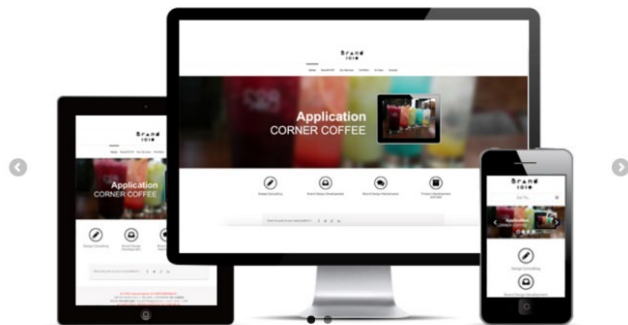


🚀 김쌤과 함께 쉽고 빠르게

부트스트랩4를 활용한 반응형 웹 만들기



STEP 04

플렉스 속성을 활용한 클래스들

INDEX

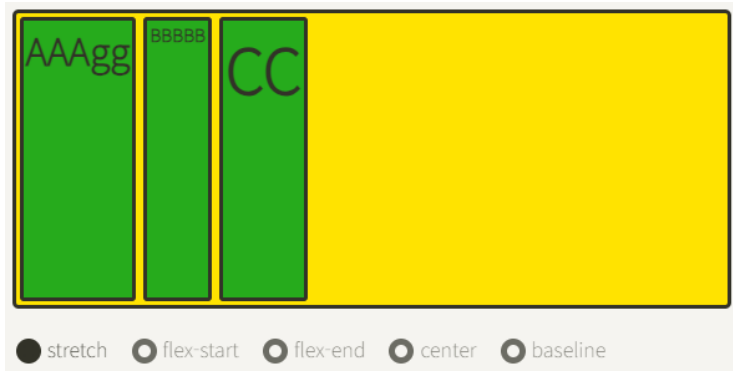
정렬 클래스

순서 클래스

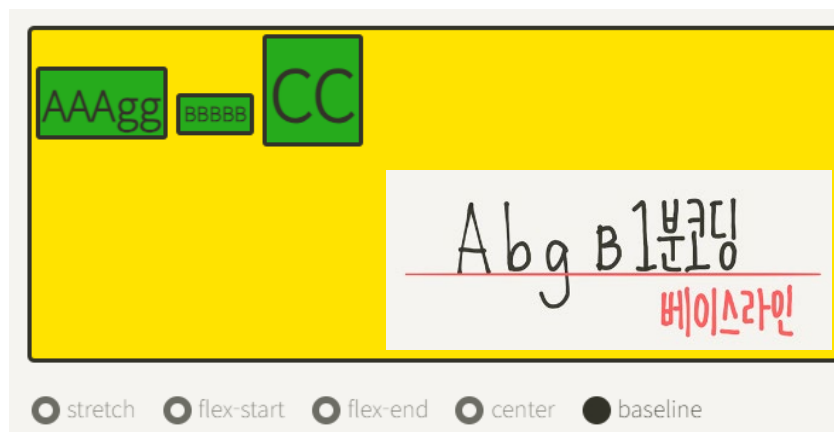
이동 클래스

.align-items-stretch / .align-items-baseline

아이템들이 수직축 방향으로 끝까지 쭉욱 늘어난다.
(기본값)



아이템들을 텍스트 베이스라인 기준으로 정렬한다.

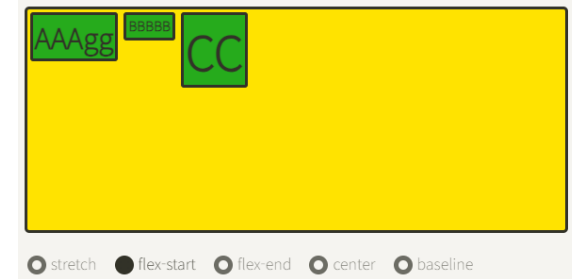


부트스트랩4 css에 정의된 클래스명으로 부모요소에 정렬관련 클래스를 주지 않으면
align-items 기본 default값인 align-items: stretch 스타일이 동작하며
.align-items-stretch / .align-items-baseline 클래스명 등을 가지고 있다.

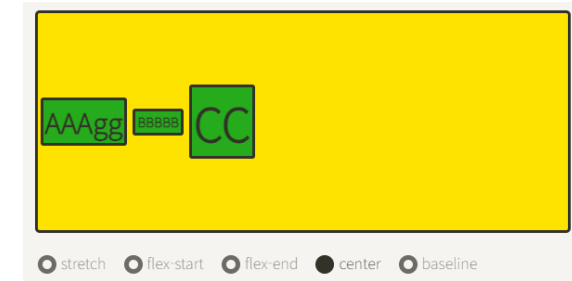
.align-items-start / .align-items-center / .align-items-end

One of three columns	One of three columns	One of three columns
One of three columns	One of three columns	One of three columns
One of three columns	One of three columns	One of three columns

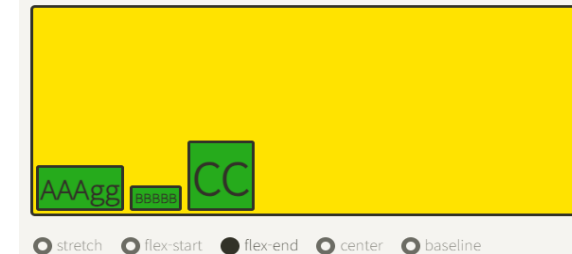
```
<div class="row align-items-start">  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
</div>
```



```
<div class="row align-items-center">  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
</div>
```



```
<div class="row align-items-end">  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
  <div class="col">  
    One of three columns  
  </div>  
</div>
```



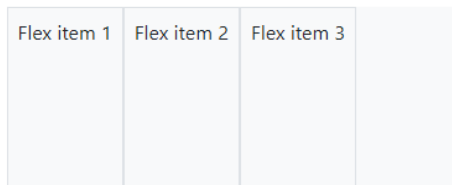
부모요소에 해당 클래스명을 주면 자식 콘텐츠들이 수직 기준으로 정렬하는
align-items : flex-start / align-items : flex-center / align-items : flex-end 스타일이 동작한다.

<https://getbootstrap.com/docs/4.6/layout/grid/#alignment>

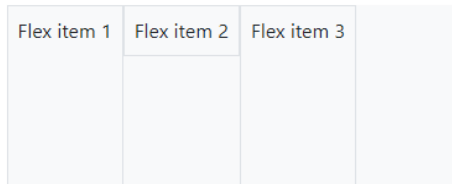
.align-self-stretch / .align-self-baseline

```
<p>none:</p>
<div class="d-flex bg-light" style="height:150px">
  <div class="p-2 border">Flex item 1</div>
  <div class="p-2 border">Flex item 2</div>
  <div class="p-2 border">Flex item 3</div>
</div>
<br>
<p>.align-self-start:</p>
<div class="d-flex bg-light" style="height:150px">
  <div class="p-2 border">Flex item 1</div>
  <div class="p-2 border align-self-start">Flex item 2</div>
  <div class="p-2 border">Flex item 3</div>
</div>
```

none:

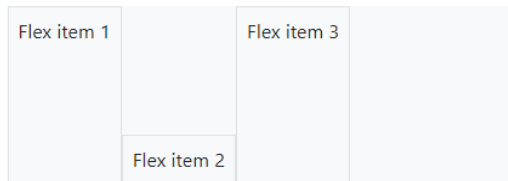


.align-self-start:

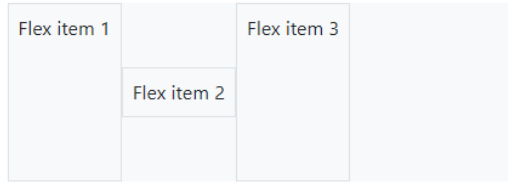


```
<p>.align-self-end:</p>
<div class="d-flex bg-light" style="height:150px">
  <div class="p-2 border">Flex item 1</div>
  <div class="p-2 border align-self-end">Flex item 2</div>
  <div class="p-2 border">Flex item 3</div>
</div>
<br>
<p>.align-self-center:</p>
<div class="d-flex bg-light" style="height:150px">
  <div class="p-2 border">Flex item 1</div>
  <div class="p-2 border align-self-center">Flex item 2</div>
  <div class="p-2 border">Flex item 3</div>
</div>
```

.align-self-end:



.align-self-center:



```
.align-self-start {
  -ms-flex-item-align: start !important;
  align-self: flex-start !important;
}
```

```
.align-self-end {
  -ms-flex-item-align: end !important;
  align-self: flex-end !important;
}
```

```
.align-self-center {
  -ms-flex-item-align: center !important;
  align-self: center !important;
}
```

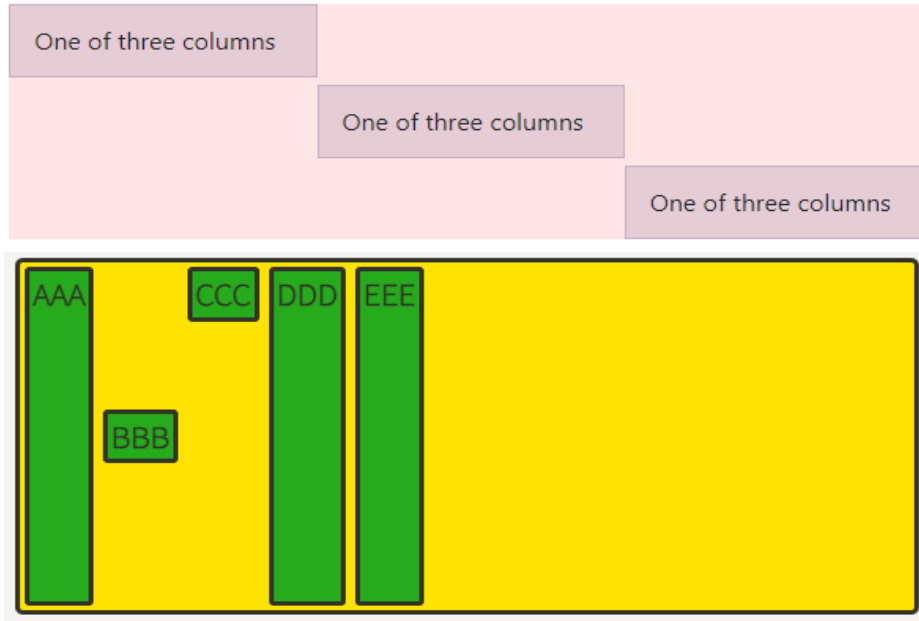
```
.align-self-baseline {
  -ms-flex-item-align: baseline !important;
  align-self: baseline !important;
}
```

```
.align-self-stretch {
  -ms-flex-item-align: stretch !important;
  align-self: stretch !important;
}
```

자신요소에 정렬관련 클래스를 주지 않으면
align-self 기본 default값인 align-self: stretch 스타일이 동작하며
.align-self-stretch / .align-self-baseline 클래스명 등을 가지고 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#alignment>

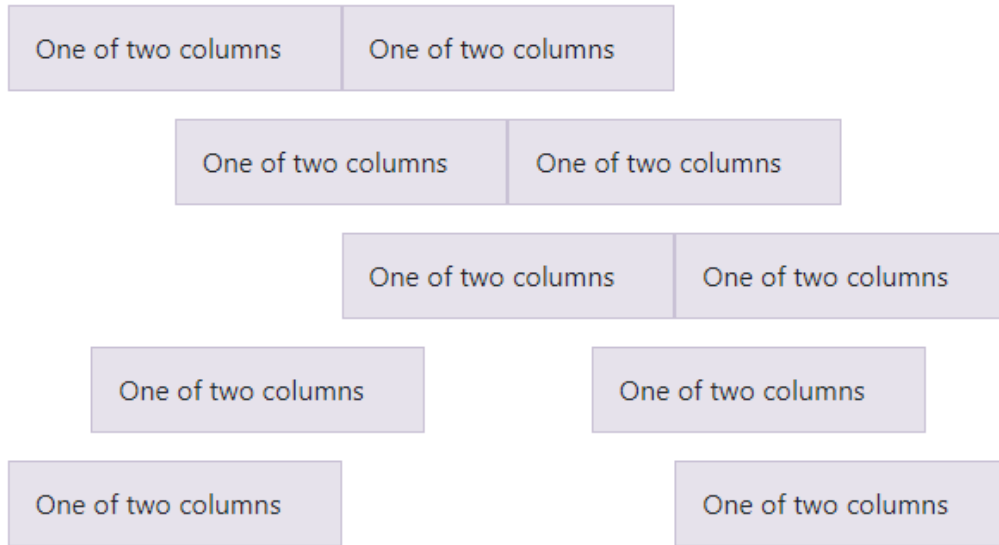
.align-self-start / .align-self-center / .align-self-end



```
<div class="container">
  <div class="row">
    <div class="col align-self-start">
      One of three columns
    </div>
    <div class="col align-self-center">
      One of three columns
    </div>
    <div class="col align-self-end">
      One of three columns
    </div>
  </div>
</div>
```

자식인 콘텐츠 자신에 클래스명을 주어 수직 기준으로 자기자신을 별개로 정렬할 수 있다.
align-self: flex-start / align-self: flex-center / align-self: flex-end 스타일이 동작한다.

.justify-content-start / .justify-content-center / .justify-content-end /



```
<div class="row justify-content-start">
  <div class="col-4">
    One of two columns
  </div>
  <div class="col-4">
    One of two columns
  </div>
</div>
<div class="row justify-content-center">
  <div class="col-4">
    One of two columns
  </div>
  <div class="col-4">
    One of two columns
  </div>
</div>
<div class="row justify-content-end">
  <div class="col-4">
    One of two columns
  </div>
  <div class="col-4">
    One of two columns
  </div>
</div>
```

아이템들을 시작점으로 정렬한다. (기본값)



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

아이템들을 가운데로 정렬한다.



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

아이템들을 끝점으로 정렬한다.

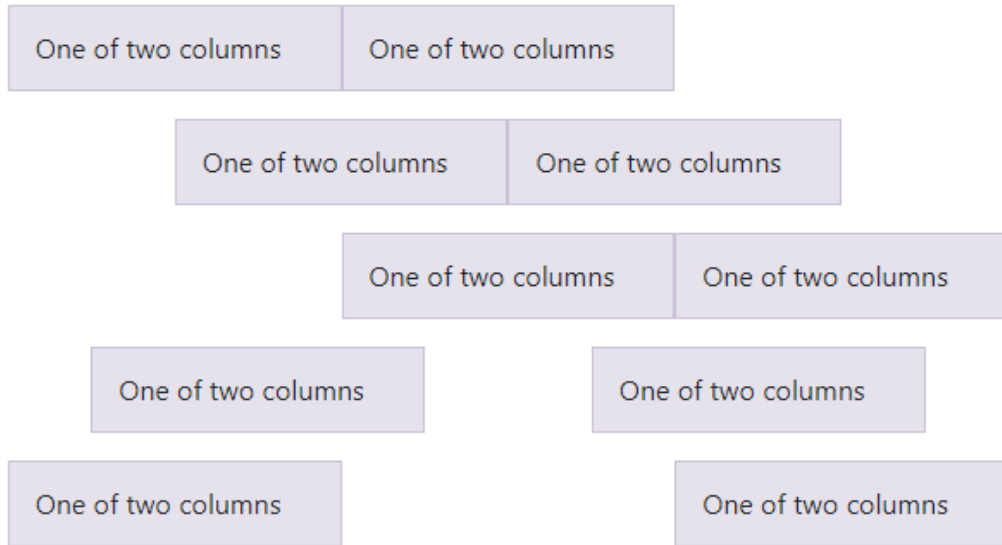


☐ flex-start ☒ flex-end ☐ center ☐ space-between ☐ space-around

부모요소에 수평 정렬관련 클래스를 주어 수평 기준으로 자식 콘텐츠 요소를 정렬할 수 있다.

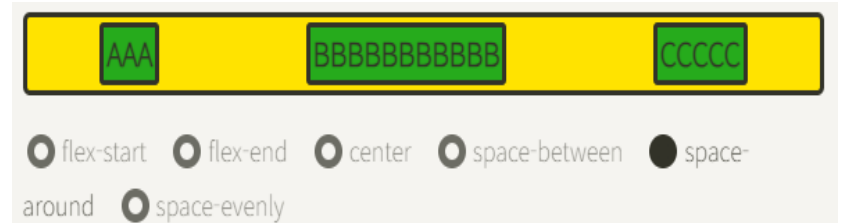
<https://getbootstrap.com/docs/4.6/layout/grid/#horizontal-alignment>

.justify-content-around / .justify-content-between



```
<div class="row justify-content-around">
  <div class="col-4">
    One of two columns
  </div>
  <div class="col-4">
    One of two columns
  </div>
</div>
<div class="row justify-content-between">
  <div class="col-4">
    One of two columns
  </div>
  <div class="col-4">
    One of two columns
  </div>
</div>
```

아이템들의 “둘레(around)”에 균일한 간격을 만들어 준다.



아이템들의 “사이(between)”에 균일한 간격을 만들어 준다.



부모요소에 수평 정렬관련 클래스를 주어 수평 기준으로 자식 콘텐츠 요소를 정렬할 수 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#horizontal-alignment>

.no-gutters

.col-sm-6 .col-md-8

.col-6 .col-md-4

```
<div class="row no-gutters">
  <div class="col-sm-6 col-md-8">.col-sm-6 .col-md-8</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>
```

[Copy](#)

```
.no-gutters {
  margin-right: 0;
  margin-left: 0;
}
```

```
.no-gutters > .col,
.no-gutters > [class*="col-"] {
  padding-right: 0;
  padding-left: 0;
}
```

.no-gutters를 .row에 주면 기본적으로 row가 가지고 있던 margin의 음수 값과 자식요소인 col이 가지고 있는 padding 양수 값을 모두 0으로 만든다.

<https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.css>

.order-first / .order-0~.order-12 / .order-last

순서를 바꾸는
.order-*

ordering


.col-*-push-*


.col-*-pull-*

```
<div class="row">  
  <div class="col-md-9 col-md-push-3">.col-md-9 .col-md-push-3</div>  
  <div class="col-md-3 col-md-pull-9">.col-md-3 .col-md-pull-9</div>  
</div>
```

.col-md-3
.col-md-pull-9

.col-md-9
.col-md-push-3

.col-md-9
.col-md-push-3

.col-md-3
.col-md-pull-9

Bs3에 있던 push, pull 키워드 클래스는 없어지고 order 키워드를 갖는 클래스가 생겼다.

<https://getbootstrap.com/docs/4.6/layout/grid/#order-classes>

.order-first / .order-0~.order-12 / .order-last

```
<p>Change the visual order of a specific flex item  
re from 0 to 12:</p>  
<div class="d-flex mb-3">  
  <div class="p-2 bg-info">Flex item 1</div>  
  <div class="p-2 bg-warning">Flex item 2</div>  
  <div class="p-2 bg-primary">Flex item 3</div>  
</div>
```

Order

Change the visual order of a specific flex item(s) with
12:

Flex item 1 Flex item 2 Flex item 3

```
<p>Change the visual order of a specific flex item(s) with the  
are from 0 to 12:</p>  
<div class="d-flex mb-3">  
  <div class="p-2 order-3 bg-info">Flex item 1</div>  
  <div class="p-2 order-2 bg-warning">Flex item 2</div>  
  <div class="p-2 order-1 bg-primary">Flex item 3</div>  
</div>
```

Order

Change the visual order of a specific flex item(s) with the .order cl
12:

Flex item 3 Flex item 2 Flex item 1

각 아이템들의 시각적 나열 순서를 결정하는 속성으로 숫자값이 들어가며, 작은 숫자일 수록 먼저 배치된다. "시각적" 순서
일 뿐, HTML 자체의 구조를 바꾸는 것은 아니므로 접근성 측면에서 사용에 주의하여야 하는데 시각 장애인분들이 사용하
는 스크린 리더로 화면을 읽을 때, order를 이용해 순서를 바꾼 것은 의미가 없다는 것을 주의하자!!

<https://getbootstrap.com/docs/4.6/layout/grid/#order-classes>

.order-first / .order-0~.order-12 / .order-last

1.col : order-미디어 사이즈-* 클 래스를 가 지고 있 어 원하 는 사 이즈 이 상에서 컨텐츠의 순서를 바꿀 수 있다.	2.order- xl-first	3.order- xl-last	4.order- xl-12	5.order- xl-1
--	----------------------	---------------------	-------------------	------------------

3.order-xl- last	1.col : order-미디어 사이즈-* 클 래스를 가 지고 있 어 원하 는 사 이즈 이 상에서 컨텐츠의 순서를 바꿀 수 있다.	2.order-xl- first	4.order-xl- 12	5.order-xl- 1
---------------------	--	----------------------	-------------------	------------------

2.order-xl-first	1.col : order-미디어 사이즈-* 클래스를 가지고 있어 원하는 사이즈 이상 에서 컨텐츠의 순서를 바 꿀 수 있다.	5.order-xl-1	4.order-xl-12	3.order-xl-last
------------------	--	--------------	---------------	-----------------

1 스마트폰일 때에는 상단에 있다가 타블렛 이상에서 우측 1/4 메뉴가 되게 하려면 col-12 col-md-3 order-md-last 를 준다.		
2	3	4

2	3	4	1 스마트폰일 때에는 상단에 있다가 타블 렛 이상에서 우측 1/4 메뉴가 되게 하 려면 col-12 col-md- 3 order-md-last 를 준다.
---	---	---	---

order-미디어사이즈-* 클래스를 이용하면 디바이스 사이즈별 순서가 바뀌는 컨텐츠를 구현할 수 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#order-classes>

요소를 이동
시키는
offset

offset

offset-1

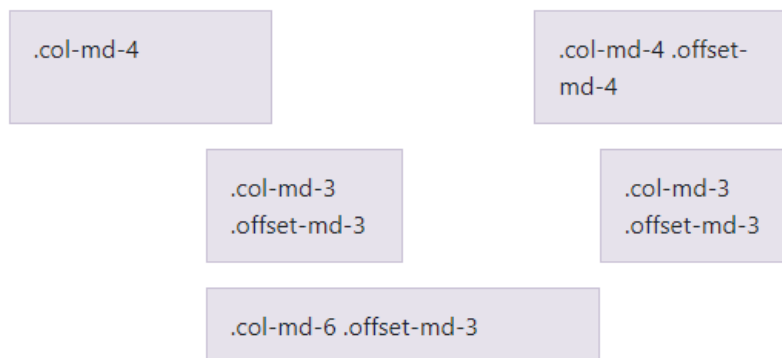
offset-11

offset-sm-0

offset-sm-11

offset-xl-11

offset-미디어사이즈-*



```
<div class="container">
  <div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
  </div>
  <div class="row">
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  </div>
  <div class="row">
    <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
  </div>
</div>
```

클래스 별로 margin-left를 1/12 %씩 추가하여 좌우 간격을 조절할 수 있도록 스타일링 되어 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#offsetting-columns>

offset-미디어사이즈-*

.col-md-4

.col-md-4 .offset-md-4

.col-md-3
.offset-md-3

.col-md-3
.offset-md-3

.col-md-6 .offset-md-3

```
<div class="container">
  <div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
  </div>
  <div class="row">
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  </div>
  <div class="row">
    <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
  </div>
</div>
```

offset 관련 클래스를 이용했을 때에는 옆의 있는 컨텐츠에 영향을 주어서 간격을 만들면서 같이 밀리며 ordering 관련 클래스는 옆에 있는 컨텐츠에 영향을 주지 않고 독단적으로 밀고 당겨오는 차이점이 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#offsetting-columns>

그리드의 중첩 구조 만들기





Level 1: .col-sm-9	
Level 2: .col-8 .col-sm-6	Level 2: .col-4 .col-sm-6

```
<div class="container">
  <div class="row">
    <div class="col-sm-9">
      Level 1: .col-sm-9
      <div class="row">
        <div class="col-8 col-sm-6">
          Level 2: .col-8 .col-sm-6
        </div>
        <div class="col-4 col-sm-6">
          Level 2: .col-4 .col-sm-6
        </div>
      </div>
    </div>
  </div>
</div>
```

기존의 .col 내에 row와 col을 추가하여 콘텐츠 안에 콘텐츠를 그리드 구조를 활용하여 추가할 수 있다.

<https://getbootstrap.com/docs/4.6/layout/grid/#offsetting-columns>

꼭 기억해야 할 포인트 정리

1. 부모 요소에 정렬 관련 클래스를 주지 않으면 align-items 기본 default값인 align-items :
 스타일이 동작한다.
2. .row 부모요소에  클래스명을 주면 자식 컨텐츠들이 수직 기준으로 센터로 정렬된다.
3. 타블렛 이상의 사이즈에서 원하는 요소를 그리드 영역 안에서 두 칸 이동시키려면  를 사용한다.
4. 그리드 영역 안에서 순서를 바꿀 때 사용하는 클래스명으로 노트북 이상의 사이즈에서 제일 마지막으로 보내려면  를 사용한다.

꼭 기억해야 할 포인트 정리

1. 부모 요소에 정렬 관련 클래스를 주지 않으면 align-items 기본 default값인 align-items : **stretch** 스타일이 동작한다.
2. .row 부모요소에 **flex-direction: column** 클래스명을 주면 자식 컨텐츠들이 수직 기준으로 센터로 정렬된다.
3. 타블렛 이상의 사이즈에서 원하는 요소를 그리드 영역 안에서 두 칸 이동시키려면 **grid-column: span 2**를 사용한다.
4. 그리드 영역 안에서 순서를 바꿀 때 사용하는 클래스명으로 노트북 이상의 사이즈에서 제일 마지막으로 보내려면 **grid-column: 1 / 12**를 사용한다.

꼭 기억해야 할 포인트 정리

1. 부모 요소에 정렬 관련 클래스를 주지 않으면 align-items 기본 default값인 align-items : **stretch** 스타일이 동작한다.
2. .row 부모요소에 **.align-items-center** 클래스명을 주면 자식 컨텐츠들이 수직 기준으로 센터로 정렬된다.
3. 타블렛 이상의 사이즈에서 원하는 요소를 그리드 영역 안에서 두 칸 이동시키려면 **grid-gap**를 사용한다.
4. 그리드 영역 안에서 순서를 바꿀 때 사용하는 클래스명으로 노트북 이상의 사이즈에서 제일 마지막으로 보내려면 **grid-column-end: 12**를 사용한다.

꼭 기억해야 할 포인트 정리

1. 부모 요소에 정렬 관련 클래스를 주지 않으면 align-items 기본 default값인 align-items : **stretch** 스타일이 동작한다.
2. .row 부모요소에 **.align-items-center** 클래스명을 주면 자식 컨텐츠들이 수직 기준으로 센터로 정렬된다.
3. 타블렛 이상의 사이즈에서 원하는 요소를 그리드 영역 안에서 두 칸 이동시키려면 **.offset-md-2** 를 사용한다.
4. 그리드 영역 안에서 순서를 바꿀 때 사용하는 클래스명으로 노트북 이상의 사이즈에서 제일 마지막으로 보내려면 **.order-md-last** 를 사용한다.

꼭 기억해야 할 포인트 정리

1. 부모 요소에 정렬 관련 클래스를 주지 않으면 align-items 기본 default값인 align-items : **stretch** 스타일이 동작한다.
2. .row 부모요소에 **.align-items-center** 클래스명을 주면 자식 컨텐츠들이 수직 기준으로 센터로 정렬된다.
3. 타블렛 이상의 사이즈에서 원하는 요소를 그리드 영역 안에서 두 칸 이동시키려면 **.offset-md-2** 를 사용한다.
4. 그리드 영역 안에서 순서를 바꿀 때 사용하는 클래스명으로 노트북 이상의 사이즈에서 제일 마지막으로 보내려면 **.order-lg-last** 를 사용한다.

영상 코딩 디자인은 영코디 김쌤



영코디 김쌤

