**方法一：**添加新的元素 、应用 clear：both；

<div class="topDiv"> <div class="textDiv">...</div>

<div class="floatDiv">float left</div>

<div class="blankDiv"></div> </div>

<div class="bottomDiv">...</div>

.blankDiv {
clear: both; // or left
}

.textDiv告诉浏览器，我的周围不允许有浮动的元素存在，请清除掉我周围的浮动元素。然而，因为浮动元素（.floatDiv）位置已经确定，浏览器在计算.textDiv的位置时，为满足其需求，将.textDiv渲染在浮动元素下方，保证了.textDiv周围没有浮动元素。同时可以看出，父元素的高度也被撑起来了，其兄弟元素的渲染也不再受到浮动的影响，这是因为.textDiv仍然在文档流中，它必须在父元素的边界内，父元素只有增加其高度才能达到此目的，可以说是一个意外收获。(clear的值为both也有相同的效果，通俗理解就是，哪边不允许有浮动元素，clear就是对应方向的值，两边都不允许就是both)

**利用overflow清除浮动**

.topDiv {//父元素
width: 500px;
padding: 4px;
border: 2px solid black;
// 区别在这里
overflow: auto;
}

**方法三：** 据说是最高大上的方法 :after 方法：（注意：作用于浮动元素的父亲）

先说原理：这种方法清除浮动是现在网上最拉风的一种清除浮动，他就是利用:after和:before来在元素内部插入两个元素块，从面达到清除浮动的效果。其实现原理类似于clear:both方法，只是区别在于:clear在html插入一个div.clear标签，而outer利用其伪类clear:after在元素内部增加一个类似于div.clear的效果。下面来看看其具体的使用方法：

两栏布局 左固定 右边自适应

#left{

width: 300px;

height: 100%;

background-color: #ccc;

float: left;

}

#right{

height: 100%;

margin-left: 300px;

background-color: #eee;

}

#left{

width: 300px;

height: 100%;

background-color: #ccc;

float: left;

}

#right{

height: 100%;

overflow:hidden;

background-color: #eee;

}