2021.Ver

2021 D.COM WEB 스터디

2. HTML/CSS Layout

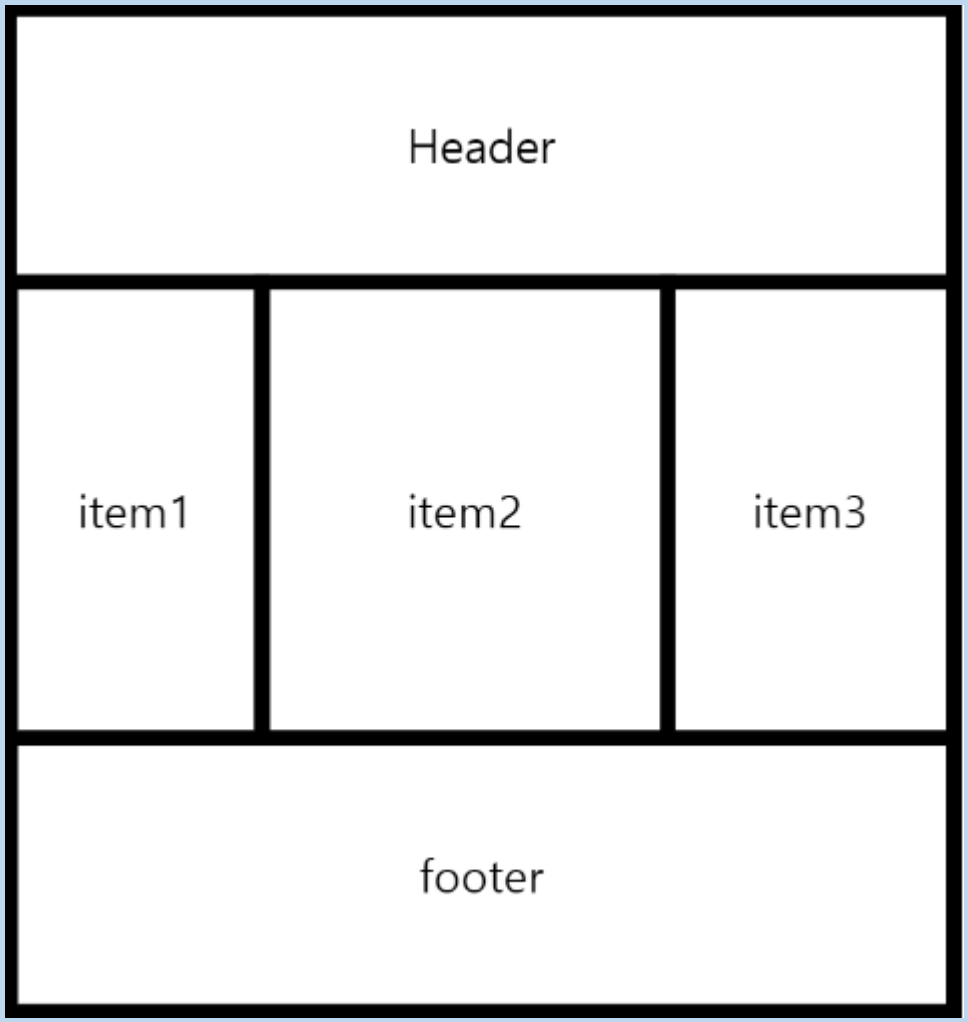
Yongwoo Song
ywsong.dev@kakao.com

Dcom-KHU/2021-Web-Study: Dcom-KHU 2021 Web Study (github.com)

이번 시간에 배울 것은?

HTML/CSS를 이용하여 본격적으로 Layout을 구성해 봅시다!

Box-Modeling, Position, Z-Index, Display, Flexbox를 알아봅니다.



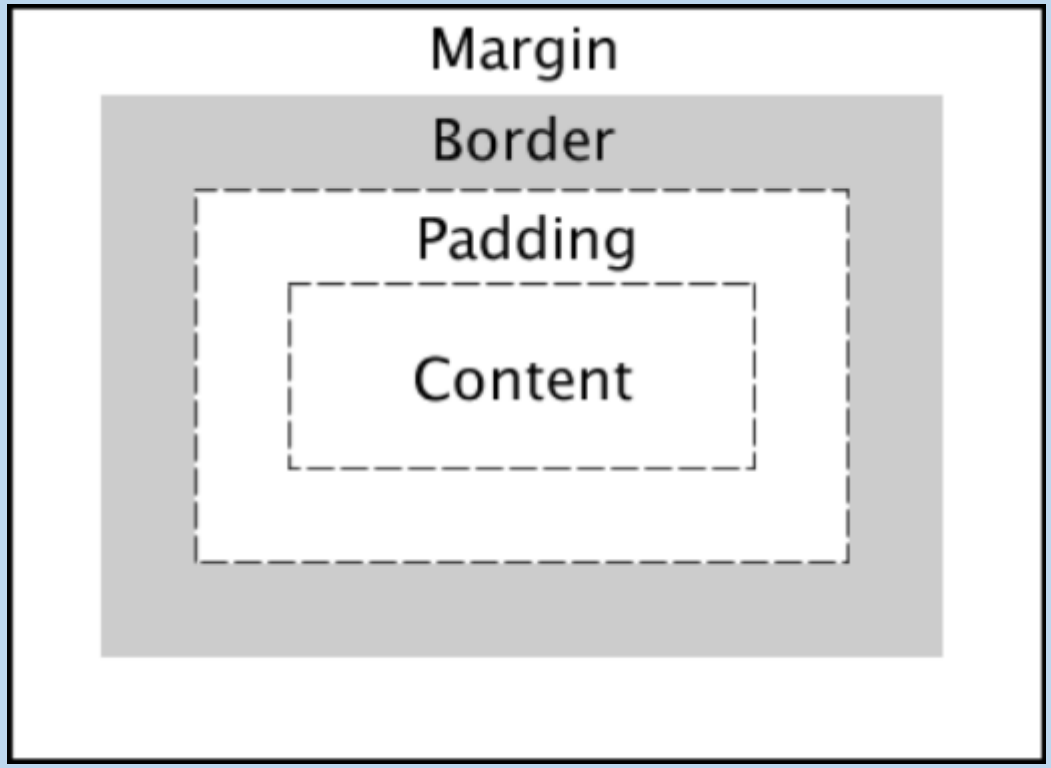
Layout

"구획을 나누고 적절한 정보를 배치하는 것"

Layout

레이아웃을 구성하기 위해서는
요소의 크기를 결정하고, (Box-Model)
어떻게 보여줄지, (Display)
어디에 놓을지, (Position)
배치 순서는 어떻게 할지, (Z-Index)
결정해야 합니다.

오늘은 CSS를 이용하여
위의 것들을 다루는 방법에 대해 알아봅니다!



HTML의 모든 요소는

“Box-Model”

로 이루어져 있다.

Q. 각 영역의 역할과 조정하는 방법은?

#쭉 훑어보고 모르는 내용이 있으면 복습하도록 합시다!

Display

"요소를 어떻게 표시할 것인가?"

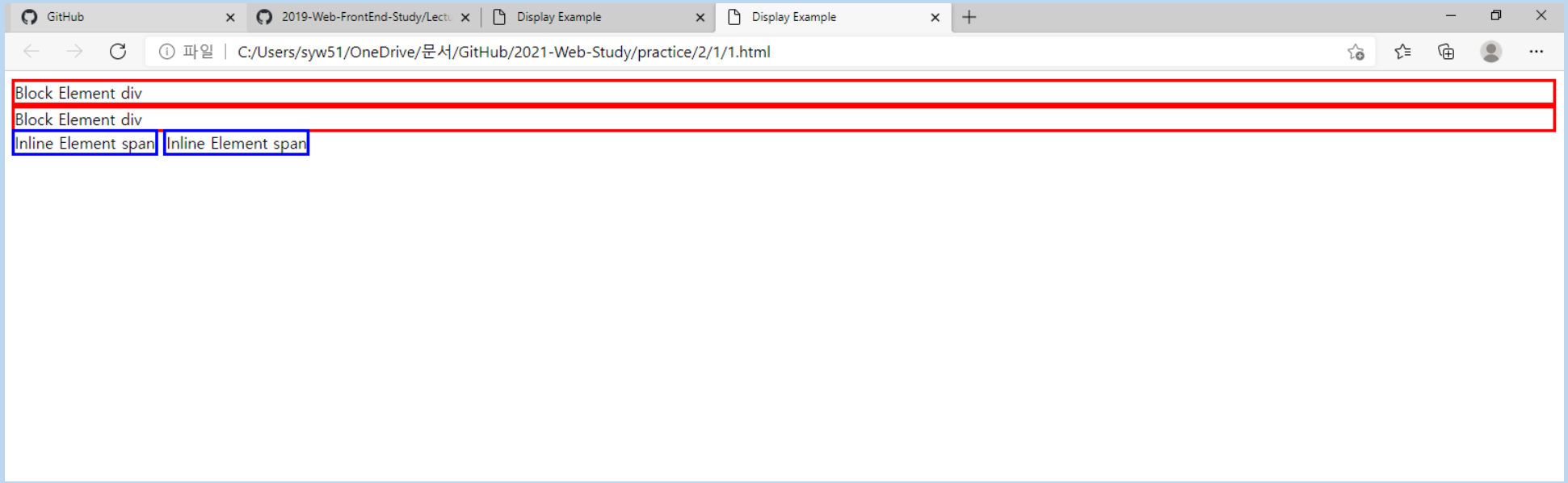
Display

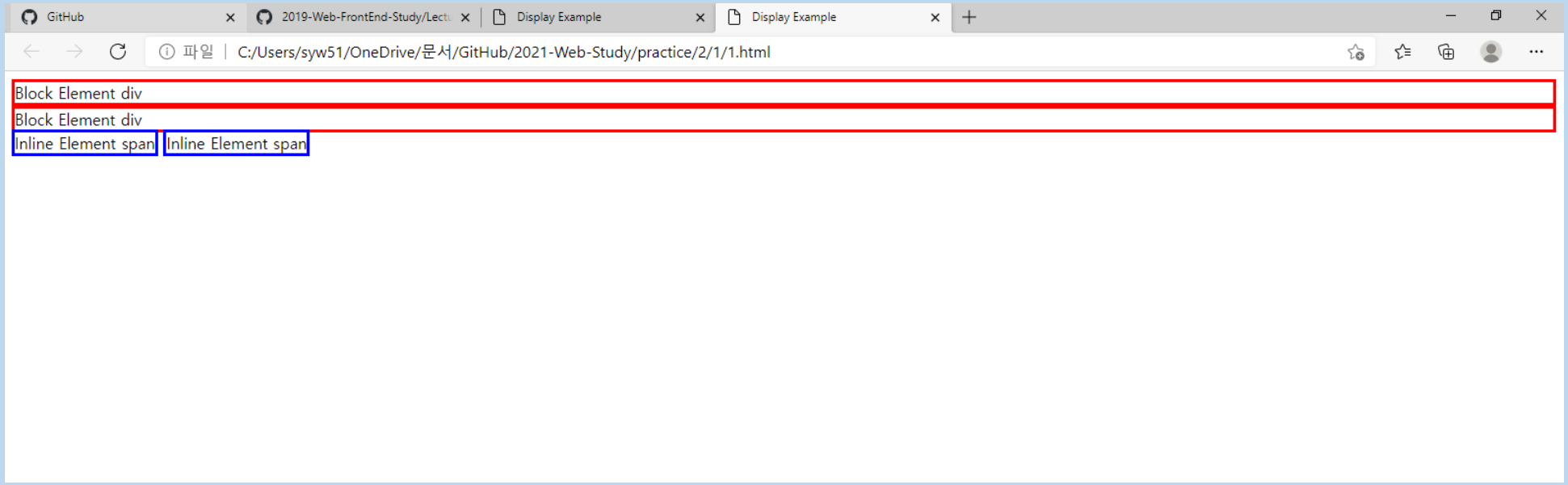
"요소를 어떻게 표시할 것인가?"

세가지 종류가 있습니다!

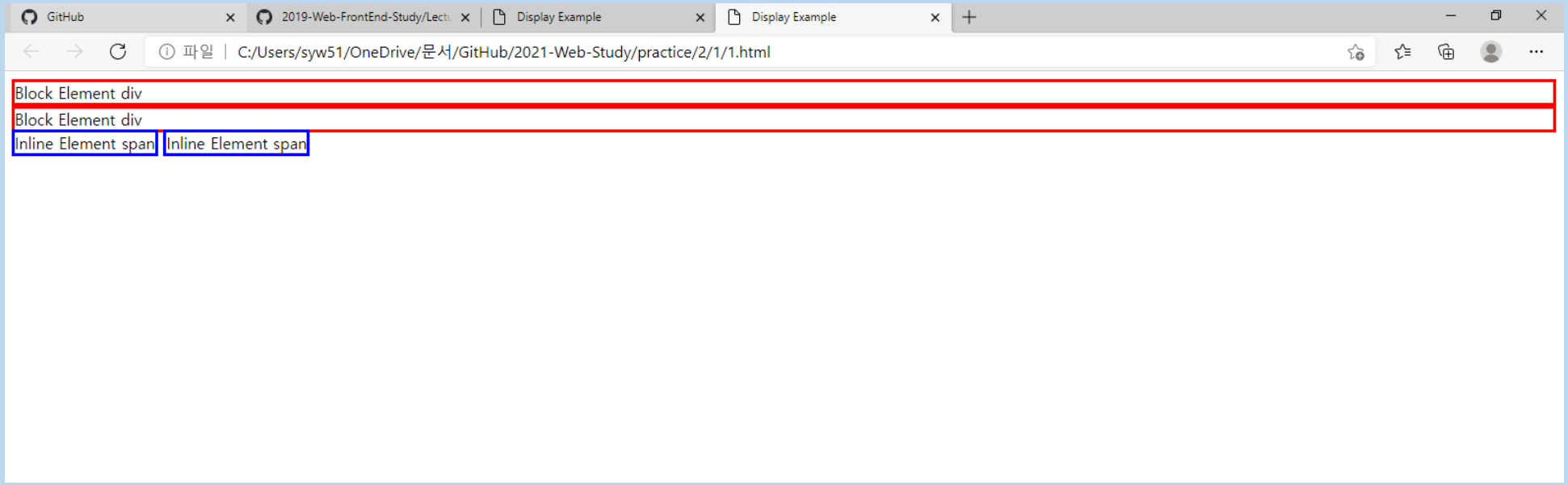
1. Inline
2. Block
3. Inline-Block

Practice 2-1
Display 실습





는 화면 한 줄 전체를 차지하며
은 화면 일부만 차지합니다.



는 화면 한 줄 전체를 차지하며 => Block Element

은 화면 일부만 차지합니다. => Inline Element

Inline vs Block

"줄을 바꾸지 않고 다른 요소와 함께
한 행에 위치하려 합니다"

<a>
 <button> <img>
<input>
<sub> <sup> <textarea>
...

"한 줄을 완전히 차지하려 합니다"

<canvas> <footer>
<form> <h1> <header>
 <p> <section>
<table> <ui> <video>
...

Inline vs Block

"줄을 바꾸지 않고 다른 요소와 함께
한 행에 위치하려 합니다"

1. 상, 하단 Margin 속성이 적용되지 않습니다.
상, 하단 여백은 line-height 속성으로 적용합니다.

2. Width, Height 속성이 적용되지 않습니다.
내부 요소의 부피 (content)에 따라 결정됩니다.

"한 줄을 완전히 차지하려 합니다"

1. 기본적으로 Width 100%로 정의되어 있습니다.

2. Margin, Width, Height 속성이 모두 적용 가능합니다.

#Margin, Width, Height가 변경되지 않는다면 Inline인지 의심해주세요!

그렇다면 display 속성을 변경시킬 수 있나요?

YES!

```
div {
    display: inline;
}
```

의 Block 속성을 Inline 속성으로 변경!

Inline 속성을 Block 속성으로 변경하는 것도 가능합니다!

Inline-Block

도 있습니다!

```
p {
    display: inline-block;
}
```

1. Inline 속성처럼 Element를 한 줄에 같이 표현
2. Block 속성처럼 Margin, Width, Height 속성 적용이 가능

Position

"요소의 위치는 어디로 할 것인가?"

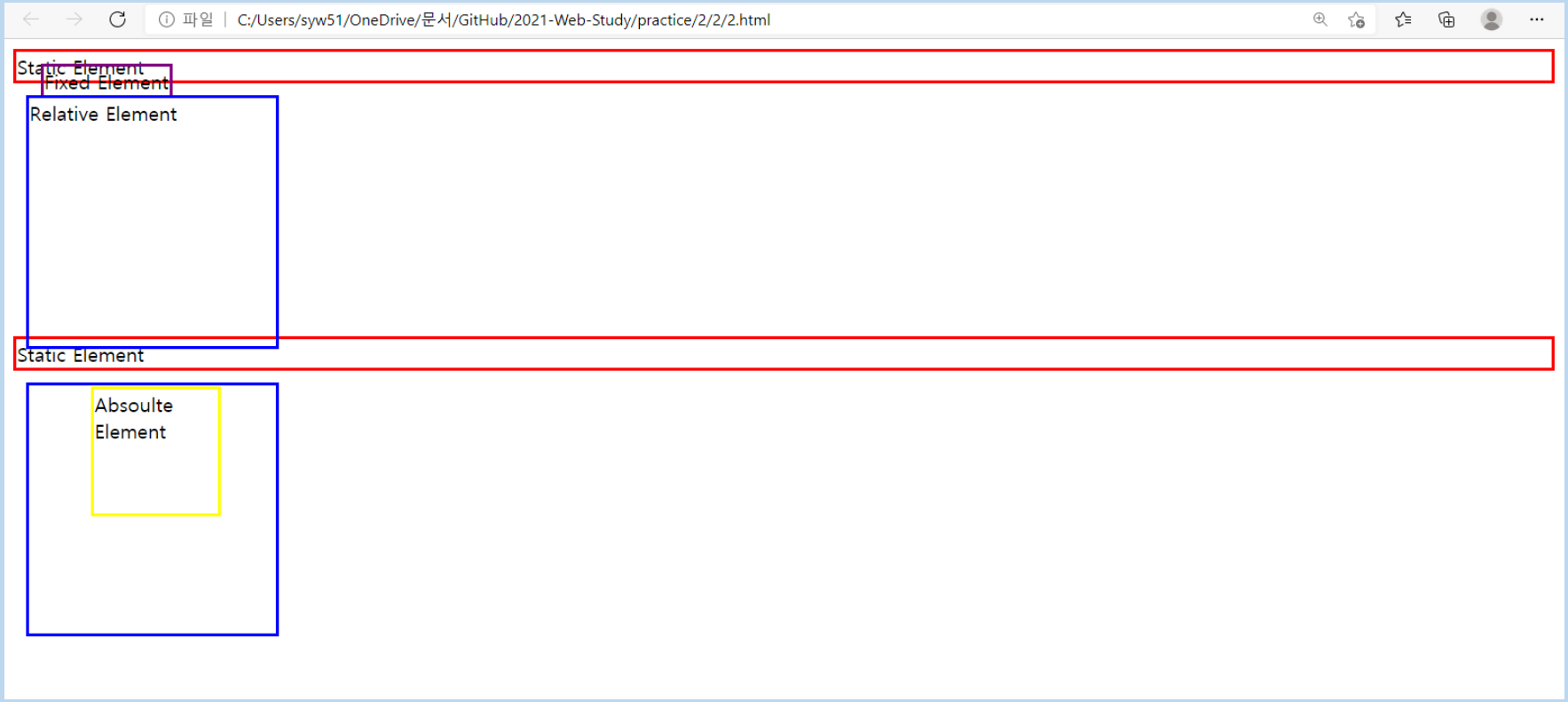
Position

"요소의 위치는 어디로 할 것인가?"

네가지 종류가 있습니다!

1. Static (Default)
2. Relative
3. Fixed
4. Absolute

Practice 2-2
Position 실습



Static

기본 값이며, Element를 위에서 아래로
순서대로 배열시켜줍니다.

Relative

상대적인 위치를 결정할 수 있습니다.
따라서 Static 속성과 다르게
Top, Right, Bottom, Left 속성 지정이 가능합니다.

```
.relative{
    position: relative;
    border: 3px solid blue;
    left: 10px;
    top: 10px;
    width: 200px;
    height: 200px;
}
```

#Relative 속성은 다른 Element에는 영향을 주지 않습니다.

Absolute

Static 속성이 아닌 부모 Element에 대해
상대적인 위치를 결정합니다.

```
.absolute{
    position: absolute;
    border: 3px solid yellow;
    width: 100px;
    height: 100px;
    left: 50px;
    }
```

#만약 Static 속성이 아닌 부모 Element가 없다면 위치는 <body>에 대해 결정됩니다.

Fixed

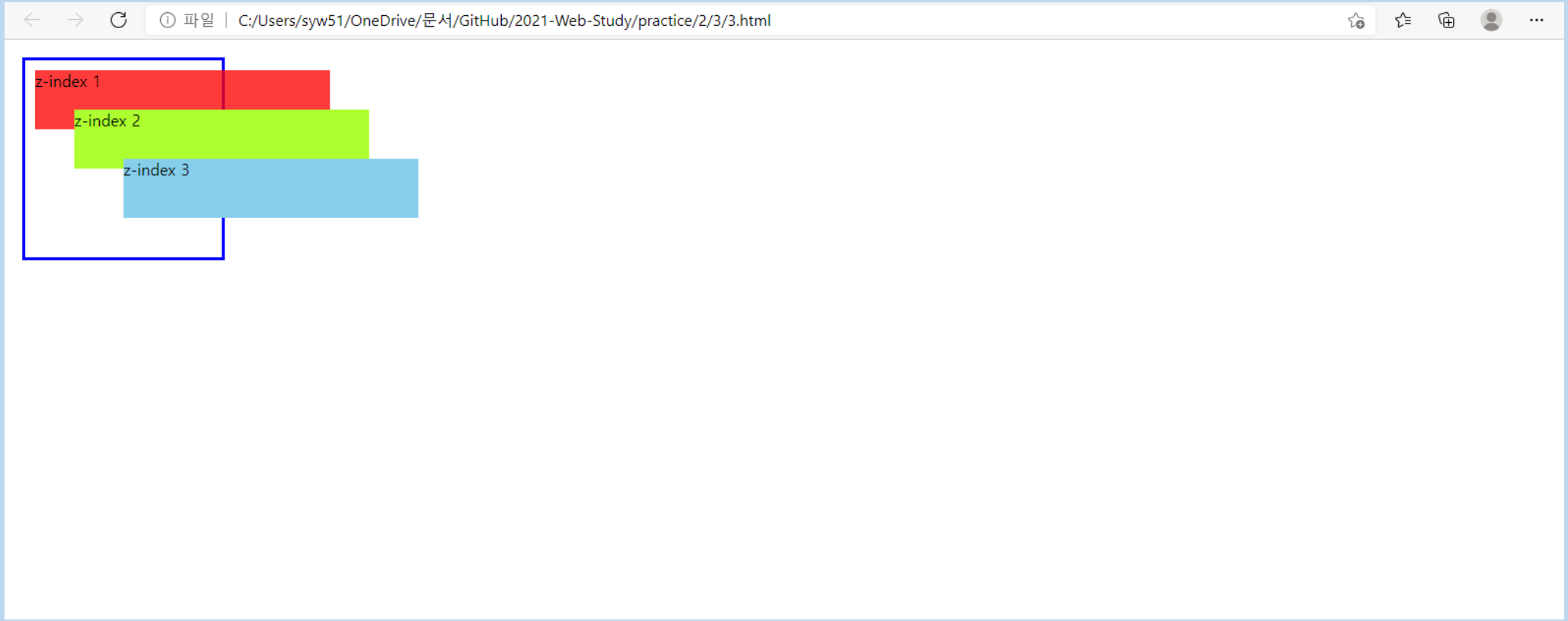
절대적인 위치를 결정할 수 있습니다.
스크롤로 화면을 내려도 사라지지 않고 같은 위치에 있습니다.
따라서 상단 메뉴 바 등을 구현할 때 유용합니다!

```
.fixed{
    position: fixed;
    border: 3px solid purple;
    top: 20px;
    left: 30px;
}
```

Z-Index

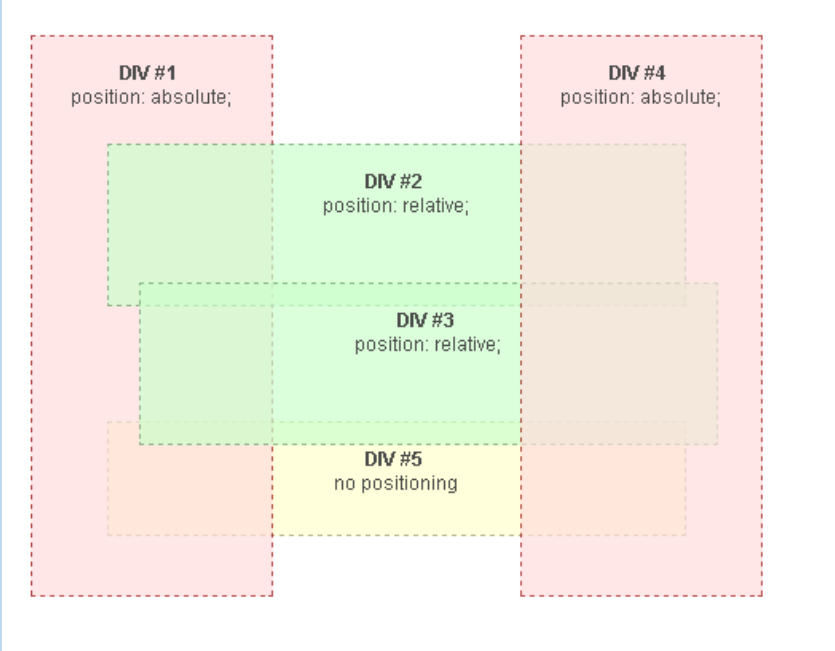
"요소의 배치 순서는 어떻게 할까?"

Practice 2-3
Z-Index 실습



Z-Index

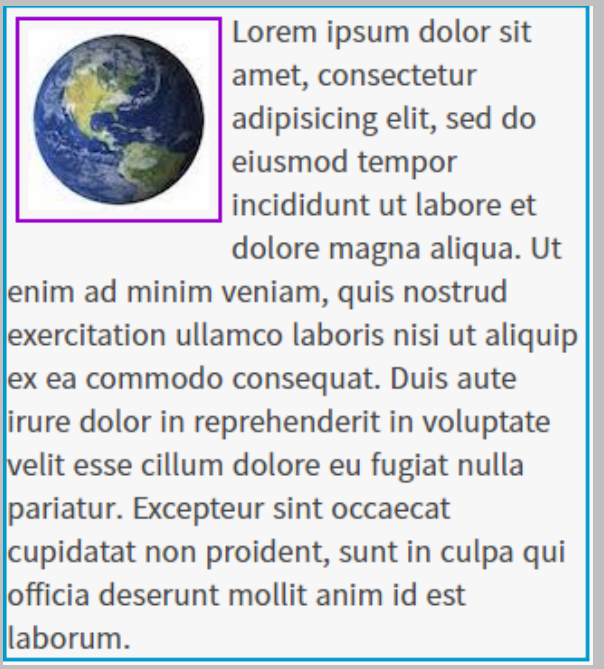
Z-Index가 클수록 위에(우선적으로) 표시됩니다.
#Z-Index 속성은 Static 속성이 아닌
Element에 적용 가능합니다.




```css
#zindex1 {
    position: absolute;
    top: 10px;
    left: 10px;
    background-color: rgba(255, 0, 0, 0.774);
    z-index: 1;
}
```

Float 속성에 대해서도 빠르게 알아봅시다.

CSS float 속성 - ofcourse



지금까지 배운
Box-Modeling, Position, Z-Index를 활용하여
자신이 원하는 레이아웃을 만들 수 있습니다.

그리고 지금 배우는

Flex-Box를 통해

더욱 **쉽고 강력한**

레이아웃을 제작할 수 있습니다.

과거에는 Position, float ,table을
주로 이용하여 레이아웃을 구성하였습니다.

최근에는 CSS3부터는
Flex-box, Grid를 활용하여
레이아웃을 구성하는 추세입니다.

#그렇다고 Position, Table을 몰라도 되는 것은 아닙니다!

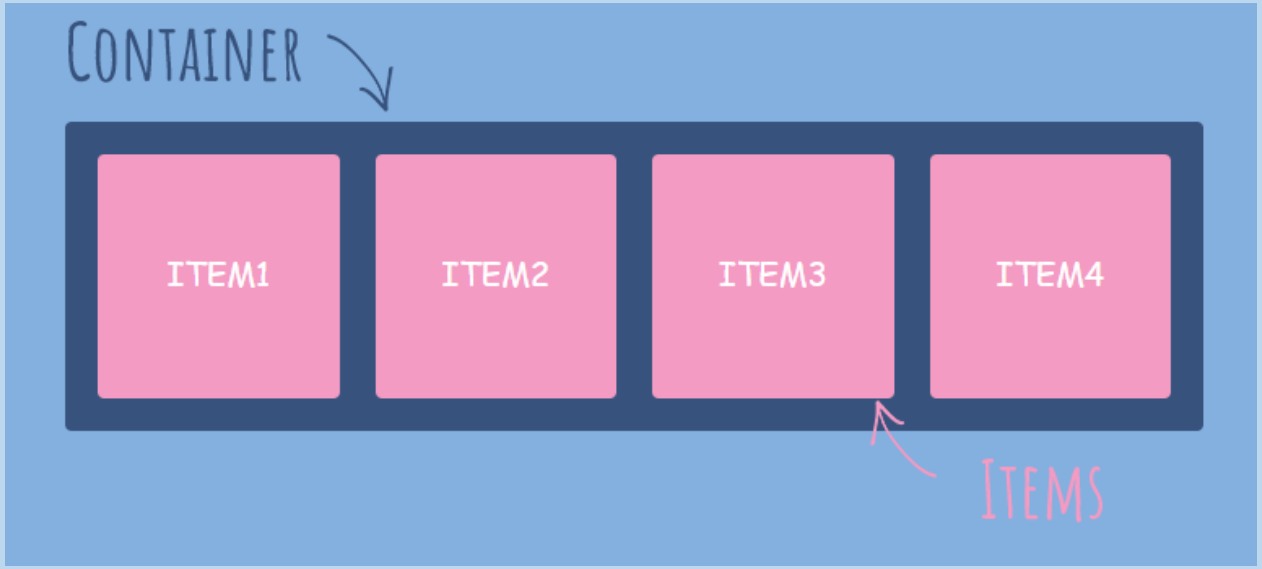
Flex Box

"요소들의 크기가 불분명하거나 동적일 경우,
각 요소들을 어떻게 배치할까?"

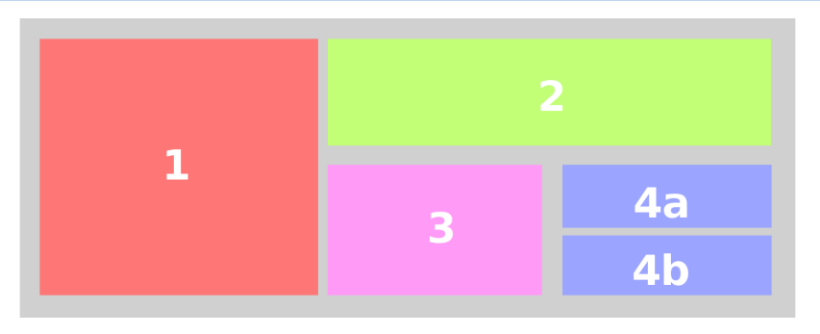
Flex Box

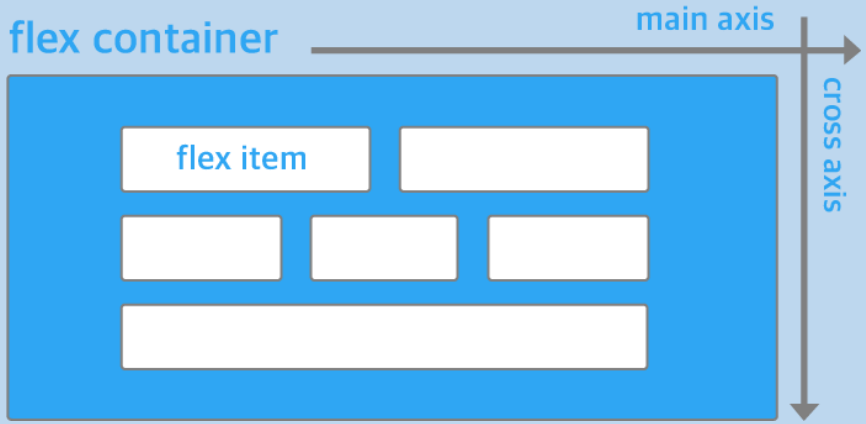
Flexbox는 Container와 Item으로 이루어집니다.

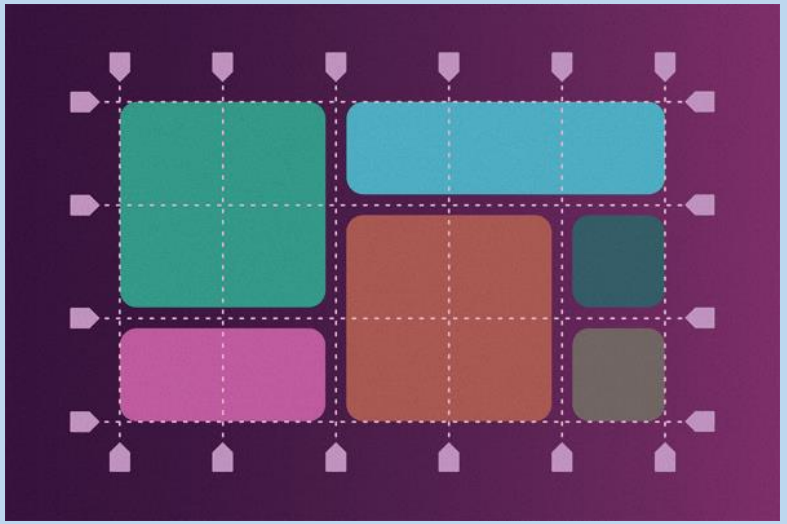
#비유하자면 상자와 물건



이제 Flexbox의 Container와 Item을 이용하여
아래와 같이 다양한 레이아웃을 구성할 수 있습니다.

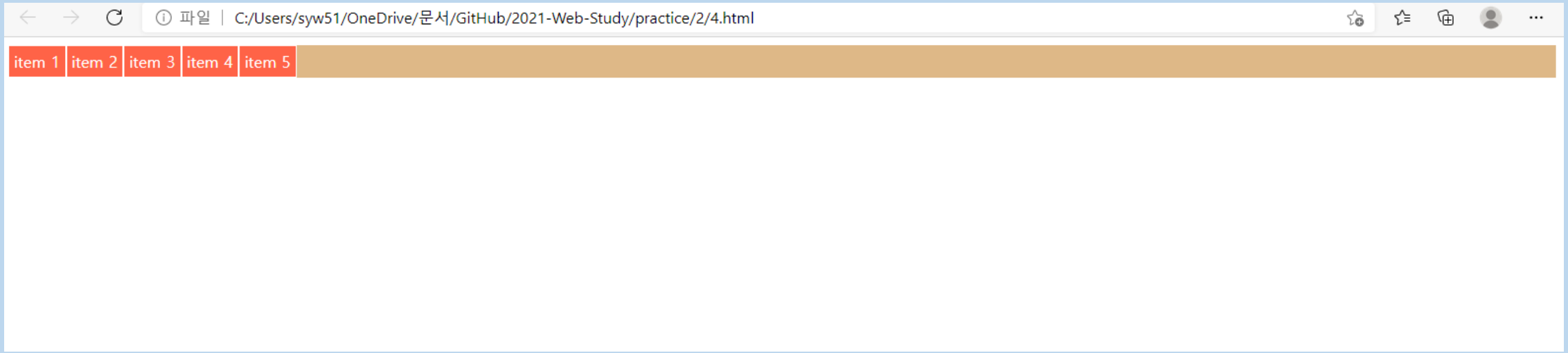


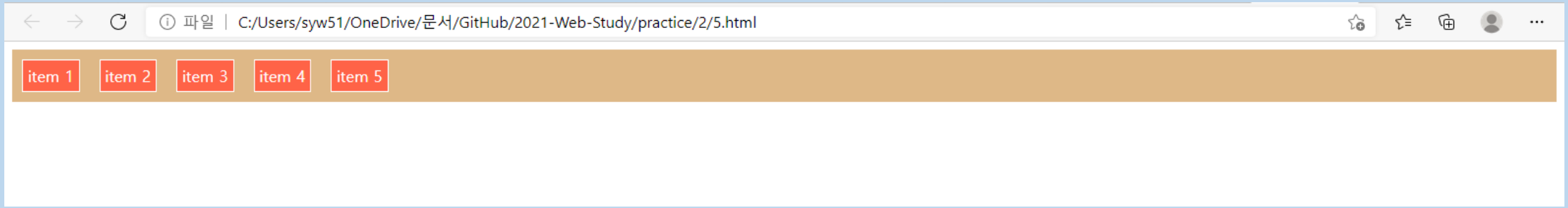




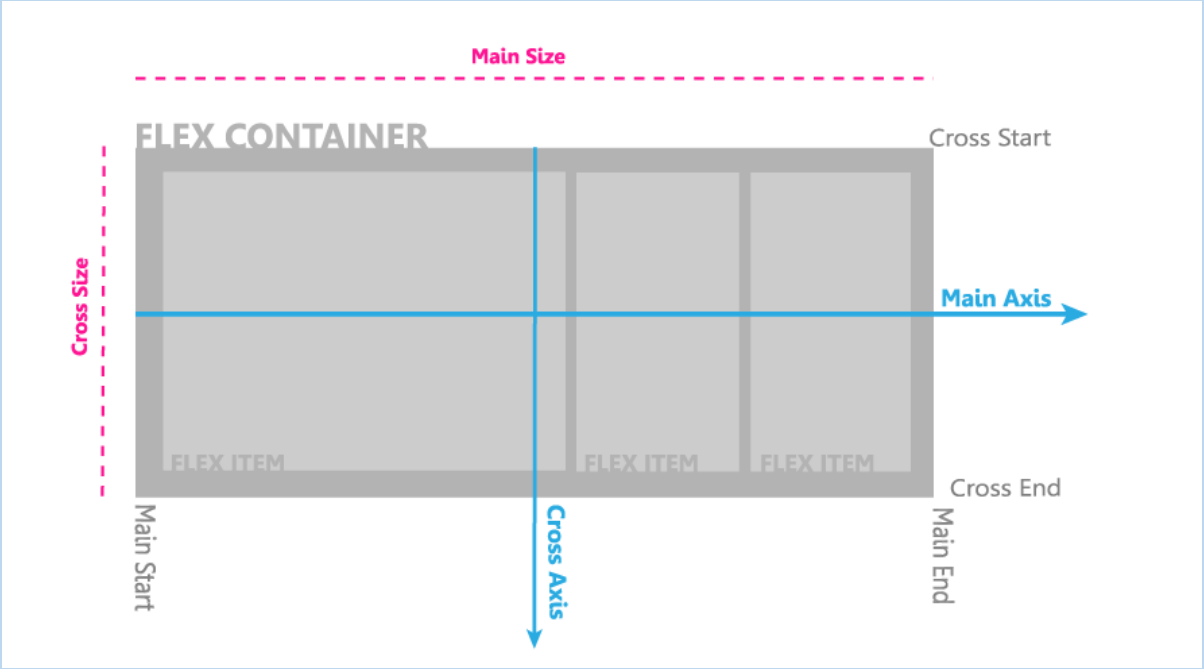
Practice 2-4
Flexbox 실습 - 1

Container와 Item이 무엇인지 알아봅시다.
Q. Flexbox를 어떻게 정의하나요?



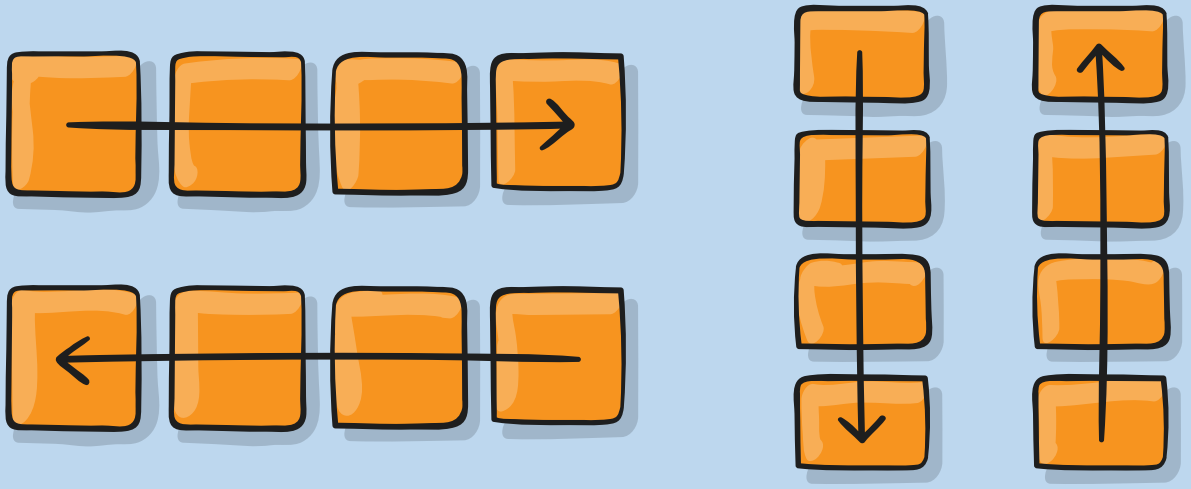


Container에 Display: Flex를 적용하면
item들은 가로 방향으로 배치되고,
자신의 Content만큼의 width만큼 차지하게 됩니다.
그리고 Container의 높이는 item의 높이가 됩니다.



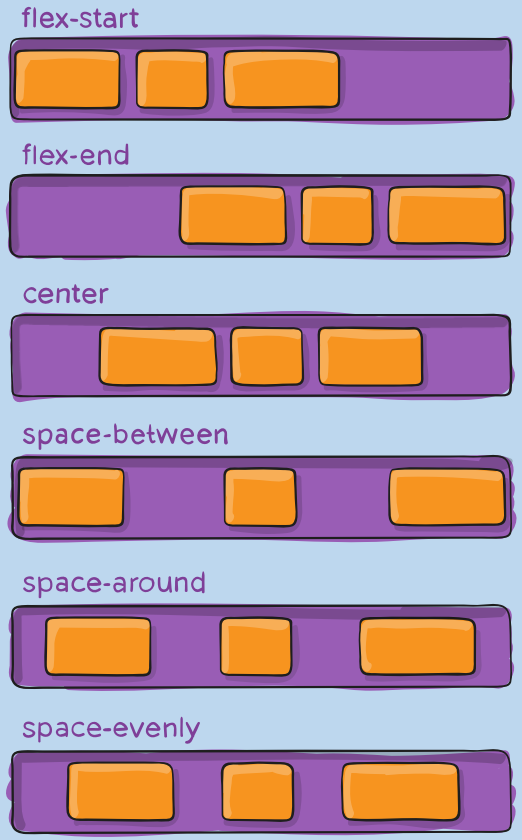
Container는 위와 같은 속성들로 구성되어 있습니다.
여기서 Item들이 배치되는 축인 Main Axis는 꼭 기억해주세요!

row | row-reverse | column | column-reverse



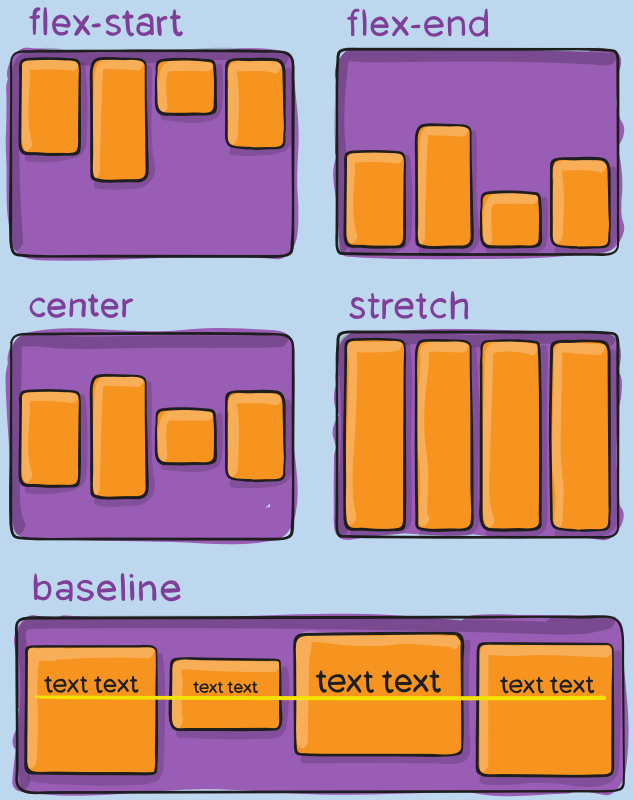
```
flex-direction: row;
```

Main Axis의 종류는 위와 같고 변경이 가능합니다.
#기본 값은 row입니다.



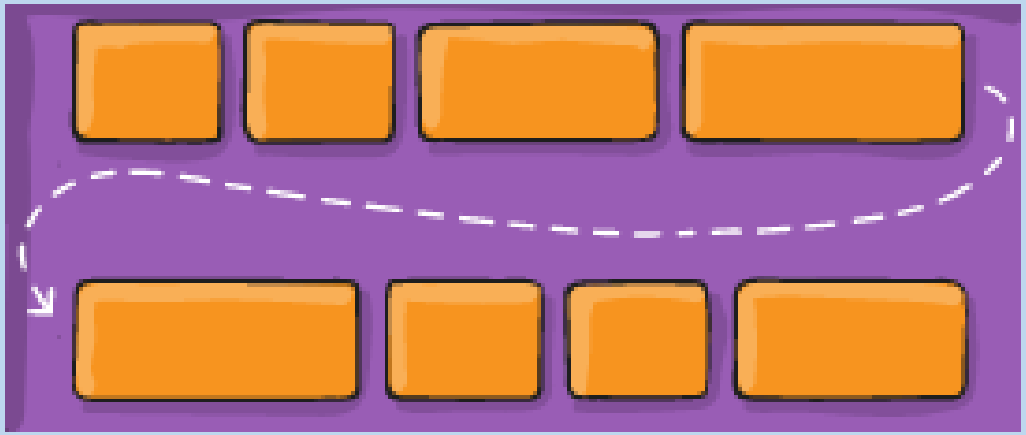

```
justify-content: flex-start;
```

Justify-content를 사용하여 Item의
배치 방법을 정할 수 있습니다.



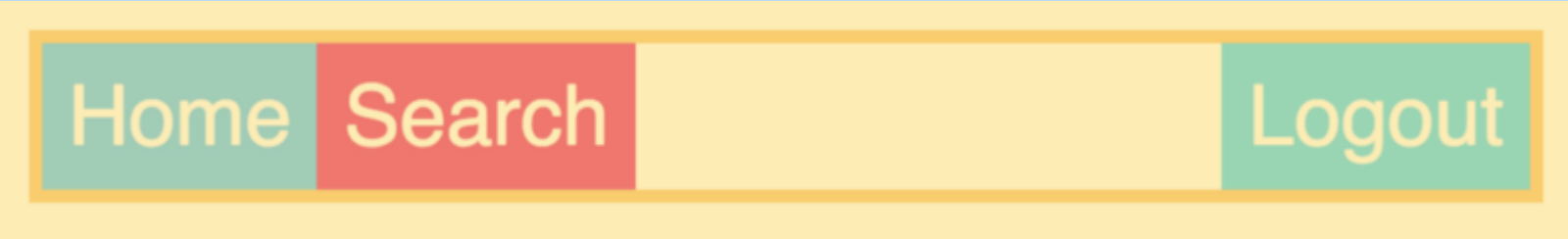

```
align-items: flex-start;
```

Align-items를 이용하여 Item의
배치 방법을 정할 수 있습니다. (주축의 수직방향)



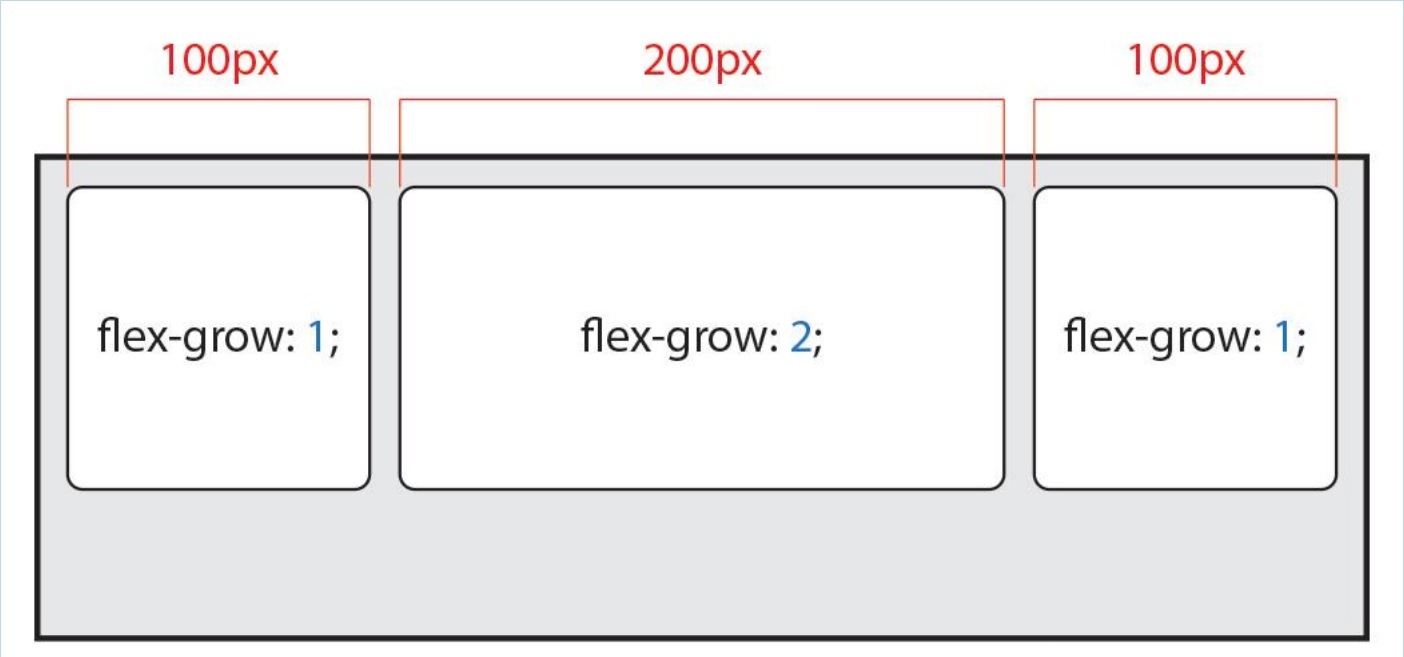
```
flex-wrap: wrap;
```

Flexbox는 기본적으로 한 개의 행으로 구성됩니다.
하지만 flex-Wrap 속성을 이용하면 Item이 하나의 행에
들어가지 않을 정도로 클 경우 Item을 다음 행에 배치시킵니다.

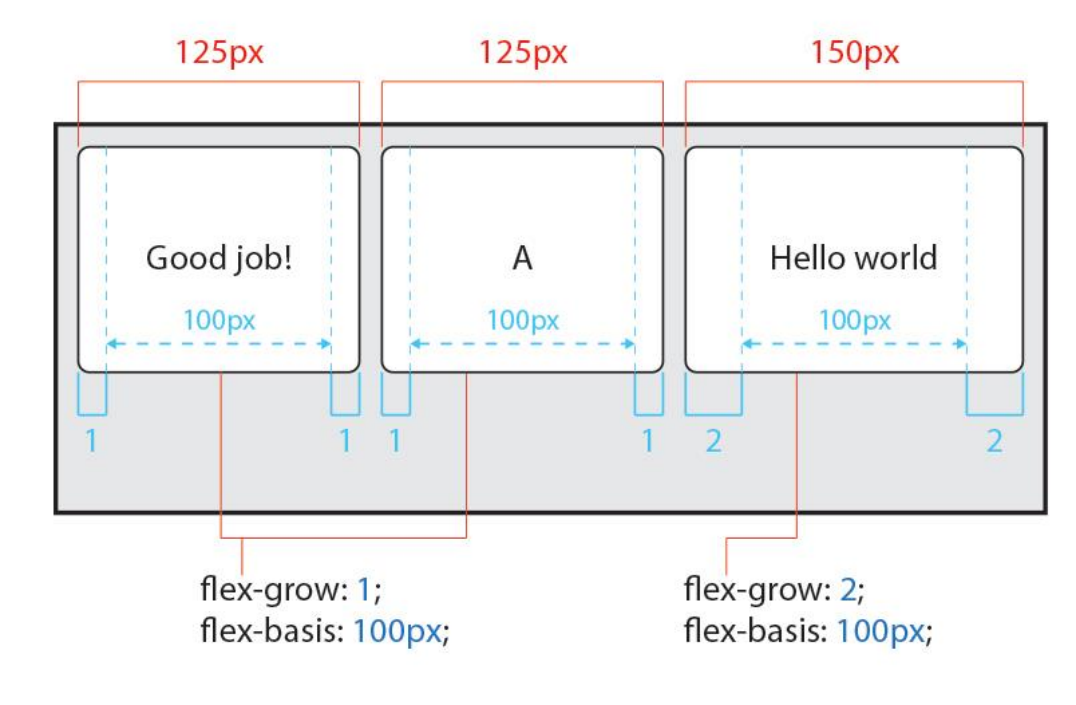



```
.logout {
margin-left: auto;
}
```

단일 Item의 제어도 가능합니다.
로그아웃 버튼만 오른쪽으로 배치하는 방법 중 하나입니다!



flex-grow 속성으로 Item의 비율로 너비를 설정할 수 있습니다.
위처럼 각 Item에 flex-grow를 지정하면
값의 비율만큼 너비를 가지게 됩니다.



flex-basis로 Item의 공간 배분 전 기본 너비를 정할 수 있습니다.

아래 자료도 함께 Flexbox를 공부해보세요!

이번에야말로 CSS Flex를 익혀보자 – 1분코딩 (studiomeal.com)

Flexbox 문법이 정리된 사이트입니다.

A Complete Guide to Flexbox | CSS-Tricks (css-tricks.com)

Flexbox로 만들 수 있는 10가지 레이아웃

flexbox로 만들 수 있는 10가지 레이아웃 | WIT블로그 (nts-corp.com)

Flexbox로 만들 수 있는 10가지 레이아웃 2

flexbox로 만들 수 있는 10가지 레이아웃 (naver.com)

Flexbox 5가지 활용 예시

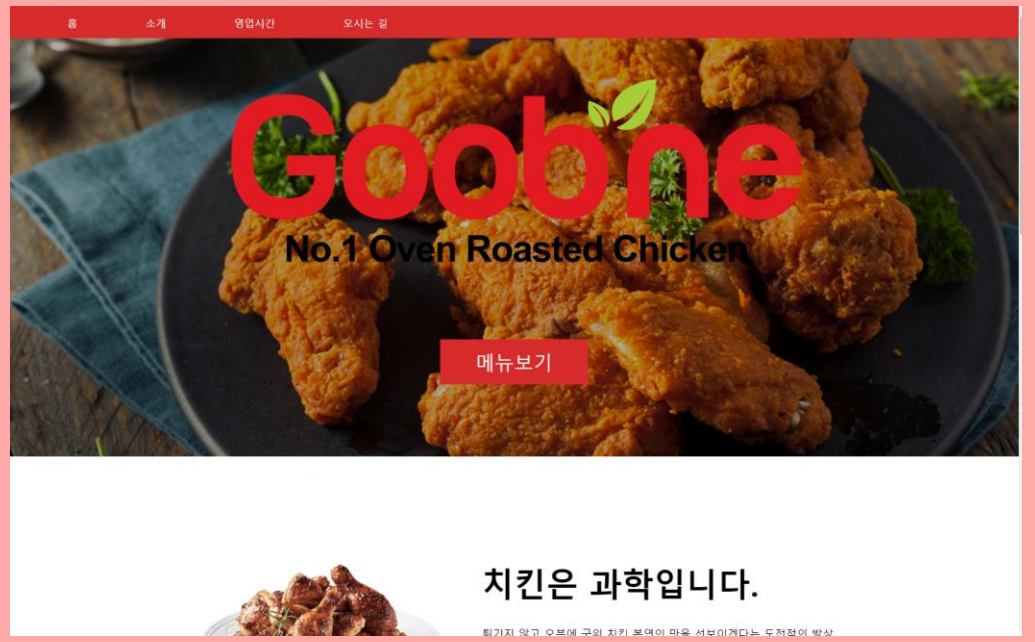
CSS Flexbox: 5 Real World Use Cases (ishadeed.com)

Practice 2-5
Flexbox 실습 - 2

Practice 2-4의 코드를 바탕으로
앞서 배운 flex-direction, flex-wrap 등의 속성들을
자유롭게 실습하여 봅시다.

2주차 프로젝트!

‘랜딩 페이지’ 를 제작해봅니다.
예제처럼 하나의 페이지를 스크롤하는 형식이며
상단에는 **고정된 메뉴바**가 존재하며,
2개 이상의 **주제별 Section**이 존재해야 합니다.
이외 페이지의 주제, 디자인 등은 모두 자유입니다.
(주제 Ex. 회사, 게임, 영화 ..)



수고하셨습니다.

다음 시간에는

'HTML/CSS 심화'

에 대해 알아봅니다.