



# Lay-out property

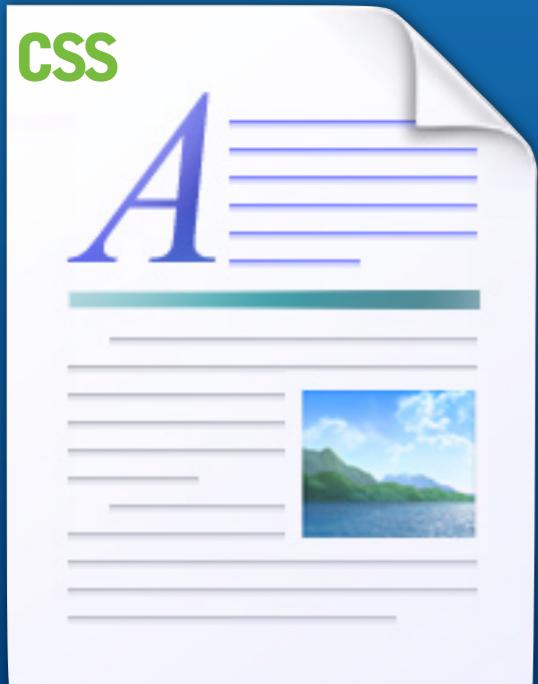
CSS에는 레이아웃 제어가 가능한 몇가지 속성이 제공됩니다.

# CSS 레이아웃 이해/활용의 시작은 화면에서 어떻게 표시되는지를 아는 것입니다!





CSS언어를 이용하여  
대상의 화면표시형태를 바꿔 볼까요?



Style code

대상 속성 값  
selector {display: value}

<head>



<style> strong {display: block} </style>  
</head>



# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



none

Style code

대상 속성 값  
selector {display: value}

<head>



<style> strong {display: block} </style>  
</head>



# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



```
<head>      ▲  
    <style> strong {display: block} </style>  
</head>
```



# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



```
<head> ▲  
  <style> strong {display: block} </style>  
</head>
```



# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



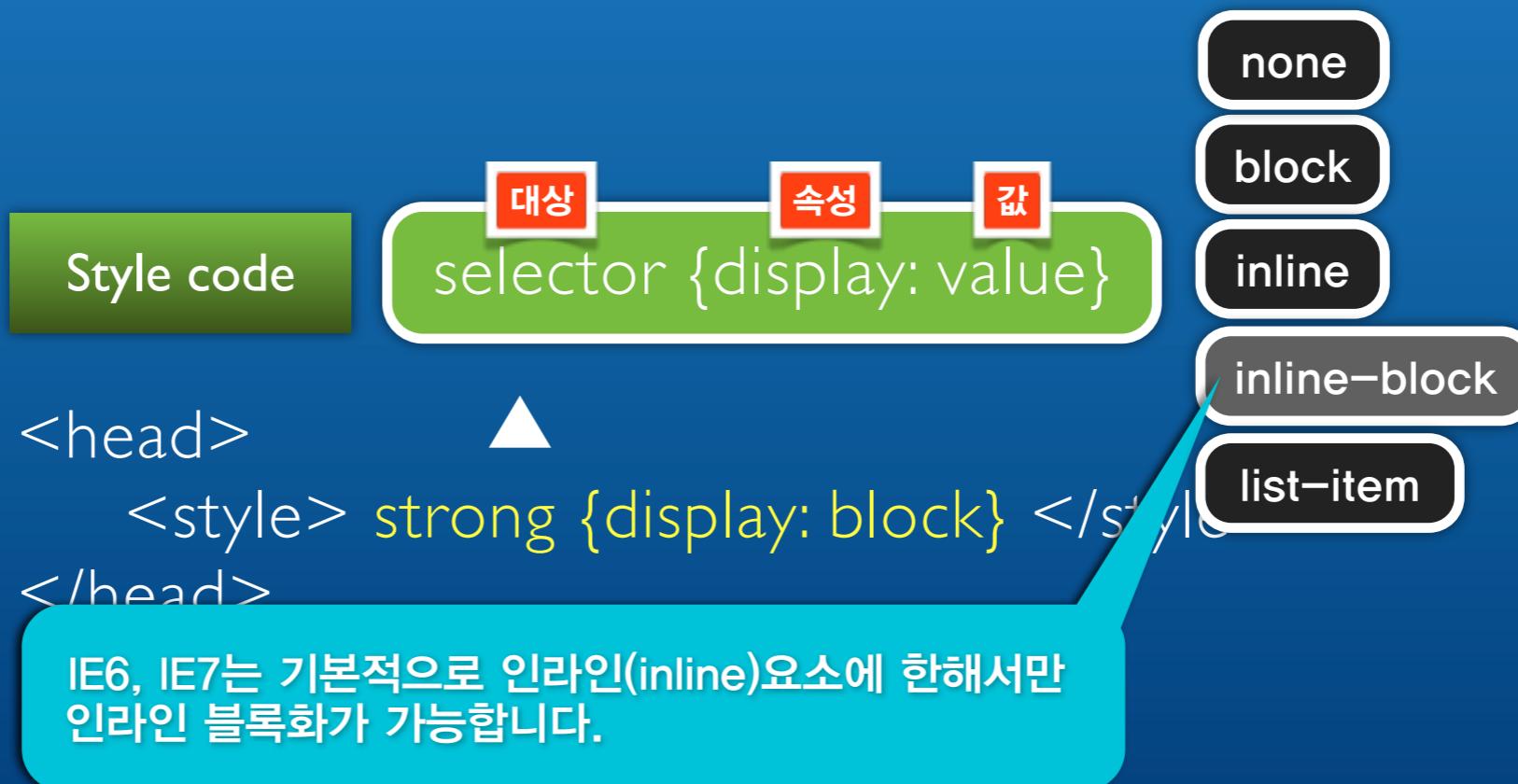
```
<head>
  <style> strong {display: block} </style>
</head>
```

IE6, IE7는 기본적으로 인라인(display: inline) 요소에 대해서만  
블록화가 가능합니다.





# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



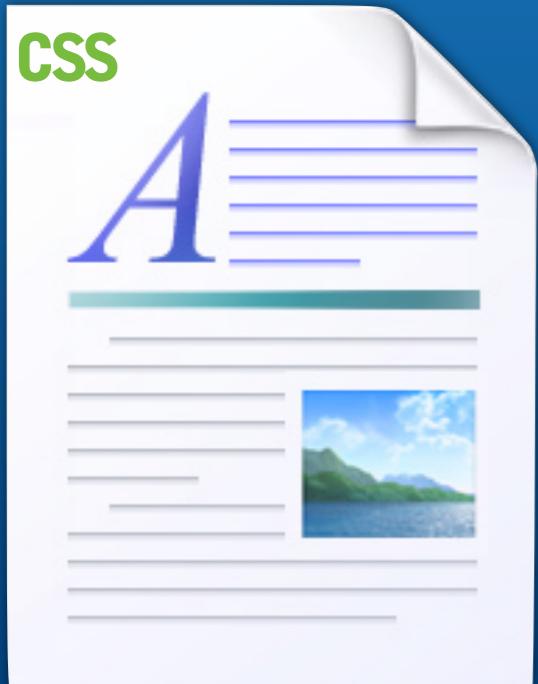


# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?





# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



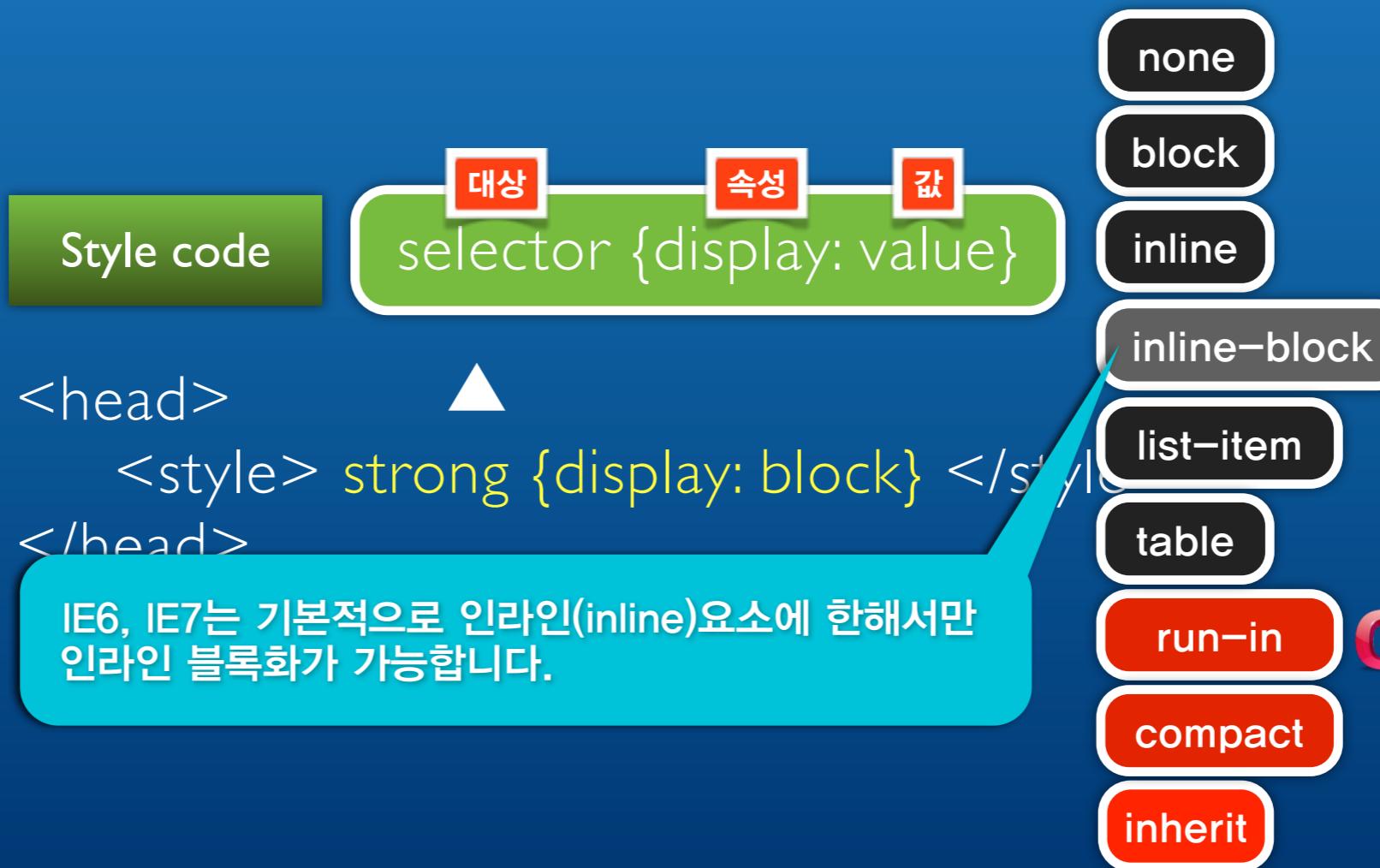


# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?





# CSS언어를 이용하여 대상의 화면표시형태를 바꿔 볼까요?



```
first {display: block}
```

```
second {display: block}
```

```
third {display: block}
```

## block

블록 요소처럼 표현하고자 할 경우 사용.

```
display: block
```

```
display: inline
```

```
display: block
```

```
display: block
```

```
display: block
```

```
display: inline
```

## inline

인라인 요소처럼 표현하고자 할 경우 사용.

```
display: block
```

```
display: block
```

## none

화면에 표시하지 않을 경우 사용.

```
display: block
```

Let's add some content to see how the block

```
display: inline-block;
```

```
width: 10em
```

```
Let's add some
```

```
content to see how the  
block behaves.
```

Let's add

some content to see how the block behaves. Let's add some content to see how the block behaves.

```
span (and not div)
```

```
with display: inline-
```

```
block; width: 10em
```

```
Let's add some
```

```
content to see how the  
block behaves.
```

Let's add some content to  
see how the block behaves.

## list-item

목록 요소처럼 표현하고자 할 경우 사용.

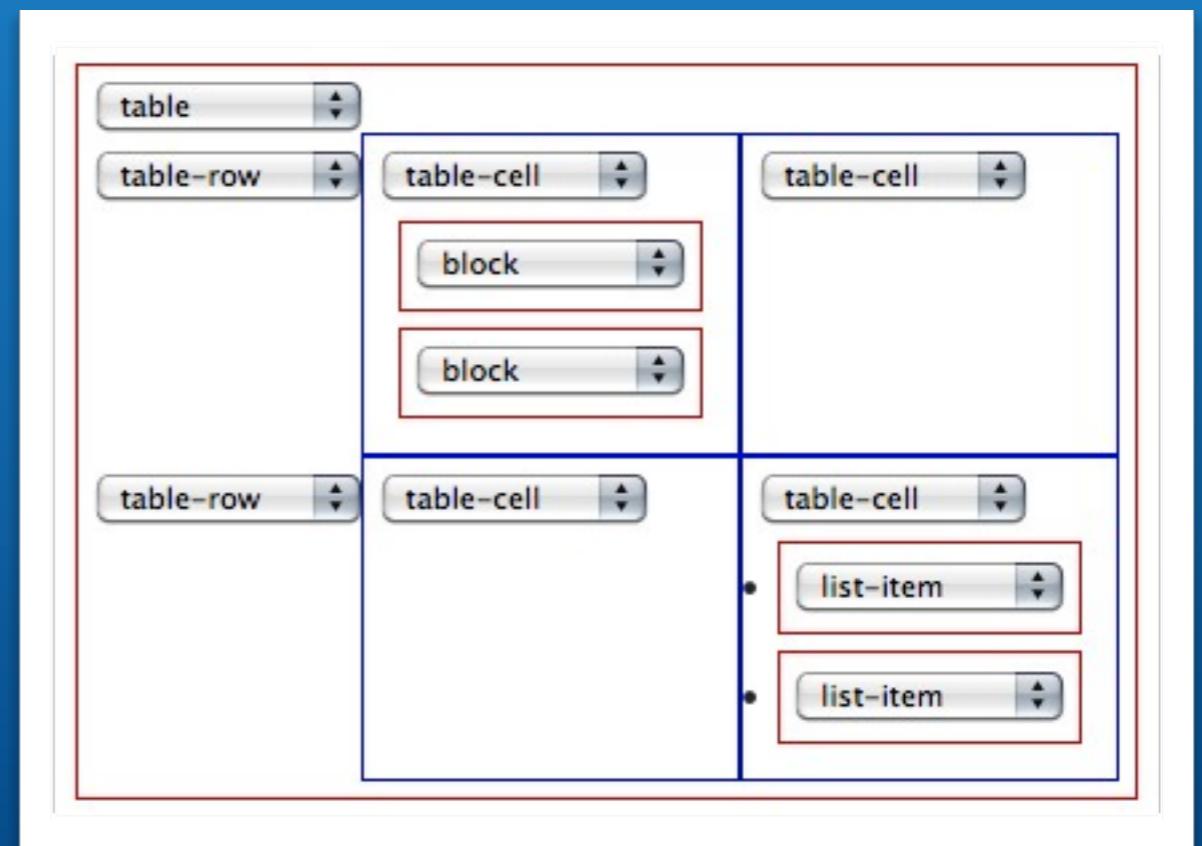
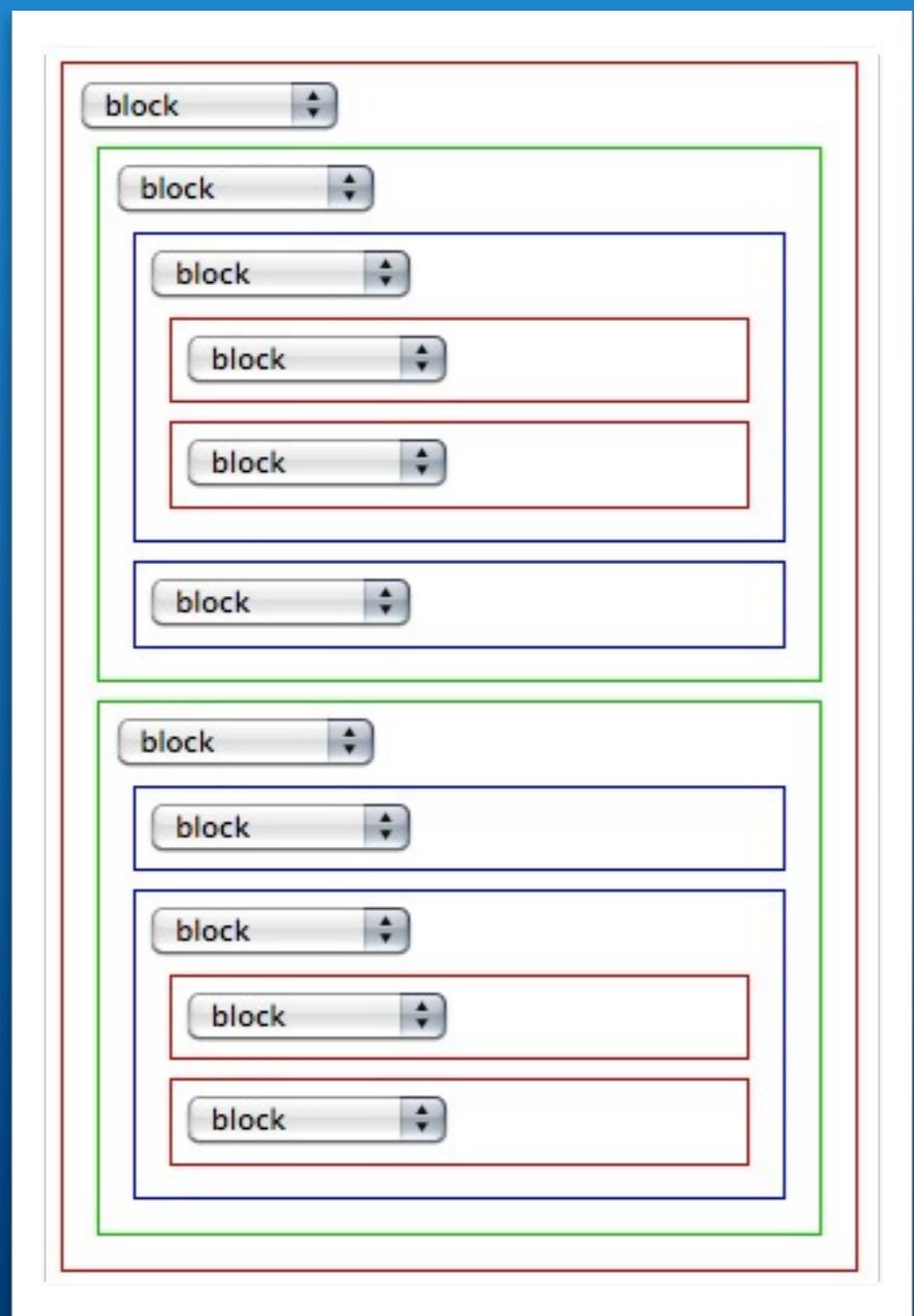
```
display: block
```

- ```
display: list-item
```

- ```
display: list-item
```

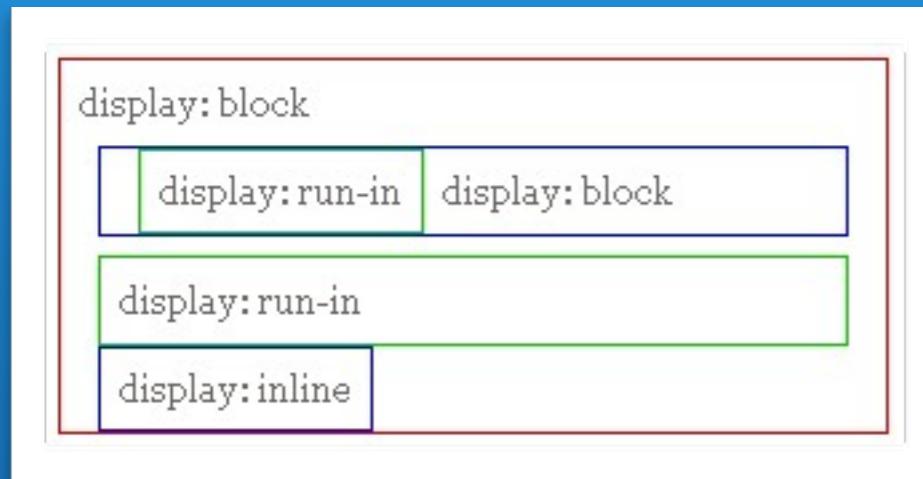
## list-item

목록 요소처럼 표현하고자 할 경우 사용.



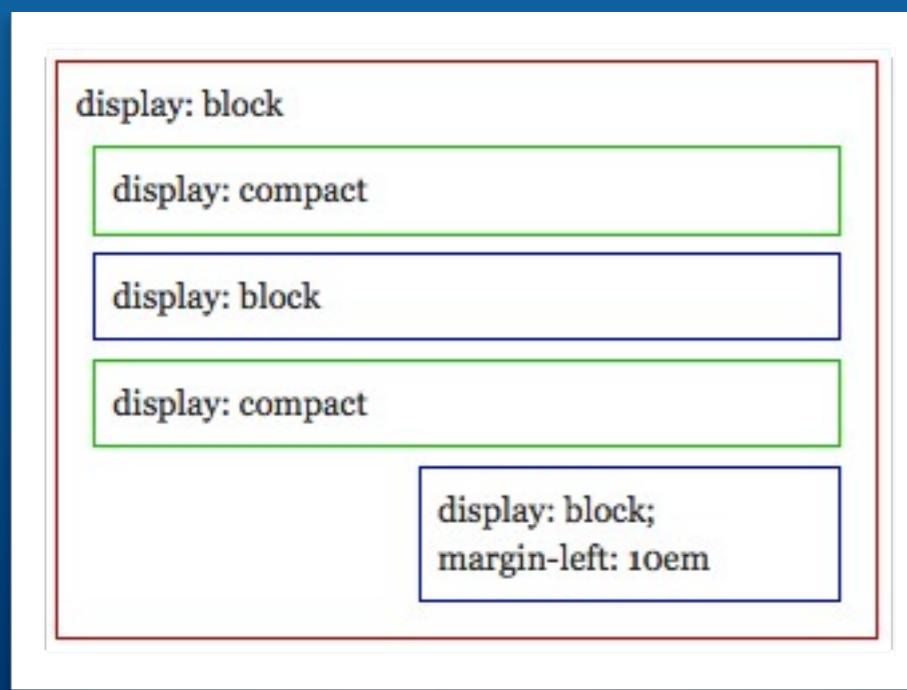
## table, table-row, table-cell

요소를 테이블, 테이블 행, 테이블 열로 표현할 경우 사용.



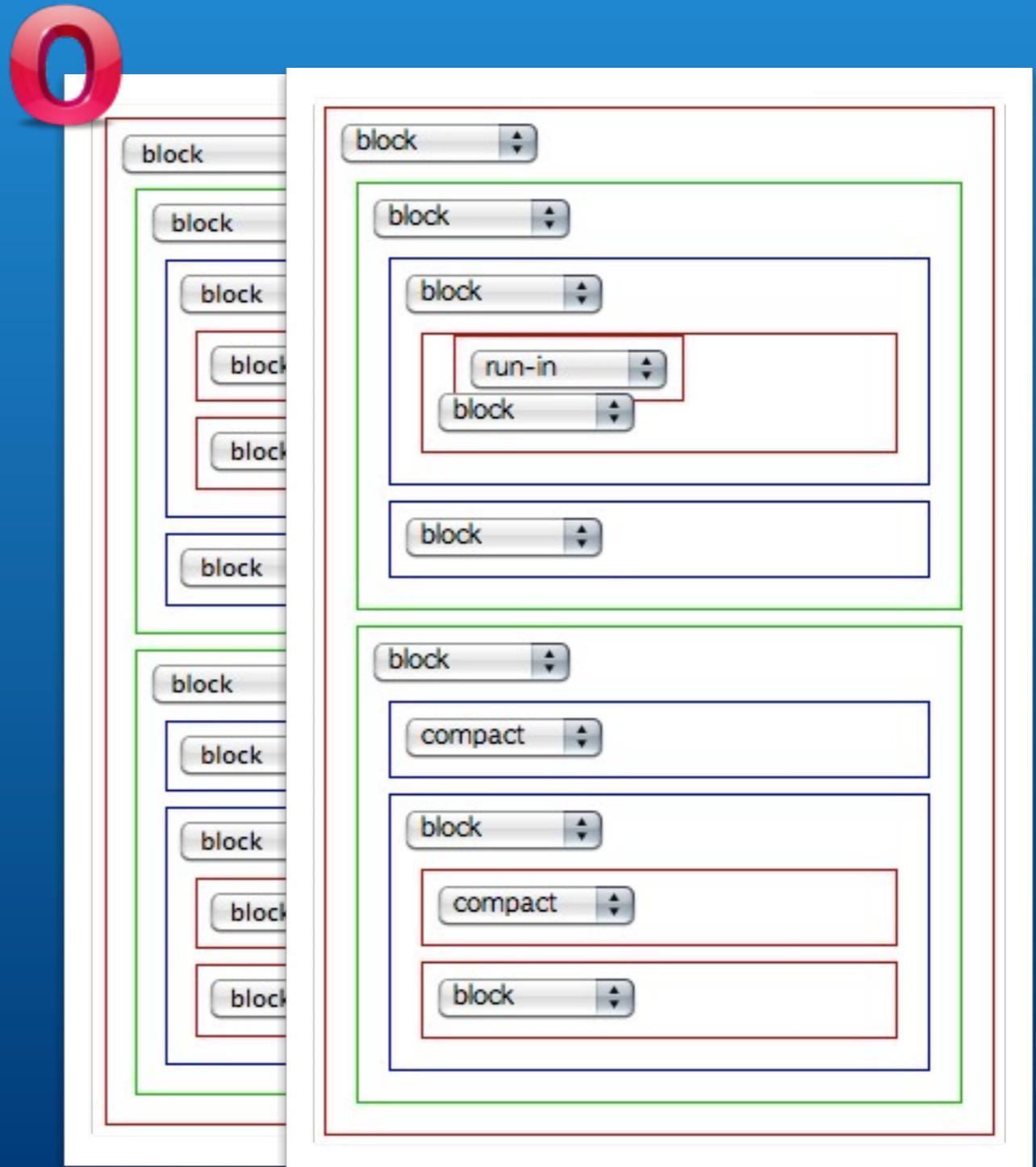
## run-in

블록 박스의 앞에 run-in박스가 있으면 블록 박스의 내부에 인라인 요소로 자리잡아 표현. 그렇지 않을 경우는 반대.



## compact

블록 박스의 앞에 compact박스가 있으면 compact 박스는 한 줄 인라인 요소로 표현. 그렇지 않을 경우는 블록 박스.



<b>value</b>	<b>description</b>	<b>value</b>	<b>description</b>
block	블록 레벨 요소처럼 표현한다.	table-row-group	인라인 테이블의 행 그룹으로 표현한다.
inline	인라인 요소처럼 표현한다.	table-header-group	테이블의 헤더 그룹으로 표현한다.
inline-block	인라인 블록 레벨 요소처럼 표현한다.	table-footer-group	테이블의 푸터 그룹으로 표현한다.
none	화면에 표시하지 않으며, 하위 요소도 마찬가지.	table-row	테이블의 행으로 표현한다.
inherit	부모 요소의 값을 상속 받는다.	table-column-group	테이블의 열 그룹으로 표현한다.
list-item	목록 항목(list-item)으로 표현한다.	table-column	테이블의 열로 표현한다.
marker	앞/뒤 생성된 내용에 표시자가 되도록 지정한다. :before 와 :after 가상 요소에만 지정할 수 있으며, 그 외 요소의 지정하면 inline로 해석된다.	table-cell	테이블의 셀로 표현한다.
run-in	문맥에 따라 인라인 또는 블록 레벨 요소로 표현.	table-caption	테이블의 캡션으로 표현한다.
table	테이블처럼 표현한다.	inline-table	인라인 테이블로 표현한다.



# Layout CSS

스타일시트 레이아웃

- block
- inline
- inline-block
- none

값이 많아 복잡해보이지만...  
왼쪽의 4가지 값만 잘 이해해도  
문제 없습니다. ^\_^



CSS 가시성(시각) 설정은  
투명인간 모드 변경 스위치 같아요!



스위치를 올리면 '짠~'하고  
마법처럼 나타날 거예요.



# CSS언어를 이용하여 대상의 보임, 안보임을 결정해볼까요?



Style code

대상 속성 값  
selector {visibility: value}

<head>



<style> strong {visibility: hidden} </style>  
</head>



# CSS언어를 이용하여 대상의 보임, 안보임을 결정해볼까요?



Style code



<head>



```
<style> strong {visibility: hidden} </style>  
</head>
```



# CSS언어를 이용하여 대상의 보임, 안보임을 결정해볼까요?



Style code



<head> ▲  
<style> strong {visibility: hidden} </style>  
</head>



# CSS언어를 이용하여 대상의 보임, 안보임을 결정해볼까요?



Style code



<head>  
    <style> strong {visibility: hidden} </style>  
  </head>



# CSS언어를 이용하여 대상의 보임, 안보임을 결정해볼까요?



Style code

<head>

대상

속성

값

selector {visibility: value}

visible

hidden

collapse

inherit

<style> strong {visibility: hidden} </style>

</head>

# Layout CSS

스타일시트 레이아웃

display: none; 설정하고  
visibility: hidden; 의 차이점이  
뭐요? 둘 다 같은 것 같아 보이는데..  
이거 참 헷갈려서...

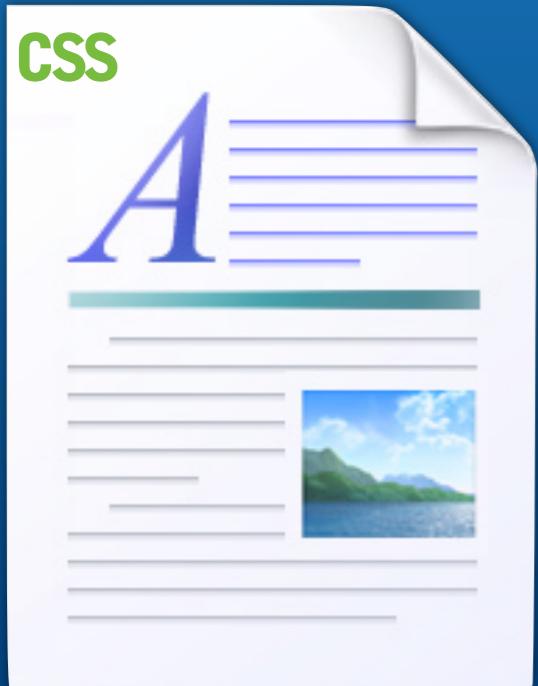


CSS 오버플로우(overflow) 설정을  
완벽하게 컨트롤하지 못하면 레이아웃이 망가집니다!





# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?



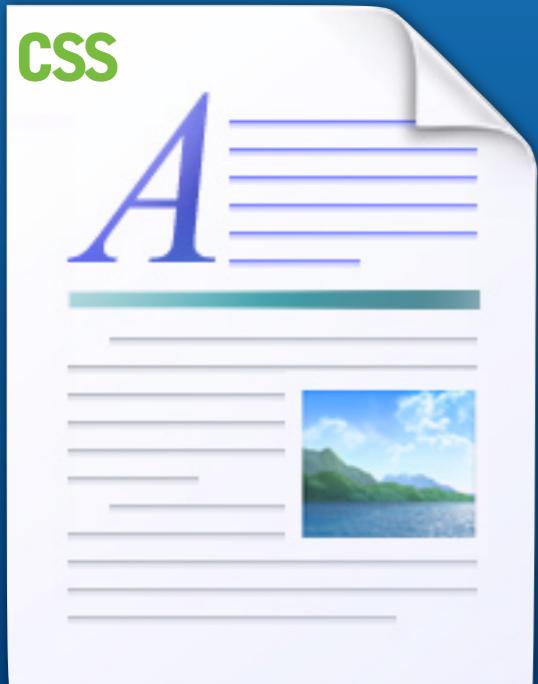
Style code

대상 속성 값  
selector {overflow: value}

```
<head>
  <style>
    div {overflow: auto}
  </style>
</head>
```



# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?



Style code

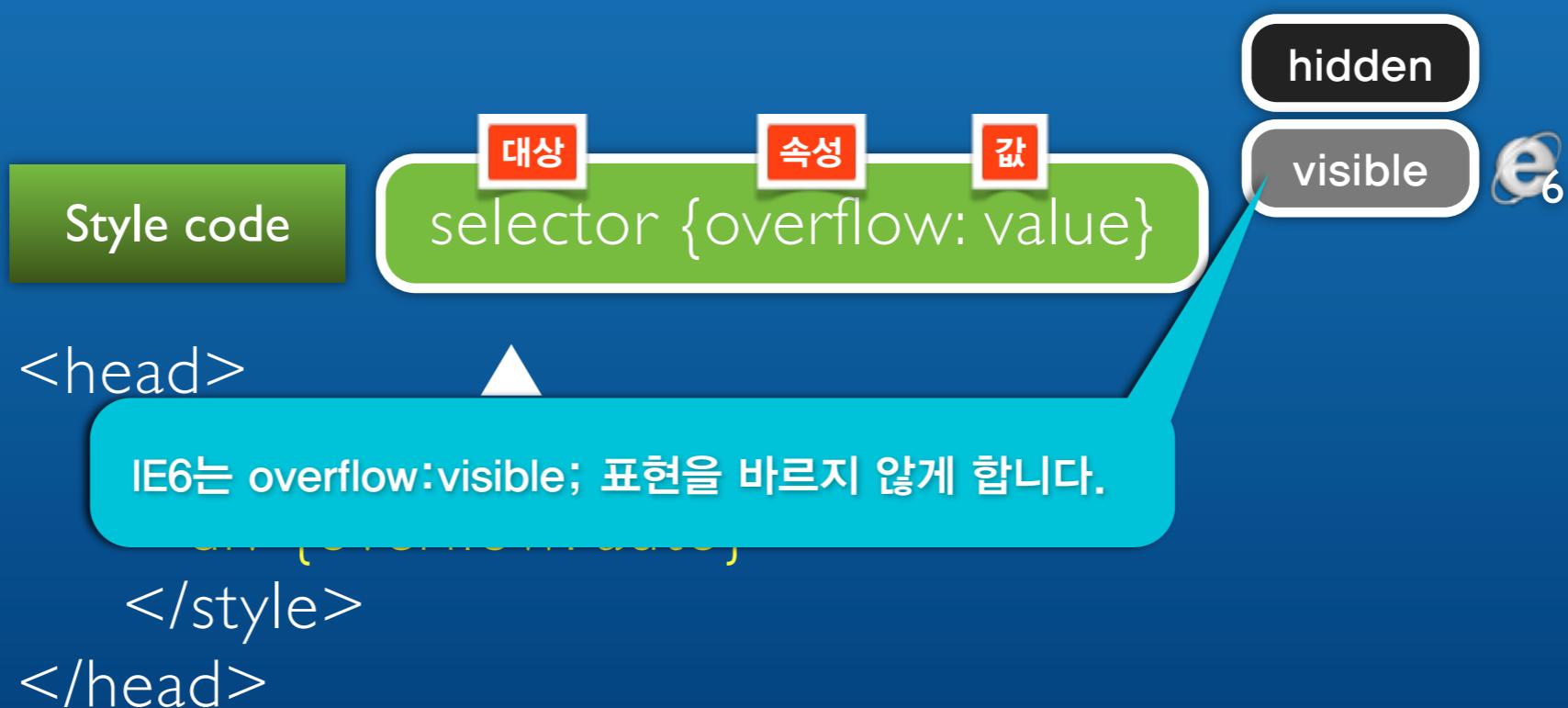
대상 속성 값 selector {overflow: value}

hidden

```
<head>
  <style>
    div {overflow: auto}
  </style>
</head>
```

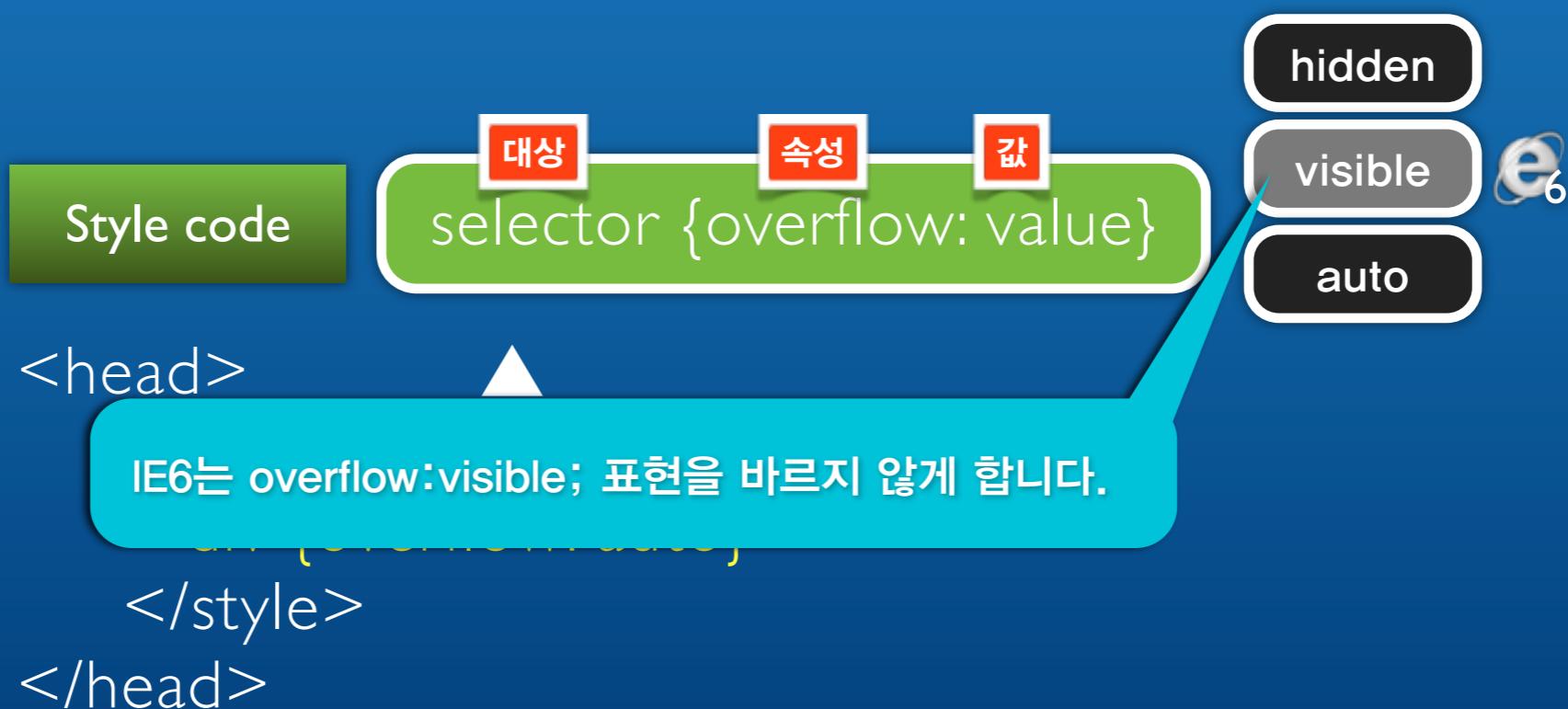
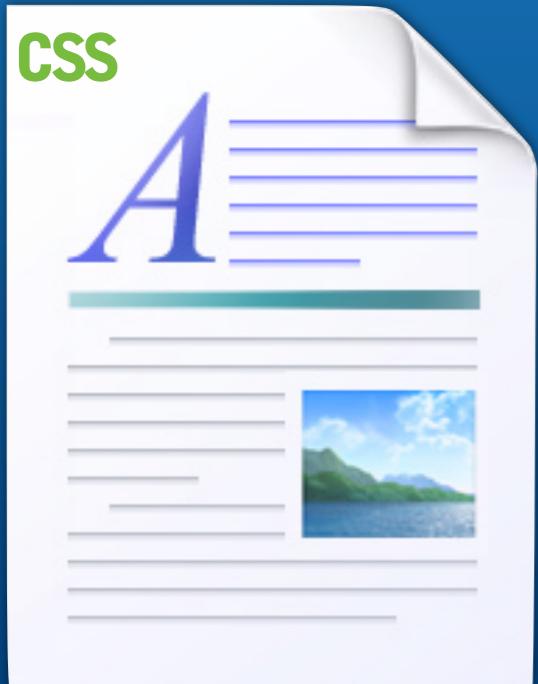


# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?



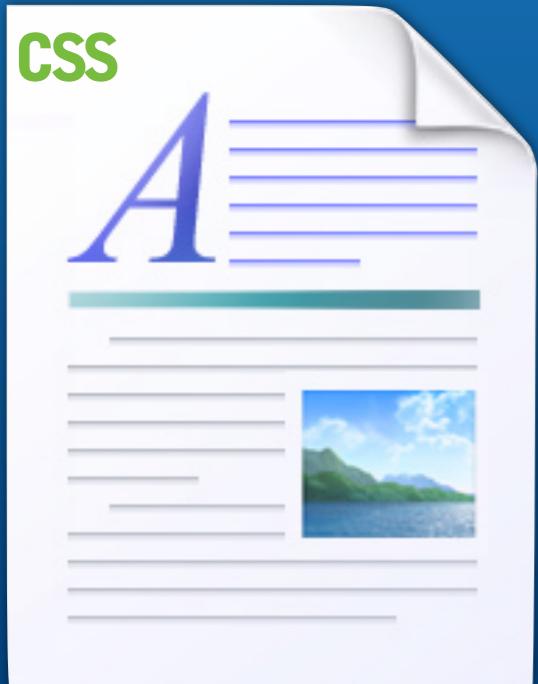


# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?





# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?



Style code

대상 속성 값  
selector {overflow: value}

<head>



IE6는 overflow:visible; 표현을 바르지 않게 합니다.

```
</style>  
</head>
```

hidden

visible



auto

scroll



# CSS언어를 이용하여 대상의 하위요소가 영역을 넘어갔을 경우 표현방법을 적용해볼까요?



Style code

<head>

```
    <style>  
        selector {overflow: value}  
    </style>  
</head>
```

대상 속성 값  
selector {overflow: value}

IE6는 overflow:visible; 표현을 바르지 않게 합니다.

hidden

visible



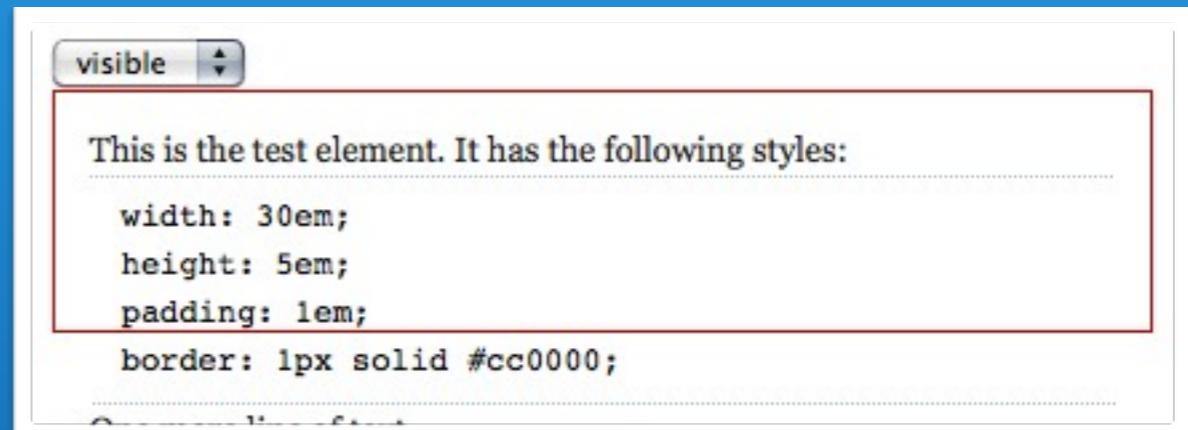
auto

scroll

inherit

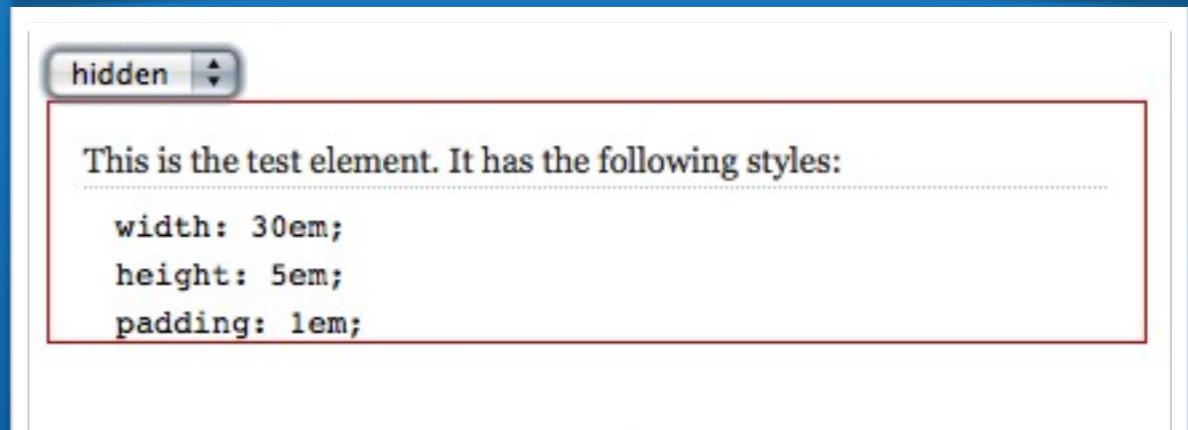
# visible

상위 요소를 넘쳐난 하위 요소가 그대로 보여짐.



# hidden

상위 요소를 넘쳐난 하위 요소를 숨겨줌.



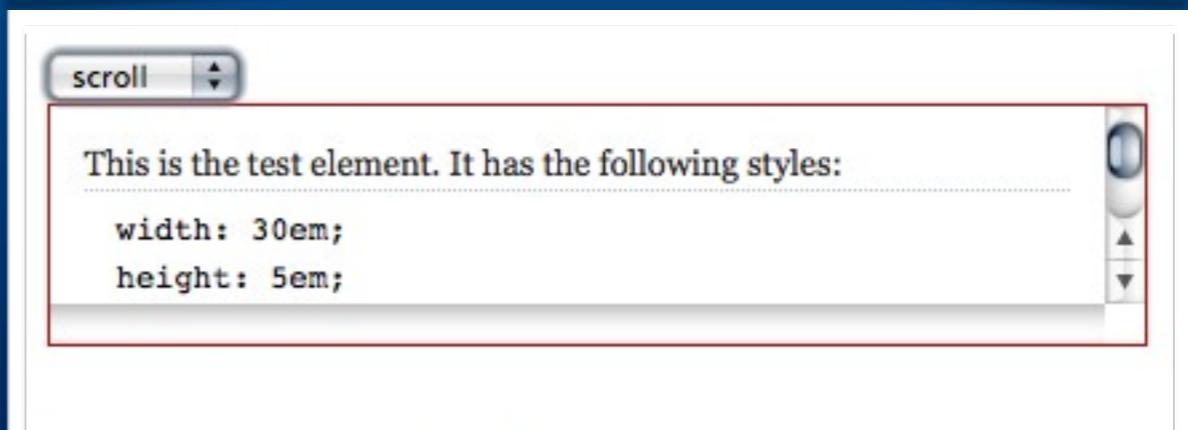
# auto

상위 요소를 하위요소가 넘쳐날 경우만,  
자동으로 스크롤을 표현함.



# scroll

상위 요소를 하위요소가 넘쳐나지 않더라도,  
스크롤을 항상 표현함.





## IE6 incorrectly

IE6는 height를 min-height처럼 표현하는 버그가 존재하기 때문에 하위 요소가 상위 요소의 영역을 넘칠 경우, 상위 요소의 높이가 늘어난다.

visible ▾

This is the test element. It has the following styles:

```
width: 30em;
height: 5em;
padding: 1em;
border: 1px solid #cc0000;
```

overflow: visible and Explorer Windows

visible ▾

This is the test element. It has the following styles:

```
width: 30em;
height: 5em;
padding: 1em;
border: 1px solid #cc0000;
```

One more line of text

overflow: visible and Explorer Windows



## IE9.js를 이용하면 overflow:visible;이 정상 작동합니다.

IE9.js를 이용하면 IE6에서 발생하는 오류(Bug)를 해결하여 브라우저에서 정상으로 보입니다.

### IE6 incorrectly

IE6는 height를 min-height처럼 표현하는 버그가 존재하기 때문에 하위 요소가 상위 요소의 영역을 넘칠 경우, 상위 요소의 높이가 늘어난다.

visible ▾

This is the test element. It has the following styles:

```
width: 30em;
height: 5em;
padding: 1em;
border: 1px solid #cc0000;
```

overflow: visible and Explorer Windows

This is the test element. It has the following styles:

```
width: 30em;
height: 5em;
padding: 1em;
border: 1px solid #cc0000;
```

One more line of text

overflow: visible and Explorer Windows

# IE6.Bug overflow:visible;

Internet Explorer

Local Remote + 08\_IE-bug\_overflow-visible\_fin.html

IE Bug

- 01\_IE-bug\_...egin.html
- 01\_IE-bug\_...g\_fin.html
- 01\_IE-bug\_...izing.mov
- 02\_IE-bug\_...egin.html
- 02\_IE-bug\_...h\_fin.html
- 02\_IE-bug\_...width.mov
- 03\_IE-bug\_...egin.html
- 03\_IE-bug\_...fin.html
- 03\_IE-bug\_...ight.mov
- 04\_IE-bug\_...egin.html
- 04\_IE-bug\_...o\_fin.html
- 04\_IE-bug\_...auto.mov
- 05\_IE-bug\_...egin.html
- 05\_IE-bug\_...g\_fin.html
- 05\_IE-bug\_...ding.mov
- 06\_IE-bug\_...egin.html
- 06\_IE-bug\_...G\_fin.html
- 06\_IE-bug\_PNG.mov
- 07\_IE-bug\_...egin.html
- 07\_IE-bug\_...t\_fin.html
- 07\_IE-bug\_...ment.mov
- 08\_IE-bug\_...egin.html
- 08\_IE-bug\_...e\_fin.html

images

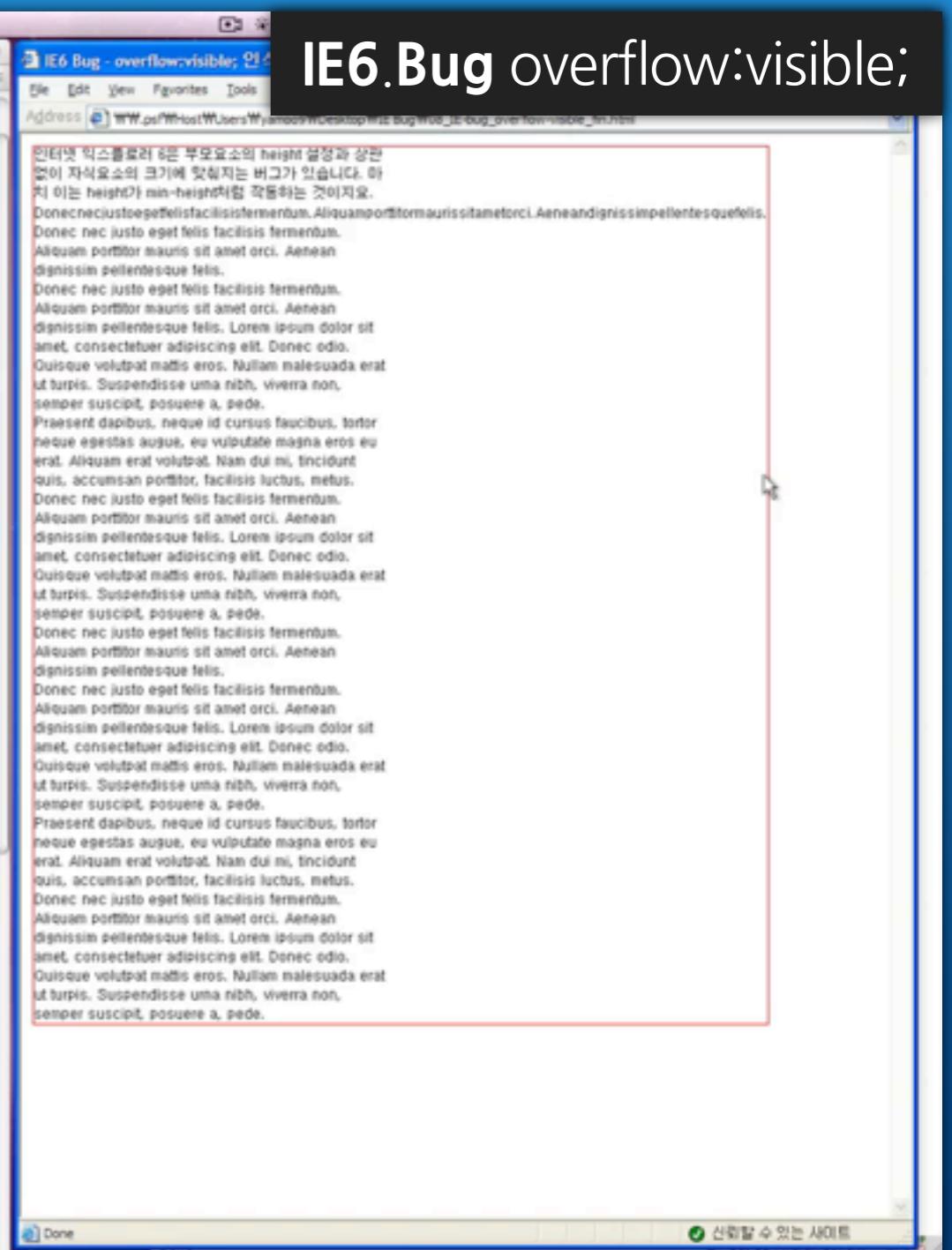
```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>IE6 Bug - overflow:visible; 인식 오류</title>
    <style type="text/css">
      <!--
        /* IE6 Bug - overflow:visible; 인식 오류
        visible, auto, scroll, hidden, inherit
        */
        * {margin:0; padding:0;}
        body {font:12px/1.3 "나눔고딕", "맑은고딕", "돋움", "굴림";}

        #p-wrap {
          margin:10px;
          width:300px;
          height:120px;
          border:1px solid red;
          overflow:visible;

        }
        #p-wrap p {

        }
      -->
    </style>
  </head>
  <body>
    <div id="p-wrap">
      <p>인터넷 익스플로러 6은 부모요소의 height 설정과 상관없이 자식요소의 크기에 맞춰지는 버그가 있습니다. 마치 이는 height가 min-height처럼 작동하는 것 같아요.</p>
      <p>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci. Aenean dignissim pellentesque felis.</p>
      <p>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci. Aenean dignissim pellentesque felis. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec odio. Quisque volutpat mattis eros. Nullam
    </div>
  </body>
</html>
```

Share Hints Clips 19:25



# Layout CSS

스타일시트 레이아웃

`overflow-y: scroll;` 설정을

html 요소에 적용하여 세로 스크롤 바를  
고정하면 화면 디자인이 움직이지 않죠!



CSS 플로트(float) 설정은  
현대 웹 디자인 레이아웃의 핵심입니다!



자유롭게~  
저 하~늘을 날아가도 놀라지 말아요~



# CSS언어를 이용하여 대상의 부유형태를 바꿔 볼까요?



Style code

대상 속성 값  
selector {float: value}

```
<head> ▲  
    <style> img {float: left} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 바꿔 볼까요?



Style code

selector {float: value}

left

<head>



```
<style> img {float: left} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 바꿔 볼까요?



Style code

selector {float: value}

left

right

<head>



```
<style> img {float: left} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 바꿔 볼까요?



Style code

대상 속성 값  
selector {float: value}

left

right

none

```
<head>
    <style> img {float: left} </style>
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 바꿔 볼까요?



Style code

```
<head>
  <style> img {float: left} </style>
</head>
```





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.

Box 1

`div#Box1 {float: right}`

Box 2

Box 3





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

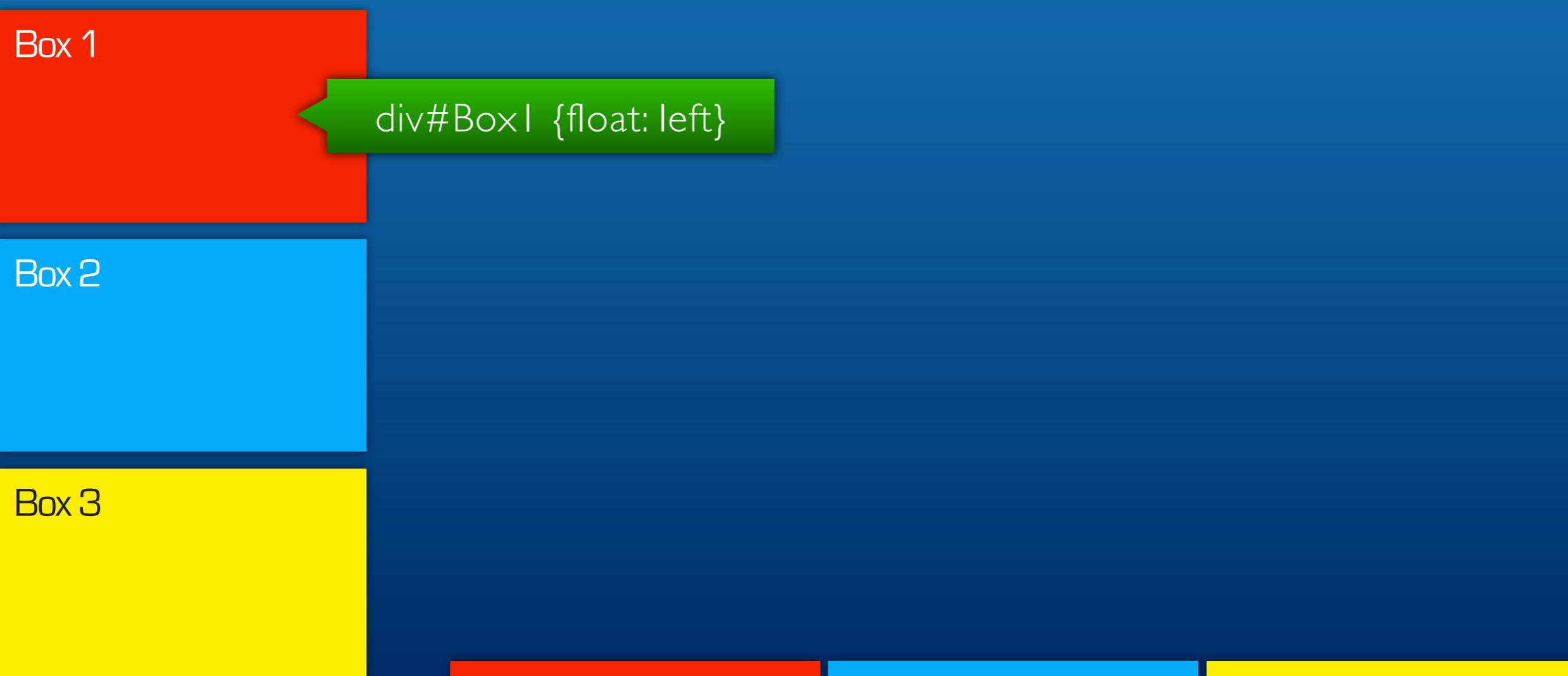
CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.

Box 1

div#Box1 {float: left}

Box 3





# CSS floating Model

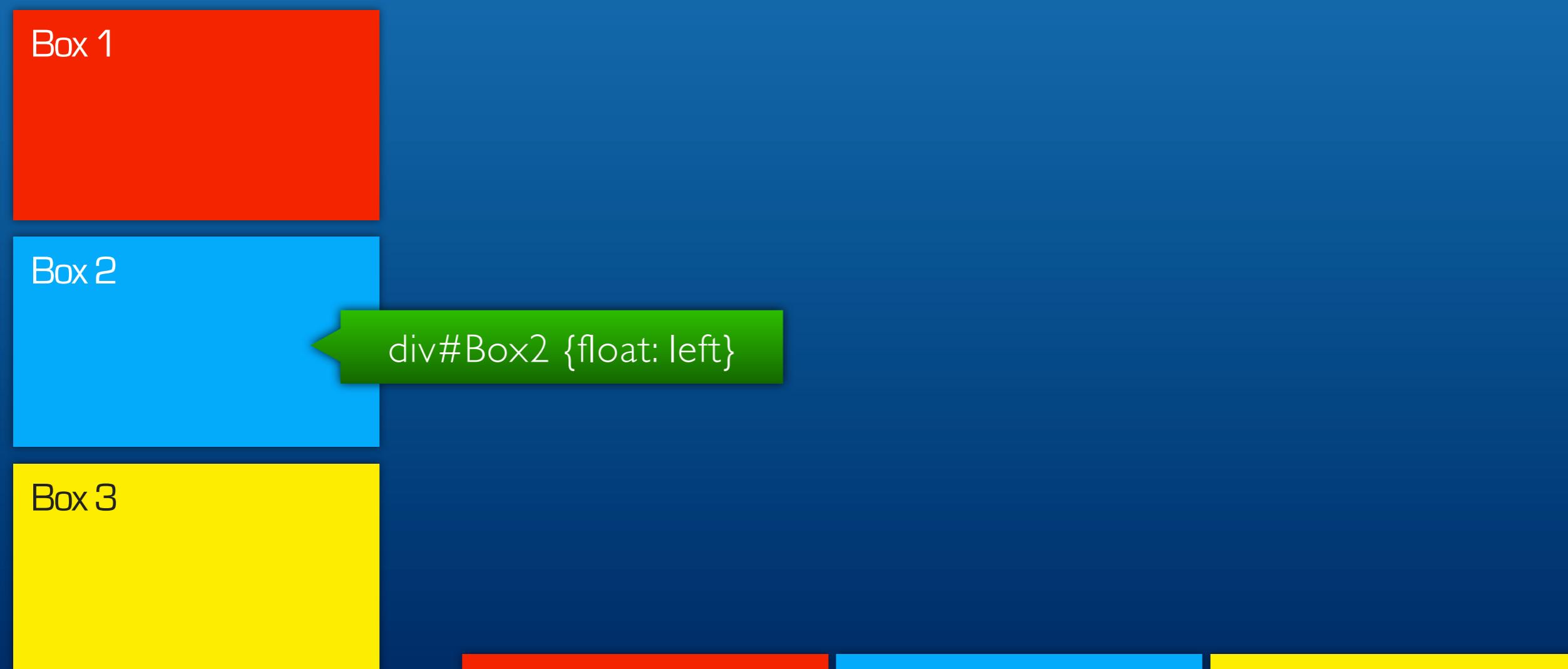
CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.

Box 1

Box 2

div#Box2 {float: left}





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

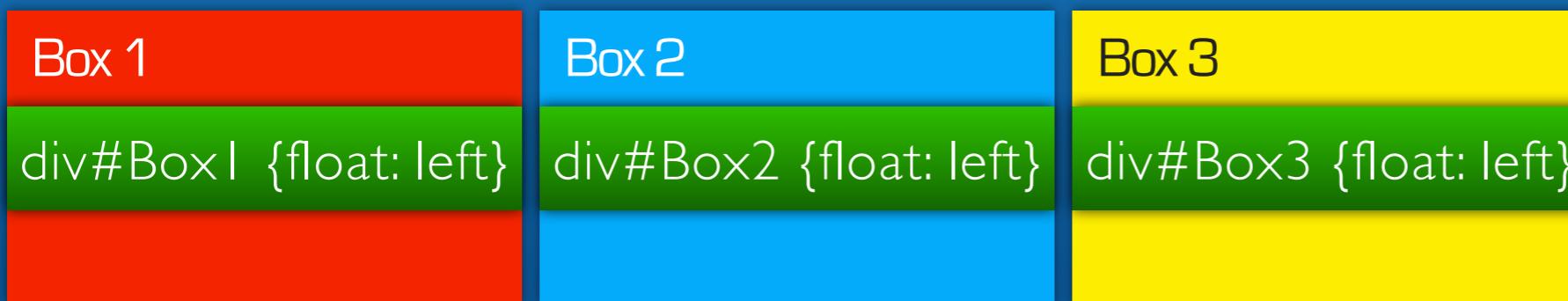
CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

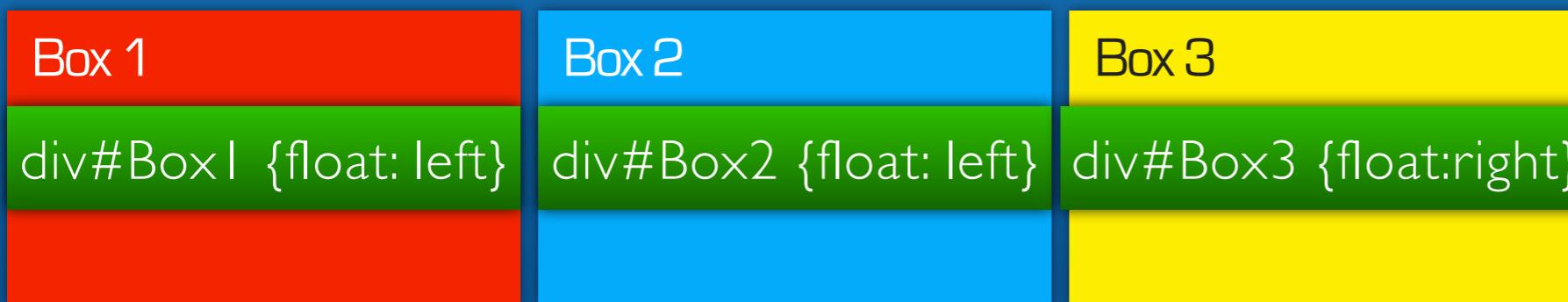
CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.

Box 1

```
div#Box1 {float: left;}
```

Box 2

```
div#Box2 {float: left;}
```

Box 3

```
div#Box3 {float:right;}
```





# CSS floating Model

CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.

Box 1

Box 2

```
div#Box1 {float: left} div#Box2 {float: right}
```

Box 3

```
div#Box3 {float:right}
```





# CSS floating Model

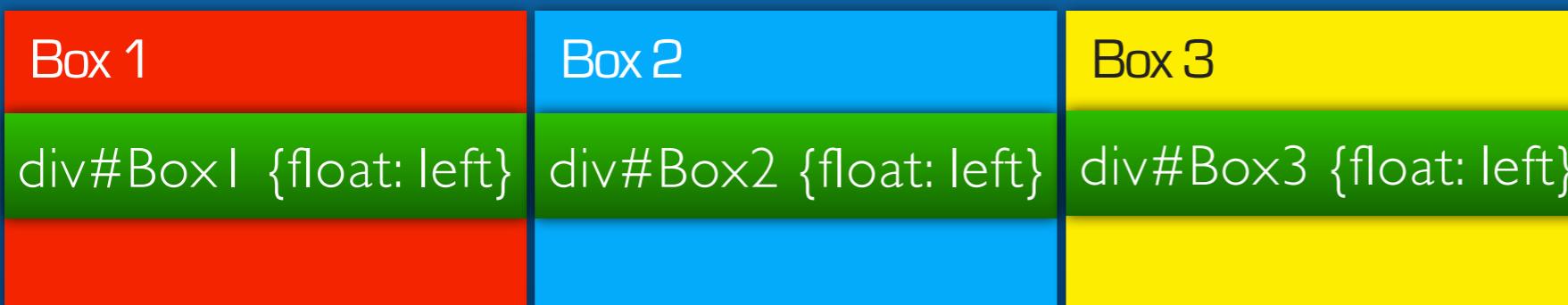
CSS float 속성은 값에 따라 대상을 문서의 흐름에서 띄워줄 수 있습니다.





# CSS float Drop

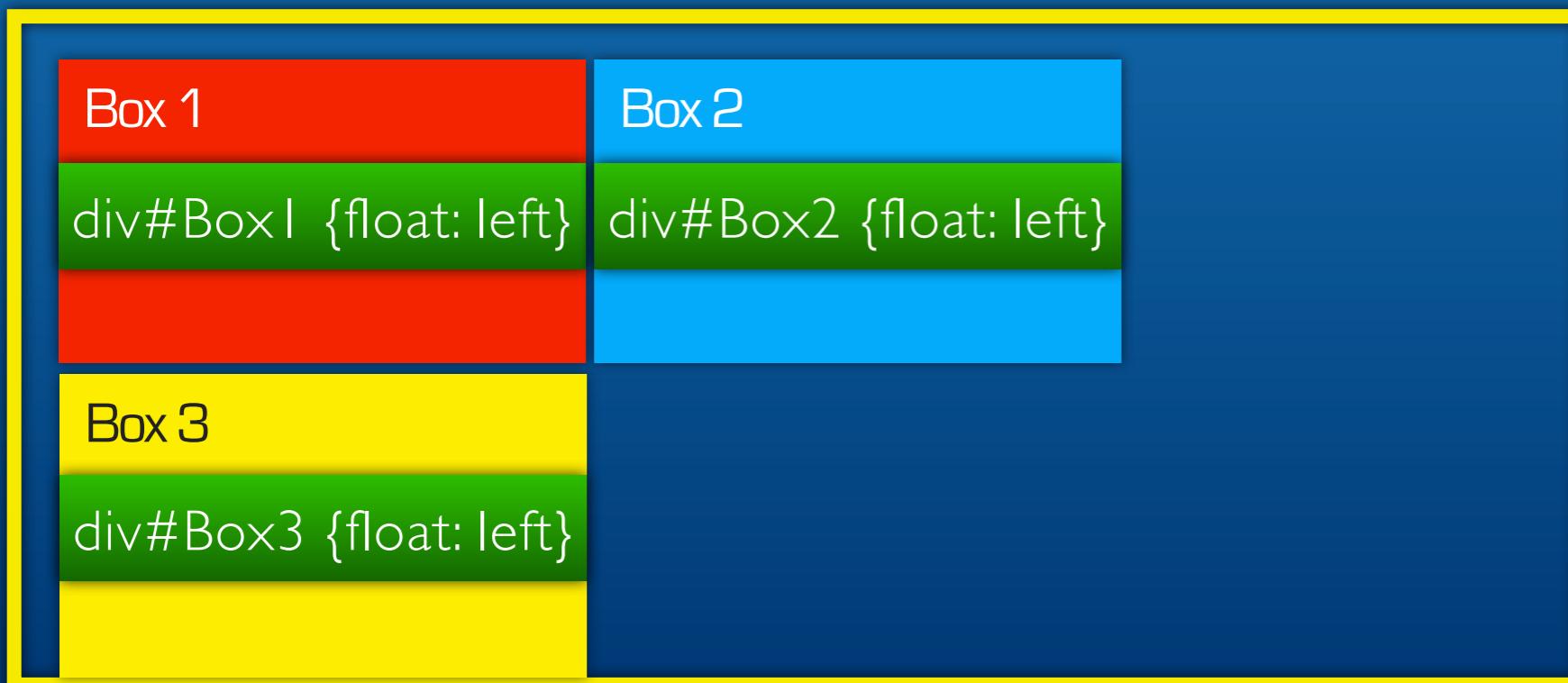
CSS float 속성을 통해 띄운 엘리먼트들의  
가로폭의 합이 포함하는 부모 엘리먼트의  
가로 폭보다 크다면?





# CSS float Drop

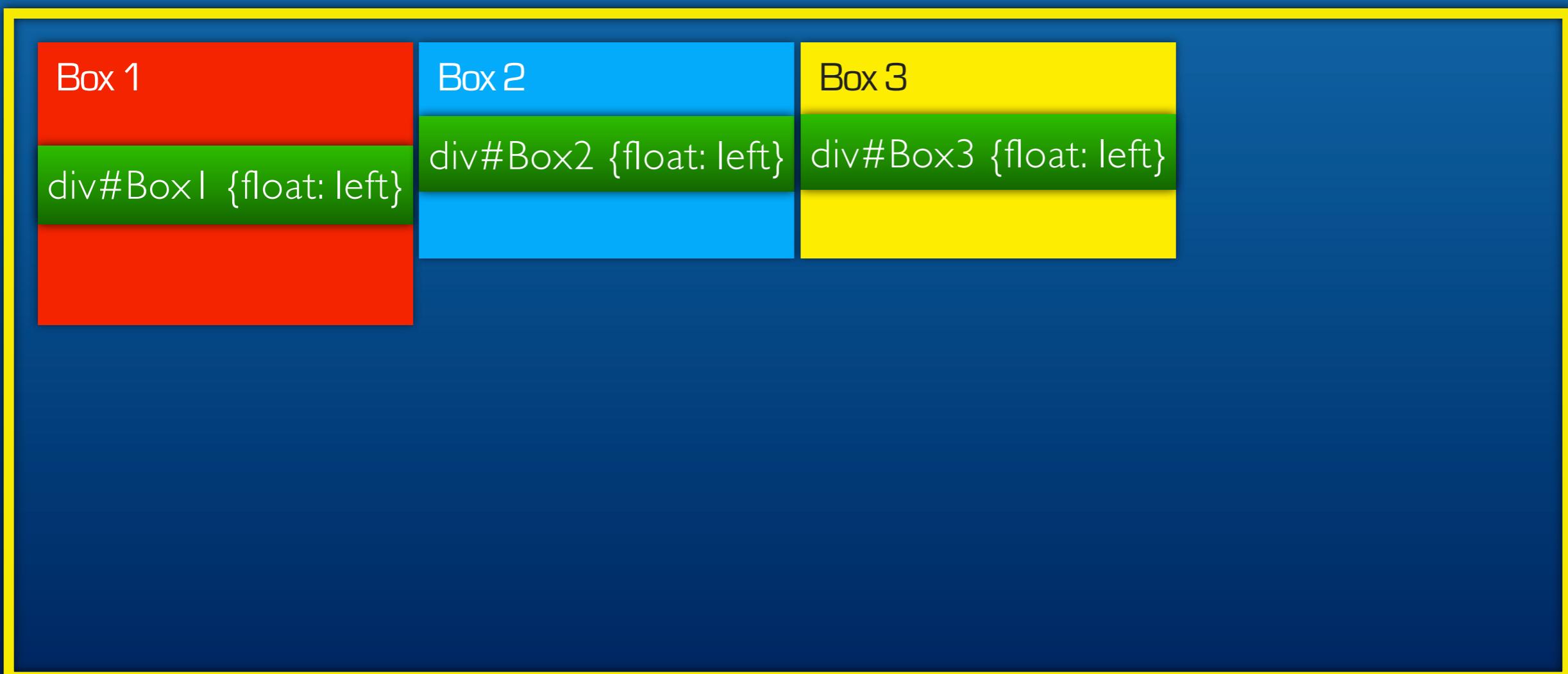
CSS float 속성을 통해 띄운 엘리먼트들의  
가로폭의 합이 포함하는 부모 엘리먼트의  
가로 폭보다 크다면?





# CSS float Drop

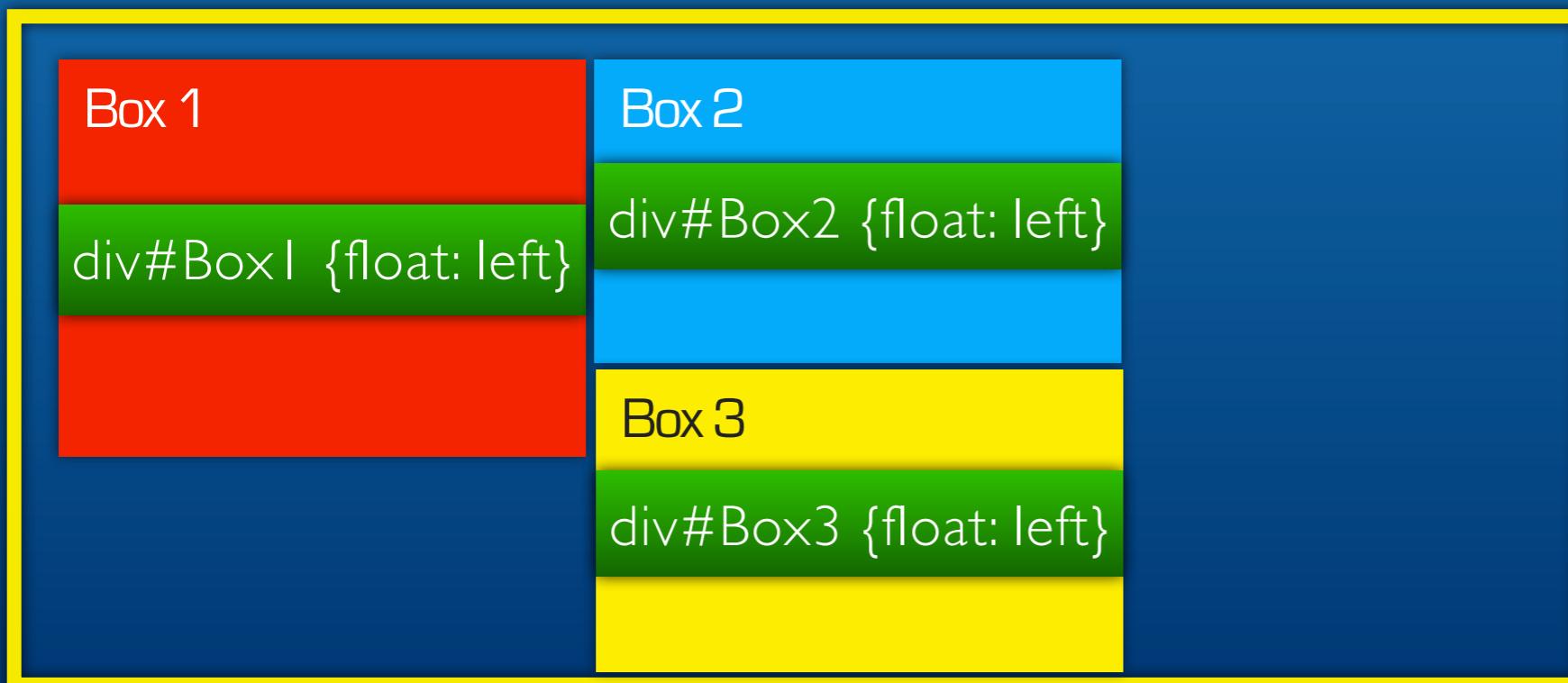
CSS float 속성을 통해 띄운 엘리먼트들의  
가로폭의 합이 포함하는 부모 엘리먼트의  
가로 폭보다 크다면?





# CSS float Drop

CSS float 속성을 통해 띄운 엘리먼트들의  
가로폭의 합이 포함하는 부모 엘리먼트의  
가로 폭보다 크다면?



# Layout CSS

스타일시트 레이아웃

요소에 float 설정을 하면  
레이아웃을 잡을 수 있어 기뻤는데...  
와? 아래 요소까지 영향을 미치는 건지  
잘 모르겠어요....





# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



Style code

대상 속성 값  
selector {clear: value}

<head>



```
<style> p {clear: both} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



Style code

대상 속성 값  
selector {clear: value}

none

```
<head>      ▲  
    <style> p {clear: both} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



```
<head> ▲  
    <style> p {clear: both} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



Style code



```
<head> ▲  
    <style> p {clear: both} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



# Style code



```
<head>      ▲  
    <style> p {clear: both} </style>  
</head>
```



# CSS언어를 이용하여 대상의 부유형태를 해제해 볼까요?



Style code

<head>  
    <style> p {clear: both} </style>  
</head>

대상 속성 값  
selector {clear: value}

- none
- left
- right
- both
- inherit



# CSS floating Image

CSS float 속성 설정을 통해 이미지를 둘러싸는 형태의 디자인이 가능합니다.



아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요.

이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# CSS floating Image

CSS float 속성 설정을 통해 이미지를 둘러싸는 형태의 디자인이 가능합니다.



```
img {float: left}
```

아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요.

이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# CSS floating Image

CSS float 속성 설정을 통해 이미지를 둘러싸는 형태의 디자인이 가능합니다.



아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요.

이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# CSS clear floating

CSS clear 속성 설정을 통해  
float 통제에서 벗어날 수 있습니다.



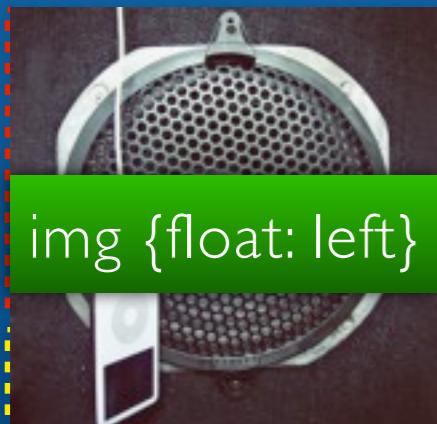
아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요.

이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# CSS clear floating

CSS clear 속성 설정을 통해 float 통제에서 벗어날 수 있습니다.



아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요.

p.clear {clear:left}

이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

float된 하위요소에 대처하는 상위요소의 자세

## Parent Element



아이팟과 스피커의 색상 매치와 재질감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

float된 하위요소에 대처하는 상위요소의 자세

## Parent Element



아이팟과 스피커의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

```
p {float: right;}
```



# Caution using float

float된 하위요소에 대처하는 상위요소의 자세

## Parent Element



img {float: left}

아이팟과 스피

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

float된 하위요소를 감싸는 상위요소는  
떠버린 하위요소를 감싸지 못합니다.



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 1

### Parent Element



img {float: left}

아이팟과 스피

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 1

Parent Element



img {float: left}

아이팟과 스피커

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

div {float: left}

float된 하위요소를 감싸는 상위요소도  
**하위요소처럼 띄워라!**



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 2

### Parent Element



img {float: left;}

아이팟과 스피커

p {float: right;}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 2

### Parent Element

아이팟과 스피커의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

br {clear: both}

float된 하위요소를 감싸는 상위요소 안에  
새로운 요소로 클리어하라!



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 3

### Parent Element



img {float: left}

아이팟과 스피

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

## 떠버린 하위요소를 감싸는 방법 3

Parent Element



img {float: left}

아이팟과 스피커

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

div {overflow: hidden}

float된 하위요소를 감싸는 상위요소에  
overflow 속성값을 hidden 또는 auto로!



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 4

### Parent Element



img {float: left}

아이팟과 스피

p {float: right}

감의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 4

Parent Element .clearfix:after {content:"."; height:0; visibility:hidden; display:block; clear:both}



img {float: left}

아이팟과 스피커의 매칭이 매우 자연스럽군요. 스피커를 반으로 가른 레이아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

float된 하위요소를 상위요소가 감싸안기 위해  
:after 가상요소를 활용하라!



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 4

Parent Element .clearfix:after {content:"."; height:0; visibility:hidden; display:block; clear:both}

img {float: left;}.IE6 .clearfix {height:1%}, IE7 .clearfix {overflow: hidden;} 피커를 반으로 가른 레이아웃



아웃 중심에서 좌측으로 조금 치우쳐진 아이팟이 하단의 무게감을 무너뜨리고, 가는 라인이 상단에서 흐름을 방해하지 않는 범위에서 연결시켜주는 역할을 수행하고 있군요. 이미지를 상단에 놓고 하단에 단락이 위치하고 있습니다. 이미지에 float 속성 값을 부여하게 되면 이미지를 둘러싸는 단락을 확인하실 수 있습니다. float는 이런 목적으로 처음 고안되어 세상에 태어나게 되었답니다.

float된 하위요소를 상위요소가 감싸안기 위해  
:after 가상요소를 활용하라!



# Caution using float

## 떠버린 하위요소를 감싸안는 방법 4

Parent Element .clearfix:after {content:"."; height:0; visibility:hidden; display:block; clear:both}



IE9.js를 이용하면 별도의 IE7, IE6 코드 적용 없이도  
.clearfix:after가 정상 작동합니다.

IE9.js를 이용하면 :after, content를 IE6, IE7에서 인식할 수 있기에 가능해집니다.

난락을 확인하실 수 있습니다. float는 이런 녹석으로 저음 고안되어 세상에 태어나게 되었답니다.

float된 하위요소를 상위요소가 감싸안기 위해  
:after 가상요소를 활용하라!

# IE6.Bug clear-fix;

The screenshot shows a Firefox browser window with two tabs open. The left tab displays the source code for a fix for an IE6 bug. The right tab shows the rendered output of the same code in Internet Explorer 6.

**Source Code (Left Tab):**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>IE6 Bug - float clear-fix</title>
    <style type="text/css">
      /* IE6 Bug - float clear-fix */
      * {margin:0; padding:0;}
      body {font:12px/1.3 "나눔고딕", "맑은고딕", "돋움", "굴림";}

      #p-wrap {
        margin:10px;
        background:#0fcfef;
        border: 1px solid #0fcfef;
      }
      #p-wrap:after {
        content:"새로 생성된 요소";
        display:block;
        clear:both;
      }
      #p-wrap p {
        float:left;
        width:120px;
        padding-right:10px;
      }
    -->
    </style>
  </head>
  <body>
    <div id="p-wrap">
      <p>인터넷 익스플로러 6은 :after, contet를 인식하지 못하기 때문입니다.</p>
      <p>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor ma...</p>
      <p>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor ...</p>
    </div>
  </body>
</html>
```

**Rendered Output (Right Tab):**

The rendered output in Internet Explorer 6 shows the following content:

```
인터넷 익스플로러 6은 :after, contet를 인식하지 못하기 때문에 새로 생성된 요소를 감지할 수 없습니다.
```

The text "인터넷 익스플로러 6은 :after, contet를 인식하지 못하기 때문에 새로 생성된 요소를 감지할 수 없습니다." is displayed in a black font on a white background. Below this, there are several paragraphs of placeholder text (Lorem ipsum dolor sit amet...) repeated across the page.

플로트 내비게이션 유형



# Navigation

web standard project

플로트 내비게이션 유형





# Navigation Design

web standard project

유형	깊이	방법
세로형	레벨 1	인라인, 플로팅, 포지셔닝
가로형	레벨 1	인라인, 플로팅, 포지셔닝
세로형	레벨 2	플로팅, 포지셔닝
가로형	레벨 2	플로팅, 포지셔닝
세로형	레벨 3	플로팅, 포지셔닝
가로형	레벨 3	플로팅, 포지셔닝

# Vertical Navigation

세로형 내비게이션은 생각 이상으로 쉽습니다.

목록(Lists)을 사용하는 내비게이션의 경우 기본적으로 블록 요소를 다루기 때문에  
신경 쓸 부분이 상대적으로 작습니다.



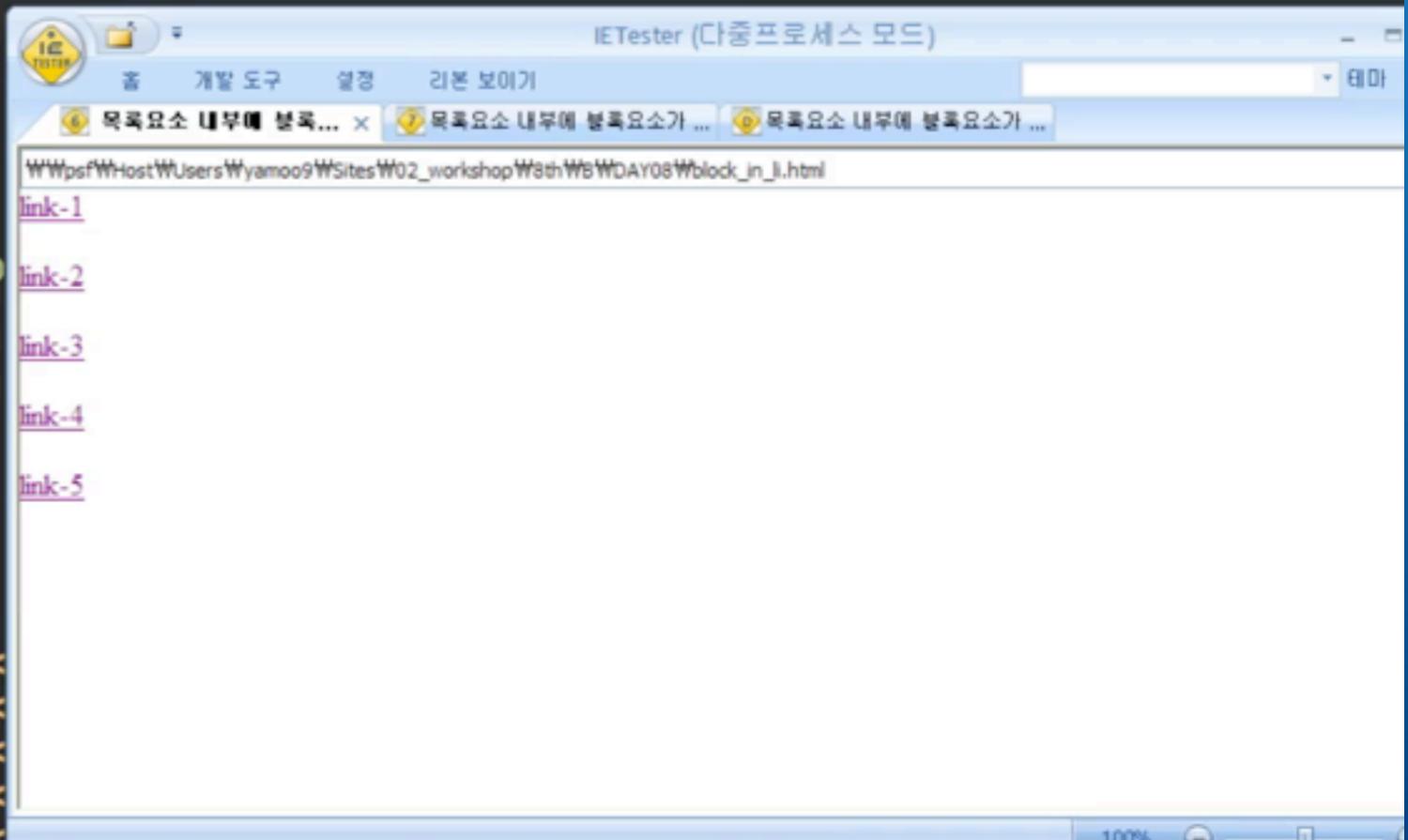
# Horizontal Navigation

가로형 내비게이션은 생각보다 처리할 것이 많습니다.

블록 요소를 인라인화 시키거나, 플로팅 시키는 방법으로 처리해야 하기 때문에  
신경 쓸 부분이 상대적으로 많습니다.



## IE6.Bug Bottom Space;



The screenshot shows a code editor and a browser window. The code editor on the left displays the HTML and CSS code for a page named 'block\_in\_li.html'. The browser window on the right, titled 'IETester (다중프로세스 모드)', shows the rendered page. The page contains a title in Korean and a list of five links labeled link-1 through link-5. There is a noticeable amount of vertical space at the bottom of the list, which is a known bug in Internet Explorer 6.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional<br/>"<br/><html lang="ko"><br/><head><br/><meta http-equiv="Content-Type" content="text/html; charset=utf-8"><br/><title>목록요소 내부에 블록요소가 있을 경우, IE6에서 발생하는 하단 공백 문제</title><br/><style type="text/css"><br/><body, ul {<br/>    margin: 0;<br/>    padding: 0;<br/>}<br/>ul {<br/>    list-style-type: none;<br/>}<br/>ul a {<br/>    display: block;<br/>}<br/></style><br/></head><br/><body><br/><ul><br/>    <li><a href="#">link-1</a></li><li><a href="#">link-2</a></li><li><a href="#">link-3</a></li><li><a href="#">link-4</a></li><li><a href="#">link-5</a></li></ul><br/></body><br/></html>
```

플로트 레이아웃 유형





# Lay-out

web standard project

플로트 레이아웃 유형





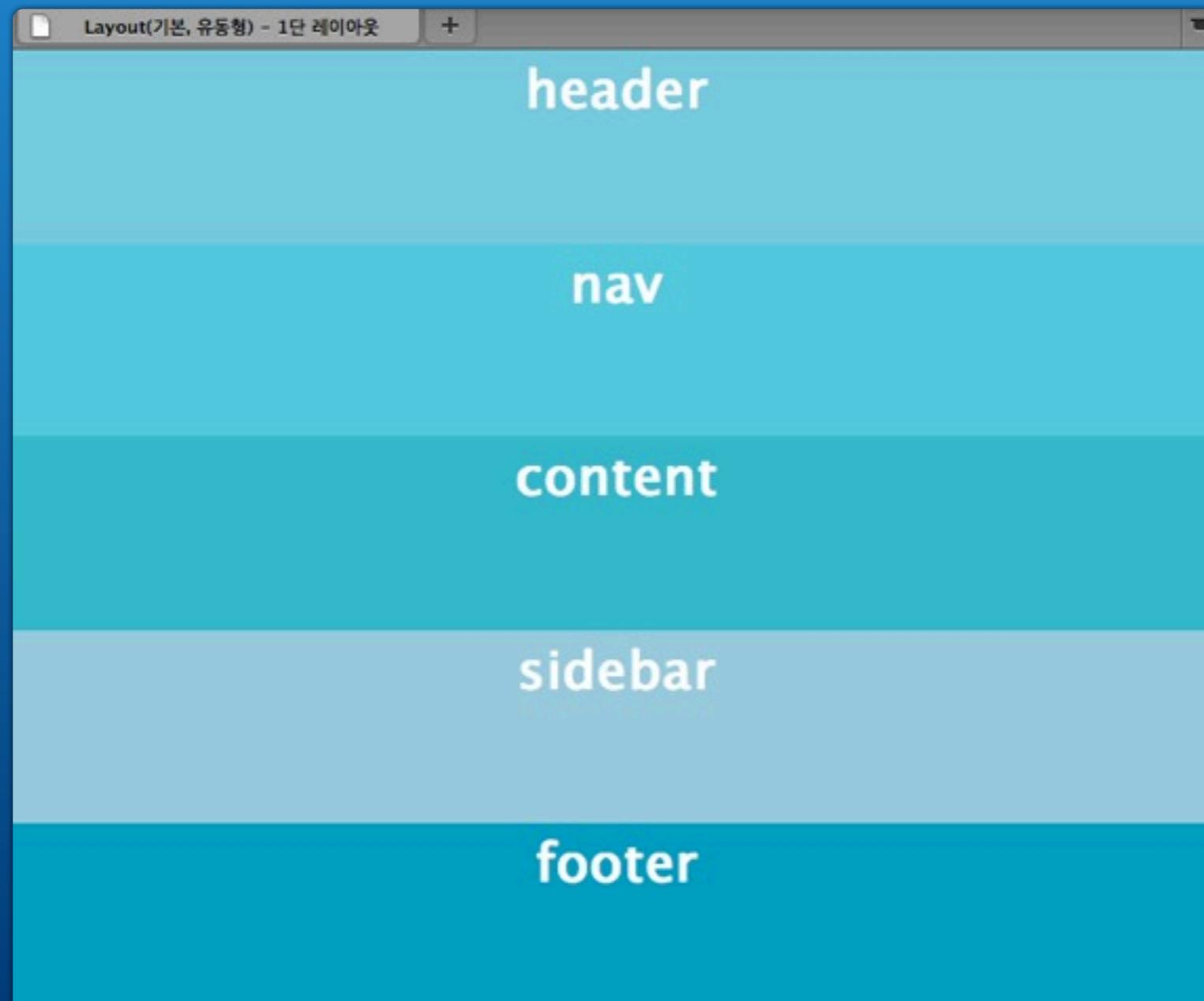
# Lay-out Design

web standard project

유형	단계	주요 컨텐츠 위치
유동 (%)	1단	.
유동 (%)	2단	왼쪽-A
유동 (%)	2단	오른쪽-A
유동 (%)	2단	왼쪽-B
유동 (%)	2단	오른쪽-B
유동 (%)	3단	가운데-A
유동 (%)	3단	가운데-B

# Lay-out Design

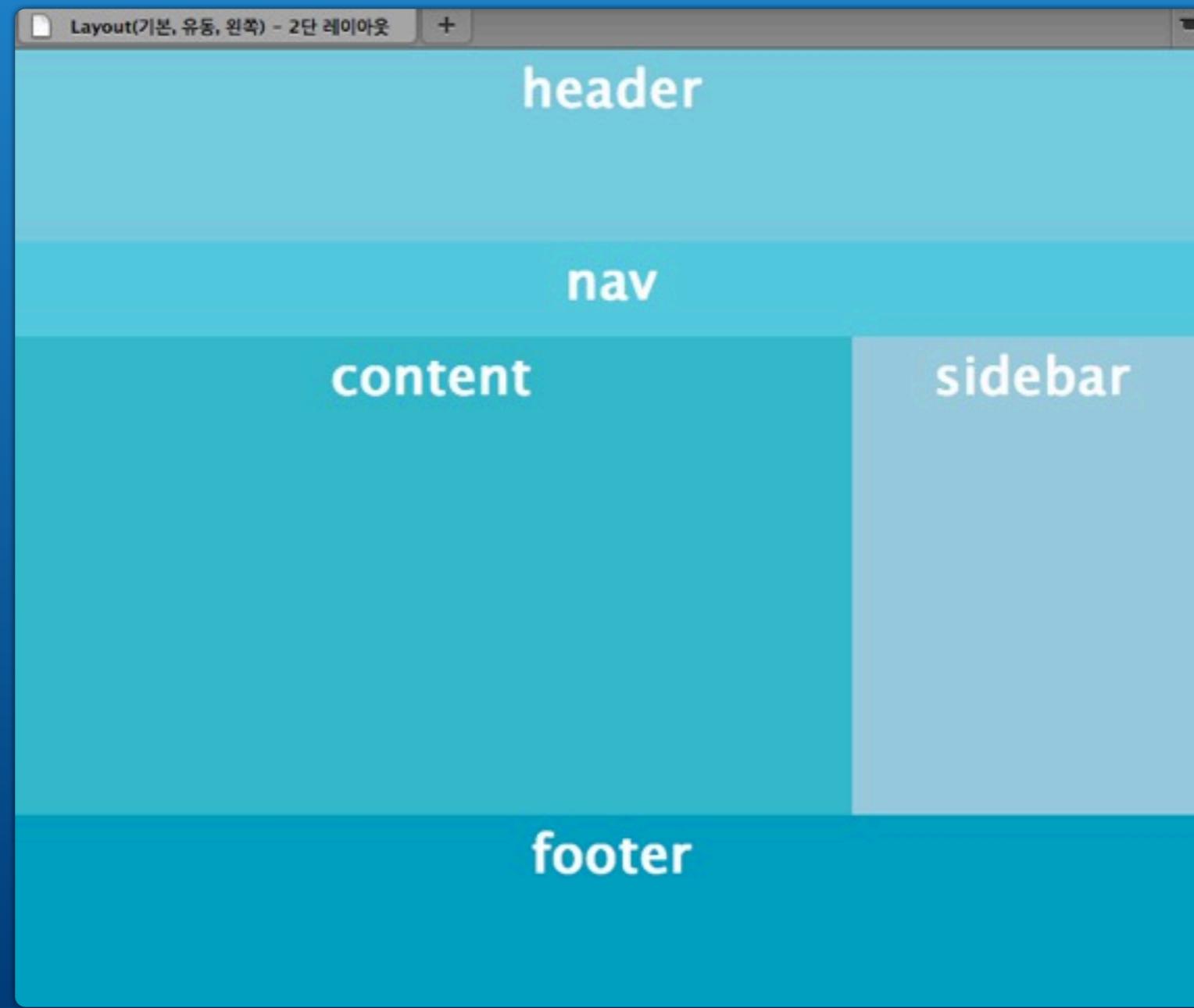
web standard project



유동 1단 레이아웃  
layout

# Lay-out Design

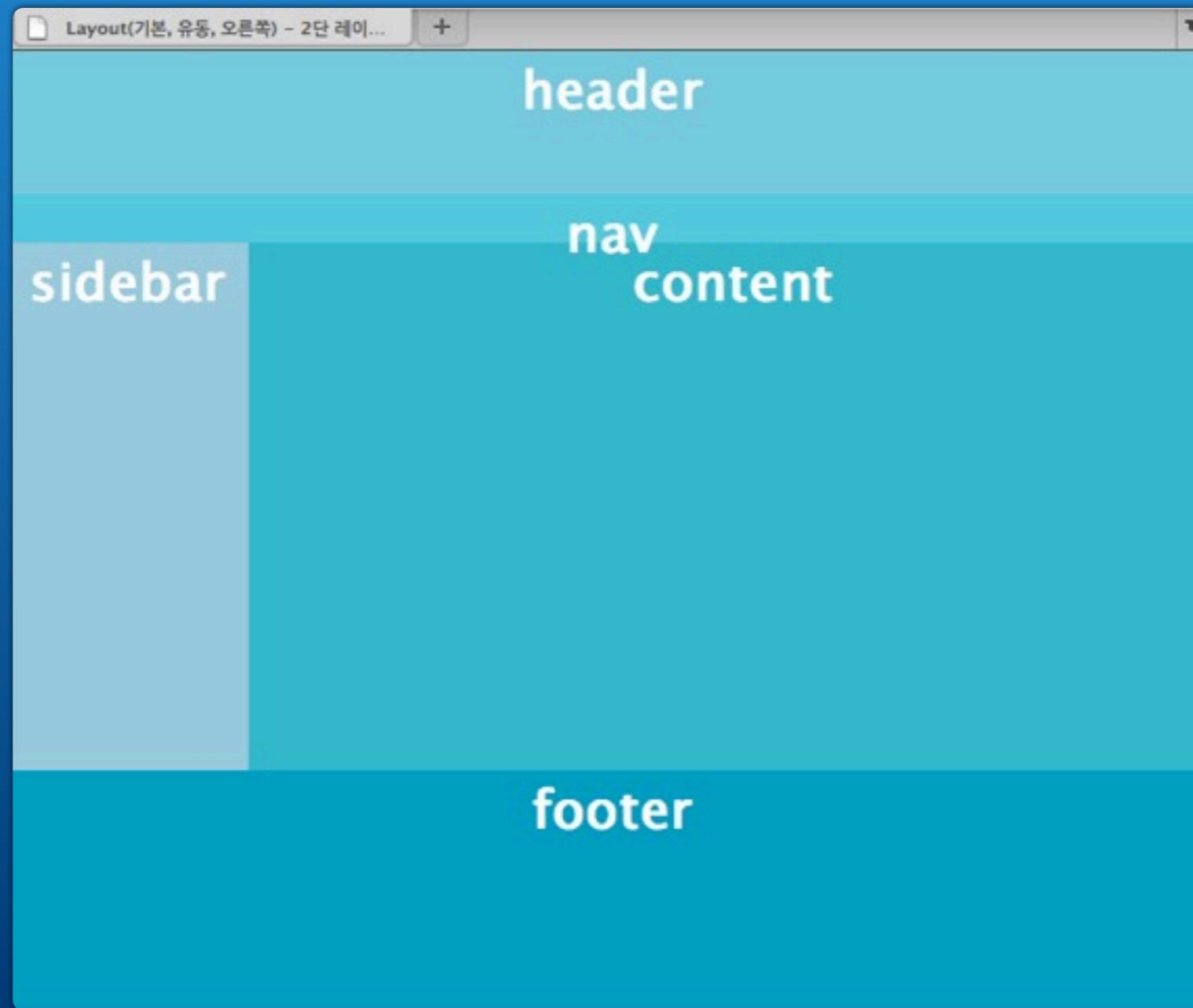
web standard project



유동 2단 좌측-A 주요 컨텐츠 레이아웃

# Lay-out Design

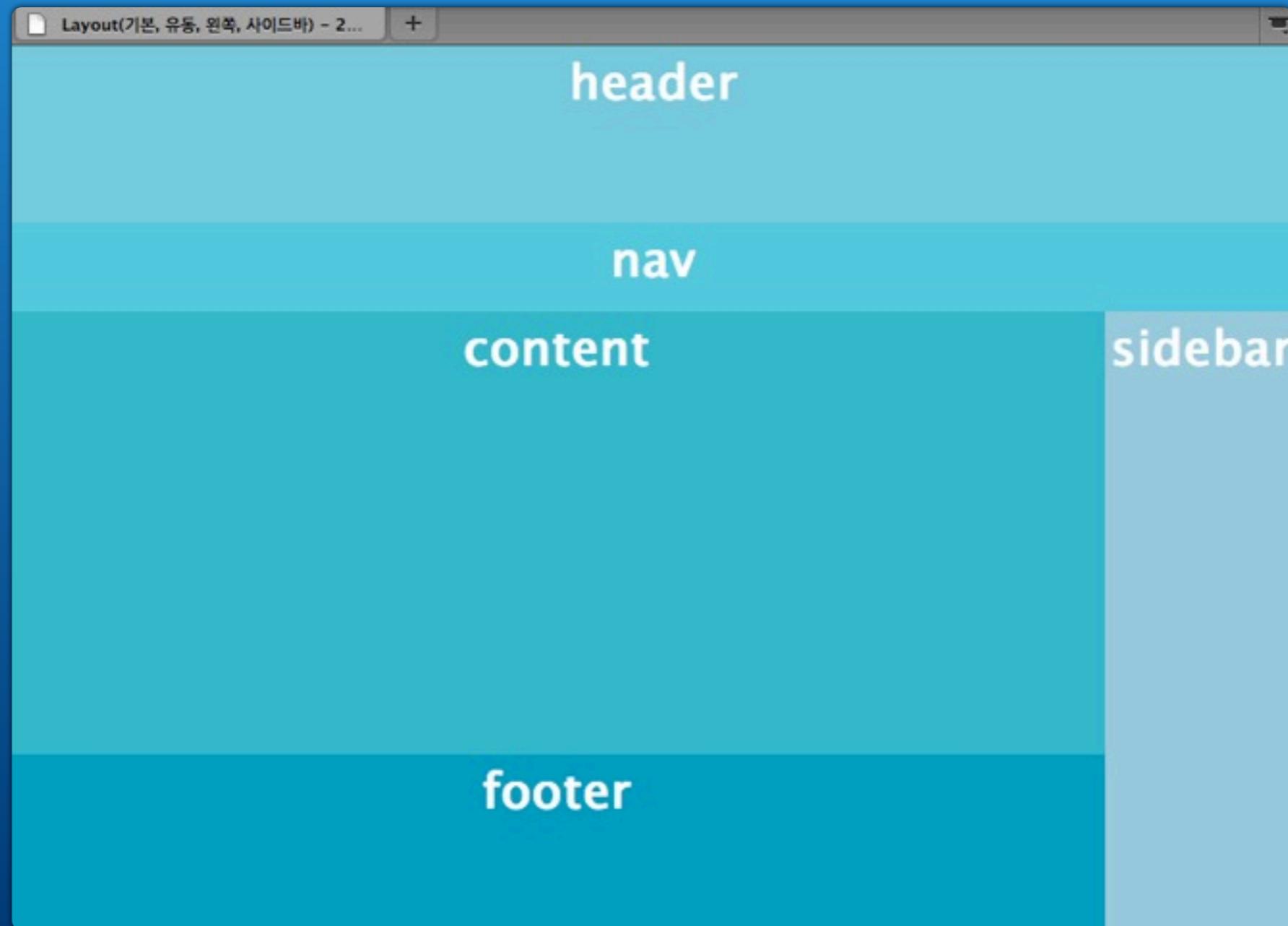
web standard project



유동 2단 우측-A 주요 컨텐츠 레이아웃

# Lay-out Design

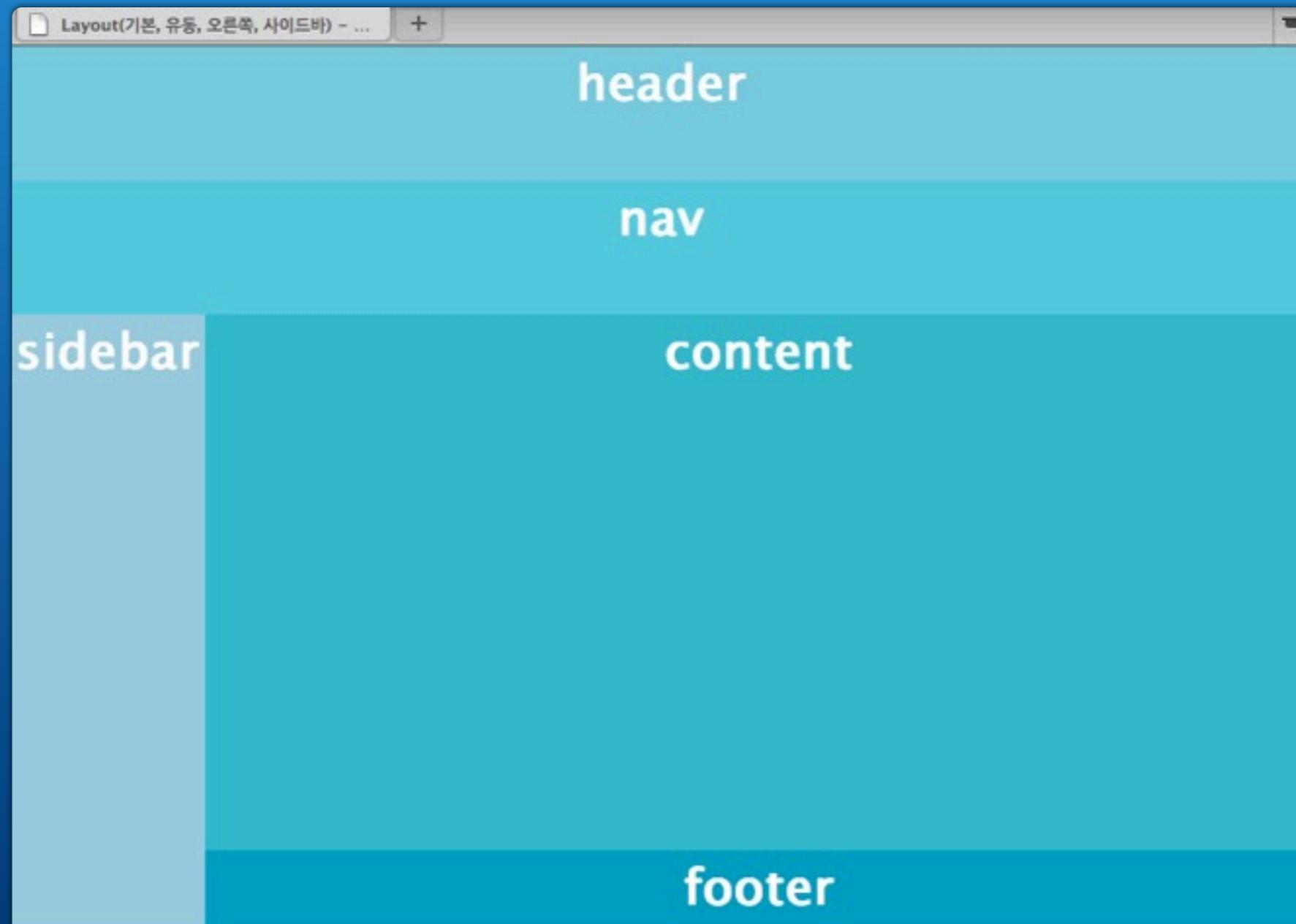
web standard project



유동 2단 좌측-B 주요 컨텐츠 레이아웃

# Lay-out Design

web standard project

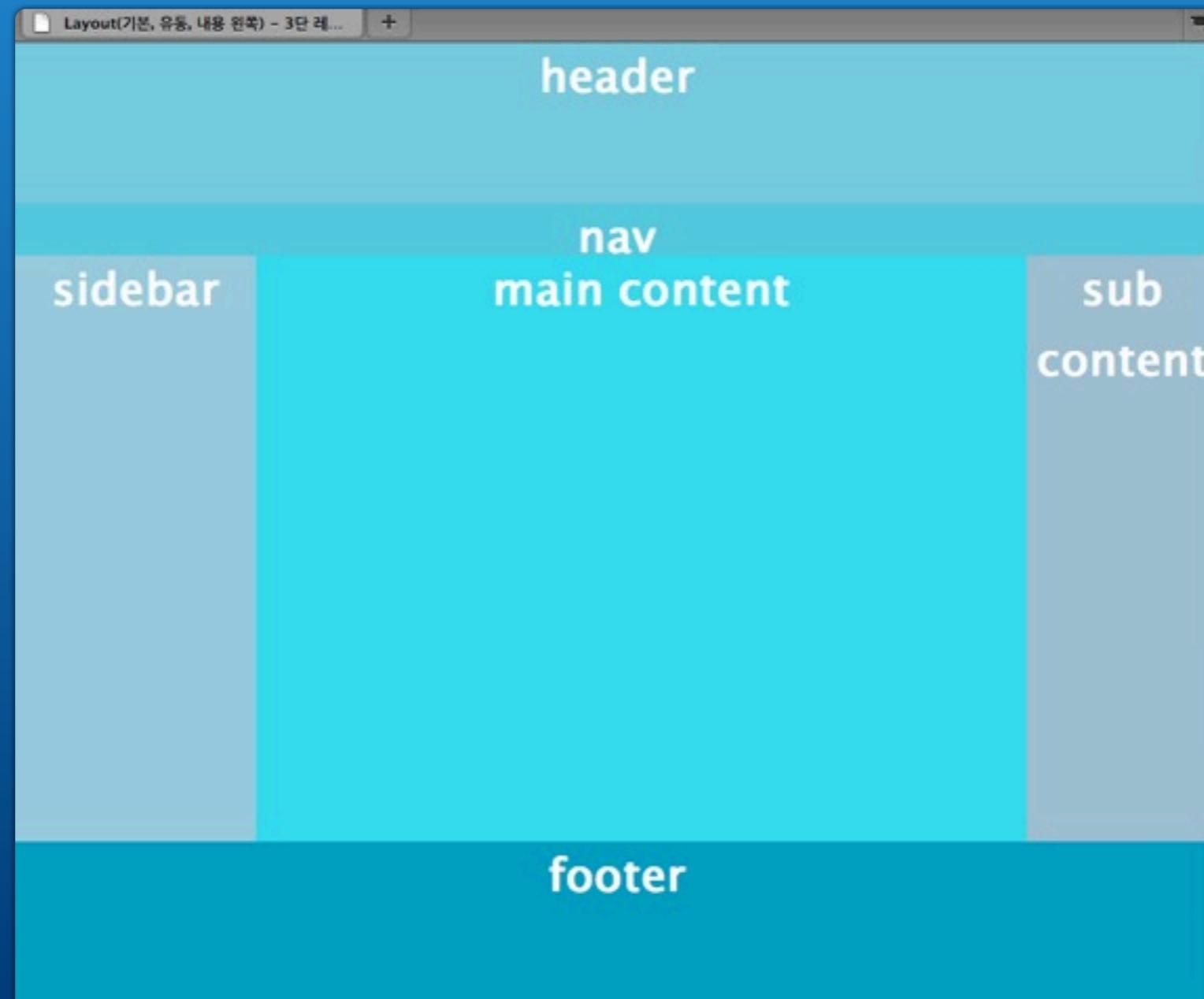


FOOTER

유동 2단 우측-B 주요 컨텐츠 레이아웃

# Lay-out Design

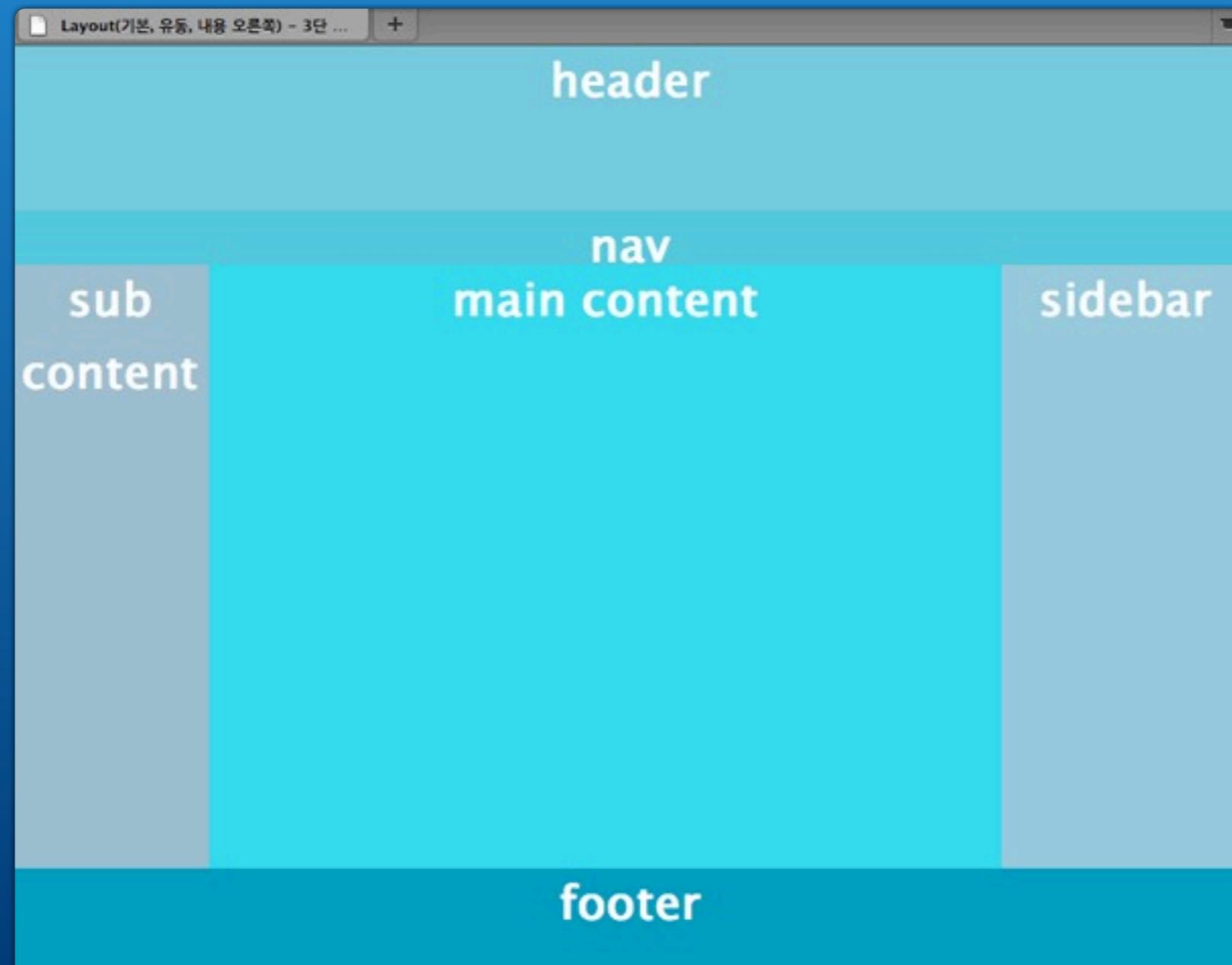
web standard project



유동 3단 중앙-A 주요 컨텐츠 레이아웃

# Lay-out Design

web standard project





# Lay-out Design

w e b s t a n d a r d p r o j e c t

유형	단단	주요 컨텐츠 위치
고정 (px)	1단	.
고정 (px)	2단	왼쪽-A
고정 (px)	2단	오른쪽-A
고정 (px)	2단	왼쪽-B
고정 (px)	2단	오른쪽-B
고정 (px)	3단	가운데-A
고정 (px)	3단	가운데-B

# Lay-out Design

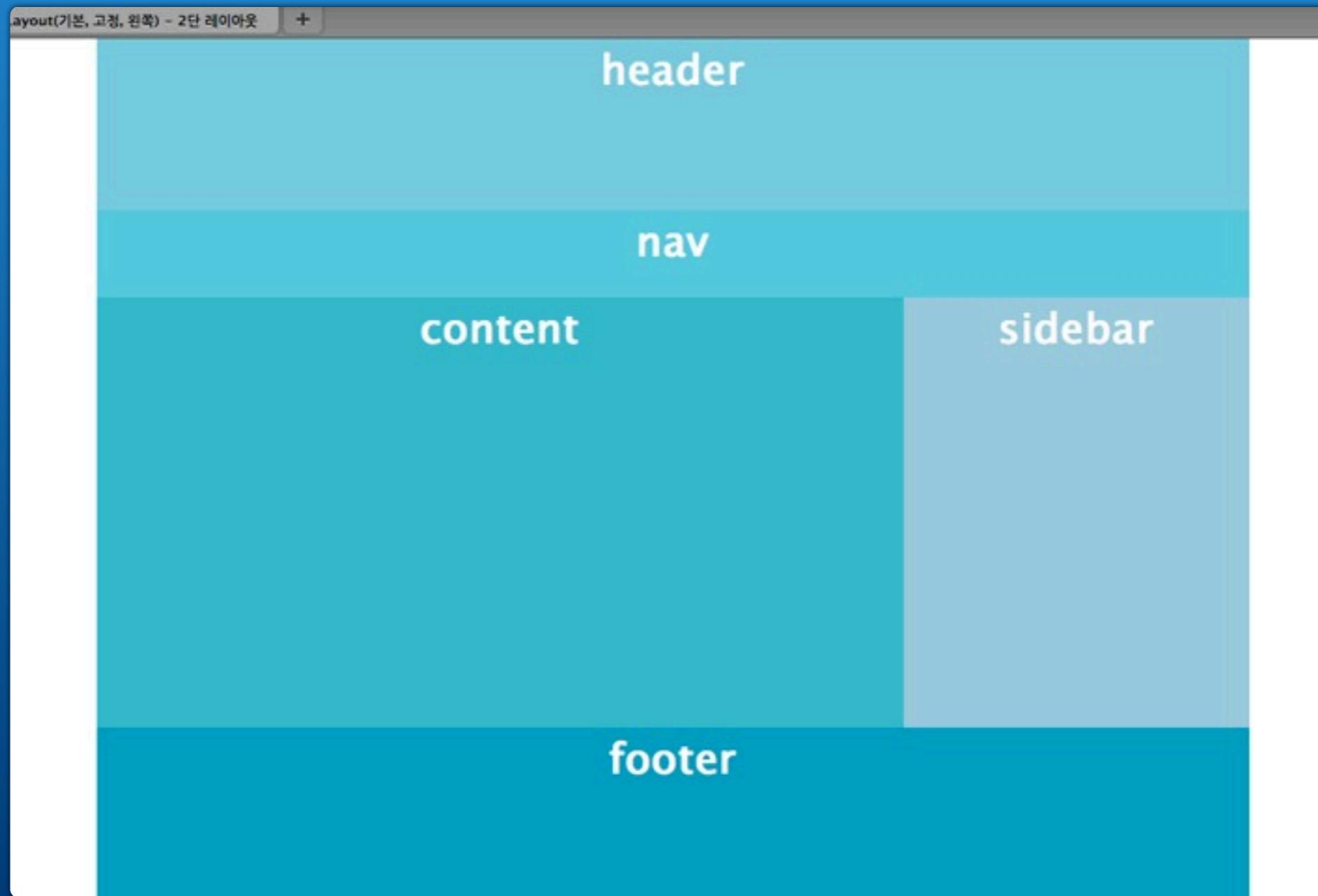
web standard project



고정 1단 레이아웃

# Lay-out Design

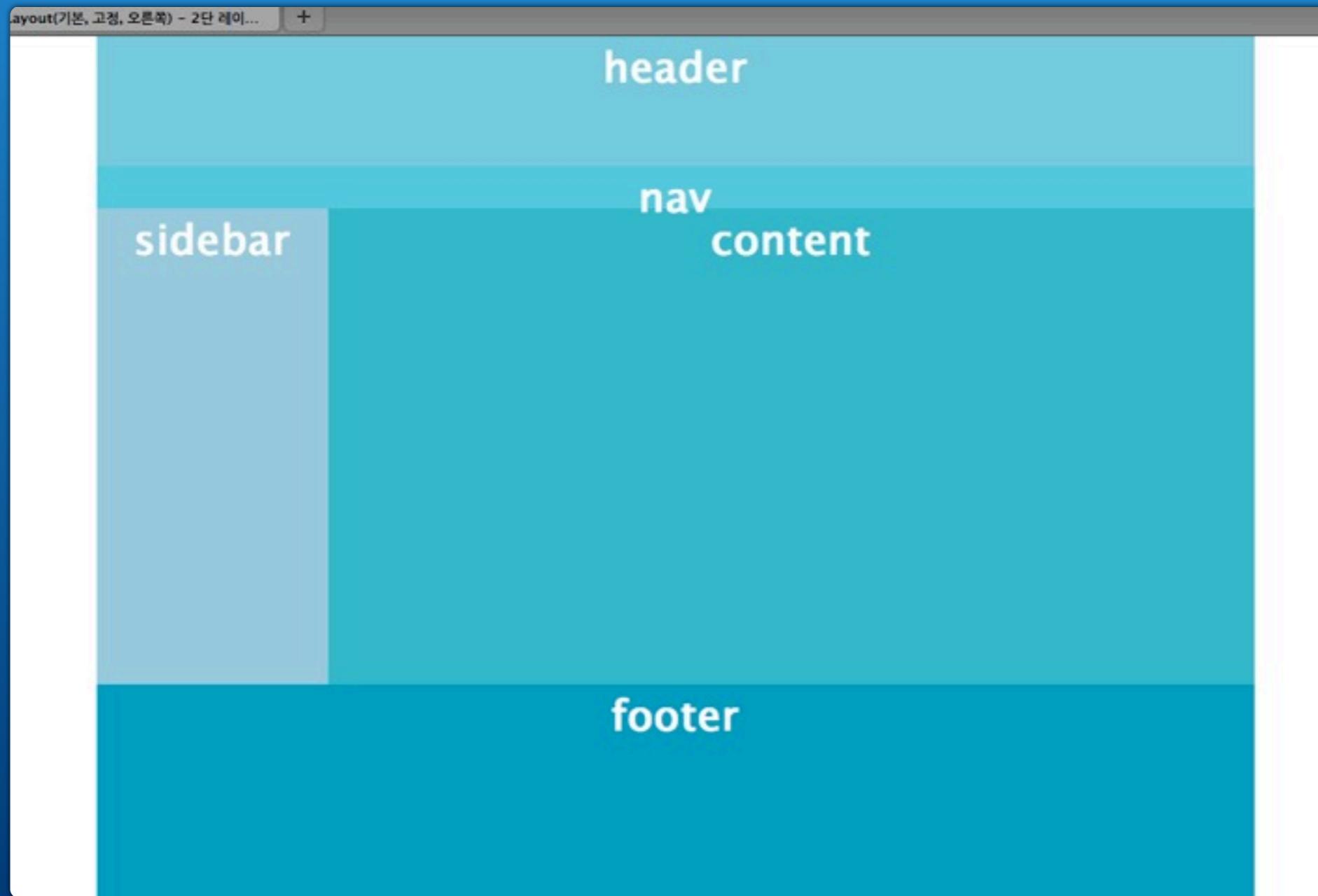
web standard project



고정 2단 좌측-A 주요 컨텐츠 레이아웃  
layout

# Lay-out Design

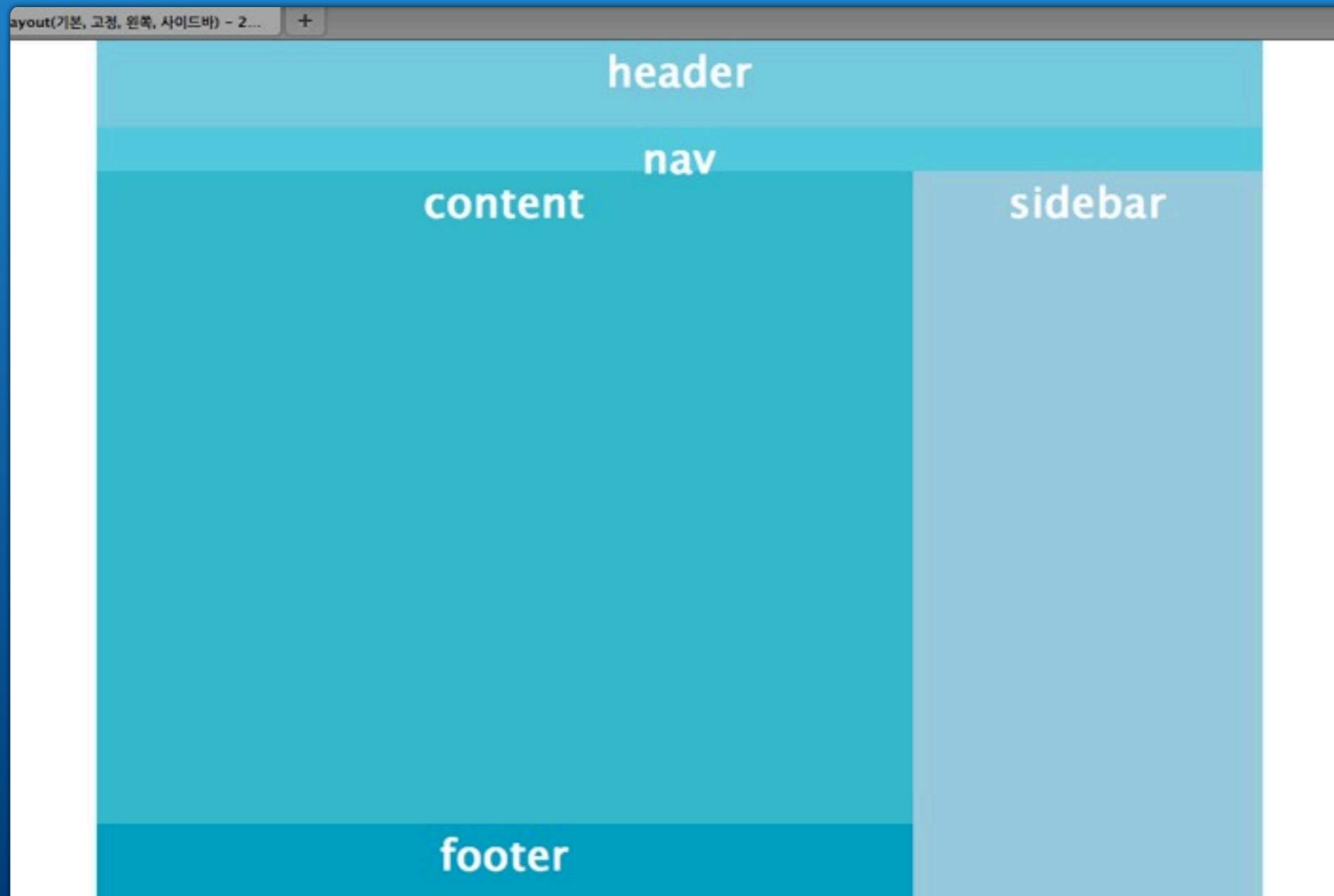
web standard project



고정 2단 우측-A 주요 컨텐츠 레이아웃

# Lay-out Design

web standard project



고정 2단 좌측-B 주요 컨텐츠 레이아웃  
FOOTER

# Lay-out Design

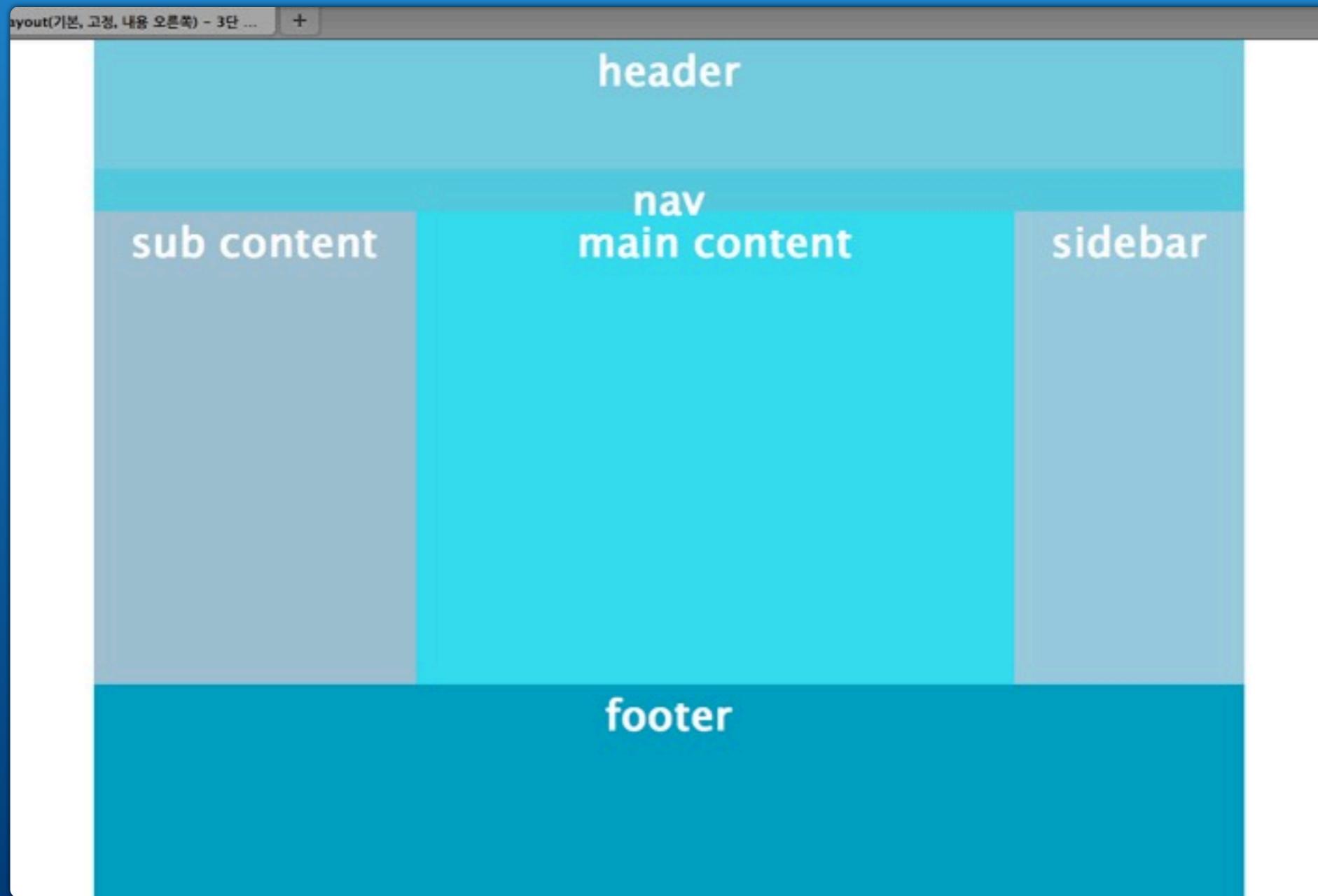
web standard project



고정 2단 우측-B 주요 컨텐츠 레이아웃

# Lay-out Design

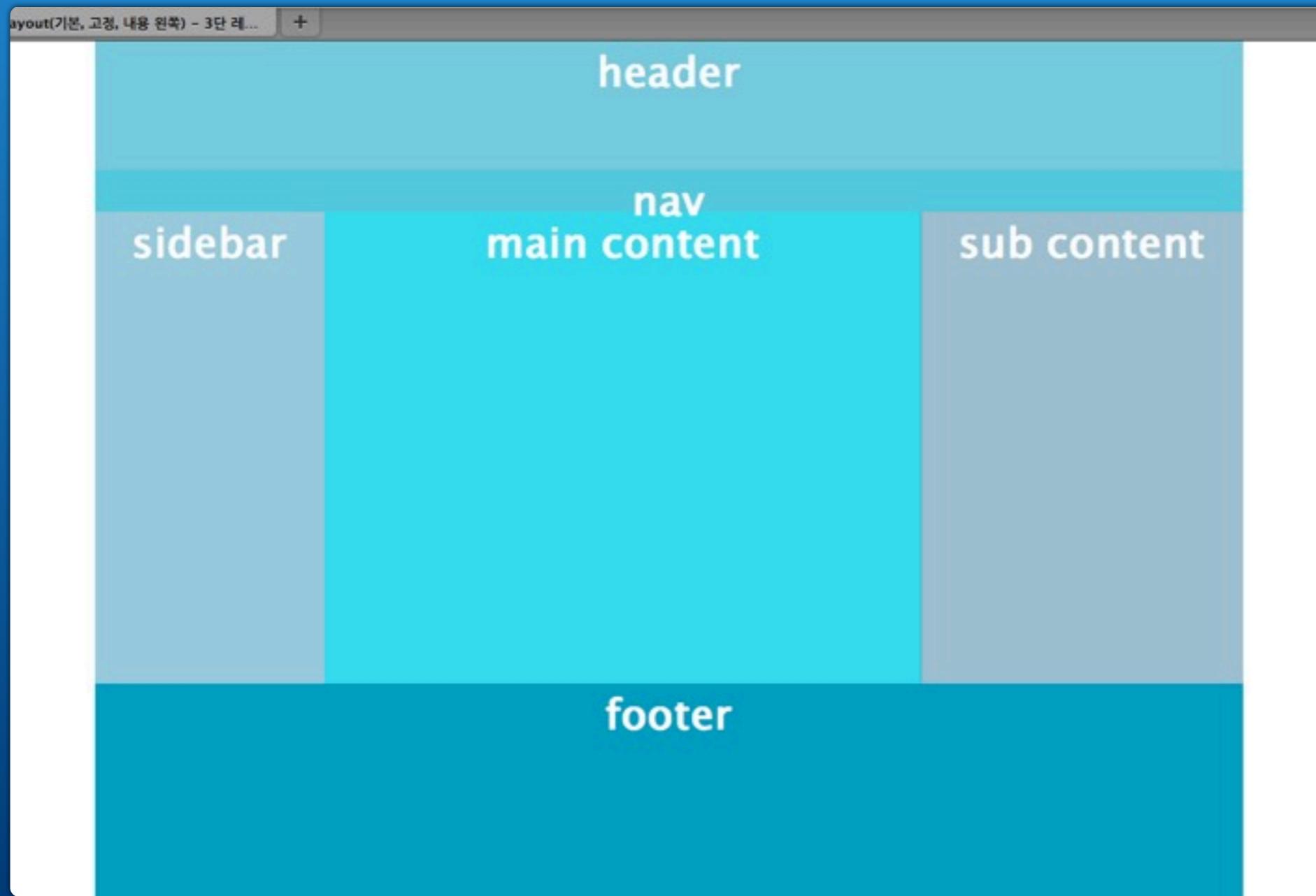
web standard project



고정 3단 중앙-A 주요 컨텐츠 레이아웃

# Lay-out Design

web standard project



고정 3단 중앙-B 주요 컨텐츠 레이아웃



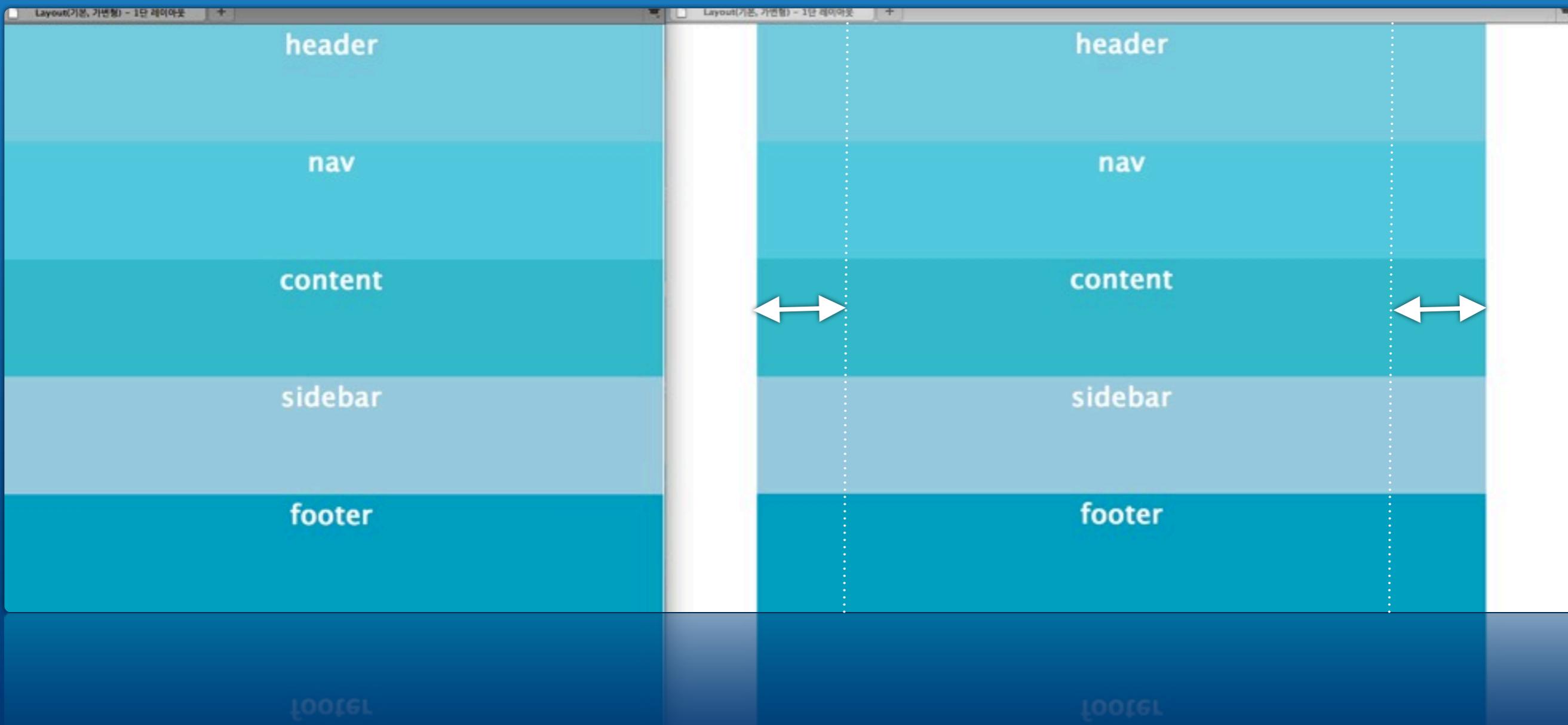
# Lay-out Design

web standard project

유형	단계	주요 컨텐츠 위치
가변 (%), px	1단	.
가변 (%), px	2단	왼쪽-A
가변 (%), px	2단	오른쪽-A
가변 (%), px	2단	왼쪽-B
가변 (%), px	2단	오른쪽-B
가변 (%), px	3단	가운데-A
가변 (%), px	3단	가운데-B

# Lay-out Design

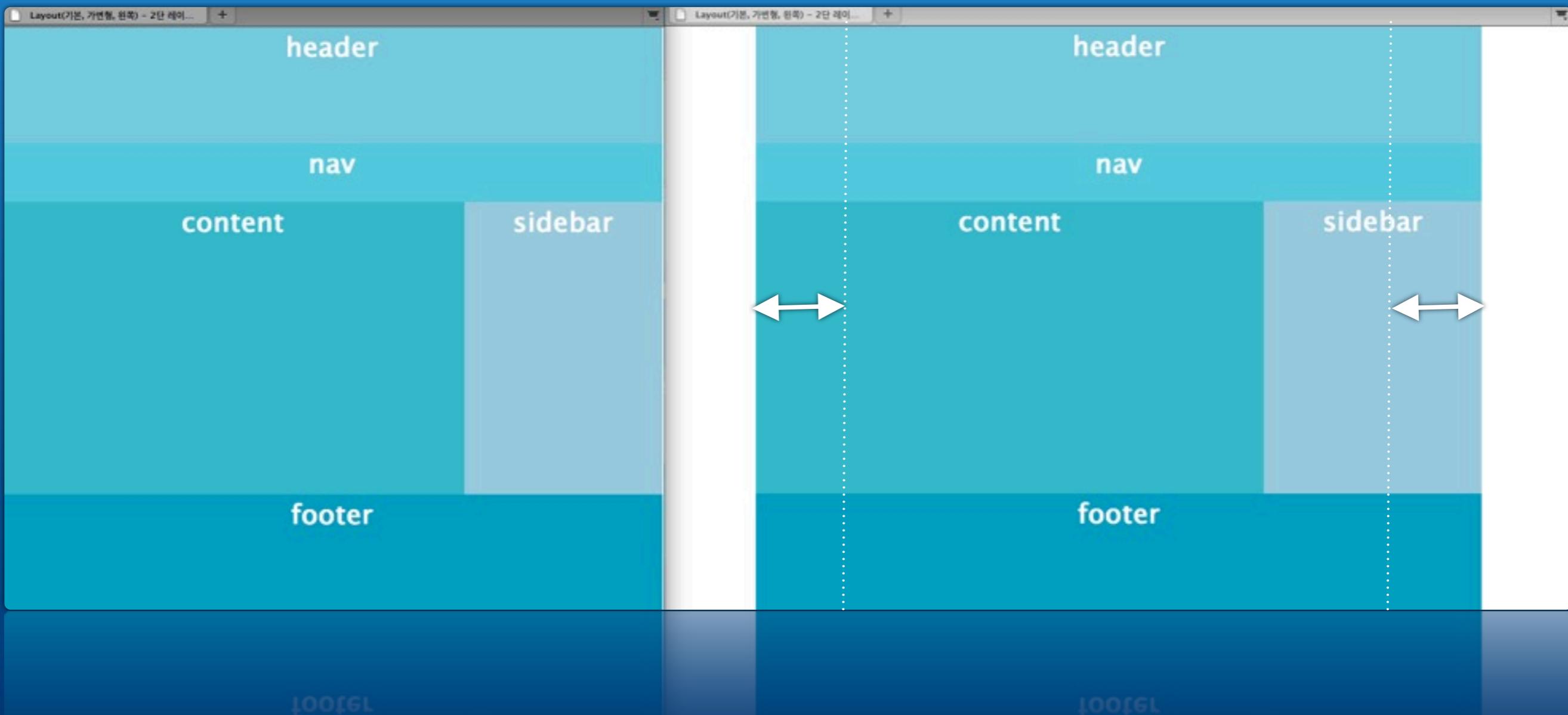
web standard project



가변 1단 레이아웃

# Lay-out Design

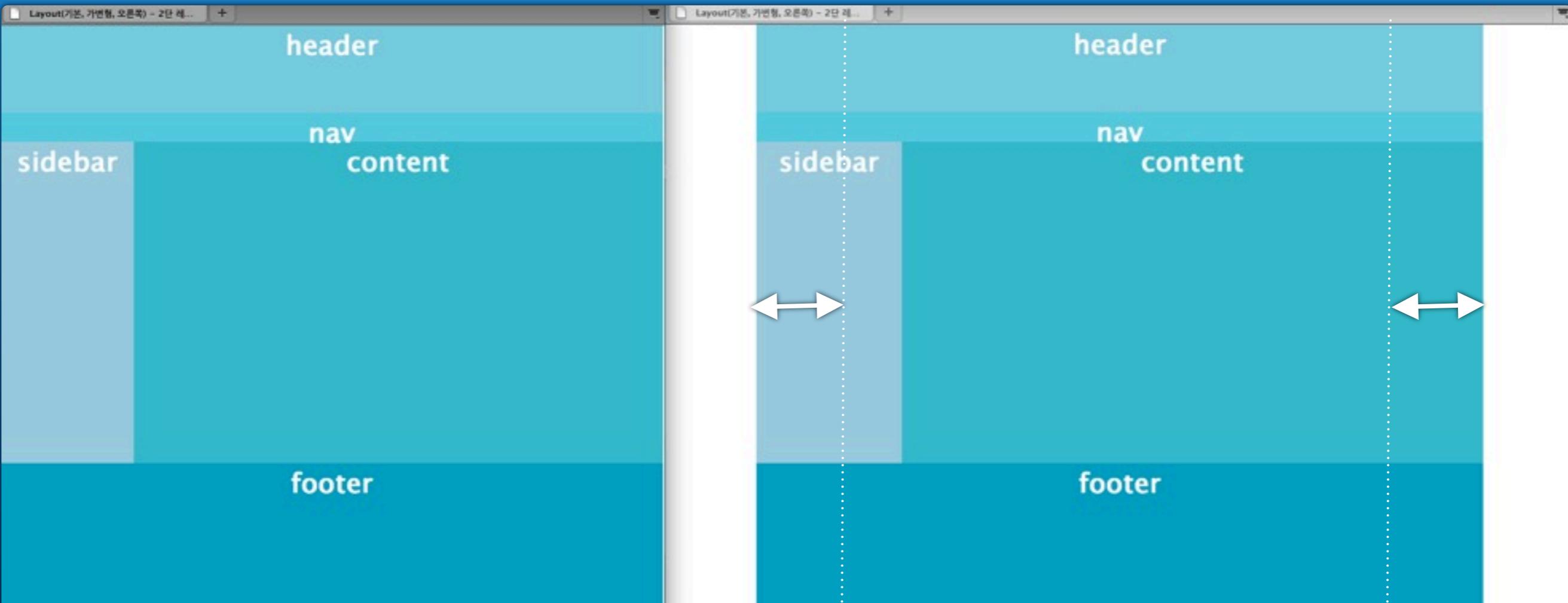
web standard project



가변 2단 좌측-A 주요 컨텐츠 레이아웃

# Lay-out Design

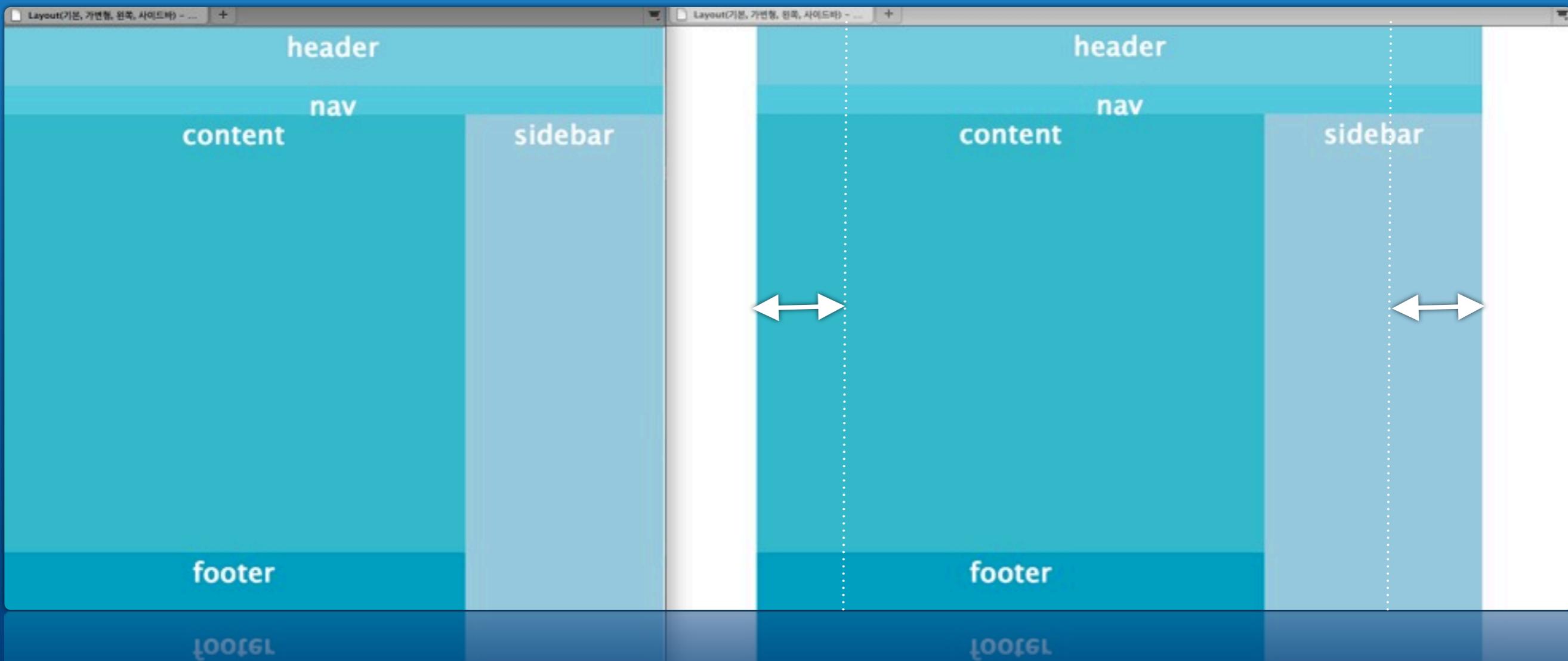
web standard project



가변 2단 우측-A 주요 컨텐츠 레이아웃

# Lay-out Design

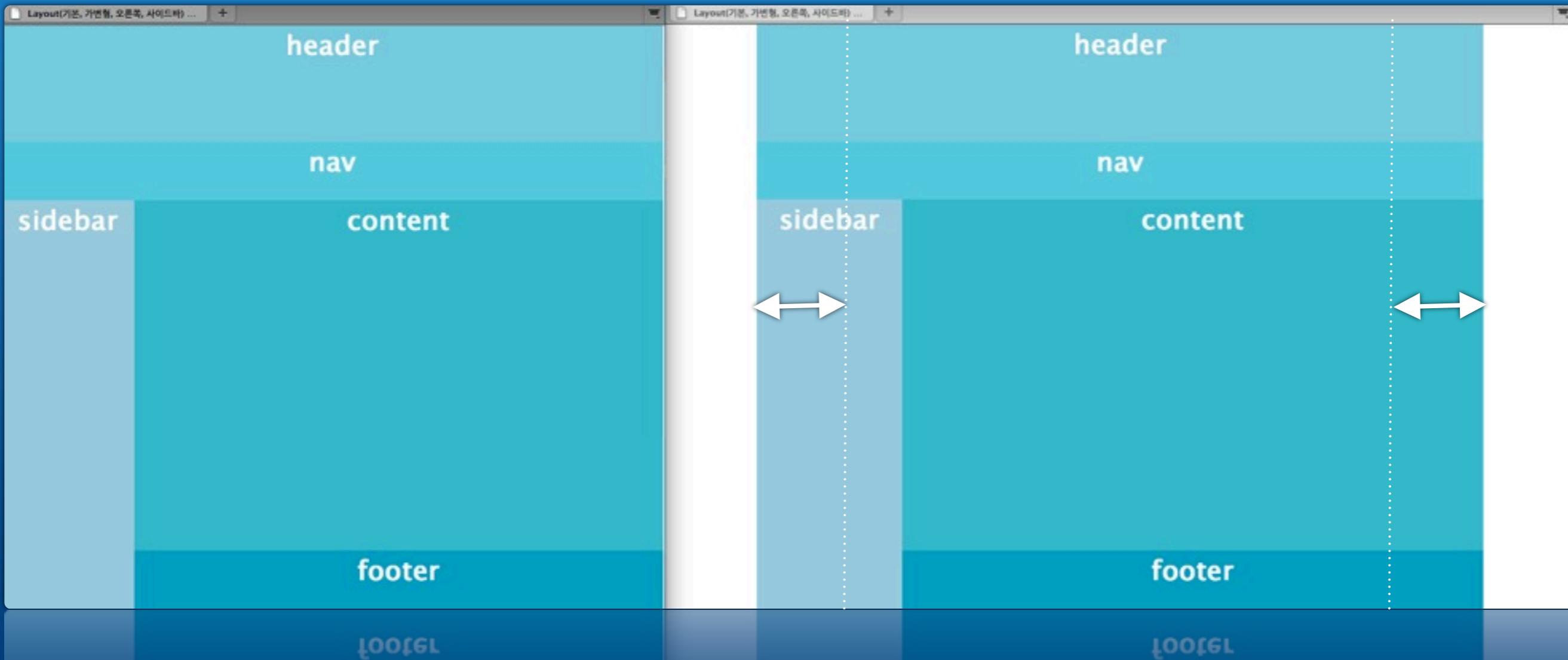
web standard project



가변 2단 좌측-B 주요 컨텐츠 레이아웃

# Lay-out Design

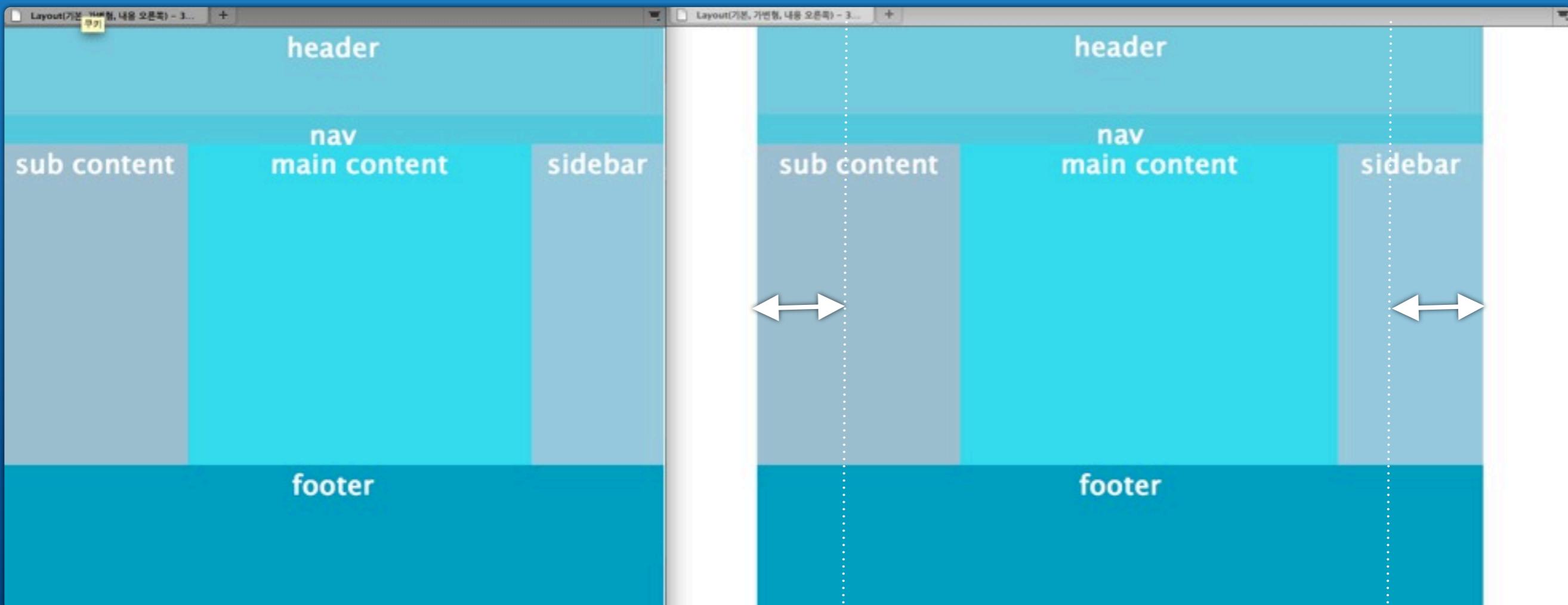
web standard project



가변 2단 우측-B 주요 컨텐츠 레이아웃

# Lay-out Design

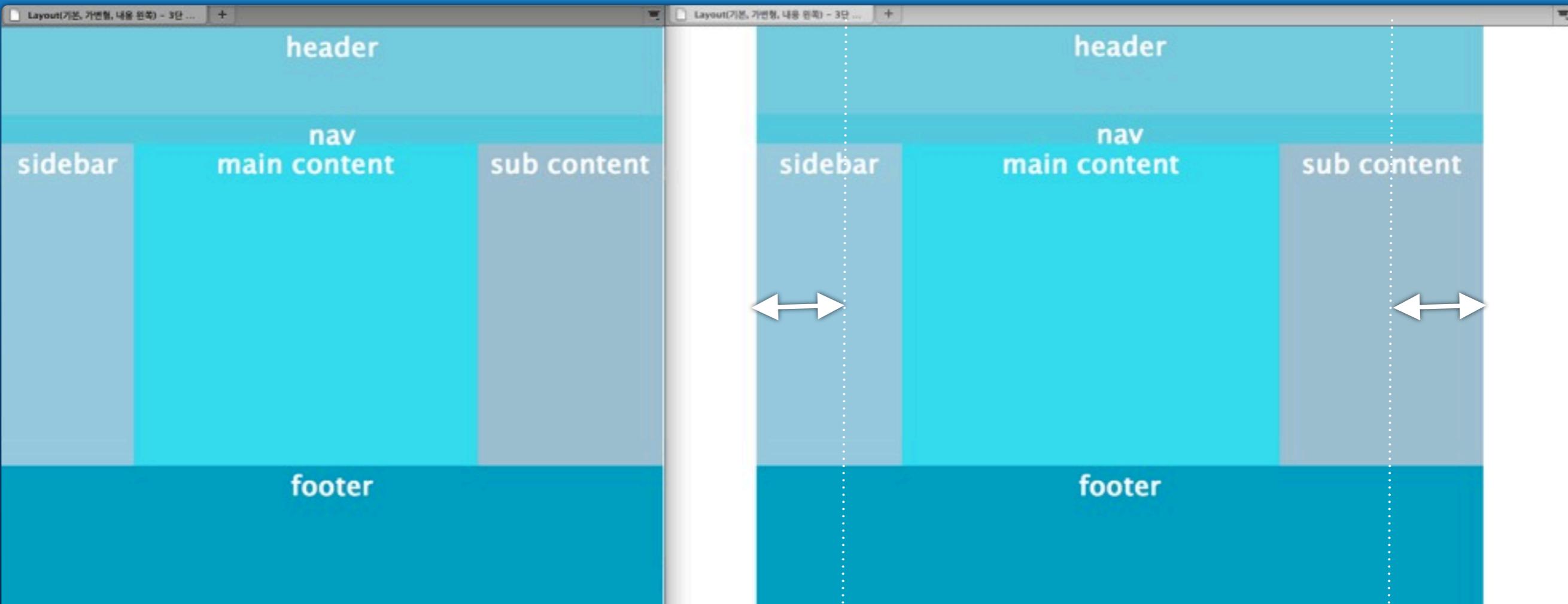
web standard project



가변 3단 중앙-A 주요 컨텐츠 레이아웃

# Lay-out Design

web standard project



가변 3단 중앙-B 주요 컨텐츠 레이아웃

CSS 포지션(Position) 설정은  
레이아웃 디자인 철학의 진정한 전승자입니다!

float는 분명 뛰어나지.

하지만 든든한 부하는 float 만이

아니야. position이라는 네이밍을

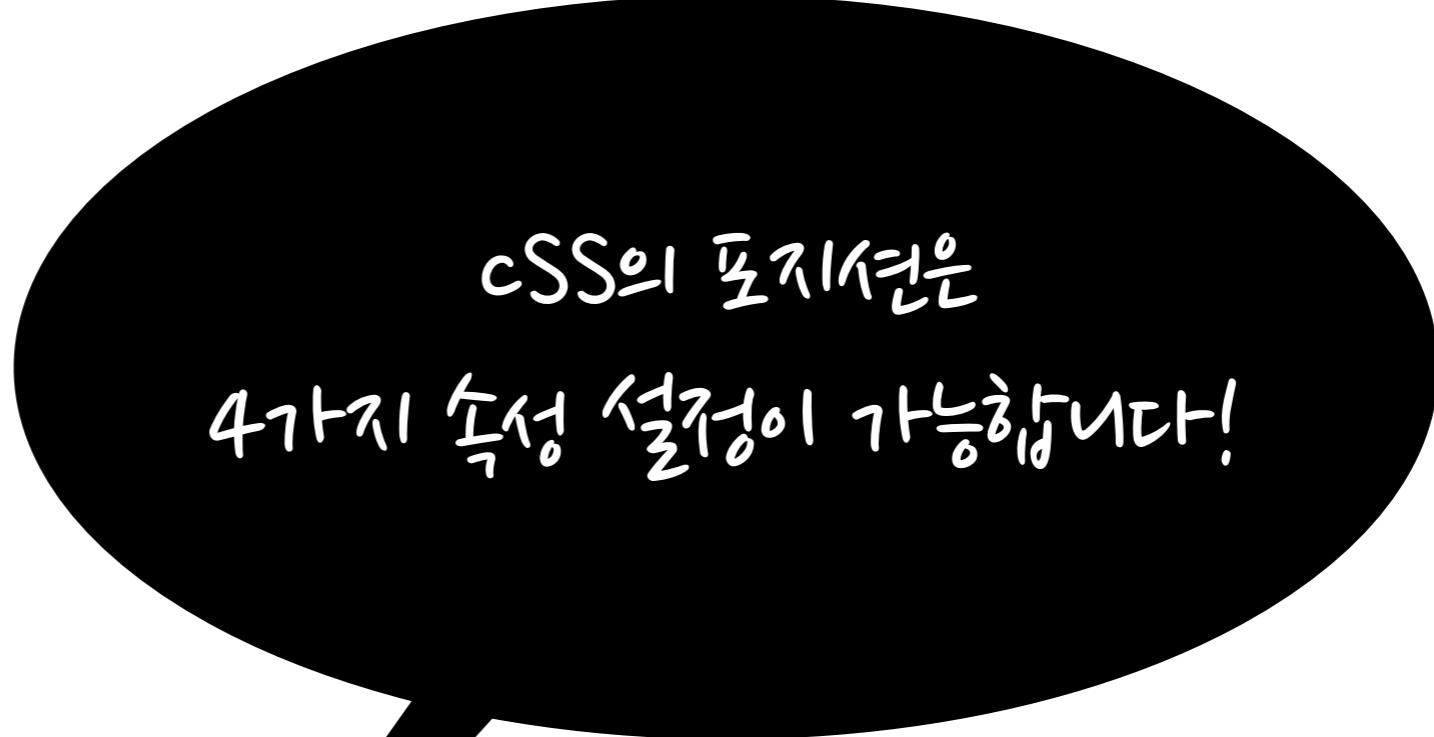
소개하지!!



# Layout CSS

스타일시트 레이아웃

- static
- relative
- absolute
- fixed

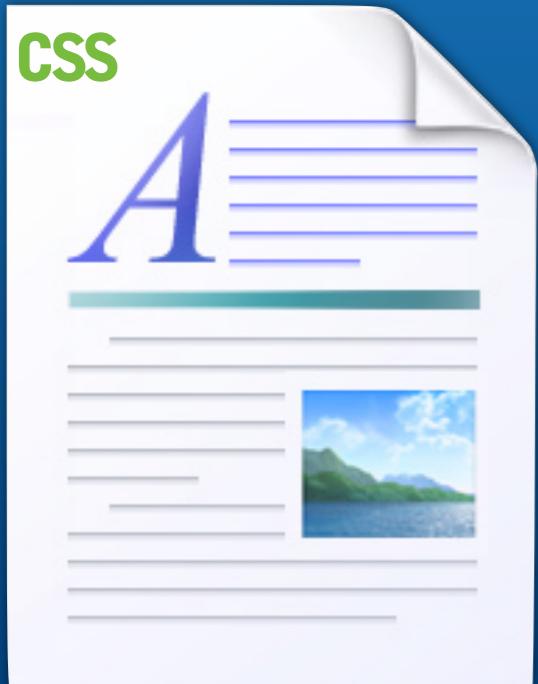


cSS의 포지션은  
4가지 속성 설정이 가능합니다!





# CSS언어를 이용하여 대상의 위치지정을 해볼까요?



Style code

대상 속성 값  
selector {position: value}

<head>



```
<style> div {position: relative; left: 3em} </style>  
</head>
```



# CSS언어를 이용하여 대상의 위치지정을 해볼까요?





# CSS언어를 이용하여 대상의 위치지정을 해볼까요?





# CSS언어를 이용하여 대상의 위치지정을 해볼까요?



IE6는 position:fixed;를 지원하지 않습니다.

Style code

대상 속성 값  
selector {position: value}

relative

absolute

fixed

```
<head> ▲  
    <style> div {position: relative; left: 3em} </style>  
</head>
```



# CSS언어를 이용하여 대상의 위치지정을 해볼까요?



IE6는 position:fixed;를 지원하지 않습니다.

Style code

대상 속성 값  
selector {position: value}

relative

absolute

fixed

static

<head>  
    <style> div {position: relative; left: 3em} </style>  
  </head>



# CSS언어를 이용하여 대상의 위치지정을 해볼까요?



IE6는 position:fixed;를 지원하지 않습니다.

Style code

대상 속성 값  
selector {position: value}

<head>



<style> div {position: relative; left: 3em; top: 1em;}</style>  
</head>

relative

absolute

fixed

static

inherit





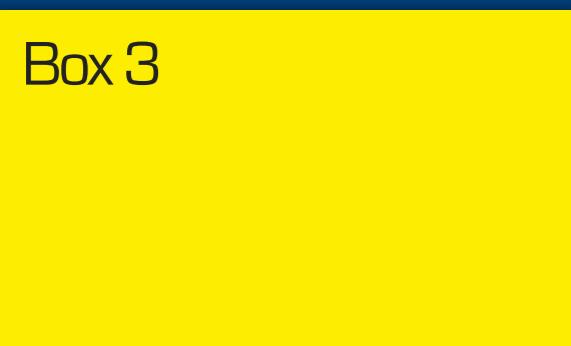
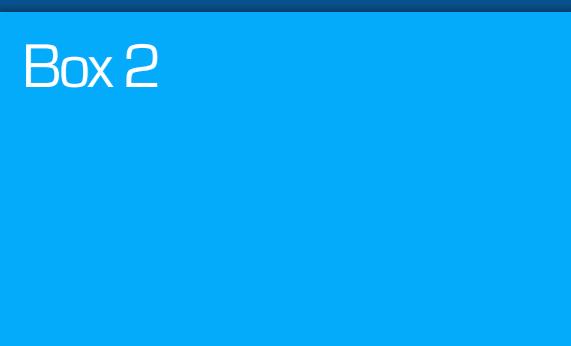
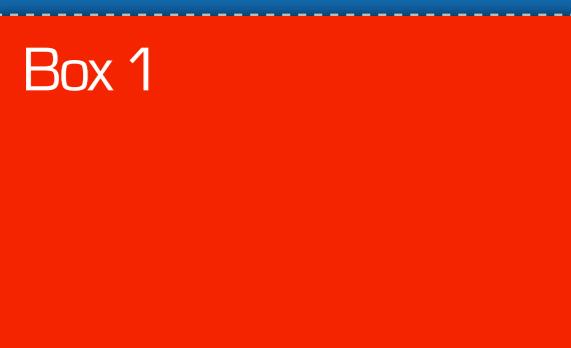
# CSS언어를 이용하여 대상의 위치지정을 해볼까요?





# CSS Position Model

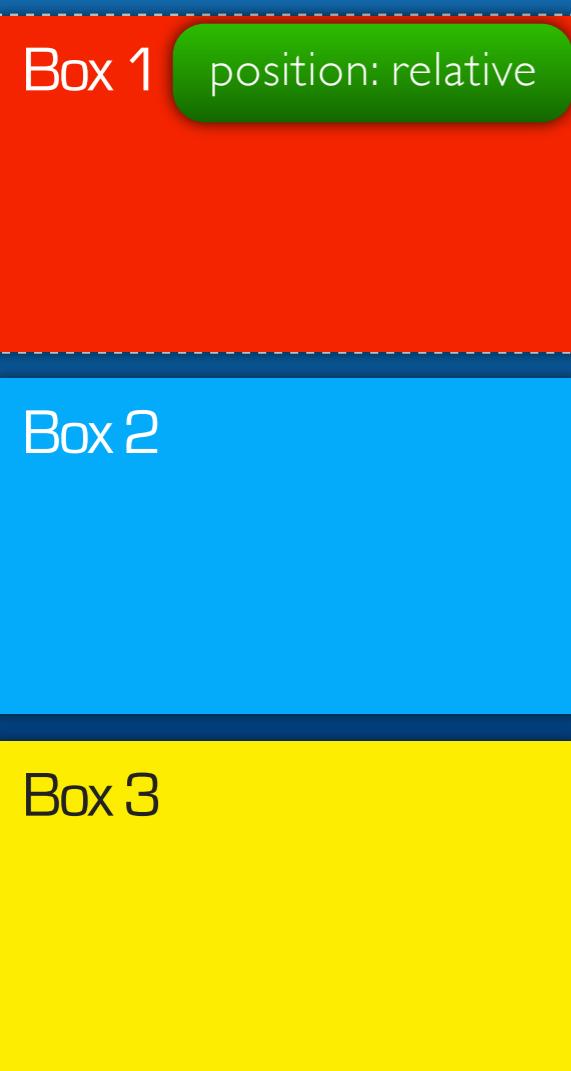
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

Box 1    position: relative  
          top: 18px  
          left: 24px

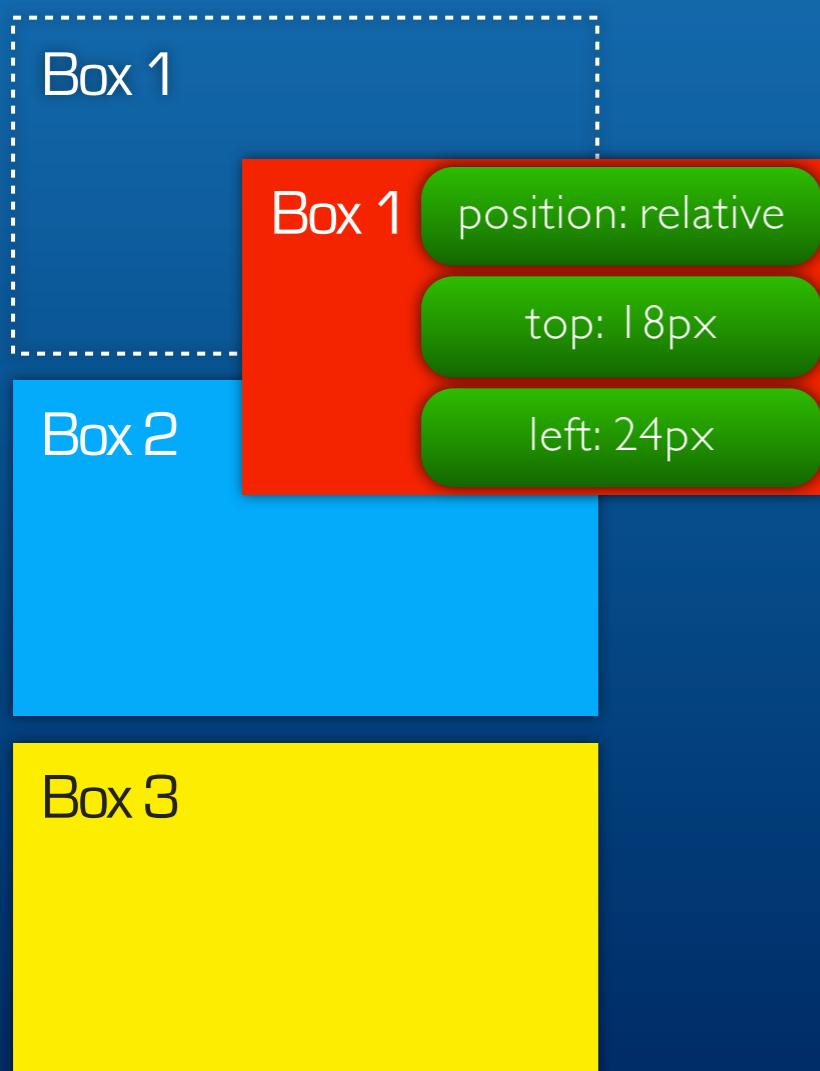
Box 2

Box 3



# CSS Position Model

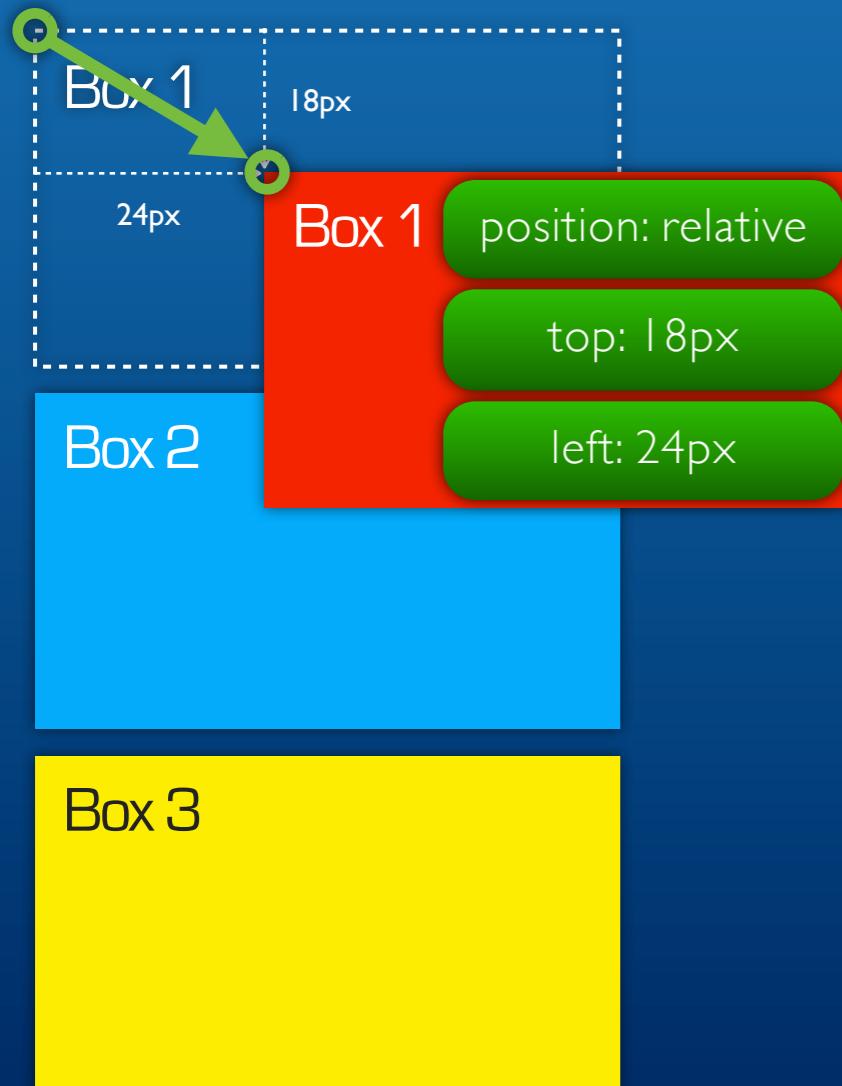
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



상대좌표 이동  
position: **relative**

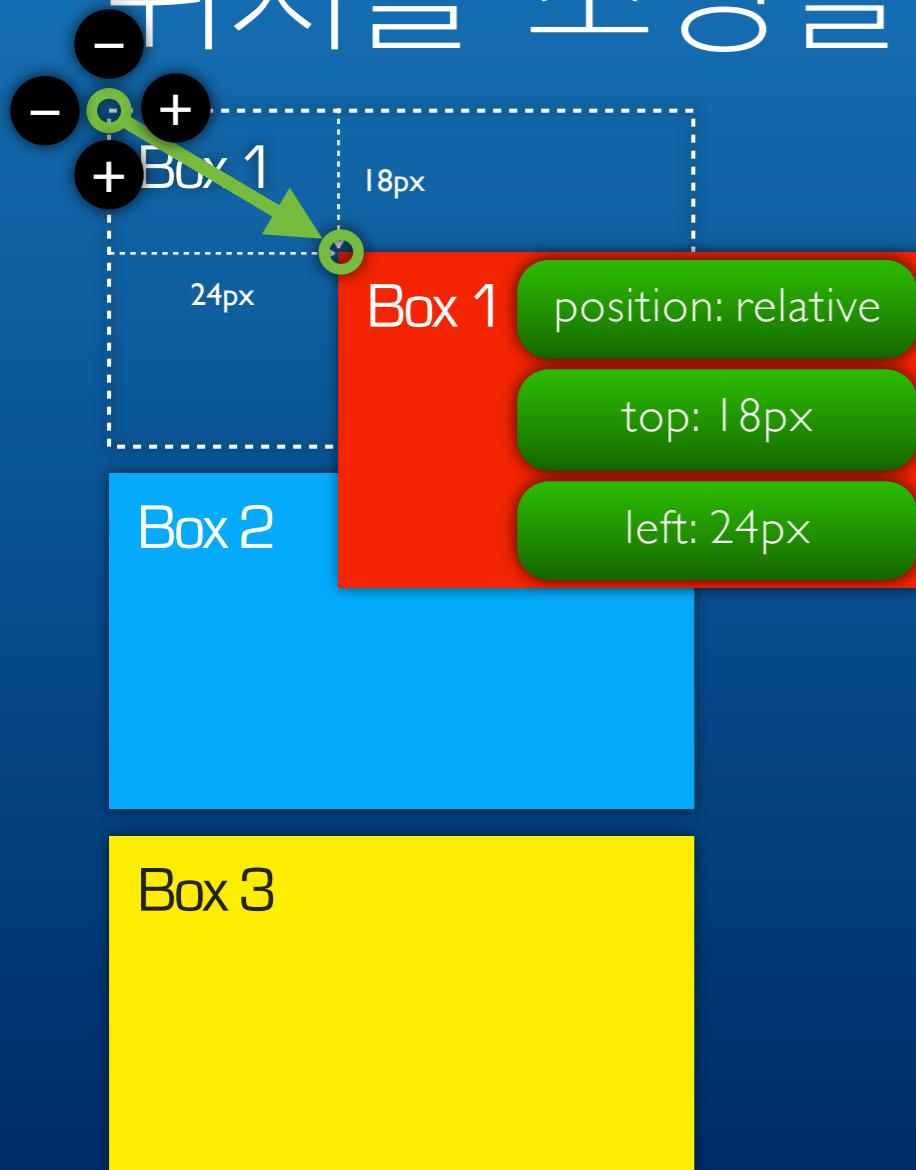
자신의 원래 위치로부터  
상대적인 거리로 이동합니다.

좌측상단을 기점으로  
시작점으로 이동



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



상대좌표 이동  
position: **relative**

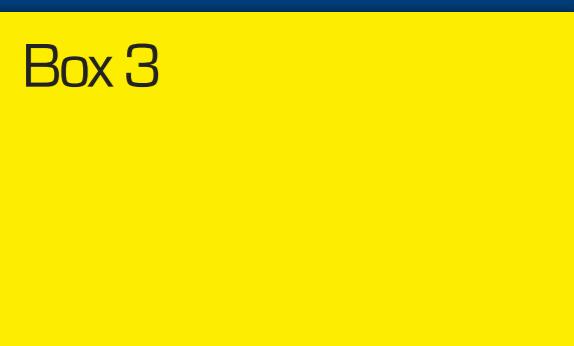
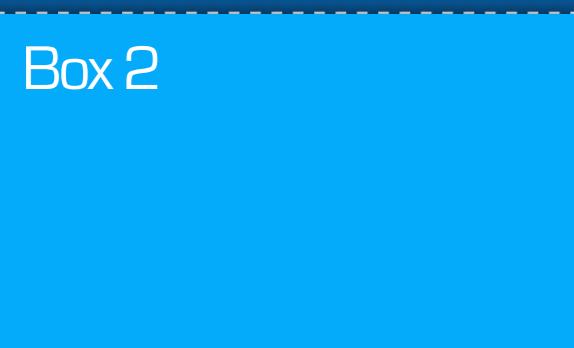
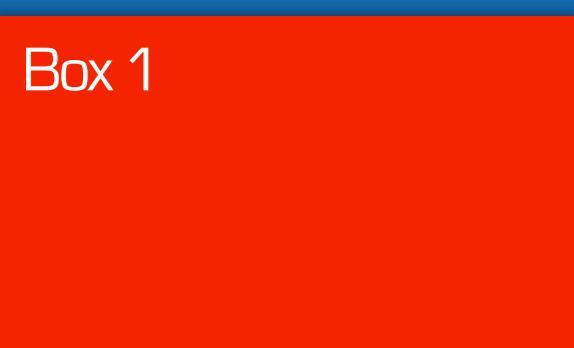
자신의 원래 위치로부터  
상대적인 거리로 이동합니다.

좌측상단을 기점으로  
시작점으로 이동



# CSS Position Model

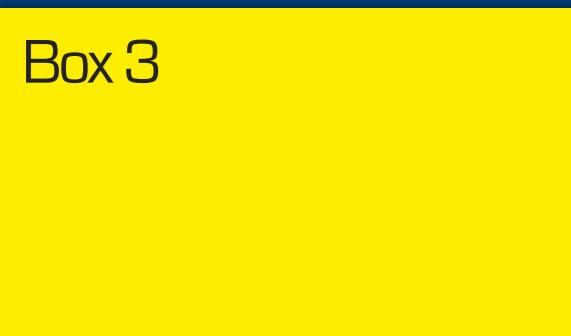
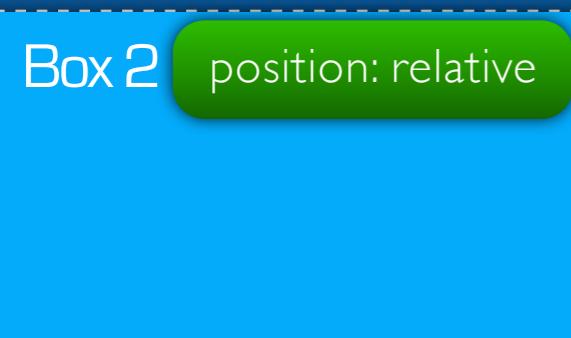
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

Box 1

Box 2

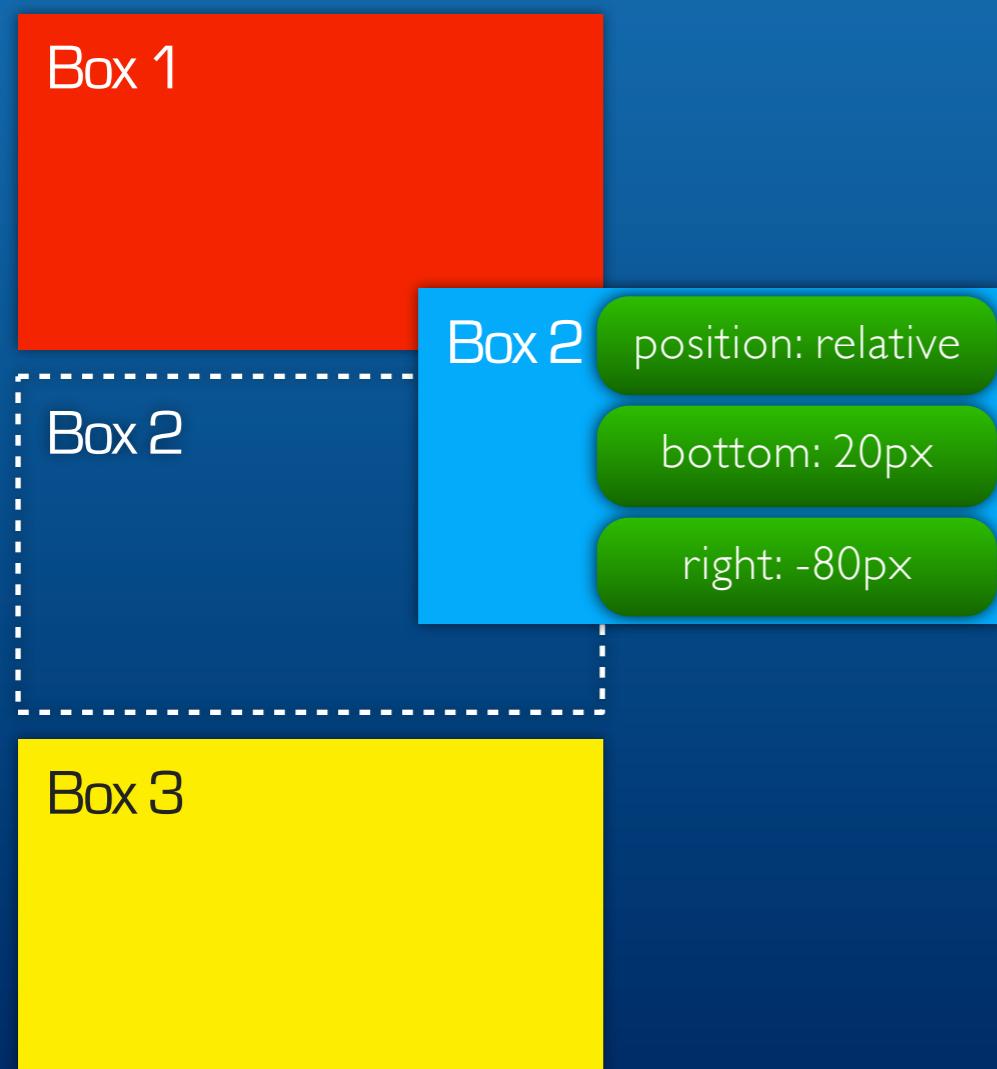
```
position: relative  
bottom: 20px  
right: -80px
```

Box 3



# CSS Position Model

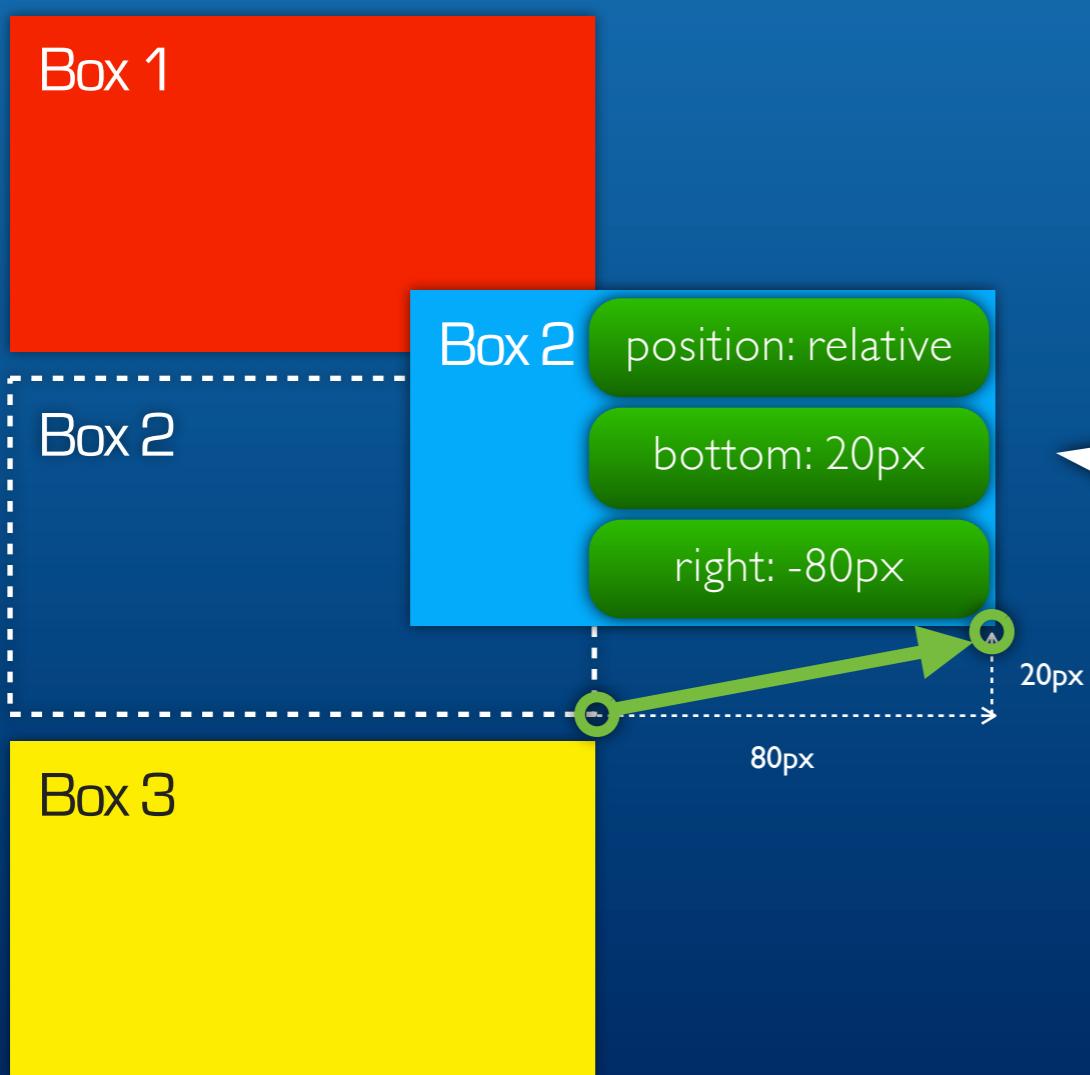
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



## 상대좌표 이동

position: **relative**

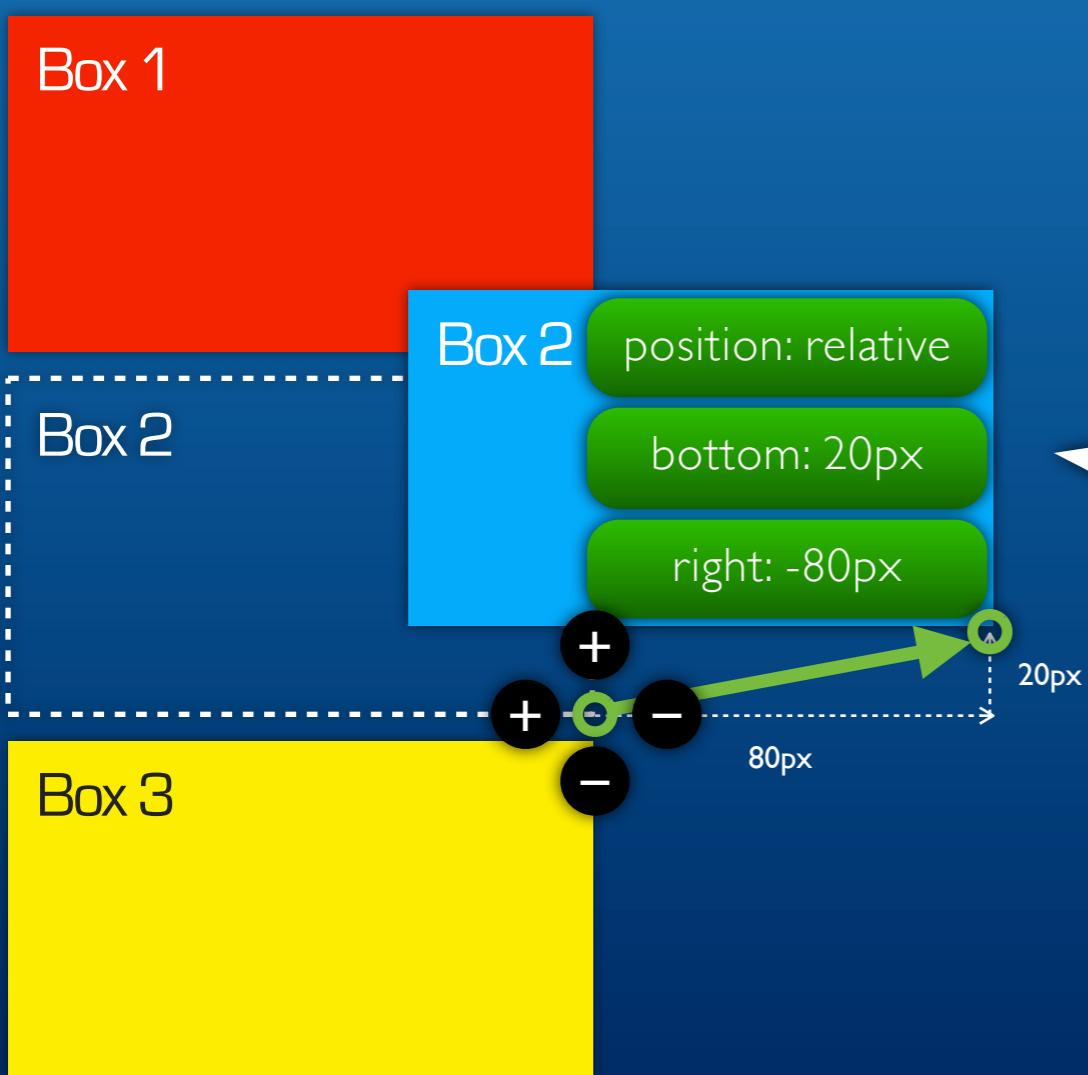
자신의 원래 위치로부터  
상대적인 거리로 이동합니다.

우측하단을 기점으로  
시작점으로 이동



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



## 상대좌표 이동

position: **relative**

자신의 원래 위치로부터  
상대적인 거리로 이동합니다.

우측하단을 기점으로  
시작점으로 이동



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

html

Box 1 position: absolute

Box 2

Box 3



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

html

```
Box 1 position: absolute  
bottom: 20px  
right: 100px
```

Box 2

Box 3

The diagram shows a black rectangular container with three colored boxes inside. Box 1 is red and has a blue rounded rectangle overlay with white text showing its CSS properties: 'position: absolute', 'bottom: 20px', and 'right: 100px'. Box 2 is a solid blue rectangle positioned below Box 1. Box 3 is a solid yellow rectangle positioned below Box 2. The text 'html' is located in the top right corner of the black container.



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

html

Box 2

Box 1

Box 3

Box 1

position: absolute  
bottom: 20px  
right: 100px



# CSS Position Model

CSS position 속성을 사용하면 대상의 위치를 조정할 수 있습니다.

## 절대좌표 이동

position: absolute

자신의 부모 위치로부터  
상대적인 거리로 이동합니다.

우측하단을 기점으로  
시작점으로 이동





# CSS Position Model

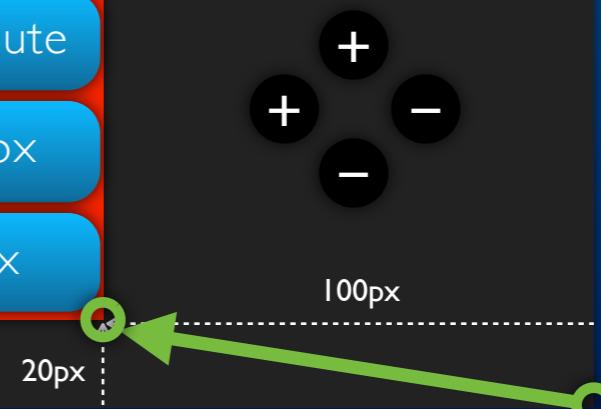
CSS position 속성을 사용하면 대상의 위치를 조정할 수 있습니다.

## 절대좌표 이동

position: absolute



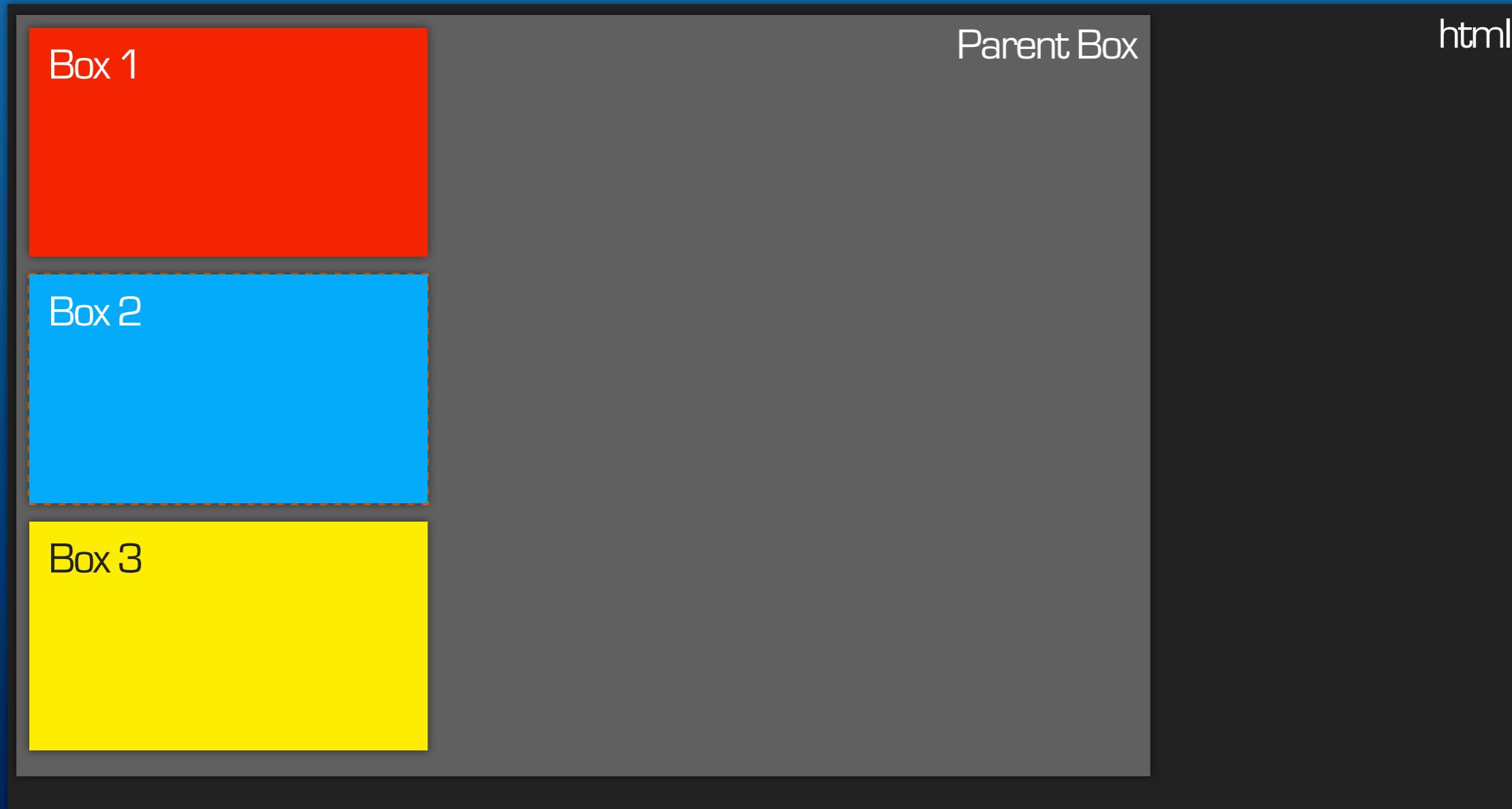
Box 1 position: absolute  
bottom: 20px  
right: 100px





# CSS Position Model

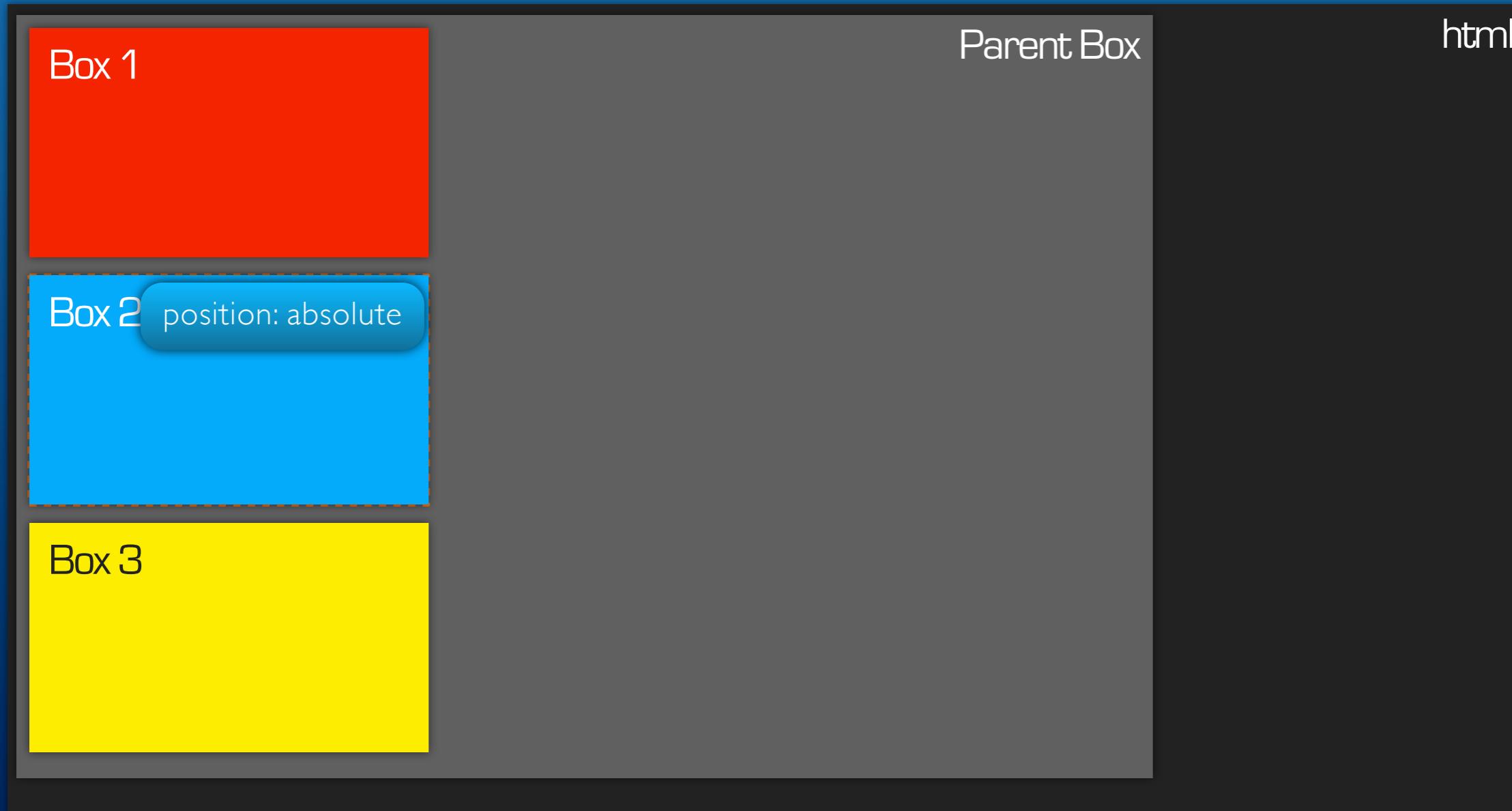
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

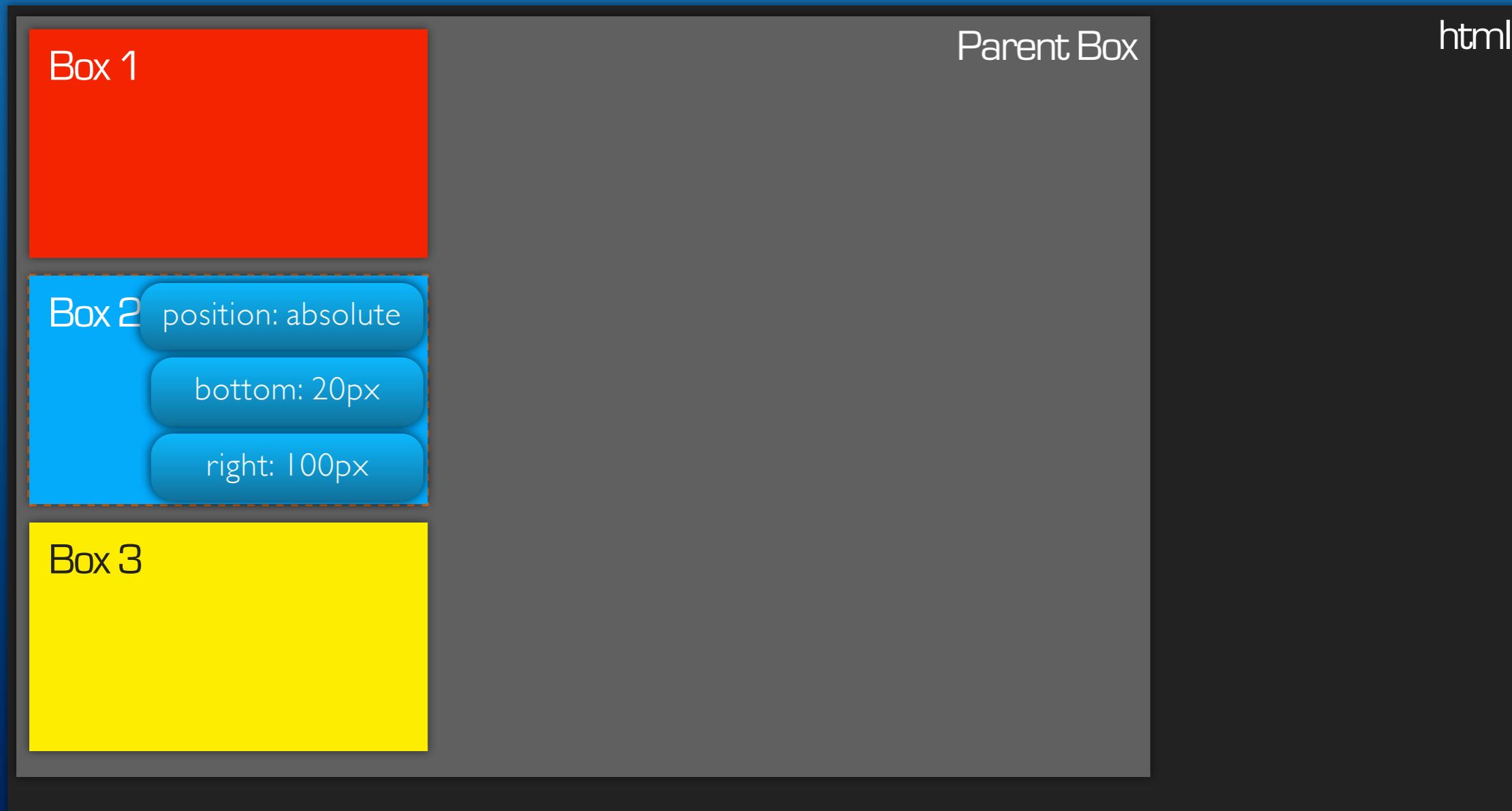
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

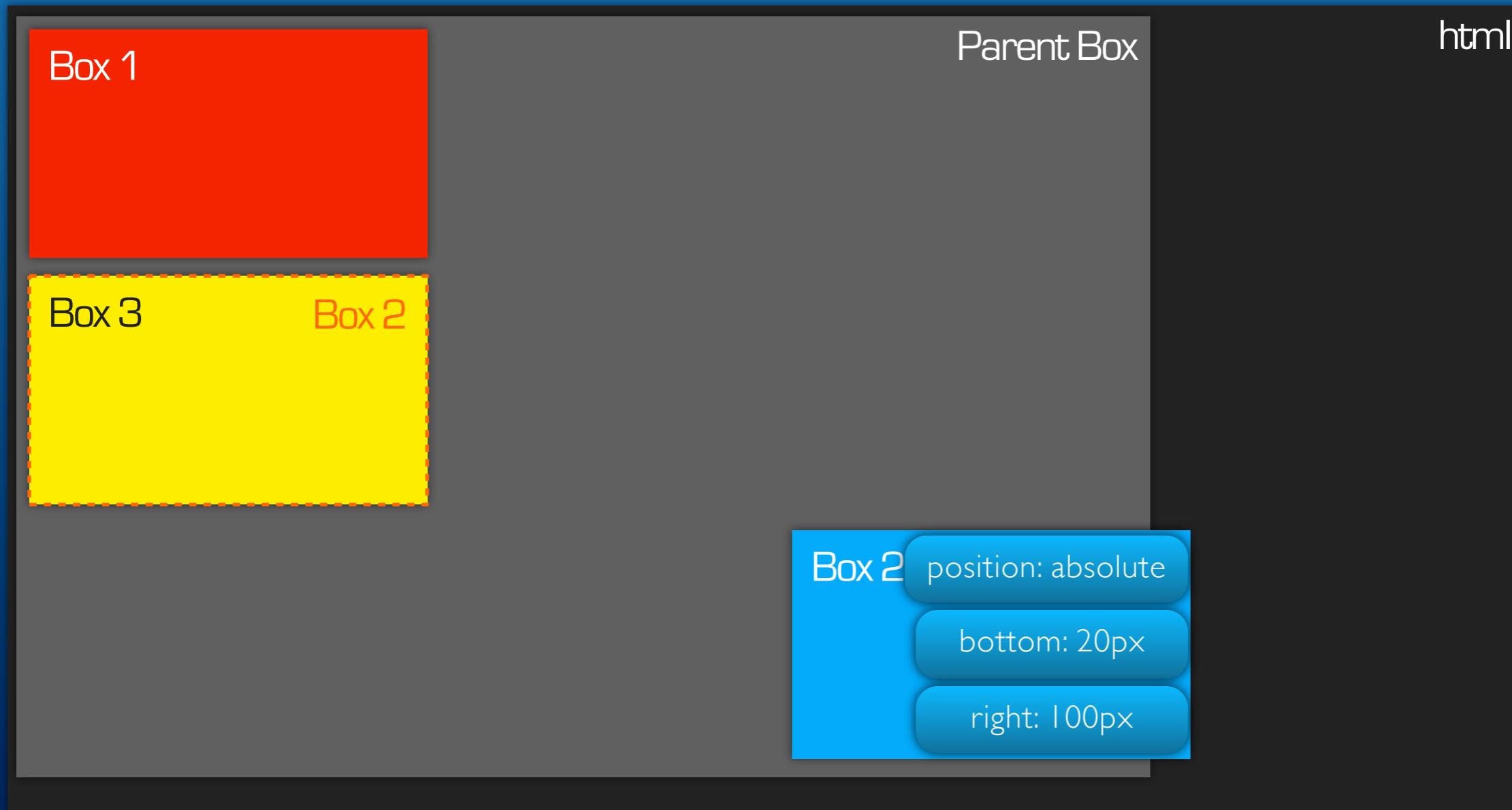
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성을 사용하면 대상의 위치를 조정할 수 있습니다.

**절대좌표 이동**  
position: **absolute**

자신의 부모 위치로부터 상대적인 거리로 이동합니다.

우측하단을 기점으로 시작점으로 이동

Box 2 position: absolute  
bottom: 20px  
right: 100px

100px  
20px

html



# CSS Position Model

CSS position 속성을 사용하면 대상의 위치를 조정할 수 있습니다.

## 절대좌표 이동

position: absolute

자신의 부모 위치로부터 상대적인 거리로 이동합니다.

우측하단을 기점으로 시작점으로 이동

Box 1

Box 3

Box 2

Box 2 position: absolute  
bottom: 20px  
right: 100px

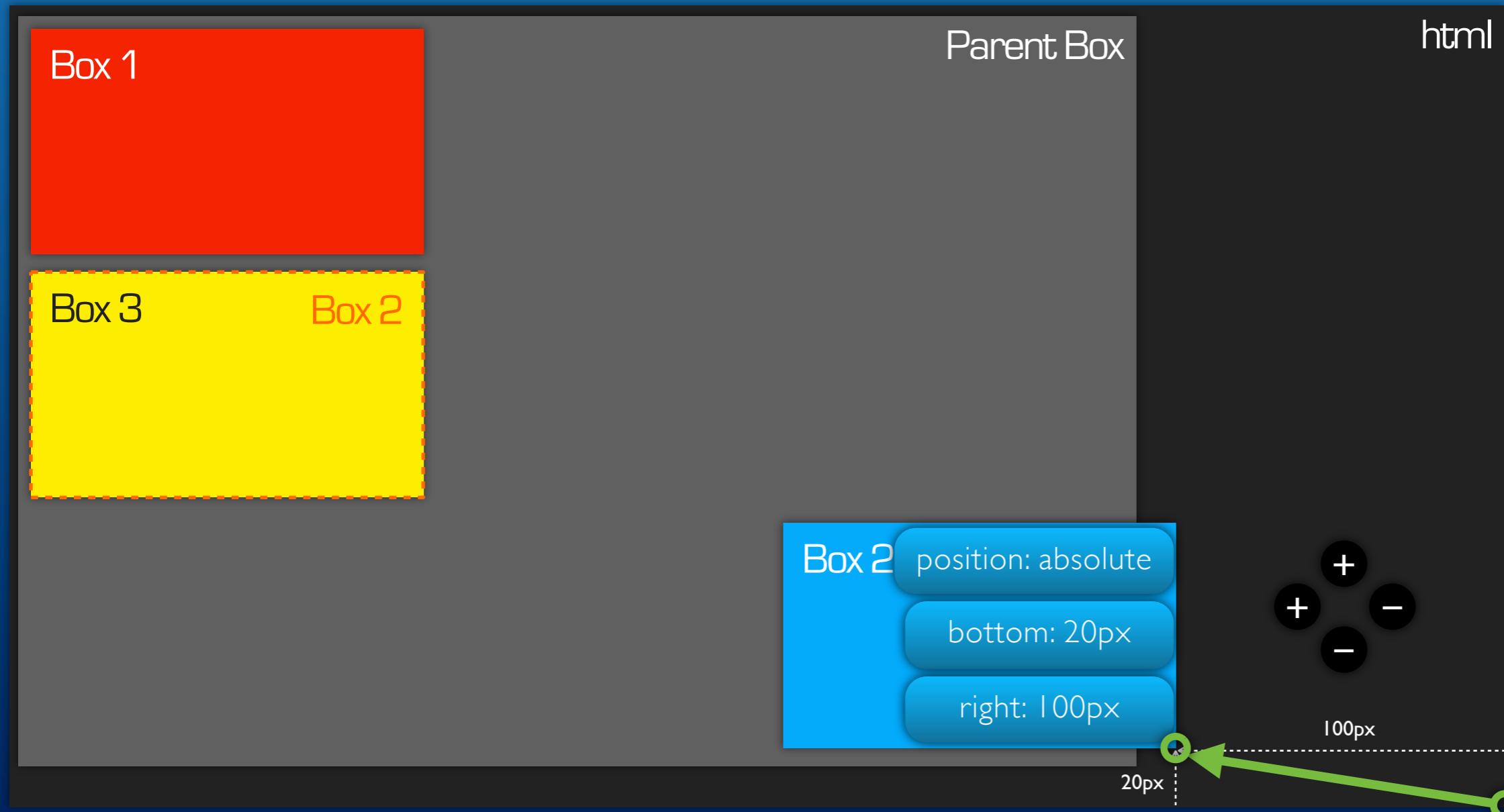
20px

100px



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

html

Box 1

Box 3

Parent Box  
position: relative

부모 찾기

position 속성을 가진 상위 요소를 찾습니다.  
단, static 속성은 제외 대상입니다.

찾은 부모 위치로부터  
상대적인 거리로 이동합니다.

Box 2  
position: absolute  
bottom: 20px  
right: 100px

+

-

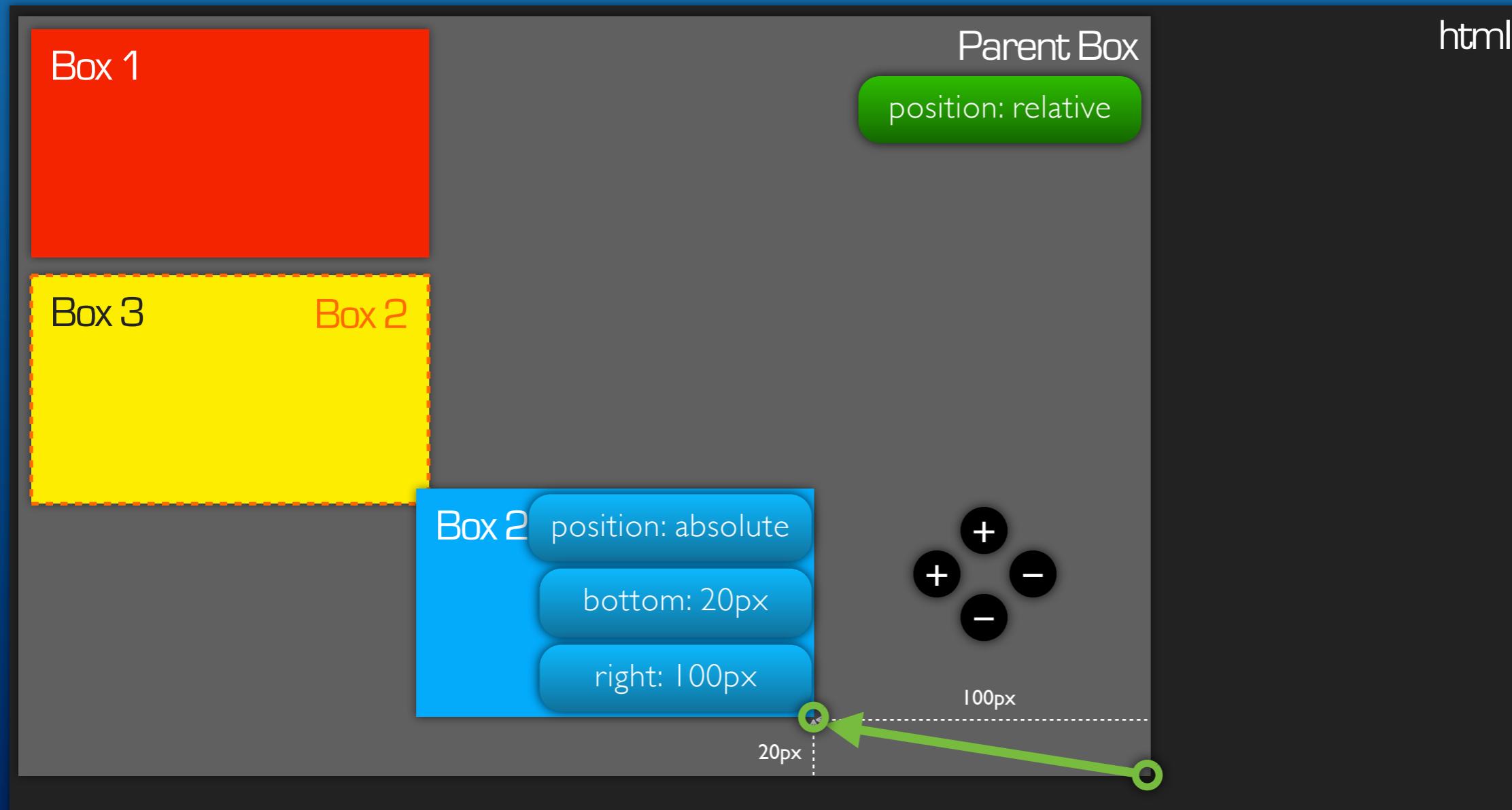
100px

20px



# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

Box 1

Box 3      Box 2

html

**부모찾기**

position 속성을 가진 상위 요소가 없다면 상위 요소를 찾고 찾아 결국 최상위 요소인 html까지 찾아 올라갑니다.

찾은 부모 위치로부터 상대적인 거리로 이동합니다.

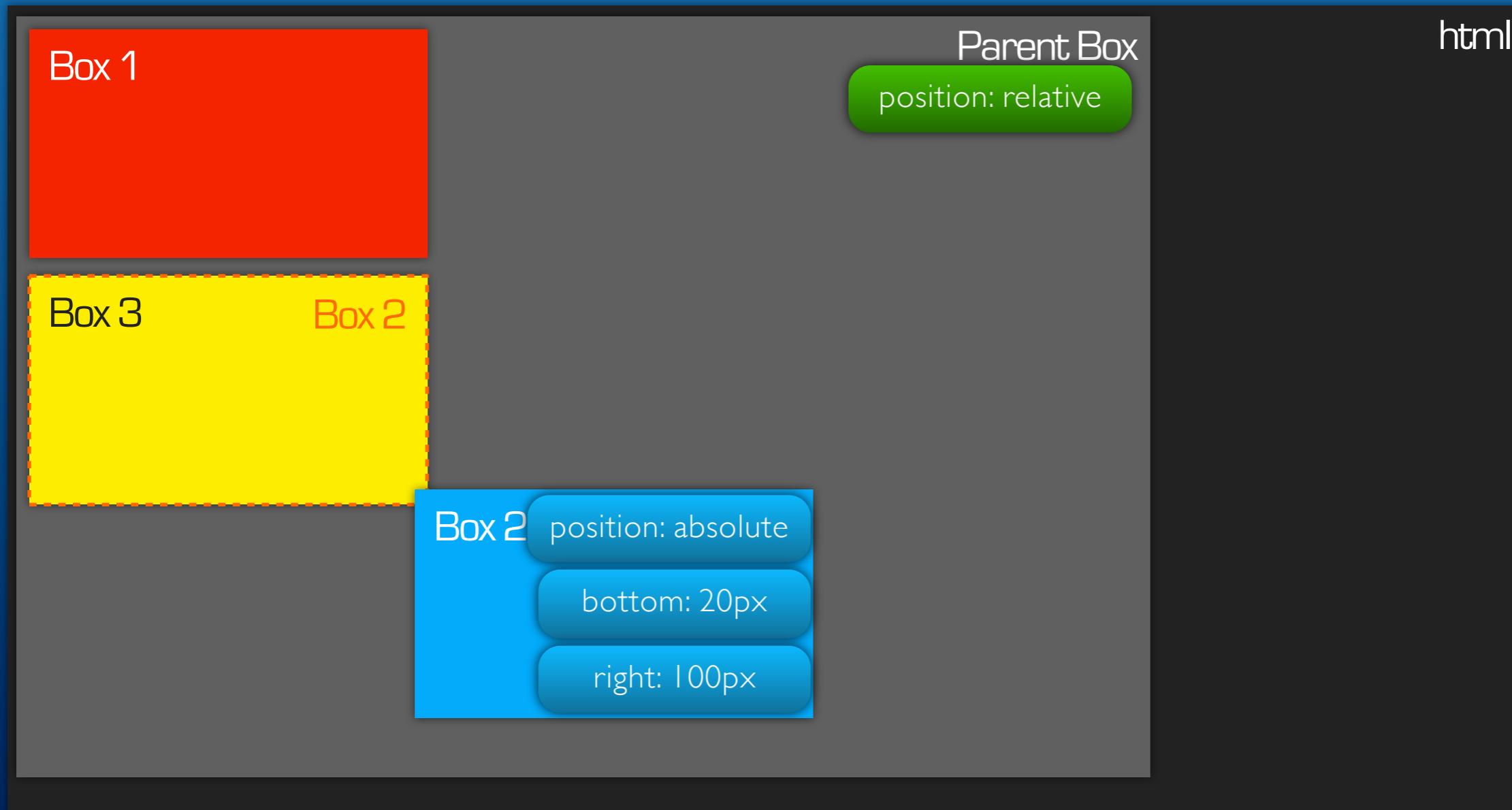
Box 2 position: absolute  
bottom: 20px  
right: 100px

100px  
20px



# CSS Position Model

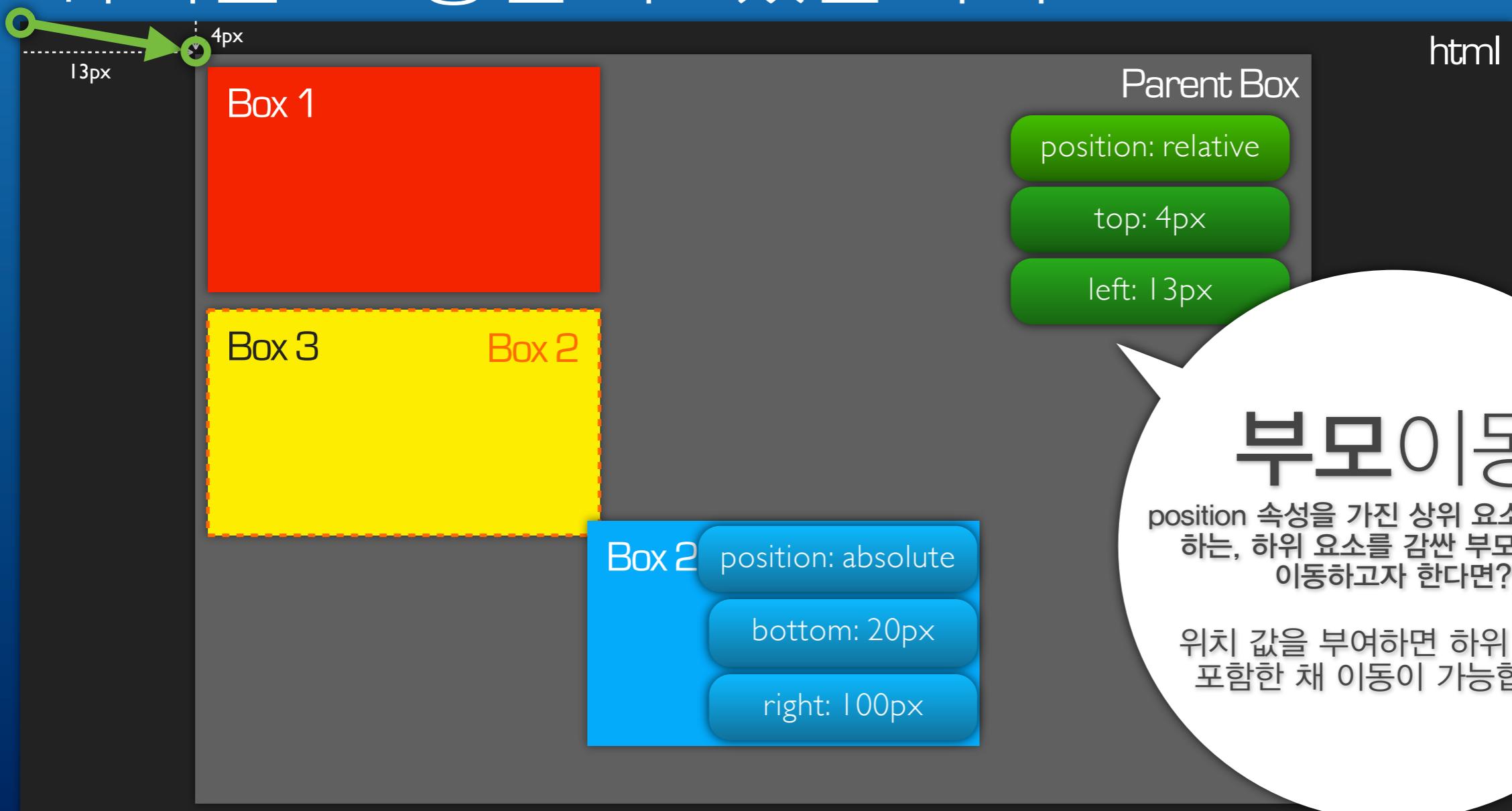
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



## IE6.Bug position

Firefox File Edit View History Bookmarks Tools Window Help

file:/// - IE6 Bug - position-fix

```
<head>
<title>IE6 Bug - position</title>
<style type="text/css">
<!--
/* IE6 Bug - position

body {
    font-size:12px;
    color:#fff;
    line-height:1.3;
    background:#3f3f;
}

#page-wrap {
    position:relative;
    width:480px;
    margin:200px auto 0;
    padding:15px;
}

#page-wrap img {
    position:absolute;
}

#page-wrap img:first-child {
    top:-40px;
    right:-250px;
}

-->
</style>
</head>
<body>
    <div id="page-wrap">
        
        
        <h1>비 표준모드 인터넷 익스플로러 6은 상대위치 지정 내 절대위치 지정에 문제가 있습니다.</h1>
        <p>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci. Aenean dignissim pellentesque felis. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec odio. Quisque volutpat mattis eros. Nullam malesuada erat ut turpis. Suspendisse urna nibh, viverra non, semper suscipit, posuere a, pede.</p>
    </div>
</body>

```

비 표준모드 인터넷 익스플로러 6은 상대위치 지정 내 절대위치 지정에 문제가 있습니다.

Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci. Aenean dignissim pellentesque felis. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec odio. Quisque volutpat mattis eros. Nullam malesuada erat ut turpis. Suspendisse urna nibh, viverra non, semper suscipit, posuere a, pede.



Set Up Firefox Sync...

Done

비 표준모드 인터넷 익스플로러 6은 상대위치 지정 내 절대위치 지정에 문제가 있습니다.

Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci. Aenean dignissim pellentesque felis. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec odio. Quisque volutpat mattis eros. Nullam malesuada erat ut turpis. Suspendisse urna nibh, viverra non, semper suscipit, posuere a, pede.

신뢰할 수 있는 사이트

Share Hints Clips

53:23

Done

비 표준모드 인터넷 익스플로러 6은 상대위치 지정 내 절대위치 지정에 문제가 있습니다.

Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci.

다.</p>

<b>Donec nec justo eget felis facilisis fermentum. Aliquam porttitor mauris sit amet orci.</b>

다.<br>

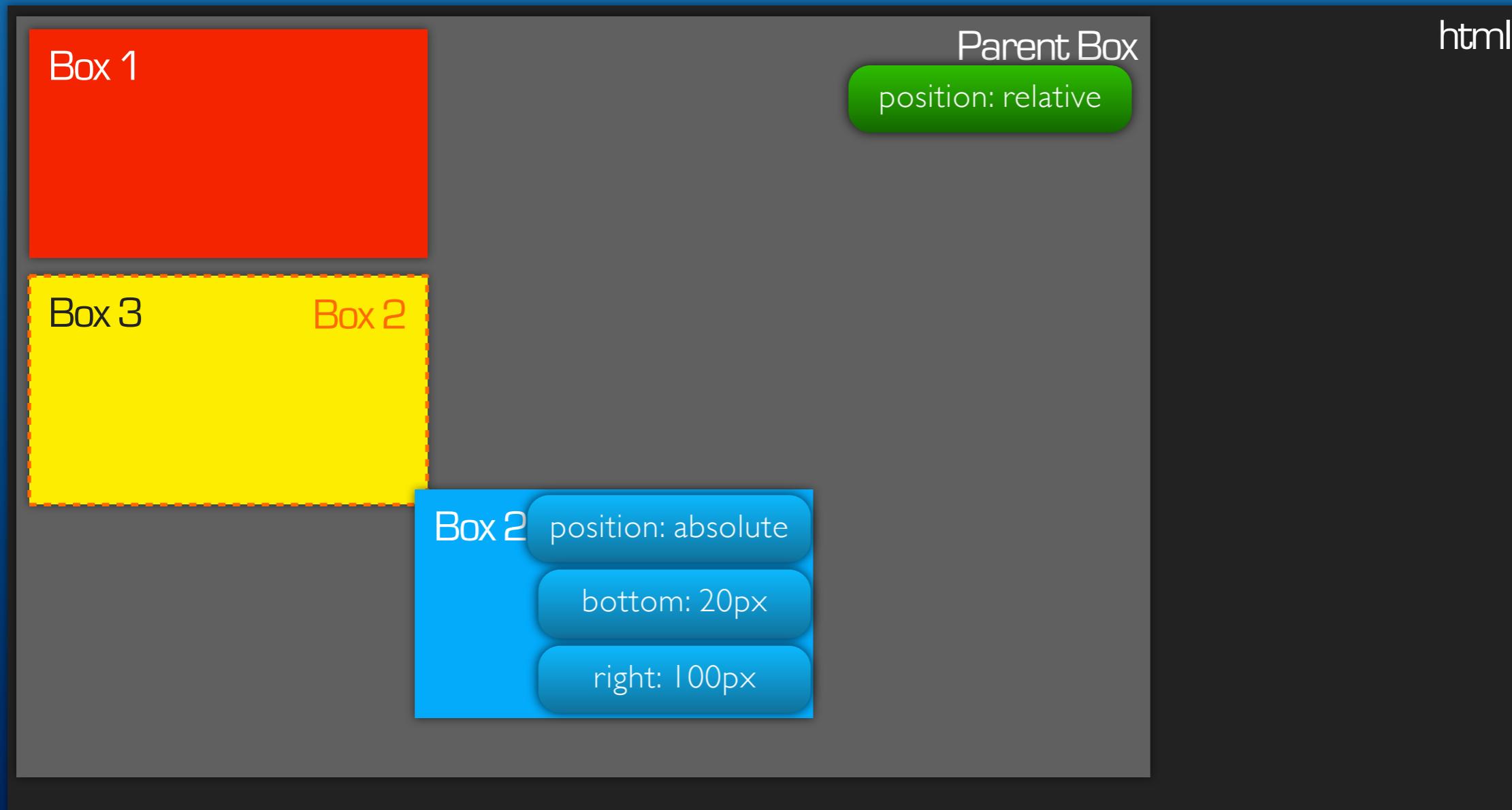
<h1>비 표준모드 인터넷 익스플로러 6은 상대위치 지정 내 절대위치 지정에 문제가 있습니다.</h1>





# CSS Position Model

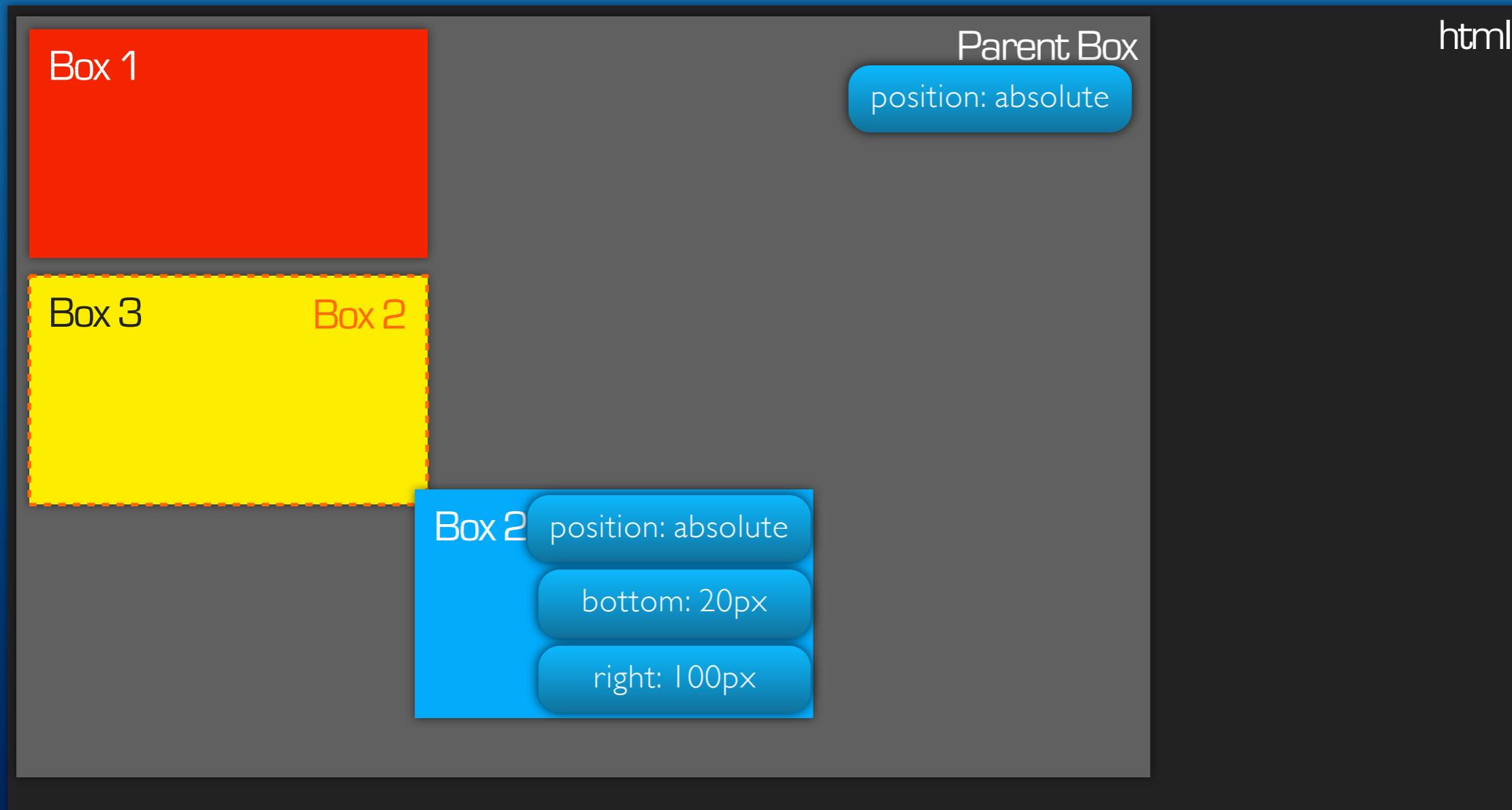
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

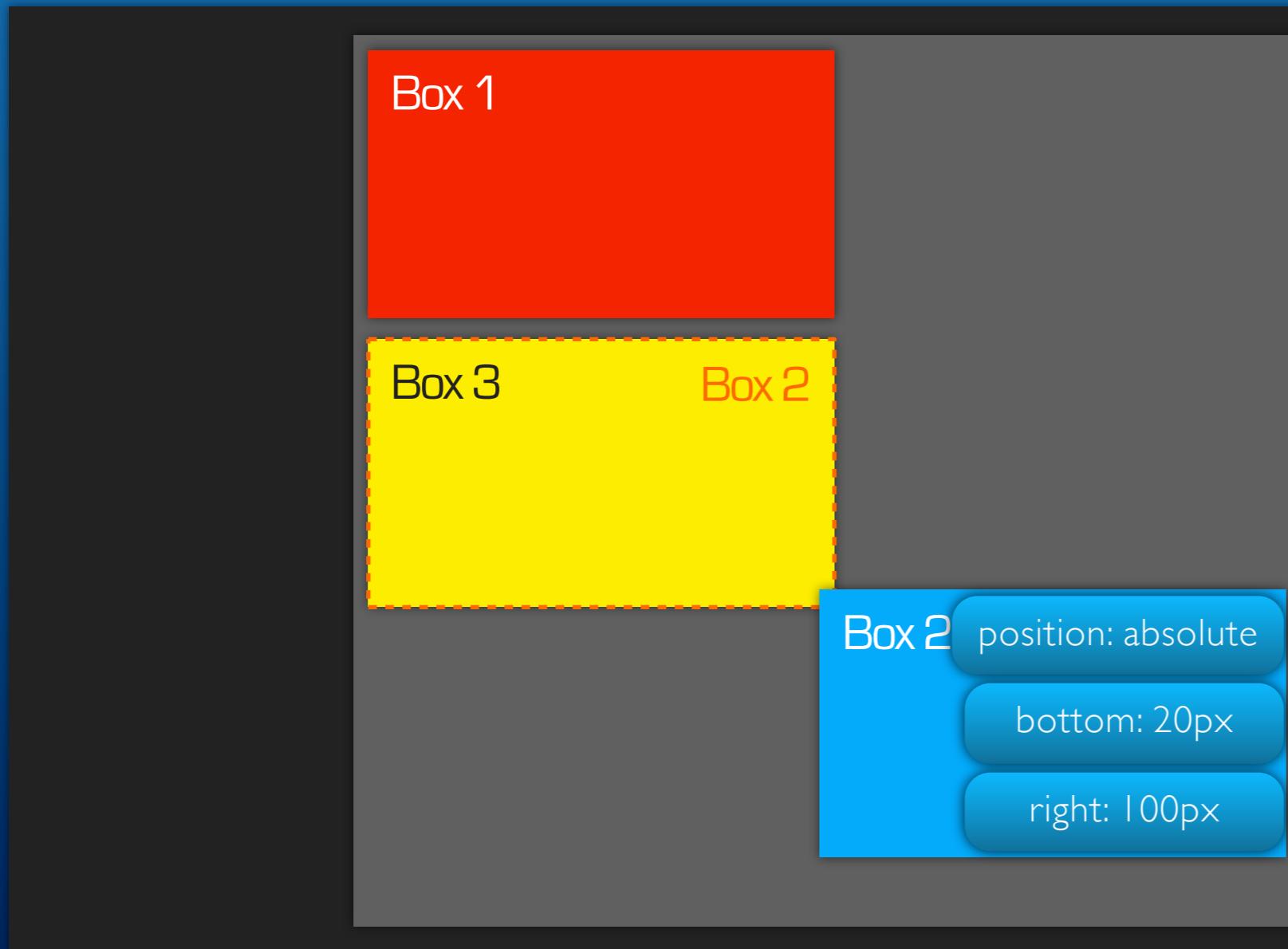
CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.



## 부모 이동

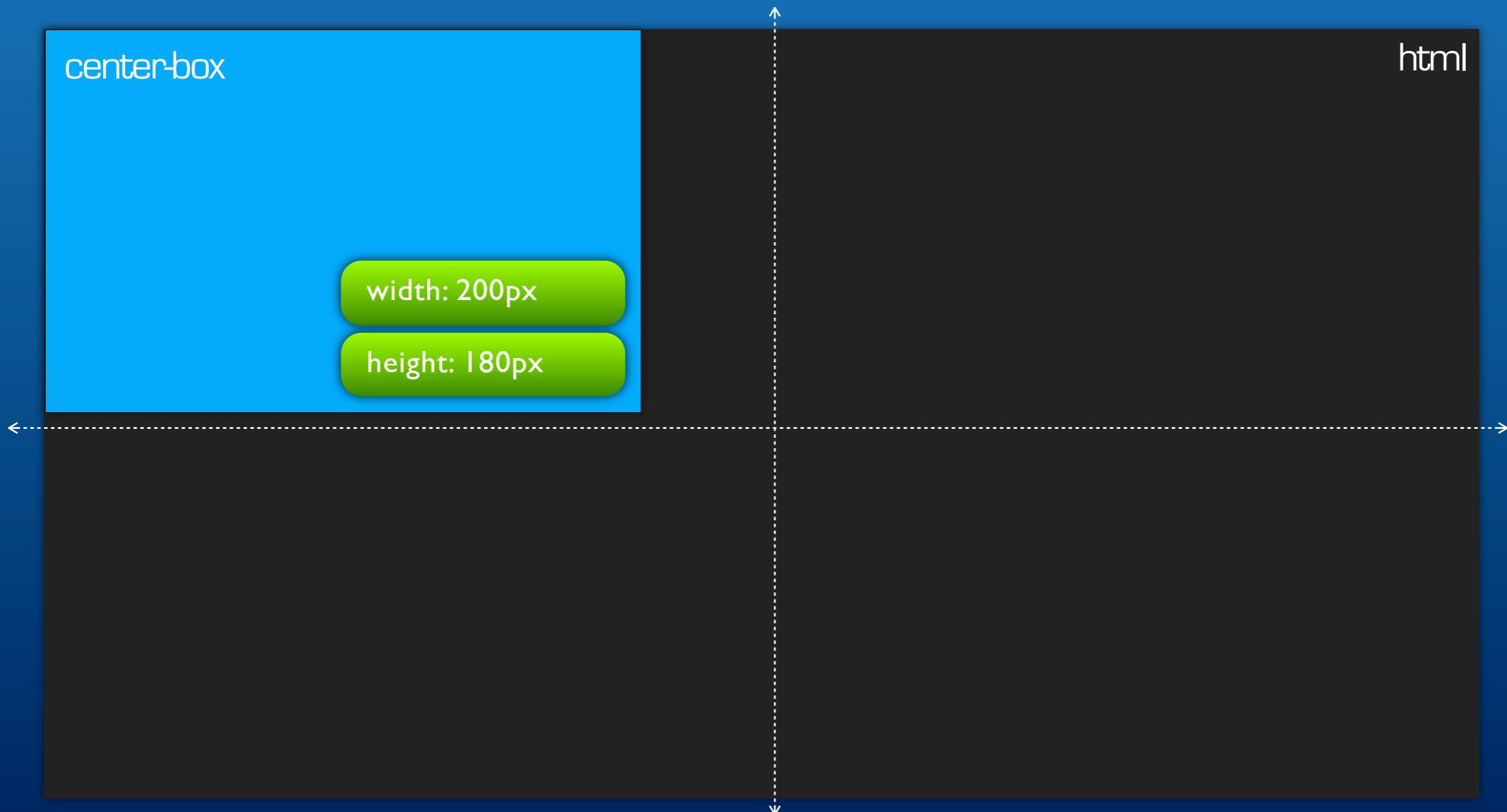
position 속성을 가진 상위 요소를 부모로 하는, 하위 요소를 감싼 부모 요소를 이동하고자 한다면?

relative → absolute 한 후,  
위치 값을 부여하면 하위 요소를  
포함한 채 이동이 가능합니다.



# CSS Center Position

브라우저 창 가운데 배치하는 방법



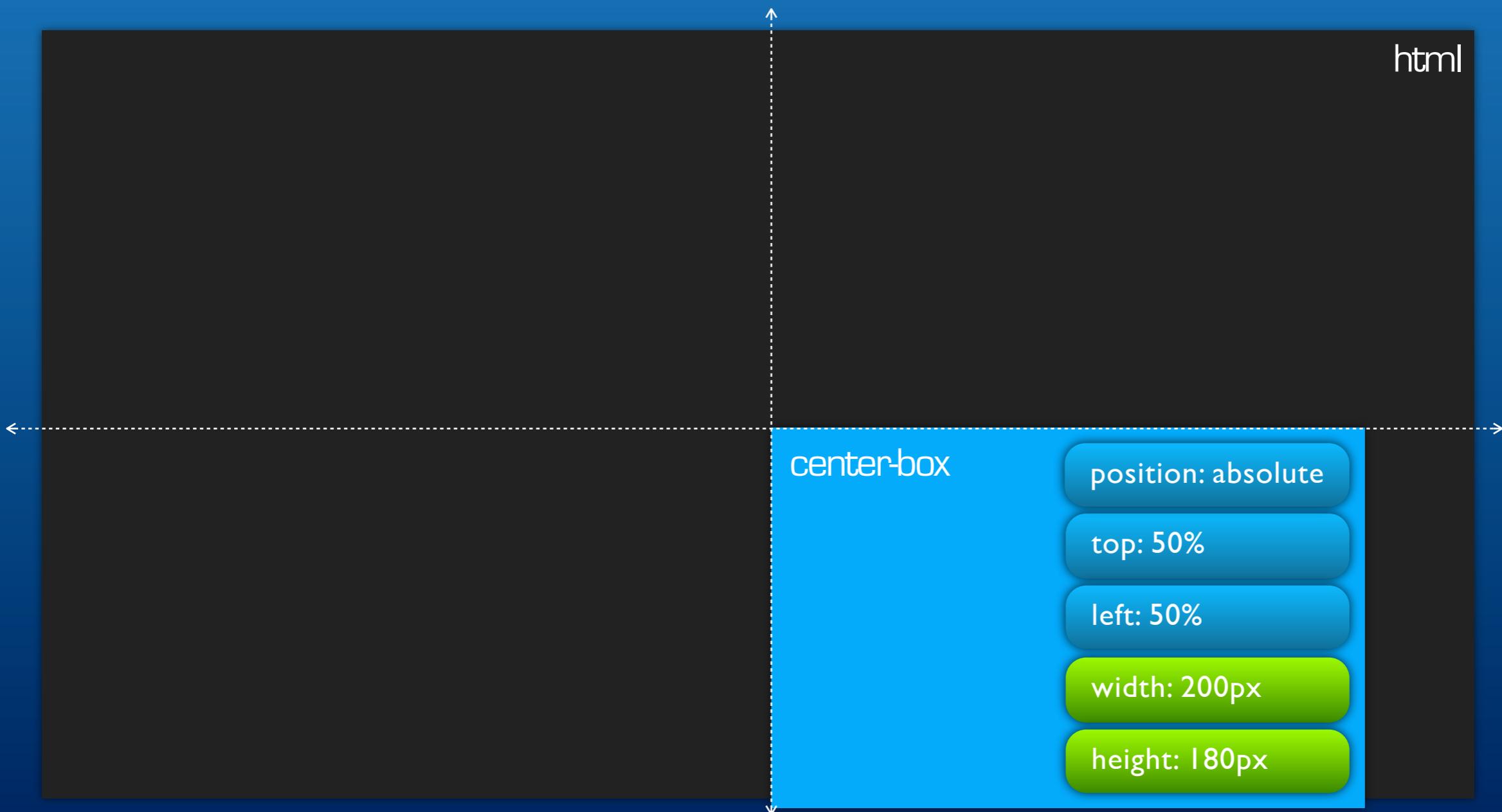
# CSS Center Position

브라우저 창 가운데 배치하는 방법



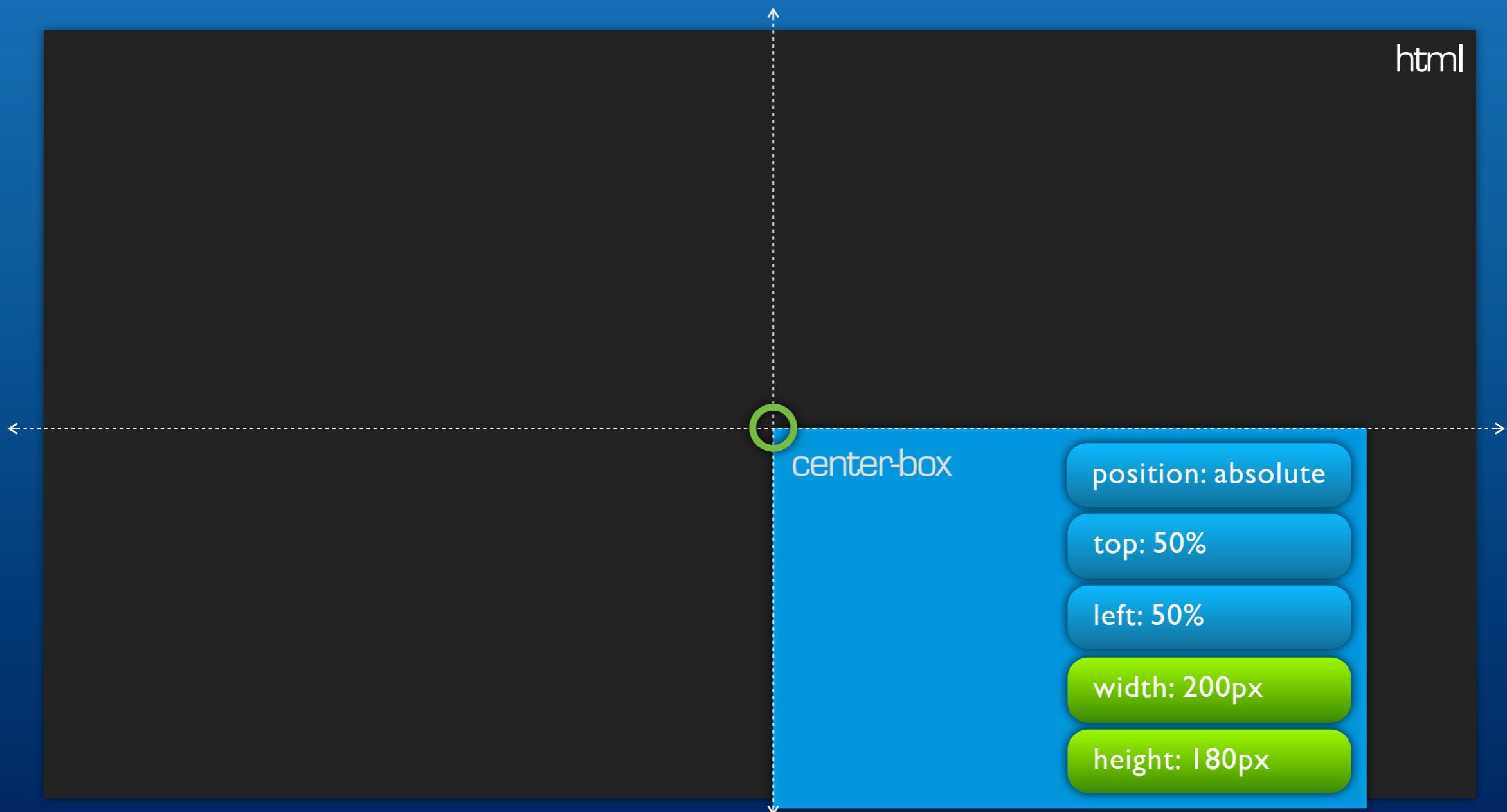
# CSS Center Position

브라우저 창 가운데 배치하는 방법



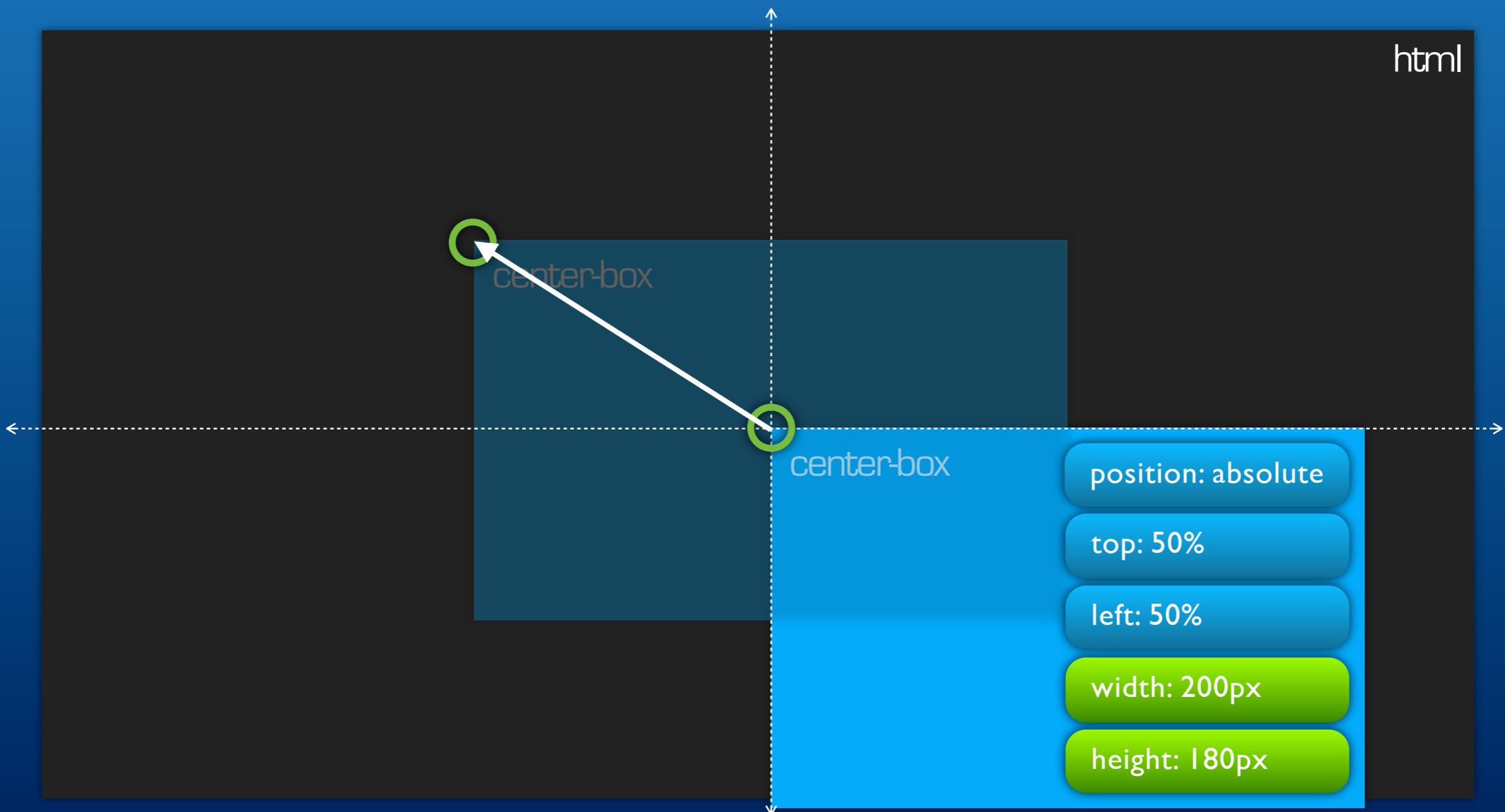
# CSS Center Position

브라우저 창 가운데 배치하는 방법



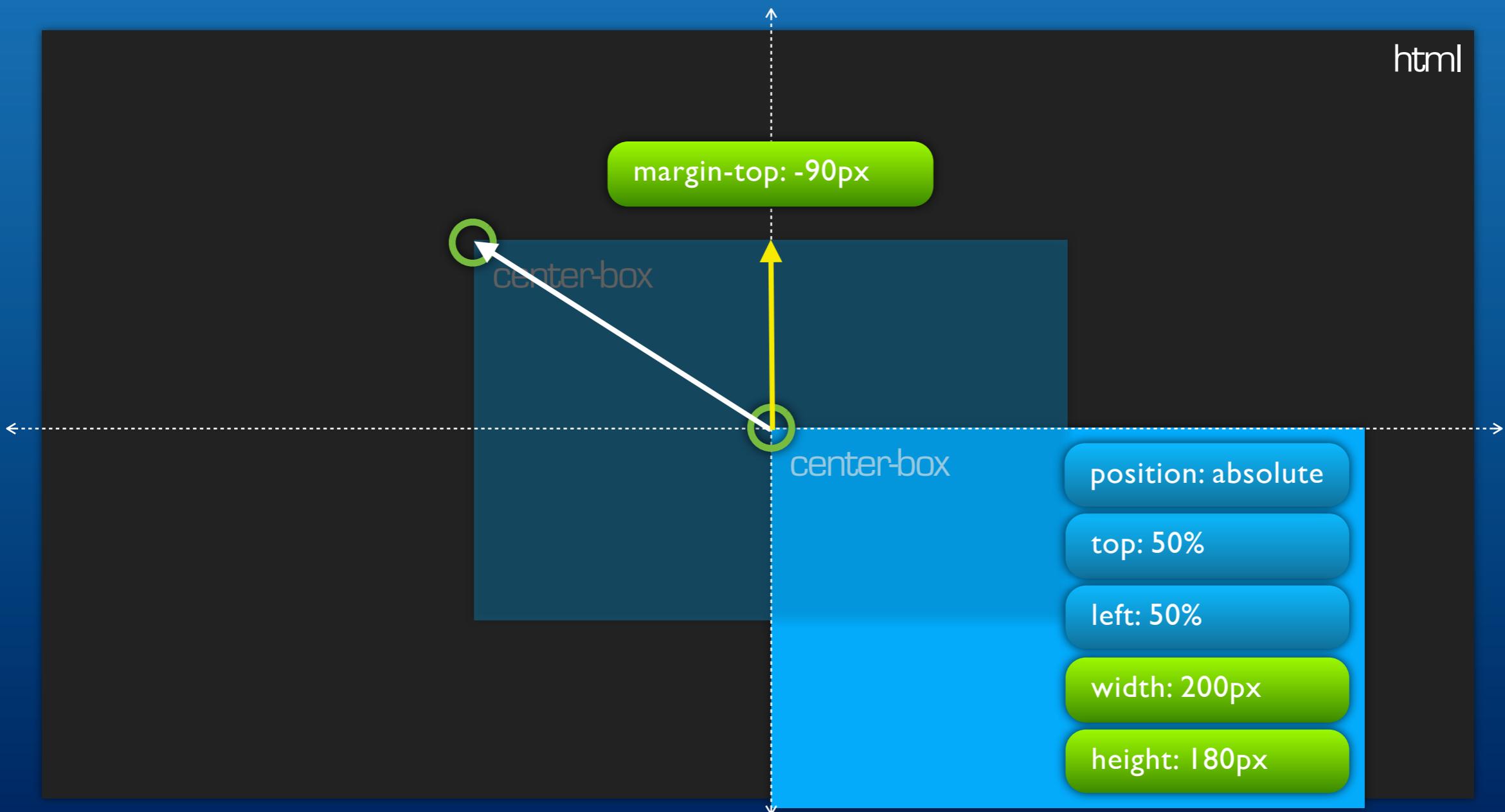
# CSS Center Position

브라우저 창 가운데 배치하는 방법



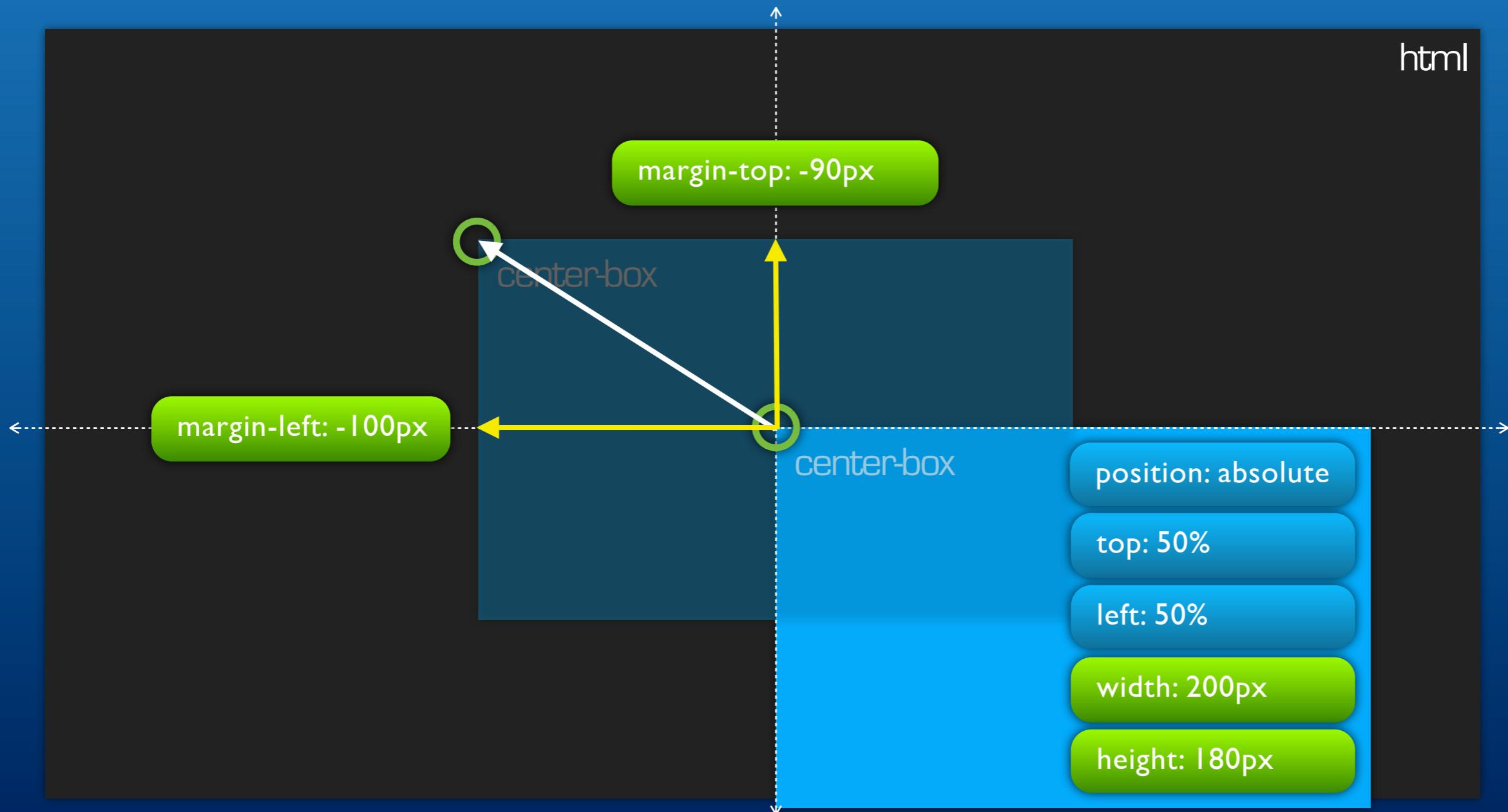
# CSS Center Position

브라우저 창 가운데 배치하는 방법



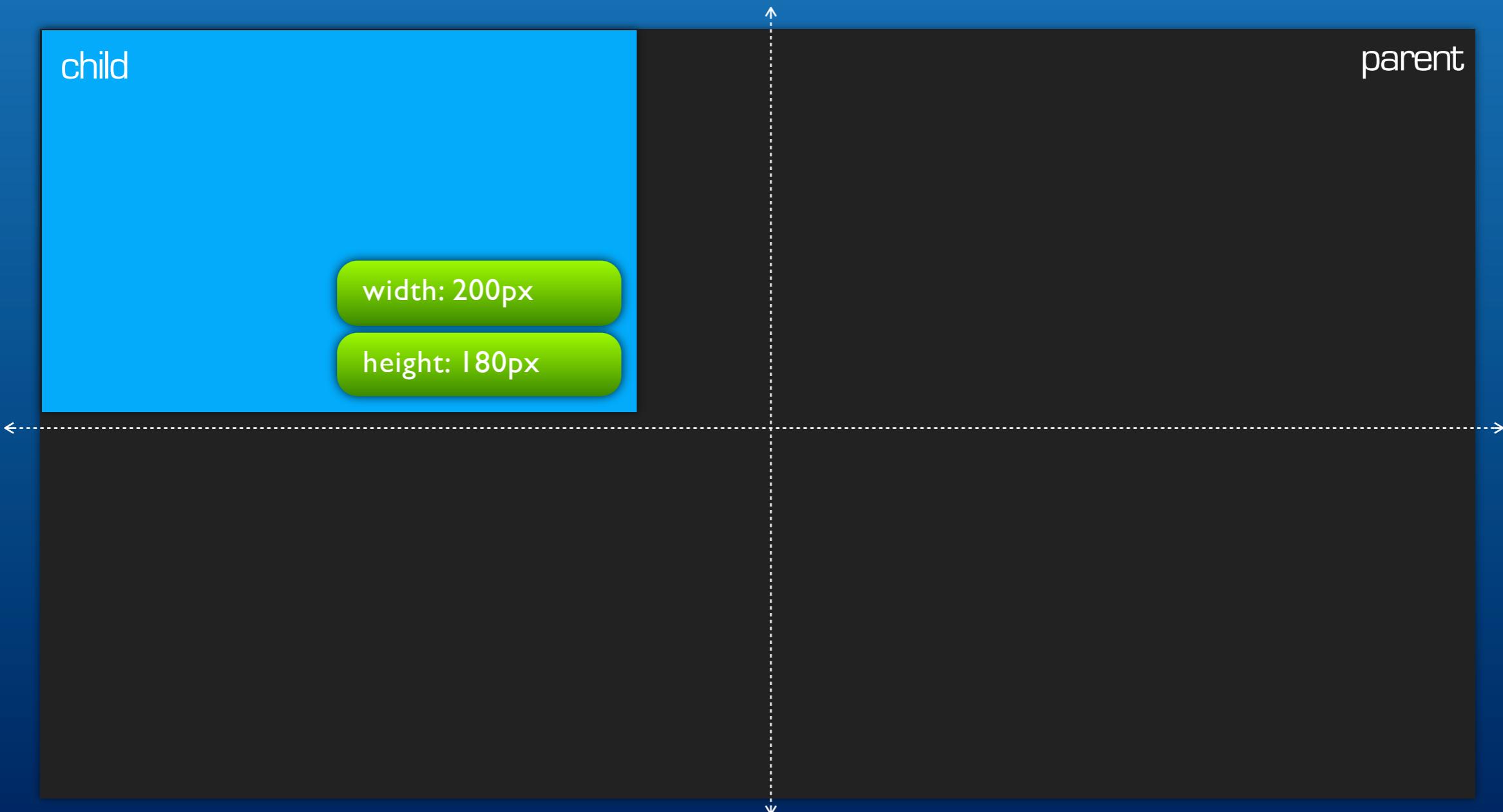
# CSS Center Position

브라우저 창 가운데 배치하는 방법



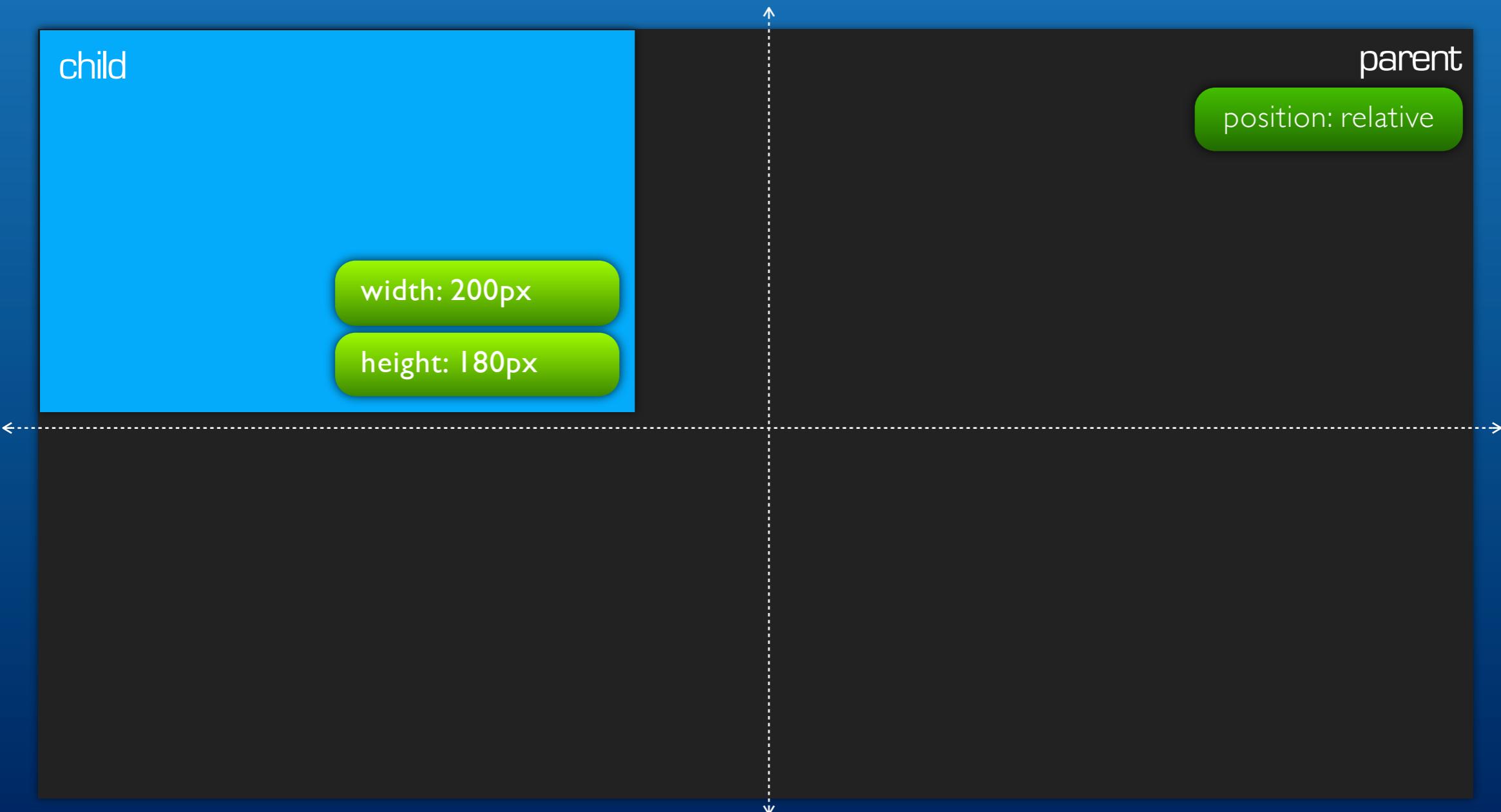
# CSS Center Position

부모 요소 가운데 자식 요소 배치하는 방법



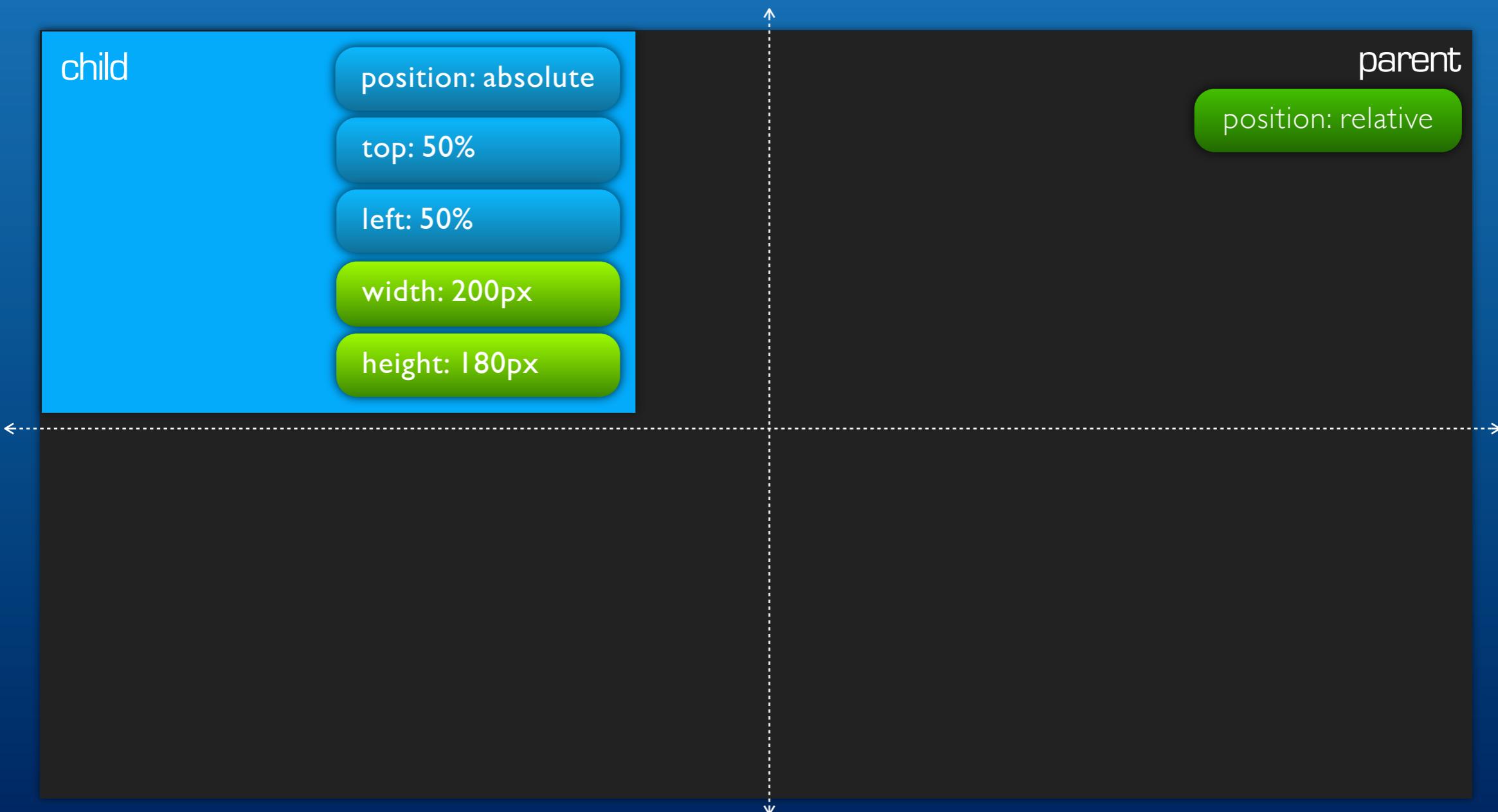
# CSS Center Position

부모 요소 가운데 자식 요소 배치하는 방법



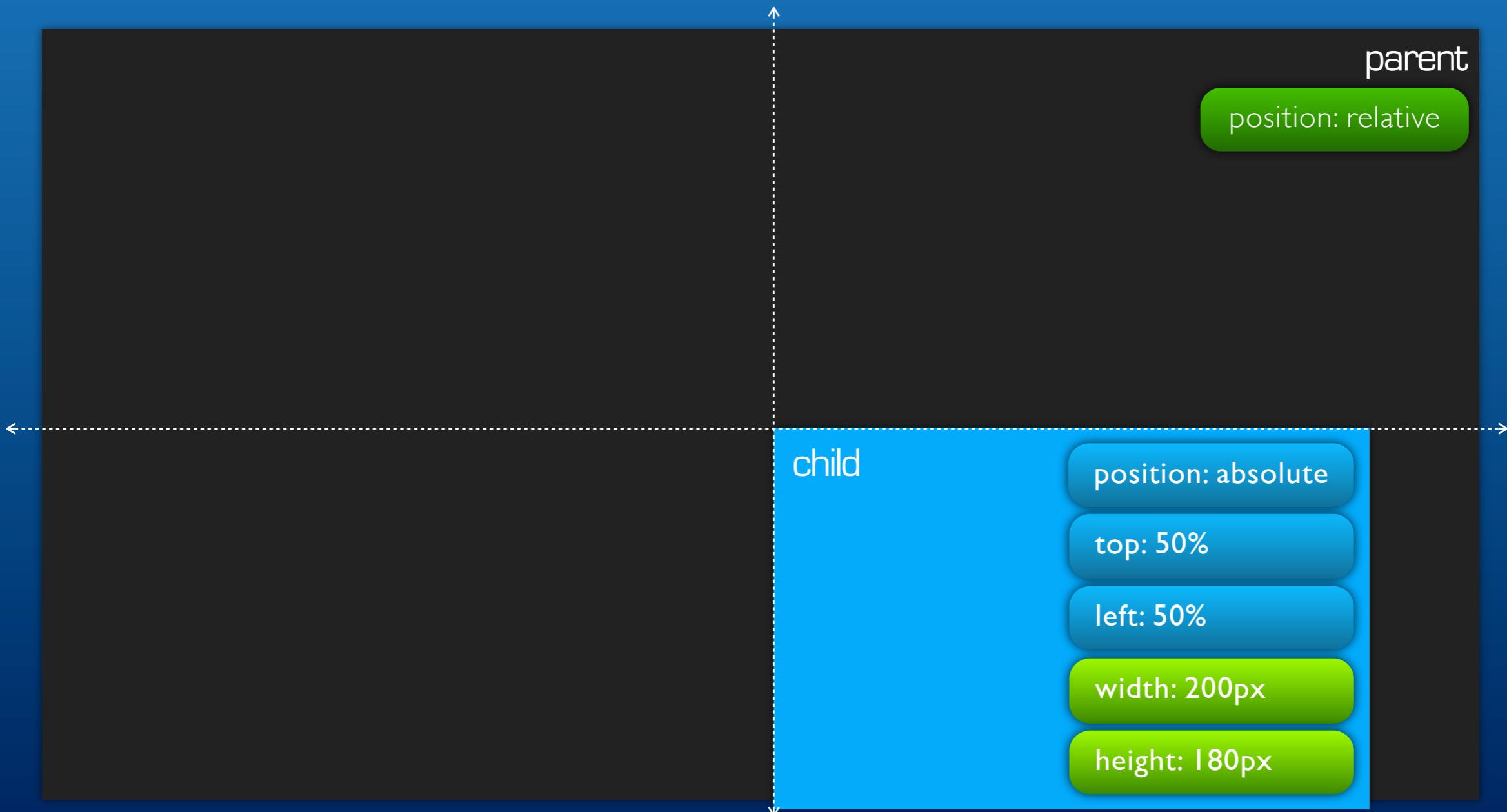
# CSS Center Position

부모 요소 가운데 자식 요소 배치하는 방법



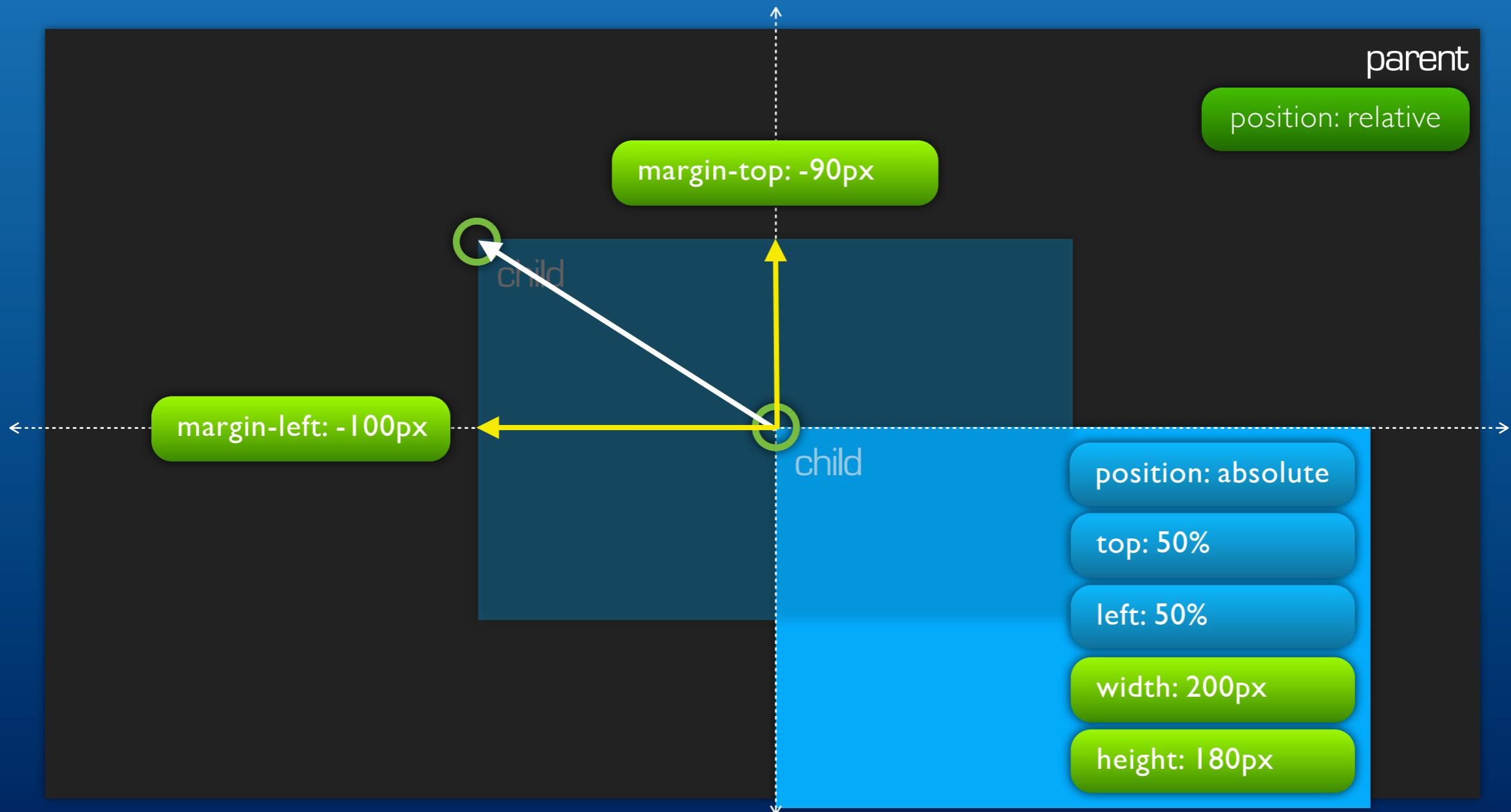
# CSS Center Position

부모 요소 가운데 자식 요소 배치하는 방법



# CSS Center Position

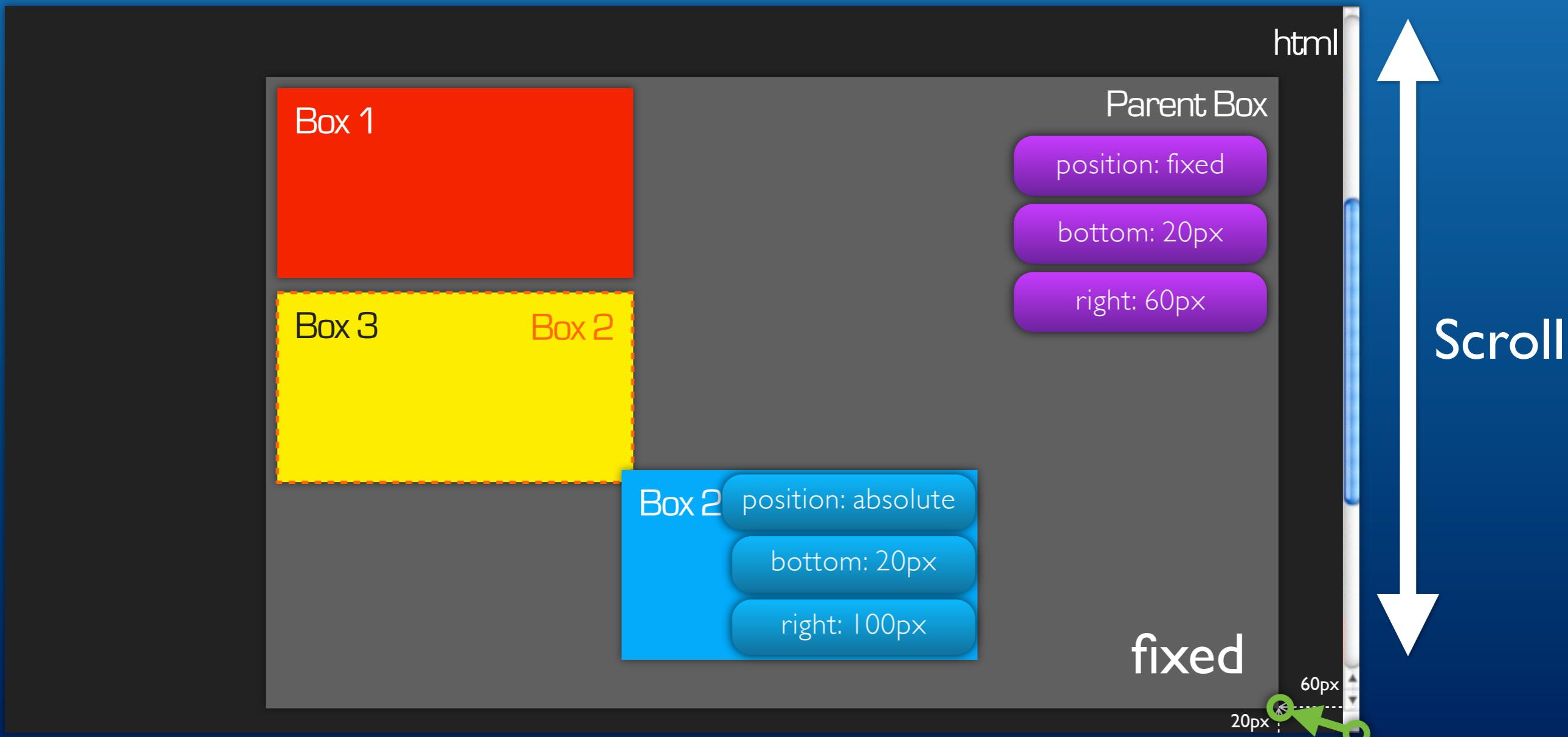
부모 요소 가운데 자식 요소 배치하는 방법





# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.





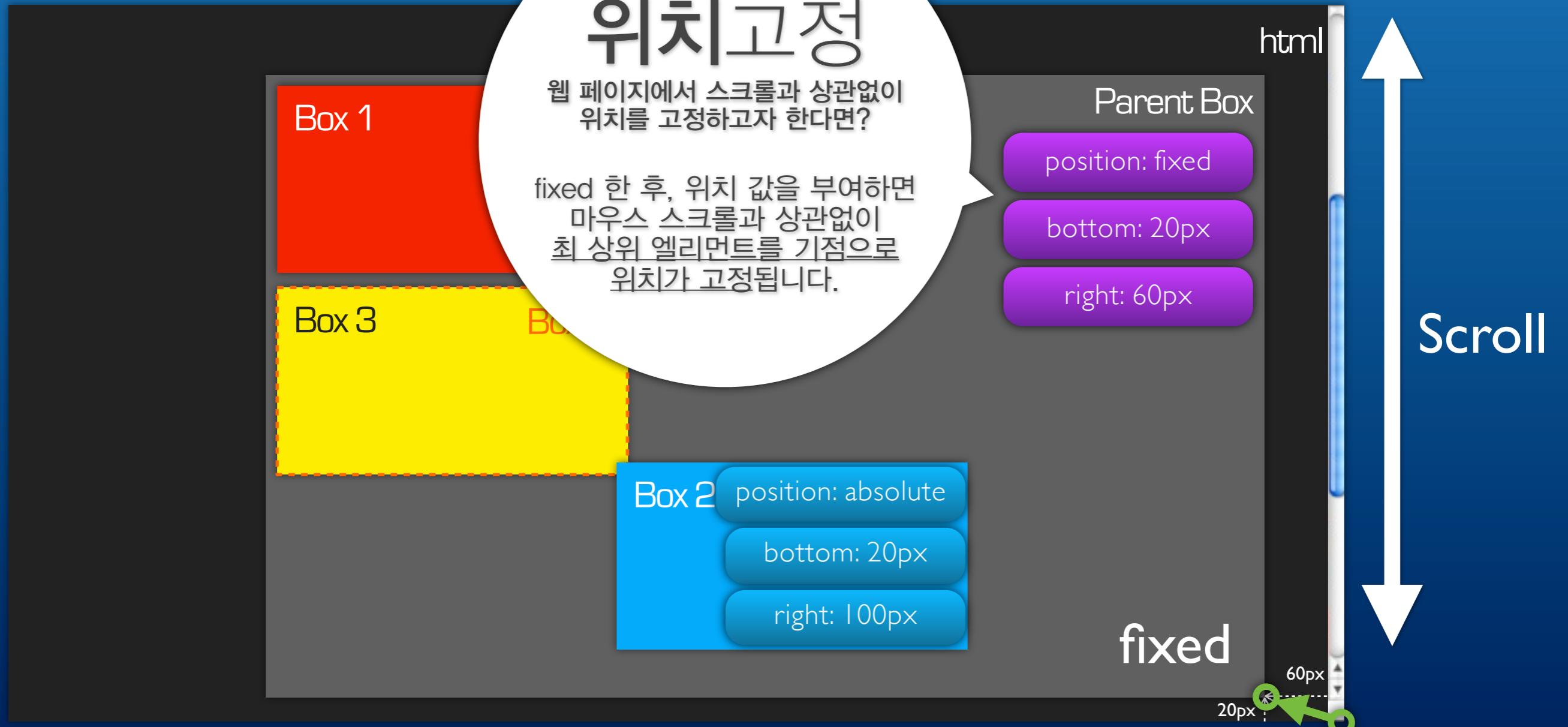
# CSS Position Model

CSS position 속성의 값에 따라 대상의 위치를 조정할 수 있습니다.

## 위치고정

웹 페이지에서 스크롤과 상관없이 위치를 고정하고자 한다면?

fixed 한 후, 위치 값을 부여하면  
마우스 스크롤과 상관없이  
최 상위 엘리먼트를 기점으로  
위치가 고정됩니다.





# CSS Position Model

IE6 position:fixed 적용방법

CSS 위치 정렬 모델

위치

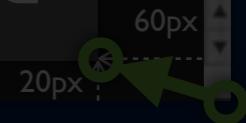
```
@charset "utf-8";  
  
/* 2010.05.24 아카데미정글, 야무(yamoo9) - IE6 Bug position:fixed 해결책 */  
  
/* 초기화 */  
* {padding: 0; margin: 0;}  
  
/* html 스크롤바 숨기기 */  
html {overflow: hidden;}  
  
/* body 화면 채우기 및 스크롤바 생성하기 */  
body {overflow: auto; width:100%; height:100%;}  
  
/* aside 절대위치로 body를 기점으로 고정하기 */  
#aside {position:absolute; bottom:0; right:0;}
```

Box 2 position: absolute

bottom: 20px

right: 100px

fixed



Scroll

## IE6 position:fixed 적용방법

```
@charset "utf-8";  
  
/*  
 2010.05.24  아카데미정글, 야무(yamoo9) - IE6 Bug position:fixed 해결책  
*/  
  
/* 초기화 */  
* {padding: 0; margin: 0;}  
  
/* html 스크롤바 숨기기 */  
html {overflow: hidden;}  
  
/* body 화면 채우기 및 스크롤바 생성하기 */  
body {overflow: auto; width:100%; height:100%;}  
  
/* aside 절대위치로 body를 기점으로 고정하기 */  
#aside {position:absolute; bottom:0; right:0;}
```

**IE6.Bug position:fixed;**

## IE6.Bug position:fixed;

The screenshot shows two windows of the Coda code editor. The left window is titled 'IE6 position:fixed - 해결방법' and displays a red box containing the text 'position:fixed' and the CSS rule '#aside { position:fixed; bottom:0; }'. The right window is titled 'IE6\_fixed\_solve.html' and shows the source code for a file named 'IE6\_pos\_fixed\_solved.css'. The code includes a DOCTYPE declaration, meta tags, and a style block for '#contents' and '#aside'.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko" lang="ko">
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
        <title>IE6 position:fixed - 해결방법</title>
        <!--<link href=".css/IE6_pos_fixed_solved.css" type="text/css"
rel="stylesheet" media="screen" />-->
        <style type="text/css" media="screen">
        <!--
            * {margin:0; padding:0;}
            html {overflow:hidden;} /* html 스크롤바 감춤 */
            body {overflow:auto; height:100%; width:100%;}
            /* body 스크롤바를 자동으로 보여줌. 화면에 가득 채움 */

            #contents {width:300px;}
            #aside {
                position: fixed;
                _position: absolute; /* IE6 */
                z-index:-1;
                top:100px;
                right:0;
                width:200px;
                padding:15px;
                background:#c00;
                color:#fff;
            }
            #aside code {
                white-space: pre;
            }
        </style>
    </head>
    <body>
        <div id="contents">
            <h1>IE6 position:fixed - 해결방법</h1>
            <p>According to CSS level 2, an element can be positioned relative to the browser window using the style position: fixed: it does not move when the page is scrolled. You can do nice layout things with this in most modern browsers - but not on IE for Windows. Unless you use this script.

According to CSS level 1, an element's background can be positioned relative to the browser window using the style background-attachment: fixed: the body of the element acts like a window onto the background image, which stays still when the page is scrolled. You can do nice layout things with this in most modern browsers... but, again, not correctly on IE/Win. (Unless, etc.)



IE still doesn't know that fixed-position elements should not make the window bigger. The upshot is that if you make a fixed-position element larger than the



Done



Sites Edit Preview CSS Terminal Books



IE6_fixed_solve.html IE6_pos_fixed_solved.css


```

## IE6.Bug position:fixed;

The screenshot shows a web page with a green header bar containing the 'doxdesk' logo and a red button labeled 'fixed.js module'. On the left, there's a sidebar with 'Software' and 'also by this author' sections. The main content area has a white background and contains several sections:

- Fixed positioning**: A paragraph explaining that according to CSS level 2, an element can be positioned relative to the browser window using the style `position: fixed;`. It notes that it does not move when the page is scrolled, which is a bug in IE6.
- Fixed backgrounds**: A paragraph explaining that according to CSS level 1, an element's background can be positioned relative to the browser window using the style `background-attachment: fixed;`. It notes that the body of the element acts like a window onto the background image, which stays still when the page is scrolled.
- Using fixed.js**: A section explaining that the script makes these CSS properties work correctly in IE/Win version 5.0 upwards. It instructs users to insert a link to the module anywhere in their HTML (e.g., in `<head>`):

```
<script type="text/javascript" src="fixed.js"></script>
```
- Caveats**: A note stating that IE still doesn't know that fixed-position elements should not make the window bigger. It mentions that if you make a fixed-position element larger than the window, you can get a page that will carry on scrolling down past the bottom of the proper page.
- Latest release**: Information about version 1.8, with links to the module and example page.

At the bottom of the page, there's a footer with the email address `and@doxdesk.com` and a small logo.

## IE6.Bug position:fixed;

doxdesk

fixed.js module

Software ◀  
also by this author

**Fixed positioning**

According to CSS level 2, an element can be positioned relative to the browser window using the `:position: fixed;` It does not move when the page is scrolled. You can do nice things with this in JS

```
// fixed.js: fix fixed positioning and fixed backgrounds in IE/Win
// version 1.8, 08-Aug-2003
// written by Andrew Clover <and@doxdesk.com>, use freely

/*@cc_on
@if (@_win32 && @_jscript_version>4)

var fixed_positions= new Array();
var fixed_backgrounds= new Array();
var fixed_viewport;

// Initialisation. Called when the <body> tag arrives. Set up viewport so the
// rest of the script knows we're going, and add a measurer div, used to detect
// font size changes and measure image sizes for backgrounds later

function fixed_init() {
    fixed_viewport= (document.compatMode=='CSS1Compat') ?
        document.documentElement : document.body;
    var el= document.createElement('div');
    el.setAttribute('id', 'fixed-measure');
    el.style.position= 'absolute';
    el.style.top= '0'; el.style.left= '0';
    el.style.overflow= 'hidden'; el.style.visibility= 'hidden';
    el.style.fontSize= 'xx-large'; el.style.height= '5em';
    el.style.setExpression('width', 'fixed_measureFont());
    document.body.insertBefore(el, document.body.firstChild);
}
```

Don't put loads of fixed position/background elements on a page - it'll get slow for many users.

Latest release

Version 1.8: [module](#), [example page](#).

and@doxdesk.com

modul@doxdesk.com

## IE6.Bug position:fixed;

**doxdesk**

fixed.js module

Software ◀ also by this author

**Fixed positioning**

According to CSS level 2, an element can be positioned relative to the browser window using the `:position: fixed;` It does not move when the page is scrolled. You can do nice things with this in Internet Explorer.

// fixed.js: fix fixed positioning and fixed backgrounds in IE/Win  
// version 1.8, 08-Aug-2003  
// written by Andrew Clover <and@doxdesk.com>, use freely

```
/*@cc_on
var fixed_positions= new Array();
var fixed_backgrounds= new Array();
var fixed_viewport;

// Initialisation. Called when the <body> tag arrives. Set up viewport so the
// rest of the script knows we're going, and add a measurer div, used to detect
// font size changes and measure image sizes for backgrounds later
function fixed_init() {
    fixed_viewport= (document.compatMode=='CSS1Compat') ?
        document.documentElement : document.body;
    var el= document.createElement('div');
    el.setAttribute('id', 'fixed-measure');
    el.style.position= 'absolute';
    el.style.top= '0'; el.style.left= '0';
    el.style.overflow= 'hidden'; el.style.visibility= 'hidden';
    el.style.fontSize= 'xx-large'; el.style.height= '5em';
    el.style.setExpression('width', 'fixed_measureFont());
    document.body.insertBefore(el, document.body.firstChild);
}
```

Don't put loads of fixed position/background elements on a page - it'll get slow for many users.

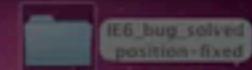
Latest release

Version 1.8: [module](#), [example page](#).

and@doxdesk.com

doxdesk.com

## IE6.Bug position:fixed;





# CSS언어를 이용하여 대상의 위치지정후 띄워볼까요?



Style code

대상 속성 값  
selector {z-index: value}

<head>



<style>

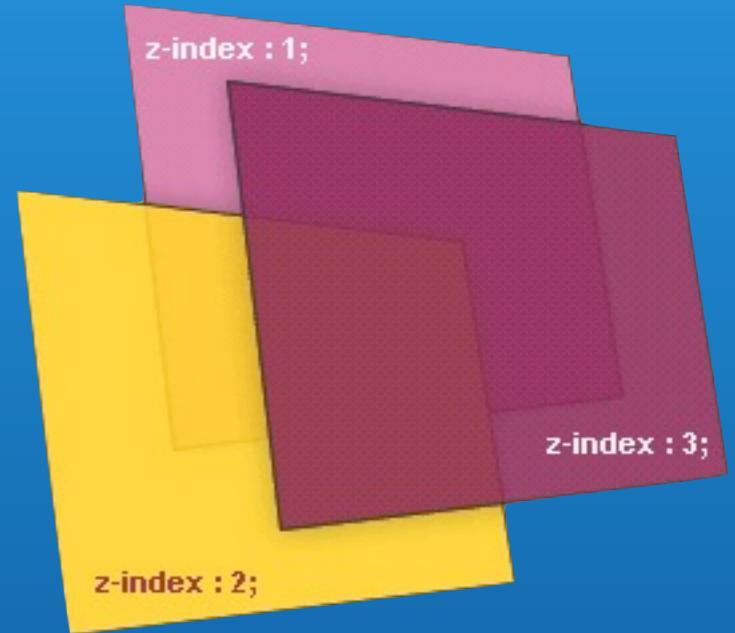
div {position: absolute; top:0; z-index:2}

</style>

</head>



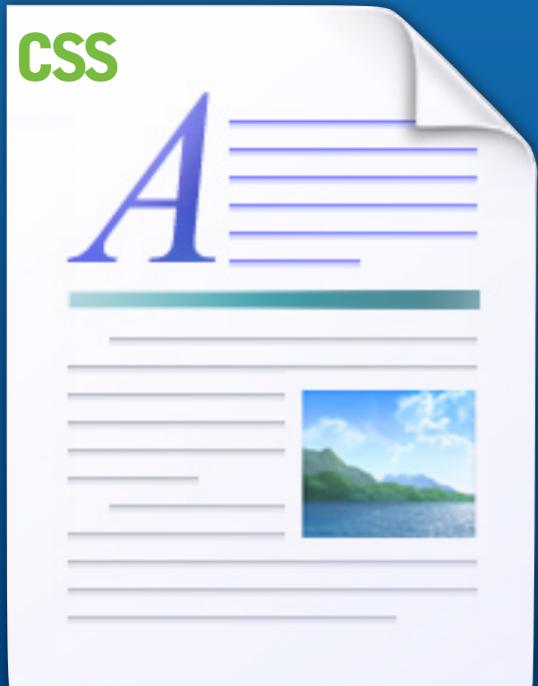
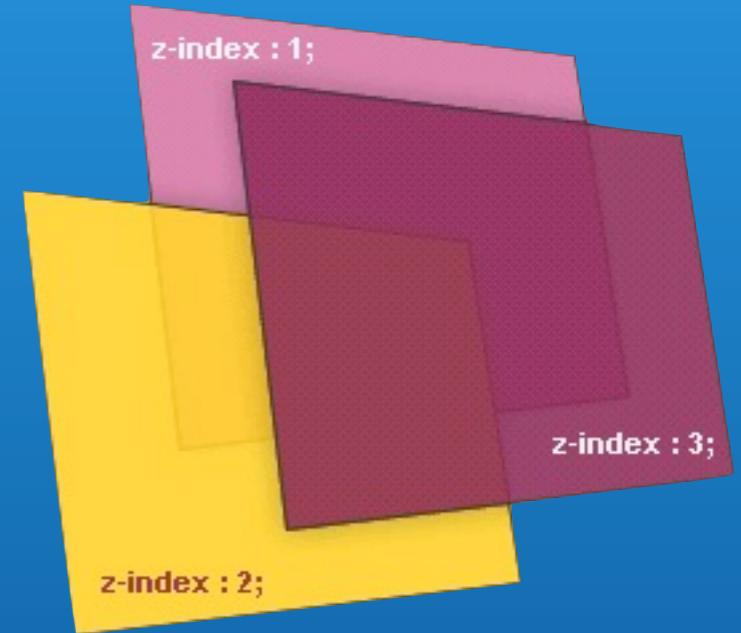
# CSS언어를 이용하여 대상의 위치지정후 띄워볼까요?



```
<head>
  <style>
    div {position: absolute; top:0; z-index:2}
  </style>
</head>
```



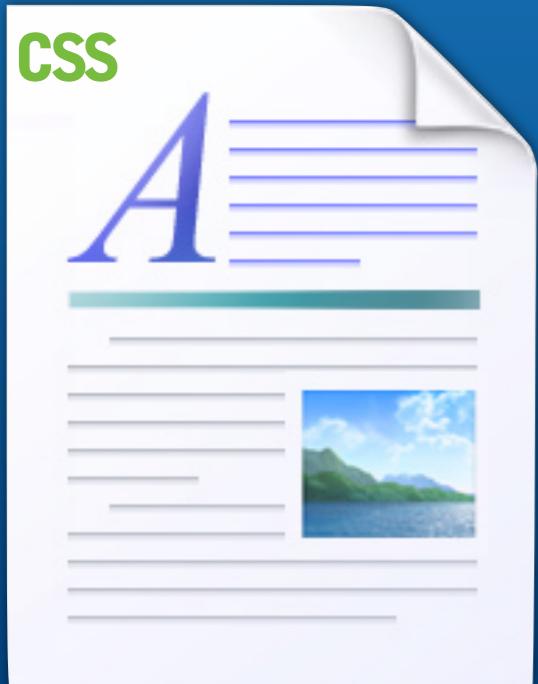
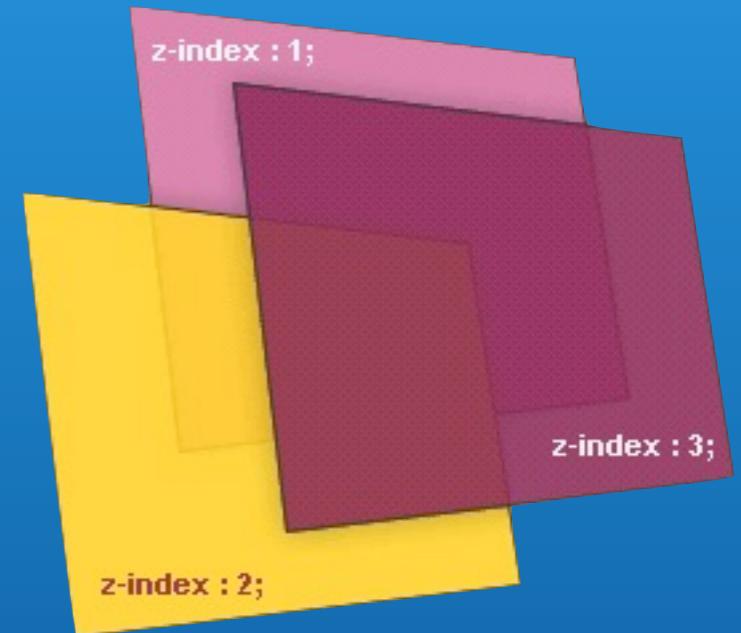
# CSS언어를 이용하여 대상의 위치지정후 띄워볼까요?



```
<head>
  <style>
    div {position: absolute; top:0; z-index:2}
  </style>
</head>
```



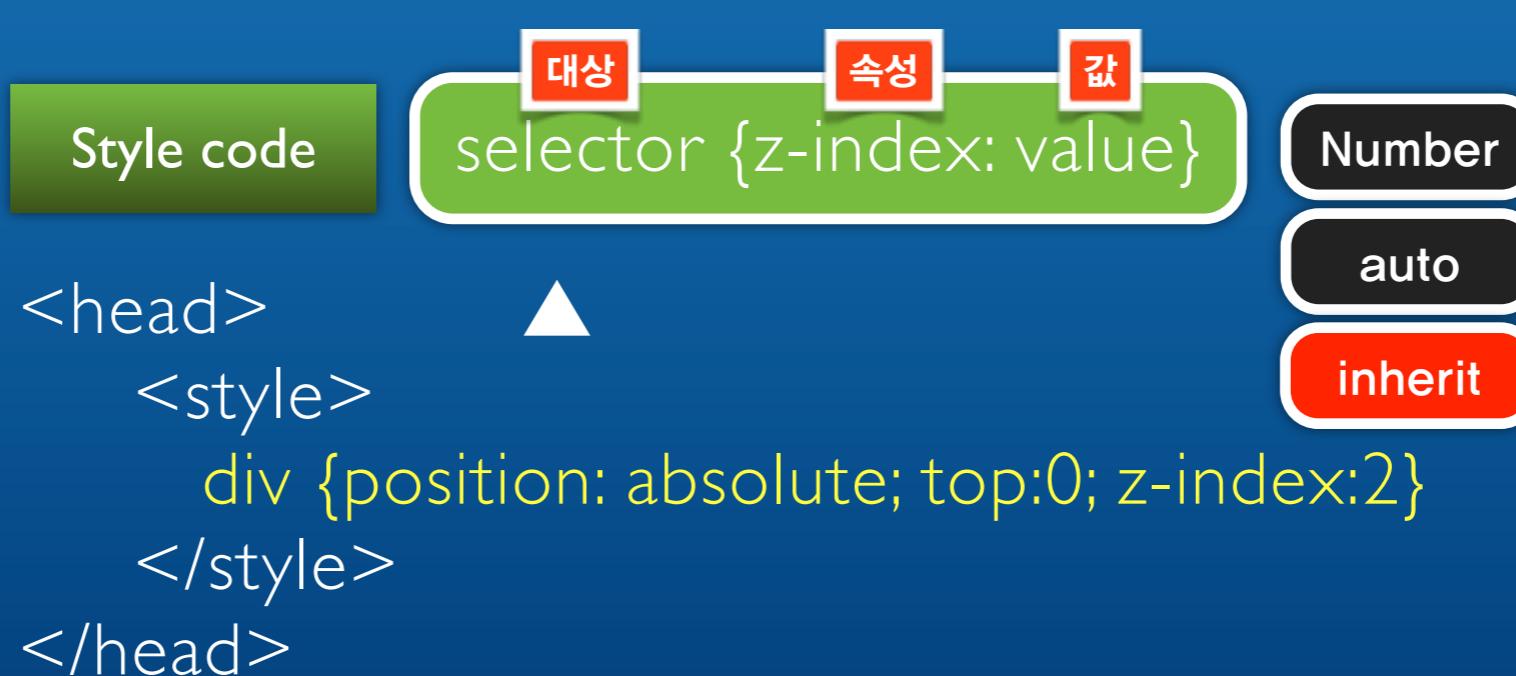
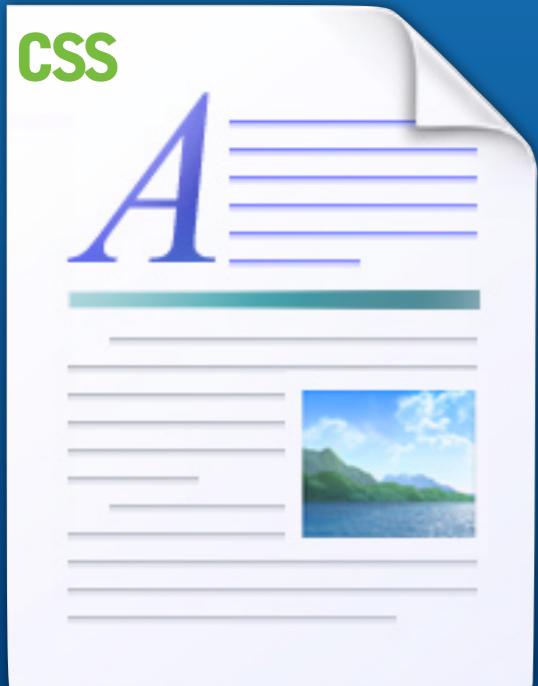
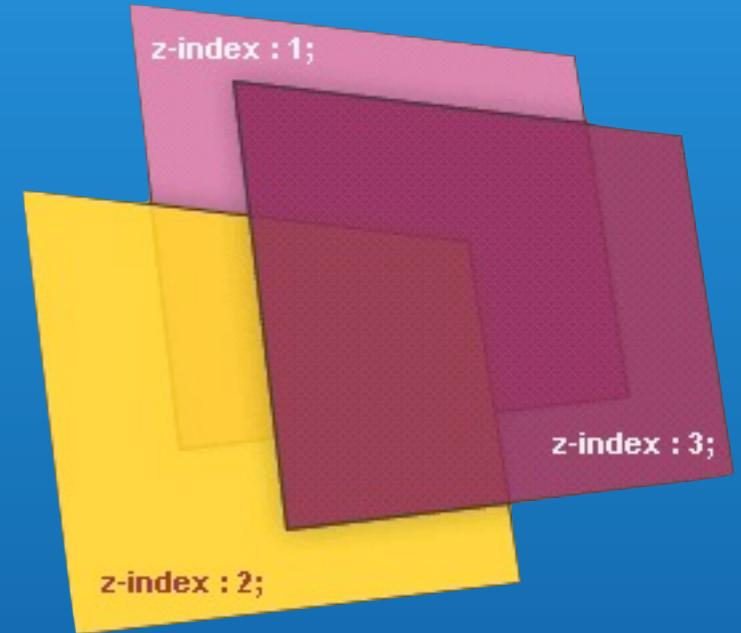
# CSS언어를 이용하여 대상의 위치지정후 띄워볼까요?



```
<head>
  <style>
    div {position: absolute; top:0; z-index:2}
  </style>
</head>
```



# CSS언어를 이용하여 대상의 위치지정후 띄워볼까요?



# CSS Position & z-index

# IE6.Bug z-index

The image shows a developer's workspace with multiple windows open:

- Coda Editor:** The main window displays a CSS file named "11\_IE-bug\_z-idx-fix-division\_begin.html". The code includes styles for a parent division (#division) and two child paragraphs (p#p1, p#p2) with absolute positioning and z-index values.
- File Explorer:** A sidebar window lists several files related to IE bugs, including "11\_IE-bug\_z-idx-fix\_division\_begin.html".
- Browsers:** Three browser windows are shown side-by-side:
  - IE6:** Shows a dark gray background with a white box containing "#division 안: 단락1".
  - IE7:** Shows a dark gray background with a white box containing "#division 안: 단락1".
  - Firefox:** Shows a dark gray background with a white box containing "#division 안: 단락2".