#1.0 Why we need Flexbox

* + 복잡한 계산 없이 박스의 크기와 순서를 배치할 수 있음
  + 정렬, 방향, 순서, 사이즈 등을 유연하게 조절하기 때문에 편리

#1.1 Basics of Flexbox

* + Flex의 기본 개념 두 가지는 Flex-container와 Flex-item
  + 문서의 영역 중, flexbox가 놓여있는 영역을 Flex-Container라고 함
  + Flex-container와 Flex-item는 직접적으로 종속관계에 있어야 함
  + Flex-item에게 규칙을 주는 게 아니라 Flex-container에게 명령을 내림
  + father에게 명령한 규칙은 children items에게 영향을 끼침
    1. space-between : 아이템을 일정한 간격으로 떨어뜨려 정렬
    2. flex-start : 아이템을 모두 앞 쪽에 정렬 (flex를 주면 flex-start가 디폴트)
    3. flex-end : 아이템을 모두 끝 쪽에 정렬
    4. space-around : 아이템 주변에 간격을 만들어 줌

#1.2 Main Axis and Cross Axis

* + align-items은 수평적으로 정렬되어 있는 item을 수직 정렬해 줌
  + 수직 정렬을 위해선 충분한 width(너비)와 height(높이)를 가져야 함
  + 수직, 수평 정렬에 있어 밑바탕이 되는 규칙을 axis(축)이라 함
  + main axis는 justify-content가 동작하는 부분
  + 즉, main axis는 justify-content가 영향을 주는 축이고, cross axis는 align-items가 영향을 주는 축임
  + main axis에 있는 아이템을 움직이려면 justify-content, cross axis에 있는 아이템을 움직이려면 align-items
  + flex 옵션을 주면 기본적으로 flex-direction: row라는 옵션을 가지게 됨 (row를 column으로 변경하면 세로로 정렬)
  + flex-direction이 row면 main axis는 수평, cross axis는 수직이 됨
* flexbox direction이 row면, main(수평) / cross(수직)
* flexbox direction이 column이면, main(수직) / cross(수평)
  + align-items
    1. flex-start : item을 모두 위쪽에 정렬
    2. flex-end : item을 모두 아래쪽에 정렬

#1.3 Flex Wrap and Direction

* + 아이템들끼리 공간이 부족할 때 flex는 기본적으로 찌그러지는 현상이 발생, 이 때 flex-wrap를 사용함
  + flex-wrap: wrap; 을 사용하 면, 아이템들이 겹치는 현상을 방지할 수 있음
  + flex에서