

6강. CSS 포지셔닝과 레이아웃



CSS 포지셔닝

CSS 포지셔닝이란?

- 브라우저 화면 안에 각 콘텐츠 영역을 어떻게 배치할지 결정하는 것
- float 속성과 position 속성이 있다.
- 박스 모델의 패딩이나 마진, 테두리 속성까지 포함해 전체적인 레이아웃이 완성 된다.

float 속성

- 요소를 왼쪽이나 오른쪽에 떠 있게 만듦
- float 속성을 사용하면 그 다음에 넣는 다른 요소들에도 똑같은 속성이 적용

float: left | right

속성 값	설명
left	해당 요소를 문서의 왼쪽으로 배치
right	해당 요소를 문서의 오른쪽으로 배치



float 속성

float 속성 예제.

박스1

박스2

박스3

박스4

```
<div class="box1">박스1</div>
<div class="box2">박스2</div>
<div class="box3">박스3</div>
<div class="box4">박스4</div>
```

float1.html

```
div{
  margin: 10px;
  padding: 10px;
  float: left;
  /*display: inline-block;*/
}
.box1{background: #ffff00;}
.box2{background: #ff0000;}
.box3{background: #00ff00;}
.box4{background: #ff00ff;}
```

float 속성





float 속성 예제.

박스1

박스2

박스3

박스4

```
div{  
  margin: 10px;  
  padding: 10px;  
}  
.box1{background:  #ffff00; float:left;}  
.box2{background:  #ff0000; float:left;}  
.box3{background:  #00ff00; float:left;}  
.box4{background:  #ff00ff; float:right;}
```

float 속성

clear 속성 - 해제하기

float 속성을 사용하면 그 다음에 넣는 다른 요소들에도 똑같은 속성이 적용되므로 해제하고 싶을 때 clear 속성을 사용한다.

clear : left | right | both



```
div{
  margin: 10px;
  padding: 10px;
}
.box1{background: #ffff00; float:left;}
.box2{background: #ff0000; float:left;}
.box3{background: #00ff00; float:right;}
.box4{background: #ff00ff; clear:both;}
```



float 속성

텍스트와 이미지 배치하기



왼쪽이나 오른쪽으로 배치하는 float 속성

웹 문서를 만들다 보면 문단과 이미지를 나란히 표시해야 할 경우가 있다. <P> 태그는 블록 레벨 요소이므로 이미지와 나란히 배치할 수 없다. 이럴때는 float 속성을 사용하여 이미지를 표시한다.

```
<div id="container">
  
  <h2>왼쪽이나 오른쪽으로 배치하는 float 속성</h2>
  <p>웹 문서를 만들다 보면 문단과 이미지를 나란히 표시해야 할 경우가 있다.
    &lt;P&gt; 태그는 블록 레벨 요소이므로 이미지와 나란히 배치할 수 없다.
    이럴때는 float 속성을 사용하여 이미지를 표시한다.
  </p>
</div>
```

float-text.html

```
#container{width: 1000px; margin: 0 auto;}
img{float: left; margin-right: 15px;}
p{line-height: 2em;}
```



2단 레이아웃 만들기

float 속성을 활용하여 레이아웃 만들기



2단 레이아웃 만들기

```
<body>
  <div id="container">
    <header>
      <h1>사이트 제목</h1>
    </header>

    <section>
      <h1>본문</h1>
    </section>

    <aside>
      <h1>사이드바</h1>
    </aside>

    <footer>
      <h1>푸터</h1>
    </footer>
  </div>
</body>
```

layout.html



2단 레이아웃 만들기

```
*{
  margin: 0; padding: 0; /*전체여백 초기화*/
}
#container{width:1000px; margin:0 auto;}
header{
  width: 978px; /*padding(20px) + border(2px)*/
  height: 100px;
  border: 1px solid #ccc;
  padding: 10px;
}
section{
  width: 620px;
  height: 600px;
  border: 1px solid #ccc;
  padding: 10px;
  float:left;
}
```

layout.css

```
aside{
  width: 320px;
  height: 600px;
  border: 1px solid #ccc;
  padding: 10px;
  background-color: #azure;
  float: right;
}
footer{
  width: 978px;
  height: 100px;
  border: 1px solid #ccc;
  padding: 10px;
  clear: both;
}
```



navbar 메뉴

PetDog

애완견 종류

입양하기

건강돌보기

더불어살기

애완견 종류

```
<div id="container">
  <header>
    <div id="logo">
      <h1>PetDog</h1>
    </div>
    <nav>
      <ul>
        <li><a href="#">애완견 종류</a></li>
        <li><a href="#">입양하기</a></li>
        <li><a href="#">건강돌보기</a></li>
        <li><a href="#">더불어살기</a></li>
      </ul>
    </nav>
  </header>
  <section>
    <h2>애완견 종류</h2>
  </section>
</div>
```

nav-pet.html



navbar 메뉴

pet.css

```
/* header 스타일*/
#container{width: 1000px; margin: 0 auto;}
header{height: 80px; background: #006699;}
header #logo{width: 200px; float: left; padding: 15px 20px;}
header h1{color: #fff; margin:0}
header h1:hover{color: #2fb2fd}
header nav{width: 700px; float:right;}
header nav ul{margin:0; list-style: none; text-align: right; margin-right: 60px; }
header nav ul li{display: inline-block; margin: 20px; padding: 10px;}
header nav ul li a{color: white; text-decoration: none;}
header nav ul li a:hover{color: #2fb2fd}

/* section 스타일 */
section{padding-left: 20px;}
```



navbar 메뉴

메뉴1

메뉴2

메뉴3

메뉴4

```
<nav>
  <ul>
    <li><a href="#">메뉴1</a></li>
    <li><a href="#">메뉴2</a></li>
    <li><a href="#">메뉴3</a></li>
    <li><a href="#">메뉴4</a></li>
  </ul>
</nav>
```

```
ul {list-style: none;}
ul li { float: left;}
a {
  display: block;
  padding: 10px 20px;
  background-color: #ccc;
  /* width: 100px; */
}
a:link, a:visited {color: black; text-decoration: none;}
a:hover {
  background-color: #000;
  color: #fff;
}
```



navbar 메뉴

메뉴1

메뉴2

메뉴3

메뉴4

```
ul {list-style: none;}
ul li { display: inline;}
a {
  display:inline-block;
  padding:10px 20px;
  background-color: #ccc;
  /* width: 100px; */
}
a:link, a:visited {color: black;text-decoration: none;}
a:hover {
  background-color: #000;
  color: #fff;
}
```



box-sizing

box-sizing 속성 - 박스 너비 기준 정하기

- **content-box** : width 속성 값을 콘텐츠 영역 너비 값으로만 사용한다.

예. { **box-sizing**: content-box }

- **border-box** : width 속성 값을 콘텐츠 + 테두리+패딩 영역까지 포함한 전체 너비 값으로 사용한다

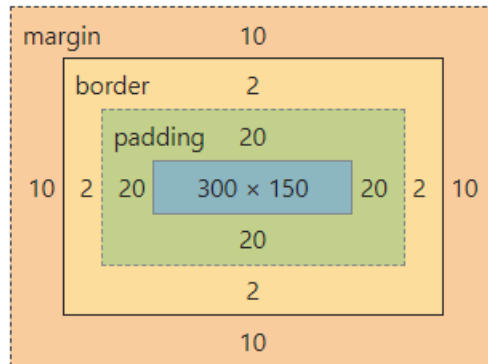
예. { **box-sizing**: border-box }



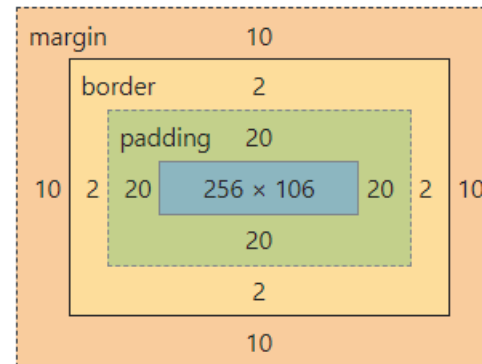
box-sizing

box-sizing 속성 – 박스 너비 기준 정하기

width=콘텐츠영역



width=콘텐츠영역+padding



box-sizing

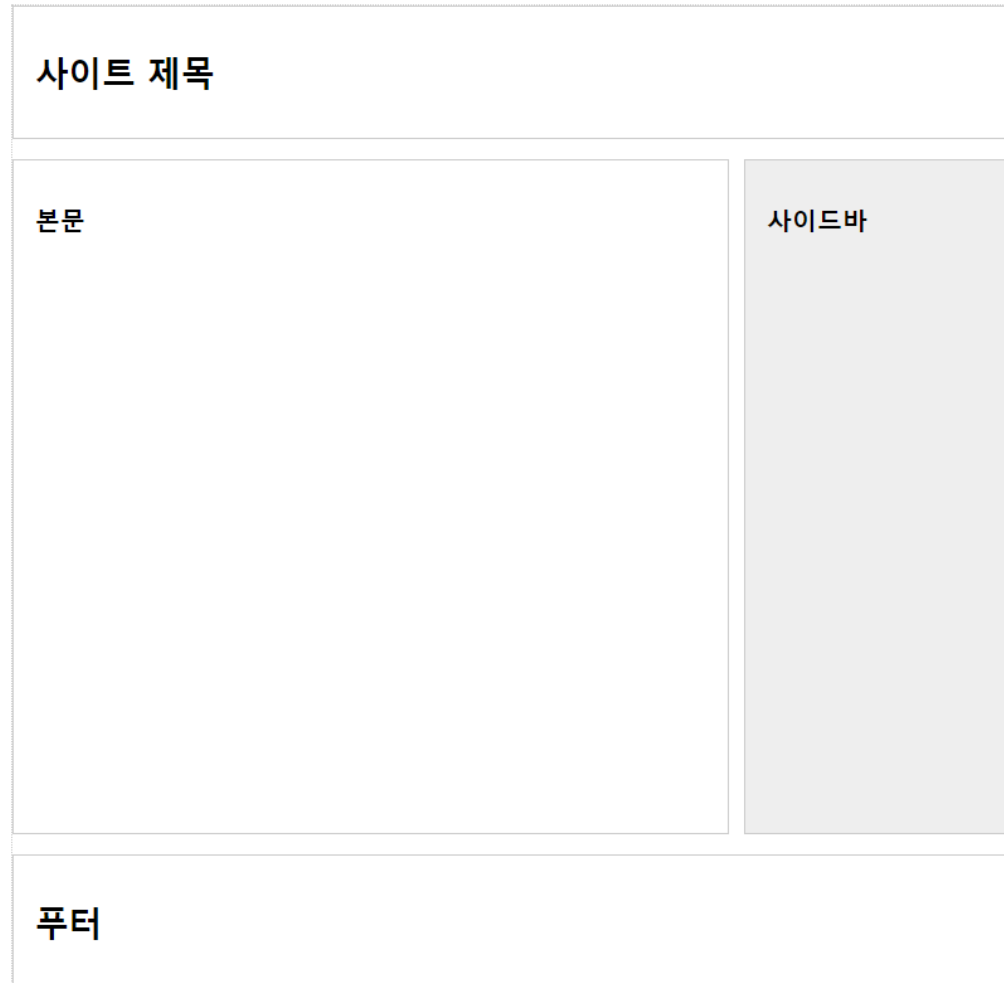
box-sizing.html

```
div{  
  width: 300px;  
  height: 150px;  
  border: 2px solid ■red;  
  margin: 10px;  
  display: inline-block;  
  padding: 20px;  
}  
#box1{  
  box-sizing: border-box;  
}  
#box2{  
  box-sizing: content-box;  
}
```



2단 레이아웃 만들기

box-sizing 적용



2단 레이아웃 만들기

```
<body>
  <div id="container">
    <header>
      <h1>사이트 제목</h1>
    </header>

    <section>
      <h1>본문</h1>
    </section>

    <aside>
      <h1>사이드바</h1>
    </aside>

    <footer>
      <h1>푸터</h1>
    </footer>
  </div>
</body>
```

layout2.html



2단 레이아웃 만들기

```
*{
  margin: 0; padding: 0;
  box-sizing: border-box;
  /* padding 값이 포함되어 계산됨 */
}
#container{width:1000px; margin:0 auto;}
header{
  width: 100%;
  height: 100px;
  border: 1px solid #ccc;
  padding: 10px;
}
section{
  width: 620px;
  height: 600px;
  border: 1px solid #ccc;
  padding: 10px;
  float:left;
}
```

layout2.css

```
aside{
  width: 320px;
  height: 600px;
  border: 1px solid #ccc;
  padding: 10px;
  background-color: #azure;
  float: right;
}
footer{
  width: 100%;
  height: 100px;
  border: 1px solid #ccc;
  padding: 10px;
  clear: both;
}
```



position 속성

position 속성

웹 문서 안에 요소들을 자유 자재로 배치하기 위한 속성
좌표를 이용해 각 요소를 배치할 수 있고, top, right, bottom, left로 지정

position: static | relative | absolute | fixed

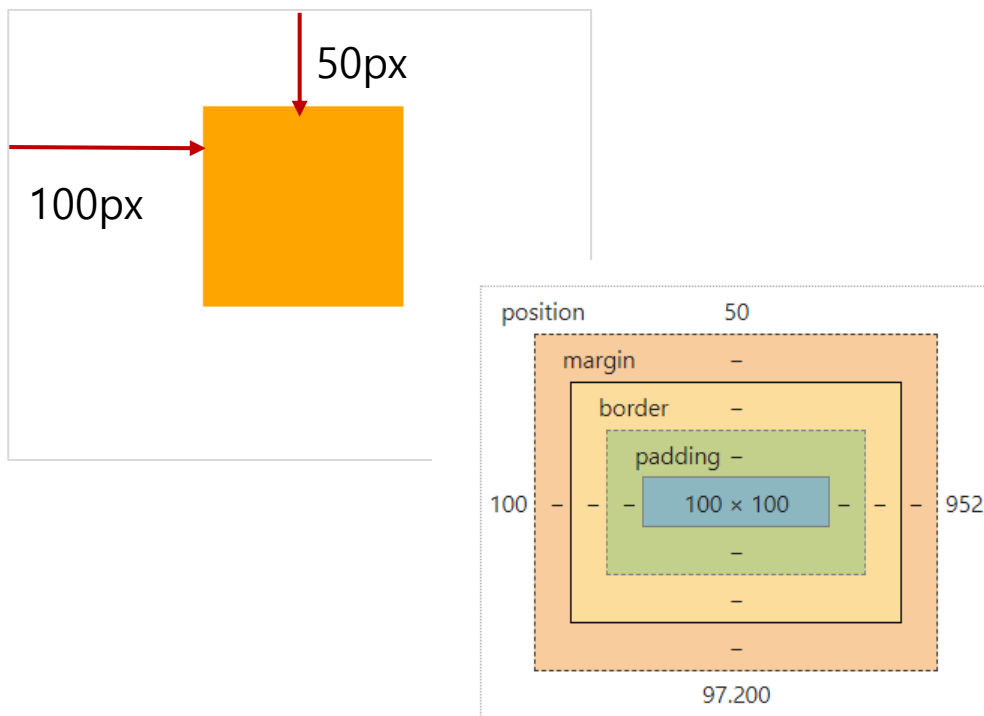
속성 값	설명
relative	이전 요소에 자연스럽게 연결해 배치하며 위치 지정 가능
absolute	원하는 위치를 지정해 배치 relative 값을 사용한 상위 요소를 기준으로 위치를 지정해 배치함
fixed	지정한 위치에 고정해 배치



position 속성

● absolute 속성

- 단독으로 사용하면 브라우저 창 기준
- 부모 요소가 relative이면 부모요소 기준

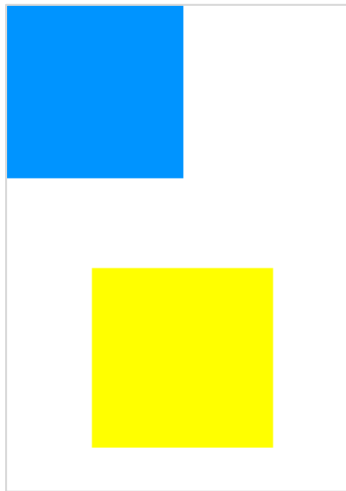


pos-abs.html

```
<style>
  div{
    width:100px;
    height: 100px;
    background: ■orange;
    position: absolute;
    left: 100px;
    top: 50px;
  }
</style>
```

position 속성

● relative 속성



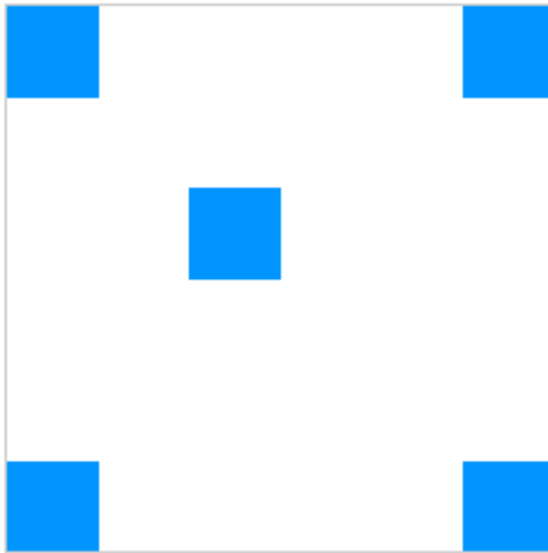
pos-rel.html

```
<body>
  <div class="box1">박스1</div>
  <div class="box2">박스2</div>
</body>
```

```
*{margin: 0; padding: 0;}
.box1{
  width: 100px;
  height: 100px;
  /* margin: 50px; */
  background: ■ #0094ff;
}
.box2{
  width: 100px;
  height: 100px;
  background: ■ #ffff00;
  position: relative;
  top: 50px;
  left: 50px;
}
```

position 속성

- absolute 속성(부모 요소가 relative인 경우)



pos-rel-abs.html

```
<div id="wrap">
  <div class="box" id="coord1"></div>
  <div class="box" id="coord2"></div>
  <div class="box" id="coord3"></div>
  <div class="box" id="coord4"></div>
  <div class="box" id="coord5"></div>
</div>
```

position 속성

- absolute 속성(부모 요소가 relative인 경우)

```
<style>
#wrap{
  width: 300px;
  height: 300px;
  border: 1px solid #ccc;
  position: relative;
}
.box{
  width: 50px;
  height: 50px;
  position: absolute;
  background: #0094ff;
}
#coord1{ top: 0; left: 0;}
#coord2{ top: 0; right: 0;}
#coord3{ bottom: 0; left: 0;}
#coord4{ bottom: 0; right: 0;}
#coord5{ left: 100px; top: 100px;}
</style>
```

기준이 되는
부모 요소



position 실습 예제



```
<title>CSS Position 속성</title>
<link rel="stylesheet" href="css/position-2.css">
</head>
<body>
  <div id="contents">
    <h1>CSS3</h1>
  </div>
</body>
```

position.html



position 속성

fixed 속성

- 문서의 흐름과 상관없이 원하는 위치에 요소를 배치
- 부모 요소가 아닌 **브라우저 창이 기준**이 됨
- 브라우저 창 화면을 스크롤 하더라도 계속 같은 위치에 고정

fixed 값을 사용하는 요소 역시 absolute 값을 사용하는 요소처럼 문서 흐름과는 상관없이 좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다 브라우저 창을 스크롤 하더라도 계속 고정되어 표시됩니다.

fixed 값을 사용하는 요소 역시 absolute 값을 사용하는 요소처럼 문서 흐름과는 상관없이 좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다 브라우저 창을 스크롤 하더라도 계속 고정되어 표시됩니다.

fixed 값을 사용하는 요소 역시 absolute 값을 사용하는 요소처럼 문서 흐름과는 상관없이 좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다 브라우저 창을 스크롤 하더라도 계속 고정되어 표시됩니다.



fixed 속성

position 속성


fixed 속성

```
<div id="container">
  <div id="content">
    <p>fixed 값을 사용하는 요소 역시 absolute값을 사용하는 요소처럼 문서 흐름과는 상관없이
    좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다
    브라우저를 스크롤하더라도 계속 고정되어 표시됩니다.</p>
    <p>fixed 값을 사용하는 요소 역시 absolute값을 사용하는 요소처럼 문서 흐름과는 상관없이
    좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다
    브라우저를 스크롤하더라도 계속 고정되어 표시됩니다.</p>
    <p>fixed 값을 사용하는 요소 역시 absolute값을 사용하는 요소처럼 문서 흐름과는 상관없이
    좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다
    브라우저를 스크롤하더라도 계속 고정되어 표시됩니다.</p>
    <p>fixed 값을 사용하는 요소 역시 absolute값을 사용하는 요소처럼 문서 흐름과는 상관없이
    좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다
    브라우저를 스크롤하더라도 계속 고정되어 표시됩니다.</p>
    <p>fixed 값을 사용하는 요소 역시 absolute값을 사용하는 요소처럼 문서 흐름과는 상관없이
    좌표로 위치를 결정하지만, 기준이 되는 요소가 부모요소가 아니라 브라우저 창이 기준이 됩니다
    브라우저를 스크롤하더라도 계속 고정되어 표시됩니다.</p>
  </div>
  <div id="fix"></div>
</div>
```



position 속성

fixed 속성

```
#container{width: 1000px; margin: 0 auto;}
#fix{
  width: 100px;
  height: 160px;
  background:  pink;
  background: url(./images/bg3.png);
  position: fixed;
  top: 20px;
  right: 20px;
}
#content{width: 500px;}
```

position 속성

fixed 속성

앵커 만들기

웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고, 그 위치로 한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)라고 한다.

- [도서 정보](#)
- [리뷰/한줄평](#)
- [배송 정보](#)

도서 정보

웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고, 그 위치로 한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)라고 한다.

웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고, 그 위치로 한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)라고 한다.

웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고, 그 위치로 한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)라고 한다.

웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고, 그 위치로 한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)라고 한다.

문의 전화

032-262-0600

- 평일: 09:00~18:00
- 주말: 09:00~13:00



CSS 포지셔닝

<p>웹 문서가 너무 길 경우 필요한 곳마다 문서 안에 이름을 붙여놓고,
한 번에 이동하는 링크를 만들 수 있는데, 이 기능을 앵커(anchor)

```
<p><a href="#menu">[메뉴로]</a></p>
```

anchor.html

```
<div id="fix">  
  <div class="cs-center">  
    <!--  -->  
    <h4>문의 전화</h4>  
    <p>032-262-0600</p>  
    <ul>  
      <li>평일: 09:00~18:00</li>  
      <li>주말: 09:00~13:00</li>  
    </ul>  
  </div>  
</div>
```



CSS 포지셔닝

```
#content{width: 800px; margin:0 auto;}
#fix{
  width: 150px;
  position: fixed;
  top: 50px;
  right: 100px;
  /*background: yellow;*/
  border: 3px solid ■green;
  border-radius: 10px;
}
#fix .cs-center h4{
  margin: 10px 15px;
  background: url("../images/dot.png") no-repeat center left;
  padding-left: 20px;
}
#fix .cs-center p{padding-left: 20px; margin: 0; font-weight: bold;}
#fix .cs-center ul{margin: 0; padding: 10px; list-style: square;}
#fix .cs-center ul li{margin:10px 10px 10px 15px; font-size: 0.8em;}
```

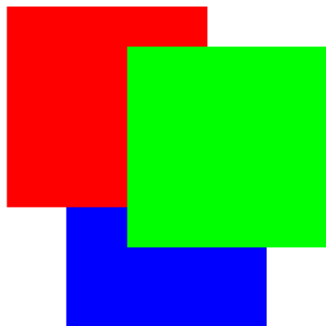
anchor.css



position 속성

z-index 속성

- 요소 쌓는 순서 정하기
- z-index 값이 크면 작은 요소보다 위에 쌓인다.
- z-index 값을 명시하지 않으면 1부터 시작 1씩 커진다.



```
<body>  
  <div class="box1"></div>  
  <div class="box2"></div>  
  <div class="box3"></div>  
</body>
```


position 속성

```
/* index 숫자가 클수록 위로 올라옴 */
div{
  width: 100px;
  height: 100px;
  font-size: 2em;
  position: absolute;
}
.box1{
  background: ■ #ff0000;
  top: 50px;
  left: 50px;
  z-index: 2;
}
.box2{
  background: ■ #00ff00;
  top: 70px;
  left: 110px;
  z-index: 3;
}
.box3{
  background: ■ #0000ff;
  top: 110px;
  left: 80px;
  z-index: 1;
}
```



Overflow 속성 정리

overflow 속성

- 요소의 박스에 내용(content)이 더 길때 어떻게 보일지를 결정하는 속성

overflow속성 - visible

CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.

- visible : 특정 요소가 박스를 넘어가도 그대로 보여준다.
- hidden : 부모 요소의 범위를 넘어가는 자식 요소의 부분은 보이지 않는다.
- auto : 내용이 넘치면 스크롤바를 표시한다.

overflow속성 - hidden

CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.

- visible : 특정 요소가 박스를 넘어가도 그대로 보여준다.

overflow속성 - auto

CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.

- visible : 특정 요소가 박스를 넘어가도 그대로 보여준다

Overflow 속성 정리

```
<div id="ex1">
  <h2>overflow속성 - visible</h2>
  <p>CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.<br>
    - visible : 특정 요소가 박스를 넘어가도 그대로 보여준다.<br>
    - hidden : 부모 요소의 범위를 넘어가는 자식 요소의 부분은 보이지 않는다.<br>
    - auto : 내용이 넘치면 스크롤바를 표시한다.</p>
</div>
<div id="ex2">
  <h2>overflow속성 - hidden</h2>
  <p>CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.<br>
    - visible : 특정 요소가 박스를 넘어가도 그대로 보여준다.<br>
    - hidden : 부모 요소의 범위를 넘어가는 자식 요소의 부분은 보이지 않는다.<br>
    - auto : 내용이 넘치면 스크롤바를 표시한다.</p>
</div>
<div id="ex3">
  <h2>overflow속성 - auto</h2>
  <p>CSS에서 overflow 속성은 요소의 박스에 내용이 더 길때 어떻게 보일지를 선택하는 속성이다.<br>
    - visible : 특정 요소가 박스를 넘어가도 그대로 보여준다.<br>
    - hidden : 부모 요소의 범위를 넘어가는 자식 요소의 부분은 보이지 않는다.<br>
    - auto : 내용이 넘치면 스크롤바를 표시한다.</p>
</div>
```



Overflow 속성 정리

overflow.css

```
div{
  width: 300px;
  height: 200px;
  margin: 20px;
  padding: 10px;
  float: left;
}
#ex1{
  overflow: visible;
  background: lightgreen;
}
#ex2{
  overflow: hidden;
  background: orange;
}
#ex3{
  overflow: auto;
  background: pink;
}
```

