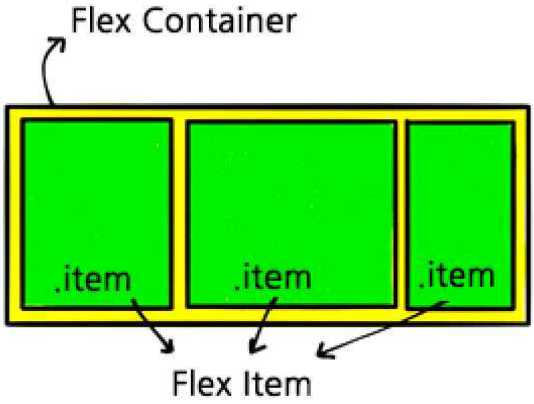
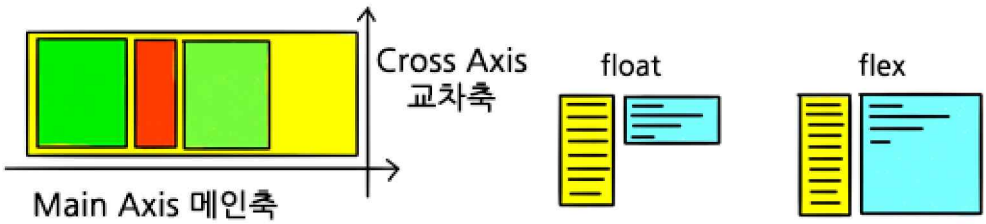
CSS Flexbox

float 또는 position을 사용하지 않고도 유연한 반응형 레이아웃 구조를 쉽게 설계할 수 있다.

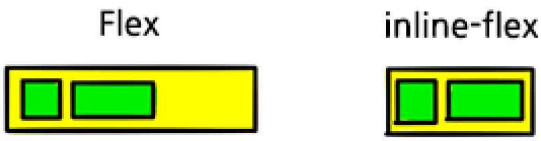
```
<div class="flex-container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
</div>
```



display: flex;



display: inline-flex;

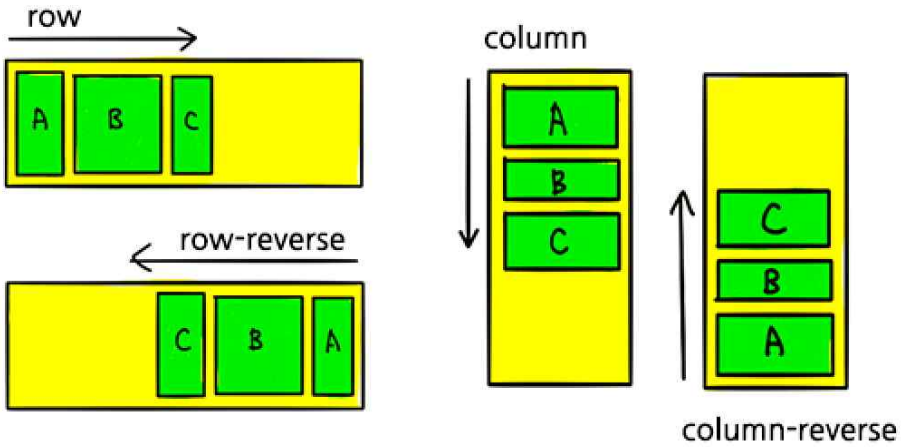


아이템들이 배치된 방향의 축을 메인축(Main Axis)
메인축과 수직인 축을 수직축 또는 교차축(Cross Axis)

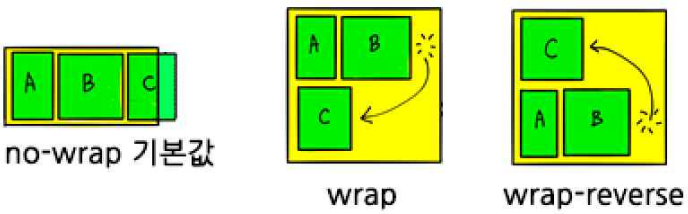
Flex 컨테이너에 적용하는 속성들

1. flex-direction
2. flex-wrap
3. flex-flow
4. justify-content
5. align-items
6. align-content

flex-direction : 컨테이너가 플렉스 항목의 방향을 정의
column – 플렉스 항목을 열(세로)방향으로 배치
column-reverse – 플렉스 항목을 역순으로 세로 배치
row(기본값) - 플렉스 항목을 가로로(왼쪽에서 오른쪽으로) 배치.
row-reverse – 플렉스 항목을 역순으로 가로 배치



flex-wrap 줄넘김 처리 설정
컨테이너가 더 이상 아이템들을 한줄에 담을 여유 공간이 없을 때 아이템 줄바꿈을 어떻게 할지 결정하는 속성
nowrap (기본값)
줄바꿈을 하지 않습니다. 넘치면 그냥 삐져 나감.
wrap
줄바꿈을 합니다. float이나 inline-block으로 배치한 요소들과 비슷하게 동작
wrap-reverse
줄바꿈을 하는데, 아이템을 역순으로 배치



flex-flow
flex-direction과 flex-wrap을 한꺼번에 지정할 수 있는 단축 속성이에요.
flex-direction, flex-wrap의 순으로 한 칸 떼고 기입.

justify-content속성은 플렉스 항목을 정렬하는 데 사용(메인축)
flex-start (기본값)
아이템들을 시작점으로 정렬
flex-direction이 row(가로 배치)일 때는 왼쪽, column(세로 배치)일 때는 위

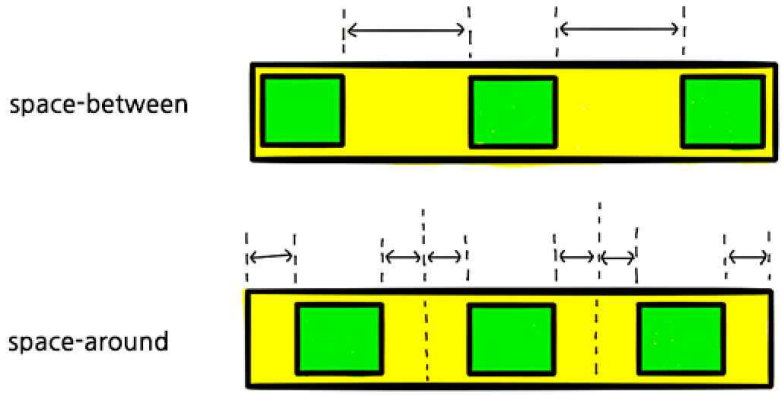
flex-end
아이템들을 끝점으로 정렬
flex-direction이 row(가로배치)일 때는 오른쪽, column(세로배치)일 때는 아래

center
아이템들을 가운데로 정렬합니다.

space-between
아이템들의 "사이(between)"에 균일한 간격

space-around
아이템들의 "둘레(around)"에 균일한 간격

space-evenly
항목 주위에 균일한 간격



align-items : 수직축 방향 정렬
stretch (기본값)
아이템들이 수직축 방향으로 끝까지 쭈욱 늘어남.

flex-start
아이템들을 시작점으로 정렬
flex-direction이 row(가로 배치)일 때는 위, column(세로 배치)일 때는 왼쪽

flex-end
아이템들을 끝으로 정렬합니다.

flex-direction이 row(가로 배치)일 때는 아래, column(세로 배치)일 때는 오른쪽

center
아이템들을 가운데로 정렬합니다.

baseline
아이템들을 텍스트 베이스라인 기준으로 정렬합니다.

align-content - 여러 행 정렬
flex-wrap: wrap;이 설정된 상태에서, 아이템들의 행이 2줄 이상 되었을 때의 수직축 방향 정렬을 결정하는 속성

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

Flex 아이템에 적용하는 속성들

1. flex-basis
2. flex-grow
3. flex-shrink
4. flex
5. align-self
6. order

flex-basis는 Flex 아이템의 기본 크기를 설정합니다(flex-direction이 row일 때는 너비, column일 때는 높이).

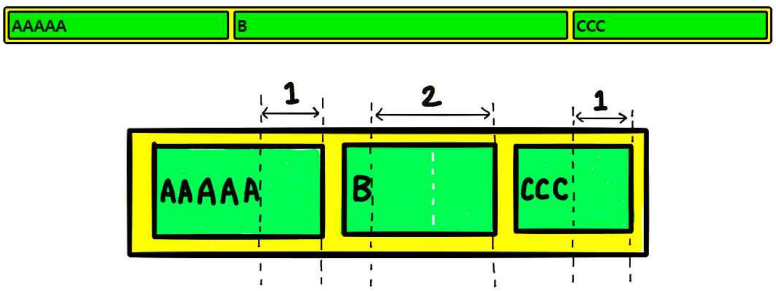
```
.item {
        flex-basis: 100px;
}
```

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

flex-grow – 유연하게 늘리기

아이템들의 flex-basis를 제외한 여백 부분을 flex-grow에 지정된 숫자의 비율로 나누어 가진다.
flex-grow는 아이템이 flex-basis의 값보다 커질 수 있는지를 결정하는 속성
flex-grow에는 숫자값이 들어가는데, 몇이든 일단 0보다 큰 값이 세팅이 되면 해당 아이템이 유연한(Flexible) 박스로 변하고 원래의 크기보다 커지며 빈 공간을 메우게 됩니다.
기본값: 0

```
/* 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; }
```

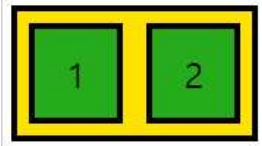


flex-shrink - 유연하게 줄이기
flex-shrink는 flex-grow와 반대 속성으로, 아이템이 flex-basis의 값보다 작아질 수 있는지를 결정
flex-shrink에는 숫자값이 들어가는데, 몇이든 일단 0보다 큰 값이 세팅이 되면 해당 아이템이 유연한(Flexible) 박스로 변하고 flex-basis보다 작아집니다.
기본값이 1이기 때문에 따로 세팅하지 않았어도 아이템이 flex-basis보다 작아질 수 있다.

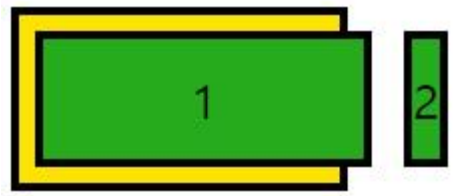
```
.flex-container {
        display: flex;
}
.item {
        flex-basis: 200px;
}
```


```
.flex-container {
   display: flex;
   width: 200px;
}
.item {
   flex-basis: 200px;
```

```
}
```



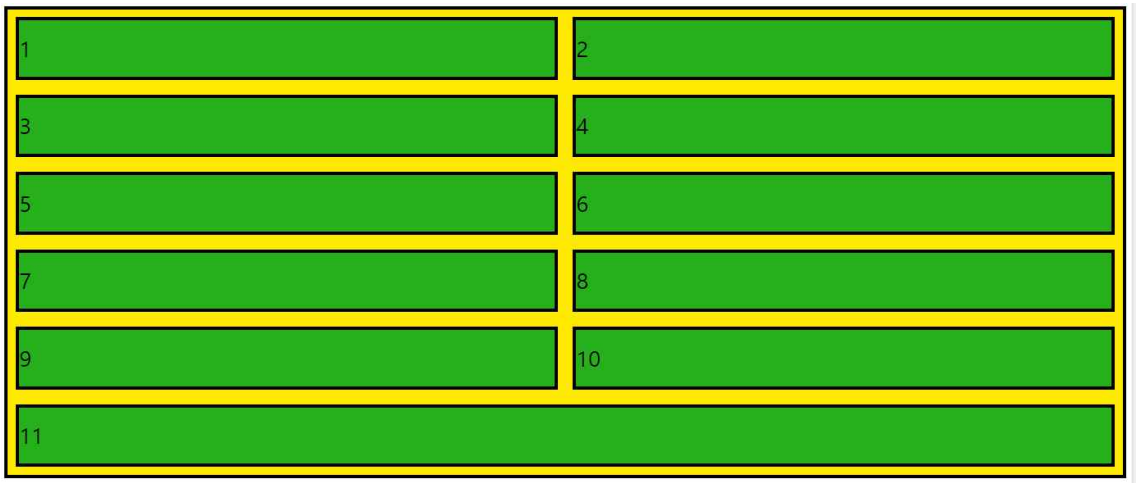

```
.flex-container {
   display: flex;
   width: 200px;
}
.item {
   flex-basis: 200px;
}
.item:first-child {
     flex-shrink: 0;
}
```



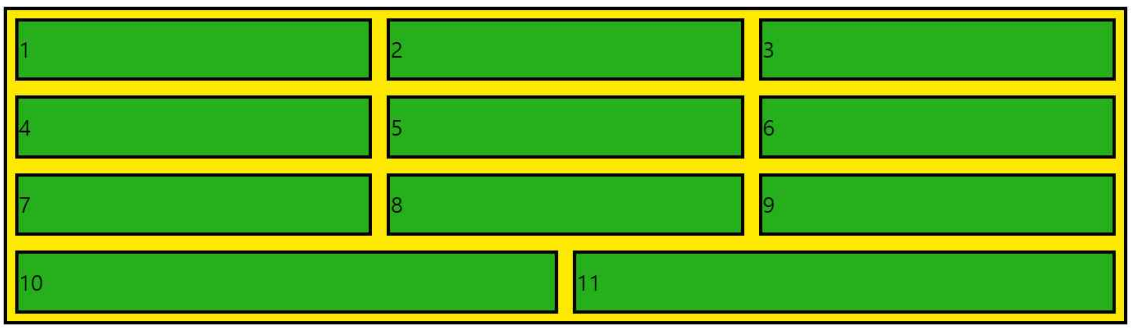
flex
flex-grow, flex-shrink, flex-basis를 한 번에 쓸 수 있는 축약형 속성

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
   display: flex;
   flex-wrap: wrap;
   background-color: #ffe300;
   border: 5px solid #000;
}
.flex-container > div {
   background-color: #26ab1c;
   color: #000;
   border: 5px solid #000;
   line-height: 75px;
   font-size: 30px;
   margin: 10px;
}
.item {
          flex: 1 1 40%;
}
```

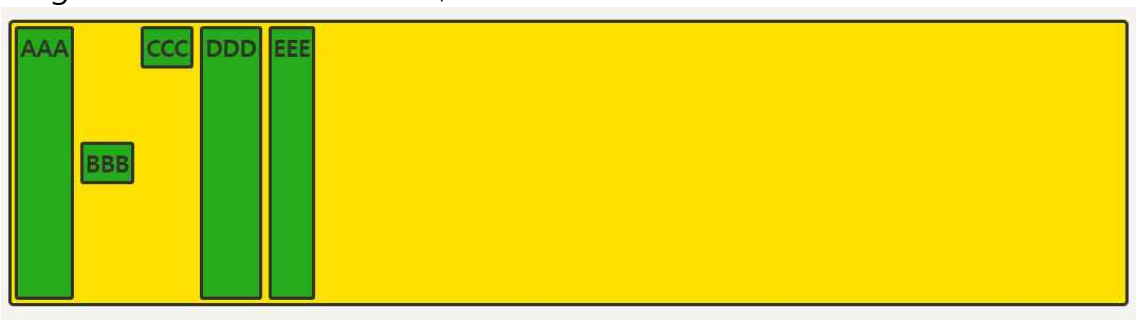
```
</style>
</head>
<body>
<div  class="flex-container">
   <div  class="item">1</div>
   <div  class="item">2</div>
   <div  class="item">3</div>
   <div  class="item">4</div>
   <div  class="item">5</div>
   <div  class="item">6</div>
   <div  class="item">7</div>
   <div  class="item">8</div>
   <div  class="item">9</div>
   <div  class="item">10</div>
   <div  class="item">11</div>
</div>
</body>
</html>
```




```
.item  {
          flex:  1  1  30%;
}
```



align-self
align-items가 전체 아이템의 수직축 방향 정렬이라면, align-self는 해당 아이템의 수직축 방향 정렬.
 align-self 값을 BBB는 center, CCC는 flex-start로 설정



order - 배치 순서
각 아이템들의 시각적 나열 순서를 결정하는 속성
숫자값이 들어가며, 작은 숫자일 수록 먼저 배치
기본값은 0

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