지난번 시간에는 미디어쿼리를 이용해서 반응형웹을 경험해보았고 float를 이용한 배치로 데스크탑 레이아웃과 태블릿 레이아웃을 만들어 보았습니다.

이번 시간에는 애니메이션과 트랜지션 , 플렉스박스를 이용한 레이아웃 배치, 부트스트랩을 해보고 css는 끝을 내도록 하겠습니다.

마크업을 담당하는 마크업개발자나 퍼블리셔는 종종 “이 레이아웃은 구현이 불가능 합니다.”, “css로는 안됩니다. ” 라고 이야기합니다. 보통 페이지 레이아웃은 display,float,position등과 같은 속성을 사용해서 구현 하는데 사용하면 구현 방법이 복잡하고 레이아웃을 표현 하는데 많은 한계가 있습니다.

그래서 css3에서 짜잔 하고 등장한 것이 flexbox입니다. Flex박스를 사용하려면 요소의 크기와 순서를 유연하게 배치할 수 있습니다.

Flex박스는 뷰포트나 요소의 크기가 불명확하거나 동적으로 변할 때에도 효율적으로 요소를 배치, 정렬,분산할 수 있는 방법을 제공하는 것! 입니다. 플렉스 박스의 장점을 한마디로 표현하면 복잡한 계산없이 요소의 크기와 순서를 유연하게 배치할 수 있습니다.

이 플렉스 박스는 ul과 li같이 부모요소는 flex containner가 되고 자식요소는 flex item이 됩니다. Flex item은 주축에 따라 정렬됩니다. 주축의 방향은 flex containner의 flexdirection 속성으로 결정합니다. 정하지않으면 기본 속성값 row로 적용되며 주축의 방향을 왼쪽에서 오른쪽방향으로 흐르게 합니다. Column값을 주면 위에서 아래방향으로 흐르게 됩니다.

부모 자식의 속성

* flex container 속성: flex-direction, flex-wrap, justify-content, align-items, align-content
* flex item 속성: flex, flex-grow, flex-shrink, flex-basis, order

자 이제 이렇게 사용해봅시다.

<style>

.container{

background-color: powderblue;

height:200px;

}

.item{

background-color: tomato;

color:white;

border:1px solid white;

}

</style>

</head>

<body>

<div class="container">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

<div class="item">4</div>

<div class="item">5</div>

</div>

자 이렇게 작성해보면 item 들은 block 이니 자신의 크기만큼 페이지에서 열을 가지겠죠 ? 그런데

display:flex;

를 컨테이너 값에 추가해주니 뭔가 달라집니다. Inline속성으로 바꿔주지 않았는데 contents크기만큼 크기가 줄어들고 태그들이 옆으로 올 수 있게 되었습니다. 부모가 플렉스로 되는 순간 하위의 item들이 변하는 것을 볼 수 있죠. 이게 플렉스박스의 기본입니다.

이제 박스의 내용을 원하는대로 정렬 해봅시다. 그럴 때 사용하는 속성이

Flex-direction 입니다.

flex-direction 속성의 기본값은 row입니다. 지금 현재처럼 왼쪽에 붙어있는 row속성을 뒤집어서 row-reverse 라고 속성값을 주면 오른쪽으로 촤락 붙죠 ? 우리는 이걸 기존에 하려면 float속성을 사용해서 right라고 주어서 사용했습니다. 그런데 container에 background color색상을 줘서 박스의 크기를 확인하려니 개판이납니다. Heigth값이 전부 깨지게 됩니다.

그리고 이 플렉스에 column 이라는 속성을 주게 되면 우리가 원래 보던 div 스타일로 변경이 되고 column. -revers 라는 속성값을 주게 되면 다음과 같이 반대에서 부터 시작합니다.

그럼 이전까지 뭐가 다른것이냐. 기존에도 inline 과 블락, float속성을 써서 사용할 수 있던거 아니냐 . ? 라고 물으신다면 heigth값을 주면 이해가 될것입니다.

방향을 row로 바꾸고 height값을 주니 block요소처럼 한쪽으로 쭈욱 뻗어나가는 박스의 끝이 가로열에서 세로로 쭈욱 뻗어나가게 되었습니다. 이게 가장 큰 flex-box의 특징입니다. 박스가 뻗어나가는 방향을 제어 할 수 있습니다.

계속해서 flext의 속성을 알아보죠.

제가 만약 row로 되어있는 2번째 컬럼에 크기를 주려고 하면

.item:nth-child(2){

Flex-basis : 300 등

}

을 주게 되면 이렇게 크기가 200이 됩니다. Column을 하게 되면 2번째 아이템의 높이가 200이 됩니다. 방향에따라서 크기를 지정 할 수 있게 됩니다.

그럼 이제 이 컬럼들이 플렉스 컨테이너를 차지하게 하고 싶으면 어떻게 하냐. 바로

Flex-grow라는 속성을 이용해서 사용하면 됩니다 . item에다가 flex-grow : 1 을 하게되니 모든 아이템 요소들이 여백을 1/5만큼 가진 것을 볼 수 있습니다. 만약 2번째 아이템에 grow:2 를 하게 되면 우선순위에 의해 2번째 아이템은 2/6 가지게 된 것을 볼 수 있습니다.

만약 item에다가 0을 주고 2번째에 값을 그대로 하면 변화가 없습니다. 3을줘도 변화가 없어요 그 이유는 2번을 제외한 나머지 아이템은 여백을 가지지 않고 2번째 컬럼에 1을 주던 1000을 주던 모든 여백을 독차지 하기 때문이죠.

이제 flex-shirink속성을 알아볼 것인데요. (2번 컬럼이 큰상태로 row)우리가 브라우저의 크기를 이렇게 줄였을때 보시면 안에있는 아이템의 박스 크기들또한 작아집니다. 그럴때 이 플렉스-shrink속성에 0을 주게되면 컨테이너가 작아져도 2번 컬럼은 크기가 작아지지 않는 모습을 보입니다. 만약 1을 주게되면 다시 크기가 변하게 됩니다. 만약 2번과 3번컬럼에 각각 2와 3을 주게되면 줄어드는 크기는 2/5 , 3/5만큼 줄어들게 될 겁니다.

그래서 정리하면 플렉스 쉬링크는 컨테이너가 작아질 때 안에 있는 컬럼들이 부모가 줄어들 때 일정비율로 아이템의 크기를 변환 하는 것이다. 고 말 할 수 있습니다.

플렉스로 스크롤없는 100% 레이아웃 만들기

그냥 float속성으로 스크롤 없는 레이아웃 만드려면 헤드와 푸터만큼 콘텐츠의 크기를 마진으로 제거해주어야 합니다. 그런 ㅏ

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

html,

body {

*height*: 100%;

}

.wrap {

*display*: flex;

*flex-direction*: column;

*height*: 100%;

}

.header {

*height*: 65px;

*background-color*: #00c73c;

}

.tablist {

*height*: 70px;

*box-shadow*: 0 2px 2px 0 rgba(0, 0, 0, 0.1);

*background-color*: white;

}

.content {

*display*: flex;

*flex*: 1;

}

.aside {

*display*: flex;

*flex*: none;

*width*: 400px;

*background-color*: #bfbab078;

}

.main {

*display*: flex;

*flex*: 1;

*overflow*: auto;

}

html,

body {

*height*: 100%;

}

.wrap {

*display*: flex;

*flex-direction*: column;

*height*: 100%;

}

.header {

*height*: 65px;

*background-color*: #00c73c;

}

.tablist {

*height*: 70px;

*box-shadow*: 0 2px 2px 0 rgba(0, 0, 0, 0.1);

*background-color*: white;

}

.content {

*display*: flex;

*flex*: 1;

}

.aside {

*display*: flex;

*flex*: none;

*width*: 400px;

*background-color*: #bfbab078;

}

.main {

*display*: flex;

*flex*: 1;

*overflow*: auto;

}

</style>

</head>

<body>

<div id="wrap" class="wrap">

<header class="header"></header>

<div class="tablist"></div>

<div class="content">

<div class="aside"></div>

<div class="main"></div>

</div>

</div>

</body>

</html>

그러나 이 플렉스 박스에서도 단점이 존재하는데 바로 ie10부터 이 플렉스박스를 지원하는 것입니다. Ie10이 공공의 적이지만 한국에서는 아직 ie 10이하의 노망난 컴퓨터를 사용하는 사람들도 좀 있기 때문에 아직 도입하기에는 조금 힘든 편입니다.

대신 모바일 브라우저에서는 대부분 사용이 가능하니 ie10이하의 브라우저를 버리실 분들은 새로운 세대를 위해 써보는 것도 좋을 듯 싶습니다.