



# CSS Flex

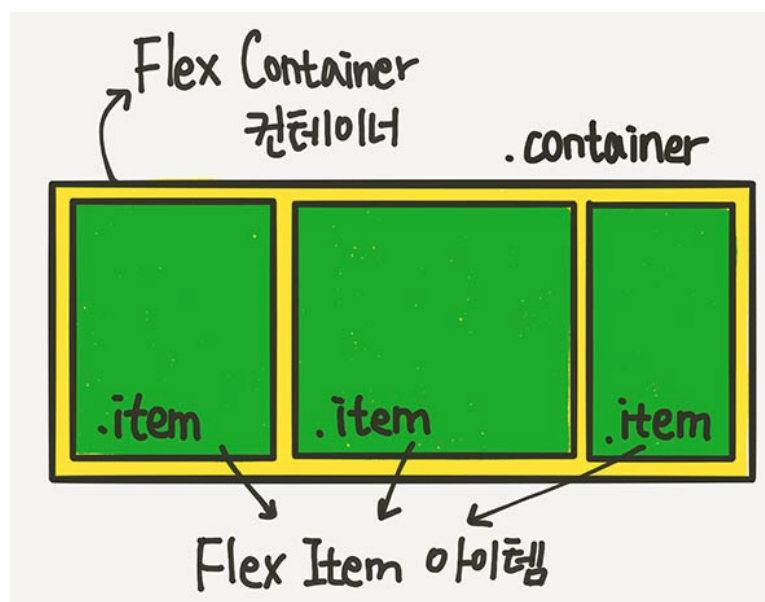
## [ Flex ]

```
<div class="container">
  <div class="item">helloflex</div>
  <div class="item">abc</div>
  <div class="item">helloflex</div>
</div>
```

**div.container** (부모 요소) : **Flex Container**(플렉스 컨테이너)

**div.item** (자식 요소) : **Flex Item**(플렉스 아이템)

컨테이너가 Flex의 영향을 받는 전체 공간이고, 설정된 속성에 따라 각각의 아이템들이 어떤 형태로 배치되는 것.



## [ Flex의 속성 ]

- 컨테이너에 적용하는 속성
- 아이템에 적용하는 속성

## [ Flex 컨테이너에 적용하는 속성들 ]

Flex 컨테이너에 **display: flex;**를 적용하는게 시작

### 1. **display: flex;**

- flex 아이템들은 **가로 방향으로 배치**됨
- 자신이 가진 내용물의 width만큼 차지하게 됨
  - display: block; / display: inline; 중 display: inline;과 유사
- height는 컨테이너의 높이만큼 늘어남

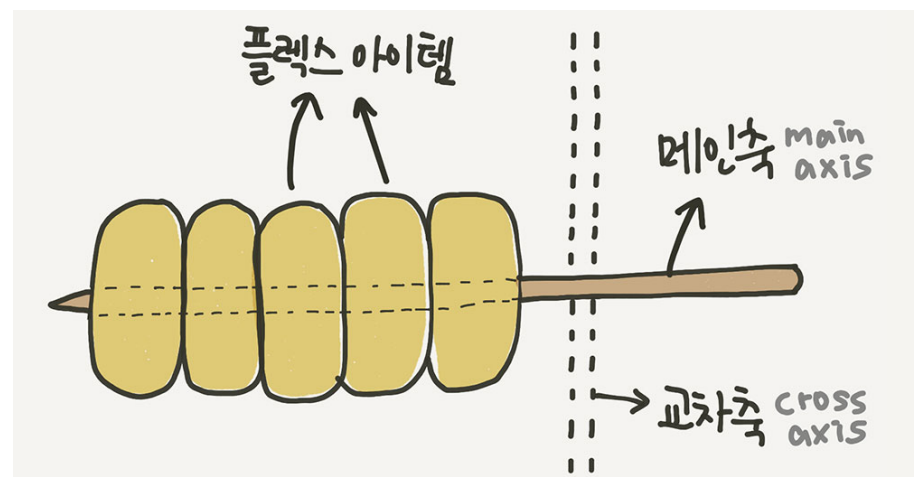
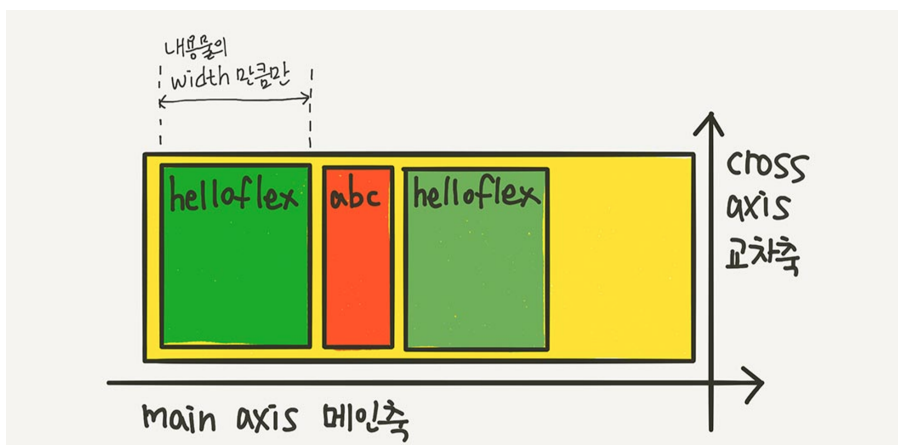
```
.container {
  display: flex;
```

```
}
```

## block



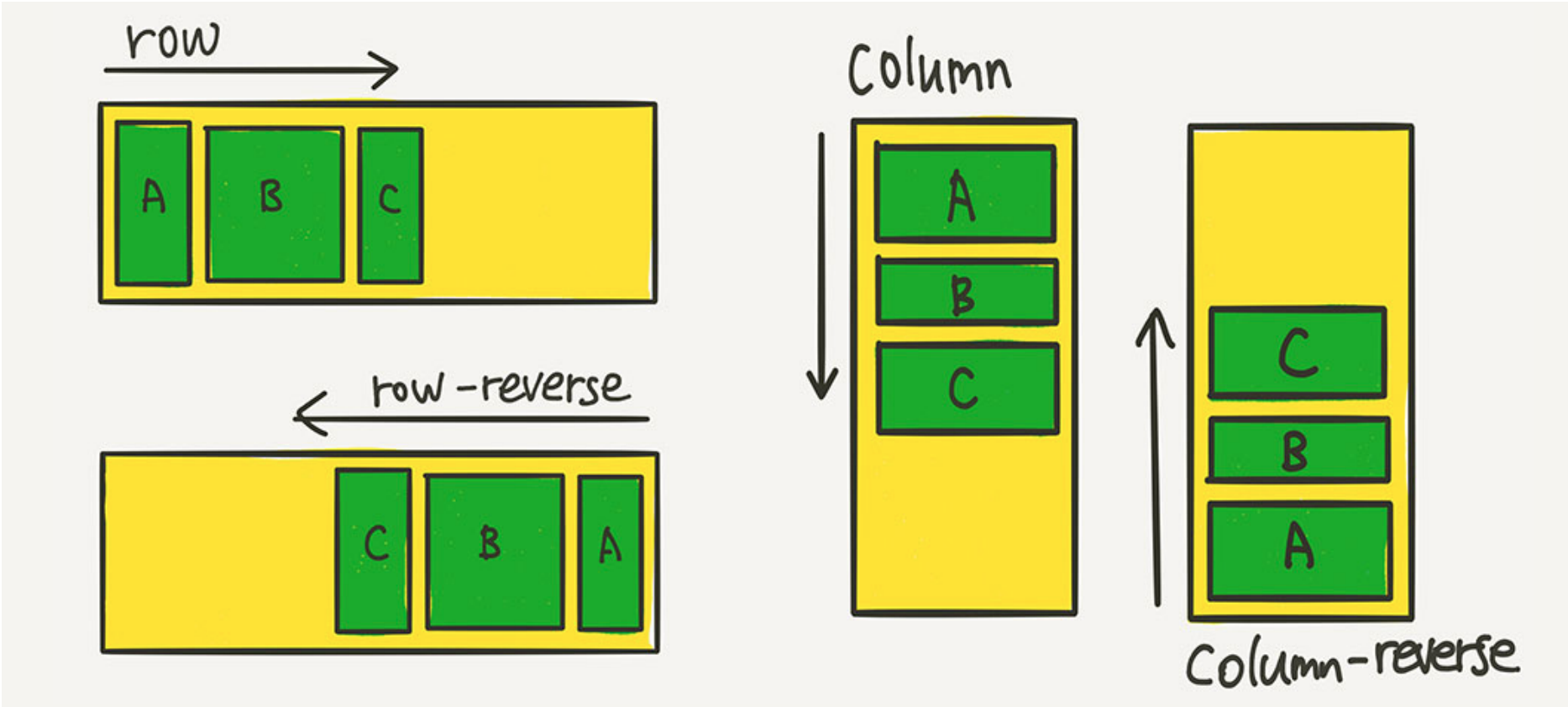
## flex



## 2. 배치 방향 설정 - flex-direction

아이템들이 배치되는 축의 방향을 결정하는 속성

```
.container {  
  flex-direction: row;  
  /* flex-direction: column; */  
  /* flex-direction: row-reverse; */  
  /* flex-direction: column-reverse; */  
}
```



row(기본값)

AAA

BBBBBBBBBBBB

CCCCC

☒ row ☐ row-reverse ☐ column ☐ column-reverse

아이템을 행(가로) 방향 배치

row-reverse

CCCCC

BBBBBBBBBBBB

AAA

☐ row ☒ row-reverse ☐ column ☐ column-reverse

아이템을 역순 가로 방향 배치

column

AAA

BBBBBBBBBBBB

CCCCC

☐ row ☐ row-reverse ☒ column ☐ column-reverse

아이템을 열(세로) 방향 배치

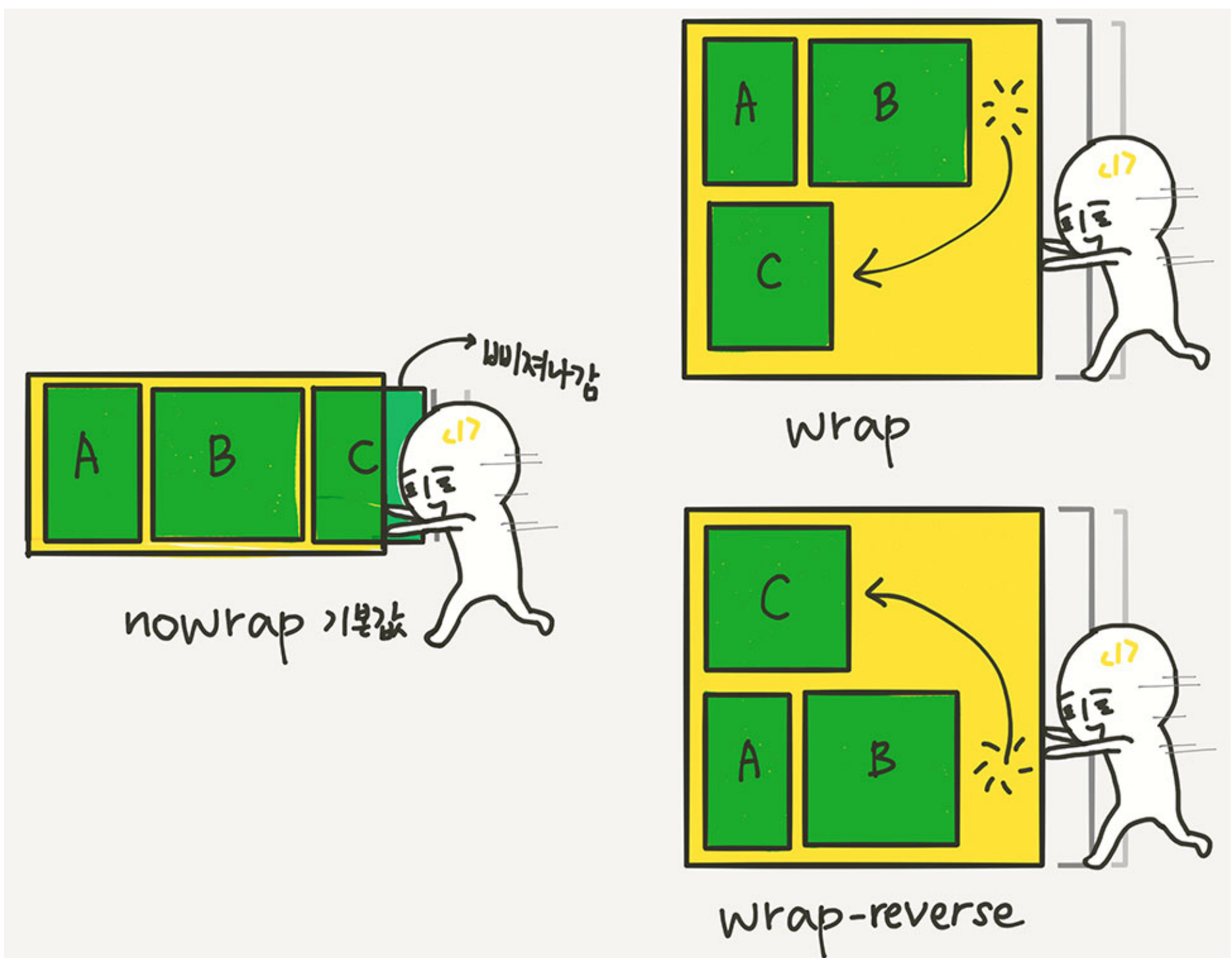
column-reverse



### 3. 줄넘김 처리 설정 - flex-wrap

컨테이너가 더 이상 아이템들을 한 줄에 담을 여유 공간이 없을 때  
아이템 줄바꿈을 어떻게 할지 결정하는 속성

```
.container {
  flex-wrap: nowrap;
  /* flex-wrap: wrap; */
  /* flex-wrap: wrap-reverse; */
}
```



nowrap(기본값)

● nowrap ○ wrap ○ wrap-reverse

줄바꿈 X

### wrap

○ nowrap ● wrap ○ wrap-reverse

줄바꿈O  
float이나 inline-block으로 배치한 요소들과 비슷하게 동작

### wrap-reverse

○ nowrap ○ wrap ● wrap-reverse

줄바꿈O  
아이템 역순 배치

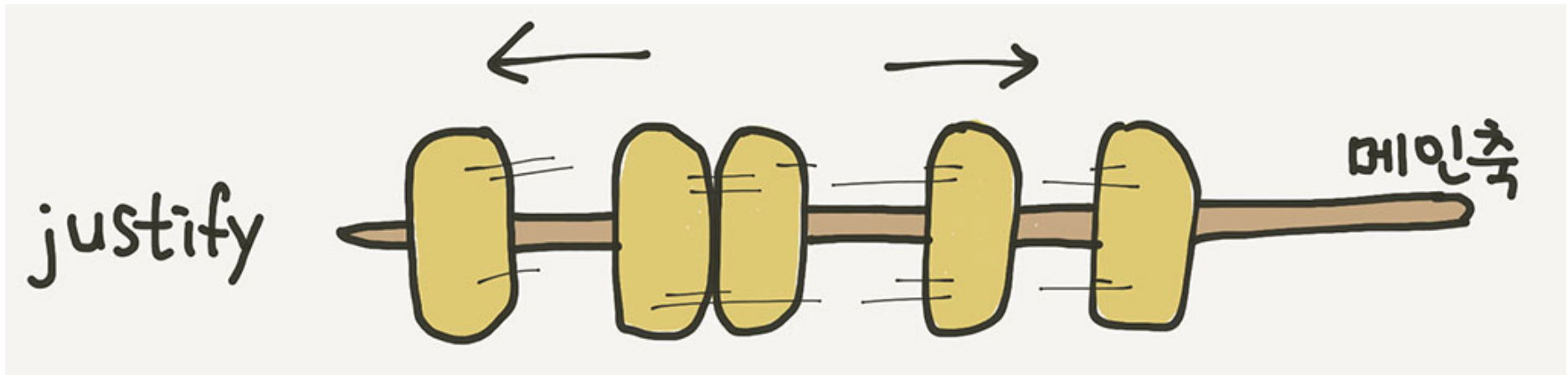
## 4. flex-flow

flex-direction과 flex-wrap을 한꺼번에 지정할 수 있는 단축 속성

```
.container {
  flex-flow: row wrap;
  /* 아래의 두 줄을 줄여 쓴 것 */
  /* flex-direction: row; */
  /* flex-wrap: wrap; */
}
```

## 5. justify-content

justify : 메인축 방향으로 정렬



#### • 4-1. 메인축 방향 정렬 - justify-content

```
.container {
  justify-content: flex-start;
  /* justify-content: flex-end; */
  /* justify-content: center; */
  /* justify-content: space-between; */
  /* justify-content: space-around; */
  /* justify-content: space-evenly; */
}
```

##### flex-start(기본값)

AAA

BBBBBBBBBBBB

CCCCC

☒ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☐ space-evenly

아이템들을 시작점으로 정렬

##### flex-end

AAA

BBBBBBBBBBBB

CCCCC

☐ flex-start
 ☒ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☐ space-evenly

아이템들을 끝점으로 정렬

##### center

AAA

BBBBBBBBBBBB

CCCCC

☐ flex-start
 ☐ flex-end
 ☒ center
 ☐ space-between
 ☐ space-around
 ☐ space-evenly

아이템들을 가운데로 정렬

##### space-between

AAA

BBBBBBBBBBBB

CCCCC

☐ flex-start
 ☐ flex-end
 ☐ center
 ☒ space-between
 ☐ space-around
 ☐ space-evenly

아이템들의 "사이"에 균일한 간격

##### space-around

☐ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☒ space-around
 ☐ space-evenly

아이템들의 "둘레"에 균일한 간격

space-evenly

☐ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☒ space-evenly

아이템들의 "사이와 양 끝"에 균일한 간격

space-evenly는 IE와 엣지(Edge)에서는 지원되지 않음

space-between

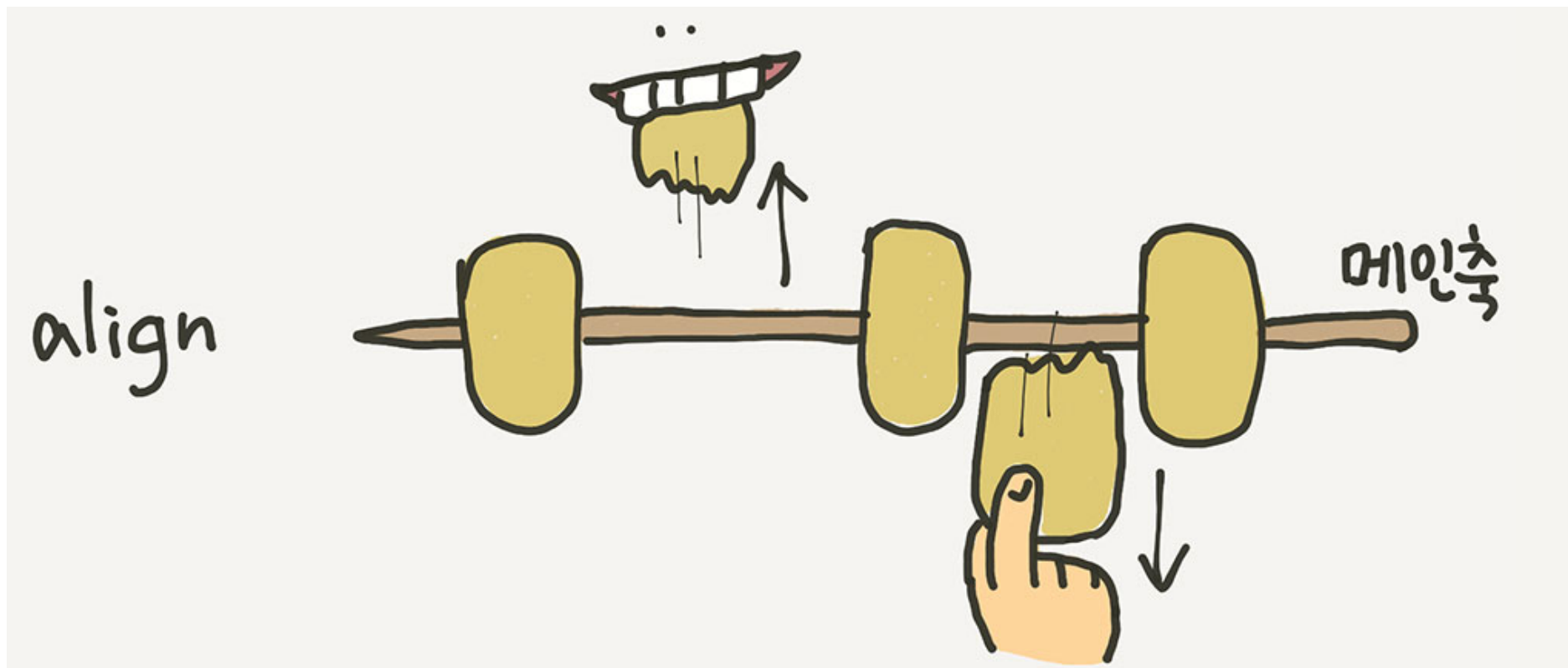
space-around

space-evenly

6. align-items, align-content

align : 수직축 방향으로 정렬

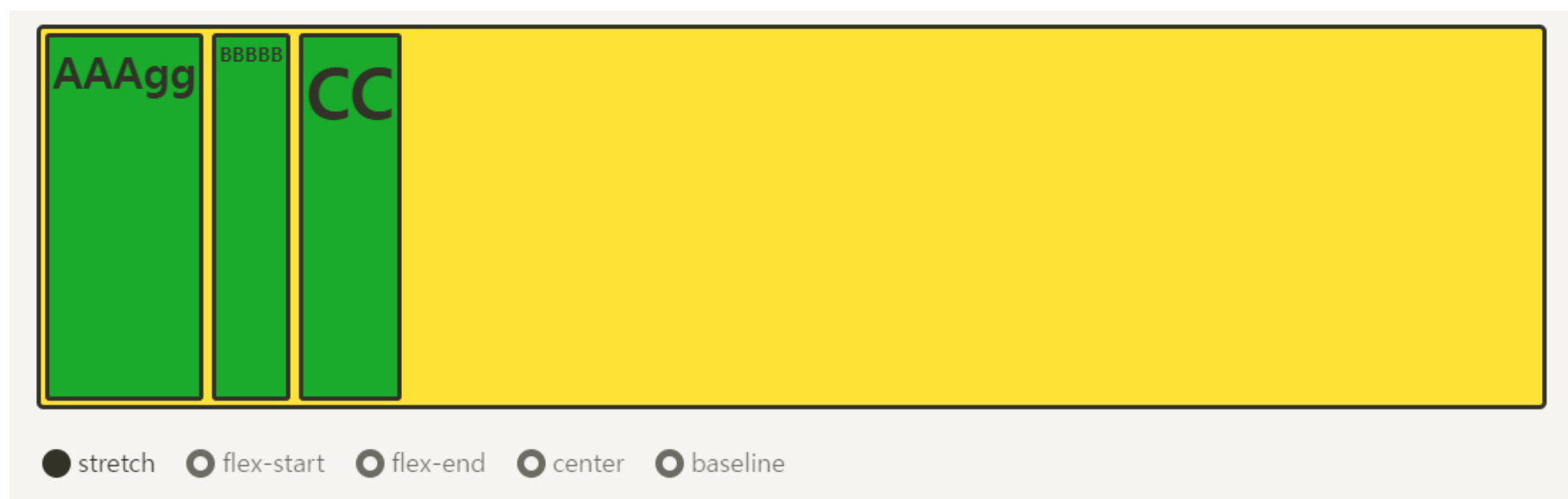




#### • 4-2. 수직축 방향 정렬 - align-items

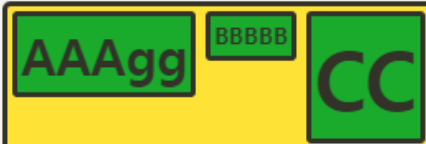
```
.container {
  align-items: stretch;
  /* align-items: flex-start; */
  /* align-items: flex-end; */
  /* align-items: center; */
  /* align-items: baseline; */
}
```

#### stretch (기본값)



#### flex-start





☐ stretch ☒ flex-start ☐ flex-end ☐ center ☐ baseline

아이템들을 시작점으로 정렬

## flex-end



☐ stretch ☐ flex-start ☒ flex-end ☐ center ☐ baseline

아이템들을 끝으로 정렬

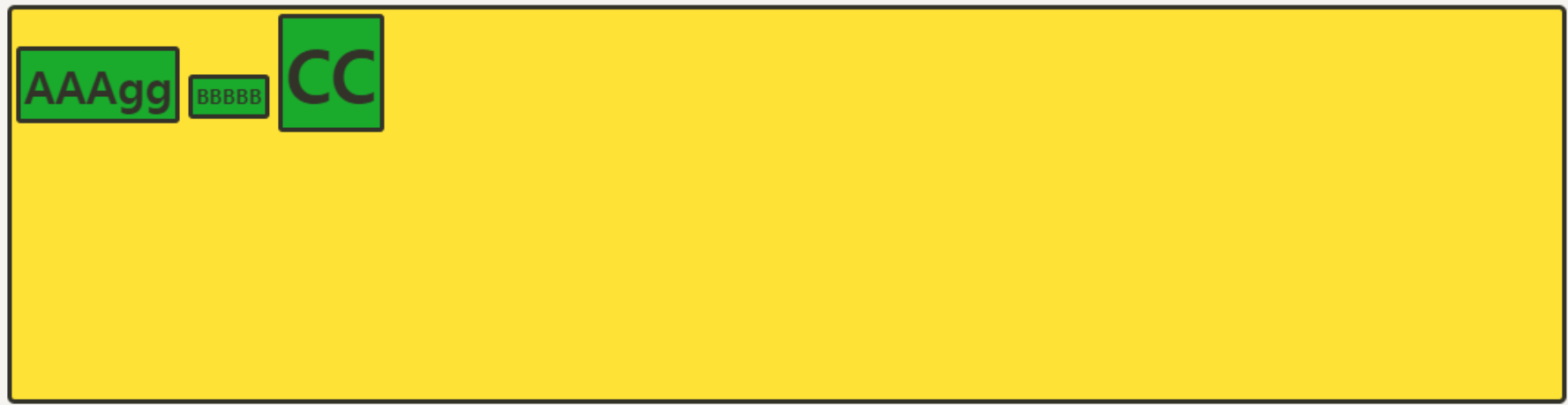
## center



☐ stretch ☐ flex-start ☐ flex-end ☒ center ☐ baseline

아이템들을 가운데로 정렬

## baseline



☐ stretch
 ☐ flex-start
 ☐ flex-end
 ☐ center
 ☒ baseline

아이템들을 텍스트 베이스라인 기준으로 정렬

만약,

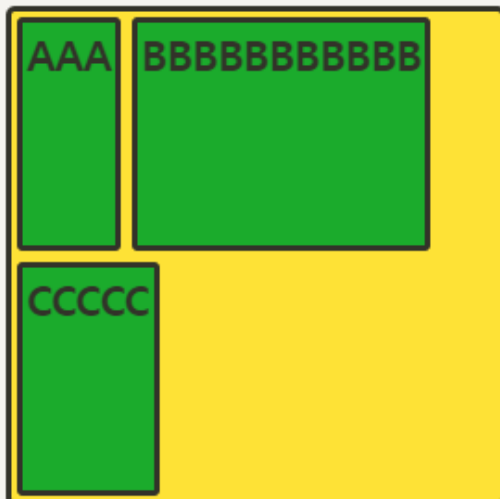
`justify-content: center;`

`align-items: center;`

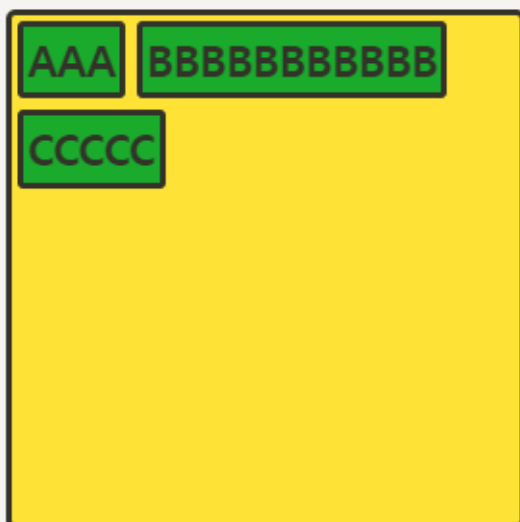
를 해주면, 아이템을 한 가운데에 위치시킬 수 있음

#### • 4-3. 여러 행 정렬 - align-content

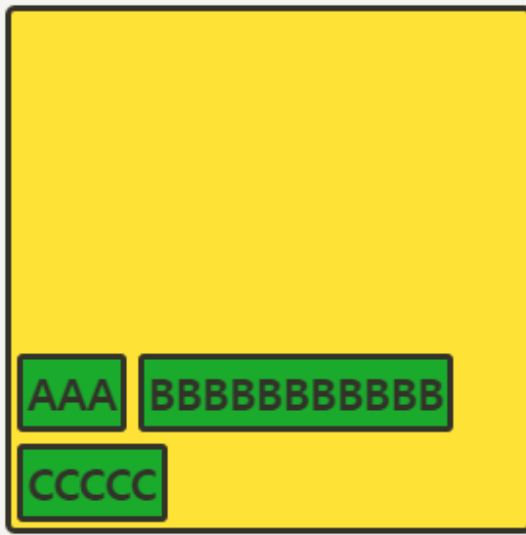
`flex-wrap: wrap;` 이 설정된 상태에서, 아이템들의 행이 2줄 이상 되었을 때의 수직축 방향 정렬을 결정



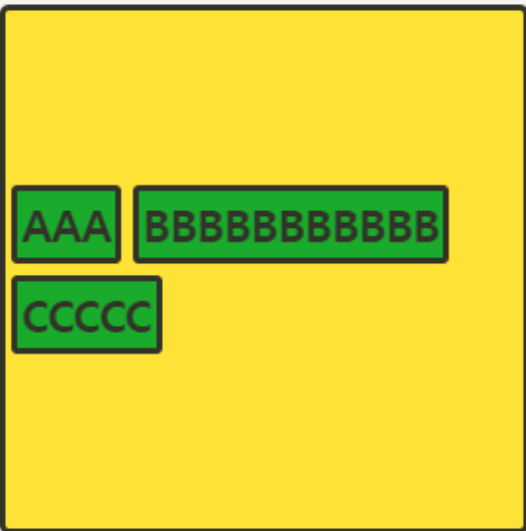
☒ stretch
 ☐ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☐ space-evenly



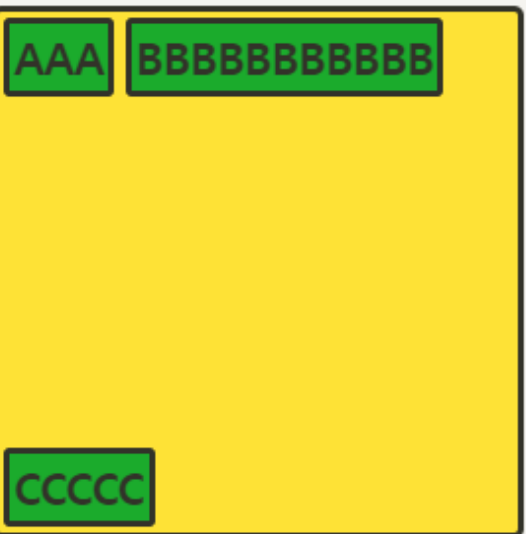
☐ stretch
 ☒ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☐ space-evenly



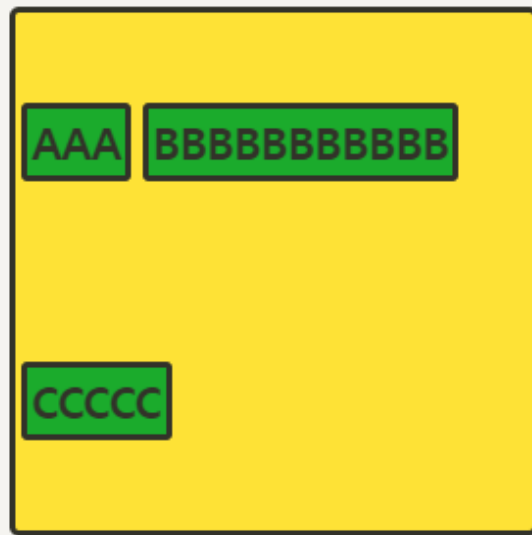
☐ stretch ☐ flex-start ☒ flex-end ☐ center ☐ space-between ☐ space-around ☐ space-evenly



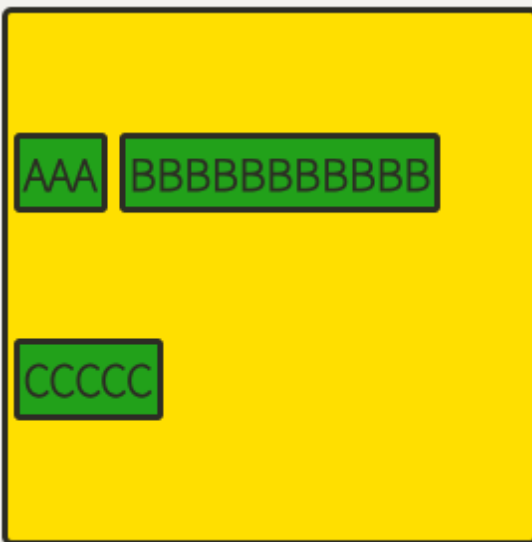
☐ stretch ☐ flex-start ☐ flex-end ☒ center ☐ space-between ☐ space-around ☐ space-evenly



☐ stretch ☐ flex-start ☐ flex-end ☐ center ☒ space-between ☐ space-around ☐ space-evenly



☐ stretch
 ☐ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☒ space-around
 ☐ space-evenly



☐ stretch
 ☐ flex-start
 ☐ flex-end
 ☐ center
 ☐ space-between
 ☐ space-around
 ☒ space-evenly

## <Flex 아이템에 적용하는 속성들>

### 1. 유연한 박스의 기본영역 - flex-basis

아이템의 기본 점유 크기를 설정 (flex-direction이 row일 때는 너비, column일 때는 높이)

- flex-basis의 값으로는 우리가 width, height 등에 사용하는 각종 단위의 수가 들어갈 수 있음.
- 기본값 auto는 해당 아이템의 width 값을 사용.
- width를 따로 설정하지 않으면 콘텐츠의 크기가 됨.

```

.item {
  flex-basis: auto; /* 기본값 */
  /* flex-basis: 0; */
  /* flex-basis: 50%; */
  /* flex-basis: 300px; */
  /* flex-basis: 10rem; */
  /* flex-basis: content; */
}
  
```

```
.item {
    flex-basis: 100px;
}
```

원래의 width가 100px이 안되는 AAA와 CCC는 100px로 늘어났고, 원래 100px이 넘는 BBB는 그대로 유지되죠~



반면에 width를 설정하면, 원래 100px을 넘는 BBB도 100px로 맞춰집니다.

(아래처럼 BBBBBBBBBBBB가 다음 줄로 넘어가도록 하려면, CSS에 **word-wrap: break-word;**를 적용해주세요. 안그러면 영역만 100px로 줄어들고 BBBBBB는 옆으로 쪽-빠져나간답니다.)

```
.item {
    width: 100px;
}
```



둘 다 설정하면?

```
.item {
    flex-basis: 100px;
    width: 100px;
}
```



## 2. 유연하게 늘리기 - flex-grow

**flex-grow**는 아이템이 **flex-basis**의 값보다 커질 수 있는지를 결정하는 속성

flex-grow에는 숫자값이 들어가는데, 몇이든 일단 0보다 큰 값이 세팅이 되면

해당 아이템이 유연한 박스로 변하고 원래의 크기보다 커지며 빈 공간을 매우게 됨.

- **flex-grow에 0을 세팅한 경우**

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

    /* flex-grow: 0; */ /* 기본값 */
}
```



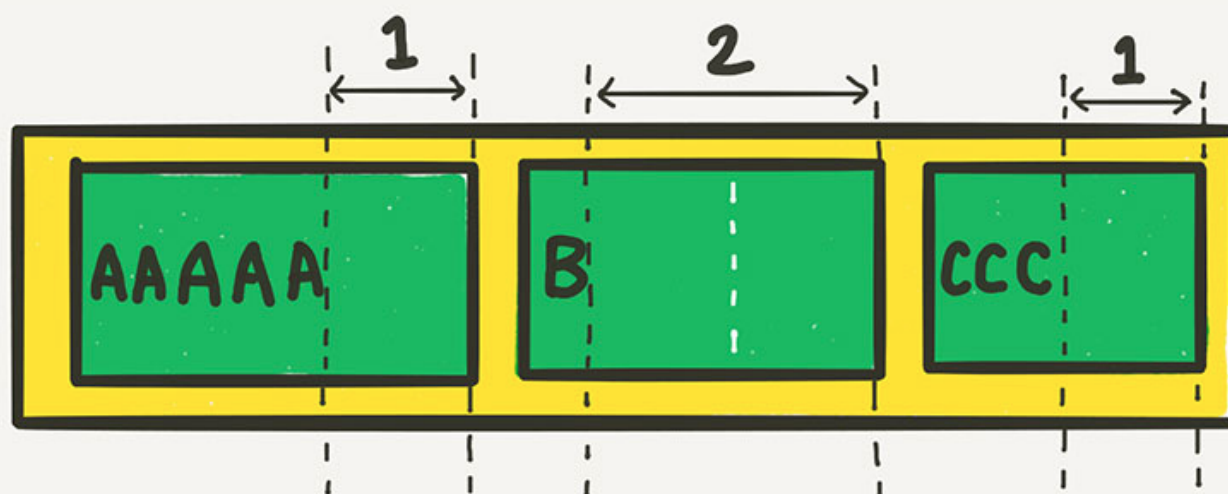
- flex-grow에 1을 세팅한 경우

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

    /* flex-grow: 0; */ /* 기본값 */
}
```



```
/* 1:2:1의 비율로 세팅할 경우 */
.item:nth-child(1) { flex-grow: 1; }
.item:nth-child(2) { flex-grow: 2; }
.item:nth-child(3) { flex-grow: 1; }
```



### 3. 유연하게 줄이기 - flex-shrink

flex-shrink는 flex-grow와 쌍을 이루는 속성으로, 아이템이 flex-basis의 값보다 작아질 수 있는지를 결정함.

flex-shrink에는 숫자값이 들어가는데, 몇이든 일단 0보다 큰 값이 세팅이 되면

해당 아이템이 유연한(Flexible) 박스로 변하고 flex-basis보다 작아짐.

기본값이 1이기 때문에 따로 세팅하지 않았어도 아이템이 flex-basis보다 작아질 수 있었음.

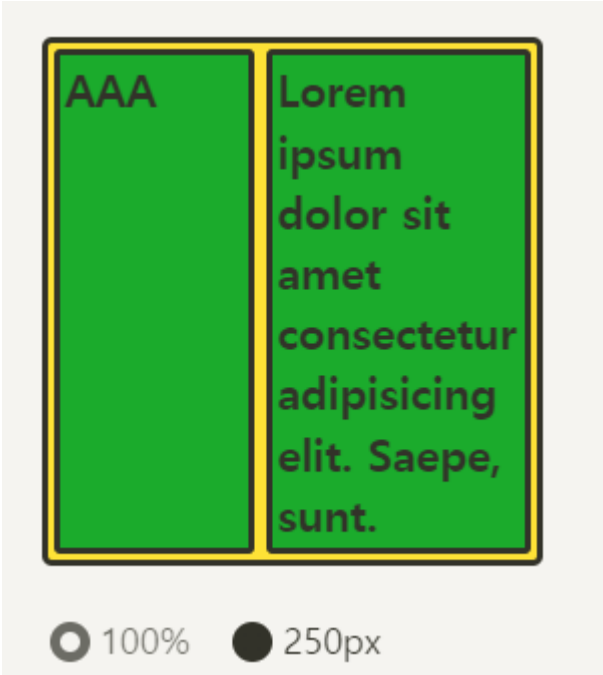
flex-shrink를 0으로 세팅하면 아이템의 크기가 flex-basis보다 작아지지 않기 때문에 고정폭의 컬럼을 쉽게 만들 수 있음.

```
.container {
  display: flex;
}
.item:nth-child(1) {
  flex-shrink: 0;
  width: 100px;
}
.item:nth-child(2) {
  flex-grow: 1;
}
```

- 컨테이너 폭 : 100%



- 컨테이너 폭 : 250px



컨테이너의 폭을 100%와 250px로 막 변경해도 flex-shrink:0 ; 덕분에 컨테이너가 아무리 작아져도 첫번째 아이템은 찌그러지지 않고 폭이 100px로 유지

## 4. flex


flex-grow, flex-shrink, flex-basis를 한번에 쓸 수 있는 축약형 속성

```
.item {
  flex: 1;
  /* flex-grow: 1; flex-shrink: 1; flex-basis: 0%; */
  flex: 1 1 auto;
  /* flex-grow: 1; flex-shrink: 1; flex-basis: auto; */
  flex: 1 500px;
  /* flex-grow: 1; flex-shrink: 1; flex-basis: 500px; */
}
```



ex)

```
.item {
    flex: 1 1 0;
}
.item:nth-child(2) {
    flex: 2 1 0;
}
```

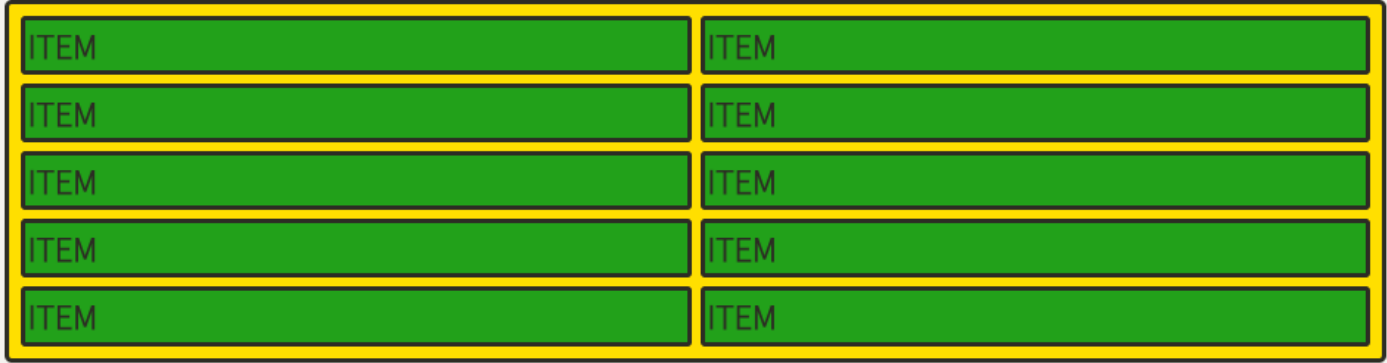


flex-basis: 0; 으로 기본 점유 크기를 0으로 만들어버려 결국 전체 크기를 1:2:1로 나누어 가져서,  
영역 자체의 크기가 정확히 1:2:1의 비율로 설정되었음

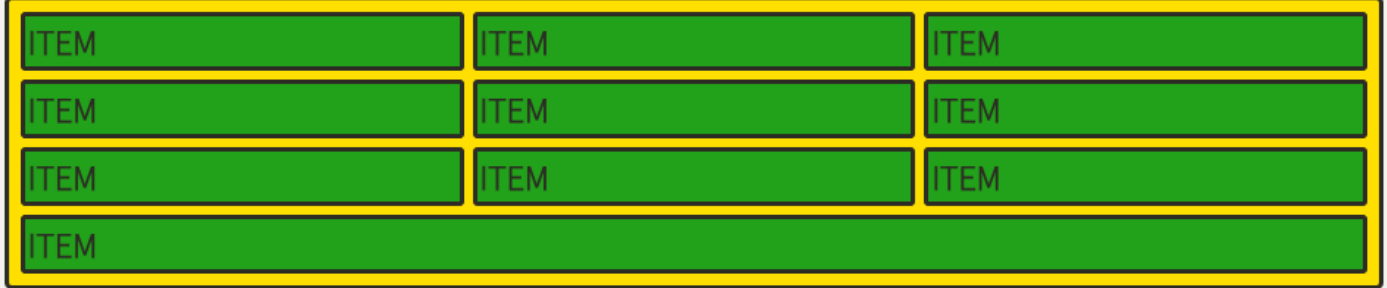
여백의 비가 아닌, 영역 자체를 원하는 비율로 분할하기를 원한다면 flex-basis를 0으로 하면 손쉽게 처리 가능

ex)

```
.container {
    display: flex;
    flex-wrap: wrap;
}
.item {
    flex: 1 1 40%;
}
```



☒ 1 1 40% ☐ 1 1 30%



☐ 1 1 40% ☒ 1 1 30%

## 5. 수직축으로 아이템 정렬 - align-self

```
.item {
    align-self: auto;
    /* align-self: stretch; */
}
```

```

/* align-self: flex-start; */
/* align-self: flex-end; */
/* align-self: center; */
/* align-self: baseline; */
}

```

align-self는 align-items보다 우선권이 있다.



align-self 값을 BBB는 center, CCC는 flex-start로 설정

## 6. 배치 순서 - order

```

.item:nth-child(1) { order: 3; } /* A */
.item:nth-child(2) { order: 1; } /* B */
.item:nth-child(3) { order: 2; } /* C */

```



## 7. z-index

숫자가 클수록 위로 올라옴

```

.item:nth-child(2) {
  z-index: 1;
  transform: scale(2);
}
/* z-index를 설정 안하면 0이므로, 1만 설정해도 나머지 아이템을 보다 위로 올라온다 */

```



## 정리)

```

/* 활용도 높은 것들 */

.parent {

```

```
display: flex;
flex-direction: row; /* (row/column/row-reverse/column-reverse) */
flex-wrap: wrap; /* (nowrap/wrap) */

justify-content: center; /* (flex-start/flex-end/center/space-between/space-around) */
align-items: center; /* (stretch/flex-start/flex-end/center/baseline) */
}

.child {
  flex: <grow(팽창 지수)> <shrink(수축 지수)> <basis(기본 크기)>
}
```