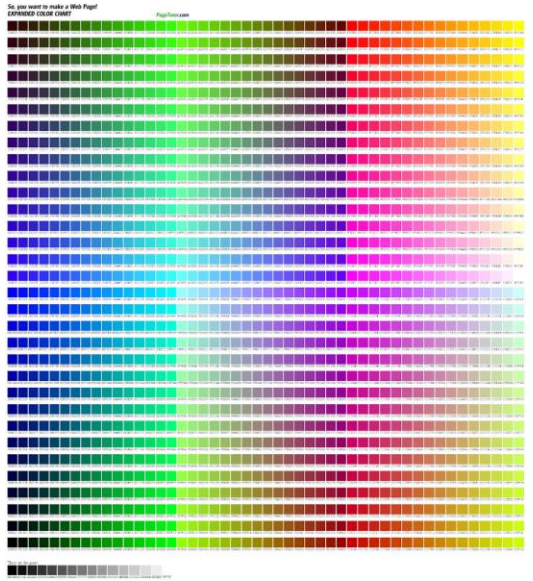
# Hex Code

#361999 #78FFF1 #FF6495



- RGB(Red Green Blue) 각각의 값을 16진수로 만들어 이어붙인것
- CSS 에서 색 이름 대신 Hex Code 사용 가능
- 추천 검색어: web color trend

```
<div class="font-test">hex code 연습</div>
```

```
/*index.css*/
.font-test {
  color: #FF6495;
}
```

#FF6495

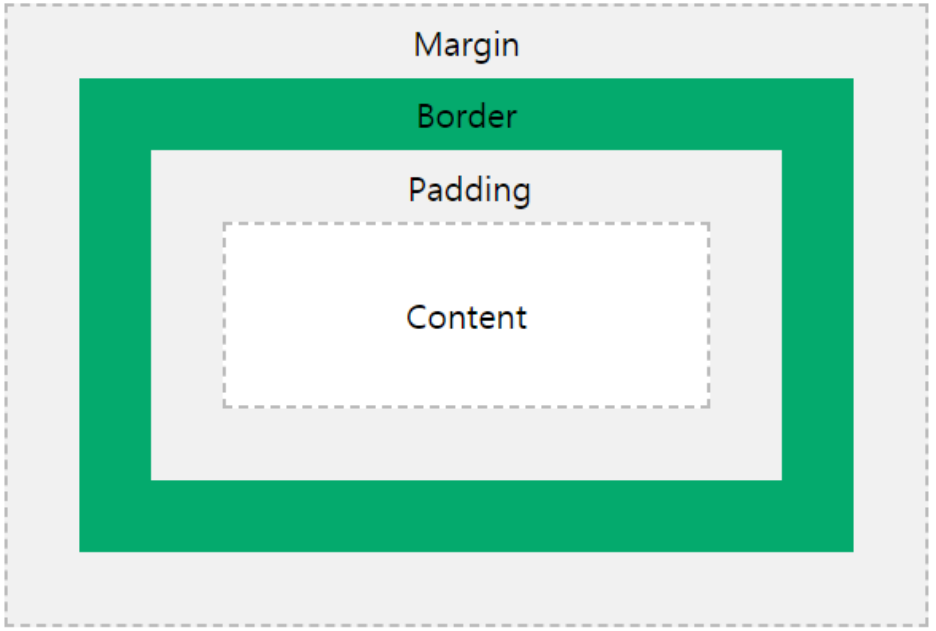
결과:

hex code 연습

# 박스모델

- 모든 태그는 박스(Box)로 이루어져있다.
  - 구글 개발자도구(F12) element 부분 참조

- border 태그를 둘러싸는 경계선
- padding 태그의 컨텐츠와 보더간의 간격
- margin 태그와 태그간의 간격
  또는 태그와 화면간의 간격

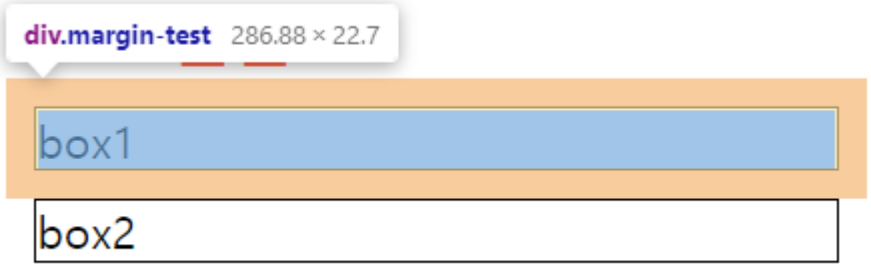


```html
<body>
    <div class="margin-test">box1</div>
    <div class="margin-test">box2</div>
</body>
```

```css
/*index.css*/
/*box model 연습*/
.margin-test {
  border: 1px solid black;
  /* margin-top: 10px;
  margin-right: 10px;
  margin-bottom: 10px;
  margin-left: 10px; */
  margin: 10px 10px 10px 10px;
}
```

- border는 경계선
- margin의 순서는
  - top
  - right
  - bottom
  - left
- 즉, 시계방향이며, 분리해서 쓸수도 있다.

결과:

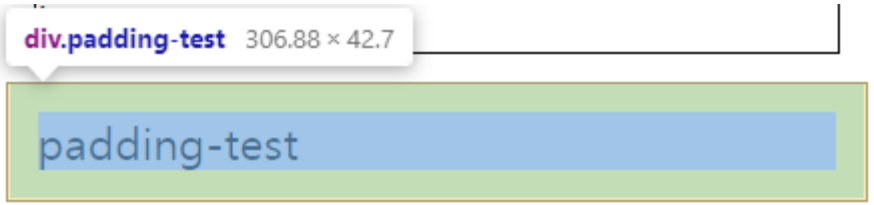



이 경우, 마진 겹침 현상이 나타난다.
마진이 큰 쪽의 태그를 따른다.

```html
<body>
    <div class="padding-test">padding-test</div>
</body>
```

```css
/*index.css*/
.padding-test {
  border: 1px solid black;
  padding: 10px 10px 10px 10px;
}
```

결과:




- 즉, 패딩은 태그의 보더와 컨텐츠와의 간격이다.

# background-color

- 태그를 채우는 색을 넣을수도 있다.

```css
.padding-test {
  border: 1px solid black;
  padding: 10px 10px 10px 10px;
  background-color: tomato;
}
```

# display

- display 해당 태그가 화면에 어떻게 나올지를 정의하는 property
- 모든 태그는 기본적으로 다음 세 가지 중 하나를 가진다
  - block 가로 전체를 차지함. 크기 지정 가능
    , <h1>
  - inline 쓰인 영역만 차지함. 크기 지정 불가능
    , <a>
  - inline-block 쓰인 영역만 차지하나, 크기 변경 가능
    <button>
- 추가적으로, 다음이 쓰인다.
  - none 화면에서 아예 안 보이게 만들 때
    JavaScript 와 같이 쓰임
  - grid 레이아웃 짤 때

# block 가로 전체 차지. 크기 지정 가능

```html
<div class="divTag">div tag</div>
```

```css
/*index.css*/
.divTag {
  border: 1px solid black;
  /* 기본값은 block인데 display는 아래와 같이 변경 가능 */
  /* display: inline; */
  /* block은 크기 변경 가능 */
  /* height: 500px; */
  /* width: 100px; */
}
```

결과:

div tag

대표적인 태그: , <h1>
화면 전체가 아니라, 정해진 영역의 가로를 말함

# inline 쓰인 영역만 차지. 크기 지정 불가능

```
<span>span tag</span>
```

```
/*index.css*/
span {
  border: 1px solid black;
  /* 기본값 */
  /* display: inline; */
  /* inline은 크기 변경 불가능 */
}
```

결과:

span tag

대표적인 태그: , <a>

즉, inline을 써야 할 의미없는 태그를 쓰고싶으면  대신  !!!

# inline-block 쓰인 영역만 차지하나, 크기 변경 가능

```
<button>button tag</button>
```

```
/*index.css*/
button {
  /* 기본값 */
  /* display: inline-block; */
  /* inline-block은 크기 변경 가능 */
  /*width: 100%;*/
  height: 50px;
}
```

결과:


대표적인 태그: <button>

# none

```
<div>안보임!</div>
```

```
/*index.css*/
div {
  display: none;
}
```

- 화면에서 안 보이게 하는 것
- 보통 JavaScript 와 결합해 사용

# Position

- 태그의 위치를 옮길 때 사용
- 단순히 옮기면 되는 간단한 문제가 아님
- 무엇을 기준으로 옮길 것인가
- 그 기준은 부모와 관련있다

```html
<div class="grandParent">
    grandDad
    <div class="parent">
        parent
        <div class="child">child</div>
    </div>
</div>
```

```css
.grandParent,
.parent,
.child {
    border: 1px solid black;
    /*position: static;*/
}
```

position의 기본값: static
이 경우, 움직이는것은 불가능

결과:

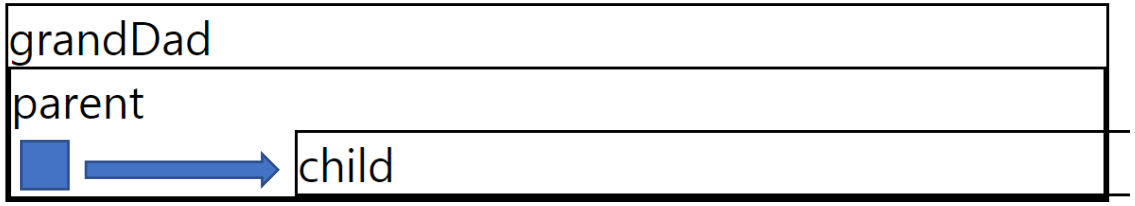
grandDad
parent
child

# 1. relative

```
.child {
    position: relative;
    left: 100px;
}
```

px등의 값을 안 주면 아무 변화 없음
움직임은 top, right, bottom, left 로 지정

결과:




left: 100px
이것은 왼쪽으로 100px 가라는 게 아니고,
왼쪽에 100px의 여백을 둔다는 뜻
부모 태그로부터 자신의 위치 결정

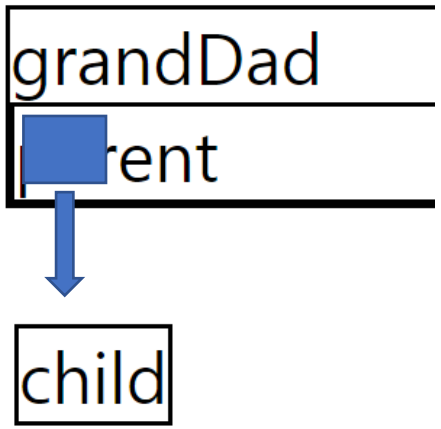
# 2. absolute

- static이 아닌 부모가 기준
- 케이스별로 나눠서 살펴보자

# absolute case1: 부모가 relative

```css
.parent {
    position: relative;
}

.child {
    position: absolute;
    top: 50px;
}
```

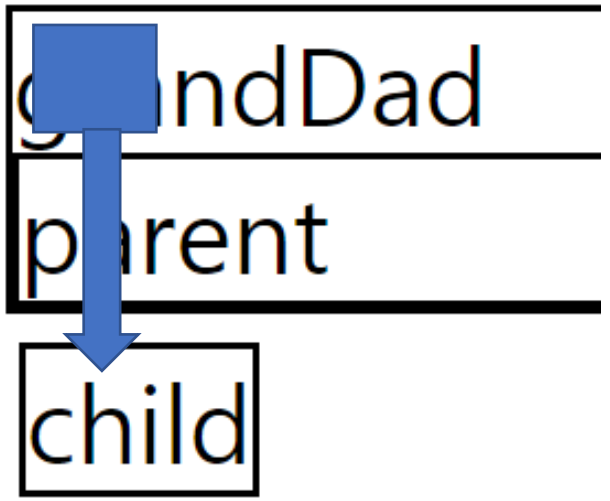



이 경우, 부모가 static 이 아니기 때문에 부모가 기준
parent 를 기준으로, 아래로 50px 내려옴

# absolute case 2:
## 부모가 static 이면서 조부모가 static 아닌 경우

```
.grandParent {
    position: relative
}

.child {
    position: absolute;
    top: 50px;
}
```

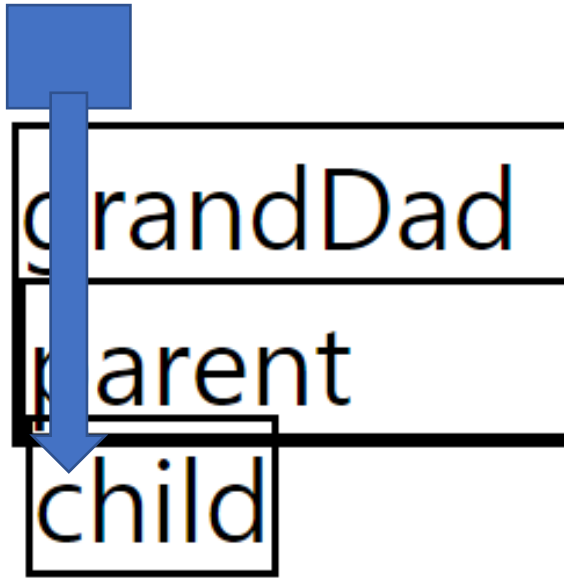



이 경우, 부모는 stati이고 조부모가 static이 아니기 때문에 조부모가 기준

조부모를 기준으로 50px 내려옴

# absolute case 3: 부모에게 아무것도 지정 안함

```
.child {
    position: absolute;
    top: 50px;
}
```




이 경우, 부모도 static이고 조부모도 static이기 때문에 화면 왼쪽 위가 기준

화면 왼쪽 끝을 기준으로 50px 내려옴

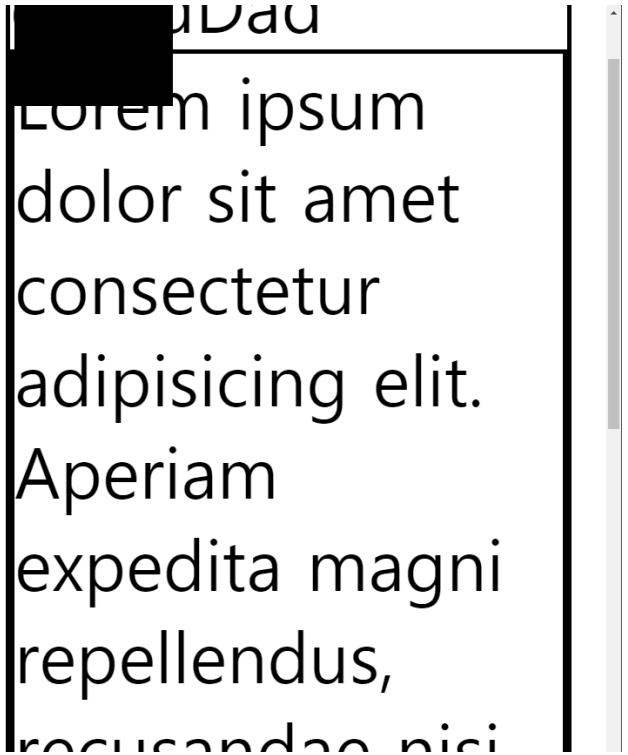
# 3. fixed

- 시조 태그의 위치가 기준
- 스크롤도 무시
- display는 더 이상 block 이 아님

```html
<div class="grandParent">
    grandDad
    <div class="parent">
        Lorem ipsum dolor sit amet consectetur
        adipisicing elit. Aperiam expedita magni
        repellendus, recusandae nisi
        ad perspiciatis dolor nulla? Et distinctio
        corrupti architecto debitis perferendis
        voluptatibus a, dolorum
        saepe rem nisi?
        <div class="child">child</div>
    </div>
</div>
```

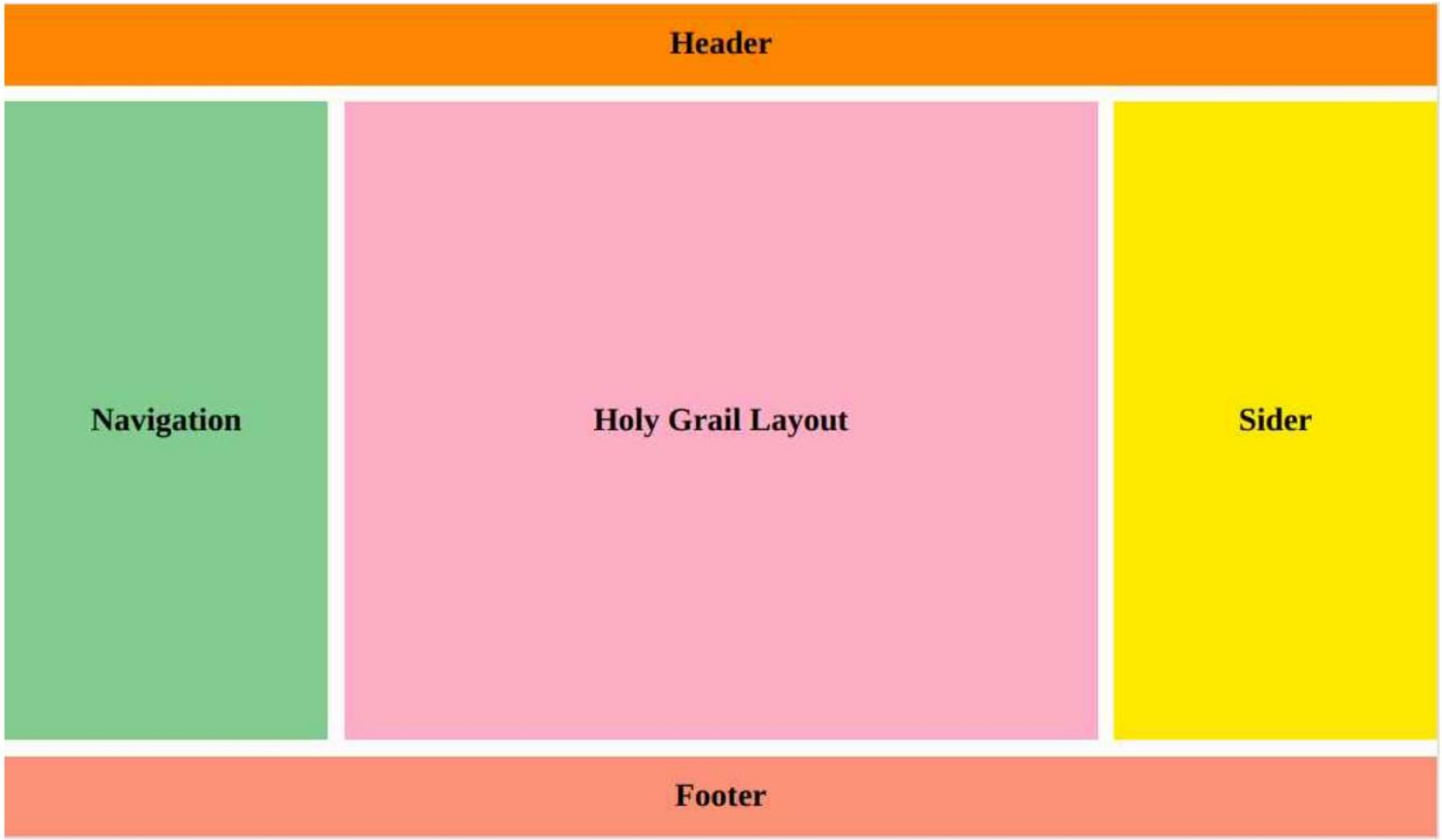
```css
.child {
    position: fixed;
    top: 0px;
    background-color: black;
}
```

결과:



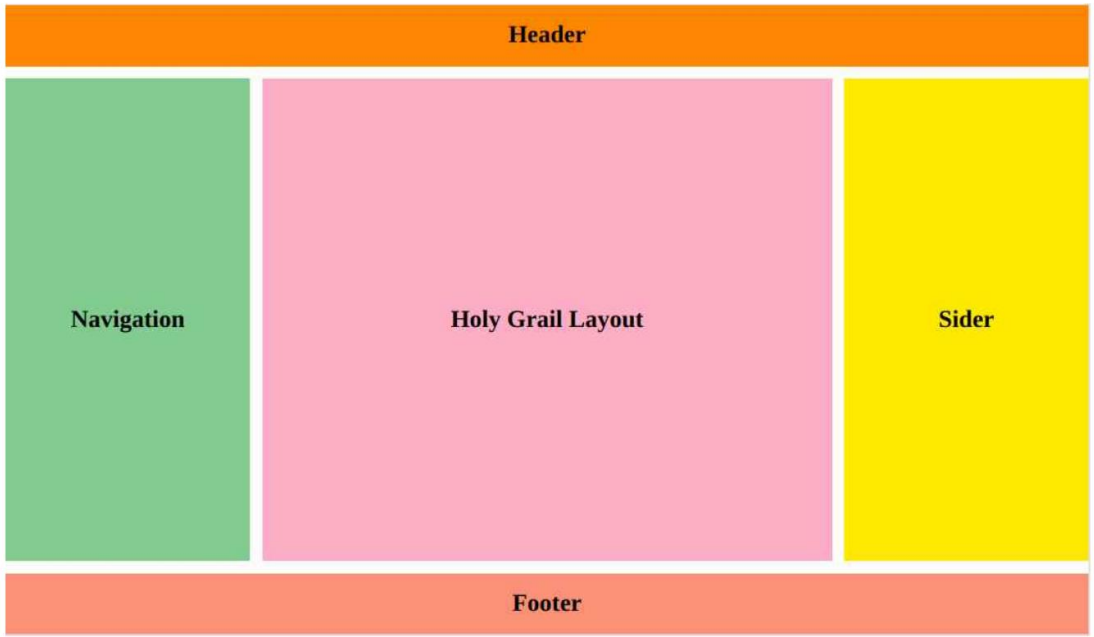

시조 태그를 기준으로 0px
스크롤을 내려도 움직이지 않음

# 레이아웃 grid 사용



```html
<div class="header">header</div>
<div class="nav">nav</div>
<div class="main">main</div>
<div class="aside">aside</div>
<div class="footer">footer</div>
```

```css
.header,
.nav,
.main,
.aside,
.footer {
    border: 1px solid black;
}
```




header

nav

main

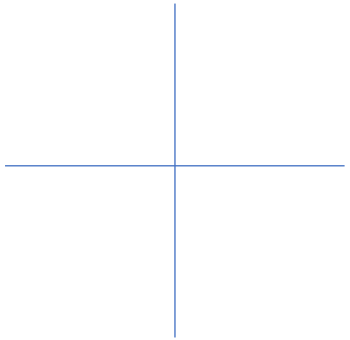
aside

footer

row column 외우기

# 부모 태그 만들기

- grid 는 부모태그에서 사용한다



가로 세로
로우 컬럼
행 열

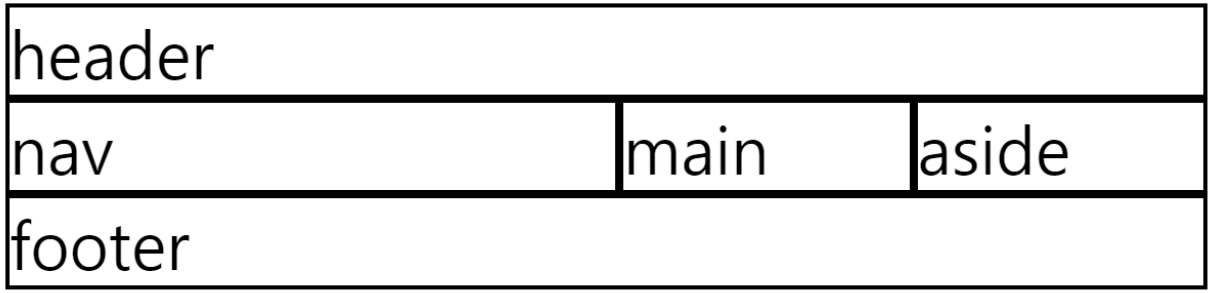
```html
<div class="header">header</div>
<div class="container">
    <div class="nav">nav</div>
    <div class="main">main</div>
    <div class="aside">aside</div>
</div>
<div class="footer">footer</div>
```

```css
.container {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
}
```

fr은 비율 의미
px 을 대신 사용시, 해당 크기로 고정
화면 크기 조절해 확인해볼 것

| header | | |
|---|---|---|
| nav | main | aside |
| footer | | |

```css
.container {
    display: grid;
    grid-template-columns: 150px 1fr 1fr;
}
```




nav 의 너비가 150px 로 고정되고
main 과 aside는 남은 비율에서 1대 1 차지
화면 크기 조절해 확인해볼 것

```css
.container {
    display: grid;
    grid-template-columns: 1fr 2fr 1fr;
}
```

셋의 비율이 1대 2대 1

| header | | |
|---|---|---|
| nav | main | aside |
| footer | | |

화면 크기 조절해 확인해볼 것

# column 뿐만 아니라 row 도 있을 것

```html
<div class="header">header</div>
<div class="container">
    <div class="nav">nav</div>
    <div class="main">
        Lorem ipsum dolor sit, amet consectetur
        adipisicing elit. Eum recusandae debitis
        laboriosam labore tempora repellat cupiditate
        non ab sed, deserunt ipsum reprehenderit
        nostrum iste suscipit quidem sit
        atque unde in!
    </div>
    <div class="aside">aside</div>
</div>
<div class="footer">footer</div>
```

```css
.container {
    display: grid;
    grid-template-rows: 1fr 2fr 1fr;
}
```

1:2:1

| header |
| --- |
| nav |
| Lorem ipsum dolor sit, amet consectetur adipisicing elit. Eum recusandae debitis laboriosam labore tempora repellat cupiditate non ab sed, deserunt ipsum reprehenderit nostrum iste suscipit quidem sit atque unde in! |
| aside |
| footer |

화면 크기 조절해 확인해볼 것