



2 天驾驭 DIV+CSS

Version 2.0

作者：KwooJan

厘米学院：www.W3Cstudy.com

前端粉丝网：www.W3Cfans.com

CSS 学习网：www.CSSXueXi.com

厘米动力：www.cmmotion.com

联合发布

厘米动力（北京）科技有限公司

总目录

总目录	2
版权声明.....	4
作者简介.....	5
一、基础篇.....	6
1.1 你必须知道的知识	6
1.1.1 [知识一] DIV+CSS 的叫法是不准确的	6
1.1.2 [知识二] DIV+CSS 将你引入误区	7
1.1.3 [知识三] 什么是 W3C ?	8
1.2 你必须掌握的基础	9
1.2.1 [基础一] CSS 如何控制页面.....	9
1.2.2 [基础二] CSS 选择器.....	14
1.2.3 [基础三] CSS 选择器命名及常用命名	19
1.2.4 [基础四] 盒子模型	22
1.2.5 [基础五] 块状元素和内联元素	24
二、实战篇.....	32
2.1 你必须灵活运用技能	32
2.1.1 [第一课] 实战小热身	32
2.1.2 [第二课] 浮动	36
2.1.3 [第三课] 清除浮动	41
2.1.4 [第四课] 导航条(上).....	44

2.1.5 [第四课] 导航条(下).....	52
2.1.6 [大练习] 前四节课大练习	59
2.1.7 [第五课] 浮动(float)布局之结构设计	60
2.1.8 [第五课] 浮动(float)布局之表现设计	69
2.1.9 [第六课] 定位	89
2.1.10 [第七课] 定位应用.....	97
2.1.11 [第八课] CSS Hack.....	101
三、技巧篇.....	104
3.1 凸显专业的小技巧	104
3.1.1 单张图片按钮实例(CSS Sprites、CSS 精灵).....	104
3.1.2 首行文字两文字缩进	106
四、结束语.....	108

版权声明

《2 天驾驭 DIV+CSS》教程由厘米动力（北京）科技有限公司旗下培训中心厘米 IT 学院首席讲师郭建（KwooJan）老师编写，版权归厘米动力（北京）科技有限公司和郭建老师所有，**任何个人或组织不得作为商业用途抄袭、转载、摘编、修改本教程内容**；任何非盈利性个人或者组织可以自由传播（禁止修改、断章取义等）本教程内容，但是必须在明显位置注明来源及作者信息，来源信息必须以链接的方式，链接至 www.W3Cfuns.com，例子如下：

样式一：

转载自：www.W3Cfuns.com 作者：KwooJan

样式二：

转载自：www.W3Cfuns.com 地址：<http://www.W3Cfuns.com> 作者：KwooJan

样式三：

自定义样式，必须以链接的样式，且注明作者信息。

如果在您转载教程后，未在明显位置添加有效链接样式的来源信息，或擅自篡改教程，厘米动力（北京）科技有限公司将根据《中华人民共和国著作权法》向北京市人民法院提交相关材料，追究侵权责任。

作者简介



郭建(KwooJan) 厘米 IT 学院首席课程讲师，7 年 Web 前端研发经验，曾就职于多家互联网名企，郭老师编写的 [《2 天驾驭 DIV+CSS》](#) 教程，指导了众多学习 DIV+CSS 页面布局的学员入门，快速掌握页面布局技巧，使郭老师在 Web 前端开发领域赢得了良好的口碑。

郭老师授课风格风趣幽默、深入浅出，善于引导学员，让学员不仅能知其然，而且知其所以然，擅长使用形象的比喻，将复杂的、难于理解的问题和知识简单化，易于理解和掌握，有效的提高学员自学和独立思考的能力。

郭老师微博：<http://t.qq.com/kwoojan>

大家可以在微博上与郭老师直接交流，另外你还可以通过其他方式与郭老师对话

郭老师博客：<http://www.w3cfuns.com/?1>

同老师对话：<http://www.w3cfuns.com/forum.php?mod=group&fid=115>

技术交流群：<http://www.w3cfuns.com/forum.php?mod=group&fid=82>

提交作业：<http://www.w3cfuns.com/forum.php?mod=group&fid=115>

厘米 IT 学院：<http://www.W3Cstudy.com>

《2 天驾驭 DIV+CSS》网络版：[点击进入>>](#)



一、基础篇

1.1 你必须知道的知识

1.1.1 [知识一] DIV+CSS 的叫法是不准确的

在整个教程的最前面必须先给大家纠正一个错误，就是“DIV+CSS”！

“DIV+CSS”这种叫法其实是一种不准确的叫法，是国人给这种布局标准页面的方法起的名字，是对技术理解不够透彻导致的，而标准的叫法是什么呢？呵呵，没错，标准叫法是 xHTML+CSS 为什么会叫 xHTML+CSS 等你学完后面的“知识二”和“知识三”后就明白了，现在讲一下下面这个问题：

为什么国人将这种布局标准页面的方法叫做 DIV+CSS？

因为第一个将这种技术引进中国的人，对这门技术理解不够透彻，单纯从代码上辨别过去的页面布局方法和现在流行的页面布局方法，认为过去布局页面用的是 Table，称之为“Table+CSS”，而现在布局页面呢，用 DIV，所以叫 DIV+CSS。

听起来也挺合理，岂不知这种叫法误导了绝大部分的网页开发者，认为这样布局出来的页面也就是标准页面，就是符合 W3C 标准的页面，更甚者走了极端，看到网站上用到 Table，就会嘲笑页面做的不够标准，结果用不用 Table 成为了判定页面是否标准的关键点。还有另外一种极端，将页面中用到的标签全部换做 DIV，那就更是大错特错了。

1.1.2 [知识二] DIV+CSS 将你引入误区

“DIV+CSS”叫法将网页制作者引入两大误区

【误区一】 网页中用了 Table，页面就不标准，甚至觉着用 Table 丢人，Table 成为了判定页面是否标准的关键点。

【误区一】 认为网页中的 DIV 标签用的越多越好，甚至有人将页面中所有的标签都替换为 DIV，DIV 的多少，决定页面标准的程度。

为了避免大家进入误区，必须要了解“Table”和“DIV”这两个网页元素诞生的目的，首先 Table 诞生的目的是为了存储数据，而 DIV 诞生的目的就是为了架设页面结构，两者有不同的工作职能，当我们存储数据的时候用 Table 是最方便快捷的，比如 W3Cfun.com 的一个主题页面“浏览器大全”，地址是：

<http://www.w3cfuns.com/portal.php?mod=topic&topicid=6>，这个时候肯定用

Table 最合适了，而表格外面组成页面结构的部分当然用 DIV 了，这是由他们两个诞生的目的决定的，也是符合 W3C 标准的，那么这个页面就是标准页面。

既然是标准页面，标签各干各的活“各司其职”，Table 就用来存储数据，怎么用 Table 就丢人了呢？怎么就不标准了呢？DIV 就用来构架页面结构，怎么会用的越多越标准呢？归根结底就是“DIV+CSS”的叫法导致。

1.1.3 [知识三] 什么是 W3C ?

我们平时说的 W3C，其实是 World Wide Web Consortium 的缩写，中文是 W3C 组织或者万维网联盟，W3C 这个组织做什么的呢？很简单，就是出网页标准的。那么由 W3C 组织出的标准就被称为 W3C 标准，那么符合 W3C 标准的页面就是标准页面了，好，问题来了~

什么是 W3C 标准？

【注意】下面对 W3C 标准的解释，需要理解一下，因为在很多 Web 前端开发工程师面试的时候会遇到这方面的问题，很多企业在面试一些 Web 前端技术人员的时候，认为如果连什么是 W3C 都不知道，那做出来的页面肯定就不能够符合 W3C 标准，所以要求大家留意下！

W3C 标准不是一个标准，而是一系列标准的集合，包含三部分的标准：**结构标准、表现标准和动作标准**。

与结构标准对应的代表语言是 xHTML，与表现标准对应的代表语言是 CSS，与动作标准对应的代表语言是 JavaScript。

当我们将一个成品的网页设计制作成一个静态页面的时候，就要符合前面两种标准，结构标准和表现标准，那么制作出来的页面就是标准页面，用他们相对应的语言来描述这种制作标准页面的技术我们就称之为“xHTML+CSS”！

【总结】知识一、知识二、知识三是大家必须知道的，知道了这些无论去面试还是和其他人沟通，都会让对方感觉你这个人很专业，对技术理解很透彻！

“你必须知道的知识”讲完了，后面我们就要讲“你必须掌握的基础”，内容不多，但是一定要细看，理解透彻了，这样对后面的“实战篇”才能够起到很好的促进作用。

1.2 你必须掌握的基础

1.2.1 [基础一] CSS 如何控制页面

本节主要讲解，两个内容：

第一：CSS 如何控制页面样式，有几种方式；

第二：这些方式出现在同一个页面时的优先级。

使用 xHTML+CSS 布局页面，其中有个很重要的特点就是结构与表现相分离，结构指 xHTML 页面代码，表现就是 CSS 代码了，如果把一个网页看成穿着衣服的人的话，人就是 xHTML，是结构，而衣服呢就是 CSS，是表现，现在出现的问题是，如何让 CSS 去控制页面？或者说，如何让衣服穿在人身上；不同的 CSS 就可以使页面出现不同的风格适用不同的网站，而不同的衣服，人穿上后就会体现出不同的职业。

第一：如何让 CSS 去控制 HTML 页面？

有 4 种样式(方式)，行内样式、内嵌样式、链接样式、导入样式

1) 行内样式

行内样式是 4 种样式中最直接最简单的一种，直接对 HTML 标签使用 style=""，例如：

```
<p style="color:#F00; background:#CCC; font-size:12px;"></p>
```

虽然这种方法比较直接，在制作页面的时候需要为很多的标签设置 style 属性，所以会导致 HTML 页面不够纯净，文件体积过大，不利于搜索蜘蛛爬行，从而导致后期维护成本高。

2) 内嵌样式

内嵌样式就是将 CSS 代码写在<head></head>之间，并且用<style></style>进行声明，例如：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>W3CFuns.com:打造中国 Web 前端开发人员最专业的贴心社区！</title>
<style type="text/css">
body,div,a,img,p{margin:0; padding:0;}
a{color:#FFF;}
img{float:left;}
#container{width:500px; height:350px;
background:url(http://www.w3cfuns.com/static/image/cm/demo/2d121/w3cBg.jpg) no-repeat; position:relative; margin:0 auto;}
#container p{width:380px; height:40px; position:absolute; left:60px;
bottom:60px; color:#fff; font-size:12px; line-height:18px; text-align:center;
font-family:"微软雅黑", Verdana, Geneva, sans-serif;}
#reg{display:block; width:114px; height:27px; position:absolute; left:191px;
bottom:28px;}
</style>
</head>
<body>
<div id="container">
<p>全国的 Web 前端开发工程师欢聚于<a href="http://www.w3cfuns.com/"
target="_blank">W3CFuns.com</a><br />我们的口号是“打造中国 Web 前端开发人员最专业的贴心社区！”</p>
```

```
<a href="http://www.w3cfuns.com/member.php?mod=register"
target="_blank" id="reg"></a>

</div>

</body>

</html>
```

演示 demo : [demo1](#)

效果如下



内嵌样式，也许大家已经意识到，即使有公共 CSS 代码，也是每个页面都要定义的，如果一个网站有很多页面，每个文件都会变大，后期维护难度也大，如果文件很少，CSS 代码也不多，这种方式还是很不错的。

3) 链接样式

链接样式是使用频率最高，最实用的方式，只需要在<head></head>之间加上<link.../>就可以了，如下：

```
<link type="text/css" rel="stylesheet" href="style.css" />
```

举个例子：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>

<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />

<title>W3CFuns.com:打造中国 Web 前端开发人员最专业的贴心社区！</title>

<link rel="stylesheet" type="text/css" href="
http://www.w3cfuns.com/static/image/cm/demo/2d121/style.css"
/>

</head>

<body>

    <div id="container">

        <p>全国的 Web 前端开发工程师欢聚于<a
href="http://www.w3cfuns.com/" target="_blank">W3CFuns.com</a><br />
我们的口号是“打造中国 Web 前端开发人员最专业的贴心社区！”</p>

        <a href="member.php?mod=register" target="_blank"
id="reg"></a>

    </div>

</body>

</html>
```

演示 demo : [demo2](#)

效果同 demo1



这种方式将 HTML 文件和 CSS 文件彻底分成两个或者多个文件，实现了页面框架 HTML 代码与表现 CSS 代码的完全分离，使得前期制作和后期维护都十分方便，并且如果保持页面风格统一，只需要把这些公共的 CSS 文件单独保存成一个文件，其他的页面就可以分别调用自身的 CSS 文件，如果需要改变网站风格，只需要修改公共 CSS 文件就 OK 了，相当的方便，这才是我们 xHTML+CSS 制作页面提倡的方式。

4) 导入样式

导入样式和链接样式比较相似，采用 import 方式导入 CSS 样式表，在 HTML 初始化时，会被导入到 HTML 文件中，成为文件的一部分，类似第二种内嵌样式。

具体导入样式和链接样式有什么区别，可以参看这篇文章 [《CSS:@import 与 link 的具体区别》](#)，不过我还是建议大家用链接样式！

第二：四种样式的优先级

如果这上面的四种样式中的两种用于同一个页面后，就会出现优先级的问题，这里我不再举例子来说明了，大家在下面自己证明一下下面的结论：

四种样式的优先级按照“就近原则”：**行内样式 > 内嵌样式 > 链接样式 > 导入样式。**

1.2.2 [基础二] CSS 选择器

上节课我们讲了一下 CSS 通过什么方式去控制页面，如果不记得，我来帮大家回忆一下，总共有四种样式（方式）**行内样式、内嵌样式、链接样式、导入样式**，大家通过这四种样式（方式）就可以实现 CSS 对 HTML 页面样式的控制，如果要想让这些样式对 HTML 页面中的元素实现一对一，一对多或者多对一的控制，这就需要用到 CSS 选择器，HTML 页面中的元素就是通过 CSS 选择器进行控制的。

CSS 选择器最基本的有四种：**标签选择器、ID 选择器、类选择器、通用选择器**。

【标签选择器】

一个完整的 HTML 页面由很多不同的标签组成，而标签选择器，则是决定哪些标签采用相应的 CSS 样式，比如，在 style.css 文件中对 p 标签样式的声明如下：

```
p{  
font-size:12px;  
background:#900;  
color:090;  
}
```

这么做，会使页面中所有 p 标签的背景都是 #900(红色)，文字大小均是 12px，颜色为 #090(绿色)，这在后期维护中，如果想改变整个网站中 p 标签背景的颜色，只需要修改 background 属性就可以了，就这么容易！当然你也可以设置整个页面中所有的 div 的属性、a 链接的属性、span 的属性，这么做方便是方便，但是不够灵活，如果页面中除了某个 p 标签背景不是红色外，其他的都是红色，就不能用这种方法定义了。

【ID 选择器】

ID 选择器在某个 HTML 页面中只能使用一次，就像只有一个身份证（ID）一样，不重复！在页面中使用 ID 选择器更具有针对性，如：

先给某个 HTML 页面中的某个 p 标签起个 ID，代码如下：

```
<p id="one">W3CFuns.com:打造中国 Web 前端开发人员最专业的贴心社区！</p>
```

在 CSS 中定义 ID 为 one 的 p 标签的属性，就需要用到“#”，代码如下：

```
#one{  
    font-size:12px;  
    background:#900;  
    color:090;  
}
```

这样页面中的某个 p 就会是 CSS 中定义的样式。针对“页面中除了某个 p 标签背景不是红色外，其他的都是红色的”情况，我们就可以用 ID 选择器单独定义那个背景不为红色的 p 标签，这样问题就解决了。

【类选择器】

这种选择器更容易理解了，就是使页面中的某些标签(可以是不同的标签)具有相同的样式，就像国庆某个方阵中，肯定都是不同的人，却均穿红色衣服，手中高举花环，样式都是一样的，如果想让这一类人都有共同的样式，该怎么做呢~呵呵，和 ID 选择器的用法类似，只不过把 id 换做 class，如下：

```
<p class="one">此处为 p 标签内的文字</p>
```

如果我还想让 div 标签也有相同的样式，怎么办呢？加上同样的 class 就可以了，如下：

```
<div class="one">此处为 DIV 标签内的文字</div>
```

这样页面中凡是加上 class="one"的标签，样式都是一样的了~

CSS 定义的时候和 ID 选择器差不多，只不过把 #换成.，如下：

```
.one{  
    font-size:12px;  
    background:#900;  
    color:090;  
}
```

补充:一个标签可以有多个类选择器的值，不同的值用空格分开，如：

<div class="one yellow left">一个标签可以有多个类选择器的值</div>

这样我们可以将多个样式用到同一个标签中，当然也可以 ID 和 class 一块用，

<div id="my" class="one yellow left">ID 和 class 可同时应用到同一个标签</div>

【通用选择器】

到这里，前三种基本的选择器说完了，但是还需要给大家介绍一个 CSS 选择器中功能最强大但是用的最少的一种选择器“通用选择器”，就是 “*” 星号。

***{此处为 CSS 代码}**

强大之处是因为他对整个网页中所有 HTML 标签进行样式定义，这种功能类似“标签选择器”，覆盖的对象更加广泛，是整个 HTML 的所有标签，功能是强大，但是这样反而限制了它的灵活性。

对于通用选择器还有一个不得不提的用法，就是为了保证作出的页面能够兼容多种浏览器，所以要对 HTML 内的所有的标签进行重置，会将下面的代码加到 CSS 文件的最顶端：

***{margin:0; padding:0;}**

为什么要这么用呢，因为每种浏览器都自带有 CSS 文件，如果一个页面在浏览器加载页面后，发现没有 CSS 文件，那么浏览器就会自动调用它本身自带的 CSS 文件，但是不

同的浏览器自带的 CSS 文件又都不一样，对不同标签定义的样式不一样，如果我们想让作出的页面能够在不同的浏览器显示出来的效果都是一样的，那么我们就需要对 HTML 标签重置，就是上面的代码了，但是这样也有不好的地方，因为 HTML4.01 中有 89 个标签，所以相当于在页面加载 CSS 的时候 先对这 89 个标签都加上了 {margin:0; padding:0;}，在这里我不建议大家这么做，因为 89 个标签中需要重置的标签是很少数，没有必要将所有的标签都重置，用到哪些标签就定义哪些标签，如下：

```
body,div,p,a,ul,li{margin:0; padding:0;}
```

如果还需要 dl、dt、dd 标签重置，那就在上面加上就可以了，如下：

```
body,div,p,a,ul,li,dl,dt,dd{margin:0; padding:0;}
```

用到哪些就写哪些，这点也可以看做衡量页面重构师制作页面水平的高低，以及是否专业的一个方面。

OK！选择器的内容大家应该都明白了，如果不能完全理解也没有关系，到实战篇的时候使用一下就清楚了，但是这些选择器的格式应该记住，也是为后面实战做准备的，只有按照老师的要求，整个教程学下来才能达到相应的效果。

下面就继续讲解一下“**选择器的集体声明**”和“**选择器的嵌套**”

【选择器的集体声明】

在我们使用选择器的时候，有些标签样式是一样的，或者某些标签都有共同的样式属性，我们可以将这些标签集体声明，不同的标签用“,”分开，比如：

```
h1,h2,h3,h4,h5,h6{color:#900;}
```

还有像上面刚刚讲的标签重置，就是利用的选择器的集体声明：

```
body,div,p,a,ul,li,dl,dt,dd{margin:0; padding:0;}
```

再举个例子，无论什么样的选择器，“标签选择器”，“ID 选择器”，“类选择器”，只要是选择器，只要有公共的 CSS 代码，就可以用“选择器的集体声明”，起到精简代码的作

用，有一段代码如下：

```
#header{font-size:14px; background:#ccc;}  
div{font-size:14px; width:960px;}  
.blue{font-size:14px; color:#009;}  
.h1{font-size:14px; font-weight:normal;}
```

我们就可以将上面的代码进行精简，把公共的 CSS 代码用选择器的集体声明提取出来，有点类似小学的“提取公因式”似的，如下：

```
#header,div,.blue,h1{font-size:14px;}  
#header{background:#ccc;}  
div{width:960px;}  
.blue{color:#009;}  
.h1{font-weight:normal;}
```

这是选择器的集体声明的经典应用，把共同的部分提取出来，这么做的好处，相同的部分共同定义，不同的部分单独定义，保证风格统一，样式修改灵活，这也是优化 CSS 代码的一块，要记住！

【选择器的嵌套】

选择器也是可以嵌套的，如：

```
#div1 p a{color:#900;}/ *意思是在 ID 为 div1 内的 p 标签内的链接 a 标签的文字  
颜色为红色*/
```

这样的好处就是不需要在单独的为 ID 为 div1 的标签内的 p 标签内的 a 标签单独定义 class 选择器或者 ID 选择器，CSS 代码不就少了嘛~同样也是 CSS 代码优化的一块。

好，这节课主要讲解了四种 CSS 代码选择器、选择器的集体声明、选择器的嵌套三块知识，要掌握好，掌握牢固了，为后面章节的实战做准备！

1.2.3 [基础三] CSS 选择器命名及常用命名

规范的命名也是 Web 标准中的重要一项，标准的命名可以使代码更加易读，而且利于搜索引擎搜索，比如定义了两个 div，一个 id 命名为“div1”，另外一个命名为“News”，肯定第二个比较易读，而且搜索引擎抓取率要高，在团队合作中还可以大大提高工作效率。为了达到这种效果我们就要规范化命名（语义化命名）！

说个题外话，规范化命名的代码，会显着你更加专业！

关于 CSS 命名法，和其他的程序命名差不多，主要有三种：**骆驼命名法**，**帕斯卡命名法**，**匈牙利命名法**。看他们的名字挺不好理解的，不要被吓到了，其实很容易，不信的话继续往下看~

【骆驼命名法】

说到骆驼大家肯定会想到它那明显的特征，背部的隆起，一高一低的，我们的命名也要这样一高一低，怎么才能这样，就用大小写字母呗~，大写的英文就相当于骆驼背部的凸起，小写的就是凹下去的地方了，但是这个也是有规则的，就是第一个字母要小写，后面的词的第一个字母就要用大写，如下：

#headerBlock

第一个单词（header）的第一个字母（h）用小写，第二个单词（block）的第一个字母用大写（B），如果第二个单词后面还有单词呢？那就是下面这种情况，

.navMenuRedButton

第一个单词（nav）的第一个字母（n）用小写，第二个单词（menu）的第一个字母用大写（M），第三个单词（red）的第一个字母也用大写（R），第四个单词（button）的第一个字母还是用大写（B），同样后面所有单词的首字母都要大写。

【帕斯卡命名法】

这种命名法同样也是大小写字母混编而成，和骆驼命名法很像，只有一点区别，就是首字母要大写，如下：

#HeaderBlock

和骆驼命名法只有一点区别，就是所有单词的首字母都要大写，当然也包括第一个单词（header）的首字母（h）了，也要大写。

.NavMenuRedButton

如果有多个，也是全部单词的首字母均要大写。

题外话，如果说“骆驼命名法”是单峰驼的话，那么“帕斯卡命名法”就是双峰驼了~

【匈牙利命名法】

匈牙利命名法，是需要在名称前面加上一个或多个小写字母作为前缀，来让名称更加好认，更容易理解，比如：

#head_navigation

.red_navMenuButton

以上三种，前两种（骆驼命名法、帕斯卡命名法）在命名 CSS 选择器的时候比较常用，当然这三种命名法可以混合使用，只需要遵守一个原则“**容易理解，方便协同工作**”就 OK 了，或者说“**即使不懂代码的人看了代码也知道这块起什么作用**”，没有必要强调是哪种命名法，根据个人喜好使用就行。

以下为页面模块的常用命名

头：header	热点：hot
内容：content/	新闻：news
尾：footer	下载：download
导航：nav	子导航：subnav
侧栏：sidebar	菜单：menu
栏目：column	子菜单：submenu
页面外围控制整体布局宽度：wrapper	搜索：search
左右中：left right center	友情链接：friendlink
登录条：loginbar	页脚：footer
标志：logo	版权：copyright
广告：banner	滚动：scroll
页面主体：main	小技巧：tips

到这节课，都是 CSS 比较基础的知识，为了照顾没有一点基础的同学，从下节课开始，在“基础四”“基础五”将介绍 CSS 布局页面中的很重要的两个概念：

1) 盒子模型

2) 内联元素 VS 块状元素

这两块内容要求大家一定要理解透彻，不然会对后面的实战练习有影响，比如做出来的页面错位，就是因为对这两块的内容理解不够导致的。

1.2.4 [基础四] 盒子模型

盒子模型，是 xHTML+CSS 布局页面中的核心！要想学会用 CSS 布局页面，就首先要理解盒子模型！

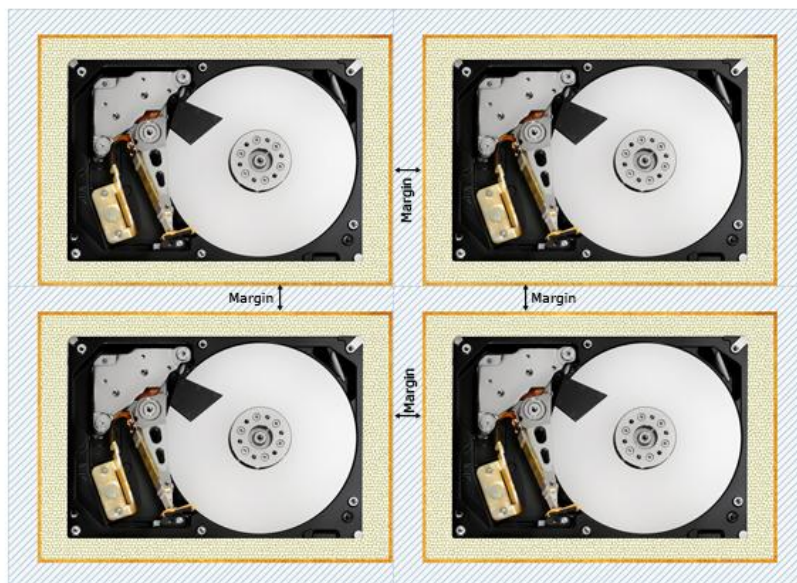
什么是盒子模型？

对于初学者来说，不容易理解，但是对于生活中的盒子大家应该熟悉，大到集装箱，小到铅笔盒，盒子模型你完全可以理解成现实生活中的盒子就可以了，不然怎么能起个名字叫“盒子模型”呢？

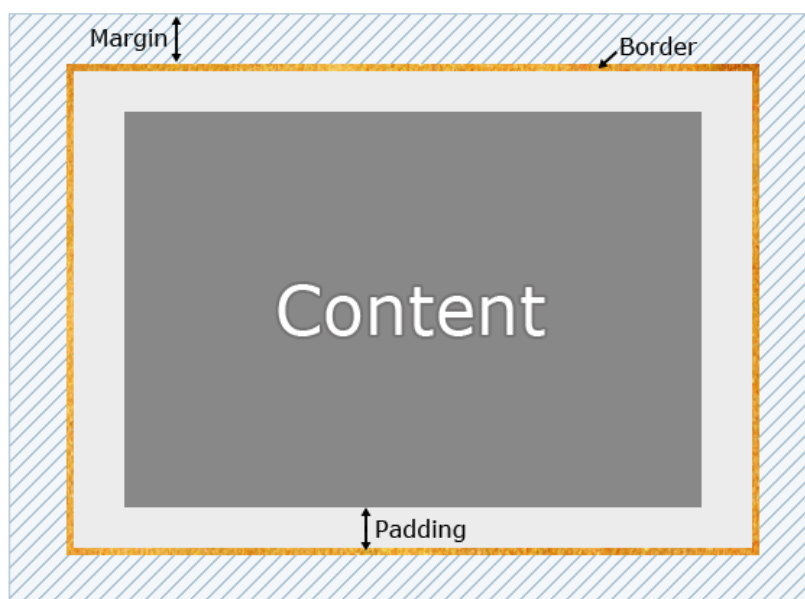
好！既然和现实生活中的盒子一样，那我们想一下，生活中的盒子内部是不是空的好用来存放东西，而里面存放东西的区域我们给他起个名字叫“content(内容)”，而盒子的盒子壁给他起个名字叫“border(边框)”，如果盒子内部的东西比如是一块硬盘，但是硬盘怕震动，所以我们需要在硬盘的四周盒子的内部均匀填充一些防震材料，这时硬盘和盒子的边框就有了一定的距离了，我们称这部分距离叫“padding(内边距)”，如果我们需要购买许多块硬盘，还是因为硬盘怕震动所以需要在盒子和盒子之间也需要一些防震材料来填充，那么盒子和盒子之间的距离我们称之为“margin(外边距)”，padding(内边距)、margin(外边距)，如下图：



Margin 就是多个盒子在一起的时候 盒子与盒子之间的距离 多个硬盘盒子放一块，为避免碰撞，在盒子间也会填充一些填充物，填充物的厚度就是 Margin 外边距，如下图：



OK~！这下盒子模型的四要素就出来了分别是：**content(内容)**、**border(边框)**、**padding(内边距)**、**margin(外边距)**，如下图：



我们的页面就是由许许多多的盒子组成的，但是现实生活中的盒子我们会忽略外边距 (margin)，但是在页面中，我们是不能忽略外边距(margin)的，只有包括外边距的盒子模型在 CSS 中才是完整的，即使外边距为零，我们也不要忽略它，要知道他是存在的。

怎么样，理解了没有？盒子模型就这么简单~，这节课就到这里，下课！

1.2.5 [基础五] 块状元素和内联元素

我们在布局页面的时候，会将 HTML 标签分成两种，**块状元素**和**内联元素**（我们平时用到的标签 div 和 p 就是块状元素，链接标签 a 就是内联元素）。他们是很重要的两个概念，既然说到概念就先看看块状元素和内联元素的定义，在定义中你要留意他们两个的不同之处。

注：这节课看似挺长，其实内容很少，通过例子帮助大家更容易理解而已，不要被眼前的文字和代码吓到。

块状元素

一般是其他元素的容器，可容纳内联元素和其他块状元素，块状元素排斥其他元素与其位于同一行，宽度(width)和高度(height)起作用。常见块状元素为 div 和 p。

内联元素

内联元素只能容纳文本或者其他内联元素，它允许其他内联元素与其位于同一行，但宽度(width)和高度(height)不起作用。常见内联元素为“a”。

做了个对比表，帮助大家更容易理解。

	块状元素	内联元素
width 和 height 起作用	YES	NO
允许其他元素与其同处一行	NO	YES

对于上面的概念，我们用实例的方式给大家讲明白，如下：

【要求】

制作一个 ID 为 div1 的红色(#900)区域，宽度和高度均为 300 像素，并且包含一个 ID 为 div2 的绿色区域，长度宽度均为 100 像素。

CSS 代码如下：

```
#div1{width:300px; height:300px; background:#900;}
```

```
#div2{width:100px; height:100px; background:#090;}
```

HTML 代码如下：

```
<div id="div1">
    <div id="div2"></div>
</div>
```

为了方便初学者更好的学习，我把完整的代码发出来，

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"
/>
<title>W3Cfuns.com---"可容纳内联元素和其他块状元素"</title>
<style type="text/css">
#div1{width:300px; height:300px; background:#900;}
#div2{width:100px; height:100px; background:#090;}
</style>
</head>
<body>
<div id="div1">
<div id="div2"></div>
</div>
```

```
</body>
```

```
</html>
```

怎么样，是不是下面的效果：



如果你做出来了，就继续往下看，咱们给刚才的要求再加一个条件，在 **div1** 里放入一个链接 **a**，内容为“可容纳内联元素和其他块状元素”，颜色为白色。

CSS 代码如下：

```
#div1{width:300px; height:300px; background:#900;}
```

```
#div2{width:100px; height:100px; background:#090;}
```

```
a{color:#fff;}
```

HTML 代码如下：

```
<div id="div1">
```

```
<div id="div2"></div>
```

```
<a href="#">可容纳内联元素和其他块状元素</a>
```

```
</div>
```

是不是下面的效果：



到这里，我们可以看到 div1 这个块状元素里面拥有两个元素，一个是块状元素 div2，另一个是内联元素 a，这就是块状元素概念里面说的“一般是其他元素的容器，可容纳内联元素和其他块状元素”，为什么要说一般呢，因为块状元素不只是用来做容器，有时还有其他用途，这方面的内容会在后面详细讲解，因为不属于本节知识，就不多说。

好~！我们继续加条件，在 div1 里面 div2 的后面再放入一个 ID 为 div3 的长宽均为 100 像素的蓝色(#009)区域块，

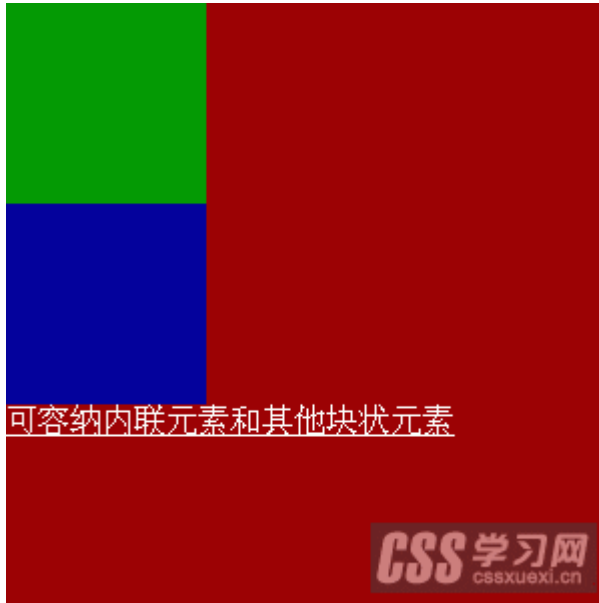
CSS 代码如下：

```
#div1{width:300px; height:300px; background:#900;}
#div2{width:100px; height:100px; background:#090;}
#div3{width:100px; height:100px; background:#009;}
a{color:#fff;}
```

HTML 代码如下：

```
<div id="div1">
  <div id="div2"></div>
  <div id="div3"></div>
  <a href="#">可容纳内联元素和其他块状元素</a>
</div>
```

是不是下面这个效果：



是不是和自己事先想象的不一样，本以为蓝色会处于绿色的右侧，可是却位于下侧，如果你再加几个 div4，div5，同样的他们还是继续位于前一个下面，垂直排列，这就是块状元素概念中说的“块状元素排斥其他元素与其位于同一行”，说白了就是块状元素比较霸道，谁都别想和他坐同一行，甭管你是和他有亲戚关系的块状元素还是毫无联系的内联元素，都不行，都到下面一行待着去，看看例子中，绿色方块和蓝色方块是不是各占一行，内联元素 a 也别想和他处一行，但是事情是没有绝对的，块状元素不是不允许其他元素和他处一行嘛，不是比较霸道嘛，没关系，咱有办法，具体什么办法，我们后面会详细讲解，知识不属于本节内容，就也不多说了，大家留意后面的教程就可以了~

到这里，我想大家对“块状元素”的概念已经比较清楚了，下面通过例子给大家继续解释“内联元素”的概念，当然还是继续加条件，加个什么条件呢，**在 a 的后面再加一个内容为“Love CSS”的链接，所有链接的背景设置为淡橙色(#F93)**

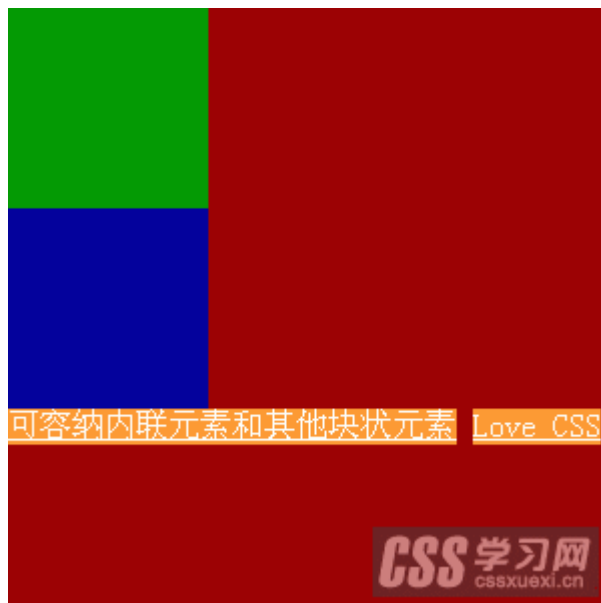
CSS 代码如下：

```
#div1{width:300px; height:300px; background:#900;}
#div2{width:100px; height:100px; background:#090;}
#div3{width:100px; height:100px; background:#009;}
a{color:#fff; background:#F93;}
```

HTML 代码如下：

```
<div id="div1">
<div id="div2"></div>
<div id="div3"></div>
<a href="#">可容纳内联元素和其他块状元素</a>
<a href="#">Love CSS</a>
</div>
```

效果是不是下面这个：



两个链接 a 是不是处于同一行(不要忘记 a 是内联元素)，这就解释了概念上说的“内联元素允许其他内联元素与其位于同一行”，为什么不说是“内联元素允许其他元素与其位于同一行”，因为其他元素包括两种元素，内联元素和块状元素，它如果和内联元素在一块那就肯定在一行了，如果和块状元素在一块，即使它同意，他后面的块状元素也不同意，块状元素会另起一行位于它的下一行。

我们继续添加条件，现在大家给内联元素 a 在 css 中加上宽度和高度，比如 **width:100px;height:50px;**看看有什么变化。

CSS 代码：

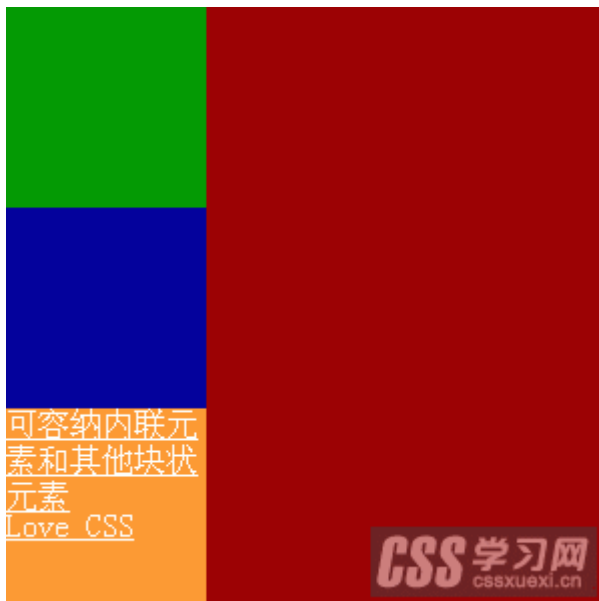
```
#div1{width:300px; height:300px; background:#900;}
#div2{width:100px; height:100px; background:#090;}
#div3{width:100px; height:100px; background:#009;}
a{color:#fff; background:#F93;width:100px;height:50px;}
```

看到效果了没有，是不是没有任何变化呢，这就说明了概念中的内联元素的宽度(width)和高度(height)不起作用，它的大小只随内部文本或者其他内联元素变化，具体证明算是给大家一个作业，自己来证明一下。

如果要让定义好的宽度和高度对内联元素起作用，有什么办法没有？答案是：当然。因为事情没有绝对的，在 CSS 上面也成立，因为 CSS 中有两种元素，内联元素和块状元素，但是宽度和高度只对块状元素起作用，内联元素不起作用，如果我们将内联元素转化成块状元素，他不就具有了块状元素的特性了嘛，当然宽度和高度也就起作用了，如果你能想到这个思路，证明你的大脑现在非常活跃哟，这时候我们只需要给相应的内联元素加上一个属性 display:block 就可以了，如下：

```
a{color:#fff; background:#F93;width:100px;height:50px; display:block;}
```

怎么样，起作用了吧，和下面的效果一样吗~



为什么两个 a 不处于同一行了呢 那是因为这两个内联元素 a 都被转化成了块状元素，既然成功转化为块状元素，就应该具有块状元素最显著的一个特点，不允许其他元素与他同行，所以这两个 a 垂直排列喽~

那有没有办法让他们处于同一行？当然有啦，后面课程会告诉大家^_^

好了，到这里，大家通过实例对内联元素的概念理解的也应该很透彻了，后面就列出所有的内联元素和块状元素，方便以后大家查阅，这里说的是“查阅”，不是让大家记住，常用的块状元素和内联元素，就那么几个，多练习练习自然而然就记住了，平时给大家做培训的时候更强调的是要理解，而不是刻意去记！硬是记下来的，持久性也不长过段时间就忘，在课堂我时常拿例子，通过合理的引导，大家就可以自己推理，自己总结，这样学习效果就非常理想！

常见的块状元素与内联元素去这个页面看看：[点击进入](#)

页面内有个彩蛋，你能找到嘛？^_^

二、实战篇

2.1 你必须灵活运用的技能

2.1.1 [第一课] 实战小热身

上课了~请同学先把下面的例子根据要求做出来。

【例子】

要求

- 1) 宽度、高度均是 200 像素；
- 2) 颜色为红色#900；

自己做做，看看能不能作出来？先不要看我的代码，如果真的做不出来，就下载下来，跟着我下面说的一步一步修改。

下面是我的代码： 【第一课】实战小热身.rar

代码下载页面：<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=247>

在 IE6 和 FF 显示的实际效果



怎么样，比较容易吧~，但是你们有没有发现，红色区域离浏览器的顶部和左边的边距 IE6 和 FF 的不一样，有没有发现？这样的话，我们作出来的页面浏览器就不兼容了，效果不一样了？为什么会这样？

这是因为每个浏览器都有一个内置的 CSS 文件，当你没有对某个标签的属性设置的时候，浏览器就会应用内置的 CSS 文件，怎么才能做到浏览器兼容？不着急，你只需要在 CSS 文件中，将我们目前应用到的标签 body 和 div 置零就 OK 了，代码这么写：

```
body,div{padding:0; margin:0;}
```

如果您对上面 CSS 的样式写法不熟悉，就返回去看一下“[【基础二】CSS 选择器](#)”，当把这句话加上之后，是不是两款浏览器显示效果一样了~如下图：



好，我们接着来，现在再加一个条件，

3) 让红色区域与浏览器的顶部和左边距离为 20 像素；

怎么样，有没有思路？没有思路没关系，继续向下看，

我们，只需要设置红色方块的外边距就可以了，代码如下：

CSS 代码：

```
margin-top:20px;
```

```
margin-left:20px;
```

效果如下图：



这样就使红色区域定位于页面坐标(20,20)处了，与浏览器上边距和左边距都为 20 像素。

不过上面的这种写法通常我们可以精简为：

```
margin:20px 0 0 20px;
```

其中的数值顺序是：上右下左。

而 `margin:20px 0;` 则和 `margin:20px 0 20px 0;` 是等价的哟~只不过是更加精简而已，这样写，[CSS](#) 加载速度会更快。

`margin:20px 0; = margin:20px 0 20px 0;`

我们接着将问题继续延伸，怎样才能让红色区域水平定位于浏览器的正中间，无论浏览器窗口的大小，显示器分辨率的大小。

也很简单，刚刚加的两句话 "`margin-top:20px;margin-left:20px;`" 修改为：

`margin:0 auto;`

怎么样，有意思吧，红色区域是不是位于浏览器的正中间了~

好~！到这里第一节结束，是不是很简单，或者太简单了!!!

如果感觉不过瘾那就赶紧学下一节！^_^

2.1.2 [第二课] 浮动

页面布局有两种方式

- 1) 浮动 Float
- 2) 定位 Position

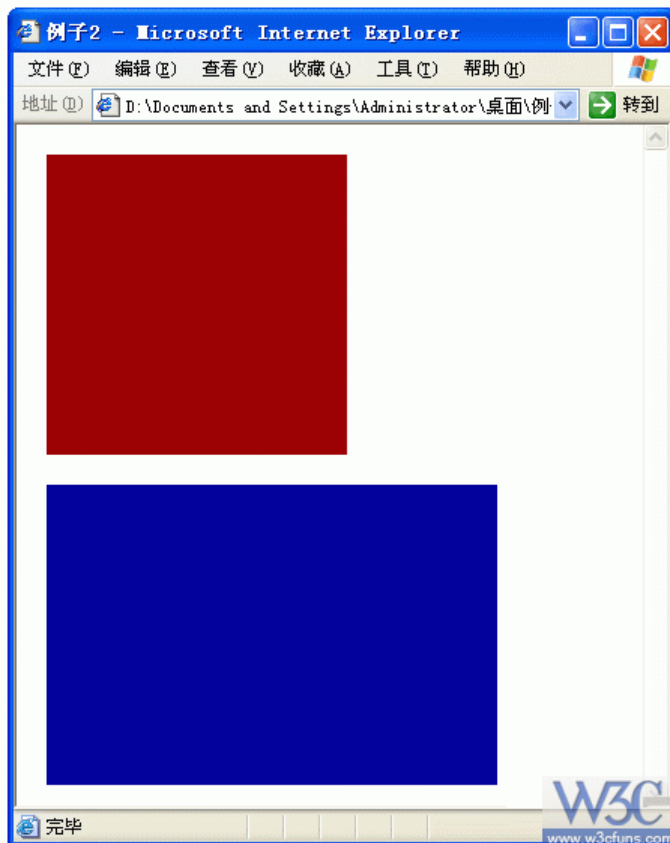
今天就通过一个小小的练习，通过引导的方式，让大家理解 Float 的含义。

【例子】

要求：

- 1) 两个方块，一个红色#900，一个蓝色#009；
- 2) 红色方块宽度和高度均为 200 像素，蓝色方块宽度为 300 像素，高度为 200 像素；
- 3) 红色方块和蓝色方块的上外边距(margin-top)和左外边距(margin-left)均为 20 像素；

页面效果如下：

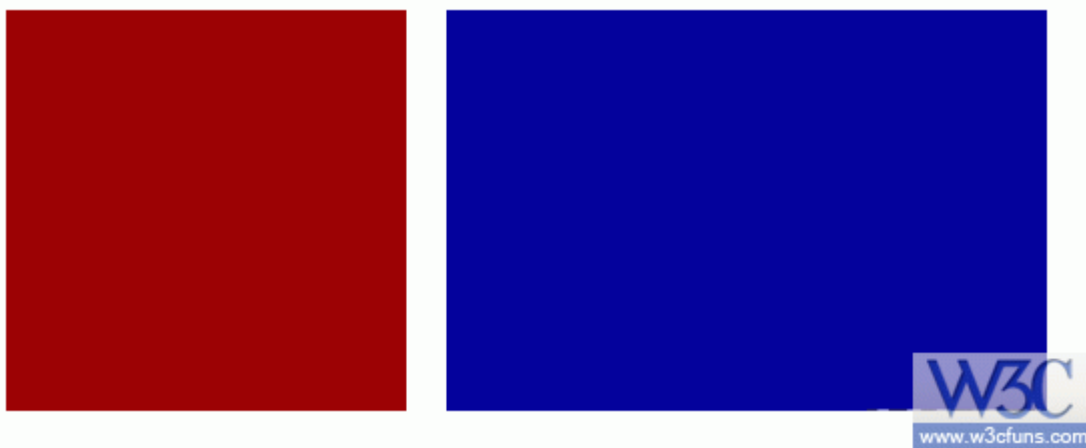


源代码：  【第二课】浮动 实例.rar

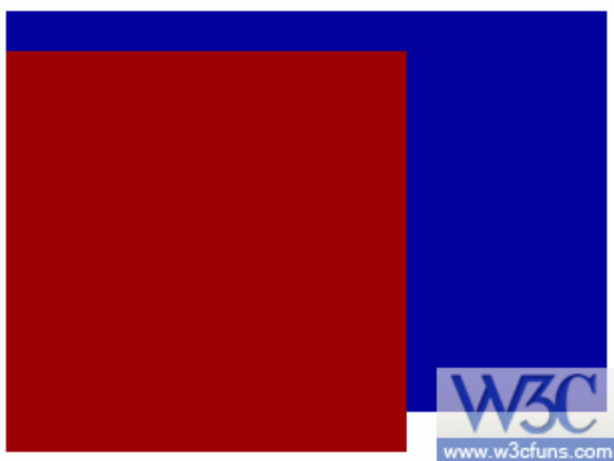
注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=248>

大家应该注意到了，虽然红色方块的宽度并不是 100%，但是蓝色并未和红色处于同一行，这就是块状元素比较“霸道”的一点，（即使块状元素的宽度不是 100%，它也不允许其他元素和他同在一行）为了消除这种“霸权”，让红色和蓝色方块都处在一行，如图：



此时就需要拿出我们的利器 Float 只需要在红色方块的 CSS 里面加上“float:left;”，这时候在 IE6 中可以看到蓝色方块的确跑到红色方块的后面了，并且处于一行了，但是在 FireFox 中却变成了如下效果：



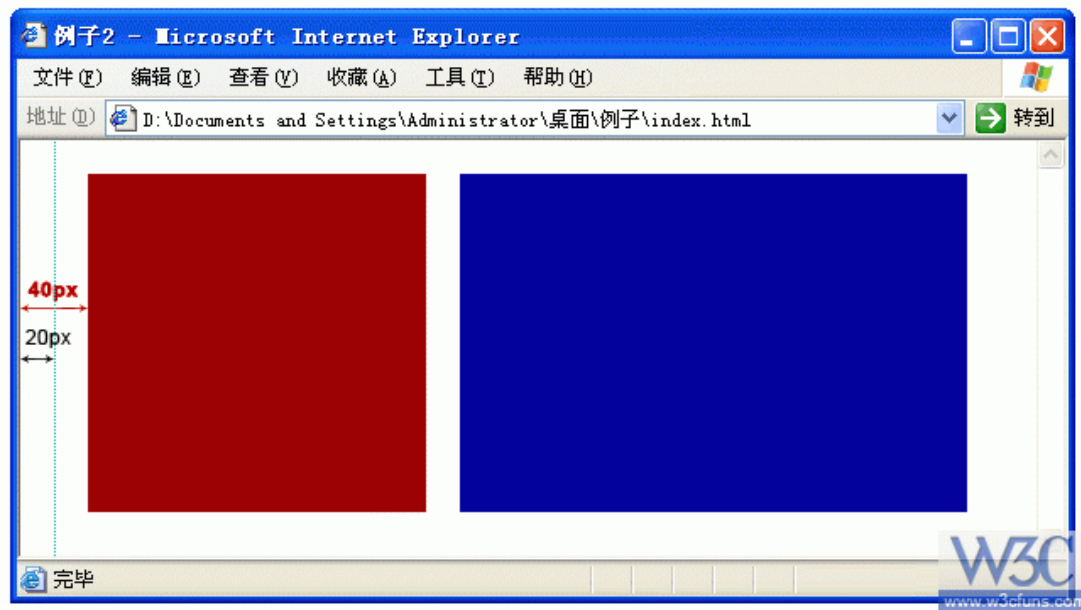
这时候就需要注意了，FF 中如果前面的区域浮动了，后面的那个区域很有可能就会和前面的区域发生重叠并错位。

怎么才能解决这个问题，解决这个浏览器兼容的问题，很容易，只需要在蓝色方块的 CSS 代码中也加入 “float:left;”，问题就解决了，加上试试，看看在 FF 中蓝色方块是不是和红色方块处于一行了~

到这里，大家应该明白 Float 的作用了吧，就是为了消除块状元素“霸权主义”的一把利器！在布局页面时有时候是需要消除块状元素霸权主义才能布局好哟，比如 KwooJan 的博客中间内容部分，分为左边(LEFT)和右边(RIGHT)，就是要用上面这个方法布局的哟，如下图所示：



细心的同学会注意到，在 IE6 中红色方块距离浏览器的左边距并不是 CSS 代码中定义的 20 像素，而是 40 像素，如下图：



其实这是 IE6 的一个 BUG (IE6 双倍边距 BUG)，只要满足下面 3 个条件才会出现这个 BUG:

- 1) 要为块状元素；
- 2) 要左侧浮动；
- 3) 要有左外边距(margin-left)；

解决这个 BUG 很容易，只需要在相应的块状元素的 CSS 属性中加入 “display:inline;”，


代码如下：

```
#redBlock{
    width:200px;
    height:200px;
    background:#900;
    margin-top:20px;
    margin-left:20px;
    float:left;
    display:inline;
}
```

现在再看看，是不是 IE6 和 FF 显示一样了昵~

呵呵，这节课也比较容易吧，如果大家有不明白的可以留言，我会做进一步解释。

下节课，我们讲讲“浮动清除(Clear)”问题！

最终代码：  【第二课】浮动 实例 最终代码.rar

精简后的 CSS 代码加载更快，大家一看就明白了^_^

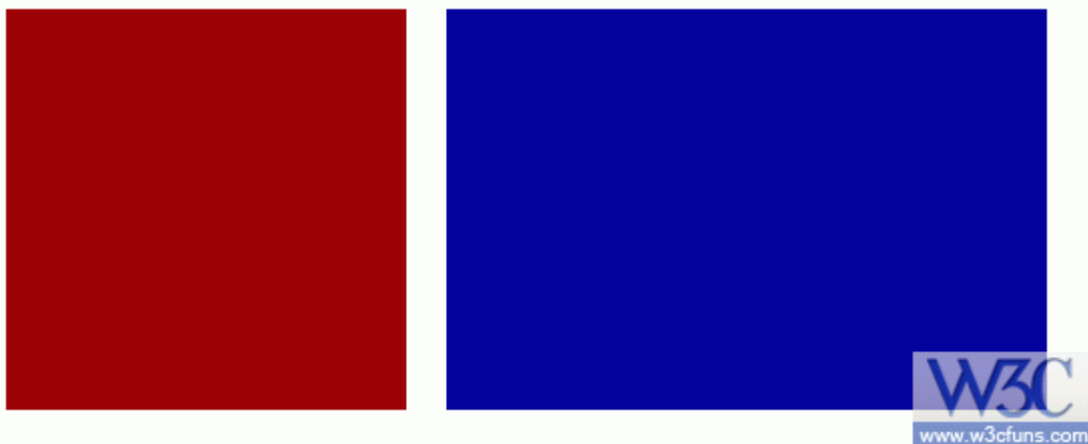
精简后代码：  【第二课】浮动 实例 最终代码 精简版.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=248>

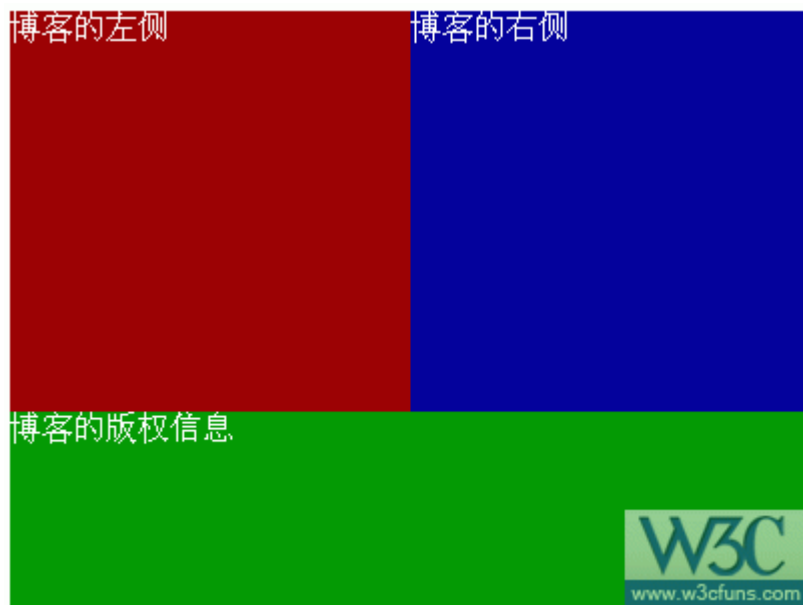
2.1.3 [第三课] 清除浮动

还记得第二课我们做的例子的效果么？最后效果是，红色方块和蓝色方块都处于一行，我们使用“**float:left;**”，打击了块状元素的“霸道”，即块状元素不允许其他元素和它处于同一行。我们将红色方块的 CSS 代码中加入了“float:left;”后，红色方块终于允许蓝色方块和它处于同一行。如图：

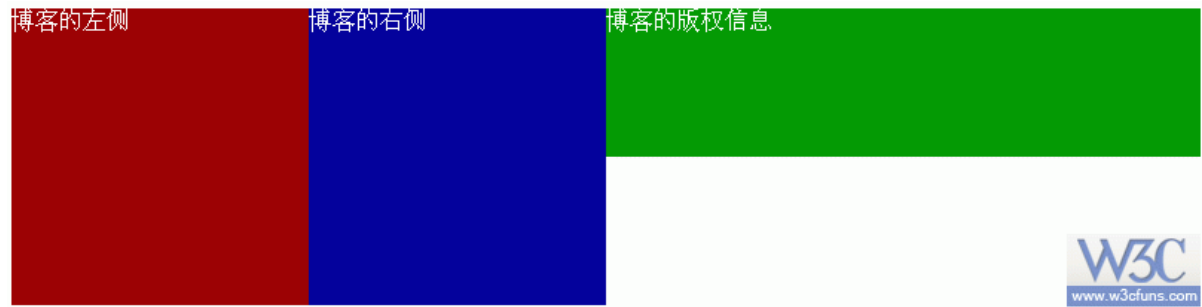


我们换一种方法表达上面的意思，因为红色方块的“左侧浮动”，才导致蓝色方块上移至红色方块的尾后；

在上个例子中，为了达到浏览器兼容性，我们在红色、蓝色方块 CSS 代码中分别加了“float:left;”，这样 IE 和 FF 中显示效果一样，如果此时我们还想放一个宽度 400 像素，高度 100 像素的绿色方块，并让其处于第二行，希望效果如下图：



可是这时候不管怎么放，在 IE 中的效果始终是：



导致绿色排到蓝色的后面这种情况就是因为蓝色方块 CSS 代码中含有 "float:left;"，但是为了浏览器兼容性，又不能去掉(什么？这句话看不明白，只能说明第二节课你没有好好学，好好品味！)，怎么办？

好办~！只要在 CSS 代码中加入下面这段代码：

```
.clear{clear:both;}
```

并在 HTML 代码中加入下面代码：

```
<div class="clear"></div>
```

上面这句话究竟加在哪个位置呢，要加在蓝色方块和绿色方块中间，然后看看效果是不是我们想要的了~^_^

```
<div id="redBlock">博客的左侧</div>
<div id="blueBlock">博客的右侧</div>
<div class="clear"></div>
<div id="greenBlock">博客的版权信息</div>
```

目的就是为了清除蓝色方块的浮动对下面绿色方块的影响！是**影响**哟~是**清除影响**，而**不是清除蓝色方块的浮动**，或者说清除蓝色方块的浮动对下面区域块产生的作用！（仔细品味我说的这句话！）

如果还是不明白，你就在红色方块和蓝色方块中间加上 “<div class="clear"></div>”，看看效果变成什么样子，然后再品品我刚才说的话！

这节课就到这里，下节课我们做一个导航条，很实用的哟！一定要把前三节吃透，不然第四节会跟不上理解不透！

第三课代码： 第三课代码.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=250>

【进阶思考】

如果不用上面说的方法，在 “blueBlock” 和 “greenBlock” 之间加上 “<div class="clear"></div>”，而是在 ID 为 “greenBlock” 的 CSS 代码里直接加上 “clear:both;” 是否管用呢？是否能够清除蓝色方块的浮动对绿色方块产生的影响呢？

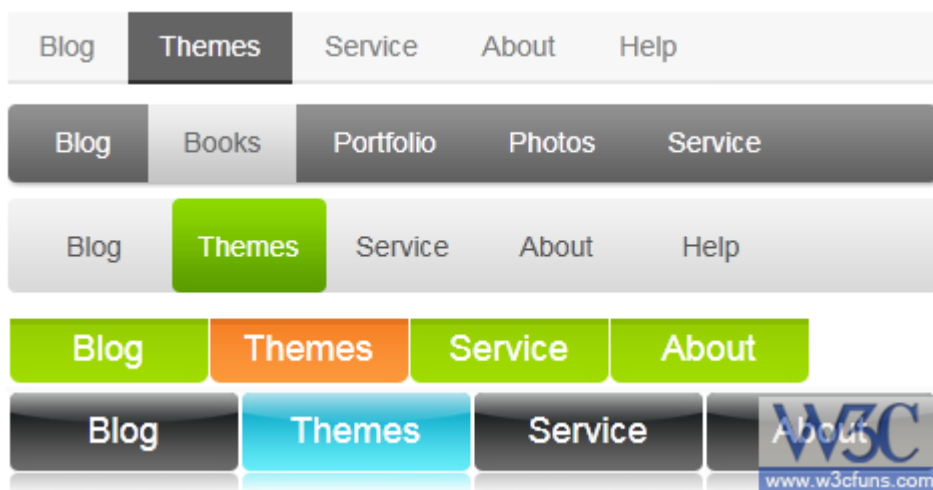
2.1.4 [第四课] 导航条(上)

【注】本节所有的知识点均来自“知识 1、2、3”，“基础篇 1、2、3、4、5”和“实战篇的前三课”，思路是本节的、知识点是原来的，如果有看不懂的地方最好返回去再巩固一下，也许看其中的某一节，你觉着简单，你真的会吗？真的能灵活运用吗？如果前面的教程你采取了一目十行的方式浏览教程，而不是学习教程，这节内容搞不好就卡在哪个前面讲过的知识点上，所以建议大家学习的时候，认真的逐句的去理解教程中的每句话！

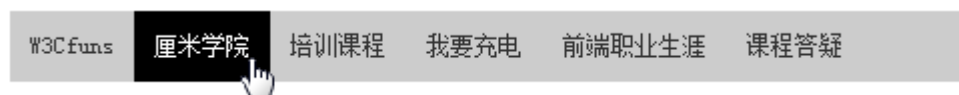
---厘米 IT 学院 KwooJan

好，开始上课！

前三节课，我们知道了什么是“块状元素和内联元素”，以及 xHTML+[CSS](#) 布局的核心概念“盒子模型”，同时又学习了一下页面布局中两种方法中的一种方法“浮动”，这次我们就利用这三个概念，来制作一款经典的导航条，别看它其貌不扬，**可是网上所有的导航条都可以在它的基础上演变而来~，厉害吧~！**其实理论都是一样的，只要你能理解并学会这节课的内容，以后再困难的导航条你都可以轻松应对，比如下面的导航条：



OK！我们要做的导航条的效果如下：**鼠标移动上去背景变黑，并且字体颜色变成白色，**



其实做这款导航条很容易的，我引导，你跟着做，这样思路慢慢就形成了~

【第一步】

我们要先做一个容器（要求：ID 为 “nav”，宽度为 960px，高度为 35px，位于页面水平正中，与浏览器顶部的距离是 30px ），这个容器就是放置我们的导航的盒子~代码如下：

HTML 代码：

```
<div id="nav"></div>
```

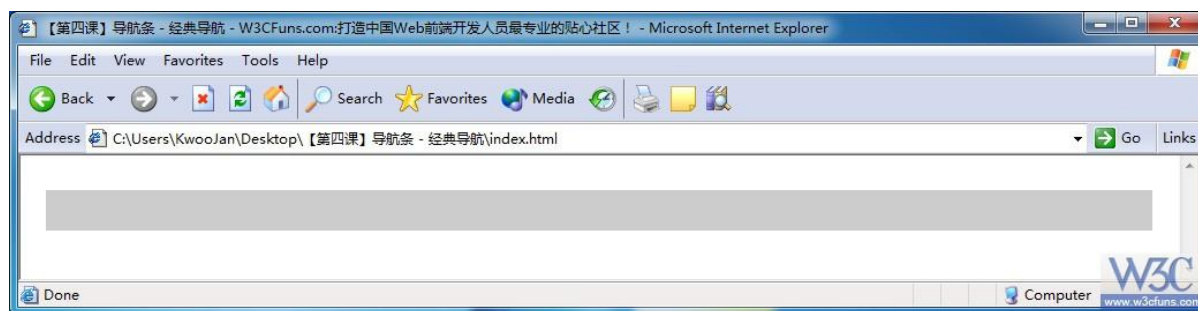
CSS 代码：

```
#nav{
    width:960px;
    height:35px;
    background:#CCC;/*为了便于查看区域范围大小，故而加个背景色*/
    margin:0 auto;/*水平居中*/
    margin-top:30px;/*距离顶部 30px*/
}
```

还有一点需要提醒的是，为了页面在浏览器中的兼容性，不要忘记在 CSS 文件顶部加入标签重置代码~

```
body,div{padding:0; margin:0;}
```

怎么样，作出来了没有，效果是不是一个灰色条（如下图），位于页面的正中间，并且所有浏览器效果一样呢~



如果没有做出来只能说明你没有认真看前面的教程~《2 天驾驭 DIV+CSS》教程本身

就属于经验性和技巧性高度整合的集合体，采取一环套一环的逐步引导的方式来教大家，所以要求大家对每节课都要认真去读，每句话都要认真品味，这样做出来的页面才能很好的兼容多种浏览器，并且代码高度精简，页面加载速度也大大提升！

第一步的源代码：  **【第四课】导航条 - 经典导航 1.rar**

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=252>

【第二步】

盒子做好了，我们就要往里面放导航条了，导航条的内容为“**W3Cfun、厘米学院、培训课程、我要充电、前端职业生涯、课程答疑**”，如果我们把这些内容(目前有 6 个)，当成酒杯的话，如果直接放到盒子里面的话，肯定会乱，并且还会东倒西歪，一点顺序都没有，但是我们平时会用一个隔板将每个酒杯隔开，这样就使酒杯很有序的放入盒子，并且牢稳而且防震，方便使用！现在我们把这个隔板叫做“有序列表”，起个英文名字叫：ul，里面的每个单元格我们也给起个英文名字叫“li”，大家想想里面的这个 ul 是不是和盒子里面的空间一样大，小了，酒杯放不进去，大了杯子就会不稳，所以我们定义 ul 的时候大小一定要和外面的盒子一样大，到这里，应该有点思路了吧~

HTML 代码

```
<div id="nav">
  <ul>
    <li>W3Cfun</li>
    <li>厘米学院</li>
    <li>培训课程</li>
    <li>我要充电</li>
    <li>前端职业生涯</li>
    <li>课程答疑</li>
  </ul>
```

</div>

CSS 代码：

```
#nav ul{
    width:960px;
    height:35px;
}
```

效果作出来了没有，下面是在 IE6 和 FF 中显示效果(其他浏览器大家自己测试，总结规律)：

W3Cfuns
厘米学院
培训课程
我要充电
前端职业生涯
课程答疑

IE6

- W3Cfuns
- 厘米学院
- 培训课程
- 我要充电
- 前端职业生涯
- 课程答疑

Firefox



效果不一样，为什么？在 IE6 中盒子被撑大，FF 中却没有，但是我们的“酒杯”却出来了，还有我们不希望我们的酒杯纵向排列，而是横向排列，怎么办呢？

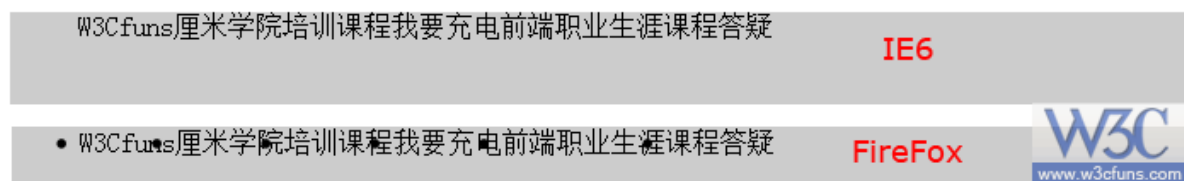
给大家一分钟时间思考~~~

时间到！想不出来也没关系，我们一起分析一下，首先 ul 是什么标签？

是块状元素吧，ul 里面的 li 标签也是块状元素，所以 li 也有块状元素的“霸道”，不允许其他元素和自己处于同一行，总共六个，所以他们六个就像台阶似的纵向排列起来了，如果想让他们横向排列，用浮动 Float 就可以了，可是让谁浮动呢，当然是标签喽~代码如下：

```
#nav ul li{ float:left;}
```

效果是不是和下面的一样呢



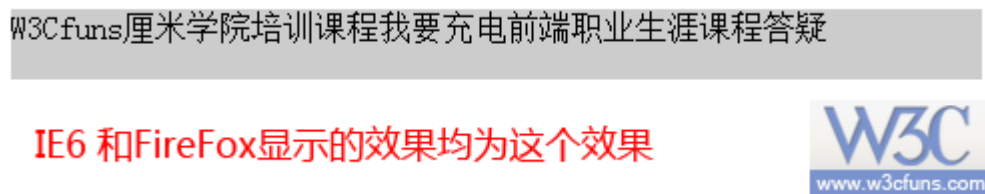
大家会发现虽然“酒杯”横向排列了，但 IE6 和 FF 中的效果还是不一样的

- 1) 盒子(#nav)高度不一样，
- 2) 在 FF 中“酒杯”前面有个大黑圆点，而 IE6 中却没有！

解决上面这两个问题，也很容易，如下：

- 1) 到目前这步，用到了两个新的标签 ul 和 li，标签 ul 和 li 有没有进行重置？只要我们在页面中新写一个标签，就要进行重置，做法是，将 ul、li 标签加入重置代码中，
“body,div,ul,li{padding:0; margin:0;}”
- 2) “酒杯”前面的大黑圆点，是 FF 给 li 标签定义的默认样式，我们只需要将 li 的默认样式去掉就是了，在 ul 标签的 CSS 属性中加入 “list-style:none;” 就 OK 了。

现在再瞅瞅，两种浏览器的显示效果是不是和下图一样了昵~



如果你做到这里的效果和我说的不一样，没关系，我把第二步的源代码发出来，对着上面说的再看看，就很容易学会了。

第二步的源代码：  【第四课】导航条 - 经典导航 2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=252>

【第三步】

第二步的效果还不是我们想要的，所有的“酒杯”都没有保持“车距”，后面的文字全部贴着前面的文字。好~！我们现在就将他们分开！设置标签的宽度为 100 像素：

CSS 代码：

```
#nav ul li{
    width:100px;
    float:left;
}
```

为了便于观察我们暂且将标签的背景设置成红色(设置背景色，是页面布局中一个很重要的方法，便于查看块状元素区域范围)

CSS 代码：

```
#nav ul li{
    width:100px;
    float:left;
    background:#f00;
}
```

效果如下：



瞧瞧，发现问题了吧，我们的标签的高度并没有和我们的盒子的高度一样，这就是为什么在布局页面的时候，经常会设置一下背景色，就是这个道理，不然的话，你发现不了隐藏的问题，但是往往这些隐藏的问题就会导致页面浏览器的兼容性大大降低！

现在暂不把标签的背景色去掉，当我们把它调成我们需要的效果的时候再去掉！

继续，我们把 li 的高度设置成盒子的高度 35 像素，代码自己写，怎么样，高度一样了吧，但是文字却位于顶端，如何将它设置成居中呢，对喽~设置行距（如果你不会，建议你看看这篇文章[《两种方法实现垂直居中》](#)），在的 [CSS](#) 代码中再加入下面这句代码：

```
line-height:35px;
```

效果是不是和下图一样呢

W3Cfuns 厘米学院 培训课程 我要充电 前端职业生涯 课程答疑

好，文字的垂直居中解决了，轮到水平居中了，这个就容易了，直接在的 CSS 代码中再加入下面这句代码：

```
text-align:center;
```

怎么样，效果有点意思了吧~到这里我再发一次代码，以保证大家每步都能理解学会！

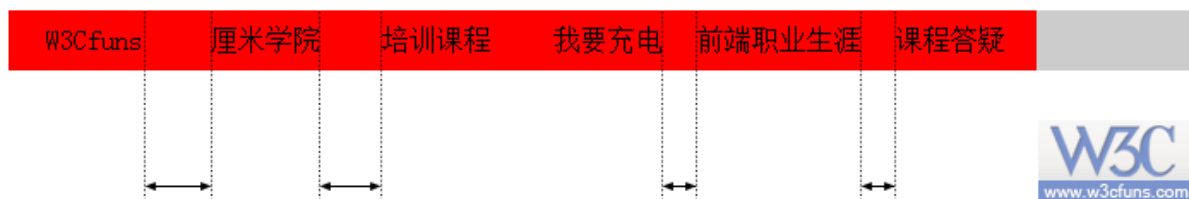
第三步源代码：  【第四课】导航条 - 经典导航 3.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=252>

【第四步】

好~ 做到这里 有没有发现一个问题，“前端职业生涯”与左右两边“我要充电”“课程答疑”的距离与“厘米学院”与左右两边“W3Cfuns”“课程培训”的距离是不一样的，如下图：



这样的效果并不是我们想要的，导航条上的每一个项之间的距离应该是相等的，这才是我们要的，为了寻找原因，我们给 li 设置一个宽度为 1px 的右外边距，

```
margin-right:1px;
```

这样我们就可以看清楚每个 li 的范围，如下图：



这时候应该发现问题的原因了吧，因为我们的标签是设置了宽度为 100 像素，已经限定了它的宽度，如果文字多了它不会自动伸缩自适应的，那这时候我们就需要去掉其宽度，这时候的宽度就会缩小至文字的宽度，如下图：



如果我们这时再添加一些文字(把我们的酒杯换成一个大个的)，这个也会跟着变大，大家去掉宽度后试试，是不是这个样子，这样我们的导航条就比较灵活了，不会对“酒杯”的大小有所顾忌了！

虽然这个宽度自适应解决了，但是给文字的空间太少，视觉上感觉不舒服，那么我们就帮它扩大一下空间，但是又要保证宽度自适应，解决方法很容易，加上左右内边距就 ok 了，这里设置边距为 10px，在标签的 CSS 代码内加上下面代码，

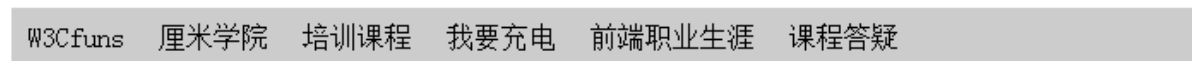
```
padding:0 10px;
```

效果是不是这样



无论你的“杯子”是增大还是缩小，不但宽度会随之增大缩小，但是杯子和杯子之间的距离永远不变！怎么样有点意思吧~！

然后我们再将 li 的背景色和右外边距去掉，得到的效果如下：



第四步的源代码：  【第四课】导航条 - 经典导航 4.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=252>

2.1.5 [第四课] 导航条(下)

上节课我们将导航条做成了下面的效果

W3CfunS 厘米学院 培训课程 我要充电 前端职业生涯 课程答疑

因为导航条的每一项是需要点击的，也就是说导航条的每一项都是链接，这个链接放到哪里呢？当然是要放在盒子里，哪个盒子呢？答案是：标签 li 这个盒子。

【第五步】

- 1) 给导航的每一项加上链接；
- 2) 链接文字大小修改为 12px;
- 3) 并且规定链接样式，鼠标移上去和移出的效果;

制作方法：

- 1) 导航加链接，为了使导航更具有真实性，我们这里就用“厘米学院”的真实网址，大家自己练习的时候，链接地址就随意了~

HTML 代码：

```
<div id="nav">
  <ul>
    <li><a href="http://www.w3cfuns.com">W3CfunS</a></li>
    <li><a href="http://www.w3cstudy.com">厘米学院</a></li>
    <li><a href="http://www.w3cstudy.com/course.aspx">培训课程</a></li>
    <li><a href="http://www.w3cstudy.com/apply.aspx">我要充电</a></li>
    <li><a href="http://www.w3cstudy.com/career.aspx">前端职业生涯</a></li>
    <li><a href="http://www.w3cstudy.com/faq.aspx">课程答疑</a></li>
  </ul>
</div>
```

2) 文字大小 12 像素，这里的文字指的是链接 a，所以我们对 a 进行样式设置，

CSS 代码

```
a{font-size:12px;}
```

3) 鼠标移动上面和移出效果，文字默认状态时黑色无下划线的链接，鼠标移至上方，链接颜色变为白色并带有下划线，鼠标移出后还原为默认状态。

CSS 代码

```
#nav ul li a{color:#333; text-decoration:none;}
```

```
#nav ul li a:hover{color:#fff; text-decoration:underline;}
```

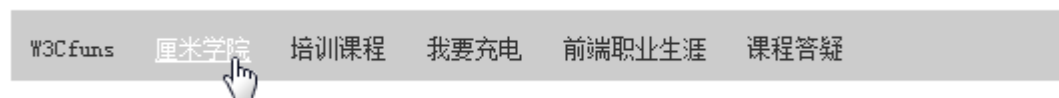
注意：在这个例子中，我们可以将（2）和（3）合并

CSS 代码

```
#nav ul li a{font-size:12px; color:#333; text-decoration:none;}
```

```
#nav ul li a:hover{color:#fff; text-decoration:underline;}
```

效果是不是和下面一样，鼠标移上去变成白色的有下划线的链接



第五步源代码：  【第四课】导航条 - 经典导航 5.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=254>

到这里，基本上一个导航条就出来了~不过为了能让大家技能再提高一个层次，我引导大家继续对导航条进行完善。

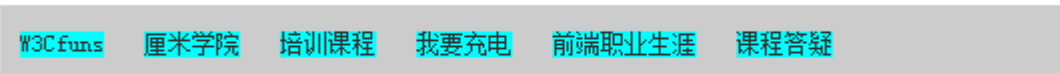
【第六步】

如果我希望鼠标移上去后，链接的背景变成黑色的，下面是我的步骤

首先把链接 a 加上一个背景，以方便看出来链接 a 的区域

```
#nav ul li a{font-size:12px; color:#333; text-decoration:none; background:#0FF;}
```

怎么样，知道 a 的区域了吧



从上面可以看出，鼠标移至链接上“箭头”转换为“手”的有效区域就是亮蓝色区域，如果将 a 的高度设定为 35px 和盒子一样高度，那么这个有效区域的高度就会变为 35px，这样用户体验比较好，现在网页上的导航条基本上都是鼠标移至这个区域“箭头”就会转换为“手”状，于是我给 a 设置一个高度 “height:35px;”，代码如下：

```
#nav ul li a{height:35px; font-size:12px; color:#333; text-decoration:none; background:#0FF;}
```

可是不管我怎么刷新浏览器，高度都没有任何变化，不起作用，这是为什么呢？！

原因就在于 a 属于内联元素，内联元素是无法设置宽度和高度的，width 和 height 只是针对块状元素，说到这里，解决办法就出来了，只要我们把内联元素 a 转化成块状元素就可以了，我们用 “**display:block;**” 将内联元素转化成块状元素。大家加上这段代码，先不要刷新预览页面效果，闭上眼想想界面会变成什么样子？

```
#nav ul li a{display:block; height:35px; font-size:12px; color:#333; text-decoration:none; background:#0FF;}
```

在 Firefox 和 IE6 里面显示的实际效果却是？！？！



两款浏览器显示效果居然大相径庭，这是在页面布局中经常出现的情况，实际效果与期望的效果不同，这就需要大家在平时多去总结。

回归问题，IE6 中为什么所有链接纵向排列了呢？其实这个也很简单，IE 认为 a 既然转化成块状元素，就拥有块状元素的特性---霸道，它是不允许其他元素和它同一行，再加上也没有对 a 的宽度进行设定，所以才导致 IE6 中这么显示，不过 FF 中为什么不这样呢，和我们想象的一样，那是因为 FireFox 认为 a 即使为块状元素，也应该受到外面 元素的影响，所以如此显示。

究竟以谁为标准？

因为大家都认为 FF 是标准浏览器，所以大家可以以 FF 为标准，不过我认为，不用管谁标准不标准，那都是相对的，大家不要在这个问题上浪费精力，应该将精力用在思考如何提高页面的浏览器兼容性，只要做出来的效果是我们想要的就行！

如何解决？

其实解决的方法也很简单，只需要在链接 a 的 CSS 代码中加入 “float:left;”

```
#nav ul li a{display:block; height:35px; font-size:12px; color:#333;  
text-decoration:none; background:#0FF; float:left;}
```

问题迎刃而解，如果到这里你仍旧不明白为何如此解决，说明你没有真正理解前面的课，特别是第二节的课，怎么做？你应该知道....回去再品品去~

第六步源代码：  【第四课】导航条 - 经典导航 6.rar

注意：本节中所有源代码下载页面：

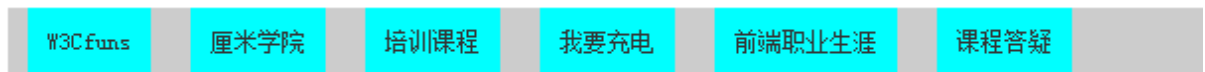
<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=254>

【第七步】

但是这样你不觉得，每个链接的左边和右边是不是太挤了，紧贴着 a 区域的左侧和右侧，应该怎么做？还是很简单，只需要再加上一句话 “padding:0 10px;”

```
#nav ul li a{display:block; height:35px; font-size:12px; color:#333;
text-decoration:none; background:#0FF; float:left; padding:0 10px;}
```

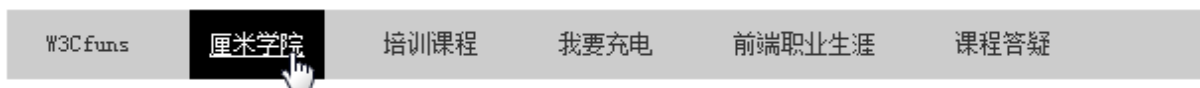
现在再瞅瞅，是不是下面的效果




这样就不那么挤了吧，看着舒服了吧，但是这离我们想要的效果，只剩下最后两步了，因为现在看到的链接效果是，鼠标移上去和拿开背景都是蓝色的，我们现在只需要将 a 链接中的背景去掉，移到 a:hover 的 CSS 代码中，并且颜色变成 “#000” 就 ok 了~

```
#nav ul li a{display:block; height:35px; font-size:12px; color:#333;
text-decoration:none; float:left; padding:0 10px;}
#nav ul li a:hover{color:#fff; text-decoration:underline; background:#000;}
```

怎么样，和下面的效果一样么？



效果好多了吧，这下面是我们想要的效果了吧~

第七步源代码：  【第四课】导航条 - 经典导航 7.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=254>

【第八步】

现在导航条上的每个链接之间不够紧凑，如下图

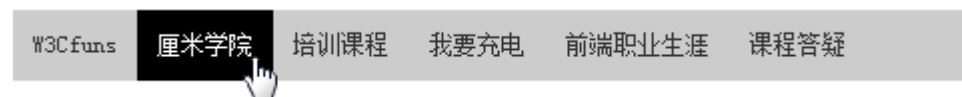


这个情况形成的原因在于给 li 设置了 “padding:0 10px;”, 解决方法就是将 li 的 CSS 样式表里的 “padding:0 10px;” 删除就可以了, 这样每个链接就变得十分紧凑, 效果如下图:



另外, 根据实际情况, 鼠标移动到链接上后, 不需要下划线, 那就去掉 a:hover 里面的 “text-decoration:underline;” 就可以了。

OK! 导航条搞定! 最终效果是:



第八步源代码:  【第四课】导航条 - 经典导航 8.rar

注意: 本节中所有源代码下载页面:

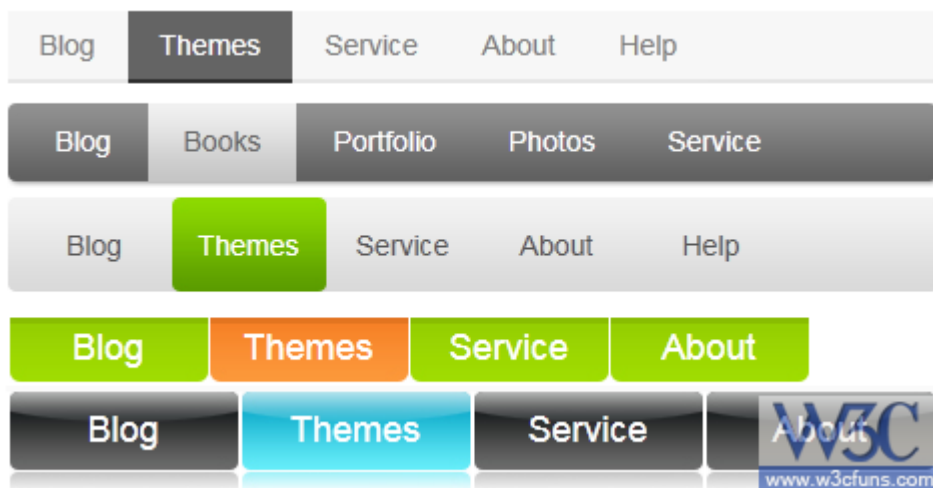
<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=254>

【进阶思考】

在本节的导航条实例中，一开始我们就给导航条外面设置了一个 div 盒子，并设定 id 为“nav”，大家可以思考一下，如果不设置这个 div 盒子，而是将同为块状元素的 ul 标签作为导航的盒子是否可以呢？如果可以的话，我们的结构就少了一层，代码就更精简了，页面加载速度也就会快些。

【结束语】

大家还可以把背景设置成其他颜色，用图片也可以！下面的漂亮的导航条就是咱们最经典的导航条演变而来，有兴趣的可以尝试一下！



现在大家明白，为什么一开始我说这款导航条可以演变出成千上万的不同特色的导航条了吧~万变不离其宗！

第四课的思路就是这样的，如果吃透了这节课，那么以后什么样子的导航都能很轻易作出来，如果你再和 js 很好的结合起来用~你就可以很自信的向老板提出加薪了 !!! ^_^

2.1.6 [大练习] 前四节课大练习

为大家设计了一个大练习，用到的知识点，全部是前四节的内容，如果您能够完全理解前四节内容，你可以在一个小时内完成！

图片已经帮大家切割好了，直接下载就可以，如果想自己切图的话，PSD 源文件也已经发上来了，具体如何制作，怎么制作标准，有哪些技巧，下节课我来引导着大家完成。



已切分的图片下载： 第四课大练习已切分的图片.rar

PSD 源文件下载： 第四课大练习 PSD 源文件.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=267>

2.1.7 [第五课] 浮动(float)布局之结构设计

《前四节大练习》做的怎么样？有没有遇到困难？如果有，没关系，这节教大家掌握“浮动布局页面技能”。

这节主要讲讲页面的结构设计，结构设计的好坏直接影响到页面的加载速度，以及搜索引擎的抓取，所以大家要认真对待。



厘米IT学院
W3CStudy.com

首页 | 培训课程 | 优秀学员 | 课程疑问 | 职业生涯 | 学员社区 | 官方博客 | 学院地址

你不必再走钢丝，厘米IT学院为你拓展发展之路！

前端开发？

2010年 Web前端开发的行业前景您了解么

Web前端开发工程师好找工作吗？

据09年全国Web前端开发行业调查统计显示，09年大型企业对于Web前端开发人才需求紧缺。Web前端开发目前是一种新兴职业，专业的前端开发人员绝大部分存在于大型企业中，如腾讯、百度等，换句话说就是，选择从事Web前端开发就等于你的一只脚已经迈进了腾讯、百度等高薪企业。

随着Web2.0的大潮席卷而来，2010年互联网发展速度空前，互联网向标准化、专业化、精细化方向发展，导致Web开发职位进一步细分，Web前端开发工程师、Web页面重构师等这些高薪职业相继诞生，如今流行的浏览器有十几种，不同的浏览器对页面的解析不同，导致相同的页面在不同的浏览器内显示效果不同，不能兼容多种浏览器，用户体验降低，搜索引擎抓取率低，加载速度慢等影响页面整体质量的因素产生，所以对页面制作要求越来越高，致使许多公司急需提供制作标准页面服务的技术人员。

现在YAHOO、MSN等国际门户网站，网易、新浪等国内门户网站，和主流的Web2.0网站，均采用xHTML(DIV)+CSS的框架模式，更加印证了xHTML(DIV)+CSS是大势所趋。

正因为如此Web前端开发人员成为市场上紧缺的人才，同时也成为一个新兴的高薪职业。

职业生涯

- Web前端开发工程师需要掌握哪些核心技能？
- 我是程序员，有必要进行Web前端开发的学习吗？
- 我是网站美工，目前发展遇到瓶颈，该如何解决？
- 我适合从事Web前端开发行业吗？
- Web前端工程师如何给自己定位？
- Web前端开发工程师好找工作吗？

职业生涯

- 盛大网络 - 前端开发工程师
- 阿里巴巴 - 前端开发工程师
- 金山软件 - XLS-VBS前端页面工程师
- 360京东商城 - Web前端开发工程师
- 阿里巴巴 - 技术部 - Web前端开发工程师 (高)
- 阿里巴巴 - 技术部 - Web前端开发工程师 (初)
- 人人网 - 技术部 - Web前端开发工程师
- 人人网 - 技术部 - Web前端开发工程师
- 搜狐 - 媒体技术产品中心 - JavaScript前端
- 新浪 - 运营部 - Web前端开发工程师
- 新浪 - 无线部 - Web前端开发工程师
- 新浪 - 技术部 - Web前端开发工程师
- 新浪 - 产品部 - JavaScript前端工程师
- 百度 - Web前端开发工程师
- 百度 - 商务搜索 - Web前端开发工程师
- 百度 - 社会化网络事业部 - JavaScript前端
- 百度 - 搜索研发部 - Web前端开发工程师
- 百度 - 系统部 - Web前端开发工程师

关于W3CStudy | 广告服务 | 提交问题 | 联系我们 | 版权声明 | 关于隐私 | 合作伙伴
京ICP备10055601号 All rights(C)2008-2010 Reserved

W3C
www.w3cfuns.com

【第一步】

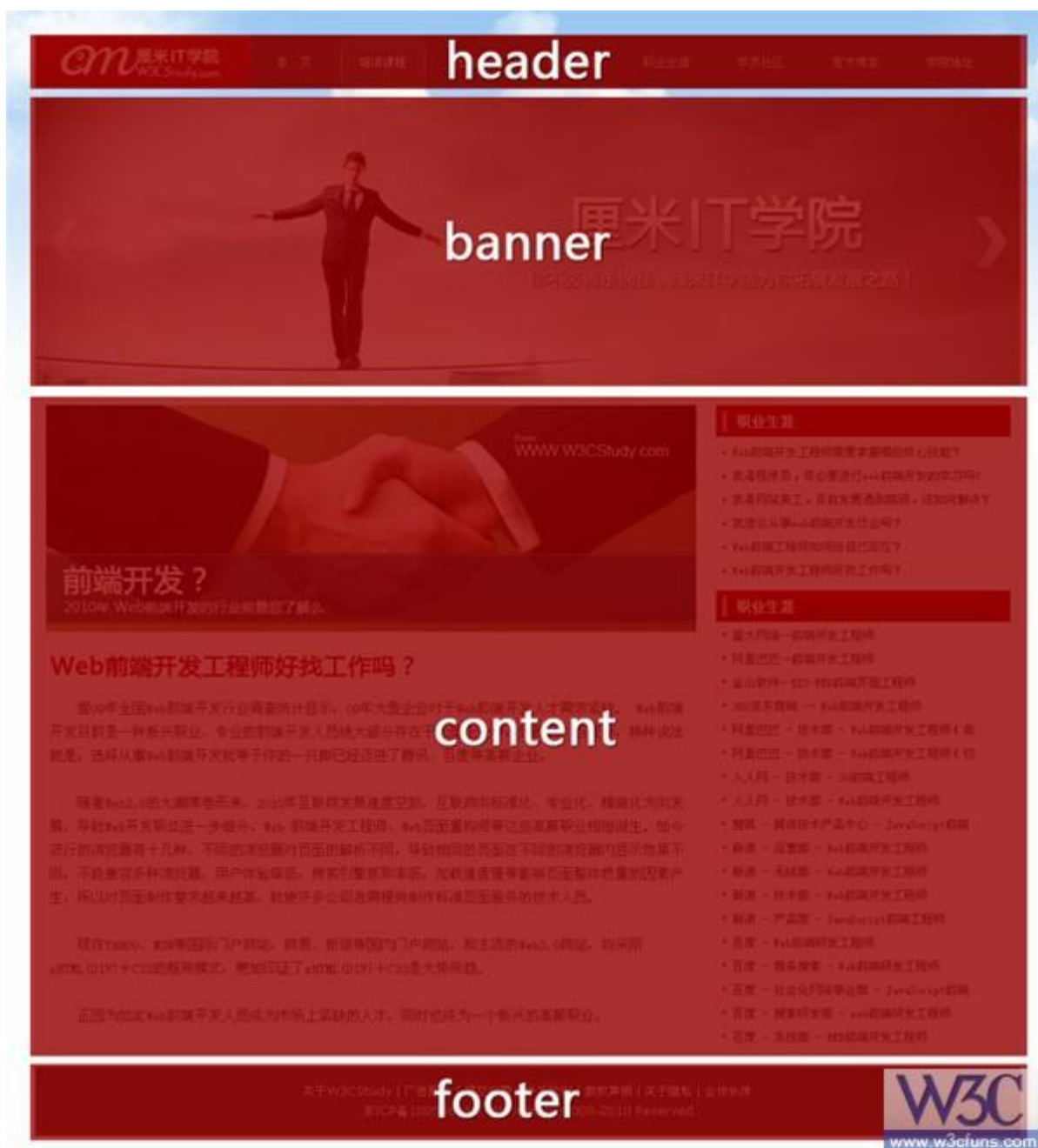
首先我们需要对页面进行分析，由几大块组成，从效果图上不难看出，由四大块组成“头部版块、广告版块、内容版块、页底版块”，分别给他们起个名字如下：

头部版块：header

广告版块：banner

内容版块：content

页底版块：footer



这样就可以很轻松的将页面第一层结构写出来

HTML 代码：

```
<div id="header"></div>

<div id="banner"></div>

<div id="content"></div>

<div id="footer"></div>
```

第一步源代码：



浮动布局页面 1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

【第二步】

头部版块有两部分组成，左侧的 logo 和右侧的导航条

1) 网站 logo 一般是张图片，而且是可以点击的，点击后回到首页，所以 logo 部分的结构为：

```
<div id="logo">

    <a href="http://www.w3cstudy.com">

    </a>

</div>
```

2) 导航我们肯定要用标签 ul 和 li，而 li 里面内容又是链接，在加上真实的链接信息后，导航条部分的结构如下：

```
<ul>

    <li><a href="http://www.w3cstudy.com">首 页</a></li>

    <li><a href="http://www.w3cstudy.com/course.aspx">培训课程</a></li>

    <li><a href="http://www.w3cstudy.com/students.aspx">优秀学员</a></li>
```



```

<li><a href="http://www.w3cstudy.com/faq.aspx">课程疑问</a></li>
<li><a href="http://www.w3cstudy.com/career.aspx">职业生涯</a></li>
<li><a href="http://www.w3cfuns.com/">学员社区</a></li>
<li><a href="http://blog.w3cstudy.com/">官方博客</a></li>
<li><a href="http://www.w3cstudy.com/address.aspx">学院地址</a></li>
</ul>

```

到此，整个头部的结构设计完成，完整的结构如下：

```

<div id="nav">
  <div id="logo">
    <a href="http://www.w3cstudy.com"></a>
  </div>
  <ul>
    <li><a href="http://www.w3cstudy.com">首 页</a></li>
    <li><a href="http://www.w3cstudy.com/course.aspx">培训课程</a></li>
    <li><a href="http://www.w3cstudy.com/students.aspx">优秀学员</a></li>
    <li><a href="http://www.w3cstudy.com/faq.aspx">课程疑问</a></li>
    <li><a href="http://www.w3cstudy.com/career.aspx">职业生涯</a></li>
    <li><a href="http://www.w3cfuns.com/">学员社区</a></li>
    <li><a href="http://blog.w3cstudy.com/">官方博客</a></li>
    <li><a href="http://www.w3cstudy.com/address.aspx">学院地址</a></li>
  </ul>
</div>

```

第二步源代码：  浮动布局页面 2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

【第三步】

Banner 部分在设计中是一款“[拉洋片](#)”效果，在布局页面的时候，我们只需要将其中的一个图片放在此处就可以，因为广告也是可以点击的，所以就需要在图片外面加上链接，那么结构代码如下：

```
<div id="banner">
    <a href="http://www.w3cstudy.com"></a>
</div>
```

第三步源代码：  浮动布局页面 3.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

【第四步】

内容 content 版块呢，是有左右两部分组成，左侧我们虽然可以起名为 left，右侧为 right，但是我们不要这么命名，左侧部分仔细看看其实主要是文章，那么我们就可以考虑给他起个 id 名为“article”，右侧为资讯信息类的内容，所以 id 名为“info”，这样会更加语义化一些，更容易让人清楚这个版块的作用，及内容信息。

当然，我们可以将“left”和“article”组合，用[骆驼命名法](#)来命名左侧的版块 id 为

“leftArticle”，右侧的 id 为“rightInfo”，OK，第二层结构如下：

```
<div id="content">
    <div id="leftArticle"></div>
    <div id="rightInfo"></div>
</div>
```


下面我们开始分析内容 content 板块的第三层，从左侧的 leftArticle 开始，

1) 左侧的文章版块，分“**文章图片，文章标题和文章内容**”三部分

文章图片：从功能上来说文章顶部图片应该是可以点击的，点击后进入文章详情，所以图片外面还是要被链接 a 包含的；

```
<a href="http://blog.w3cstudy.com/?p=5">
    
</a>
```

文章标题：文章标题就是文章内容的总结，对搜索引擎来说文章标题的权重要比内容要高，怎么才能让搜索引擎知道这里是标题呢，当然是用标题标签 h1~h6 标签了，这里我们选用 h1，还有一点不要忘记标题也是可以点击的，所以也要用 a 进行包含，

```
<h1>
<a href="http://blog.w3cstudy.com/?p=5">Web 前端开发工程师好找工作吗？</a>
</h1>
```

文章内容：既然是内容，肯定就少不了段落标签 p 了，有几段文字就用几个“<p></p>”，这样文章部分的结构，就逐渐清晰明朗了，

```
<p> 据 09 年全国 Web 前端开发行业调查统计显示，09 年大型企业对于 Web 前端开发人才需求紧缺。Web 前端开发目前是一种新兴职业，专业的前端开发人员绝大部分存在于大型企业中，如腾讯、百度等，换种说法就是：选择从事 Web 前端开发就等于你的一只脚已经迈进了腾讯、百度等高薪企业。</p>
```

```
<p>... ..</p>
```

```
<p>... ..</p>
```

```
<p>... ..</p>
```

leftArticle 版块 OK！

到此步源代码：  浮动布局页面 4-1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

左侧布局完成，那么右侧又如何布局呢，听我细细分解，首先右侧分两个版块“职业生涯”和“好职推荐”，仔细看会发现他们有共同的结构，共同的样式，只要将其中一个做出来，另外一个结构和样式直接复用就可以了。

那就分析上面的“职业生涯”的结构，首先有个标题，其次下面是标题的一个列表，这样很容易让人想到用“ul+li”的组合，但是我们这里有一个更好方法，用“dl+dt+dd”组合，这种组合在这里是一个很不错的选择，要比“ul+li”要好，很多同学对“dl+dt+dd”组合不熟悉，这里正好练一下，熟悉一下，经过上面的分析我们就可以得出“职业生涯”版块的结构。

```
<dl>
  <dt>职业生涯</dt>
  <dd><a href="http://blog.w3cstudy.com/?p=60">Web 前端开发工程师需要掌握
哪些核心技能？</a></dd>
  <dd><a href="http://blog.w3cstudy.com/?p=51">我是程序员，有必要进行 web 前
端开发的学习吗？</a></dd>
  <dd><a href="http://blog.w3cstudy.com/?p=32">我是网站美工，目前发展遇到瓶
颈，该如何解决？</a></dd>
  <dd><a href="http://blog.w3cstudy.com/?p=22">我适合从事 web 前端开发行业
吗？</a></dd>
  <dd><a href="http://blog.w3cstudy.com/?p=15">Web 前端工程师如何给自己定
位？</a></dd>
  <dd><a href="http://blog.w3cstudy.com/?p=5">Web 前端开发工程师好找工作
吗？</a></dd>
</dl>
```

“职业生涯”下面的“好职推荐”版块就好布局了，换换标题，换换内容就可以了
rightInfo 版块 OK！也就是说整个 Content 版块结构设计完成！

到此步源代码： 浮动布局页面 4-2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

【第五步】

关于页底版块的布局，效果图上是有两行文字组成，第一行文字全部为链接，

```
<p>
    <a href="#">关于 W3CStudy</a> | <a href="#">广告服务</a> | <a href="#">
    提交问题</a> | <a href="#">联系我们</a> | <a href="#">版权声明</a> | <a
    href="#">关于隐私</a> | <a href="#">合作伙伴</a>
</p>
```

第二行左边的是备案号，右侧是版权信息

```
<p>
    <a href="#">京 ICP 备 10055601 号</a>
    All rights(C)2008-2010 Reserved
</p>
```

到这里页底版块结构也出来了

```
<div id="footer">
    <p>
        <a href="#">关于 W3CStudy</a> | <a
        href="#">广告服务</a> | <a href="#">提交问题</a> | <a
        href="#">联系我们</a> | <a href="#">版权声明</a> | <a
        href="#">关于隐私</a> | <a href="#">合作伙伴</a>
    </p>
</div>
```

</p>

<p>

京 ICP 备 10055601 号

All rights(C)2008-2010 Reserved

<p>

</div>

但是这么做并不是最好的,既然文字分两行,那就在第一行的最后加上一个
就可以了,每个标签都有自己的用途,没有必要每一行都用 p 标签包含起来,这样做我们的结构是不是又少一层,代码是不是更精简了呢,所以结构就可以精简为:

<div id="footer">

关于 W3CStudy | 广告服务 | 提交问题 | 联系我们 | 版权声

明 | 关于隐私 | 合作伙伴

京 ICP 备 10055601 号 All rights(C)2008-2010 Reserved

</div>

OK 了~!

至此,我们整个页面结构设计完成!怎么样挺简单吧,这节把页面的骨架搭建完成,下节课我们就讲一下样式设计,给页面穿上衣服,成为一个真正的页面!

本节最终源代码为：  浮动布局页面 5.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=460>

2.1.8 [第五课] 浮动(float)布局之表现设计

[W3C 标准](#)建议大家结构和表现相分离，“结构”就是上节课我们搭建的 HTML 页面框架，“表现”则是这节课的内容，给页面穿上衣服，也就是下面要学的“CSS 样式设计”，样式设计的好坏也会直接影响页面的加载速度以及改版的复杂度。我们紧接着上节课

【第六步】

为了页面能够具有更好的兼容性，所以在设计表现的时候，首先要对标签重置，不明白的可以去看看[\[基础二\] CSS 选择器的通用选择器部分](#)，在上节设计的结构里面，我们用到哪些标签就重置哪些标签，上节课用到“body,div,p,ul,li,dl,dt,dd,h1,a”这么几个标签，所以重置代码为：

```
body,div,p,ul,li,dl,dt,dd,h1,a{margin:0; padding:0;}
```

打开效果图，我们先看一下主背景，是一个从上至下由蓝色渐变为白色，且带有云彩的背景，这个无规律的渐变背景我们应该怎么做？

针对背景的渐变我们好处理，将渐变背景切割为一个宽度为 10px 的小图，起名为 bg，然后水平方向平铺（repeat-x）



而对于云彩这种无规律的背景，直接切割出来作为背景，起名为 clouds，如下图：



这两张背景图，在每步最后提供的源代码的 images 文件夹内可以找到，名字分别为 bg.gif 和 clouds.gif。

这么处理的好处是为了提升网页的用户体验，如果我们不做任何处理直接将页面内的背景整个切割下来，当用户打开页面，会看到背景一片空白，然后瞬间显示背景，给用户的感觉很突兀，而我们的做法是，先加载前面那个 10px 宽度的小图片，这样背景加载速度快，用户打开网页的时候，先把这种背景图加载上，不至于展现在浏览者前面的是一片空白，然后等云彩背景下载完毕后，再加载在页面内，用户感觉会很舒服，这就是为什么 Web 前端开发人员还要懂一些用户体验设计方面的知识。

具体如何实现背景的显示顺序，方法有很多，针对本例，最适用的办法就是设置为不同的盒子的背景，bg.gif 就当做 html 盒子的背景，而 clouds.gif 就当做 body 盒子的背景，因为网页会先加载 html 盒子，然后再加载 body 盒子，那么 CSS 代码如下：

```
html{background:url(../images/bg.gif) repeat-x;}
```

```
body{background:url(../images/clouds.gif) repeat-x;}
```

效果是像下面这样，背景加载上了



此时内容有点乱这是肯定的，因为我们还没有对他们定义样式~

第六步源代码： 浮动布局页面 6.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

【第七步】

页面的宽度为 1000px，所以四大板块 header、banner、content、footer 的宽度也为 1000px，且水平居中于浏览器，所以用 CSS 集体声明的方法，对四大板块的共同样式进行定义：

```
#header,#banner,#content,#footer{width:1000px; margin:0 auto;}
```

其中“margin:0 auto;”的作用就是将页面元素水平居中，这个很重要，布局页面经常用到需要记住！

现在再看一下效果，四大板块是不是全部水平居中于浏览器了~

另外效果图中的头部是与上边沿有 45px 的距离，为实现这个效果，我们采用设置 body 上内边距的办法,将 padding-top 加到 body 的样式内：

CSS 代码

```
body{background:url(../images/clouds.gif) repeat-x; padding-top:45px;}
```

离我们的最后完成页面制作又前进一步~

第七步源代码：  浮动布局页面 7.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

【第八步】

注意：第八步细节上的内容多些，有一点做的不到位，就会导致页面兼容性问题，所以大家静下心来，仔细品味每一句话。

大家有没有发现一个问题，网页里所有能够点击的链接图片，全部都有一个宽度为 2px 的紫色边框，不美观并且还会导致后面做出的页面出现兼容性的问题，这个不是我们想要的，所以我们要将这个边框去掉！CSS 代码如下所写：

```
img{border:none;}
```

上面这段代码是对 img 标签边框的重置，所以我们要把它和之前标签内外边距重置的代码放一块，如下：

```
body,div,p,ul,li,dl,dt,dd,h1,a,img{margin:0; padding:0;}

img{border:none;}
```


这样页面上的图片清爽多了，后面我们还有对其他标签的重置，用到的时候，我们再添加到第二条的后面。

上节我们分析过了，头部 header 包含两部分，左侧的 logo 和右侧的导航 nav。

先说说左侧 logo 部分，回忆一下上节课 logo 部分的结构：

HTML 代码：

```
<div id="logo">
    <a href="http://www.w3cstudy.com"></a>
</div>
```

分析一下上面的结构，首先最外层由一个块状元素 div 构成的盒子，第二层由内联元素 a 构成，第三层由内联元素 img 构成，如果对这些常用标签分不清哪些是块状元素哪些是内联元素，就去看看 [xHTML 标签](#) 页面。

接着上面，img 标签是一个很特别的标签，因为它本身是内联元素，但却体现出块状元素宽高起作用的特性，这是很矛盾的地方，这就为页面布局埋下隐患，要么为内联元素，要么为块状元素，在这里我们更需要它的块状元素的属性，所以我们将身为内联元素的 img 标签转化为块状元素，用 “**display:block;**”。

既然第三层的 img 转化为块状元素，根据 W3C 规范，内联元素是不能包含块状元素的，所以我们还必须把第二层的链接 a，也要转化为块状元素，还是用 “**display:block;**”。

我们希望点击 logo 图片的任何一个地方都可以回到首页，也就是说第二层 a 的有效点击区域大小就为图片的大小，而外面的盒子的大小也等于图片的大小。

那么相应的样式代码如下：

CSS 代码：

```
#logo{width:220px; height:54px; float:left; background:#991616;}

#logo a{display:block; width:220px; height:54px;}

#logo a img{display:block;}
```

针对代码中的“#logo a img{”的写法不清楚的，就去看看[\[基础二\] CSS 选择器](#)的选择器的嵌套

代码中的“float:left;”的作用这里不再说了，不明白的同学去看一下[\[第二课\] 浮动](#)

代码中的“background:#991616;”是一个从 logo 图片上取下来的红色，目的和前面的设置页面背景一样，为了保证在网速比较慢的情况下，logo 图片还未加载完成之时，先显示红色背景，增强用户体验！

此步源代码： 浮动布局页面 8-1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

我们就可以进一步思考，既然 a 已经设置了宽度和高度，外面的那个 id 为 logo 的 div，其实就没有存在的必要了，倒不如直接去掉外层的 div，并将 a 的 id 设置为 logo，这样我们的结构就少一层，代码就少一些，这正是我们期望的，所以 logo 部分的 HTML 结构就可以变为：

```
<a id="logo" href="http://www.w3cstudy.com">
    
</a>
```

CSS 代码就可以变为：

```
#logo{display:block; width:220px; height:54px; float:left; background:#991616;}
```

而 “#logo a img{display:block;}” 就要转化为 “#logo img{display:block;}”

因为现在的#logo 已经为 a 的 id 了，不再是原来最外层的 div 的 id。

此步源代码： 浮动布局页面 8-2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

导航 nav 这块的 CSS 代码就不说了 如果 CSS 代码写不出来 就去回顾一下[\[第四课\] 导航条\(上\)](#) | [导航条\(下\)](#)

这里直接将 CSS 代码列出来

```
#nav{width:780px; height:54px; float:left; background:#393838; list-style:none;}
#nav li{float:left;}
#nav li a{display:block; width:86px; height:54px; line-height:54px;
text-align:center; font-size:12px; color:#ccc; text-decoration:none; float:left;}
#nav li a:hover{color:#fff;}
```

有一点需要补充，现在我们实现的效果是鼠标移至导航条栏目的上面，文字由灰色变为白色，鼠标移出，文字又转为灰色，而处于激活状态的栏目的背景是一个灰色且两边带有光影的图片，如下图：



激活状态的栏目是不受鼠标的移入和移出影响的,这个应该怎么做呢?我们需要定义一个类选择器,专门加在处于激活状态的栏目上:

CSS 代码

```
#nav li .navActive{ background:url(../images/navHoverBg.png) no-repeat;
color:#fff;}
```

如果激活状态的栏目是“首页”,结构代码就这么写

HTML 代码

```
<li><a href="http://www.w3cstudy.com" class="navActive">首 页</a></li>
```

到这里 header 的样式定义基本完成

此步源代码： 浮动布局页面 8-3.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

注意：一般情况下,很多人会认为 header 部分的样式定义就完成了,其实没有注意到这里面隐藏着一个潜在危险,就是<div id="header">高度自适应的问题,尽管里面的 logo 和导航条都设置了高度,但是 header 的高度在 IE7、IE8、IE9、FF 内,高度是不能自动适应里面元素的高度,为了更清楚的说明问题,大家可以将 HTML 结构代码中的导航条部分删除,然后再给 header 设置一个背景色为黑色#000,看看 header 的高度是否能够自适应 logo 的高度。

删除导航条代码后的 header 结构

HTML 代码

```
<div id="header">  
    <a id="logo" href="http://www.w3cstudy.com">  
          
    </a>  
</div>
```

CSS 代码

```
#header{background:#000;}
```

在 IE6 和 FF 内的显示效果如下：



发现问题了没有，在 FireFox 内，看不到背景为黑色的 header，因为他不能够自适应内部元素的高度，所以它的高度变为零，而在 ie6 里面却可以正常显示，这是一个很严重的问题。如果高度不能自适应的话，那内部子元素 logo 和 nav 的浮动产生的影响就会“外泄”，对 header 外面的版块产生影响，从而产生版块错位！

解决这个问题其实很简单，不是高度不能自适应嘛，那就给他设置一个高度，高度值为子级元素的高度，那这样 header 这个盒子就可以刚好装下内部的子 logo 和 nav，而不至于浮动的影响“外泄”，但是这种办法，不够灵活，如果有一天 header 内部多了一些元素，那就需要再次计算一下子级元素的高度，比较麻烦，下面介绍第二种方法，在不设置 header 的高度情况下，无论内部元素再多，header 的高度也会自动适应，直接在 header 的样式里写入“overflow:hidden;”就可以了。

```
#header{overflow:hidden;}
```

现在再去看看 FF 下的页面，是不是黑色的 header 显示出来了，既然高度自适应了，黑色的背景也就没用了，去掉它，然后再在 HTML 结构代码内恢复导航代码，这样做才是真正的完成了页面 header 部分的样式定义~

一个 Web 前端开发人员是否专业，就体现在这些小细节上！

第八步源代码：  浮动布局页面 8-4.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

【第九步】

针对 banner 版块的样式定义，相对第八步就简单多了，结构和 logo 版块一样，对结构进行优化，将最外层的 div 去掉，然后将里面的 a 定义为 banner，所以 banner 版块的结构如下：

```
<a id="banner" href="http://www.w3cstudy.com">
    
</a>
```

CSS 代码

```
#banner{display:block; width:1000px; height:292px; margin:10px auto;}
#banner img{display:block;}
```

其中 “margin:10px auto;” 等价于 “margin:10px auto 10px auto;”,四个数值的顺序是“**上右下左**”，根据第七步我们知道如果设置左右为 auto，那么版块就会居中，但是在这里我们还想让 banner 距离上面的 header 和下面的 content 的距离为 10px，那就设置上下为 10px 就可以了，于是得出了 “margin:10px auto;”

到这里，效果和设计上的是一样的，但是作为 Web 前端开发人员，一定要思考全面，如果图片的大小不是 1000px*292px 怎么办？图片大了，就会溢出，图片小了不美观，所以我们这里还要加一句，无论图片的大小如何，只要在页面上显示，宽高永远是 1000px*292px

```
#banner img{display:block; width:1000px; height:292px;}
```

这句话的意思就是#banner 里面的图片大小为 1000px*292px，无论图片原本大小，只要在#banner 内，就是这个宽高。

如果第九步里面有看不明白的，说明第八步没有仔细看，返回去再去学习一下。

第九步源代码：  浮动布局页面 9.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

第九步完成后的效果(图片在下一页)

厘米IT学院
你不必再走钢丝，厘米IT学院为你拓展发展之路！

前端开发？
2010年 Web前端开发的行业前景您了解么

Web前端开发工程师好找工作吗？

据09年全国Web前端开发行业调查统计显示，09年大型企业对于Web前端开发人才需求紧缺。Web前端开发目前是一种新兴职业，专业的前端开发人员绝大部分存在于大型企业中，如腾讯、百度等。换句话说就是，选择从事Web前端开发就等于你的一只脚已经迈进了腾讯、百度等高薪企业。

随着Web2.0的大潮席卷而来，2010年互联网发展速度空前，互联网向标准化、专业化、精细化方向发展，导致Web开发职位进一步细分，Web前端开发工程师、Web页面重构师等这些高薪职业相继诞生。如今流行的浏览器有十几种，不同的浏览器对页面的解析不同，导致相同的页面在不同的浏览器内显示效果不同，不能兼容多种浏览器，用户体验降低，搜索引擎抓取率低，加载速度慢等影响页面整体质量的因素产生，所以对页面制作要求越来越高，致使许多公司急需提供制作标准页面服务的专业技术人员。

现在YAHOO、MSN等国际门户网站，网易、新浪等国内门户网站，和主流的Web2.0网站，均采用xHTML(DIV)+CSS的框架模式，更加印证了xHTML(DIV)+CSS是大势所趋。

正因为如此Web前端开发人员成为市场上紧缺的人才，同时也成为一个新兴的高薪职业。

职业生涯

Web前端开发工程师需要掌握哪些核心技能？
我是程序员，有必要进行Web前端开发的学习吗？
我是网站美工，目前发展遇到瓶颈，该如何解决？
我适合从事Web前端开发行业吗？
Web前端工程师如何给自己定位？
Web前端开发工程师好找工作吗？

求职推荐

- 盛大网络 - 前端开发工程师
- 阿里巴巴 - 前端开发工程师
- 金山软件 - KIS-Web前端开发工程师
- 360京东商城 - Web前端开发工程师
- 阿里巴巴 - 技术部 - Web前端开发工程师 (高)
- 阿里巴巴 - 技术部 - Web前端开发工程师 (初)
- 人人网 - 技术部 - 3G前端工程师
- 人人网 - 技术部 - Web前端开发工程师
- 搜狐 - 媒体技术产品中心 - JavaScript前端
- 新浪 - 运营部 - Web前端开发工程师
- 新浪 - 市场部 - Web前端开发工程师
- 新浪 - 技术部 - Web前端开发工程师
- 新浪 - 产品部 - JavaScript前端工程师
- 百度 - Web前端开发工程师
- 百度 - 商务搜索 - Web前端开发工程师
- 百度 - 社会化网络事业部 - JavaScript前端
- 百度 - 搜索研发部 - Web前端开发工程师
- 百度 - 系统部 - Web前端开发工程师

关于W3CStudy | 广告服务 | 提交问题 | 联系我们 | 版权声明 | 关于隐私 | 合作伙伴
京ICP备10055601号 All rights(C)2008-2010 Reserved

W3C
www.w3cfuns.com

【第十步】

下面将针对 content 版块进行样式设计，从效果图上我们可以看出，content 版块分为两块，左侧的文章（leftArticle）和右侧的资讯（rightInfo）。

首先分析下思路，content 内部的 leftArticle 和 rightInfo 两个都要左侧浮动，内部元素浮动就会导致外面的 content 的高度不能够自适应内部元素的高度，所以我们首先要在 content 的 CSS 代码中加入 “**overflow:hidden;**”，顺便将背景色、文字大小和文字行距也定义了，如下：

```
#content{overflow:hidden; background:#eaeaea; font-size:12px;
line-height:24px;}
```

然后我们再针对 leftArticle 和 rightInfo 单独定义样式，仔细观察效果图能够很清楚的发现，leftArticle 和 rightInfo 上下左右都有 10 像素的外边距，如果我们用左侧浮动来实现 2 栏效果，在存在外边距的情况下就会出现 IE6 的双倍边距 Bug（在[\[第二课\] 浮动](#)讲过），为了避免出现 bug，我们就利用 “**display:inline;**”，所以他们两个的 CSS 就这么定义

```
#leftArticle,#rightInfo{margin:10px; float:left; display:inline;}
```

OK，他们两个的公共样式定义完了，下面就开始对他们单个定义，在这之前，你需要意识到，原本 leftArticle 和 rightInfo 的宽度分别为 680px 和 320px，当存在 10px 的外边距后，如果再给他们用 CSS 定义宽度的话，那么宽度的数值就变为 660px 和 300px，都要减去 20 像素（为什么是 20px，因为存在 10px 左外边距和 10px 右外边距）

```
#leftArticle{width:660px;}
```

```
#rightInfo{width:300px;}
```

此步源代码： 浮动布局页面 10-1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

现在再看看效果，页面雏形是不是已经出来了~，开心吧~



我们继续观察左侧的 leftArticle，是不是有个图片，图片是不是可以点击的，以后遇到图片链接这种情况，第一反应就是先将这两块给设置成块状元素，我们在这里用骆驼命名法，给图片外面的链接起个 id 叫 articlePicA，意思是文章图片外部的链接

```
<a id="articlePicA" href="http://blog.w3cstudy.com/?p=5">  
      
</a>
```

相应的 CSS 如下

```
#articlePicA,#articlePicA img{display:block;}
```

图片链接搞定！下面轮到文章的标题，在 HTML 结构中用的是<h1>标签，现在我们要按照效果图上的设计，标题距离上面的图片和下方的文字的距离为 20px，文字的大小为 24px，首选字体为“微软雅黑”，其次为“黑体”

```
#leftArticle h1{margin:20px 0; font-size:24px; font-family:"微软雅黑", "黑体";}
```

对于标题与上下的距离这里用的是外边距 margin 来实现，当然也可以用 padding，如果这样就会把盒子给撑大，就需要对文字再次定义，让其垂直居中，反而麻烦了，方法不止一种，作为 Web 前端开发工程师，一定要用最简单的代码实现效果，这也是体现是否专业的一方面。

预览页面后，发现字体变了，但是大小却没有变化，原因是由于浏览器存在默认样式表产生的，所以我们就需要对 h1 进行重置，如下：

```
h1{font-size:100%;}
```

加上这句话是不是我们前面设置的 h1 标签的文字大小就变为 24px 了

既然是重置代码，我们就需要将它和其他的重置代码放一块，好，现在我们的重置代码有三行了，分别是针对标签的内外边距的重置，图片边框的重置，以及 h1 文字大小的重置。

```
body,div,p,ul,li,dl,dt,dd,h1,a{margin:0; padding:0;}
```

```
img{border:none;}
```

```
h1{font-size:100%;}
```

将制作出来的标题和效果图中的标题对比，发现效果图内的链接是红色的，鼠标移上去会出现下划线，为实现这个效果就需要对标题<h1>内的链接<a>的样式进行定义

```
#leftArticle h1 a{color:#900; text-decoration:none;}
```

```
#leftArticle h1 a:hover{text-decoration:underline;}
```

标题定义完了，下面说说内容，还是要和效果图进行对比，对比的结果是效果图上的文字颜色并不是纯黑，而是数值为“#333”的黑色，并且每段文字都会缩进两个文字，每段文字距离下段文字之间的距离是 30px，文字大小为 14px，CSS 如下定义：

```
#leftArticle p{color:#333; text-indent:2em; margin-bottom:30px; font-size:14px;}
```

怎么样到这里左侧 leftArticle 的效果是不是和效果图里的一样了呢

此步源代码： 浮动布局页面 10-2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

下面轮到定义 rightInfo 板块的样式了，从效果图里不难看出，“职业生涯”和“好职推荐”之间有 10px 的距离，实现这个效果我们就采取设置 dl 的下外边距的方法

```
#rightInfo dl{margin-bottom:10px;}
```

针对每个板块的标题 dt 的定义，如下：

```
#rightInfo dl dt{
    background:url(../images/title.jpg) no-repeat;
    height:32px;
    line-height:32px;
    color:#FFF;
    font-size:14px;
    font-weight:bold;
    text-indent:20px;
}
```

背景我们就直接采用已经切好的 title.jpg 且不平铺，dt 高度就是 title.jpg 的高度 32px，为了使文字垂直居中，采取设置行高的方法，不会此方法的去看文章《[两种方法实现 CSS 垂直居中](#)》，然后设置文字大小为 14px、粗细为加粗、以及文字缩进 20px。

下面是针对 dd 的定义

```
#rightInfo dl dd{
    height:24px;
    line-height:24px;
    background:url(../images/dot.gif) no-repeat 7px 10px;
    text-indent:16px;
}
```

重点解释一下第三句，第三句的目的是将做好的小红点图片 dot.gif 设置为 dd 的背景且不平铺，距离 dd 左侧的距离为 7px，上面的距离为 10px；

dd 里面的链接颜色还不是我们效果图上的效果，定义如下：

```
#rightInfo dl dd a{color:#333; text-decoration:none;}
#rightInfo dl dd a:hover{color:#900; text-decoration:underline;}
```

第十步源代码：  浮动布局页面 10-3.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

OK！右侧版块样式定义完成！页面效果如下：

[首页](#)
[培训课程](#)
[优秀学员](#)
[课程疑问](#)
[职业生涯](#)
[学员社区](#)
[官方微博](#)
[学院地址](#)

Web前端开发工程师好找工作吗？

据09年全国Web前端开发行业调查统计显示，09年大型企业对于Web前端开发人才需求紧缺。Web前端开发目前是一种新兴职业，专业的前端开发人员绝大部分存在于大型企业中，如腾讯、百度等，换句话说就是：选择从事Web前端开发就等于你的一只脚已经迈进了腾讯、百度等高薪企业。

随着Web2.0的大潮席卷而来，2010年互联网发展速度空前，互联网向标准化、专业化、精细化方向发展，导致Web开发职位进一步细分，Web前端开发工程师、Web页面重构师等这些高薪职业相继诞生。如今流行的浏览器有十几种，不同的浏览器对页面的解析不同，导致相同的页面在不同的浏览器内显示效果不同，不能兼容多种浏览器，用户体验降低，搜索引擎抓取率低，加载速度慢等影响页面整体质量的因素产生，所以对页面制作要求越来越高，致使许多公司急需提供制作标准页面服务的技术人员。

现在YAHOO、MSN等国际门户网站，网易、新浪等国内门户网站，和主流的Web2.0网站，均采用HTML (DIV) + CSS的框架模式，更加印证了HTML (DIV) + CSS是大势所趋。

正因为如此Web前端开发人员成为市场上紧缺的人才，同时也成为一个新兴的高薪职业。

关于W3CStudy | 广告服务 | 提交问题 | 联系我们 | 版权声明 | 关于隐私 | 合作伙伴
京ICP备10055601号 All rights (C) 2008-2010 Reserved

职业生涯

- Web前端开发工程师需要具备哪些核心技能？
- 我是程序员，有必要进行Web前端开发的学习吗？
- 我是网站美工，目前发展遇到瓶颈，该如何解决？
- 我适合从事Web前端开发行业吗？
- Web前端工程师如何给自己定位？
- Web前端开发工程师好找工作吗？

好职业推荐

- 盛大网络 - 前端开发工程师
- 阿里巴巴 - 前端开发工程师
- 金山软件 - XUS-Web前端页面工程师
- 360京东商城 - Web前端开发工程师
- 阿里巴巴 - 技术部 - Web前端开发工程师 (高级)
- 阿里巴巴 - 技术部 - Web前端开发工程师 (初级)
- 人人网 - 技术部 - Web前端开发工程师
- 人人网 - 技术部 - Web前端开发工程师
- 腾讯 - 媒体技术产品中心 - Javascript前端
- 新浪 - 运营部 - Web前端开发工程师
- 新浪 - 无线部 - Web前端开发工程师
- 新浪 - 技术部 - Web前端开发工程师
- 新浪 - 产品部 - Javascript前端工程师
- 百度 - Web前端开发工程师
- 百度 - 商务智能 - Web前端开发工程师
- 百度 - 社会化网络事业部 - Javascript前端
- 百度 - 搜索研发部 - Web前端开发工程师
- 百度 - 系统部 - Web前端开发工程师

【第十一步】

如果前面几步掌握的不错，那么这一步就是最简单的一步，学起来会轻松很多，还是要看效果图，footer 是一个颜色为#393838 灰色块，与上部的 content 距离是 10px，自身高度是 70px，文字颜色为灰白色(#ccc)且水平居中，文字与 footer 顶部的距离为 18px，行距也是 18px，样式定义如下：

```
#footer{
    background:#393838;
    height:52px;
    line-height:18px;
    margin-top:10px;
```



```
padding-top:18px;

text-align:center;

color:#ccc;

font-size:12px;

}
```

因为设置了 padding-top:18px,这个上内边距，所以定义 footer 的高度就要由原来的高度，减去内边距，即 70px-18px=52px，这样 footer 在视觉上的高度才是 70px。

这时的效果与效果图上的很接近了，唯独就是链接的颜色和样式，加上下面的代码瞅瞅

```
#footer a{color:#ccc; text-decoration:none;}

#footer a:hover{text-decoration:underline;}
```

怎么样页面和效果图一样了吧~！

细心的同学也许会发现，页面内的英文的字体全部是“宋体”，不是十分的美观，不如英文常用的字体 verdana，那我们就给整个页面内的文字字体设置首选字体为“verdana”，只需要在 body 的样式里，加入“font-family”就可以了

```
body{background:url(..images/clouds.gif) repeat-x; padding-top:45px;

font-family:Verdana, Geneva, sans-serif;}
```

第十一步源代码：  浮动布局页面 11.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=461>

【结束】

此时此刻，我们将一个成品的网页设计成功的制作成网页！恭喜大家！

如果您学习的目的仅仅是将页面布局出来，那么你毕业了，如果你是 Web 前端开发工程师或者打算从事 Web 前端开发，那就不能将自己局限于“能把页面制作出来就行”，网页不只是给人看的，做出来摆在那里就行，我们要实现网页的价值！

实现网页价值，就必须先学会，如何提升页面对搜索引擎的亲和度，如何将代码高度压缩，如何提升页面加载速度，如何为企业节省互联网成本，如何提升网页的整体价值，将页面优化至极致！这些内容将在[中高级课程](#)中告诉大家。

好！下课！

2.1.9 [第六课] 定位

很多学习布局的同学，都卡在定位这块，难就难在相对定位与绝对定位的配合上，不知道他们两个之间具体有什么区别。

这节课虽然文字性的内容比较多，但是基本上每一句话都很重要，如果想学好，那就耐下心来仔细看，对教程中的每个假设都要自己验证，自己总结规律，如果自己懒的证明，那就记住本节最后的话就行了。

好，上课！

提问：如果页面内某个元素没有设定 position 属性，那么他是否具有 position 属性？

回答：具有 position 属性，并且属性值是 static。原因在于网页里任一元素的默认 position 属性值均是 static（静态）。

上面这个问题主要是给大家补充一个知识点 很多 Web 前端开发工程师都不知道这点，所以在这节开头给大家补补课。

这节课主要讲讲 **absolute**（绝对定位）和 **relative**（相对定位）。

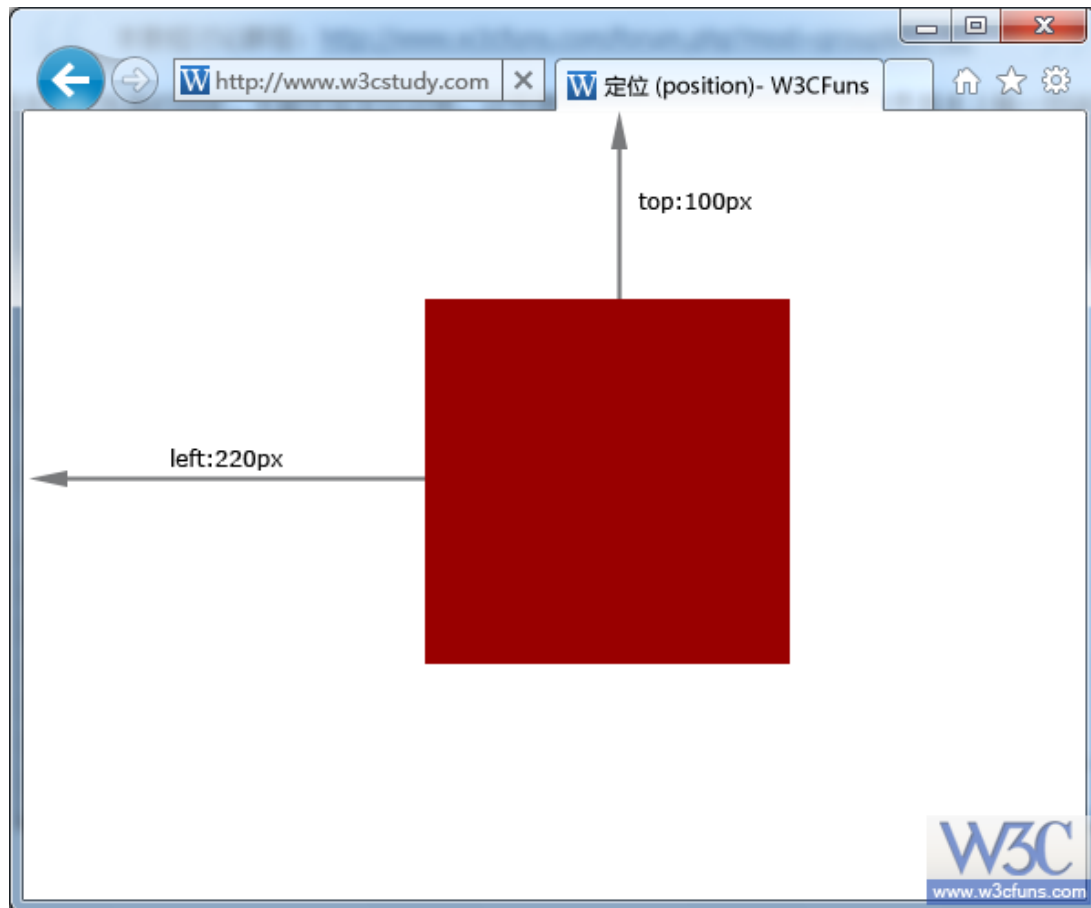
【position:absolute】

意思是：绝对定位，他默认参照浏览器的左上角，配合 TOP、RIGHT、BOTTOM、LEFT(下面简称 TRBL)进行定位。

为了让大家更容易理解这句话，我们举个例子

例子

制作一宽度为 200px 高度为 200px 背景色为红色(#900)的方块 距离浏览器左侧 220px，距离浏览器顶部为 100px，效果：



分析：如果想实现这个效果，我们用外边距也是可以实现的，但是我们这节学的是定位，那么我们就要用定位的知识来实现。

要知道，当一个元素具有了定位属性（特指绝对定位和相对定位）后，想把它精确定位于某一个位置，只需要设置 TRBL 中的任意相邻的两个就可以，针对上面的例子我们用 left 和 top

HTML 代码：

```
<div>定位</div>
```

CSS 代码：

```
body,div{margin:0; padding:0;}
div{width:200px; height:200px; background:#900; position:absolute; left:220px;
top:100px;}
```

源代码： 定位 1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=258>

绝对定位具有以下属性：（下面的这些属性大家自己证明，对于下面说的“父级”，就是在原有的盒子外面，在套一层宽度和高度大于原有盒子尺寸的盒子）

1. 如果没有 TRBL，以父级的左上角，在没有父级的时候，他是参照浏览器左上角，如果在没有父级元素的情况下，存在文本，则以它前面的最后一个文字的右上角为原点进行定位但是不断开文字，覆盖于上方。
2. 如果设定 TRBL，并且父级没有设定 position 属性，那么当前的 absolute 则以浏览器左上角为原始点进行定位，位置将由 TRBL 决定。
3. 如果设定 TRBL，并且父级设定 position 属性(无论是 absolute 还是 relative)，则以父级的左上角为原点进行定位，位置由 TRBL 决定。即使父级有 Padding 属性，对其也不起作用，说简单点就是：它只坚持一点，就以父级左上角为原点进行定位，父级的 padding 对其根本没有影响。

以上三点可以总结出，若想把一个定位属性为 absolute 的元素定位于其父级元素内，必须满足两个条件：

1. 设定 TRBL
2. 父级设定 Position 属性

上面的这个总结非常重要，可以保证各位在用 absolute 布局页面的时候，不会错位，并且随着浏览器的大小或者显示器分辨率的大小，而不发生改变。

只要有一点不满足，元素就会以浏览器左上角为原点，这就是初学者容易犯错的一点，已经定位好的版块，当浏览器的大小改变，父级元素会随之改变，但是设定 Position 属性为 absolute 的版块和父级元素的位置发生改变，错位了，这就是因为此时元素以浏览器的右上角为原点的原因。

【position:relative】

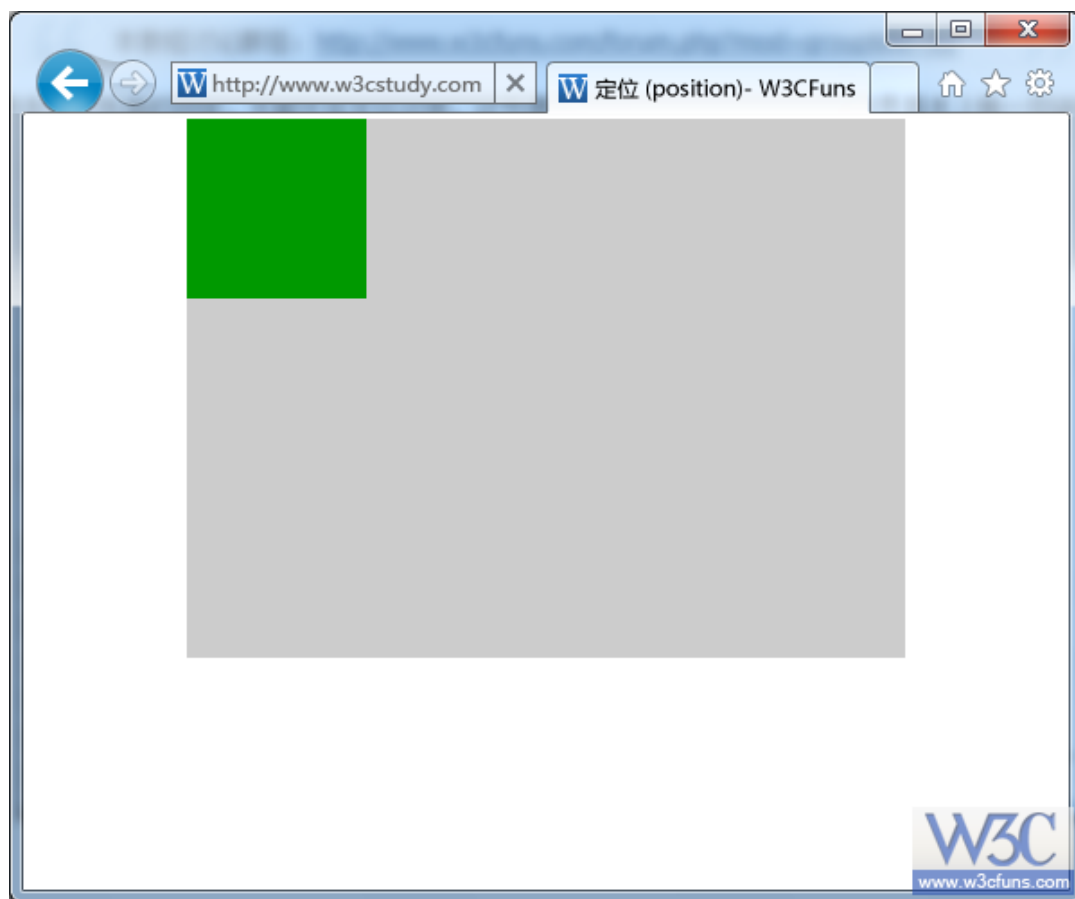
意思是：相对定位，他是默认参照父级的原始点为原始点，配合 TRBL 进行定位，当父级内有 padding 等 CSS 属性时，当前级的原始点则参照父级内容区的原始点进行定位。

为了帮助大家理解上面的那句话，还是拿个例子来说明

例子：

制作一宽度为 400px，高度为 300px，背景色为灰色(#ccc)的方块，水平居中于浏览器，id 为 div1，在 div1 内，制作一宽度为 100px，高度为 100px，背景色为绿色(#090)的方块。

效果：



HTML 代码

```
<div id="div1">
  <div id="div2"></div>
</div>
```

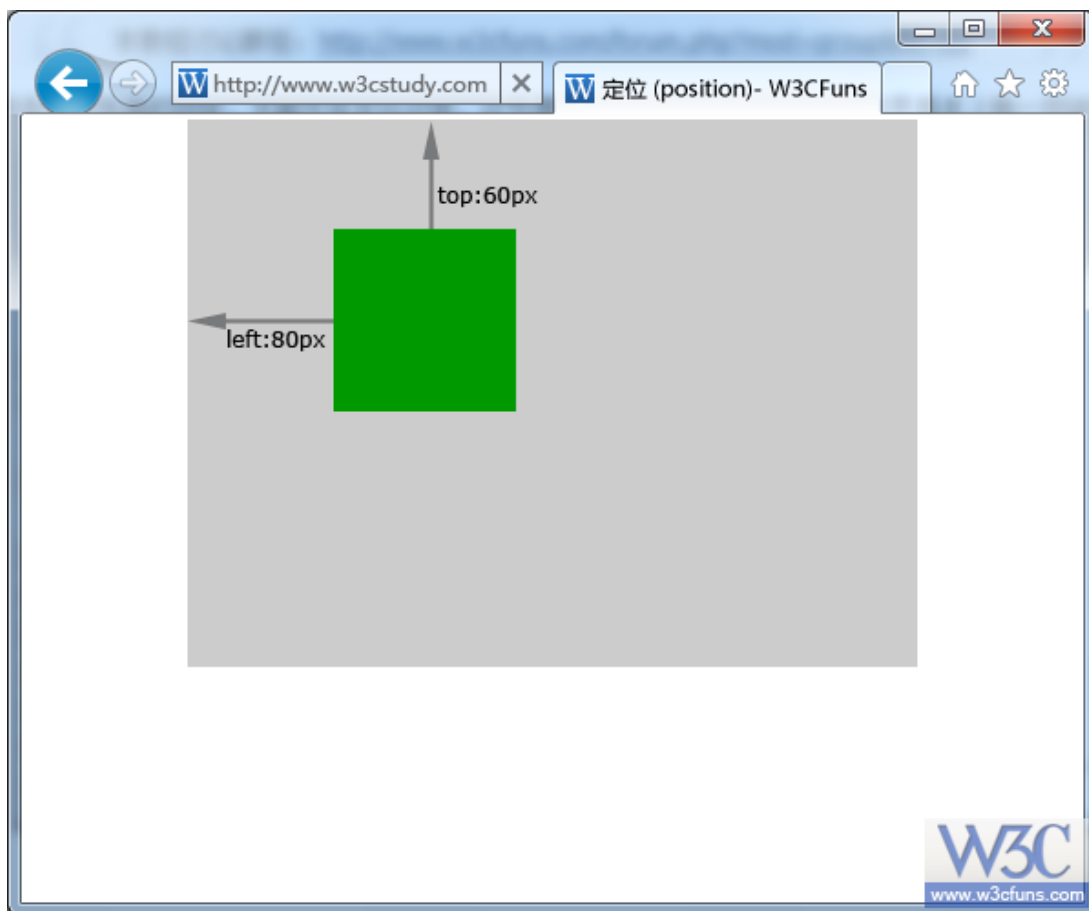
CSS 代码

```
body,div{margin:0; padding:0;}

#div1{width:400px; height:300px; background:#ccc;margin:0 auto;}

#div2{width:100px; height:100px; background:#090; position:relative; }
```

从上面代码是不是可以看出 相对定位的元素 div2 是以父级 div1 的左上角为原始点的，如果需要将 div2 定位于 div1 的某处的话，肯定需要 TRBL 中的任意相邻两个，现在将 div2 定位于 div1 内（80px,60px）处，也就是 div2 距离 div1 的左边为 80px，顶部为 60px，效果如下：



为了达到这个效果，只需要给 div2 一个 left 和 top 的值就可以了

```
#div2{width:100px; height:100px; background:#090; position:relative; left:80px;
top:60px; }
```

此步源代码： 定位 2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=258>

如果现在再给 div1 一个内边距 padding:50px，那么 div2 以哪里为原点呢？

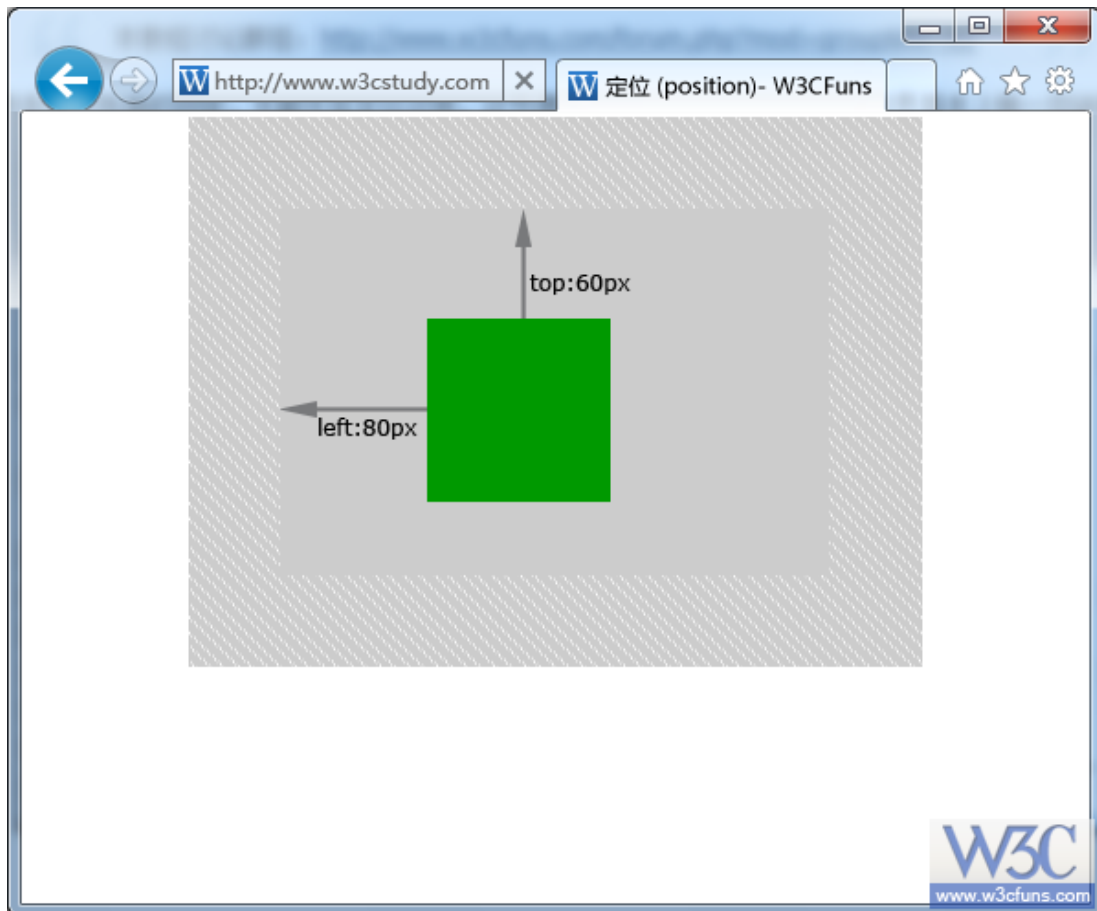
html 代码没变和上面一样，下面只列出 CSS 代码

```
body,div{margin:0; padding:0;}
```

```
#div1{width:300px; height:200px; background:#ccc; margin:0 auto; padding:50px;}
```

```
#div2{width:100px; height:100px; background:#090; position:relative; left:80px; top:60px; }
```

效果如下：(斜线部分为内边距区域，在实际效果中是没有的)



现在知道 div2 以哪里为原点了吧~

相对定位有以下属性：(同样大家在下面自己证明)

1. 如果没有 TRBL，以父级的左上角，在没有父级的时候，他是参照浏览器左上角(到这里和 absolute 第一条一样)，如果在没有父级元素的情况下，存在文本，则以文本的底部为原始点进行定位并将文字断开(和 absolute 不同)。
2. 如果设定 TRBL，并且父级没有设定 position 属性，仍旧以父级的左上角为原点进行定位(和 absolute 不同)
3. 如果设定 TRBL，并且父级设定 position 属性(无论是 absolute 还是 relative)，则以父级的左上角为原点进行定位，位置由 TRBL 决定(前半段和 absolute 一样)。如果父级有 Padding 属性，那么就以内容区域的左上角为原点，进行定位(后半段和 absolute 不同)。

以上三点可以总结出，无论父级存在不存在，无论有没有 TRBL，均是以父级的左上角进行定位，但是父级的 Padding 属性会对其影响。

综合上面对 relative 的叙述，我们就可以将 position 属性为 relative 的 div 视成可以用 TRBL 进行定位的普通 div，或者说只要将我们平时布局页面的 div 的 [CSS](#) 属性中加上 position:relative 后，就不只是用 float 布局页面了，还可以用 TRBL 进行布局页面了，或者说加上 position:relative 的 div 也可以像普通的 div 进行布局页面了，只不过还可以用 TRBL 进行布局页面。

但是，position 属性为 absolute 不可以用来布局页面，因为如果用来布局的话，所有的 DIV 都相对于浏览器的左上角定位了，所以只能用于将某个元素定位于属性为 absolute 的元素的内部某个位置，这样我们就可以总结比较重要的结论

属性为 relative 的元素可以用来布局页面，属性为 absolute 的元素用来定位某元素在父级中的位置 既然属性为 absolute 的元素用来定位某元素在父级中位置 就少不了 TRBL，这时候根据一开始讲的 absolute 的第三条，如果父级元素没有 position 属性那么

absolute 元素就会脱离父级元素，但是如果是布局页面，父级元素 position 的属性又不能为 absolute，不然就会以浏览器左上角为原点了，所以父级元素的 position 属性只能为 relative !

=====

总结：如果用定位来布局页面，父级元素的 position 属性必须为 relative，而定位于父级内部某个位置的元素，最好用 absolute，因为它不受父级元素的 padding 的属性影响，当然你也可以用 relative，计算的时候不要忘记计算 padding 的值。

=====

2.1.10 [第七课] 定位应用

上一节主要讲了下定位，今天讲一下定位如何应用，我们分别拿两个例子来重点讲解一下绝对定位(absolute)和相对定位(relative)。

【绝对定位实例】

打开[厘米 IT 学院](http://www.w3cstudy.com)的首页，右上角有一个“我要充电”



鼠标移动上去会从橙色变为红色，无论浏览器的放大缩小，位置永远处于页面的右上角，大家可以打开页面看一下实际效果，地址是 www.w3cstudy.com

现在我们就来做一个这个效果

分析：右上角的“我要充电”是不会跟随浏览器大小的改变，而去改变它在浏览器右上角的位置，所以这个毫不犹豫的用“绝对定位”，另外它是可以点击的，所以结构中就是链接。

思路：把身为内联元素的链接 a，先转化为块状元素，设置宽高，然后将图片设置为 a 的背景，当鼠标移动上去的时候换背景就可以了。

这里用到这两张图片，图片大小为 107 像素 X107 像素。



HTML 代码：

```
<a href="http://www.w3cstudy.com/apply.aspx"></a>
```

CSS 代码：

```
a{
    display:block;
    width:107px;
    height:107px;
    background:url(images/applyNormal.png) no-repeat;
    position:absolute;
    top:0;
    right:0;
}
a:hover{
    background:url(images/applyHover.png) no-repeat;
}
```

就这么简单！放大缩小浏览器后看看，是不是永远处于右上角~这就是绝对定位。

源代码：  定位应用 1.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=259>

【相对定位实例】

如果能把“我要充电”放到 banner 的右上角怎么办？



根据上节课说的，只需要让外面的盒子的定位属性为相对定位 relative 就可以了，也就是说让 banner 这个盒子的 position 的值为 relative 就可以

HTML 代码

```
<div id="banner">
    <a href="http://www.w3cstudy.com/apply.aspx"> </a>
</div>
```

CSS 代码

```
body,div{margin:0; padding:0;}
#banner{width:1000px; height:292px; background:url(../images/banner.jpg)
no-repeat; position:relative; margin:0 auto;}
a{display:block; width:107px; height:107px;
background:url(../images/applyNormal.png) no-repeat; position:absolute;
top:0; right:0;}
a:hover{background:url(../images/applyHover.png) no-repeat;}
```

这个例子很好的体现了相对定位（relative）和绝对定位（absolute）如何一块使用。

源代码： 定位应用 2.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=259>

【总结】

经过这两个练习，大家对定位应该掌握的差不多了，那就再仔细品味一下上节课总结的话

=====

如果用定位来布局页面，父级元素的 position 属性必须为 relative，而定位于父级内部某个位置的元素，最好用 absolute，因为它不受父级元素的 padding 的属性影响，当然你也可以用 relative，计算的时候不要忘记计算 padding 的值。

=====

2.1.11 [第八课] CSS Hack

说到浏览器兼容性问题，就必须说 CSS Hack！

提到 Hack 大家肯定会想到电脑黑客(hacker)，和病毒程序联系到一块，不过在 CSS 中，Hack 是指一种兼容 CSS 在不同浏览器中正确显示的技巧方法。说的更直白一些就是，你平时做个页面，布局正确，CSS 正确，可就是在不同的浏览器中显示的效果不一样，要么错位，要么多几个像素，怎么都找不到原因，这时候我们就会用一些技巧方法来让不同的浏览器显示一样的效果，这种方法我们就称之为 CSS Hack，记住！CSS Hack 是解决页面不能很好兼容多种浏览器的技巧方法，是一种方法，不要理解偏差。

大家必须知道一点，CSS Hack 都属于个人对 CSS 代码的非官方修改，所以编写的 CSS 代码不会通过官方 W3C 的认证！以后经常会遇到这种情况，CSS 写的正确，通过 W3C 验证，但是不同浏览器显示效果不一样，用了 CSS Hack，显示的效果一样了，却又通不过 W3C 验证了，很是郁闷，不要为了标准而标准，W3C 验证只是帮你检查一下 [CSS](#) 语法错误，通过验证只不过是说明你到目前写的 CSS 代码没有语法错误而已，不要计较是否通过验证，只要做出来的页面代码量少，利于搜索引擎搜索，加载速度快，能为企业节省成本就可以！

好，我们开讲！

这节课我主要讲两个最常用的 CSS Hack，如果这两个能明白，再学其他的 Hack 就容易了

!important

作用：用来解决一些在 IE6 上显示的效果与 IE7/IE8/IE9/FireFox 上的效果不一样的情况。

比如有下面的一段代码：

```
#content{  
    height:960px !important;  
    height:900px;  
}
```

IE7/IE8/IE9/FireFox 可以识别上面附加 “!important” 的语句，看到此语句后，就不会再去执行第二句，尽管他们也“认识”第二句，但是附加 “!important” 的语句拥有绝对优先级，只要有它存在，第二句就不允许执行。附加 “!important” 语句 IE6 无法识别，所以 IE6 会跳过附加 “!important” 的语句直接去执行第二句 “height:900px”，第一句在 IE6 看来就是不存在的语句。

利用浏览器对加了 “!important” 语句的识别能力，来解决一些在 IE6 上显示的效果与 IE7/IE8/IE9/FireFox 上的效果不一样的情况。

***(星号)**

作用：用来解决一些在 IE6/IE7 上显示的效果与 IE8/IE9/FireFox 上的效果不一样的情况。

比如有下面的一段代码：

```
#content{  
    height:960px;  
    *height:900px;  
}
```

IE8/IE9/FireFox 不能识别附加有*的 CSS 属性语句，所以 IE8/IE9/FireFox 只能读第一句 “height:960px;” 而忽略第二句，IE6/IE7 可以识别附加有*的 CSS 属性语句，也就是说第一句和第二句都认识，所以它们先读第一句，将高度定为 960px，而后又读第二句

“*height:900px;” ，将高度从 960px 修改为 900px，所以我们在 IE 中看到的最终效果就是高度为 900px;

利用浏览器对加了 “*” 语句的识别能力，来解决一些在 IE6/IE7 上显示的效果与 IE8/IE9/FireFox 上的效果不一样的情况。

三、技巧篇

3.1 凸显专业的小技巧

3.1.1 单张图片按钮实例(CSS Sprites、CSS 精灵)

一般我们做按钮基本上都需要两张图片，一张正常状态的图片，一张按下去效果图片，做这种按钮思路就是，设置链接 a 的背景为第一张图片，a:hover 的背景为第二张图片



代码如下：

HTML 代码：

```
<a href="#" id="theLink"></a>
```

CSS 代码：

```
#theLink{  
    display:block;  
    width:120px;  
    height:41px;  
    margin:0 auto;  
    background:url(../images/normal.gif) no-repeat;  
}  
#theLink:hover{background:url(../images/press.gif) no-repeat;}
```


源代码： 两张图片按钮的源代码.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=264>

这节课，主要给大家介绍第二种思路，其实也很简单，首先我们将上面的两个图片合并成一张图片，如下：



其次，将上面的图片设置成按钮的背景，最后，将 a: hover 的背景向上移动 41 个像素就 OK 了

HTML 代码：

```
<a href="#" id="theLink"></a>
```


CSS 代码：

```
#theLink{
    display: block;
    width: 120px;
    height: 41px;
    margin: 0 auto;
    background: url(../images/buttonBG.gif) no-repeat;
}

#theLink: hover { background: url(../images/buttonBG.gif) no-repeat 0 -41px; }
```

这种图片整合技术，就是 **CSS Sprites**，也就是 **CSS 精灵**！

学到这里，大家是不是可以把[\[第七课 定位应用\]](#)的例子用 CSS Sprites 实现呢？

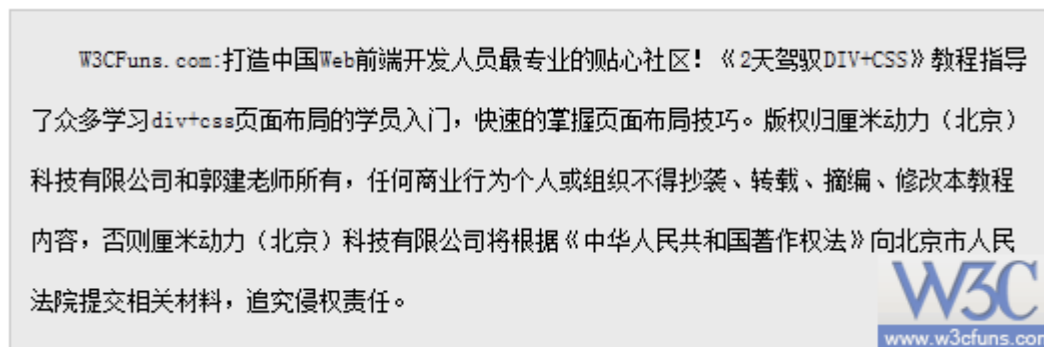
源代码： 单张图片按钮的源代码.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=264>

3.1.2 首行文字两文字缩进

记得过去刚开始学习制作页面的时候，要想让一段文字首行缩进两个文字，如下图：



总是在前面加上 8 个 “ ”，因为过去大家对 CSS 也不熟悉，这种方法实现虽然比较直接，但是文字多的时候会有很多 “ ” 充斥在代码中，代码显着比较乱，现在我们实现这种效果就简单多了，直接在 CSS 代码中加入

```
text-indent:2em;
```

就很容易实现文本段落的缩进~

text-indent:2em;

解释一下：text 的意思是文本，indent 在计算机英语中意思是缩进，至于后面的 2em 意思就是 2 个相对单位；

em 又是什么单位？

em 这个单位的意思就是文字的高度，1em 就是 1 个文字的高度，2em 就是两个文字高度，所以我们写的“text-indent:2em;”的意思就是，文本缩进 2 个文字的高度，因为汉字是方块字，高度和宽度是一样的，所以缩进 2 个文字的高度，就等于缩进两个文字的宽度，最后的效果就是缩进了两个文字。

【说明一点】

对于“text-indent:2em;”属性，只能加在块状元素上面，内联元素是不起作用的。

源代码： 文本缩进.rar

注意：本节中所有源代码下载页面：

<http://www.w3cfuns.com/forum.php?mod=viewthread&tid=241>

四、结束语

到此，页面布局入门级的教程《2 天驾驭 DIV+CSS》，到此结束了，在学习的过程中有任何问题，可以到 W3Cfuns.com 来提问，W3Cfuns.com 致力于打造中国 Web 前端开发人员最专业的贴心社区！在这里可以与更多的 Web 前端开发工程师互动，你在这里也会得到飞速的提高！

您还可以加入我们的 QQ 群：[点击查看 QQ 群列表](#)，希望大家遵守群规则！

当然你也可以参加我们[厘米 IT 学院](#)的学习班，学习中高级的课程，将自己的能力最大化提升，在短时间内成为一个经验丰富的 Web 前端开发工程师，[点击查看课程列表](#)，也许里面就有你最需要学习的课程。

有时间也可以到[厘米 IT 学院](#)和郭老师面对面沟通，增加对 Web 前端开发行业的了解，突破事业发展瓶颈，找准定位，拓展发展之路！

网络课堂即将上线，教程光盘即将发售！

敬请关注 www.w3cstudy.com 和 www.w3cfuns.com。

