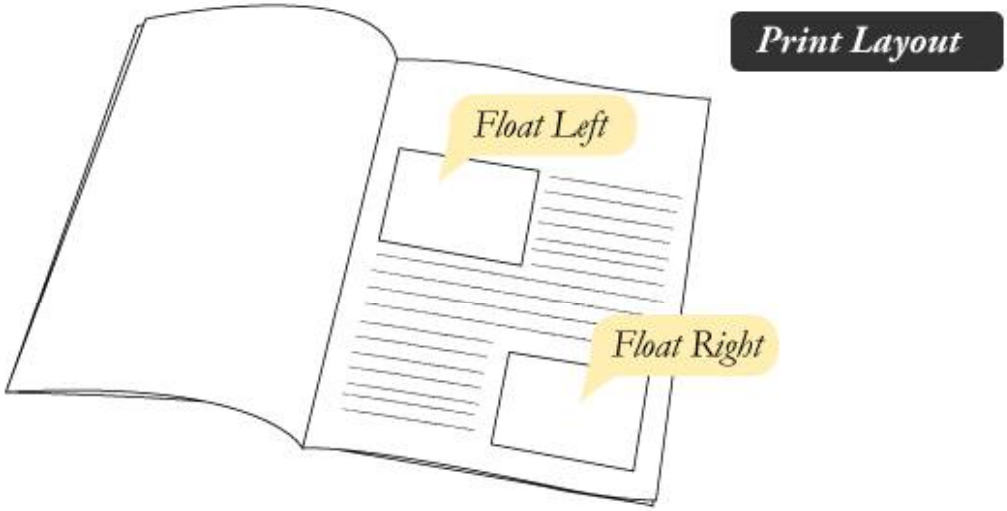
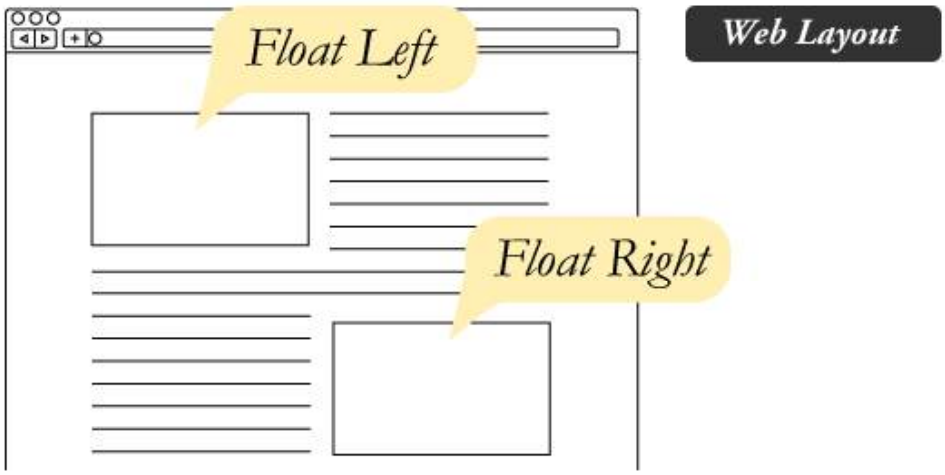
CSS
Float layout

“Float” 은

CSS 위치관련 속성이다. 목적과 기원을 이해하기 위해서는 인쇄디자인을 살펴보면 된다. 인쇄레이아웃에서 이미지는 아래 그림과 같이 텍스트에 둘러 쌓이게 되며 이것을 일반적으로 text wrap” 이라고 말한다.



이러한 텍스트랩(Text lab)은 아래 그림과 같이 박스와 같은 영역으로 페이지레이아웃의 텍스트 흐름(flow)속에 있거나 혹은 별도로 위치하게 된다. 웹디자인도 이와 유사하다.



웹 디자인에서 CSS float 속성은 인쇄레이아웃에서 이미지와 같이 해당 이미지 근처의 텍스트 흐름에 영향을 받게된다. Float된 요소는 (floated elements) 웹페이지의 flow의 한 부분으로 남으며 이것은 고정된 위치 값을 가진 요소들과 명백히 다르다. 위치가 고정된 페이지요소는 웹페이지내의 흐름속에서 제외되며 다른 요소에도 영향을 끼치지 못한다.

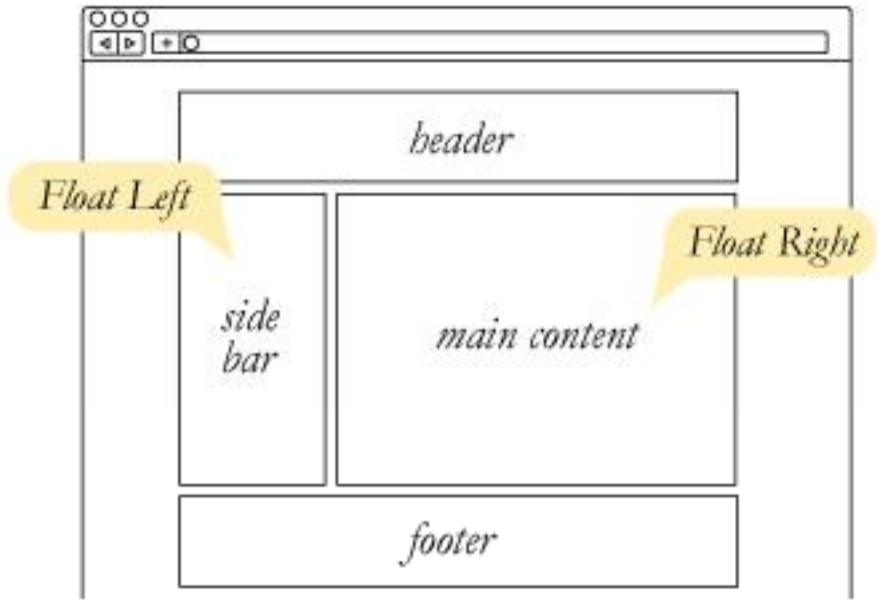
float속성은 다음과 같이 사용한다.

```
#sidebar {
   float: right;
}
```

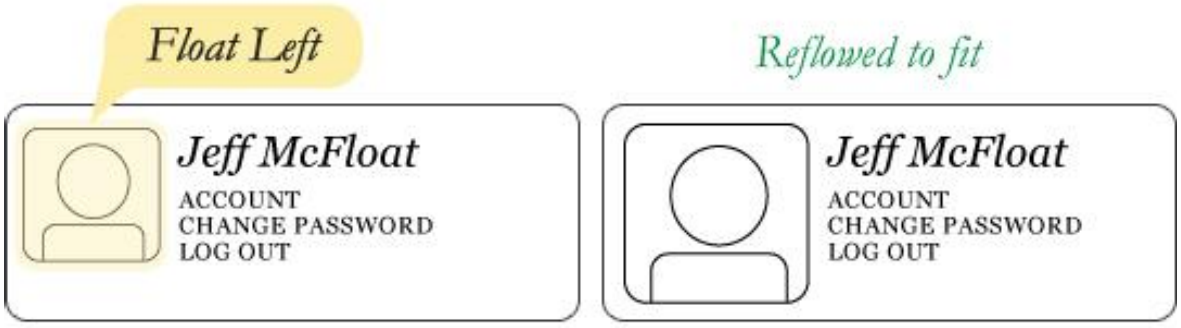
float속성은 4개의 유효값을 가지고 있다 좌우방향을 표시하는 left와 right, 기본디폴트값인 none, 그리고 부모요소로 부터 상속(cascading)된 inherit속성이 그것이다.

CSS
Float layout

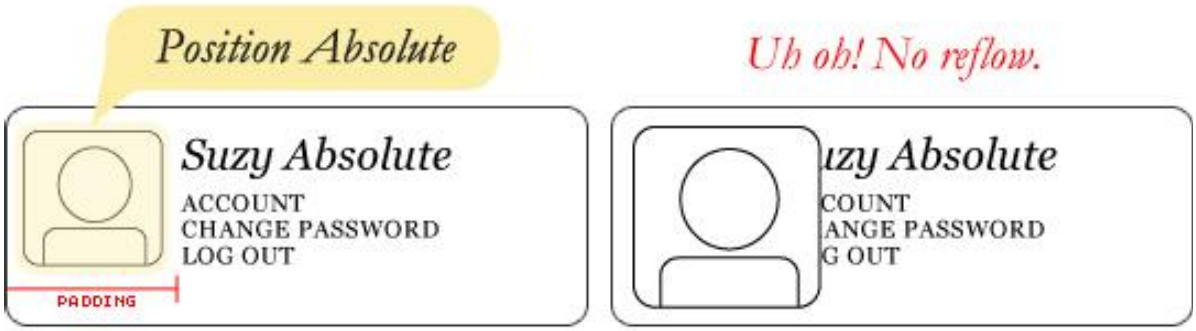
Float속성은 어디에 사용되는가?



float속성값은 이미지를 둘러싸고 있는 텍스트에 사용되나 위 그림과 같이 전체 웹레이아웃을 생성하는데 사용되기도 한다.
또한 아래와 같이 전체레이아웃과 유사한 작은 구성 영역에도 사용된다.
아래의 예는 아바타 이미지와 해당 계정의 내용을 표시한것으로 float 속성을 사용한것이다.
아바타 이미지가 커지더라도 적절한 간격을 유지하며 포지셔닝된다.



그러나 위와 같이 같은 레이아웃이라도 상대적인 위치나 절대적위치값을 이용하여 사용한다면 아래 그림과 같이 이미지사이즈가 변할경우 레이아웃 자체가 깨질수 있다.
데꾸벅 주: 대부분의 경우 디자이너나 에디터의 경우 relative, absolute로 작업하는 경향이 많다. 이는 한국형 웹페이지(?)만의 특징이라고 볼수도 있겠지만 가장 화면레이아웃을 쉽고 빠르게 셋팅할수 있기 때문이다. 이러할 경우 웹표준으로 작업하더라도 그만큼의 효과를 얻지 못한다는 것이다.



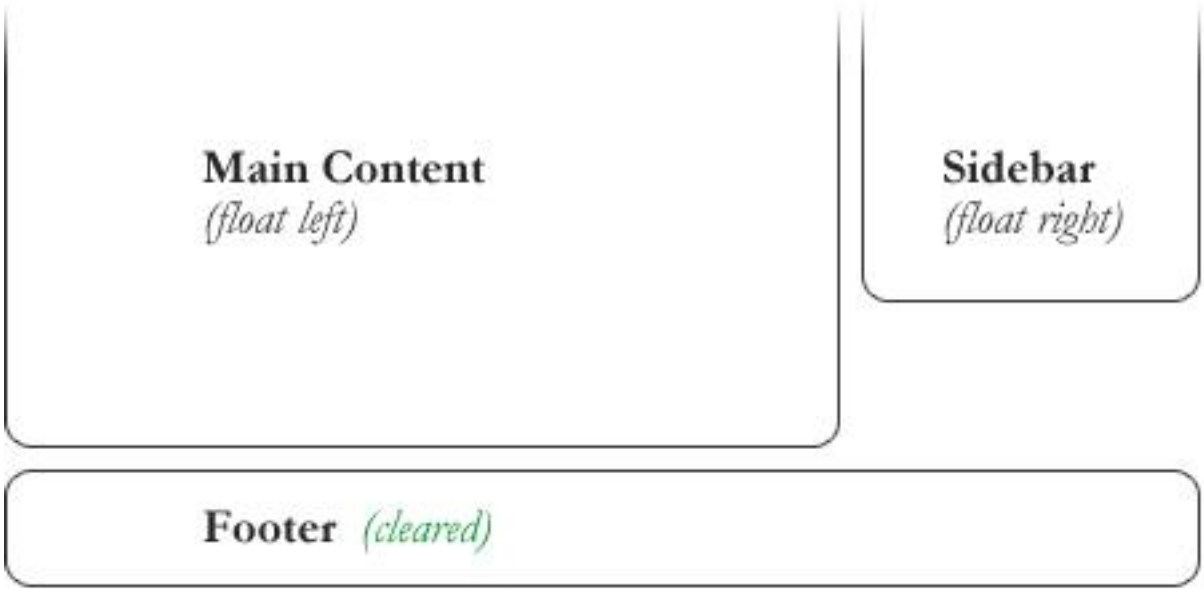
CSS
Float layout

흐름제거 (Clearing the Float)

float의 속성의 자매속성으로 clear가 있다. HTML 요소는 float과 같이 흐름속성이 있다면 이와 같은 흐름을 제거하는 속성(clear)를 가지고 있다

위그림에서 보는바와 같이 사이드바(sidebar)는 메인컨텐츠(Main content)영역보다 짧게 오른쪽으로 float이 되어있으며 footer영역이 main content영역만큼을 채우려고 올라와 있다. 이와같은 문제를 해결하려면 아래와 같이 footer에 clear 속성으로 흐름을 제어하면 된다.

```
#footer {
   clear: both;
}
```

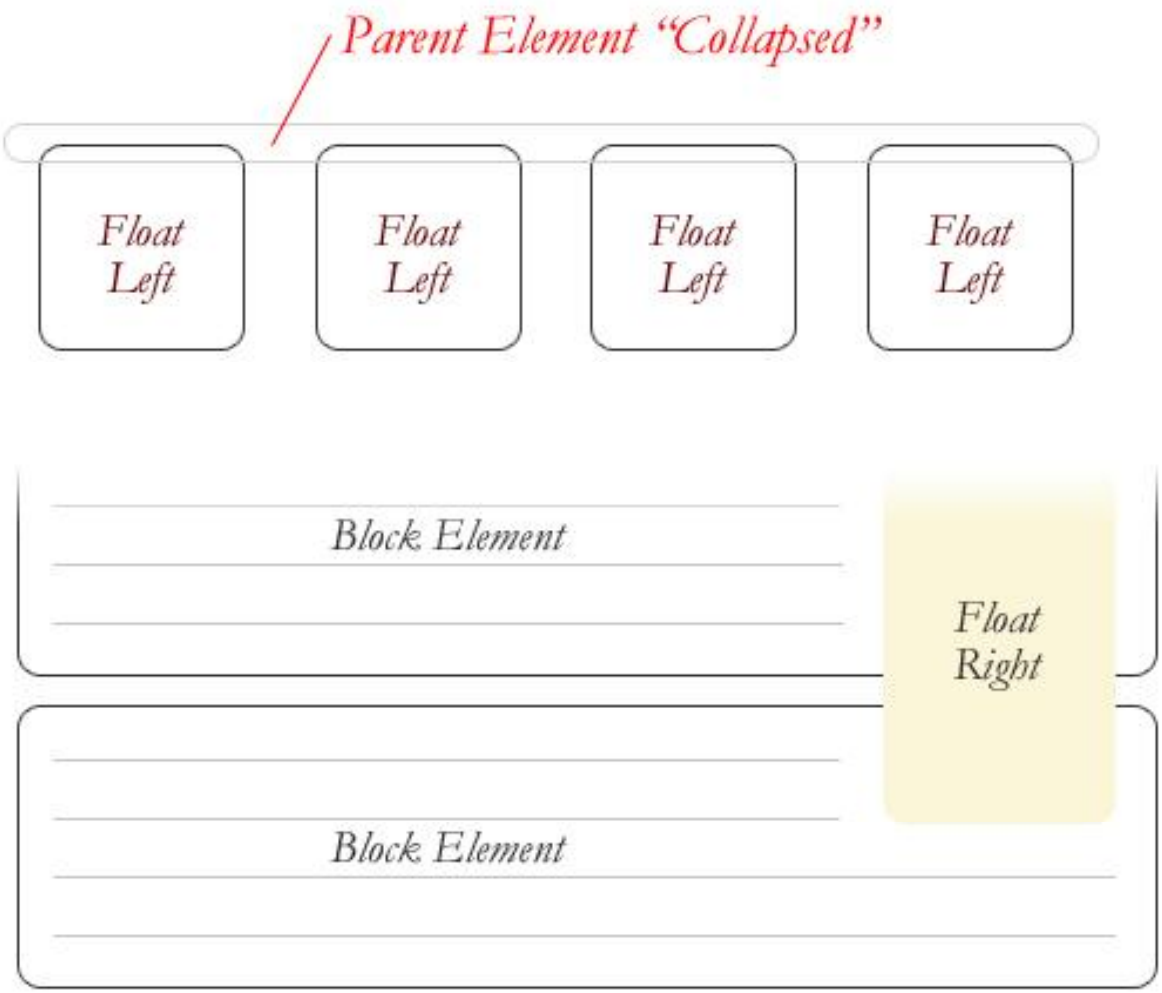


clear속성은 4가지 속성값을 가지고 있는데 보통 양쪽흐름을 끊는 “Both” 속성을 많이 사용한다. 왼쪽과 오른쪽 흐름을 끊는 “left”, “right” 와 기본값으로 “none”, 부모객체의 속성값을 상속(cascading)받는 “inherit” 이 있다.

CSS
Float layout

겹침문제

float속성을 사용해서 작업하다보면 곤혹스런경우를 많이 당하게 되는데 아래 그림과 같이 float속성을 가진 객체의 부모객체밖으로 삐져나오는 경우가 있다



흐름을 끊는 기법(Techniques for clearing Floats)

clear기법을 사용하는 방법으로 아래와 같이 여러가지 방법이 사용되고 있으나 상황에 맞게 적절하게 사용하여야 한다.

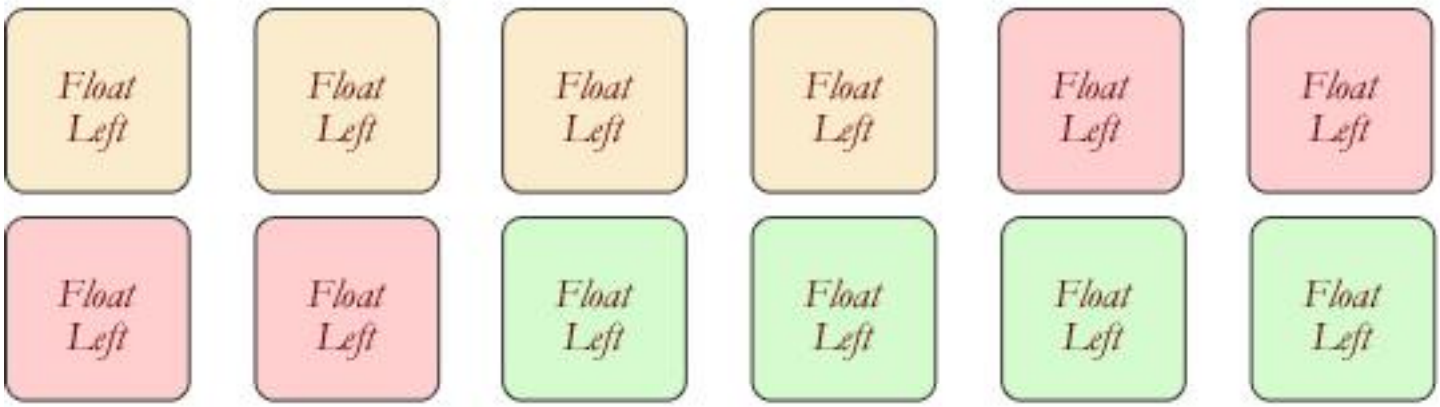
■ **빈div 태그에 직접 clear속성 넣기** : 〈div style=" clear:both;" 〉〈/div〉를 이용하는 것으로 〈br /〉과 같은 효과를 준다. 그러나 구조적인 마크업의 흐름을 깨게 되는 경우가 생기므로 사용에 주의하여야 한다.

■ **overflow 방법** : 부모객체에 overflow속성을 넣는 방법으로 float되는 객체의 부모객체의 넓이를 조정하여 흐름을 제어하는 방법이다. 그러나 이 방법은 원하지 않는 스크롤바를 생성할 수도 있다.
(참조사이트 : wystan' s tables)

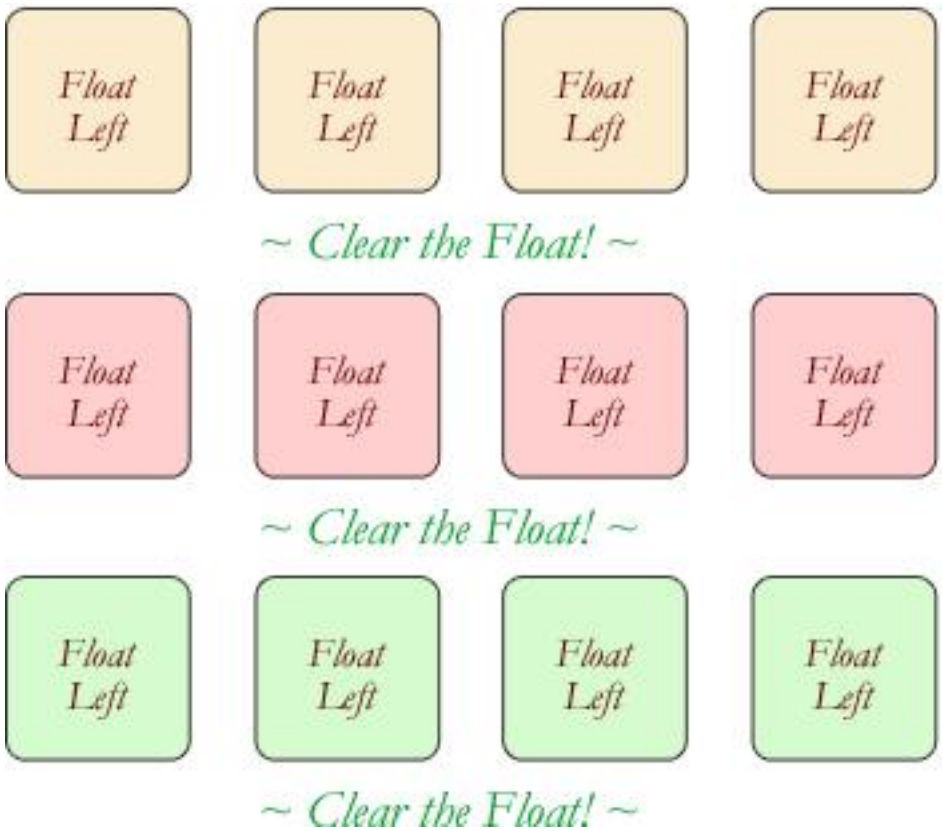
■ **가장쉬운 clear 방법** : 가상선택자(pseduo selector) :after를 이용하여 float을 clear할수 있다. float되는 객체의 부모객체에 다음과 같이 clearfix 클래스를 줌으로써 간단하게 해결할 수 있으며 오래된 브라우저도 수용할 수 있다.

CSS
Float layout

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



위 그림과 같이 서로 다른 float 속성이 먹은 객체들이 나열되어 있을때 그룹화하여 흐름을 제어하고 싶다면 아래 그림과 같이 그룹화된 부모객체에 위의 방법을 쓸 수 있다.



CSS
Float layout

Float속성의 문제점

Float속성을 쓰면 화면이 깨지기 쉽다. IE6은 float과 관련된 버그를 가지고 있다. 점점더 많은 디자이너들이 IE6지원을 줄여가고 있는 시점이나 그렇다고 포기할만한 시점도 아니다.

- **밀려내려감(Pushdown)** : 아래 그림과 같이 밀려내려가는 증상은 float된 객체가 영역보다 클때 일어난다. 이럴 경우는 overflow:hidden을 사용하여 처리할수 있다.
- **더블마진버그(Double Margin Bug)** : IE6에서 inline-level element가 같은 방향으로 float된 경우 일어난다. 이럴경우는 해당 요소를 display:inline을 써서 inline-level요소로 만들어 주면된다.
- **3픽셀 밀림 버그(3px Jog)** :IE 6에서 CSS 먹일때 Float + Margin or Padding 값을 동시에 먹이면. Margin,Padding 값이 두배가 되는버그가 있는데 이상하게도 float된 요소의 다음에 오는 텍스트에 오는 버그로 이럴경우 text의 width와 height값을 주면 해결된다.
- **IE7의 Bottom Margin 버그** : float된 부모객체 안의 자식객체에 나타나는 버그로 부모객체에 의해 자식객체의 bottom-margin을 무시하게 되는 버그로써 부모객체에 bottom padding을 줘서 해결해야 한다.

Float속성 사용시 지켜야 할점

Float을 사용시 지켜야 할 내용들이다. 크로스 브라우징에 가장 잘 다가가기 위해서 생각해야하는 항목을 적어놨다.

- **밀려내려가면 width를 확인하라** : Float속성을 사용했는데 밀려서 내려갈 경우는 width가 모자랄 경우이다. 직접적인 E의 밀림도 떨어지지만 자손 E의 넓이도 영향을 받는 경우도 있다. 밀려내려갈 경우에는 반듯이 Boxmodel의 넓이값을 계산하여야 한다.
- **Float가 필요한 모든 E에 써준다** : Float가 필요한 E(가로형태가 되는 E)는 모두에게 Float속성을 사용한다. 부분적으로 사용할 경우 크로스브라우징을 할때 문제가 생기는 경우가 있다.
- **Float는 모두 left를 써준다.** : Float는 right값을 가지게 되면 순서가 바뀌게되기 때문에 float는 모두 left값을 써준다. 단 마지막 E경우에는 right를 써줘도 무방하다.
- **더이상 Float가 필요없으면 Clear!** : float된 객체아래에서 더이상 Float속성을 받지 않고 싶은경우에는 Clear을 써서 흐름을 끈어줘야한다.
- **부모 E에 높이값이 없으면?** : float된 객체의 부모E에 높이값이 유동적일경우 흐름을 끈어서 높이값을 만들어준다.