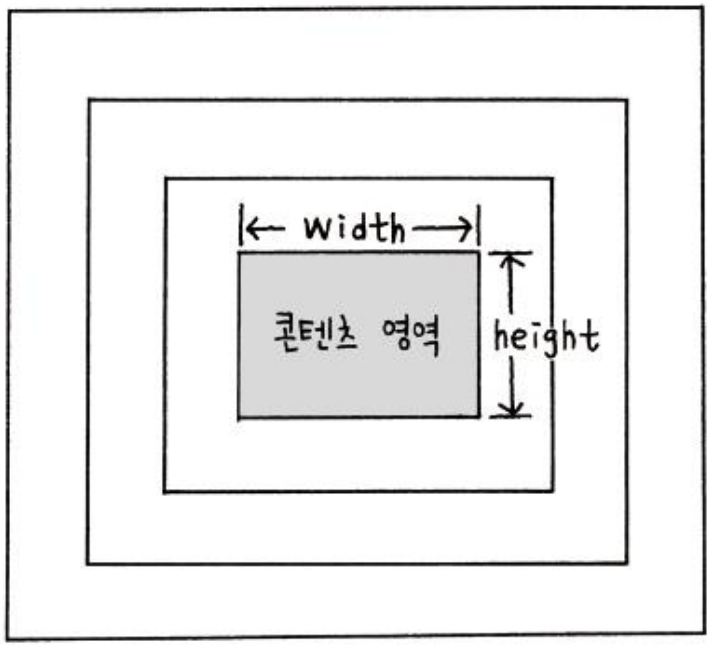
03-3. 레이아웃을 위한 스타일

- 박스 모델
- 레이아웃 구성하기

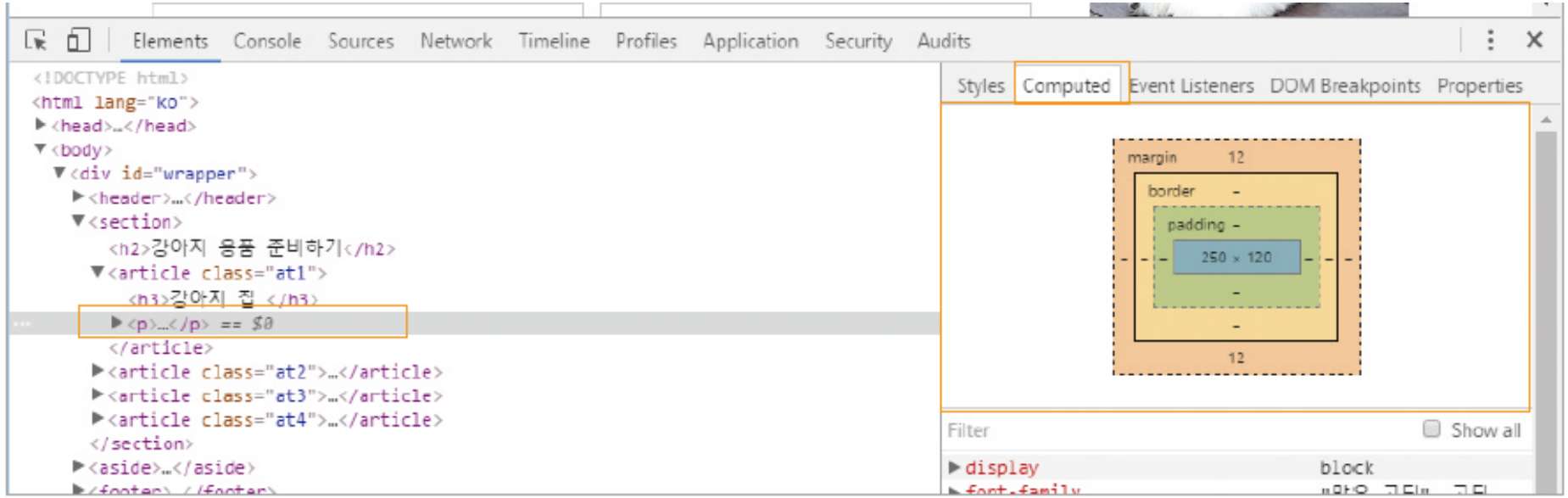
박스 모델

영역을 나눌 때는 주로 div태그를 이용

박스 모델



- **박스 모델**
  - 실제 콘텐츠 영역, 패딩(padding), 박스의 테두리(border), 그리고 마진(margin) 등의 요소로 구성됨
  - 개발자 도구 창에서 박스 모델 확인 가능
    - 크롬기준: F12



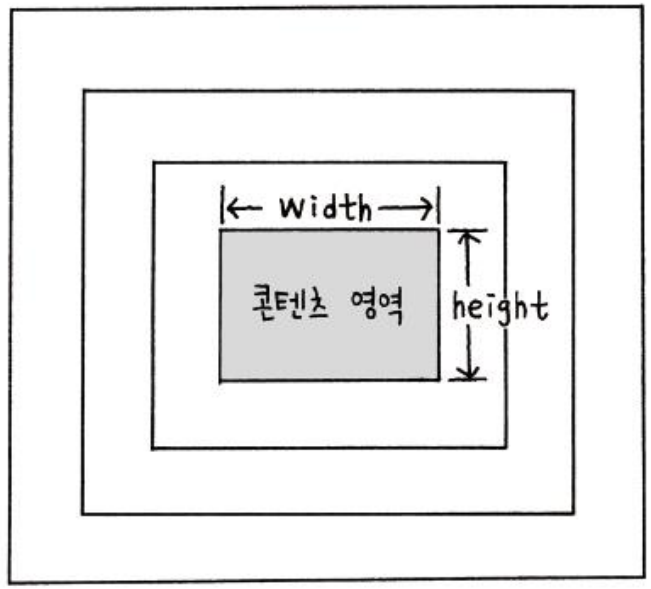
박스 모델

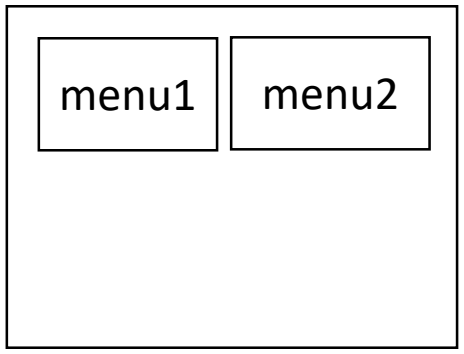
- **width, height 속성**
  - 실제 콘텐츠 영역의 크기 지정

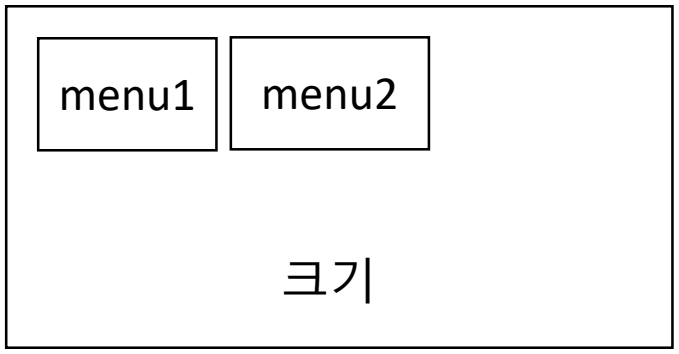
기본형

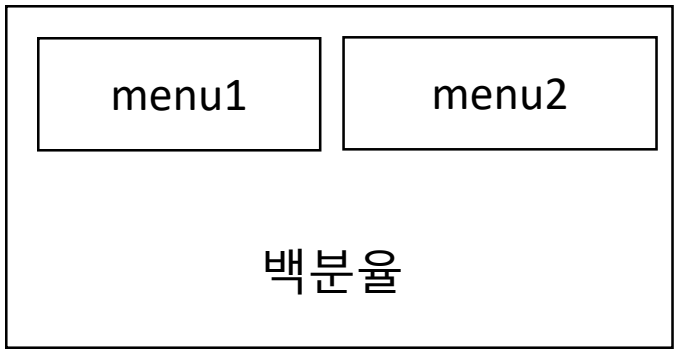
```
width: <크기> | <백분율> | auto
height: <크기> | <백분율> | auto
```

속성 값	설명
<크기>	너비나 높이 값을 px(픽셀)이나 cm(센티미터) 같은 단위와 함께 수치로 지정합니다.
<백분율>	박스 모델을 포함하는 부모 요소를 기준으로 너비나 높이 값을 백분율(%)로 지정합니다.
auto	박스 모델의 너비와 높이 값이 콘텐츠 양에 따라 자동으로 결정됩니다. 기본 값입니다.★



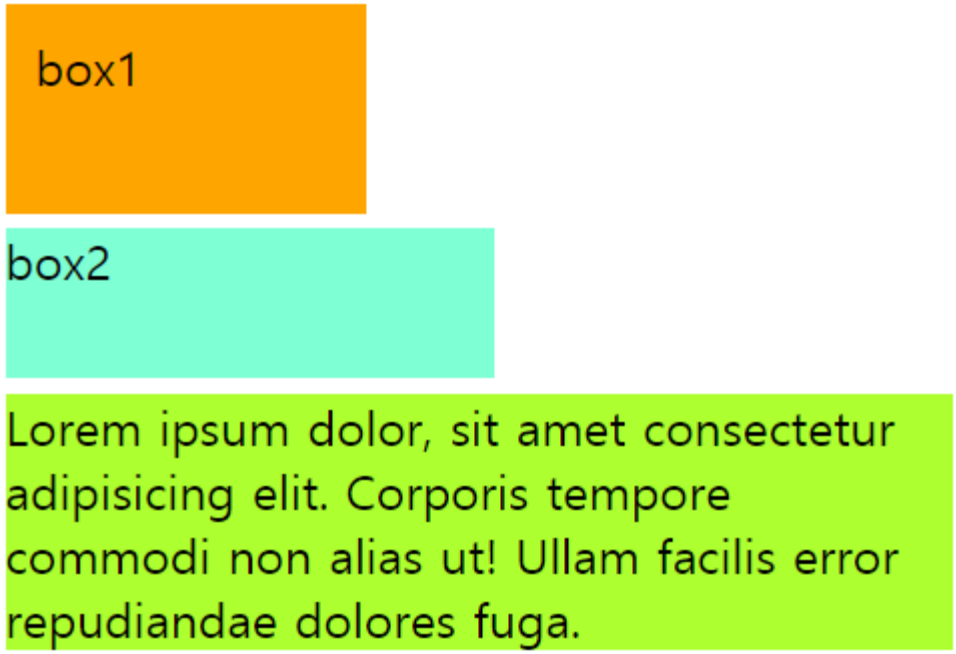






박스 모델

```
.box1 {
   width: 200px; /* 고정 너비 */
   height: 100px; /* 높이 */
   padding: 20px;
   background:orange; /* 배경색 */
 }
.box2 {
   width: 50%;    /* 브라우저 창 너비의 50% */
   height: 100px; /* 높이 */
   background:aquamarine; /* 배경색 */
}
.box3 {
   width: auto;height: auto;
   background:greenyellow; /* 배경색 */
}
```

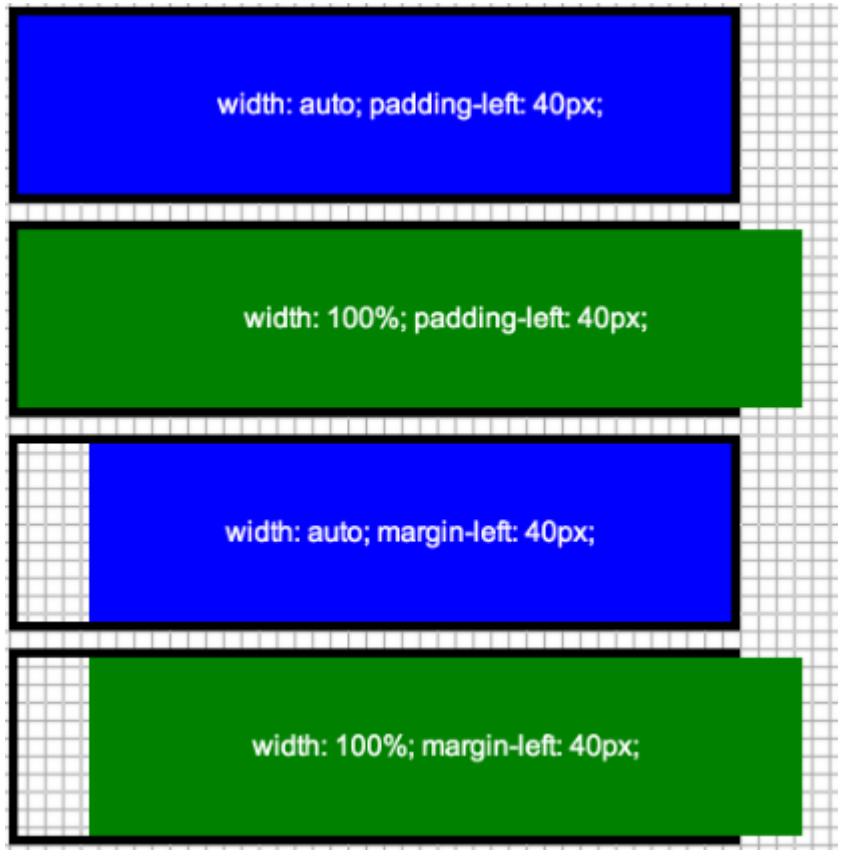



```
<body>
   <div class="box1">box1</div>
   <div class="box2">box2</div>
   <div class="box2">lorem20</div>
</body>
```

https://www.w3schools.com/cssref/tryit.asp?filename=trycss_dim_width

박스 모델

- Width auto VS Width 100%
  - Width auto
    - 블록레벨요소의 초기 width는 auto로 지정됨
    - 마진, 패딩, 경계 영역에 추가적인 공간이 설정되더라도 부모요소 안에 자식요소가 들어감
  - Width 100%
    - 블록레벨요소의 초기 width는 auto로 지정됨
    - 마진, 패딩, 경계 영역에 추가적인 공간이 설정되면 부모요소를 벗어날 수 있음
  - Box-sizing: border-box
    - border-box로 설정할 경우 width를 100%로 설정하더라도 자식요소가 부모요소안에 들어감

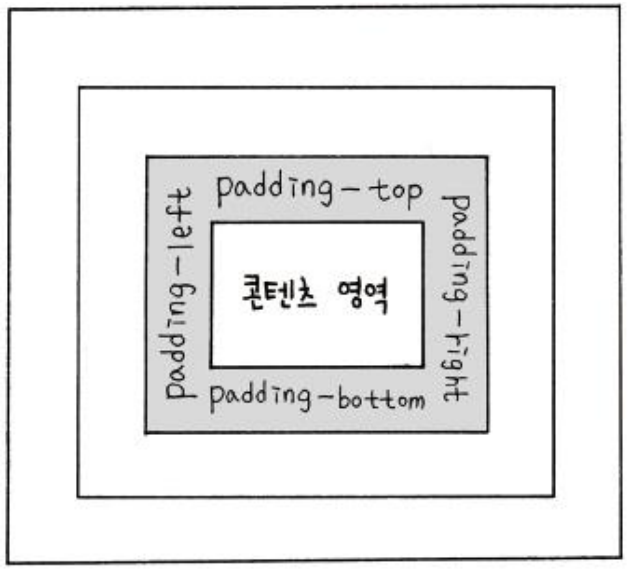


박스 모델

- **padding 속성**
  - 콘텐츠 영역과 테두리 사이의 여백(테두리 안쪽 여백)
  - 일반적으로 한 번에 padding지정

```
<style>
.box1 { padding:10px 30px 10px 30px;} /* 위,오,아,좌*/
.box2 { padding:10px 30px;} /* 위,아: 10, 좌우: 30*/
.box3 { padding:10px;} /* 모두 10*/
</style>
```

```
padding-top: <크기> | <백분율> | auto
padding-right: <크기> | <백분율> | auto
padding-bottom: <크기> | <백분율> | auto
padding-left: <크기> | <백분율> | auto
padding: <크기> | <백분율> | auto
```



박스 모델

- **border-style**
  - 기본 값이 none → 화면에 테두리 표시 안됨
  - 테두리를 그리기 위해서는 맨 먼저 테두리 스타일부터 지정
    - | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset | initial | inherit

```
<style>
.box1 { border-style:solid; }
.box2 { border-style:dotted; }
.box3 { border-style:dashed; }
</style>
```

박스 모델

- **border 속성**

  - 두께 지정

```
기본형   border-top-width: <크기> | thin | medium | thick
         border-right-width: <크기> | thin | medium | thick
         border-bottom-width: <크기> | thin | medium | thick
         border-left-width: <크기> | thin | medium | thick
         border-width: <크기> | thin | medium | thick
```

  - 색상 지정

```
기본형   border-top-color: <색상>
         border-right-color: <색상>
         border-bottom-color: <색상>
         border-left-color: <색상>
         border-color: <색상>
```

1) 1개라면 : 네 방향에 모두 같은 두께

```
.box1 { border-width:2px;}
```

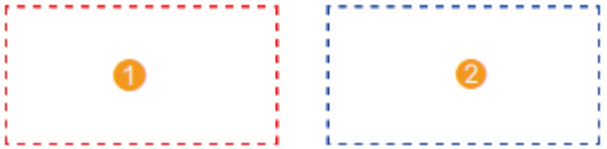
2) 2개라면 : 위아래, 좌우 묶어서

```
.box2 { border-width:thick thin; }
```

3) 4개라면 : top → right → bottom → left

```
.box3 { border-width:5px 10px 15px 20px; }
```

```
<style>
div {
  ……
  border-style:dashed;
  border-width:2px;
}
.box1 { border-color:red;}
.box2 { border-color:blue; }
</style>
```



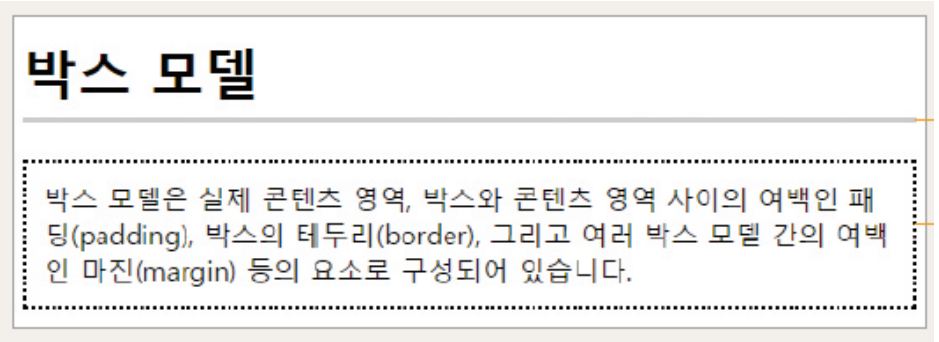
박스 모델

- **border 속성**
  - 테두리 스타일과 두께, 색상 등을 묶어 표기
  - 순서는 상관없음

기본형	border-top: <두께> | <색상> | <스타일>
	border-right: <두께> | <색상> | <스타일>
	border-bottom: <두께> | <색상> | <스타일>
	border-left: <두께> | <색상> | <스타일>
	border: <두께> | <색상> | <스타일>

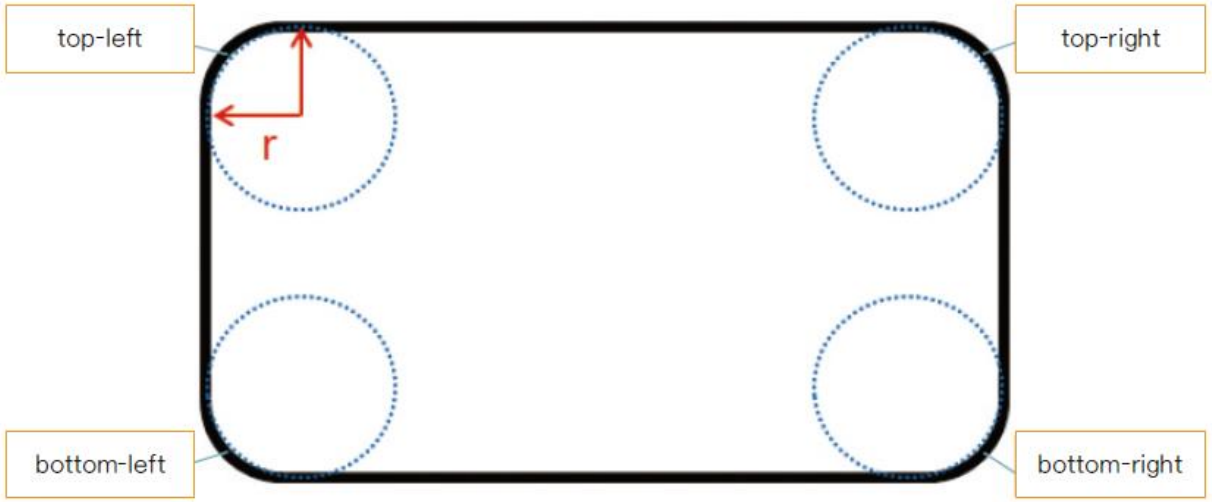
```
<style>
h1 {
  padding-bottom: 5px;
  border-bottom: 3px solid #ccc; /* 아랫 부분 - 3px짜리 회색
실선*/
}
p {
  padding: 10px;
  border: 2px dotted black; /* 모든 방향 - 3px 검정 점선 */
}
</style>

<h1>박스 모델</h1>
<p>박스 모델은 실제 콘텐츠 영역 …… 있습니다. </p>
```



박스 모델

- **border-radius 속성**
  - 박스 모서리 부분을 둥글게 처리



border-radius 속성과 반지름 값

```
<style>
.round {
  width:100px;
  height:50px;
  border:2px solid red; /* 2px짜리 빨강 실선 */
  border-radius:20px;  /* 모서리 20px 만큼 라운딩 */
}
#bg {
  width:100px;
  height:50px;
  background:url(images/pic1.jpg) no-repeat;
  background-size:cover;
}
</style>
```

`border-radius:50%;` → 원으로 만들겠다

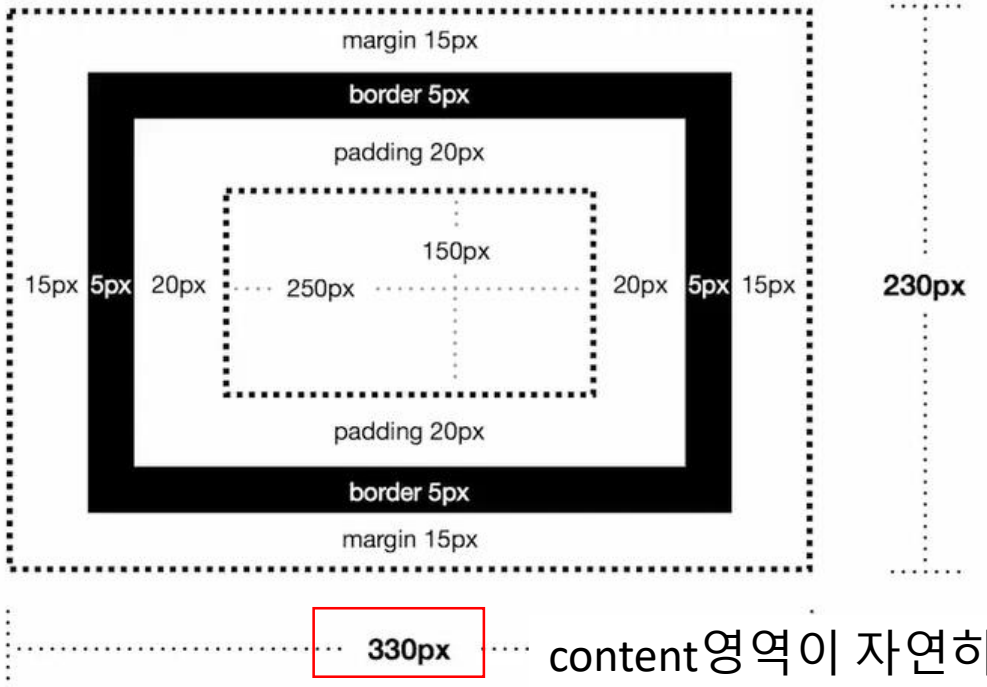
박스 모델

- **Box-sizing**
  - 내가 지정한 width는 어디까지 포함되는 것인가?

```
기본형   box-sizing: content-box | border-box
```

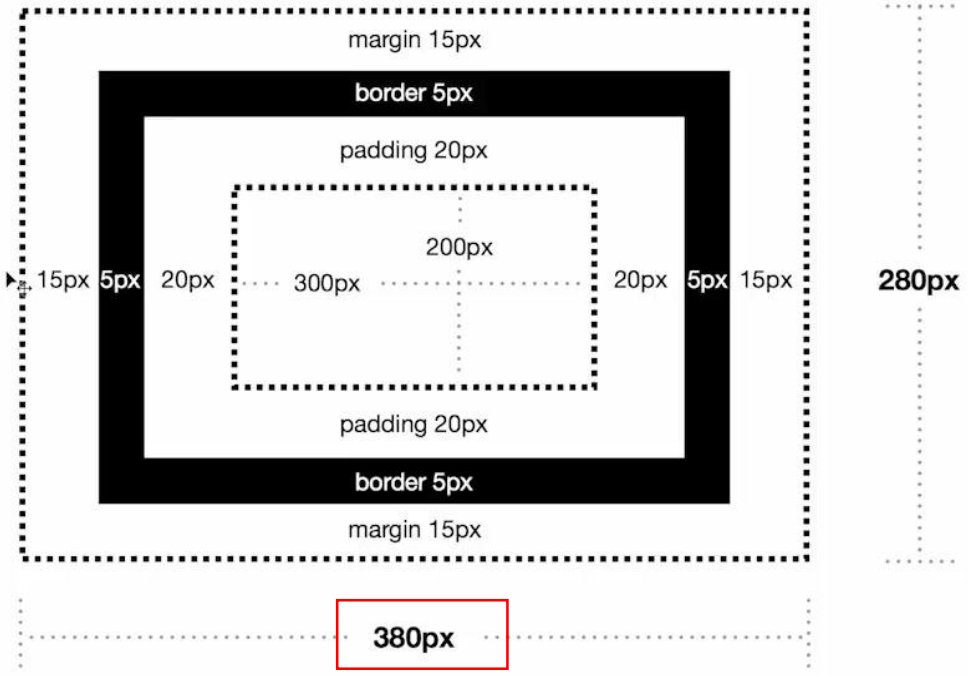
- content-box: width속성 값을 콘텐츠 영역 너비로
- border-box: width 속성 값을 테두리까지 포함

width: 300; height:200px; padding:20px; border: 5px solid black; margin: 15px



content영역이 자연히 축소

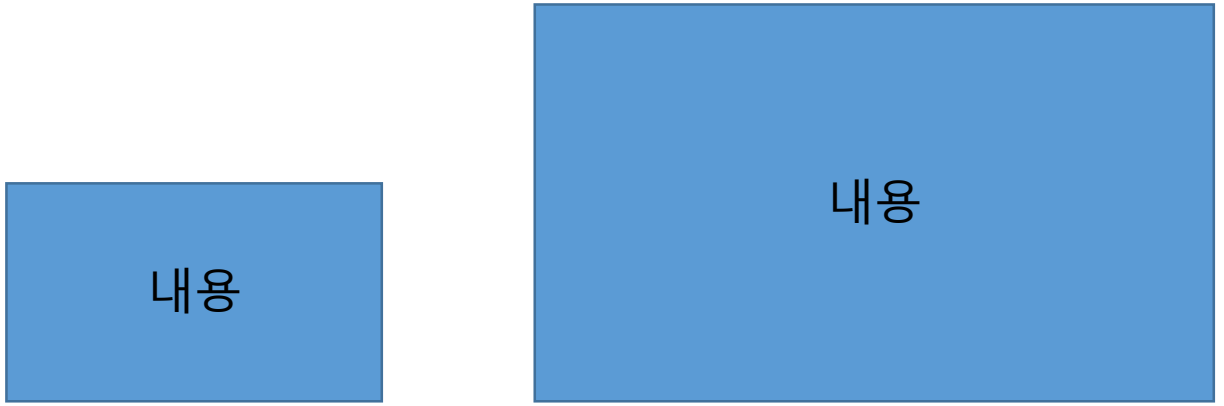
box-sizing: border-box (border까지 width에 포함)



box-sizing: content-box (기본값)

박스 모델

- **크기를 지정하는 단위의 고려사항**
  - 반응형 웹, 시스템 유지 보수를 위해선 em혹은 rem을 사용해야 함
    - padding을 지정할 경우에도 px보다는 em 혹은 rem
    - 대체로 글씨 크기는 rem으로, padding은 em으로 지정 (주변과의 어울림을 위해)
    - html의 기본 font크기는 16px
    - 16=1em



여백은 자신을 포함하는 바로 위의
부모와의 어울림이 필요

레이아웃 구성하기

레이아웃

- **CSS 포지셔닝**
  - CSS를 이용하여 각 요소들을 웹 문서에 적절히 배치

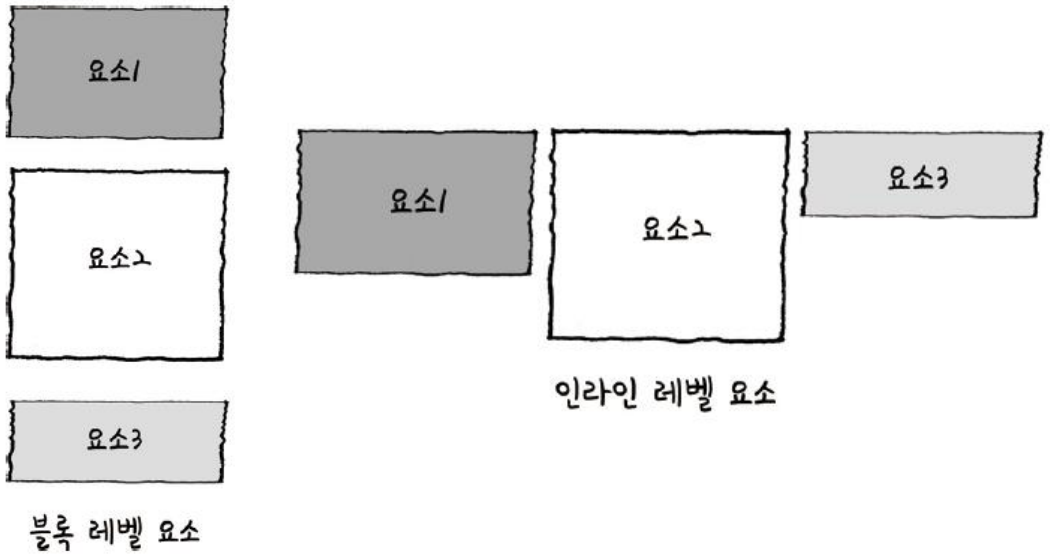
HTML 마크업으로 작성한 웹 문서

CSS
포지셔닝

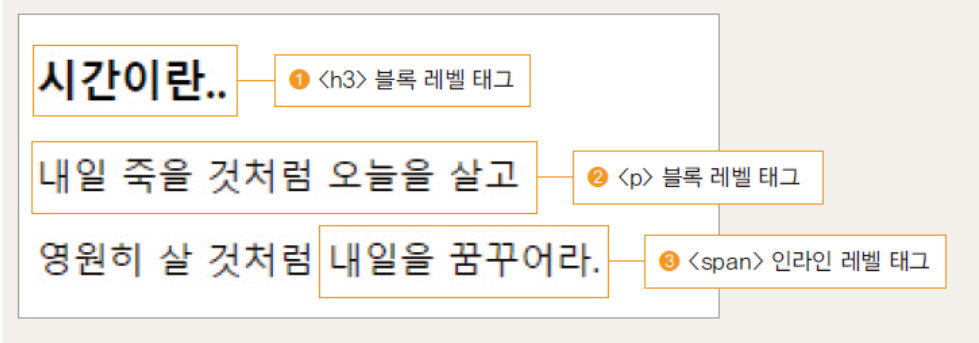
우리가 보는 웹 사이트

레이아웃

- **블록 요소**
  - 요소를 삽입했을 때 혼자 한 줄을 차지하는 요소
  - 요소의 너비가 100%
  - 예) , <p> 등
- **인라인 요소**
  - 줄을 차지하지 않는 요소
  - 화면에 표시되는 콘텐츠만큼만 영역을 차지하고 나머지 공간에는 다른 요소가 올 수 있음
  - 예) <img>, 등




```
<h3>시간이란..</h3>
<p>내일 죽을 것처럼 오늘을 살고</p>
<p>영원히 살 것처럼 <span>내일을 꿈꾸어라.</span></p>
```



레이아웃

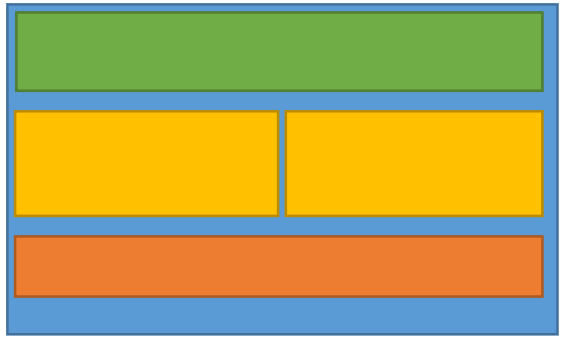
- **display 속성**
  - 블록 레벨 요소를 인라인 레벨 요소로 바꾸거나 인라인 레벨 요소를 블록 레벨 요소로 바꿈

```
기본형   display: none | contents | block | inline | inline-block | table | table-cell 등
```

  - display:block -> 해당 요소를 블록 레벨로 지정
  - display:inline → 블록 레벨 요소를 인라인 레벨로 지정
  - display:inline-block → 요소를 인라인 레벨로 배치하면서 내용에는 블록 레벨 속성을 지정(**너비**나 **높이**)

  - 최근에는 flex를 이용하여 positioning을 유연하게 할 수 있음

레이아웃



```
.block {
    display: block;
    width: 200px;
    height: 100px;
    background-color: indianred;
    margin: 10px;
}
```

box
box
box

```
.inline {
    display: inline;
    width: 200px;
    height: 100px;
    background-color:yellowgreen;
    margin: 10px;
}
```

box box box

```
.inline-block {
    display: inline-block;
    width: 200px;
    height: 100px;
    background-color: aquamarine;
    margin: 10px;
}
```

box box box

box를 옆으로 붙일 수 있지만
내가 지정한 높이와 넓이 적용 안됨

레이아웃

4) display:none https://www.w3schools.com/w3css/tryit.asp?filename=tryw3css_sidebar_over

해당 요소를 화면에 표시하지 않음

화면에서 공간도 차지하지 않음

→화면에 보였다 사라졌다 할 때 많이 사용

⇔**visibility:hidden**

공간은 차지하지만 요소가 보이지는 않음

레이아웃

- inline-block + ul을 이용한 메뉴 만들기

```
<ul>
    <li>FreeBoard</li>
    <li>Archive</li>
    <li>전공지식</li>
</ul>

ul>li{
    display:inline-block;
    width: 250px;
    height: 80px;
    line-height: 80px;
    background-color: black;
    border-right:3px solid red;
    text-align: center;
    color:#fff;
}
```

레이아웃

- inline-block을 사용할 때 주의할 점(feat. 원하지 않는 공백)


```
<ul>
    <li>FreeBoard</li>
    <li>Archive</li>
    <li>전공지식</li>
</ul>
```

코드의 가독성을 위해 입력한 엔터가 하나의 입력으로 작용

코드의 가독성을 위해 입력한 엔터가 하나의 입력으로 작용

- inline-block의 공백 없애기

1. enter입력 없애기

```
<ul>
    <li>FreeBoard</li><li>Archive</li><li>전공지식</li>
</ul>
```

2. 오른쪽 마진 음수 값 적절하게 입력

```
margin-right: -10px;
```

레이아웃

- inline-block의 공백 없애기

3. ul의 크기 조절

```
ul{
    font-size:0px;
}
ul>li{
    font-size:1rem;
}
```

ul에서 지정한 글자 크기가 상속되므로
li 반드시 크기를 재 정의(단 em은 안됨)

4. li를 float left로 지정(다음 장에서 float을 알아 봄)

```
ul{
      list-style-type: none;
}
ul>li{
      float: left;
}
```

레이아웃

https://www.w3schools.com/cssref/tryit.asp?filename=trycss_float

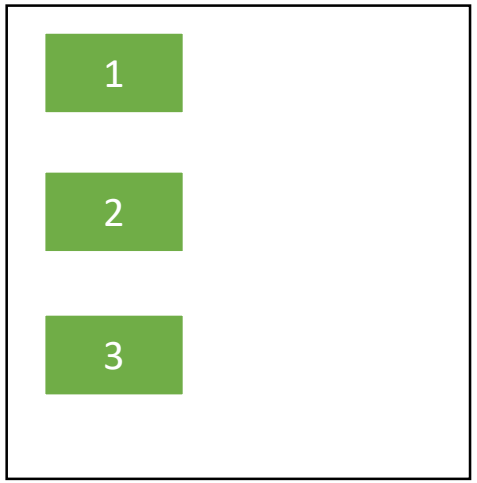
- **float 속성**
  - 요소를 왼쪽이나 오른쪽에 떠 있게 만듦
  - **이미지**를 어떻게 띄워서 텍스트와 조화롭게 배치할지를 결정
  - float: left | right

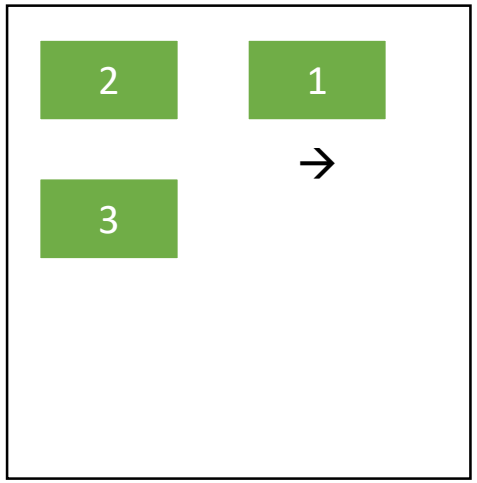
```
<style>
    .float-container{
        width: 400px;
        border: 2px solid #09c;
    }
    .float-container img{
        float: left;
        margin: 5px;
        padding: 5px;
        border: 2px solid #90C;
    }
</style>
```


```
<div class="float-container">
    <img src="img/air.jpg">
    lorem100
</div>
```

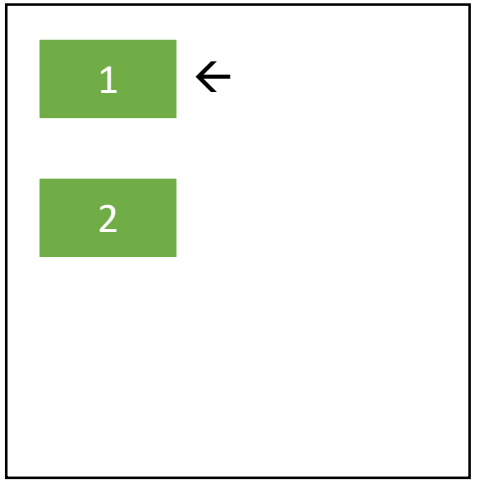
레이아웃

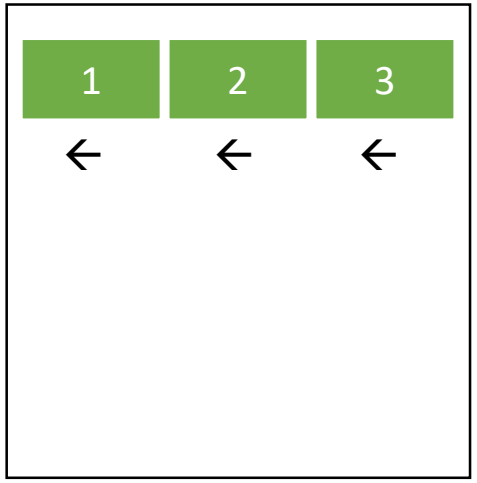
- **레이아웃에서 사용되는 float 속성**

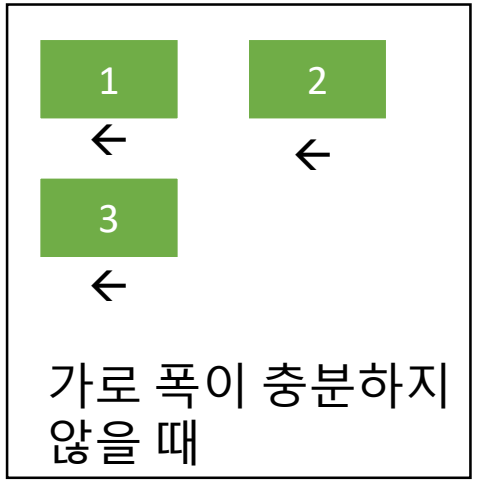


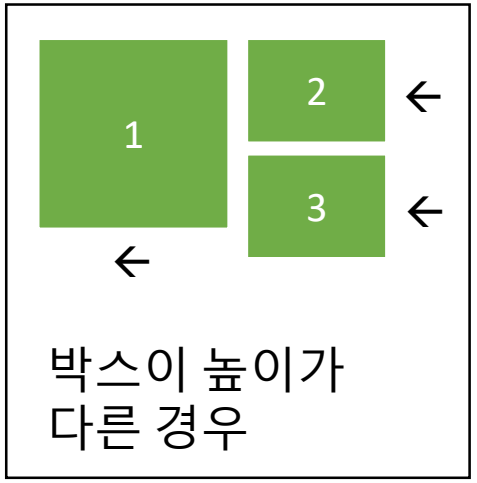


1을 왼쪽으로 float









레이아웃

- **clear 속성**
  - float 속성을 무효화 시키는 속성
  - 플로팅한 요소는 문서의 흐름에서 벗어난 상태이므로 레이아웃을 무너뜨리게 됨
  - float:left | float:right | float:both 중 하나를 이용
  - 플로팅을 이용하여 레이아웃을 구성할 때, 이전 요소에서 적용된 float속성이 다음 속성으로 흘러가지 않도록 차단하는 방법

```
선택자::after{
   content:"";
   display:block;
   clear:both;
}
```

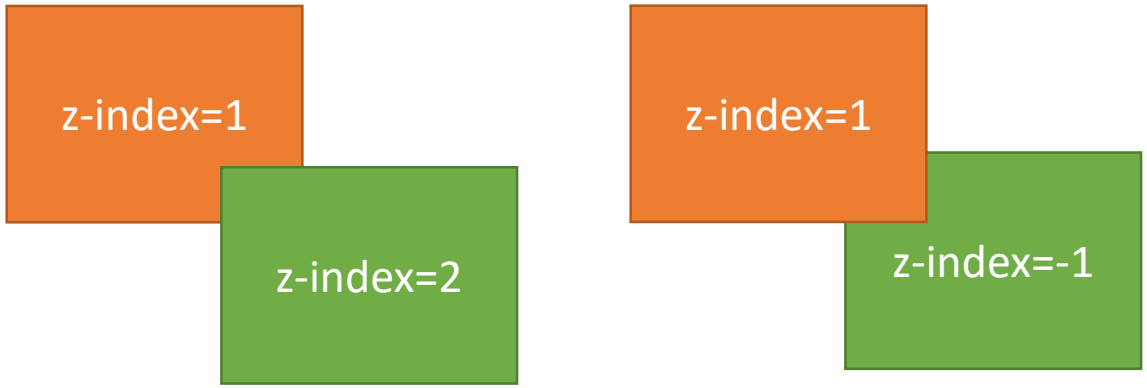
선택되는 요소::after→float속성이 더 이상 적용되지 않기를 바라는 지점

https://www.w3schools.com/css/css_templates.asp

1) 반드시 들어가서 확인해보고
2) 참조코드를 보지 않고 스스로 작성해보기!

레이아웃

- **Z-Index**
  - z축을 기준으로 어떤 요소가 더 위에 올 것인지를 결정



  - position 속성이 설정된 요소에 대해서만 적용(relative, absolute, fixed 중 하나)
  - 정수 가능(양수/음수)
  - https://www.w3schools.com/cssref/tryit.asp?filename=trycss_zindex

레이아웃

- **position 속성**
  - 웹 문서 안에 요소들을 배치하기 위한 속성
  - static, relative, absolute, fixed
  - position은 offset(top, right, bottom, left)을 이용하여 떨어진 정도를 설정할 수 있으며 각 속성의 기준이 어디인지를 아는 것이 중요
    - static
      - 기본값(특별히 위치를 변경하지 않고 원래 나타나야 하는 위치→normal-flow)
      - top, right, bottom, left를 지정해도 반영되지 않음
    - relative
      - offset을 지정하여 기준 위치에서 떨어진 곳으로 이동
      - 기준위치란?: 다른 요소의 위치를 신경 쓰지 않고 원래 나타나야 하는 그 위치
      - relative를 지정하더라도 offset을 지정하지 않으면 static과 동일

레이아웃

```
body{
    border: 1px solid black;
}
div {width:200px;height:200px;}
div.box1 {
    background-color:red;
    position: relative;
    top:250px;
}
div.box2 {
    background-color:yellow;
}
```

```
<div class="box1">testdiv1</div>
<div class="box2">testdiv2</div>
```

position속성이 absolute인 경우
z-index 기본값은 1

relative → 공간을 차지하고 이동한다

레이아웃

- **position 속성**
  - absolute
    - 가장 가까운 곳에 위치한 조상(position이 static이 아닌) 요소가 기준
    - 그러한 기준 부모가 없으면 body태그가 기준이 됨
    - position이 static이 아니다 ⇔ position이 relative, absolute, fixed인 부모
    - 일반적으로 부모 요소는 relative, 자식 요소 absolute로 설정

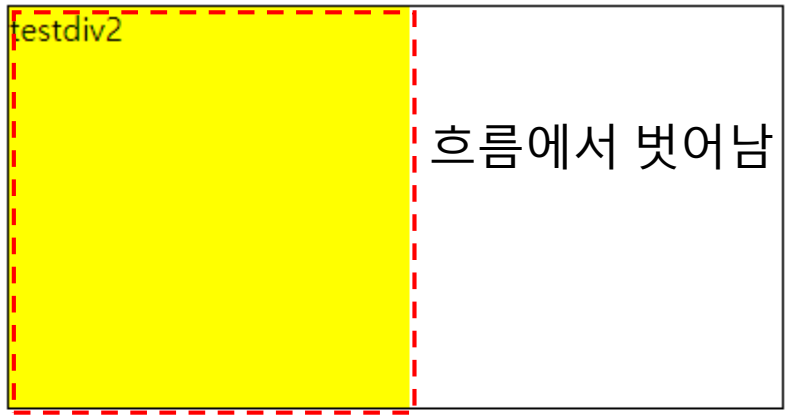
레이아웃

```
body{
    border: 1px solid black;
}
div {width:200px;height:200px;}
div.box1 {
    background-color:red;
    position: absolute;
    top:250px;
}
div.box2 {
    background-color:yellow;
}
```

```
<div class="box1">testdiv1</div>
<div class="box2">testdiv2</div>
```

absolute → 붕 떠있는 느낌

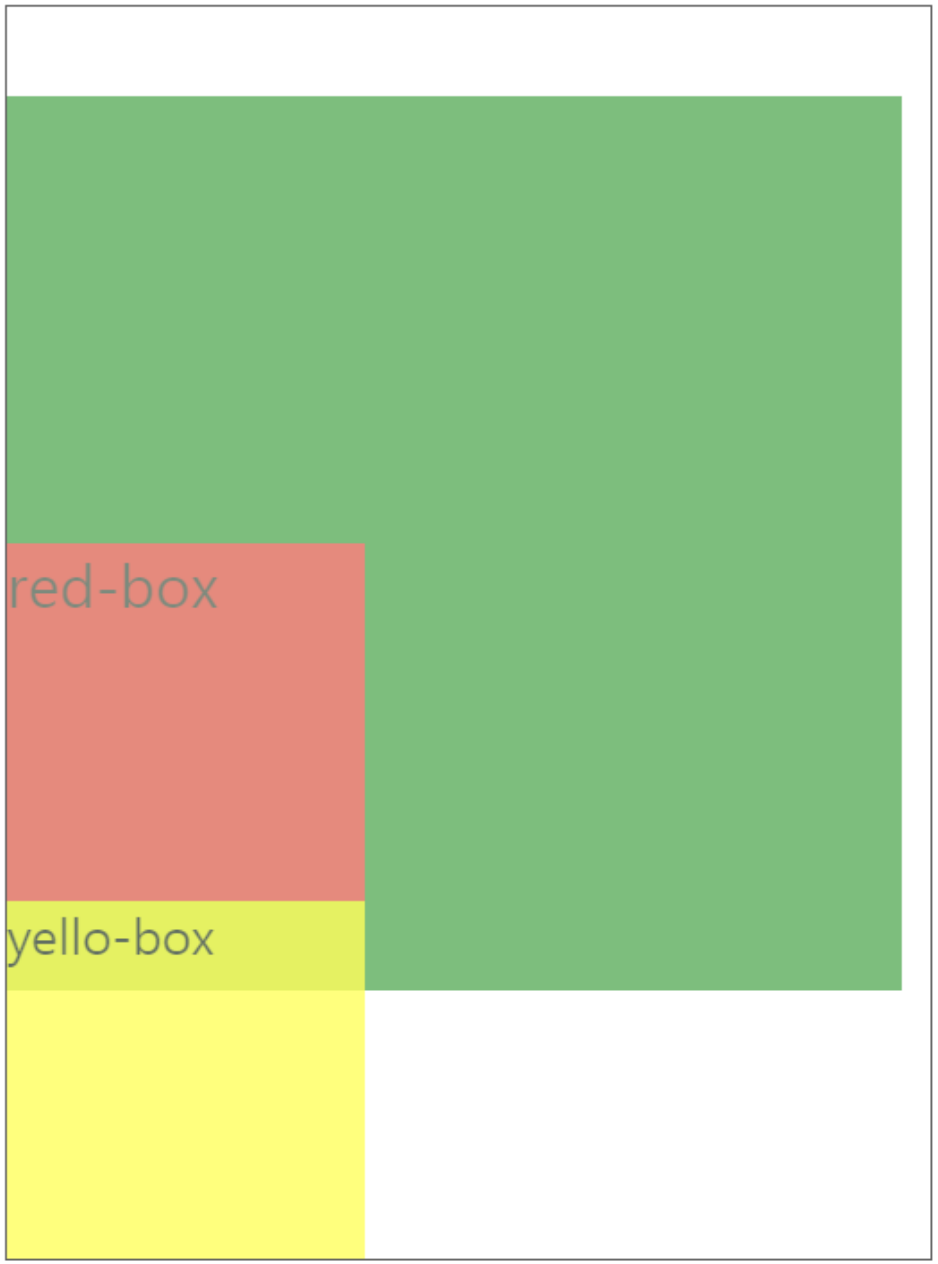
붉은색 박스의 원래 위치



(부모요소 중 position이 static이 아닌 요소가 없으므로 기준은 body가 됨)

```
        body{
            border: 1px solid black;
        }
        div {width:200px;height:200px;}
        div.box1 {
            background-color:red;
            position: absolute;
            top:250px;
        }
        div.box2 {
            background-color:yellow;
        }
        .container{
            width:500px;height:500px;
            background-color:green;
            position:relative;
            top:50px
        }
```

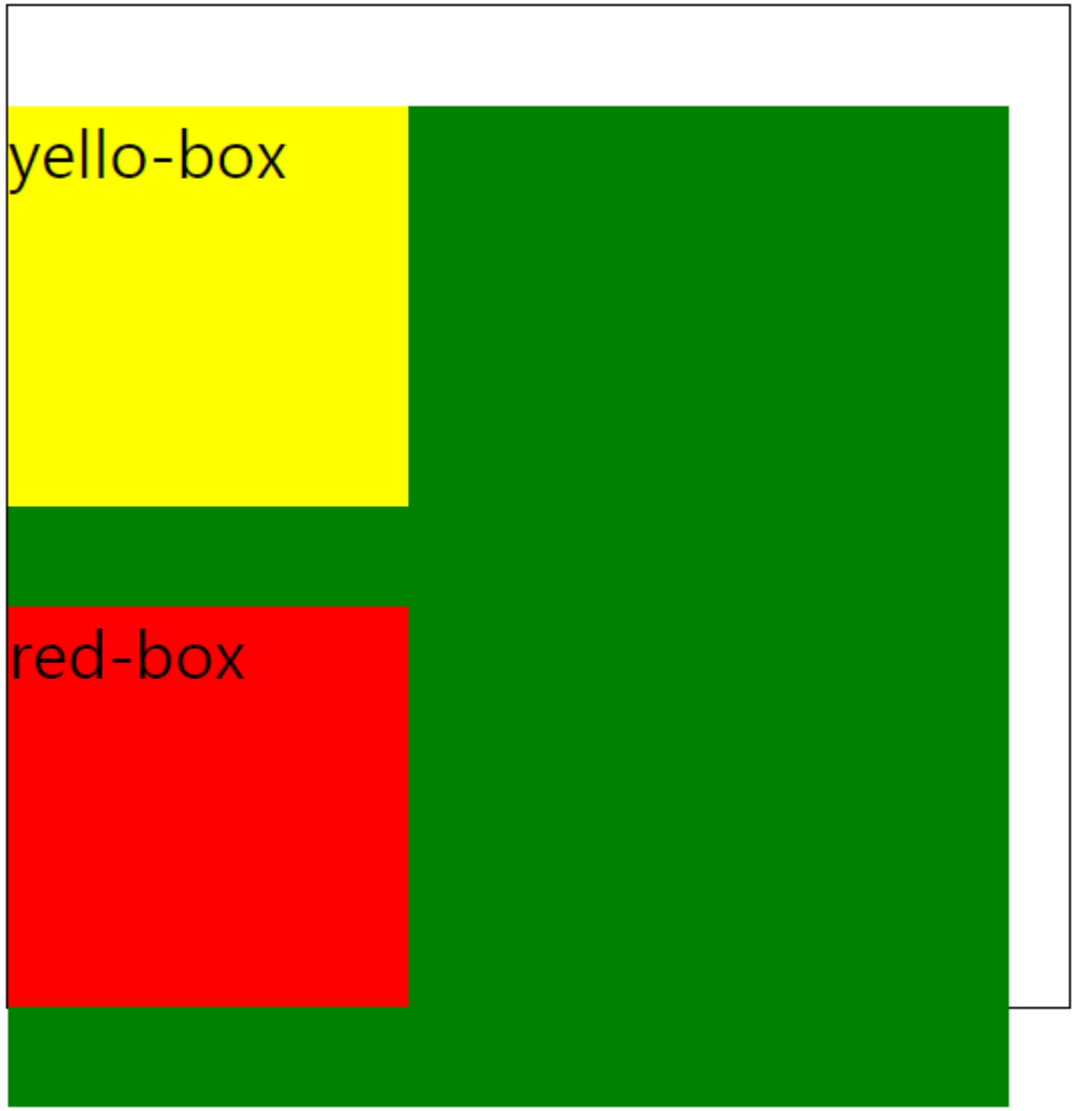
```
<div class="container">
        <div class="box1">red-box</div>
</div>
<div class="box2">yello-box</div>
```



위 예제와 동일한 css

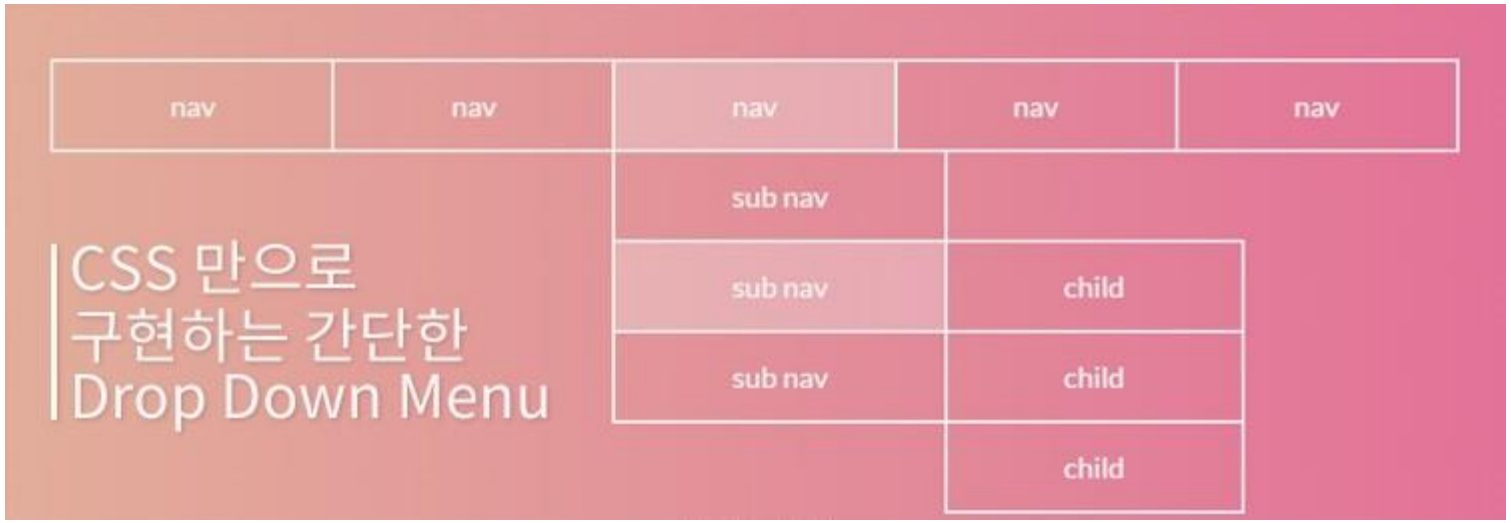
```
body{
    border: 1px solid black;
}
div {width:200px;height:200px;}
div.box1 {
    background-color:red;
    position: absolute;
    top:250px;
}
div.box2 {
    background-color:yellow;
}
.container{
    width:500px;height:500px;
    background-color:green;
    position:relative;
    top:50px
}
```

```
<div class="container">
    <div class="box1">red-box</div>
    <div class="box2">yello-box</div>
</div>
```



레이아웃

- **position 속성**
  - relative(부모)-absolute(자식) 관계는 1) 부모요소를 기준으로 자식요소의 위치를 결정하거나 2) 자식이 공간을 차지하지 않고 떠있게 하고 싶을 경우 유용



  - http://rwdb.kr/cssdmenu/
  - dropdown메뉴는 il를 inline-block으로 처리할 수도 있고 float left로 처리할 수도 있음
  - dropdown메뉴는 ul > li >ul..형태 혹은 ul > li >div..형태 등으로 정의할 수 있으며 유튜브, 웹 상에 나온 다양한 자료를 참고할 것

레이아웃

- **position 속성**
  - fixed
    - normal-flow랑 관계 없이(다른 어떤 요소와도 상관 없이) 브라우저의 0,0(좌측 상단)을 기준으로 offset 결정
    - navigation bar, sidebar, footer와 같이 스크롤, 창의 크기 등에 관계 없이 항상 고정된 위치에 요소를 배치하고 싶은 경우 사용
    - 공간을 차지하지 않고 떠 있음
    - https://www.w3schools.com/howto/tryit.asp?filename=tryhow_js_sidenav

레이아웃

- **margin 속성**
  - 현재 요소 주변의 여백
  - 마진을 이용하면 요소와 요소 간의 간격 조절 가능

기본형

```
margin-top: <크기> | <백분율> | auto
margin-right: <크기> | <백분율> | auto
margin-bottom: <크기> | <백분율> | auto
margin-left: <크기> | <백분율> | auto
margin: <크기> | <백분율> | auto
```

속성 값	설명
<크기>	너비나 높이 값을 px(픽셀)이나 cm(센티미터) 같은 단위와 함께 수치로 지정합니다. 예 margin:10px;
<백분율>	박스 모델을 포함하고 있는 부모 요소를 기준으로 너비나 높이 값을 %로 지정합니다. 예 margin:0.1%;
auto	display 속성에서 지정한 값에 맞게 적절한 값을 자동으로 지정합니다.

창을 가운데로 두고 싶을 때 margin:100px auto

레이아웃

빠진 값은 마주 보는 방향의 속성 값 사용
속성의 적용방향은 위(top)에서부터 시계방향

- 속성 값이 1개라면
  - 네 방향 모두에 같은 값 적용

```
p { margin: 50px;}  /* 네 방향 마진 모두 50px */
```

- 속성 값이 2개라면
  - 첫번째 값은 위아래, 두번째 값은 좌우 마진 값

```
p { margin: 30px 50px;}  /* 위아래 마진 - 30px, 좌우 마진 - 50px */
```

- 속성 값이 3개라면

```
p { margin: 30px 20px 50px;}  /* 위 마진 - 30px, 좌우 마진 - 20px, 아래 마진 - 50px */
```

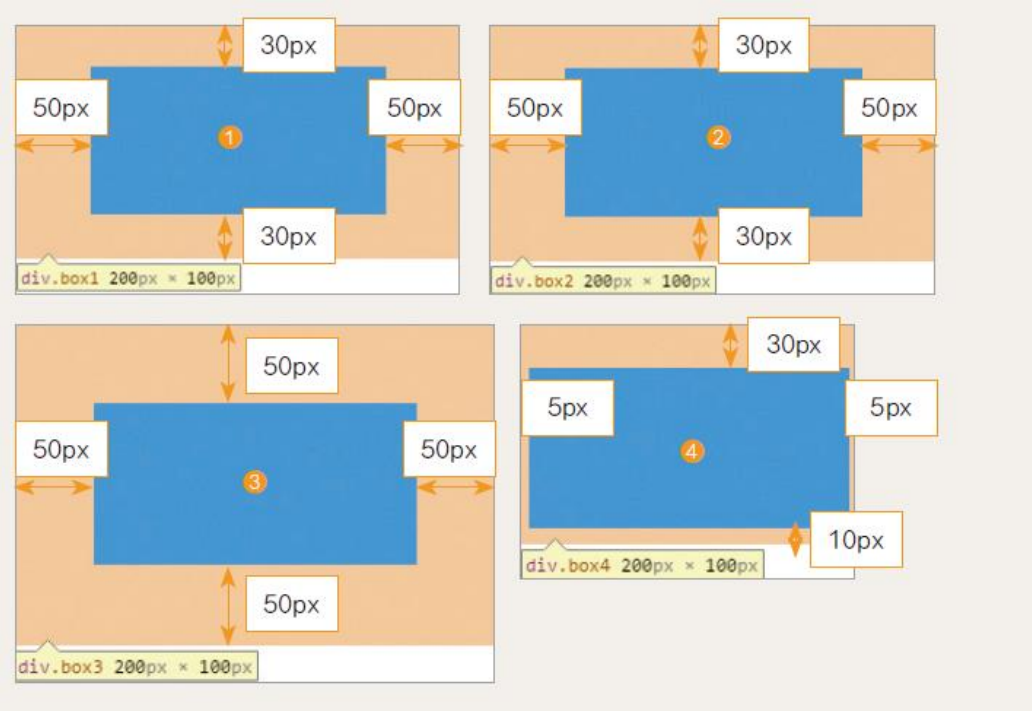
- 속성 값이 1개라면
  - top, right, bottom, left 순으로 적용

```
p { margin:30px 50px 30px 50px;} /* 위아래 마진 - 30px, 좌우 마진 - 50px */
```

레이아웃

- **margin 예**

```
<style>
.box1 { margin:30px 50px 30px 50px;}
.box2 { margin:30px 50px;}
.box3 { margin:50px;}
.box4 { margin:30px 5px 10px; }
</style>
```




```
<style>
.box {
  width:200px;  /* 너비 */
  height:300px;  /* 높이 */
  background:#ff6a00;  /* 배경색 */
  margin:0 auto; /* 마진 - 0 auto 0 auto */
}
</style>
```

min-width and max-width

화면의 크기가 늘어나거나 줄어들더라도
최대/최소로 보장되는 너비의 크기 지정

레이아웃

- **추가적으로 알아야 할 내용**
  - Table→Float→Flex→Grid
  - CSS Flexbox(2009년에 처음 제안) →bootstrap
  - CSS Grid(2017년 크롬, 파이어폭스, 오페라, 사파리에 탑재)
  - CSS preprocessor: less, sass