**[前端精选文摘：BFC 神奇背后的原理](http://www.cnblogs.com/lhb25/p/inside-block-formatting-ontext.html)**

　　BFC 已经是一个耳听熟闻的词语了，网上有许多关于 BFC 的文章，介绍了如何触发 BFC 以及 BFC 的一些用处（如清浮动，防止 margin 重叠等）。虽然我知道如何利用 BFC 解决这些问题，但当别人问我 BFC 是什么，我还是不能很有底气地解释清楚。

**一、BFC是什么？**

　　在解释 BFC 是什么之前，需要先介绍 Box、Formatting Context的概念。

**Box: CSS布局的基本单位**

　　Box 是 CSS 布局的对象和基本单位， 直观点来说，就是一个页面是由很多个 Box 组成的。元素的类型和 display 属性，决定了这个 Box 的类型。 不同类型的 Box， 会参与不同的 Formatting Context（一个决定如何渲染文档的容器），因此Box内的元素会以不同的方式渲染。让我们看看有哪些盒子：

* block-level box:display 属性为 block, list-item, table 的元素，会生成 block-level box。并且参与 block fomatting context；
* inline-level box:display 属性为 inline, inline-block, inline-table 的元素，会生成 inline-level box。并且参与 inline formatting context；
* run-in box: css3 中才有， 这儿先不讲了。

**Formatting context**

　　Formatting context 是 W3C CSS2.1 规范中的一个概念。它是页面中的一块渲染区域，并且有一套渲染规则，它决定了其子元素将如何定位，以及和其他元素的关系和相互作用。最常见的 Formatting context 有 Block fomatting context (简称BFC)和 Inline formatting context (简称IFC)。

　　CSS2.1 中只有 BFC 和 IFC, **[CSS3](http://www.cnblogs.com/lhb25/category/146075.html" \t "_blank)**中还增加了 GFC 和 FFC。

**BFC 定义**

　　BFC(Block formatting context)直译为"块级格式化上下文"。它是一个独立的渲染区域，只有Block-level box参与， 它规定了内部的Block-level Box如何布局，并且与这个区域外部毫不相干。

**BFC布局规则：**

1. 内部的Box会在垂直方向，一个接一个地放置。
2. Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠
3. 每个元素的margin box的左边， 与包含块border box的左边相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。
4. BFC的区域不会与float box重叠。
5. BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。
6. 计算BFC的高度时，浮动元素也参与计算

**二、哪些元素会生成BFC?**

1. 根元素
2. float属性不为none
3. position为absolute或fixed
4. display为inline-block, table-cell, table-caption, flex, inline-flex
5. overflow不为visible

**三、BFC的作用及原理**

**1. 自适应两栏布局**

　　代码：

<style>

body {

width: 300px;

position: relative;

}

.aside {

width: 100px;

height: 150px;

float: left;

background: #f66;

}

.main {

height: 200px;

background: #fcc;

}

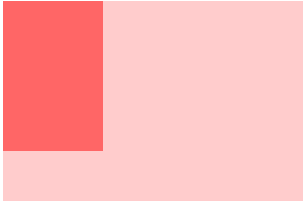
</style>

<body>

<div class="aside"></div>

<div class="main"></div>

</body>

　　页面：  
　　

　　根据BFC布局规则第3条：

每个元素的margin box的左边， 与包含块border box的左边相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。

　　因此，虽然存在浮动的元素aslide，但main的左边依然会与包含块的左边相接触。

　　根据BFC布局规则第四条：

BFC的区域不会与float box重叠。

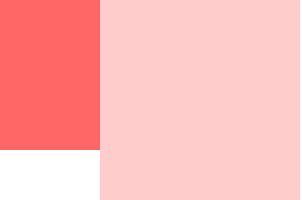
　　我们可以通过通过触发main生成BFC， 来实现自适应两栏布局。

.main {

overflow: hidden;

}

　　当触发main生成BFC后，这个新的BFC不会与浮动的aside重叠。因此会根据包含块的宽度，和aside的宽度，自动变窄。效果如下：

**

**2. 清除内部浮动**

　　代码：

<style>

.par {

border: 5px solid #fcc;

width: 300px;

}

.child {

border: 5px solid #f66;

width:100px;

height: 100px;

float: left;

}

</style>

<body>

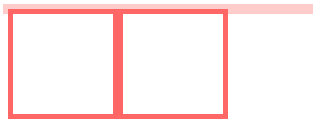
<div class="par">

<div class="child"></div>

<div class="child"></div>

</div>

</body>

　　页面：  
　　

　　根据BFC布局规则第六条：

计算BFC的高度时，浮动元素也参与计算

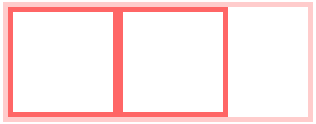
　　为达到清除内部浮动，我们可以触发par生成BFC，那么par在计算高度时，par内部的浮动元素child也会参与计算。

　　代码：

.par {

overflow: hidden;

}

　　效果如下：  
　　 ￼

**3. 防止垂直 margin 重叠**

　　代码：

<style>

p {

color: #f55;

background: #fcc;

width: 200px;

line-height: 100px;

text-align:center;

margin: 100px;

}

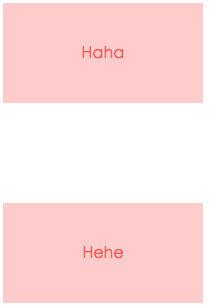
</style>

<body>

<p>Haha</p>

<p>Hehe</p>

</body>

　　页面：  
　　

　　两个p之间的距离为100px，发送了margin重叠。  
　　根据BFC布局规则第二条：

Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠

　　我们可以在p外面包裹一层容器，并触发该容器生成一个BFC。那么两个P便不属于同一个BFC，就不会发生margin重叠了。  
　　代码：

<style>

.wrap {

overflow: hidden;

}

p {

color: #f55;

background: #fcc;

width: 200px;

line-height: 100px;

text-align:center;

margin: 100px;

}

</style>

<body>

<p>Haha</p>

<div class="wrap">

<p>Hehe</p>

</div>

</body>

　　效果如下:  
　　

**总结**

　　其实以上的几个例子都体现了BFC布局规则第五条：

BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。

　　因为BFC内部的元素和外部的元素绝对不会互相影响，因此， 当BFC外部存在浮动时，它不应该影响BFC内部Box的布局，BFC会通过变窄，而不与浮动有重叠。同样的，当BFC内部有浮动时，为了不影响外部元素的布局，BFC计算高度时会包括浮动的高度。避免margin重叠也是这样的一个道理。