### 1. 水平居中

* 方法一：margin:0 auto; （最常用的居中布局方式）
* 方法二：text-align和inline-block的结合（设置父元素的text-align为center, 子元素display:inline-block）。

text-align这种方式最好应用于图片、按钮、文字之类的居中模式，否则就需要借助inline-block来进行居中布局。

* 方法三：position绝对定位来实现居中布局。

适用于块级元素不给出宽高的情况下(需要借助transtrom的tanslateX方法)。

**#parent**{

position: relative;

}

**#child**{

position: absolute;

top: 0;

left: 50%;

transform: translateX(-50%);

}

* 方法四：利用flex弹性布局的一个属性。

子元素宽度已知的情况下:

**#parent**{

display: flex;

justify-content: center;

}

* 其他还有很多方法，一般用的不太多。并且各种方法优缺点不一样，可选择性使用。

### 2. 水平垂直居中

* 方法一：先说一种神奇的方式吧，
  1. 子元素 div 绝对定位
  2. 父元素需要被定位
  3. 子元素 top、bottom、left、right 四个位置值均为 0
  4. 子元素 margin: auto;

**#parent**{

width: 100%;

height:100%;

position: fixed;

}

**#child**{

width: 400px;

height: 200px;

position: absolute;

top: 0;

bottom: 0;

left: 0;

right: 0;

margin: auto;

background-color: #ccc;

}

* 方式二：利用position的绝对定位及负边框来实现。

**#parent**{

position: fixed;

width: 100%;

height: 100%;

}

**#child**{

position: absolute;

left: 50%;

top: 50%;

width: 400px;

height: 200px;

margin-top: -100px;

margin-left: -200px;

background-color: #ccc;

}

对于未给出宽高的元素，又需要transform，同时需要做好各浏览器的兼容。

* 方式三：利用flex弹性布局的

设置父元素的css:display: flex; justify-content: center; align-items: center;

### 3. 左边固定右边自适应的两列布局

* 方式一：float+margin的方式

这种方式一定要记得给父元素清除浮动啊，不然就尴尬了呢，这里插播一种全局性(这个词似乎不太对)的清除浮动的伪元素方法.

**.clearfix:after {**

content: '.';

height: 0;

overflow: hidden;

clear: both;

display: block;

visibility: hidden;

}

**.clearfix {**

zoom: 1; <!--hack-->

}

借助触发BFC的方式来解决（偷偷告诉你，常用的方式就是给你的父元素设置**overflow: hidden;**）。

布局代码：

**#left**{

float: left;

width: 100px;

}

**#right**{

margin-left: 120px;

}

* 方式二：float+overflow的方式

这就是传说中利用BFC的规则来实现两列布局啊啊啊！

这种方式直接不需要清除浮动什么的就可以了，也不会影响前后文

**#left{**

float: left;

width: 100px;

margin-right: 20px;

}

**#right{**

overflow: hidden;

}

* 方式三：float+margin+position的方式

这个方式呢也用到过，但是要考虑的比较多一点，不过其实也还好。  
接下来请看实现代码：

**#parent**{

position: relative;

}

**#left**{

float: left;

width: 100px;

background-color: #ccc;

}

**#right**{

position: absolute;

top: 0;

left: 120px;

background-color:pink;

}

* 方式四：flex方式

**#parent**{

display: flex;

}

**#left**{

width: 100px;

margin-right: 20px;

}

**#right**{

flex: 1;

}

### 4. 左边自适应右边固定

* 方式一： position

**#parent** {

position: relative;

}

**#left** {

margin-right:220px;

}

**#right** {

position: absolute;

right:0;

top:0;

width: 200px;

}

### 5. 两边固定中间自适应的三列布局

其实这个布局用的也挺多的啊哈，嗯，昨天写的作业就是这个！

* 方式一：纯float方式

注意：

* 1. 左侧元素与右侧元素优先渲染，分别向左和向右浮动
  2. 中间元素在文档流的最后渲染，自动插入到左右两列元素的中间，随后设置 margin 左右边距分别为左右两列的宽度，将中间元素调整到正确的位置。

.left{

float: left;

width: 200px;

height: 200px;

}

.right{

float: right;

width: 100px;

height: 100px;

}

.middle{

margin:0 120px 0 220px;

}

但凡用float的时候都要想一下父元素上清除浮动这个问题！

* 方式二：position的绝对定位

其实感觉跟float的原理差不多，都是将左右两侧的块先固定好，再对中间部分进行处理，只不过自己可以在不同情况下选择float或者position。

.parent{

position: relative;

}

.left{

position: absolute;

width: 200px;

height: 200px;

top: 0;

left: 0;

}

.right{

position: absolute;

top: 0;

right: 0;

width: 100px;

height: 100px;

}

.middle{

margin:0 120px 0 220px;

}

* 方式三：flex的弹性布局

不得不说的是其实很多布局都可以用flex来实现(简单粗暴嘿嘿)，但是flex的兼容性不是很好，并且还有别问题，所以保险起见还是选择常用的，这里简单介绍下。

.parent{

display: flex;

}

.left{

width: 200px;

height: 200px;

}

.right{

width: 100px;

height: 100px;

}

.middle{

flex: 1;

margin:0 20px;

}

* 方式四：最后该淘宝的双翼布局闪亮登场了

要注意的是这种布局方式需要将主栏优先渲染，然后再加上两边的翅膀，即双翼，不过话又说话来，虽然小花是按照这个套路写的，但也不确定自己写的就是双翼布局。

* 第一步，先将主栏左浮动，并设宽度为100%，即铺满父元素
* 第二步，将左栏左浮动，并设左外边距为-100%
* 第三步，将右栏左浮动，并设左外边距为负的自身宽度
* 第四步，设置主栏，嘿嘿，这时候不管你使用什么方法都达不到效果，解决办法就是给主栏的内容加一个包裹层，并设左右外边距为左右两侧的宽度。
* 第五步，好了，完美！（但是别忘了，要清除浮动哦~）