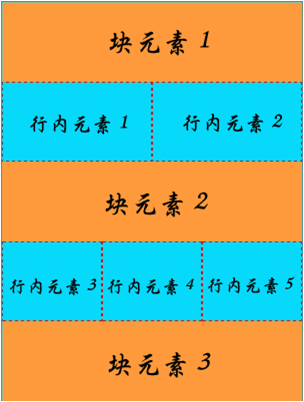
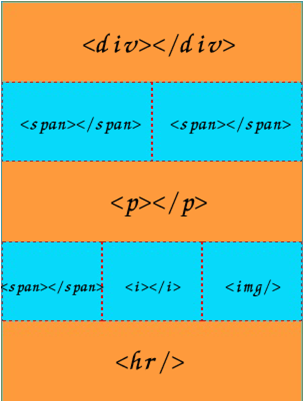
# **正常文档流**

## **一、文档流**

### **1、什么是“正常文档流”？**

在学习浮动布局之前，我们先来认识一下什么叫“正常文档流”？深入了解正常文档流，对后续的浮动布局和定位布局是非常重要的一个前提。什么叫文档流？简单来说，就是元素在页面出现的先后顺序。

那什么叫“正常文档流”呢？我们先来看一下正常文档流的简单定义：正常文档流，将窗体自上而下分成一行一行，块元素独占一行，相邻行内元素在每行中按从左到右地依次的排列元素。

上面的HTML代码的文档流如下：

|  |  |
| --- | --- |
| 1  2  3  4  5 | <div><div>  <span></span><span><span>  <p></p>  <span></span><i><i><img/>  <hr/> |

因为div、p、hr都是块元素，因此独占一行。而span、i、img都是行内元素，因此如果两个行内元素相邻，就会会位于同一行，并且从左到右排列。

### **2、什么叫“脱离正常文档流”？**

脱离文档流是相对正常文档流而言的。正常文档流就是我们没有用CSS样式去控制的HTML文档结构，你写的界面的顺序就是网页展示的顺序。比如写了5个div元素。正常文档流就是按照依次显示这5个div元素。由于div元素是块元素，因此每个div元素独占一行，正常排列下来的页面布局就叫作正常文档流

|  |  |
| --- | --- |
| 1  2  3  4  5 | <div id="div1"></div>  <div id="div2"></div>  <div id="div3"></div>  <div id="div4"></div>  <div id="div5"></div> |



**图1 正常文档流显示**

上图1就是按照正常文档流显示的效果图。

然后，所谓的脱离文档流就是指它所显示的位置和文档代码的顺序不一致了，比如可以用CSS控制，把最后一个div元素显示在第一个div元素的位置，如下图：



**图2 脱离文档流显示**

在图2中，在不改变HTML代码顺序的前提下，我们可以通过CSS来将id="div5"的div元素从正常文档流“抽”出来，然后显示在其他div元素之前。在这种情况下，id="div5"的div元素就已经“脱离正常文档流”了。在CSS布局中，我们可以使用浮动或者定位这两种技术来实现“脱离正常文档流”，从而随心所欲地控制着页面的布局。

# **浮动float**

## **一、浮动float**

在之前“文字环绕--float那一页里”，我们已经接触过浮动是怎样一回事了。

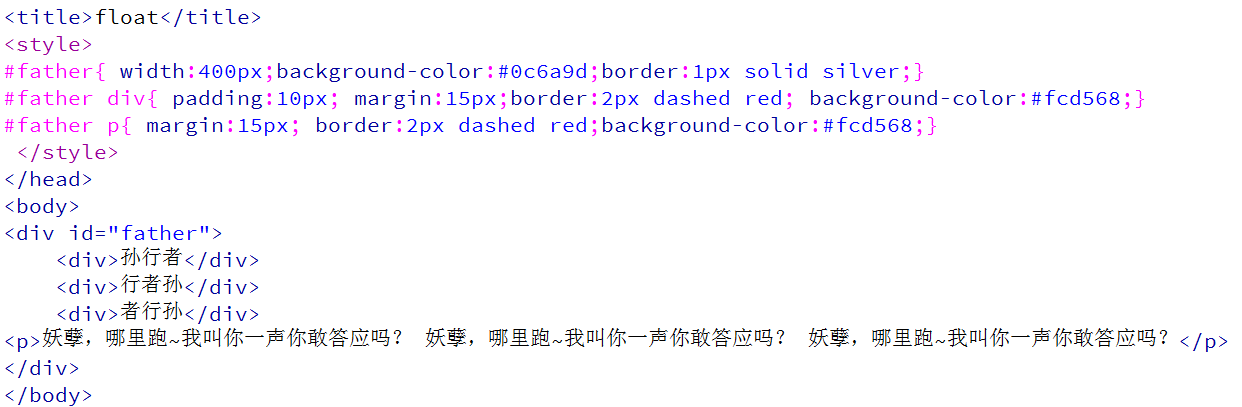
在传统的印刷布局中，文本可以按照实际需要来围绕图片。一般把这种方式称为“文本环绕”。在网页设计中，应用了CSS的float属性的页面元素就像在印刷布局里被文字包围的图片一样。

浮动属性float是CSS布局的最佳利器，我们可以通过不同的浮动属性值灵活地定位div元素，以达到布局网页的目的。我们可以通过CSS的属性float使元素向左或向右浮动。也就是说，让盒子及其中的内容浮动到文档的右边或者左边。以往这个属性总应用于图像，使文本围绕在图像周围，不过在CSS中，任何元素都可以浮动。浮动元素会生成一个块级框，而不论它本身是何种元素。

float:取值;

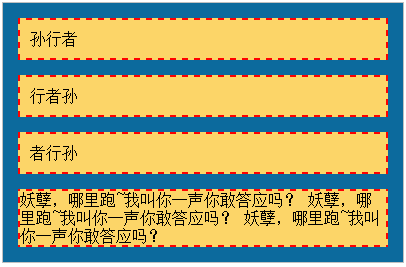
| **表1 float属性** | |
| --- | --- |
| **float属性值** | **说明** |
| left | 元素向左浮动 |
| right | 元素向右浮动 |

默认情况下，元素是不浮动的。

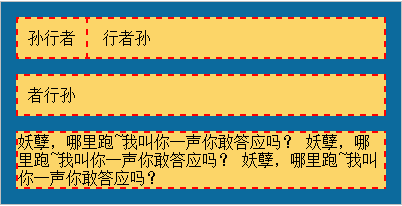


上面代码定义了4个div块，一个是父块，另外3个是它的子块。为了便于观察，将各个块都加上了边框以及背景颜色，并且让body以及各个div有一定的margin（外边距）。

如果3个子块都没有设置浮动方式，在父盒子中，由于div元素是块元素，因此4个盒子各自向右伸展，



### **1、设置第1个div浮动**



由于第一个孙行者设置为左浮动，所以孙行者变成了浮动元素，因此此时孙行者的宽度不再延伸，其宽度为容纳内容的最小宽度，而相邻的下一个div元素（行者孙）就会紧贴着第一个，这是由于浮动引起的效果。

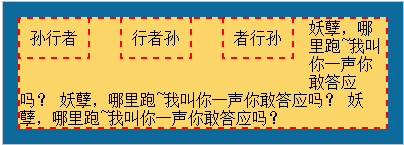
### **设置第2个div浮动**

### 

由于行者孙变成了浮动元素，因此第二个也跟第一个一样，宽度不再延伸，而是由内容确定宽度。并且相邻的下一个div元素（者行孙）变成紧贴着浮动的行者孙。

大家会觉得很奇怪，为什么这个时候1和2之间有一定的距离呢？其实原因是这样的：我们在CSS中设置了1、2和3都有一定的外边距（margin:15px;），如果1为浮动元素，而相邻的2不是浮动元素，则box2就会紧贴着1；但是如果1和2同时为浮动元素，外边距就会生效。这是由于浮动元素的特性决定的。

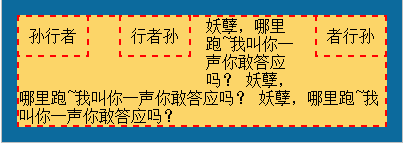
### **3、设置第3个div浮动**



由于者行孙变成了浮动元素，因此3跟2和1一样，宽度不再延伸，而是由内容确定宽度，并且相邻的下一个p元素变成紧贴着浮动的者行孙。

### **4、改变浮动的方向**

在这里，我们将3浮动方式改为“float:right”



float属性是CSS布局中非常重要的属性，我们常常通过对div元素应用float浮动来进行布局，不但对整个版式进行规划，也可以对一些基本元素，如导航等进行排列。

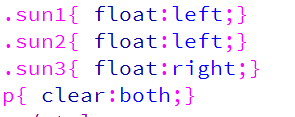
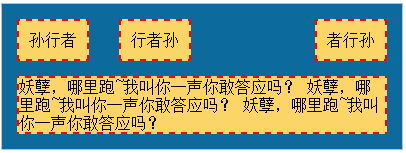
## **二、清除浮动**

在CSS中，清除浮动都是在设置左浮动或者右浮动之后的元素内设置。

clear:取值;

| **clear属性值** | **说明** |
| --- | --- |
| left | 清除左浮动 |
| right | 清除右浮动 |
| both | 左右浮动一起清除 |

使用clear属性清除浮动，我们比较少使用“clear:left;”或者“clear:right;”来判断是清除左浮动，还是清除右浮动。我们往往直截了当地使用“clear:both;”来把所有浮动清除，还省事。也就是说，我们在这一章里只要学会“clear:both;”这一个属性就足够啦。



由于p元素清除了浮动，所以p元素的前一个元素产生的浮动就不会对后续元素产生影响，因此p元素的文本不会环绕在浮动元素的周围。