

# 11. 플렉스 박스 레이아웃

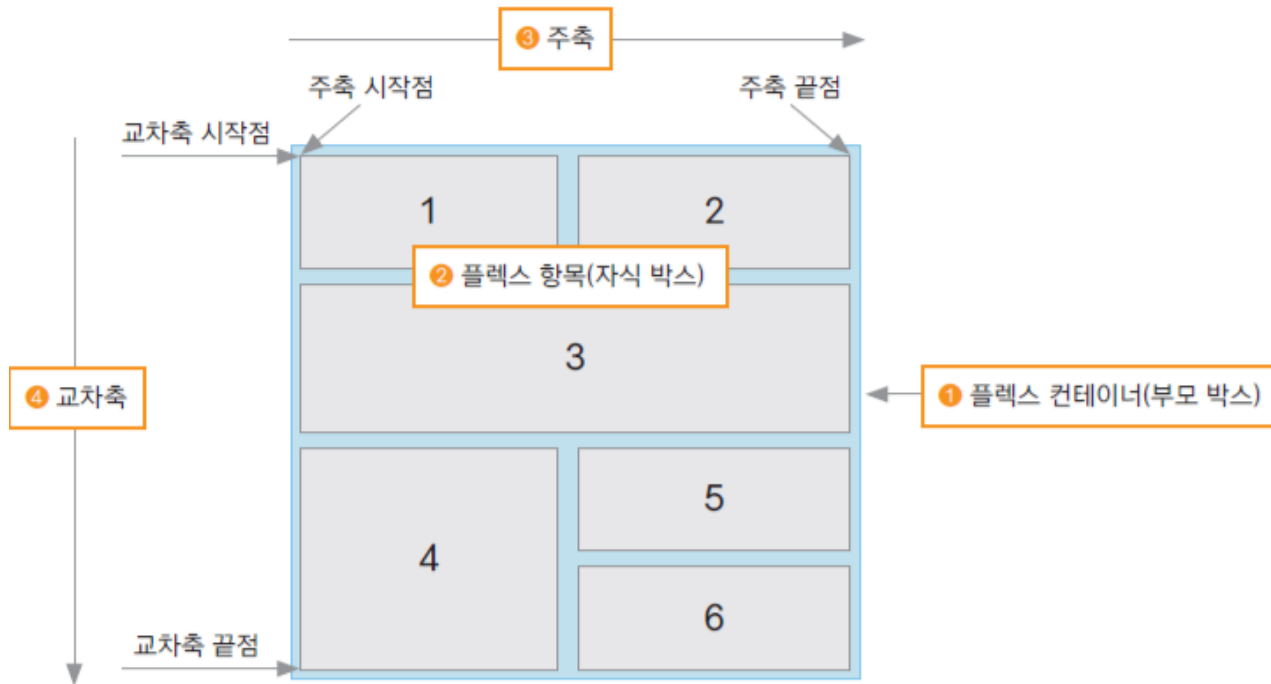


## 11-1 배치를 위한 기본 속성 살펴보기

## 11-2 반응형 웹을 위한 속성 살펴보기

# 플렉스 박스 레이아웃 (flex box layout) 에서 사용하는 용어

- 플렉스 박스를 원하는 위치에 배치하는 것.
- 여유 공간에 따라 너비나 높이, 위치를 자유롭게 변형할 수 있음



- 1 플렉스 컨테이너(부모 박스): 플렉스 박스 레이아웃을 적용할 대상을 묶는 요소입니다.
- 2 플렉스 항목(자식 박스): 플렉스 박스 레이아웃을 적용할 대상으로 그림 12-18에서 1~6까지 작은 박스들이 모두 해당합니다.
- 3 주축(main axis): 플렉스 컨테이너 안에서 플렉스 항목을 배치하는 기본 방향입니다. 기본적으로 왼쪽에서 오른쪽이며 수평 방향으로 배치합니다. 플렉스 항목의 배치가 시작되는 위치를 '주축 시작점', 끝나는 위치를 '주축 끝점'이라고 합니다.
- 4 교차축(cross axis): 주축과 교차하는 방향을 말하며 기본적으로 위에서 아래로 배치합니다. 플렉스 항목의 배치가 시작되는 위치를 '교차축 시작점', 끝나는 위치를 '교차축 끝점'이라고 합니다.

# 플렉스 박스 레이아웃

## 플렉서블 박스 레이아웃을 만드는 순서

- ① CSS를 사용해 적용할 대상을 플렉스 컨테이너로 지정
- ② 플렉스 컨테이너 안에 플렉스 항목 작성
- ③ CSS를 사용해 주축을 지정.  
가로로 배치할지(기본), 세로로 배치할지 결정  
주축을 가로로 했다면 교차축은 세로가 되고,  
주축을 세로로 했다면 교차축은 가로
- ④ CSS를 사용해 교차축의 배치 방법 지정

## 플렉스 컨테이너에서 사용하는 속성

속성값	설명
<b>justify-content</b>	주축의 정렬 방법
<b>align-items</b>	교차축의 정렬 방법
<b>align-content</b>	교차축에 여러 줄로 표시할 경우 정렬 방법

## 플렉스 항목에서 사용하는 속성

속성값	설명
<b>align-self</b>	플렉스 항목을 개별적으로 정렬

# 플렉스 박스 레이아웃 기본 속성

## display 속성

배치 요소들을 감싸는 부모 요소를 플렉스 컨테이너로 지정

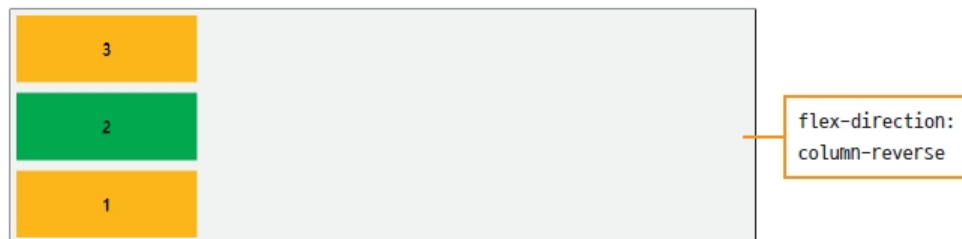
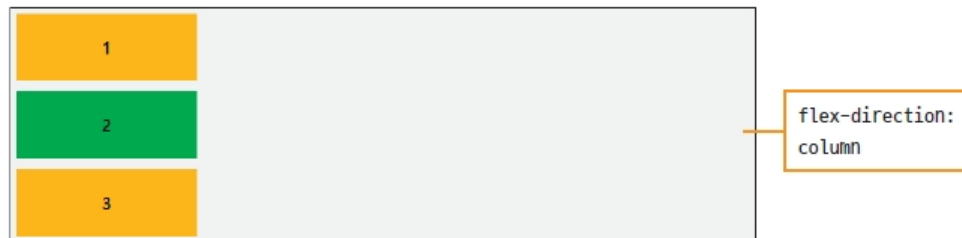
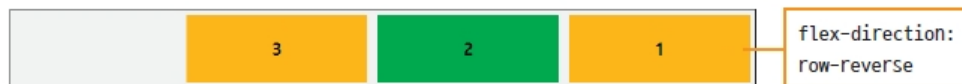
종류	설명
flex	컨테이너 안의 플렉스 항목을 블록 레벨 요소로 배치합니다.
inline-flex	컨테이너 안의 플렉스 항목을 인라인 레벨 요소로 배치합니다.

## flex-direction 속성

플렉스 항목의 배치를 위해 주축과 방향 지정.

종류	설명
row	주축을 가로로 지정하고 왼쪽에서 오른쪽으로 배치합니다. 기본값입니다.
row-reverse	주축을 가로로 지정하고 반대로 오른쪽에서 왼쪽으로 배치합니다.
column	주축을 세로로 지정하고 위쪽에서 아래쪽으로 배치합니다.
column-reverse	주축을 세로로 지정하고 아래쪽에서 위쪽으로 배치합니다.

```
<style>
  #opt1{ flex-direction: row; }      /* 왼쪽에서 오른쪽으로 */
  #opt2{ flex-direction: row-reverse; } /* 오른쪽에서 왼쪽으로 */
  #opt3{ flex-direction: column; }    /* 위에서 아래로 */
  #opt4{ flex-direction: column-reverse; } /* 아래에서 위로 */
</style>
```



# 플렉스 박스 레이아웃 기본 속성

## flex-wrap 속성

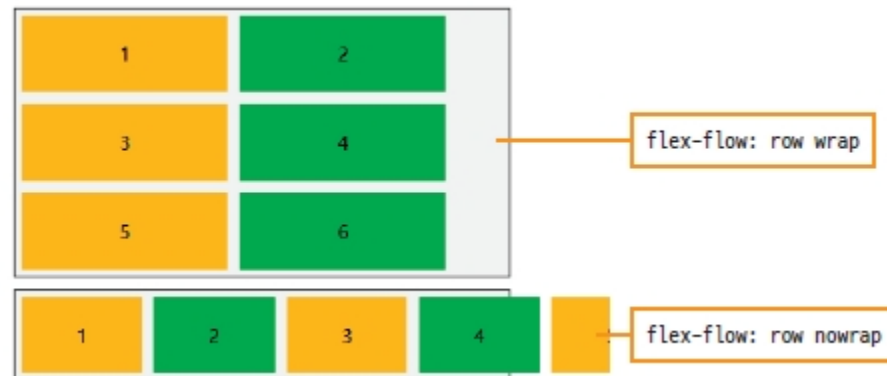
플렉스 항목을 한 줄 또는 여러 줄로 배치

종류	설명
nowrap	플렉스 항목을 한 줄에 표시합니다. 기본값입니다.
wrap	플렉스 항목을 여러 줄에 표시합니다.
wrap-reverse	플렉스 항목을 여러 줄에 표시하되, 시작점과 끝점이 바뀝니다.

## flex-flow 속성

- 플렉스 배치 방향과 여러 줄 배치를 한꺼번에 지정
- 기본 값은 flex-flow: row nowrap

```
<style>
  #opt1 { flex-flow: row wrap; }
  #opt2 { flex-flow: row nowrap; }
</style>
```



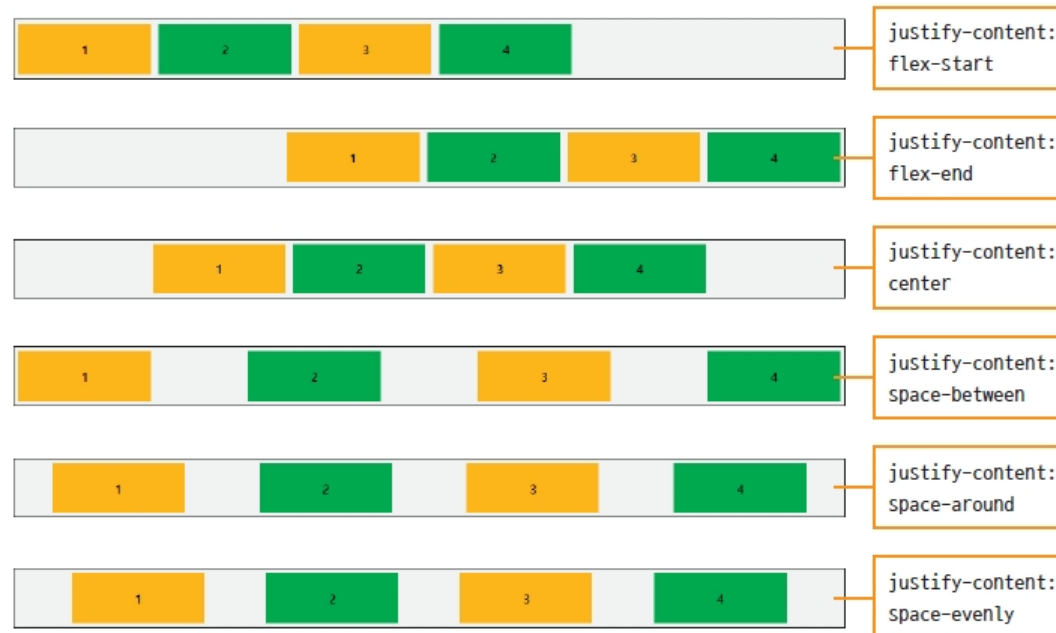
# 플렉스 박스 레이아웃 기본 속성

## justify-content 속성

플렉스 항목을 주축 방향으로 배치할 때의 배치 기준

속성값	설명
<b>flex-start</b>	주축의 시작점에 맞춰 배치. 기본값
<b>flex-end</b>	주축의 끝점에 맞춰 배치
<b>center</b>	주축의 중앙에 맞춰 배치
<b>space-around</b>	항목을 고르게 정렬. 각 항목은 양쪽 여백의 절반만큼 자리 차지
<b>space-between</b>	첫 번째 항목은 주축 시작점에, 마지막 항목은 주축 끝점에 배치한 후 나머지 항목은 같은 간격으로 배치
<b>space-evenly</b>	항목을 고르게 정렬. 각 항목의 여백은 모두 동일

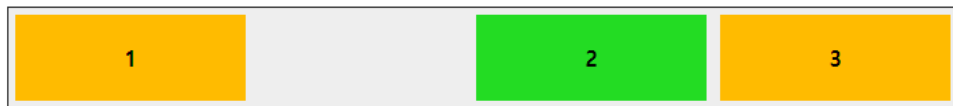
```
<style>
#opt1{ justify-content: flex-start; }
#opt2{ justify-content: flex-end; }
#opt3{ justify-content: center; }
#opt4{ justify-content: space-between; }
#opt5{ justify-content: space-around; }
#opt6{ justify-content: space-evenly; }
</style>
```



# justify-content 속성과 margin 속성 함께 사용하기

- 주축에 플렉스 항목을 한 줄로 배치할 때 justify-content 속성과 margin 속성을 함께 사용하면 좀 더 다양하게 배치할 수 있다.
- justify-content 속성은 플렉스 컨테이너에 사용하고 margin 속성은 플렉스 항목에서 사용

```
<style>
.container {
  width:700px;
  display:flex;
  justify-content: flex-end;
}
#box1 { margin-right: auto; }
</style>
```



div#box1.box 170 × 63

마진

```
<style>
.container {
  width:700px;
  display:flex;
  justify-content: flex-end;
}
#box3 { margin-left: auto; }
</style>
```



div#box3.box 170 × 63

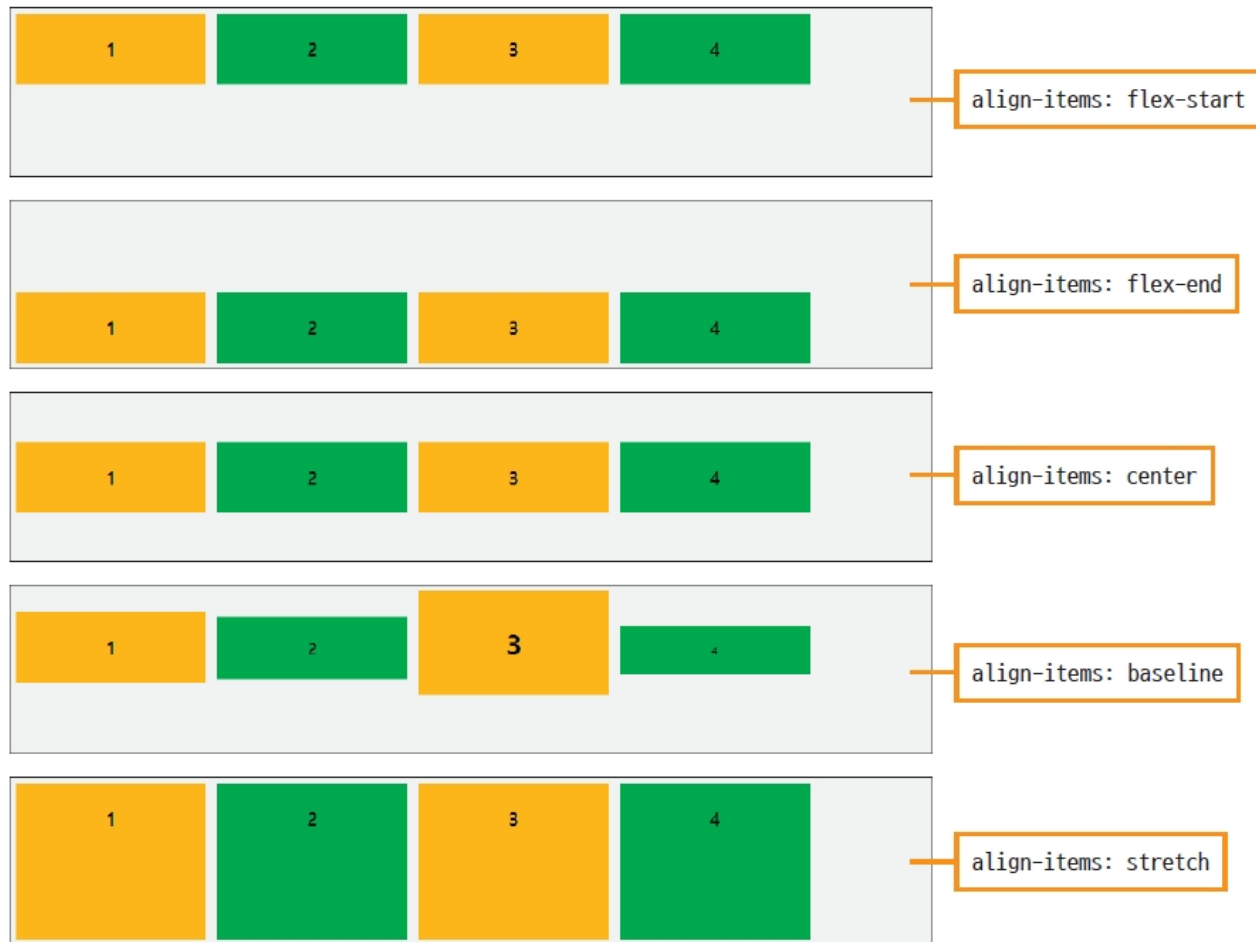
# 플렉스 박스 레이아웃 기본 속성

## align-items 속성

교차축의 정렬 방법

종류	설명
flex-start	교차축의 시작점에 맞춰 배치합니다.
flex-end	교차축의 끝점에 맞춰 배치합니다.
center	교차축의 중앙에 배치합니다.
baseline	교차축의 문자 기준선에 맞춰 배치합니다.
stretch	플렉스 항목을 늘려 교차축에 가득 차게 배치합니다.

```
<style>
#opt1{ align-items: flex-start; }
#opt2{ align-items: flex-end; }
#opt3{ align-items: center; }
#opt4{ align-items: baseline; }
#opt5{ align-items: stretch; }
</style>
```





# 플렉스 박스 레이아웃 기본 속성

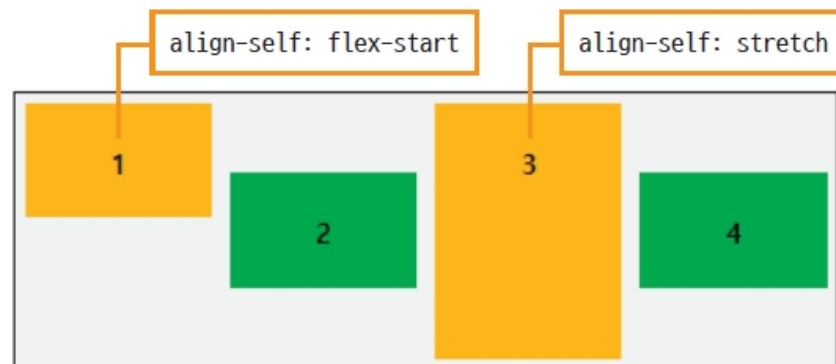
## align-self 속성

특정 플렉스 항목만 정렬 방법을 지정할 때 사용하는 속성

속성값	설명
auto	부모 요소의 align-items 값을 사용해 정렬. 기본값
normal	레이아웃 형태나 브라우저에 따라 다르게 정렬
flex-start	플렉스 컨테이너의 시작점에 맞춰 정렬
flex-end	플렉스 컨테이너의 끝점에 맞춰 정렬
self-start	플렉스 항목의 시작 위치에 맞춰 정렬. 텍스트를 포함하고 있을 경우 언어의 방향에 따라 시작 위치가 결정됨
self-end	플렉스 항목 자체의 끝에 맞춰 정렬. 텍스트를 포함할 경우 언어의 방향에 따라 끝 위치가 결정됨.
baseline	플렉스 항목에 있는 텍스트 기준선에 맞춰 정렬
center	플렉스 컨테이너의 가운데로 정렬
stretch	플렉스 컨테이너의 높이에 가득차게 늘려서 정렬

```
<style>
.container {
  display: flex;
  align-items: center;
}
#box1 { align-self: flex-start; }
#box3 { align-self: stretch; }
.....
```

```
<div class="container">
  <div class="box" id="box1"><p>1</p></div>
  <div class="box"><p>2</p></div>
  <div class="box" id="box3"><p>3</p></div>
  <div class="box"><p>4</p></div>
</div>
```



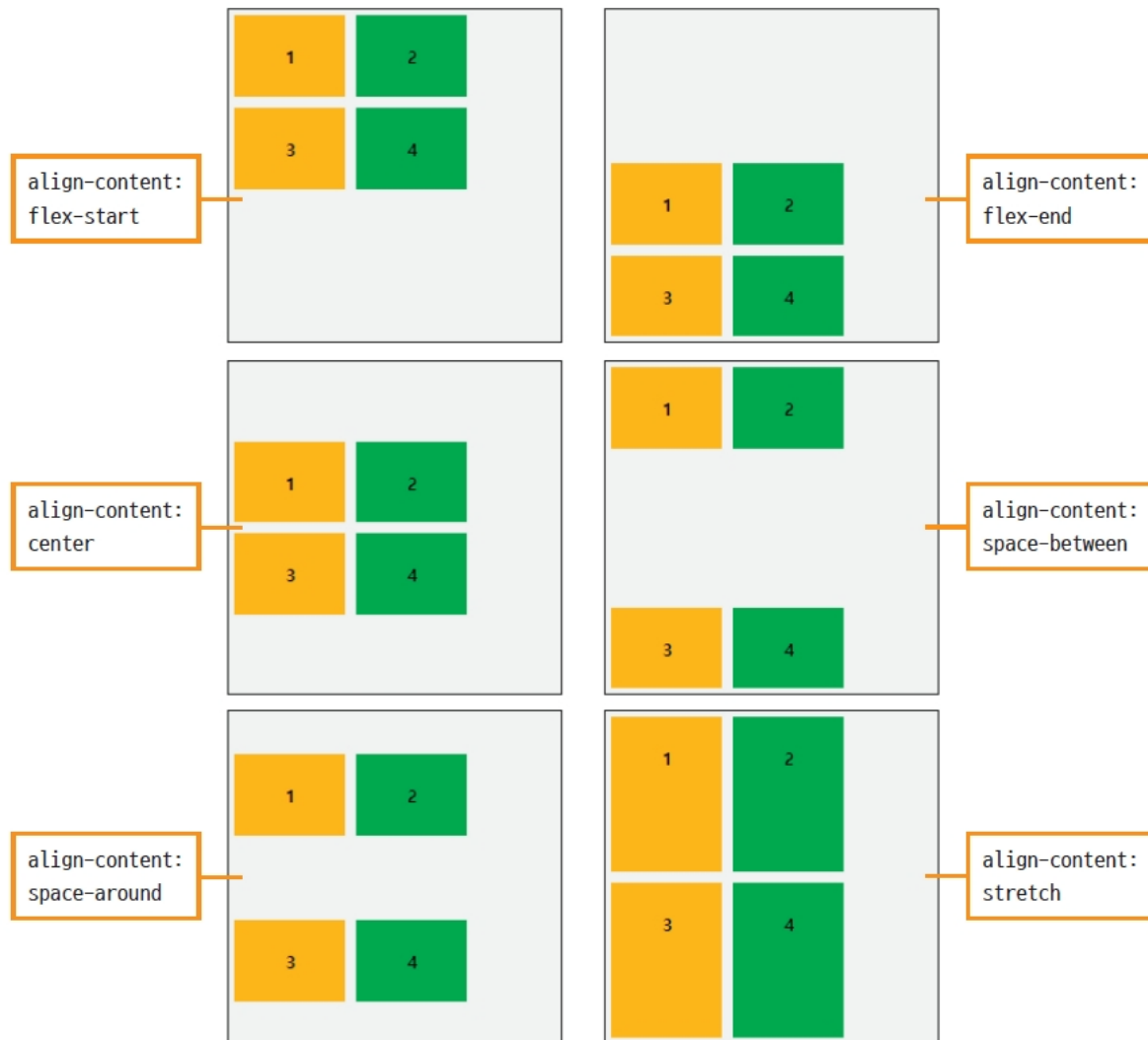
# 플렉스 박스 레이아웃 기본 속성

## align-content 속성

플렉스 항목이 여러 줄로 표시될 때 교차 축 기준의 배치 방법 지정

종류	설명
flex-start	교차축의 시작점에 맞춰 배치합니다.
flex-end	교차축의 끝점에 맞춰 배치합니다.
center	교차축의 중앙에 맞춰 배치합니다.
space-between	첫 번째 항목과 끝 항목을 교차축의 시작점과 끝점에 맞추고 나머지 항목은 그 사이에 같은 간격으로 배치합니다.
space-around	모든 항목을 교차축에 같은 간격으로 배치합니다.
stretch	플렉스 항목을 늘려서 교차축에 가득 차게 배치합니다.

```
<style>
#opt1 { align-content: flex-start; }
#opt2 { align-content: flex-end; }
#opt3 { align-content: center; }
#opt4 { align-content: space-between; }
#opt5 { align-content: space-around; }
#opt6 { align-content: stretch; }
</style>
```



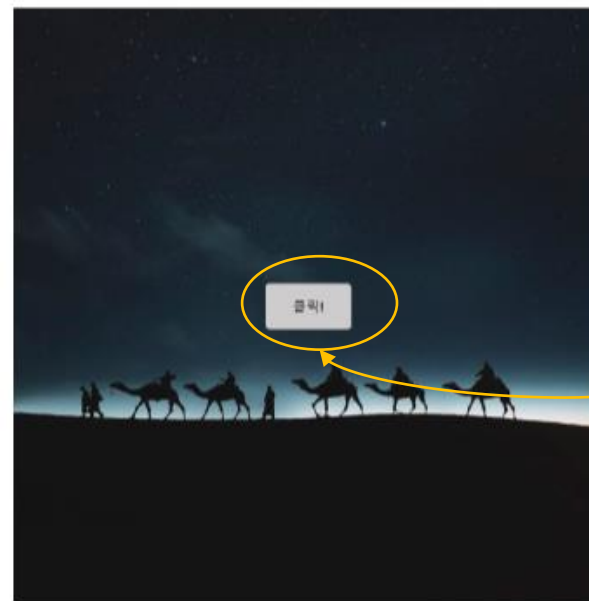
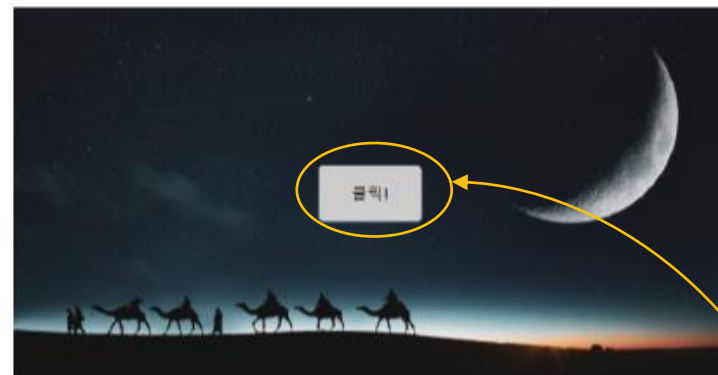
# 플렉스 박스 레이아웃 기본 속성

## 플렉스 박스 레이아웃을 사용해 화면 중앙에 배치하기

플렉스 박스 레이아웃을 가장 많이 사용하는 예

```
<style>
* {
  margin: 0;
  box-sizing: border-box;
}
body {
  background: url('images/bg5.jpg') no-repeat left top fixed;
  background-size: cover;
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
}
button { ..... }
</style>

<button>클릭!</button>
```



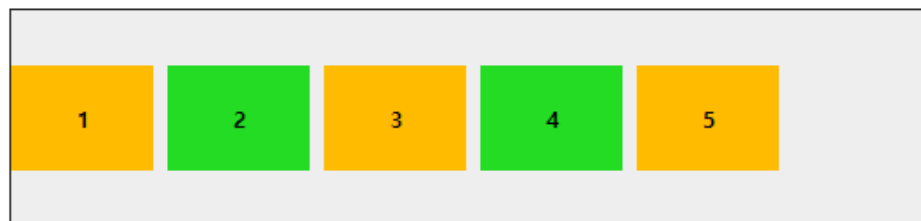
화면 크기에 상관없이  
버튼은 항상 화면 중앙  
에 표시됨

# 플렉스 박스 레이아웃 기본 속성

## gap 속성

- 플렉스 컨테이너 안에 있는 플렉스 항목들 간의 간격
- 인접한 항목이 있을 경우에만 간격이 생김(margin과의 차이점)
- 플렉스 컨테이너에서 적용
- row-gap과 column-gap으로 나누어 사용할 수도 있음

```
<style>
.container {
  width:650px;
  height:150px;
  display:flex;
  align-items:center;
  background-color:#eee;
  border:1px solid #222;
  gap: 10px; /* 플렉스 항목 간의 간격 10px */
}
.....
</style>
```



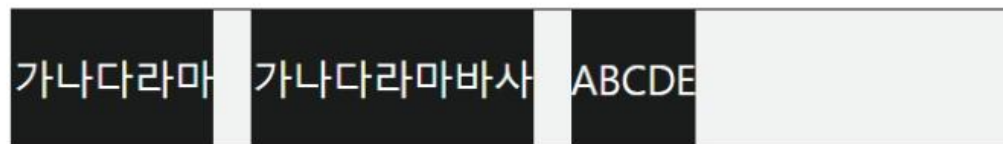
# 반응형 웹을 위한 속성 살펴보기

속성	설명
<b>flex-basis</b>	플렉스 항목의 기본 크기
<b>flex-grow</b>	남은 공간을 채우기 위해 플렉스 항목을 늘림
<b>flex-shrink</b>	공간이 부족할 경우 플렉스 항목을 줄임
<b>flex</b>	앞의 3가지 속성을 한꺼번에 지정

## flex-basis 속성

- 플렉스 항목의 기본 크기
- 기본값 (auto)
  - width 값이 지정되어 있다면 그 값 사용
  - width 값이 없다면 콘텐츠 영역만큼의 크기

```
<style>
.box {
  background-color:#222;
}
</style>
```



```
<style>
.box {
  background-color:#222;
  flex-basis: 150px; /* 플렉스 항목의 기본 크기 150px */
}
</style>
```



# 반응형 웹을 위한 속성 살펴보기

## flex-grow 속성

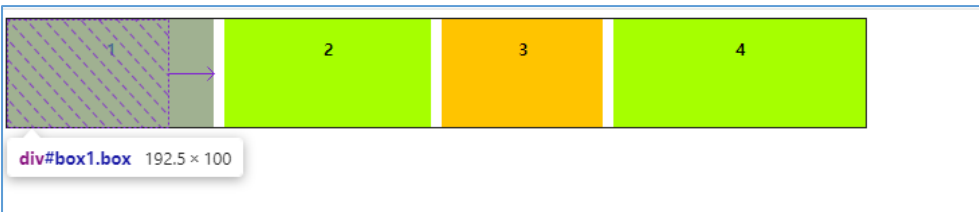
- 플렉스 컨테이너에 남는 공간이 있을 때 어떻게 나눌지 지정
- 기본값 0 (플렉스 항목을 늘리지 않음)

컨테이너 안에 여러 개의 flex-grow 값이 있을 경우

```
<style>
.container { width:800px; ..... }
.box { flex-basis: 150px; }
#box1 { flex-grow: 1; }
#box2 { flex-grow: 1; }
#box4 { flex-grow: 2; }
</style>
```

남는공간 =  $800 - (150 * 4) - (10 * 3)$

#box1의 너비 = 기본 크기 +  $\frac{1}{4} * \text{남는공간} = 192.5\text{px}$   
#box2의 너비 = 기본 크기 +  $\frac{1}{4} * \text{남는공간} = 192.5\text{px}$   
#box3의 너비 = 기본 크기 = 150px  
#box4의 너비 = 기본 크기 +  $\frac{2}{4} * \text{남는공간} = 235\text{px}$



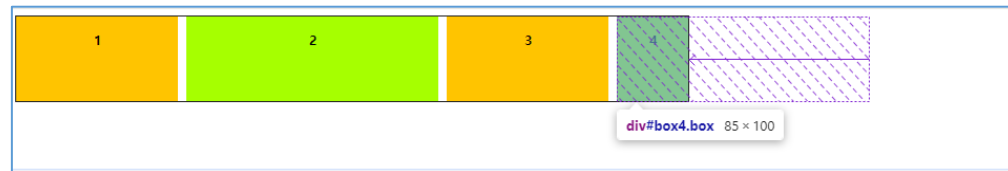
## flex-shrink 속성

- 플렉스 컨테이너 크기를 줄였을 때 플렉스 항목이 얼마나 줄어들지 지정
- 기본값 1 (플렉스 항목이 균등하게 줄어듦)

컨테이너 안에 여러 개의 flex-shrink 값이 있을 경우

```
<style>
.container { width:800px; ..... }
.box { flex-basis: 300px; }
#box1 { flex-shrink: 1; }
#box2 { flex-shrink: 0; }
#box3 { flex-shrink: 1; }
#box4 { flex-shrink: 2; }
</style>
```

#box1의 너비 =  $300 - 107.5 = 192.5\text{px}$   
#box3의 너비 =  $300 - 107.5 = 192.5\text{px}$   
#box4의 너비 =  $300 - 215 = 85\text{px}$



# 반응형 웹을 위한 속성 살펴보기 – flex 속성

- flex-basis와 flex-grow, flex-shrink를 한꺼번에 지정

## flex : m

flex-grow: m, flex-shrink: 1, flex-basis: 0,

(값 0 : 확장 /축소하는 상황에 따라 기본 크기가 달라짐)

## flex : m, n

첫번째 숫자 m은 flex-grow 값

두번째 숫자 n에 단위가 있다면 flex-basis 값, 단위가 없다면 flex-shrink 값

```
flex: 1;          /* flex-grow: 1, flex-shrink: 1, flex-basis: 0 */
flex: 3;          /* flex-grow: 3, flex-shrink: 1, flex-basis: 0 */
flex: 0 1 auto;   /* flex-grow: 0, flex-shrink: 1, flex-basis: auto */
flex: 1 300px;    /* flex-grow:1, flex-shrink: 1, flex-basis: 300px */
```

# 반응형 웹을 위한 속성 살펴보기 – flex 속성

- flex-basis와 flex-grow, flex-shrink를 한꺼번에 지정

## **flex : none**

- flex: 0 0 auto로 지정한 것과 같음
- 플렉스 컨테이너 안에서 플렉스 항목이 확장하거나 축소되지 않음
- 미리 정해 놓은 width, height 값을 사용하거나 내용에 따라 크기 결정됨

## **flex : auto**

- flex: 1 1 auto로 지정한 것과 같음
- 컨테이너에 남는 공간이 있다면 똑같은 비율로 늘리고, 공간이 부족하면 똑같은 비율로 줄임