**普通流、浮动流、定位流**

**1、普通流**

**div、h1 或 p 元素常常被称为块级元素。**

这意味着这些元素显示为一块内容，即“块框”

CSS 有三种基本的定位机制：普通流、浮动和绝对定位。

除非专门指定，否则所有框都在普通流中定位。也就是说，普通流中的元素的位置由元素在 (X)HTML 中的位置决定。

块级框从上到下一个接一个地排列，框之间的垂直距离是由框的垂直外边距计算出来。

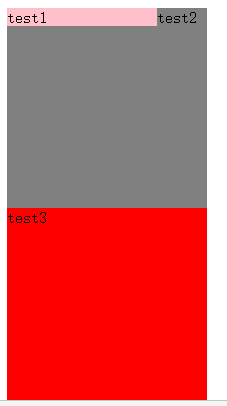
行内框在一行中水平布置。可以使用水平内边距、边框和外边距调整它们的间距。但是，垂直内边距、边框和外边距不影响行内框的高度。由一行形成的水平框称为行框（Line Box），行框的高度总是足以容纳它包含的所有行内框。不过，设置行高可以增加这个框的高度。

**2、Float用法（浮动流）**

浮动的框可以向左或向右移动，直到它的外边缘碰到包含框或另一个浮动框的边框为止。由于浮动框不在文档的普通流中，所以文档的普通流中的块框表现得就像浮动框不存在一样。

（参考<http://www.w3school.com.cn/css/css_positioning_floating.asp>）

文档的普通流中的块框表现得就像浮动框不存在一样，但是浮动框旁边的行框被缩短，从而给浮动框留出空间，行框围绕浮动框。参考如下例子1。



<html>

<head>

<style type="text/css">

#d1

{

float:left;

width:150px;

background-color: pink;

}

#d2

{

width:200px;

height:200px;

background-color: gray;

}

#d3

{

width:200px;

height:200px;

background-color: red;

}

</style>

</head>

<body>

<div id="d1">

test1

</div>

<div id="d2">

test2

</div>

<div id="d3">

test3

</div>

</body>

</html>

例1

如果不想行框围绕浮动框，可以使用“clear”，#d2增加clear:left;如例2（效果像没用float？？？）

<html>

<head>

<style type="text/css">

#d1

{

float:left;

width:100px;

height:150px;

background-color: pink;

}

#d2

{

width:200px;

height:100px;

background-color: gray;

**clear1:left;**

}

#d3

{

width:200px;

height:200px;

background-color: red;

}

</style>

</head>

<body>

<div id="d1">

test1

</div>

<div id="d2">

test2 test2 test2

test2 test2 test2



</div>

<div id="d3">

test3 test3 test3

test3 test3 test3

test3 test3 test3

test3 test3 test3

test3 test3 test3

test3 test3 test3

test3 test3 test3

test3 test3 test3

</div>

</body>

</html>

例2

**3、CSS position 属性（定位流）**

通过使用 position 属性，我们可以选择 4 种不同类型的定位，这会影响元素框生成的方式。

position 属性值的含义：

static

元素框正常生成。块级元素生成一个矩形框，作为文档流的一部分，行内元素则会创建一个或多个行框，置于其父元素中。

relative

元素框偏移某个距离。元素仍保持其未定位前的形状，它原本所占的空间仍保留。

absolute

元素框从文档流完全删除，并相对于其包含块定位。包含块可能是文档中的另一个元素或者是初始包含块。元素原先在正常文档流中所占的空间会关闭，就好像元素原来不存在一样。元素定位后生成一个块级框，而不论原来它在正常流中生成何种类型的框。

fixed

元素框的表现类似于将 position 设置为 absolute，不过其包含块是视窗本身。

提示：相对定位实际上被看作普通流定位模型的一部分，因为元素的位置相对于它在普通流中的位置。

**补充（总结）说明：**

1、如果没有修饰的div，用了margin-xxx: auto;则该div会自动中对齐，width必须赋值

2、网页中，最外的父div应该是中对齐，则网页内容可以保证都显示在网页的中间

因此需要对例子进行居中处理

3、绝对定位是相对于父div的，如果用绝对定位div的父div没有任何的定位和浮动修饰，该绝对定位效果不是针对于父div的，

解决的方式可以在父div上根据实际情况加相对定位或绝对定位，且left和top=0

4、绝对定位的div会在普通流中删除，不占用位置，相对的，相对定位div不会在普通流中删除，仍然占用原来的位置

5、不同方法时候宽度规则：

（1）什么都不加的div默认继承父块的宽度

（2）float div的宽度默认是内容的宽度

（3）absolute的宽度默认是内容的宽度

（4）relative的宽度默认是继承父块的宽度（和不加占位一样）

因此（1）和（4）用来实现满屏宽度情况下实现自动宽度（自动宽度=文档宽度-margin-left-margin-right，不需要设置宽度值，自动由浏览器计算而来）

6、不同方法时候高度规则：

（1）什么都不加的div默认是内容的高度（内部的DIV relative占，float、absolute不占，但float加了<div class="clear"></div>清除浮动占），设置高度百分比不起作用（默认高度），像素高度起作用

（2）float div的高度默认是内容的高度（内部DIV float和relative占，absolute不占），设置高度百分比不起作用，像素高度起作用

（3）absolute的高度默认是内容的高度（内部DIV float和relative占，absolute不占），设置高度百分比起作用，像素高度起作用

（4）relative的高度默认是内容的高度（内部的DIV，relative占，float、absolute不占，但float加了<div class="clear"></div>清除浮动占），设置高度百分比不起作用（默认高度），像素高度起作用

可以看出，div默认规则是relative。

因此实现充满高度必须要设置height，控制充满高度比较困难

应用总结：

最外部的container采用relative，方便居中，且方便宽度固定或自适应，高度需要自适应的话需要js控制高度

内部根据需求选择