web 시험 정리

HTML

1. HTML(Hyper Text Markup Language)

💡웹페이지를 작성하기 위한(구조를 잡기 위한) 언어. 웹 컨텐츠의 의미와 구조를 정의

(HTTP: Hyper Text Transfer Protocol)

Hyper Text: 기존의 선형적인 텍스트가 아닌 비 선형적으로 이루어진 텍스트. 인터넷의 등장과 함께 대두. 기본적으로 HyperLink를 통해 텍스트를 이동한다.

Markup language: 태그를 이용해 문서나 데이터의 구조를 명시. 프로그래밍 언어와 달리 단순히 데이터를 표현.

- html 요소: HTML문서의 최상위 요소, 문서의 root를 뜻한다. head와 body로 구분
- 💡head 요소: 문서제목, 문자 코드(meta charset="UTF-8")와 같이 해당 문서 정보를 담고 있으며 브라우저에 나타나지 않는다. CSS선언 혹은 외부 로딩 파일 지정 등 작성. HEAD 테그에 메타 정보가 있어 정보 전달가능(OGP)
- body 요소: 브라우저 화면에 나타나는 정보. 실제 내용에 해당.

DOM 트리(Document Object Model)

: 문서들을 객체 지향적으로 표현한 것

```
<body>
    <h1> 웹문서 </h1>
    <ul>
        <li>Html</li>
        <li>CSS</li>  <-이 한줄이 요소
    </ul>
</body>
```

- 요소(element): 태그와 내용으로 구성되어 있다.
- 속성(attribute): a href="https//google.com"></a 이 때 href가 속성. 태그별로 사용할 수 있는 속성은 다르다.

모든 Html 요소가 공통으로 사용할 수 있는 속성 (동작x경우도 있음)

id, class, hidden, lang, style, tabindex, title

2. 시맨틱 태그

: HTML5에서 등장 의미론적 요소를 담은 태그의 등장. 검색엔진 등에 의미 있는 정보 그룹을 태그로 표현

div, span은 Non semantic 요소, h1, talbe은 semantic

검색엔진최적화(SEO)를 위해 메타태그, 시맨틱 태그등을 통한 마크업 필요.

- header: 머릿말
- nav: 내비게이션
- aside: 사이드에 위치, 메인 컨텐츠와 관련성 적음
- section: 문서의 일반적 부분, 컨텐츠의 그룹
- article: 독립적으로 구분되는 영역
- footer: 푸터, 마지막부분

3. 그룹 컨텐츠

- p: 하나의 문단 만들때
- hr: 수평선 그리기
- ol, ul > li: 목록 만듬(순서 있는 목록, 순서 필요없는 목록)
- pre, : 미리 정의된 형식의 텍스트 정의할 때 사용
- blockquote: 긴 인용문 나타냄
- div: display 속성이 block인 특별한 기느을 갖지 않는 태그

4. 텍스트 관련 요소

- a: 하이퍼링크 걸어줌
- b vs strong : 글씨 굵게
- i vs em: 이탤릭체
- span: display 속성이 inline인 특별한 기능 갖지 않는 태그
- br: 줄바꿈
- img: 이미지 삽입(src, alt)

5. table

- tr, td, th/ thead, tbody, tfoot/ caption/ scope/ col, colgroup

6. input

: 다양한 타입 가지는 입력 데이터 필드

- 공통 속성: name, placeholder, required, autofocus
- type: button, checkbox, color, date, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, url, week

CSS

1. CSS (Cascading Sytyle Sheets)

: 스타일, 레이아웃 등을 통해 문서(HTML)를 표시하는 방법을 지정하는 언어

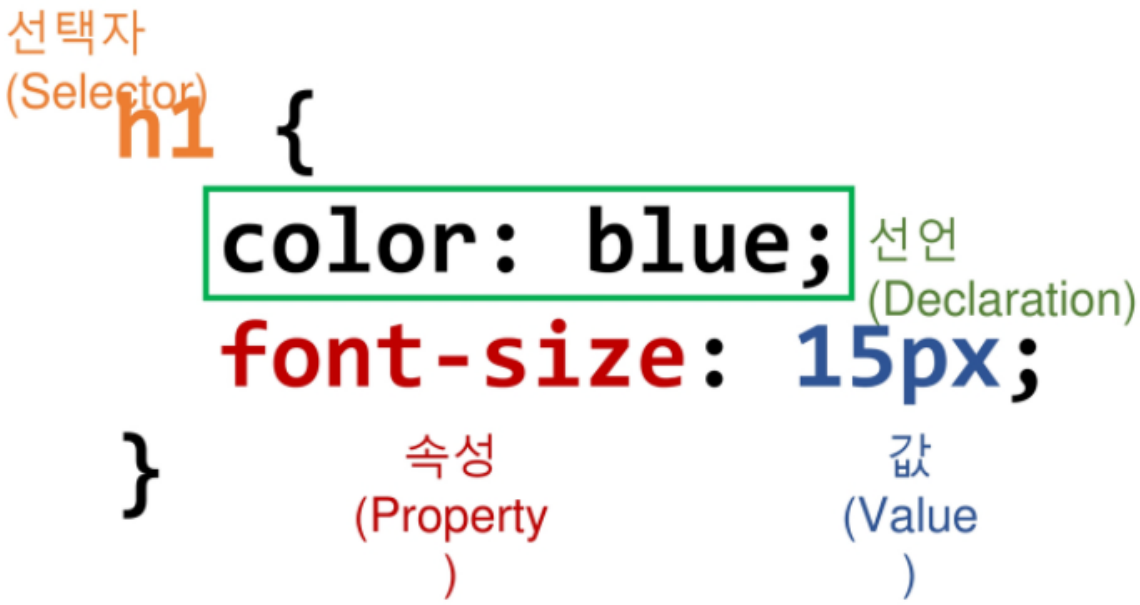
- 정의 방법: 인라인, 내부참조. 외부참조

2. 선택자(Selector)

: HTML문서에서 특정한 요소를 선택하여 스타일링 하기 위해 반드시 선택자라는 개념이 필요

html 페이지 안의 특정 element들을 선택해 선언 블록의 내용을 적용시킴

- 기본선택자: 전체(*), 태그- 요소(p), 속성(div), 클래스(.ssafy), 아이디(#) ,
- 결합자(Combinarors): 자손 결합자, 자식 결합자, 일반 형제, 인접 형제



3. 💡CSS적용 우선 순위

중요도: !important

우선순위: 인라인 > id 선택자> class 선택자 > 요소 선택자

4. CSS 상속

:부모 요소의 속성을 자식에게 상속한다.

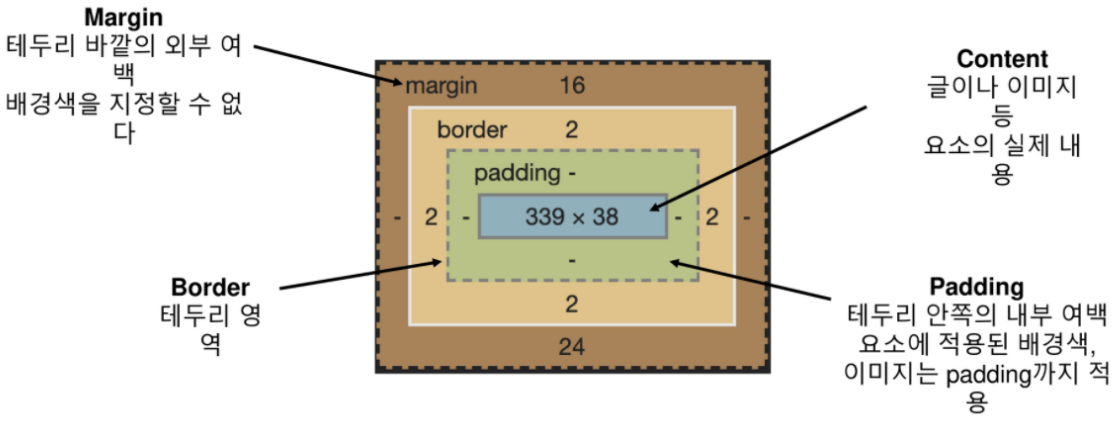
- 상속되는것: text관련 요소 (font, color, text-align), opacity, visibility
- 상속되지 않는 것: box model관련 요소(width, height, margin, padding, border, box-sizing, display), position관련 요소(position, top, right, bottom, left, z-index)

5. CSS 단위

- px: 화소
- %
- em: 배수 단위, 요소에 지정된 사이즈에 상대적인 사이즈
- rem: 최상위 요소(html 16px)의 사이즈를 기준으로 배수단위 가짐.
- viewport
- RGB

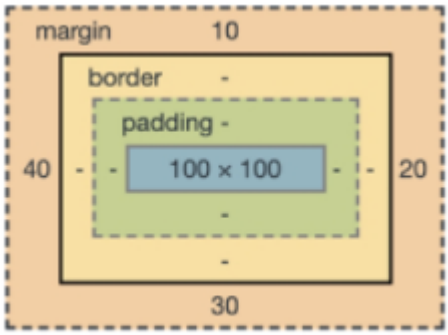
모든 요소는 네모(Box model)이고, 어떻게 보여지는지(display)에 따라 문서에서 배치가 달라질 수 있다.

6. Box model



margin

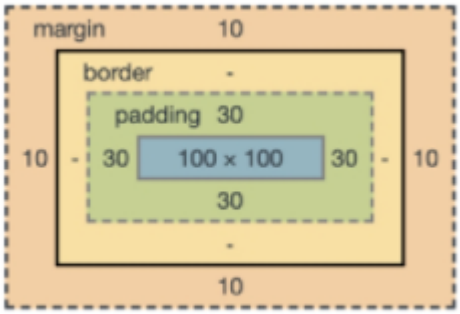
:테두리 바깥의 외부 여백, 배경색을 지정할 수 없다




```
<style>
    .margin{
      margin-top: 10px;
      margin-right: 20px;
      margin-bottom: 30px;
      margin-left: 40px;
    }
</style>
```

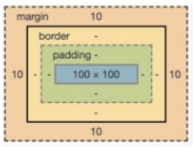
padding

: 테두리 안 쪽의 내부 여백 요소에 적용된 배경색, 이미지는 padding까지 적용

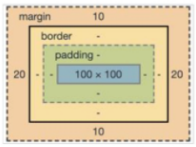



```
<style>
    .margin-padding {
        margin: 10px;
        padding: 30px;
    }
</style>
```

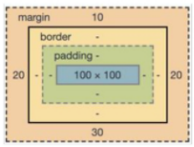
- shorthand



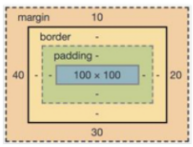
.margin-1 {
margin: 10px;
}



.margin-2 {
margin: 10px 20px;
}

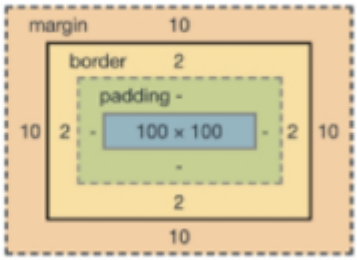


.margin-3 {
margin: 10px 20px 30px;
}



.margin-4 {
margin: 10px 20px 30px 40px;
}

border




```
.border {
  border-width: 2px;
  border-style: dashed;
  border-color: black;
}
```

content-box: 기본, padding을 제외한 순수 contents 영역만 박스로 지정

border-box: 일반적으로 보이는 border까지 너비를 100px로 보고 싶을 경우

7. 💡Display

- display: block

- 줄바꿈이 일어남
- 화면 크기 전체의 가로폭 차지
- 블록 레벨 요소 안에 인라인 레벨 요소가 들어갈 수 있다.
- 기본은 너비의 100%, 너비를 못 가지면 컨텐츠영역 만큼
- ex) div / ul, ol, li / p / hr / form

- display: inline

- 줄바꿈이 일어나지 않는 행의 일부 요소
- content 너비만큼 가로 폭을 차지한다
- width, height, margin-top, margin-bottom을 지정할 수 없다.
- 상하 여백은 line-height로 지정
- ex) span / a / img / input, label / b, em, i, strong

- display: inline-block

- block과 inline 레벨 요소의 특징을 모두 갖는다.
- inline처럼 한 줄에 표시 가능
- block처럼 width, height, margin 속성을 모두 지정할 수 있다.

- display: none

- 해당 요소를 화면에 표시하지 않는다. (공간도 사라짐)
- 이와 비슷한 visibility: hidden 은 해당 요소가 공간은 차지하나 화면에 표시만 하지 않는다.

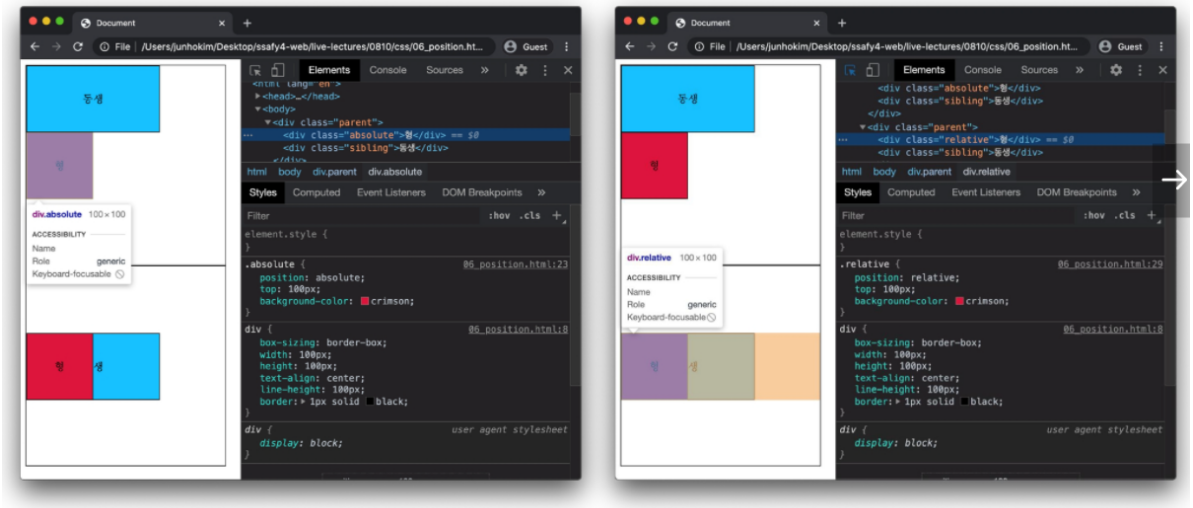
8. 💡CSS position

- static: default, 기본적인 요소 순서에 따름

아래는 좌표 프로퍼티(top,bottom, left, right) 사용해 이동 가능

- relative: static 위치 기준으로 이동 (상대 위치)
- absolute: static이 아닌 가장 가까이 있는 부모 조상 요소 기준으로 이동
- fixed: 부모 요소와 관계없이 브라우저를 기준으로 이동 (고정 위치), 스크롤 시에도 항상 같은 곳 위치

> absolute vs relative



absolute하면 본래의 위치를 빼주지만 relative하면 원래 위치도 차지. Absolute는 relative와 달리 static이 아닌 조상의 position기준으로 하기 때문.

9. Float

: float된 이미지 좌, 우측 주변으로 텍스트를 둘러싸는 레이아웃을 위해 도입

{float: left} 이렇게 쓰임

10. 💡Flex box (Flexible Box)

: 요소 간 공간 배분과 정렬 기능을 위한 1차원(단방향) 레이아웃

- 요소

  Flex Container (부모 요소)

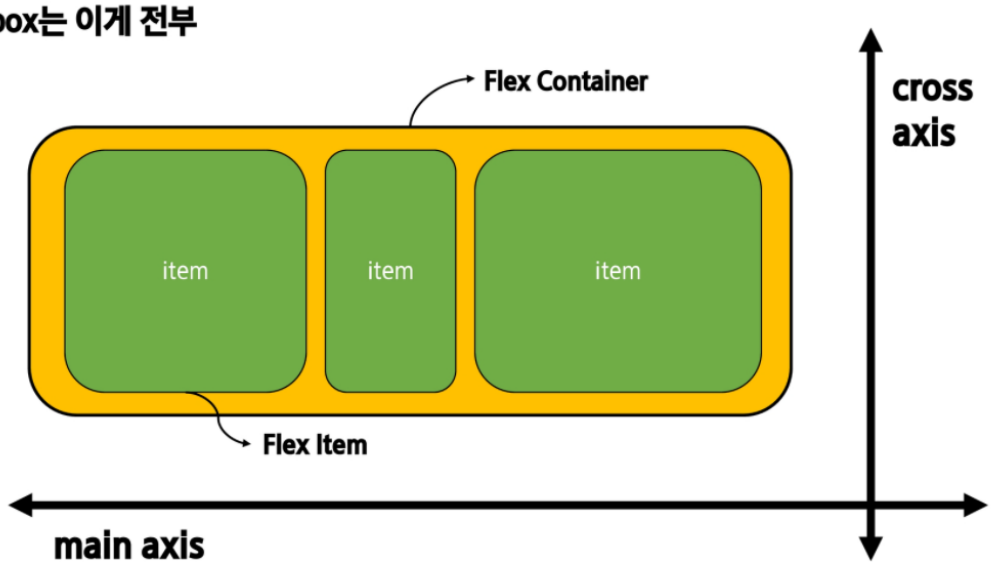
  Flex Item (자식 요소)

- 축

  main axis (메인 축)

  cross axix (교차 축)

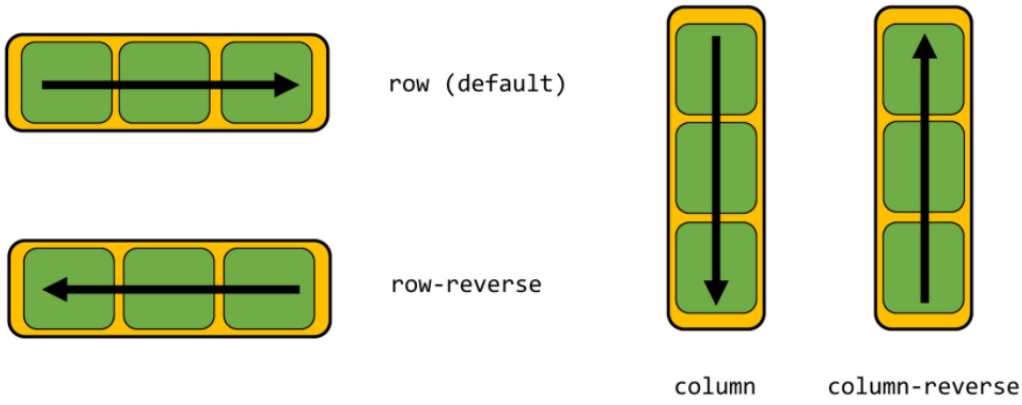
Flexbox는 이게 전부



부모 요소에 display: flex or inline-flex를 작성하는 것부터 시작.

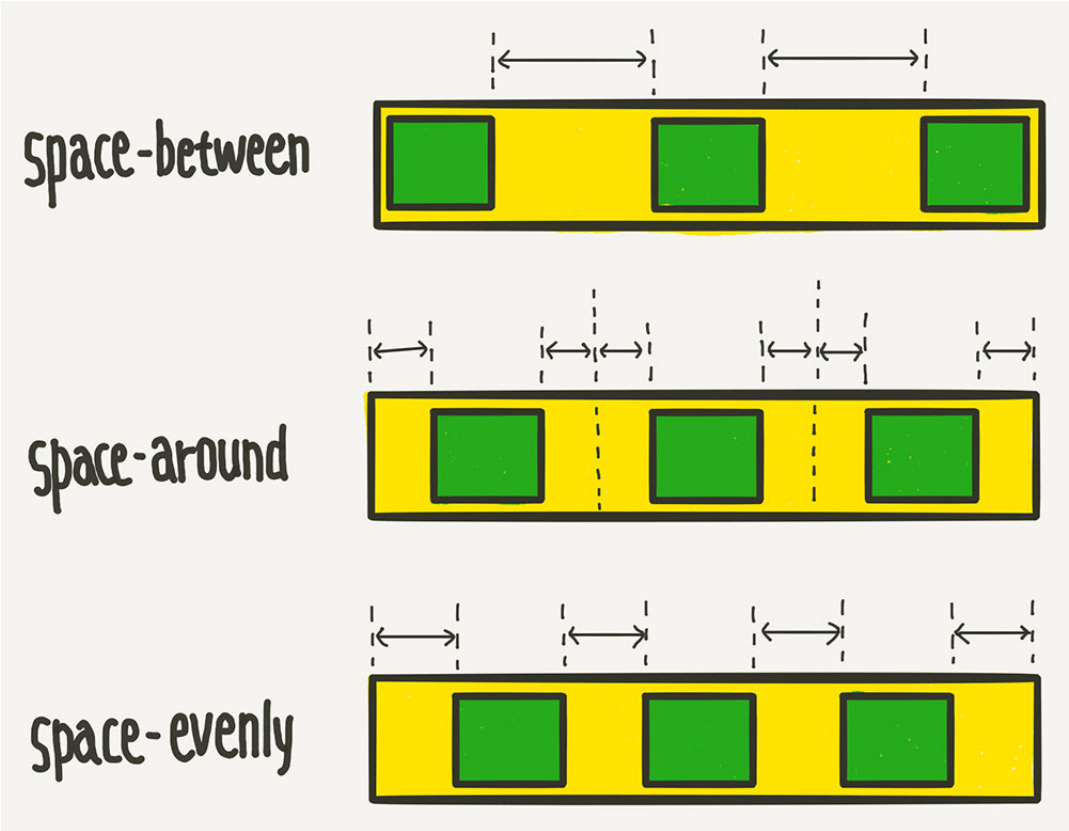
- 속성
  - 배치 방향 설정: flex-direction

    : row, row-reverse, column, column-reverse



  - 메인축 방향 정렬: justify-content

    : flex-start, flex-end, center, space-between, space-around, space-evenly



- 교차축 방향 정렬

  align-items: flex-start, flex-end, center, stretch, baseline

  align-content: flex-start, flex-end, center, stretch, space-between, space-around

  align-self: auto, flex-start, flex-end, center, baseline, stretch

- flex-wrap
- flex-flow: flex-direction, flex-wrap 속성 축약형

Bootstrap

1. CND (Content Delivery(Distribution) Network)

:컨텐츠(CSS, JS, Image, Text 등)을 효율적으로 전달하기 위해 여러 노드에 가진 네트워크에 데이터를 제공하는 시스템

2. 💡spacing

m	margin
p	padding

t	top
b	bottom
s	left
e	right
x	left, right
y	top, bottom

0	0 rem	0px
1	0.25 rem	4px
2	0.5 rem	8px
3	1 rem	16px
4	1.5 rem	24px
5	3 rem	48px

- mt-1
- mt-0
- py-0
- mx-auto: 수평 중앙 정렬

3. Color

```
:root {
  --primary: #007bff;
  --secondary: #6c757d;
  --success: #28a745;
  --info: #17a2b8;
  --warning: #ffc107;
  --danger: #dc3545;
  --light: #f8f9fa;
  --dark: #343a40;
}
```

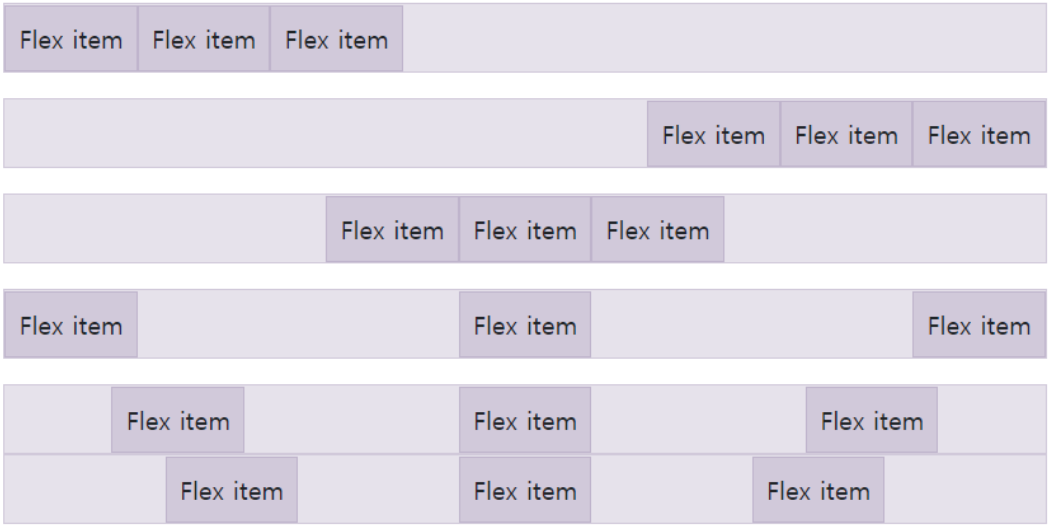
4. 💡d-flex

- ◦ 배치 방향 설정(flex-direction)

  : flex-row, flex-row-reverse, flex-column, flex-column-reverse

  ◦ 메인축 방향 정렬(justify-content)

  : justify-content-start, justify-content-end, justify-content-center, justify-content-between, justify-content-around, justify-content-evenly

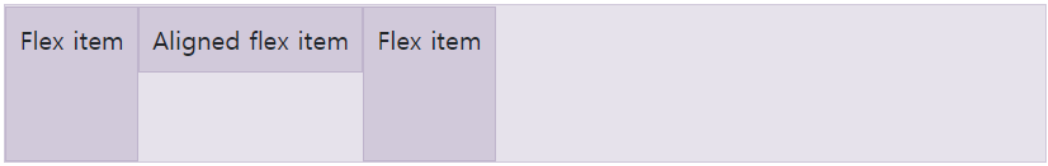


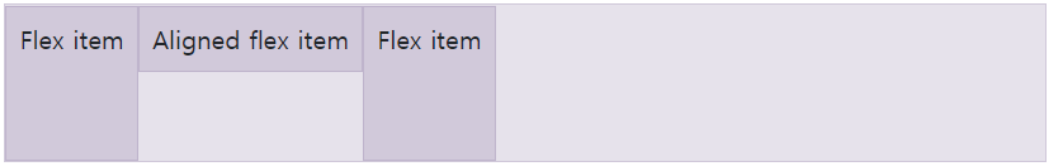
- 교차축 방향 정렬

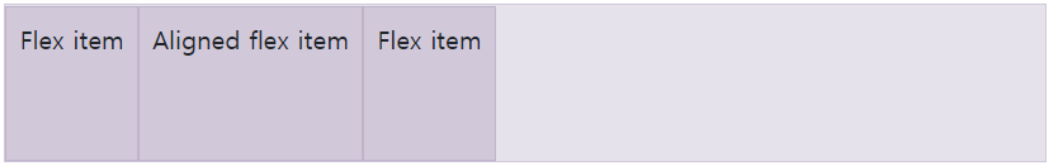
  align-items: align-items-start, align-items-end, align-items-center, align-items-baseline, align-items-stretch

  align-content: align-content-start, align-content-end, align-content-center, align-content-between, align-content-around, align-content-stretch (flex-wrap 뒤에 붙이기)

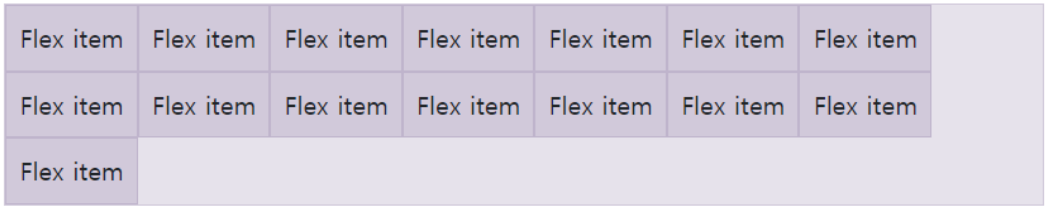
  align-self: align-self-start, align-self-end, align-self-center, align-self-baseline, align-self-stretch

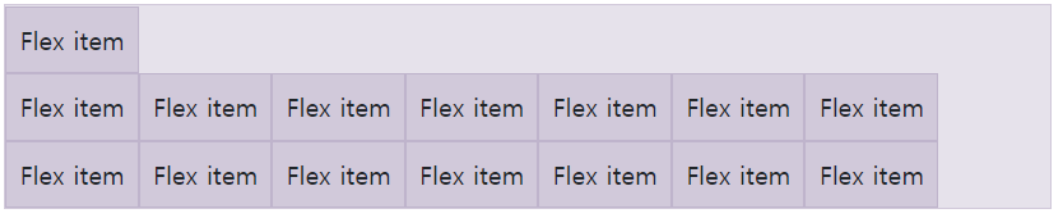






- flex-wrap: flex-wrap, flex-wrap-reverse





5. 💡Responsive

- `container, fows, column` 으로 컨텐츠 배치하고 정렬!

```
<div class="container">
    <div class="row">
        <div class="col-4 col-md-4"></div>
        <div class="col"></div>
        <div class="col"></div>
    </div>
</div>
```

- 🌡️`12개의 column`

- 6개의 grid breakpoints

  xs: <576px

  sm: >= 576px

  md: >= 768px

  lg: >=992px

  xl: >=1200px

  xxl: >= 1400px

Nesting

Level 1: .col-sm-3	Level 2: .col-8 .col-sm-6	Level 2: .col-4 .col-sm-6

```
<div class="container">
  <div class="row">
    <div class="col-sm-3">
      Level 1: .col-sm-3
    </div>
    <div class="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>
```

offset


```
  <div class="row">
      <div class="item col-4 col-md-4 offset-md-4 offset-lg-7 col-lg-5">
        <p>item1</p>
      </div>
      <div class="item col-4 offset-4 col-md-4 offset-md-0 offset-lg-2 col-lg-
8">
        <p>item2</p>
      </div>
    </div>
```