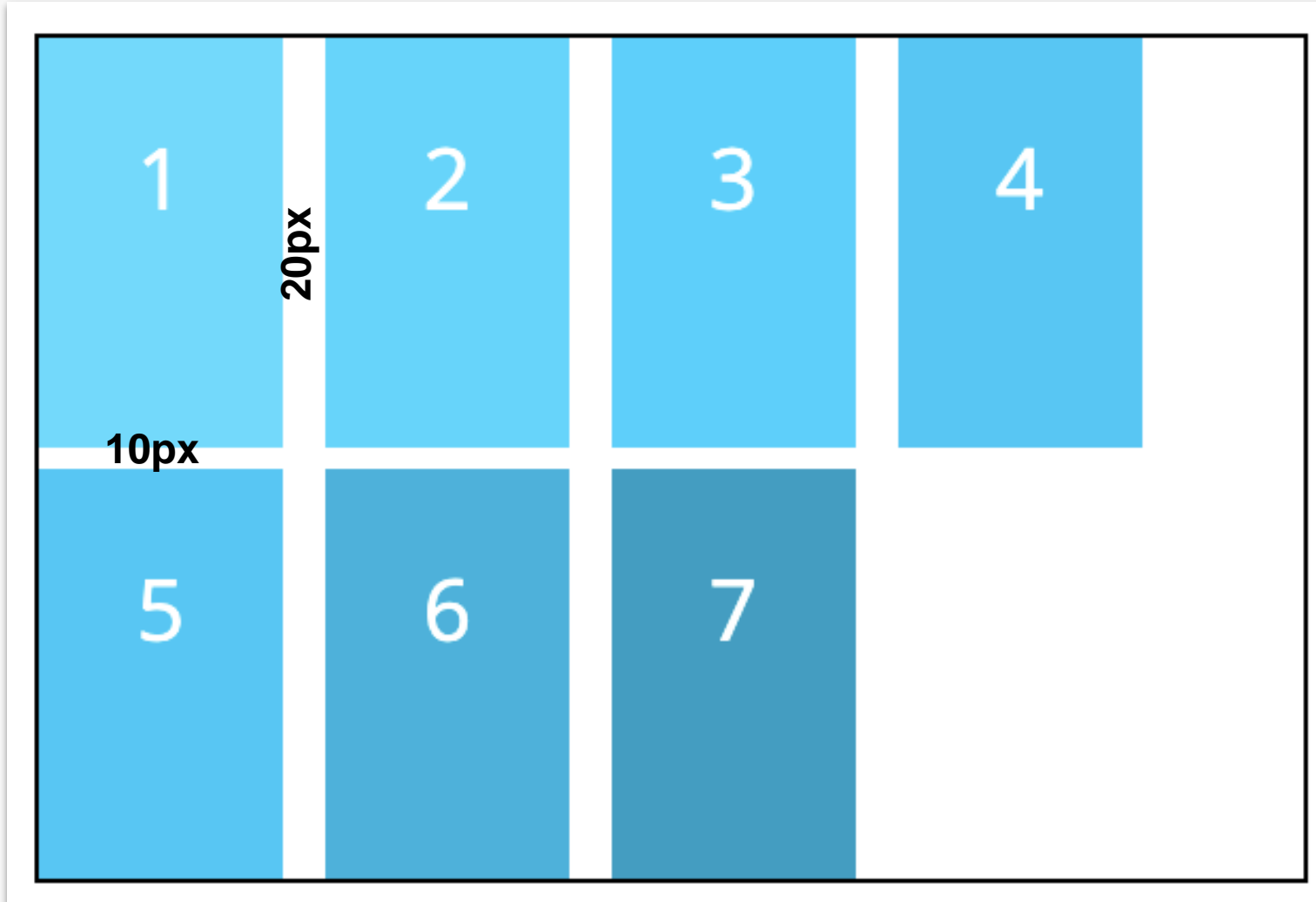


## gap: ☆

다중 컬럼, 플렉스, 그리드 아이템 사이의 간격.

- 값: <'row-gap'> <'column-gap'>?
- 초기 값: normal // == 다중 컬럼에서 1em 그렇지 않으면 0
- 적용: 컬럼/플렉스/그리드 컨테이너.

gap: 10px 20px



“

# SUMMARY



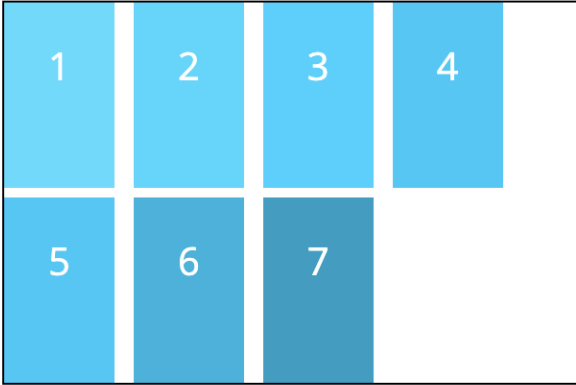
## “ Summary

- flex-grow/-shrink/-basis 대신 flex 단축 속성 사용을 권장. flex 단축 속성을 통해 팽창, 수축, 기본 크기를 이해하고 제어하는 것이 핵심.
- justify-content, align-items 속성을 통해 정렬 방식을 이해하고 align-self/-content, flex-direction/-wrap/-flow, order 속성은 필요할 때 학습.
- 진행 축을 뒤집는(-reverse) 것을 추천하지 않음.

# “Summary

TEST CSS FLEXBOX RULES

CHANGE CHILD COUNT



**FLEX-DIRECTION**

- ☒ row (default)
- ☐ row-reverse
- ☐ column
- ☐ column-reverse

**FLEX-WRAP**

*whether items wrap to the next row (only applies if combined width of items is greater than container's)*

- ☐ nowrap (default)
- ☒ wrap
- ☐ wrap-reverse

**JUSTIFY-CONTENT**

*alignment along the x axis*

- ☒ flex-start (default)
- ☐ flex-end
- ☐ center
- ☐ space-around
- ☐ space-between

**ALIGN-ITEMS**

*alignment along the y axis*

- ☒ stretch (default)
- ☐ baseline
- ☐ center
- ☐ flex-start
- ☐ flex-end

**ALIGN-CONTENT**

*only applies if there is more than one row of items*

- ☒ stretch (default)
- ☐ center
- ☐ flex-start
- ☐ flex-end
- ☐ space-around
- ☐ space-between

<http://flexbox.help/>

“

## 예습 과제





# Grid garden

## GRID GARDEN

단계 1 of 28

Grid 정원에 오신것을 환영합니다. 여러분들은 CSS 코드를 작성하여 여러분의 당근 정원을 키울 수 있습니다. 당근이 있는 지역에 `grid-column-start` 속성을 이용하여 물을 주십시오.  
예를 들어, `grid-column-start: 3;`와 같이 입력시 그리드 세번째 세로선에서 시작되는 영역에 물을 넣어 넣으 있습니다. 이는 그리드의 왼쪽에서 세번째 열(column) 항목을 뜻합니다.

```
1 #garden {  
2   display: grid;  
3   grid-template-columns: 20% 20% 20% 20% 20%;  
4   grid-template-rows: 20% 20% 20% 20% 20%;  
5 }  
6  
7 #water {  
8     
9 }  
10  
11  
12  
13  
14
```

다음

Grid Garden은 다음에 의해 개발되었습니다 [Codecademy](#) • [GitHub](#) • [Twitter](#) • [한국어](#)

CSS Flexbox를 배워볼까요? 여기를 클릭 [Flexbox Froggy](#)

<https://cssgridgarden.com/#ko>



Thank you !

