# 《Div+CSS 布局大全》

整理者：Jesse Zhao

网站： http://JesseZhao.cnblogs.com

送给我最爱的女朋友蜜蜜，希望她可以永远快乐幸福!!!

# 目录

# div＋css 布局入门

你正在学习 CSS 布局吗？是不是还不能完全掌握纯 CSS 布局？通常有两种情况阻碍你的学习：

第一种可能是你还没有理解 CSS 处理页面的原理。在你考虑你的页面整体表现效果前，你应当先考虑内容的语义和结构，然后再针对语义、结构添加 CSS。这篇文章将告诉你应该怎样把 HTML 结构化。

另一种原因是你对那些非常熟悉的表现层属性(例如：cellpadding,、hspace、align="left"等等)束手无策，不知道该转换成对应的什么 CSS 语句。当你解决了第一种问题，知道了如何结构化你的 HTML，我再给出一个列表，详细列出原来的表现属性用什么 CSS 来代替。

结构化 HTML

我们在刚学习网页制作时，总是先考虑怎么设计，考虑那些图片、字体、颜色、以及布局方案。然后我们用 Photoshop 或者 Fireworks 画出来、切割成小图。最后再通过编辑 HTML 将所有设计还原表现在页面上。

如果你希望你的 HTML 页面用 CSS 布局(是 CSS-friendly 的)，你需要回头重来，先不考虑“外观”，要先思考你的页面内容的语义和结构。

外观并不是最重要的。一个结构良好的 HTML 页面可以以任何外观表现出来，CSS Zen Garden 是一个典型的例子。CSS Zen Garden 帮助我们最终认识到 CSS 的强大力量。

HTML 不仅仅只在电脑屏幕上阅读。你用 photoshop 精心设计的画面可能不能显示在 PDA、移动电话和屏幕阅读机上。但是一个结构良好的 HTML 页面可以通过 CSS 的不同定义，显示在任何地方，任何网络设备上。

开始思考

首先要学习什么是"结构"，一些作家也称之为"语义"。这个术语的意思是你需要分析你的内容块，以及每块内容服务的目的，然后再根据这些内容目的建立起相应的 HTML 结构。

如果你坐下来仔细分析和规划你的页面结构，你可能得到类似这样的几块：

标志和站点名称

主页面内容

站点导航(主菜单)

子菜单

搜索框

功能区(例如购物车、收银台)

页脚(版权和有关法律声明)

我们通常采用 DIV 元素来将这些结构定义出来，类似这样：

```
<div id="header"></div>
<div id="content"></div>
<div id="globalnav"></div>
<div id="subnav"></div>
<div id="search"></div>
<div id="shop"></div>
<div id="footer"></div>
```

这不是布局，是结构。这是一个对内容块的语义说明。当你理解了你的结构，就可以加对应的 ID 在 DIV 上。DIV 容器中可以包含任何内容块，也可以嵌套另一个 DIV。内容块可以包含任意的 HTML 元素---标题、段落、图片、表格、列表等等。

根据上面讲述的，你已经知道如何结构化 HTML，现在你可以进行布局和样式定义了。每一个内容块都可以放在页面上任何地方，再指定这个块的颜色、字体、边框、背景以及对齐属性等等。

使用选择器是件美妙的事

id 的名称是控制某一内容块的手段,通过给这个内容块套上 DIV 并加上唯一的 id,你就可以用 CSS 选择器来精确定义每一个页面元素的外观表现,包括标题、列表、图片、链接或者段落等等。例如你为#header 写一个 CSS 规则，就可以完全不同于#content 里的图片规则。

另外一个例子是：你可以通过不同规则来定义不同内容块里的链接样式。类似这样：#globalnav a:link 或者 #subnav a:link 或者#content a:link。你也可以定义不同内容块中相同元素的样式不一样。例如，通过#content p 和#footer p 分别定义#content 和#footer 中 p 的样式。从结构上讲，你的页面是由图片、链接、列表、段落等组成的，这些元素本身并不会对显示在什么网络设备中(PDA 还是手机或者网络电视)有影响，它们可以被定义为任何的表现外观。

一个仔细结构化的 HTML 页面非常简单，每一个元素都被用于结构目的。当你想缩进一个段落，不需要使用 blockquote 标签，只要使用 p 标签，并对 p 加一个 CSS 的 margin 规则就可以实现缩进目的。p 是结构化标签，margin 是表现属性，前者属于 HTML，后者属于 CSS。(这就是结构于表现的相分离.)

良好结构的 HTML 页面内几乎没有表现属性的标签。代码非常干净简洁。例如，原先的代码<table width="80%" cellpadding="3" border="2" align="left">，现在可以只在 HTML 中写<table>，所有控制表现的东西都写到 CSS 中去，在结构化的 HTML 中，table 就是表格，而不是其他什么(比如被用来布局和定位)。

亲自实践一下结构化

上面说的只是最基本的结构，实际应用中，你可以根据需要来调整内容块。常常会出现 DIV 嵌套的情况，你会看到"container"层中又有其它层，结构类似这样：

```
<div id="navcontainer">
<div id="globalnav">
<ul>a list</ul>
</div>
<div id="subnav">
<ul>another list</ul>
</div>
</div>
```

嵌套的 div 元素允许你定义更多的 CSS 规则来控制表现，例如：你可以给#navcontainer 一个规则让列表居右，再给#globalnav 一个规则让列表居左，而给#subnav 的 list 另一个完全不同的表现。

用 CSS 替换传统方法

下面的列表将帮助你用 CSS 替换传统方法：

HTML 属性以及相对应的 CSS 方法

HTML 属性 CSS 方法 说明

align="left"

align="right" float: left;

float: right; 使用 CSS 可以浮动 任何元素:图片、段落、div、标题、表格、列表等等

当你使用 float 属性，必须给这个浮动元素定义一个宽度。

marginwidth="0" leftmargin="0" marginheight="0" topmargin="0" margin: 0; 使用 CSS, margin 可以设置在任何元素上, 不仅仅是 body 元素.更重要的，你可以分别指定元素的 top, right, bottom 和 left 的 margin 值。

vlink="#333399" alink="#000000" link="#3333FF" a:link #3ff;

a:visited: #339;

a:hover: #999;

a:active: #00f;

在 HTML 中，链接的颜色作为 body 的一个属性值定义。整个页面的链接风格都一样。使用 CSS 的选择器，页面不同部分的链接样式可以不一样。

bgcolor="#FFFFFF" background-color: #fff; 在 CSS 中，任何元素都可以定义背景颜色，不仅仅局限于 body 和 table 元素。

bordercolor="#FFFFFF" border-color: #fff; 任何元素都可以设置边框(boeder)，你可以分别定义 top, right, bottom 和 left

border="3"

cellspacing="3" border-width: 3px; 用CSS,你可以定义table的边框为统一样式,也可以分别定义top, right, bottom and left 边框的颜色、尺寸和样式。

你可以使用 table, td or th 这些选择器.

如果你需要设置无边框效果，可以使用CSS定义： border-collapse: collapse;

<br clear="left">

<br clear="right">

<br clear="all">

clear: left;

clear: right;

clear: both;

许多2列或者3列布局都使用 float 属性来定位。如果你在浮动层中定义了背景颜色或者背景图片，你可以使用clear属性.

cellpadding="3"

vspace="3"

hspace="3" padding: 3px; 用CSS,任何元素都可以设定padding属性,同样,padding可以分别设置top, right, bottom and left。padding是透明的。

align="center" text-align: center;

margin-right: auto; margin-left: auto;

Text-align 只适用于文本.

象div，p这样的块级怨毒可以通过margin-right: auto; 和margin-left: auto;来水平居中

一些令人遗憾的技巧和工作环境

由于浏览器对CSS支持的不完善，我们有时候不得不采取一些技巧(hacks)或建立一种环境(Workarounds)来让CSS实现传统方法同样的效果。例如块级元素有时侯需要使用水平居中的技巧，盒模型 bug 的技巧等等。所有这些技巧都在 Molly Holzschlag 的文章《Integrated Web Design: Strategies for Long-Term CSS Hack Management》中有详细说明。

另外一个关于CSS技巧的资源站点是Big John和Holly Bergevin的“Position is Everything”。

# XHTML下css+div布局总结

**xml**(extensible Markup Language)的出现，结构化文档和数据有了一个通用的、科适应的格式，不仅仅应用在web上，也可以应用在任何地方。标准称为可能。

**XHTML** 是 The Extensible HyperText Markup Language 可扩展标识语言的缩写。在HTML4.0的基础上，用XML的规则对其进行扩展，得到了XHTML。它实现HTML向XML的过渡。

CSS是Cascading Style Sheets层叠样式表的缩写。纯CSS布局与结构式XHTML相结合能帮助设计师分离外观与结构，使站点的访问及维护更加容易。

**1)为页面添加正确的DOCTYPE**

DOCTYPE是document type的简写。主要用来说明你用的XHTML或者HTML是什么版本。浏览器根据你 DOCTYPE定义的DTD(文档类型定义)来解释页面代码。

XHTML1.0提供了三种DOCTYPE可选择：

(1)过渡型（Transitional ）--使用非常普遍。

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

(2)严格型（Strict ）

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1 -strict.dtd">

(3)框架型（Frameset ）

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

**2)设定一个名字空间（Namespace）**

直接在 DOCTYPE 声明后面添加如下代码：

<html XMLns="http://www.w3.org/1999/xhtml" >

一个 namespace 是收集元素类型和属性名字的一个详细的 DTD，namespace 声明允许你通过一个在线地址指向来识别你的 namespace。只要照样输入代码就可以。

**3)声明你的编码语言**

为了被浏览器正确解释和通过标识校验，所有的 XHTML 文档都必须声明它们所使用的编码语言。代码如下：

<meta http-equiv="Content-Type" content="text/html; charset=GB2312" />

这里声明的编码语言是简体中文 GB2312，你如果需要制作繁体内容，可以定义为 BIG5。

**4)用小写字母书写所有的标签**

XML 对大小写是敏感的，所以，XHTML 也是大小写有区别的。所有的 XHTML 元素和属性的名字都必须使用小写。否则你的文档将被 W3C 校验认为是无效的。例如下面的代码是不正确的：

**5)为图片添加 alt 属性**

为所有图片添加 alt 属性。alt 属性指定了当图片不能显示的时候就显示供替换文本，这样做对正常用户可有可无，但对纯文本浏览器和使用屏幕阅读机的用户来说是至关重要的。只有添加了 alt 属性，代码才会被 W3C 正确性校验通过。注意的是我们要添加有意义的 alt 属性，象下面这样的写法毫无意义：

<img src="logo.gif" alt="logo.gif">

正确的写法：

<img src="logo.gif" alt="互动力工作室标志，点击返回首页">

**6)给所有属性值加引号**

在 HTML 中，你可以不需要给属性值加引号，但是在 XHTML 中，它们必须被加引号。还必须用空格分开属性。

例：<hr width="75%"size="7"/> 这也是不正确的

**7)关闭所有的标签**

在 XHTML 中，每一个打开的标签都必须关闭。空标签也要关闭，在标签尾部使用一个正斜杠 "/"来关闭它们自己。例如：

<br />

**8)收藏夹小图标**

例如：首先制作一个 16x16 的 icon 图标，命名为 favicon.ico，放在根目录下。然后将下面的代码嵌入 head 区：

<link rel="icon" href="/favicon.ico" type="image/x-icon" />

<link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />

**9)用 CSS 定义元素外观**

用 css 布局的一个好处是可以批量对页面进行修改，它能将文档结构和表现层分离开来，减轻工作量和服务器的负荷，增加站点的扩展能力和应用。

css 是不区别空格和大小写的，下面是一些基础的归纳

(1)颜色值

颜色值可以用 RGB 值写，例如：color : rgb(255,0,0)，也可以用十六进制写，就象上面例子 color:#FF0000。如果十六进制值是成对重复的可以简写，效果一样。例如:#FF0000 可以写成#F00。但如果不重复就不可以简写，例如#FC1A1B 必须写满六位。

(2)定义字体

web 标准推荐如下字体定义方法：

```
body { font-family : "Lucida Grande", Verdana, Lucida, Arial, Helvetica, 宋体,sans-serif; }
```

字体按照所列出的顺序选用。如果用户的计算机含有 Lucida Grande 字体，文档将被指定为 Lucida Grande。没有的话，就被指定为 Verdana 字体，如果也没有 Verdana，就指定为 Lucida 字体，依此类推，；

Lucida Grande 字体适合 Mac OS X；

Verdana 字体适合所有的 Windows 系统；

Lucida 适合 UNIX 用户

"宋体"适合中文简体用户；

如果所列出的字体都不能用，则默认的 sans-serif 字体能保证调用；

(3)群选择器

当几个元素样式属性一样时，可以共同调用一个声明，元素之间用逗号分隔，：

```
p, td, li { font-size : 12px ; }
```

(4)派生选择器

可以使用派生选择器给一个元素里的子元素定义样式，例如这样：

```
li strong { font-style : italic; font-weight : normal; }
```

就是给 li 下面的子元素 strong 定义一个斜体不加粗的样式。

(5)id 选择器

用 CSS 布局主要用层"div"来实现，而 div 的样式通过"id 选择器"来定义。例如我们首先定义一个层

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

然后在样式表里这样定义：

```
#menubar {MARGIN: 0px;BACKGROUND: #FEFEFE;COLOR: #666;}
```

其中"menubar"是你自己定义的 id 名称。注意在前面加"#"号。

id 选择器也同样支持派生，例如：

```
#menubar p { text-align : right; margin-top : 10px; }
```

这个方法主要用来定义层和那些比较复杂，有多个派生的元素。

(6)类别选择器

在 CSS 里用一个点开头表示类别选择器定义，例如：

```
.14px {color : #f60 ;font-size:14px ;}
```

在页面中，用 class= "类别名"的方法调用：

```
<span class="14px">14px 大小的字体</span>
```

这个方法比较简单灵活，可以随时根据页面需要新建和删除。

(7)定义链接的样式

CSS 中用四个伪类来定义链接的样式，分别是：a:link、a:visited、a:hover 和 a : active，例如：

```
a:link{font-weight : bold ;text-decoration : none ;color : #c00 ;}
a:visited {font-weight : bold ;text-decoration : none ;color : #c30 ;}
a:hover {font-weight : bold ;text-decoration : underline ;color : #f60 ;}
a:active {font-weight : bold ;text-decoration : none ;color : #F90 ;}
```

以上语句分别定义了 "链接、已访问过的链接、鼠标停在上方时、点下鼠标时"的样式。注意，必须按以上顺序写，否则显示可能和你预想的不一样。记住它们的顺序是“LVHA”。

(8)组合使用选择器创造精致的设计效果

用漂亮的图案代替普通无序列表前沉闷的黑点。站点 http://marine.happycog.com/

先用 css 规则告诉类别属性 inventory 的无序列表。

```
ul.inventory{
      list-style:disc url(/images/common/lister2.gig) inside;}
```

它的调用标记:

```
<ul class="inventory">
<li><a href="/angelfish">Angelfish</a>(67 items)</li>
<li><a href="/angeld">Angels/Frogfish</a>(35 items)</li>
<li><a href="/anthias">Angelfish</a>(5526 items)</li>
<li><a href="/basslets">Angelfish</a>(15 items)</li>
<ul>
```

(9)缩写是按照顺时针的顺序

```
margin:25px 0 25px 0;
```

(10)行高

```
line-height:150% 说明行距为正常的 150%
```

**10)结构化代码 div(division)、id、class**

用它们来书写紧凑的 xhtml，更明智的使用 css.

(1)结构化 id 标签，与 class 的有区别：

如果你的属性页面包含了一个 div，它的 id 为"content",它就不可能有另外一个 div 或者其他元素拥有相同的名字。相反，class 属性可以在意个页面中一次又一次地使用。

(2)id 的规则

一个 id 值必须用一个字母或者下划线开头，它不能用一个数字进行开头，然后跟随字母、数字和下划线。空格和连字符-都是不允许的。

**11）制作好的网站可以到 w3c 进行标准校正**

http:validator.w3.org

http://jigsaw.w3.org/css-validator/

# 网页设计 DIV+CSS——第 1 天:选择什么样的 DOCTYPE

## 前言

大家好！这个系列文章是按阿捷自己制作这个站点的过程编写的。之前阿捷也一直没有制作过一个真正符合 web 标准的网站。现在边参考国外资料边制作，同时把过程中的心得和经验记录下来，希望对大家有点帮助。好了，让我们开始吧

## 第一天

开始制作符合标准的站点，第一件事情就是声明符合自己需要的 DOCTYPE。

查看本站首页原代码，可以看到第一行就是：

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

打开一些符合标准的站点，例如著名 web 设计软件开发商 Macromedia，设计大师 Zeldman 的个人网站，会发现同样的代码。

而另一些符合标准的站点(例如 k10k.net)的代码则如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

那么这些代码有什么含义？一定要放置吗？

# 什么是 DOCTYPE

上面这些代码我们称做 DOCTYPE 声明。DOCTYPE 是 document type(文档类型)的简写，用来说明你用的 XHTML 或者 HTML 是什么版本。

其中的 DTD(例如上例中的 xhtml1-transitional.dtd)叫文档类型定义，里面包含了文档的规则，浏览器就根据你定义的 DTD 来解释你页面的标识，并展现出来。

要建立符合标准的网页，DOCTYPE 声明是必不可少的关键组成部分；除非你的 XHTML 确定了一个正确的 DOCTYPE，否则你的标识和 CSS 都不会生效。

XHTML 1.0 提供了三种 DTD 声明可供选择：

- 过渡的(Transitional):要求非常宽松的 DTD，它允许你继续使用 HTML4.01 的标识(但是要符合 xhtml 的写法)。完整代码如下：

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

- 严格的(Strict):要求严格的 DTD，你不能使用任何表现层的标识和属性，例如<br>。完整代码如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

- 框架的(Frameset):专门针对框架页面设计使用的 DTD，如果你的页面中包含有框架，需要采用这种 DTD。完整代码如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

# 我们选择什么样的 DOCTYPE

理想情况当然是严格的 DTD，但对于我们大多数刚接触 web 标准的设计师来说，过渡的 DTD(XHTML 1.0 Transitional)是目前理想选择(包括本站，使用的也是过渡型 DTD)。因为这种 DTD 还允许我们使用表现层的标识、元素和属性，也比较容易通过 W3C 的代码校验。

注：上面说的"表现层的标识、属性"是指那些纯粹用来控制表现的 tag，例如用于排版的表格、背景颜色标识等。在 XHTML 中标识是用来表示结构的，而不是用来实现表现形式，我们过渡的目的是最终实现数据和表现相分离。

打个比方：人体模特换衣服。模特就好比数据，衣服则是表现形式，模特和衣服是分离的，这样你就可以随意换衣服。而原来 HTML4 中，数据和表现是混杂在一起的，要一次性换个表现形式非常困难。呵呵，有点抽象了，这个概念需要我们在应用过程中逐步领会。

# 补充

DOCTYPE 声明必须放在每一个 XHTML 文档最顶部，在所有代码和标识之上。

# 网页设计 DIV+CSS——第 2 天:什么是名字空间

DOCTYPE 声明好以后，接下来的代码是：

<html xmlns="http://www.w3.org/1999/xhtml" lang="gb2312">

通常我们 HTML4.0 的代码只是<html>，这里的"xmlns"是什么呢？

这个"xmlns"是 XHTML namespace 的缩写，叫做"名字空间"声明。名字空间是什么作用呢？阿捷自己的理解是：

由于 xml 允许你自己定义自己的标识，你定义的标识和其他人定义的标识有可能相同，但表示不同的意义。当文件交换或者共享的时候就容易产生错误。为了避免这种错误发生，XML 采用名字空间声明，允许你通过一个网址指向来识别你的标识。
例如：
小王和小李都定义了一个<book>标识，如果小王的名字空间是"http://www.xiaowang.com"，小李的名字空间是"http://www.xiaoli.com"，那么当两个文档交换数据时，也不会混淆<book>标识，因为它属于不同的名字空间。
更通俗的解释是：名字空间就是给文档做一个标记，告诉别人，这个文档是属于谁的。只不过这个"谁"用了一个网址来代替。
XHTML 是 HTML 向 XML 过渡的标识语言，它需要符合 XML 文档规则，因此也需要定义名字空间。又因为 XHTML1.0 不能自定义标识，所以它的名字空间都相同，就是"http://www.w3.org/1999/xhtml"。如果你还不太理解也不要紧，目前阶段我们只要照抄代码就可以了。
后面的 lang="gb2312",指定你的文档用简体中文。

# 网页设计 DIV+CSS——第 3 天:定义语言编码

第三步是定义你的语言编码，类似这样：
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
为了被浏览器正确解释和通过 W3C 代码校验，所有的 XHTML 文档都必须声明它们所使用的编码语言，我们一般使用 gb2312(简体中文)，制作多国语言页面也有可能用 Unicode、ISO-8859-1 等，根据你的需要定义。
通常这样定义就可以了。但是要补充说明的是，XML 文档并不是这样定义语言编码的，XML 的定义方式如下：
<?xml version="1.0" encoding="gb2312"?>
你在 Macromedia.com 的首页代码第一行就可以看见类似的语句，这也是 W3C 推荐使用的定义方法。那为什么我们不直接采用这种方法呢？原因是一些浏览器对标准的支持不完善，不能正确理解这样的定义方法，比如 IE6/windows。所以在目前过渡方案下，我们依然推荐使用 meta 方式。当然，你可以两种方法都写。
看本站源代码，你会发现语言编码定义的地方还多一句：
<meta http-equiv="Content-Language" content="gb2312" />
这是针对老版本浏览器写的，以保证各种浏览器都能正确解释页面。
注意：在上面声明语句的最后，你看到有一个斜杠"/"，这和我们以前的 HTML4.0 的代码写法不同。原因是 XHTML 语法规则要求所有的标识都必须有开始和结束。例如<body>和</body>、 和 等，对于不成对的标识，要求在标识最后加一个空格，然后跟一个"/"。例如<br>写成<br />、<img>写成<img />，加空格的原因是避免代码连在一起浏览器不识别。

# 网页设计 DIV+CSS——第 4 天:调用样式表

用 web 标准设计网站，过渡的方法主要是采用 XHTML+CSS，css 样式表是必不可少的。这就要求所有网页设计师必须熟练掌握 CSS，如果你以前不常用，那么现在就开始学习吧。要制作符合 web 标准的网站，不懂 CSS 是设计不出漂亮的页面的。
事实上，所有表现的地方都需要用 CSS 来实现。我们以前都习惯用 table 来定位和布局，现在要改用 DIV 来定位和布局。这是思维方式的变化，一开始有些不习惯。呵呵，任何变革都会有阻力的，为了享受标准带来的"益处"，放弃一些老的传统做法是值得的。

## 外部调用样式表

在以前，我们通常采用 2 种方法使用样式表：

- 页面内嵌法：就是将样式表直接写在页面代码的 head 区。类似这样：

<style type="text/css"> <!-- body { background : white ; color : black ; } --> </style>

- 外部调用法：将样式表写在一个独立的.css 文件中，然后在页面 head 区用类似以下代码调用。

<link rel="stylesheet" rev="stylesheet" href="css/style.css" type="text/css" media="all" />

在符合 web 标准的设计中，我们使用外部调用法，好处不言而喻，你可以不修改页面只修改.css 文件而改变页面的样式。如果所有页面都调用同一个样式表文件，那么改一个样式表文件，可以改变所有文件的样式。

## 双表法调用样式表

查看某些符合标准站点的原代码，你可能看到，在调用样式表的地方有如下 2 句：

<link rel="stylesheet" rev="stylesheet" href="css/style.css" type="text/css" media="all" /> <style type="text/css" media="all">@import url( css/style01.css );</style>

为什么要写两次呢？

实际上一般情况下用外联法调用(就是第一句)就足够了。我这里使用双表调用只是一种示例。其中的"@import"命令用于输入样式表。而"@import"命令在 netscape 4.0 版本浏览器是无效的。也就是说，当你希望某些效果在 netscape 4.0 浏览器中隐藏，在 4.0 以上或其它浏览器中又显示的时候，你可以采用"@import"命令方法调用样式表。

# 网页设计 DIV+CSS——第 5 天:head 区的其他设置

这些技巧主要讲 meta 标签设置的，其实与符合 web 标准关系不大，只要注意在最后加"/"关闭标签就可以，但是既然是入门教程，就写得详细一点吧。

## 收藏夹小图标

如果你将本站加入收藏夹，可以看到在收藏夹网址之前的 IE 图标变成了本站特别的图标。要实现这样效果很简单，首先制作一个 16x16 的 icon 图标，命名为 favicon.ico，放在根目录下。然后将下面的代码嵌入 head 区：

<link rel="icon" href="/favicon.ico" type="image/x-icon" />

<link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />

## 为搜索引擎准备的内容

代码如下，替换成你自己站点的内容就可以：

- 允许搜索机器人搜索站内所有链接。如果你想某些页面不被搜索，推荐采用 robots.txt 方法

<meta content="all" name="robots" />

- 设置站点作者信息

<meta name="author" content="ajie@netease.com,阿捷" />

- 设置站点版权信息

<meta name="Copyright" content="www.w3cn.org,自由版权,任意转载" />

- 站点的简要介绍(推荐)

<meta name="description" content="新网页设计师。web 标准的教程站点，推动 web 标准在中国的应用" />

- 站点的关键词(推荐)

<meta content="designing, with, web, standards, xhtml, css, graphic, design, layout, usability, ccessibility, w3c, w3, w3cn, ajie" name="keywords" />

先介绍这么多。补充说明，前面花了 5 节都是讲 head 区的代码，实际页面内容还一字未提，呵呵，不要急，其实 head 区是非常重要的，看一个页面的 head 的代码就可以知道设计师是否够专业。

# 网页设计 DIV+CSS——第 6 天:XHTML 代码规范

在开始正式内容制作之前，我们必须先了解一下 web 标准有关代码的规范。了解这些规范可以帮助你少走弯路,尽快通过代码校验。

# 1.所有的标记都必须要有一个相应的结束标记

以前在 HTML 中，你可以打开许多标签，例如 和<li>而不一定写对应的 和</li>来关闭它们。但在 XHTML 中这是不合法的。XHTML 要求有严谨的结构，所有标签必须关闭。如果是单独不成对的标签，在标签最后加一个"/"来关闭它。例如:

<br /><img height="80" alt="网页设计师" src="../images/logo_w3cn_200x80.gif" width="200" />

# 2.所有标签的元素和属性的名字都必须使用小写

与 HTML 不一样，XHTML 对大小写是敏感的，<title>和<TITLE>是不同的标签。XHTML 要求所有的标签和属性的名字都必须使用小写。例如：<BODY>必须写成<body> 。大小写夹杂也是不被认可的，通常 dreamweaver 自动生成的属性名字"onMouseOver"也必须修改成"onmouseover"。

# 3.所有的 XML 标记都必须合理嵌套

同样因为 XHTML 要求有严谨的结构，因此所有的嵌套都必须按顺序，以前我们这样写的代码：

必须修改为：

就是说，一层一层的嵌套必须是严格对称。

# 4.所有的属性必须用引号""括起来

在 HTML 中，你可以不需要给属性值加引号，但是在 XHTML 中，它们必须被加引号。例如:

<height=80>

必须修改为：

<height="80">

特殊情况，你需要在属性值里使用双引号，你可以用"，单引号可以使用'，例如：

<alt="say'hello'">

# 5.把所有<和&特殊符号用编码表示

- 任何小于号（<），不是标签的一部分，都必须被编码为& l t ；
- 任何大于号（>），不是标签的一部分，都必须被编码为& g t ；
- 任何与号（&），不是实体的一部分的，都必须被编码为& a m p;

注：以上字符之间无空格。

# 6.给所有属性赋一个值

XHTML 规定所有属性都必须有一个值，没有值的就重复本身。例如：

<td nowrap> <input type="checkbox" name="shirt" value="medium" checked>

必须修改为：

<td nowrap="nowrap"> <input type="checkbox" name="shirt" value="medium" checked="checked">

# 7.不要在注释内容中使“--”

“--”只能发生在 XHTML 注释的开头和结束，也就是说，在内容中它们不再有效。例如下面的代码是无效的:

<!--这里是注释----------这里是注释-->

用等号或者空格替换内部的虚线。

<!--这里是注释==========这里是注释-->

以上这些规范有的看上去比较奇怪，但这一切都是为了使我们的代码有一个统一、唯一的标准，便于以后的数据再利用。

# 网页设计 DIV+CSS——第 7 天:CSS 入门

在了解 XHTML 代码规范后，我们就要进行 CSS 布局。首先先介绍一些 CSS 的入门知识。如果你已经很熟悉了，可以跳过这一节，直接进入下一节。

CSS 是 Cascading Style Sheets(层叠样式表)的缩写。是一种对 web 文档添加样式的简单机制，属于表现层的布局语言。

## 1.基本语法规范

分析一个典型 CSS 的语句：

```
p {COLOR:#FF0000;BACKGROUND:#FFFFFF}
```

- 其中"p"我们称为"选择器"(selectors)，指明我们要给"p"定义样式；
- 样式声明写在一对大括号"{}"中；
- COLOR 和 BACKGROUND 称为"属性"(property)，不同属性之间用分号";"分隔；
- "#FF0000"和"#FFFFFF"是属性的值(value)。

## 2.颜色值

颜色值可以用 RGB 值写，例如：color : rgb(255,0,0)，也可以用十六进制写，就象上面例子 color:#FF0000。如果十六进制值是成对重复的可以简写，效果一样。例如:#FF0000 可以写成#F00。但如果不重复就不可以简写，例如#FC1A1B 必须写满六位。

## 3.定义字体

web 标准推荐如下字体定义方法：

```
body { font-family : "Lucida Grande", Verdana, Lucida, Arial, Helvetica, 宋体,sans-serif; }
```

- 字体按照所列出的顺序选用。如果用户的计算机含有 Lucida Grande 字体，文档将被指定为 Lucida Grande。没有的话，就被指定为 Verdana 字体，如果也没有 Verdana，就指定为 Lucida 字体，依此类推,；
- Lucida Grande 字体适合 Mac OS X；
- Verdana 字体适合所有的 Windows 系统；
- Lucida 适合 UNIX 用户
- "宋体"适合中文简体用户；
- 如果所列出的字体都不能用，则默认的 sans-serif 字体能保证调用；

## 4.群选择器

当几个元素样式属性一样时，可以共同调用一个声明，元素之间用逗号分隔，： p, td, li { font-size : 12px ; }

## 5.派生选择器

可以使用派生选择器给一个元素里的子元素定义样式，例如这样：

```
li strong { font-style : italic; font-weight : normal; }
```

就是给 li 下面的子元素 strong 定义一个斜体不加粗的样式。

## 6.id 选择器

用 CSS 布局主要用层"div"来实现，而 div 的样式通过"id 选择器"来定义。例如我们首先定义一个层

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

然后在样式表里这样定义：

```
#menubar {MARGIN: 0px;BACKGROUND: #FEFEFE;COLOR: #666;}
```

其中"menubar"是你自己定义的 id 名称。注意在前面加"#"号。

id 选择器也同样支持派生，例如：

```
#menubar p { text-align : right; margin-top : 10px; }
```

这个方法主要用来定义层和那些比较复杂，有多个派生的元素。

# 6.类别选择器

在 CSS 里用一个点开头表示类别选择器定义，例如：

```
.14px {color : #f60 ;font-size:14px ;}
```

在页面中，用 class="类别名"的方法调用：

```
<span class="14px">14px 大小的字体</span>
```

这个方法比较简单灵活，可以随时根据页面需要新建和删除。

# 7.定义链接的样式

CSS 中用四个伪类来定义链接的样式，分别是：a:link、a:visited、a:hover 和 a : active，例如：

```
a:link{font-weight : bold ;text-decoration : none ;color : #c00 ;}
a:visited {font-weight : bold ;text-decoration : none ;color : #c30 ;}
a:hover {font-weight : bold ;text-decoration : underline ;color : #f60 ;}
a:active {font-weight : bold ;text-decoration : none ;color : #F90 ;}
```

以上语句分别定义了"链接、已访问过的链接、鼠标停在上方时、点下鼠标时"的样式。注意，必须按以上顺序写，否则显示可能和你预想的不一样。记住它们的顺序是“LVHA”。

呵呵，看了这么多，有点头晕吧，实际上 CSS 的语法规范还有很多，这里列的只是一些常用的，毕竟我们是循序渐进，不可能一口吃成胖子:)

# 网页设计 DIV+CSS——第 8 天:CSS 布局入门

CSS 布局与传统表格(table)布局最大的区别在于：原来的定位都是采用表格，通过表格的间距或者用无色透明的 GIF 图片来控制文布局版块的间距；而现在则采用层(div)来定位，通过层的 margin,padding,border 等属性来控制版块的间距。

# 1.定义 DIV

分析一个典型的定义 div 例子：

```
#sample{ MARGIN: 10px 10px 10px 10px;
PADDING:20px 10px 10px 20px;
BORDER-TOP: #CCC 2px solid;
BORDER-RIGHT: #CCC 2px solid;
BORDER-BOTTOM: #CCC 2px solid;
BORDER-LEFT: #CCC 2px solid;
BACKGROUND: url(images/bg_poem.jpg) #FEFEFE no-repeat right bottom;
COLOR: #666;
TEXT-ALIGN: center;
LINE-HEIGHT: 150%; WIDTH:60%; }
```

说明如下：

- 层的名称为 sample，在页面中用
  就可以调用这个样式。
- MARGIN 是指层的边框以外留的空白，用于页边距或者与其它层制造一个间距。"10px 10px 10px 10px"分别代表"上右下左"(顺时针方向)四个边距，如果都一样，可以缩写成"MARGIN: 10px;"。如果边距为零，要写成"MARGIN: 0px;"。

注意：当值是零时，除了 RGB 颜色值 0%必须跟百分号，其他情况后面可以不跟单位"px"。MARGIN 是透明元素，不能定义颜色。

- PADDING 是指层的边框到层的内容之间的空白。和 margin 一样，分别指定上右下左边框到内容的距离。如果都一样，可以缩写成"PADDING:0px"。单独指定左边可以写成"PADDING-LEFT: 0px;"。PADDING 是透明元素，不能定义颜色。
- BORDER 是指层的边框，"BORDER-RIGHT: #CCC 2px solid;"是定义层的右边框颜色为"#CCC"，宽度为"2px"，样式为"solid"直线。如果要虚线样式可以用"dotted"。
- BACKGROUND 是定义层的背景。分 2 级定义，先定义图片背景，采用"url(../images/bg_logo.gif)"来指定背景图片路径；其次定义背景色"#FEFEFE"。"no-repeat"指背景图片不需要重复，如果需要横向重复用"repeat-x",纵向重复用"repeat-y",重复铺满整个背景用"repeat"。后面的"right bottom;"是指背景图片从右下角开始。如果没有背景图片可以只定义背景色 BACKGROUND: #FEFEFE
- COLOR 用于定义字体颜色，上一节已经介绍过。
- TEXT-ALIGN 用来定义层中的内容排列方式，center 居中,left 居左,right 居右。
- LINE-HEIGHT 定义行高，150%是指高度为标准高度的 150%，也可以写作：LINE-HEIGHT:1.5 或者 LINE-HEIGHT:1.5em，都是一样的意思。
- WIDTH 是定义层的宽度，可以采用固定值，例如 500px，也可以采用百分比，象这里的"60%"。要注意的是:这个宽度仅仅指你内容的宽度，不包含 margin,border 和 padding。但在有些浏览器中不是这么定义的，需要你多试试。

下面是这个层的实际表现:

这里是演示内容，这里是演示内容，这里是演示内容，这里是演示内容，这里是演示内容，这里是演示内容，这里是演示内容，这里是演示内容，

这里是演示内容，这里是演示内容，

这里是演示内容，这里是演示内容，

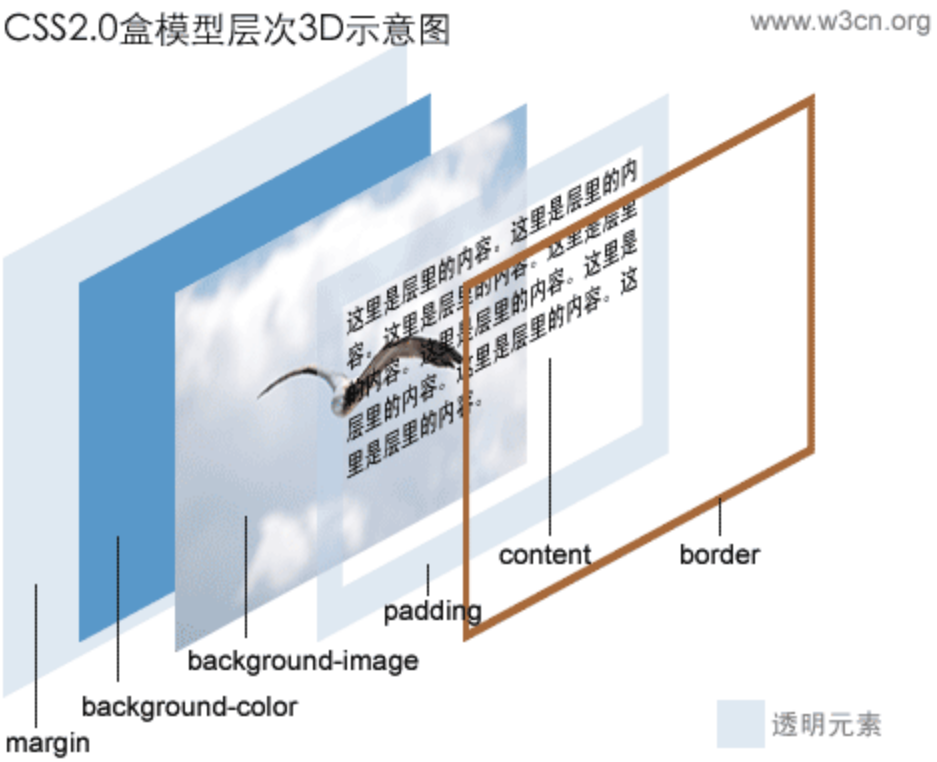
这里是演示内容...

我们可以看到边框是 2px 的灰色，背景图片在右下没有重复，内容距离上和左边框 20px，内容居中，一切和预想的一样。hoho，虽然不好看，但它是最基本的，掌握了它，你就已经学会一半的 CSS 布局技术了。就是这样，不算难吧！(另一半是什么？另一半是层与层之间的定位。我会在后面逐步讲解。)

# 2.CSS2 盒模型

自从 1996 年 CSS1 的推出，W3C 组织就建议把所有网页上的对像都放在一个盒(box)中，设计师可以通过创建定义来控制这个盒的属性，这些对像包括段落、列表、标题、图片以及层

。盒模型主要定义四个区域：内容(content)、边框距(padding)、边界(border)和边距(margin)。上面我们讲的 sample 层就是一个典型的盒。对于初学者，经常会搞不清楚 margin，background-color，background-image，padding，content，border 之间的层次、关系和相互影响。这里提供一张盒模型的 3D 示意图，希望便于你的理解和记忆。




# 3.辅助图片一律用背景处理

用 XHTML+CSS 布局，有一个技术一开始让你不习惯，应该说是一种思维方式与传统表格布局不一样，那就是：所有辅助图片都用背景来实现。类似这样：

```
BACKGROUND: url(images/bg_poem.jpg) #FEFEFE no-repeat right bottom;
```

尽管可以用 直接插在内容中，但这是不推荐的。这里的"辅助图片"是指那些不是作为页面要表达的内容的一部分，而仅仅用于修饰、间隔、提醒的图片。例如：相册中的图片、图片新闻中的图片，上面的 3d 盒模型图片都属于内容的一部分，

它们可以用 元素直接插在页面里，而其它的类似 logo，标题图片，列表前缀图片都必须采用背景方式或者其他 CSS 方式显示。

这样做的原因有 2 点：

- 将表现与结构彻底相分离(参考阅读另一篇文章:《理解表现与结构相分离》)，用 CSS 控制所有的外观表现，便于改版。
- 使页面更具有易用性，更有亲和力。例如：盲人使用屏幕阅读机，用背景技术实现的图片就不会被朗读出来。

# 网页设计 DIV+CSS——第 9 天:第一个 CSS 布局实例

接下来开始要真正设计布局了。和传统的方法一样，你首先要在脑海里有大致的轮廓构想，然后用 photoshop 把它画出来。你可能看到有关 web 标准的站点大都很朴素，因为 web 标准更关注结构和内容，实际上它与网页的美观没有根本冲突，你想怎么设计就怎么设计，用传统表格方法实现的布局，用 DIV 也可以实现。技术有一个成熟的过程，把 DIV 看成和 TABLE 一样的工具，如何运用就看你的想象力了。

注：在实际应用过程中，DIV 在有些地方的确不如表格方便，比如背景色的定义。但任何事情都有得有失，取舍在于你的价值判断。好，不罗嗦了，我们开始：

# 1.确定布局

w3cn 的最初设计草图如下：

用表格的设计方法的话，一般都是上中下三行布局  。用 DIV 的话，我考虑使用三列来布局,成为这样  。

# 2.定义 body 样式

先定义整个页面的 body 的样式,代码如下：

```
body { MARGIN: 0px;
PADDING: 0px;
BACKGROUND: url(../images/bg_logo.gif) #FEFEFE no-repeat right bottom;
FONT-FAMILY: 'Lucida Grande','Lucida Sans Unicode','宋体','新宋体',arial,verdana,sans-serif;
COLOR: #666;
FONT-SIZE:12px;
LINE-HEIGHT:150%; }
```

以上代码的作用在上一天的教程有详细说明，大家应该一看就明白。定义了边框边距为 0；背景颜色为#FEFEFE，背景图片为 bg_logo.gif，图片位于页面右下角，不重复；定义了字体尺寸为 12px；字体颜色为#666;行高 150%。

# 3.定义主要的 div

初次使用 CSS 布局，我决定采用固定宽度的三列布局(比自适应分辨率的设计简单,hoho，别说我偷懒，先实现简单的，增加点信心嘛！)。分别定义左中右的宽度为 200:300:280，在 CSS 中如下定义：

```
/*定义页面左列样式*/
#left{ WIDTH:200px;
MARGIN: 0px;
PADDING: 0px;
BACKGROUND: #CDCDCD;
}
/*定义页面中列样式*/
#middle{ POSITION: absolute;
LEFT:200px;
```

```
TOP:0px;
WIDTH:300px;
MARGIN: 0px;
PADDING: 0px;
BACKGROUND: #DADADA;
}
/*定义页面右列样式*/
#right{ POSITION: absolute;
LEFT:500px;
TOP:0px;
WIDTH:280px;
MARGIN: 0px;
PADDING: 0px;
BACKGROUND: #FFF; }
```

注意：定义中列和右列 div 我都采用了 POSITION: absolute;，然后分别定义了 LEFT:200px;TOP:0px;和 LEFT:500px;TOP:0px;这是这个布局的关键，我采用了层的绝对定位。定义中间列距离页面左边框 200px，距离顶部 0px；定义右列距离页面左边框 500px，距离顶部 0px；。

这时候整个页面的代码是：

```
<!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" lang="gb2312">
<head>
<title>欢迎进入新《网页设计师》:web 标准教程及推广</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<meta http-equiv="Content-Language" content="gb2312" />
<meta content="all" name="robots" />
<meta name="author" content="ajie(at)netease.com,阿捷" />
<meta name="Copyright" content="www.w3cn.org,自由版权,任意转载" />
<meta name="description" content="新网页设计师,web 标准的教程站点,推动 web 标准在中国的应用." />
<meta content="web 标准,教程,web, standards, xhtml, css, usability, accessibility" name="keywords" />
<link rel="icon" href="/favicon.ico" type="image/x-icon" />
<link rel="shortcut icon" href="http://www.w3cn.org/favicon.ico" type="image/x-icon" />
<link rel="stylesheet" rev="stylesheet" href="css/style01.css" type="text/css" media="all" />
</head>
<body>
<div id="left">页面左列</div>
<div id="middle">页面中列</div>
<div id="right">页面右列</div>
</body>
</html>
```

这时候页面的效果仅仅可以看到三个并列的灰色矩形，和一个背景图。但是我希望高度是满屏的，怎么办呢？

# 4.100%自适应高度?

为了保持三列有同样的高度，我尝试在#left、#middle 和#right 中设置"height:100%;"，但发现完全没有预想的自适应高度效果。经过一番尝试后，我只好给每个 div 一个绝对高度:"height:1000px;"，并且随着内容的增加，需要不断修正这个值。难道没有办法自适应高度了吗？随着阿捷自己学习的深入，发现一个变通的解决办法，实际上根本不需要设置 100%，我们已经被 table 思维禁锢太深了，这个办法在下一节的学习中详细介绍。

# 网页设计 DIV+CSS——第 10 天:自适应高度

如果我们想在 3 列布局的最后加一行页脚，放版权之类的信息。就遇到必须对齐 3 列底部的问题。在 table 布局中，我们用大表格嵌套小表格的方法，可以很方便对齐三列；而用 div 布局，三列独立分散，内容高低不同，就很难对齐。其实我们完全可以嵌套 div，把三列放进一个 DIV 中，就做到了底部对齐。下面是实现例子(白色背景框模拟一个页面)：

Body

这里是#header{ MARGIN: 0px; BORDER: 0px; BACKGROUND: #ccd2de; WIDTH: 580px; HEIGHT: 60px;}

这里是#mainbox { MARGIN: 0px; WIDTH: 580px; BACKGROUND: #FFF; }包含了#menu,#sidebar 和#content

这里是#menu{ FLOAT: right; MARGIN: 2px 0px 2px 0px; PADDING:0px 0px 0px 0px; WIDTH: 400px; BACKGROUND: #ccd2de; }

这里是#sidebar{ FLOAT: left; MARGIN: 2px 2px 0px 0px; PADDING: 0px; BACKGROUND: #F2F3F7; WIDTH: 170px; }，背景颜色用的是#main 的背景色

这里是#content{ FLOAT: right; MARGIN: 1px 0px 2px 0px; PADDING:0px; WIDTH: 400px; BACKGROUND: #E0EFDE;}

这里是主要内容，根据内容自动适应高度

这里是主要内容，根据内容自动适应高度

这里是主要内容，根据内容自动适应高度

这里是#footer{ CLEAR: both; MARGIN: 0px 0px 0px 0px; PADDING: 5px 0px 5px 0px; BACKGROUND: #ccd2de; HEIGHT: 40px; WIDTH: 580px; }。

这个例子的页面主要代码如下：

```
<div id="header"></div>
<div id="mainbox">
    <div id="menu"></div>
    <div id="sidebar"></div>
    <div id="content"></div>
</div>
<div id="footer"></div>
```

具体样式表都写在相应版块里了。重点在于#mainbox 层嵌套了#menu,#sidebar 和#content 三个层。当#content 的内容增加，#content 的高度就会增高，同时#mainbox 的高度也会撑开，#footer 层就自动下移。这样就实现了高度的自适应。

另外值得注意的是：#menu 和#content 都是浮动在页面右面"FLOAT: right;",#sidebar 是浮动在#menu 层的左面"FLOAT: left;"，这是浮动法定位，还可以采用绝对定位来实现这样的效果。

这个方法存在另一个问题，就是侧列#sidebar 的背景无法百分之百。一般的解决办法就是用 body 的背景色来填充满。(不能使用#mainbox 的背景色，因为在 Mozilla 等浏览器中#mainbox 的背景色失效。)

好了，主要的框架已经搭建完毕,剩下的工作只是往里面添砖加瓦。如果你希望尝试其他布局，推荐看看以下文章:

- CSS 布局 16 例
- onestab:三栏复合布局演示
- onestab:自由伸展的三栏式版面

Tips:[onestab 的"P.I.E"专题] 还有更多精彩介绍，推荐去看看。

# 网页设计 DIV+CSS——第 11 天:不用表格的菜单

布局初步搭建起来，我开始填充里面的内容。首先是定义 logo 图片：

样式表：#logo {MARGIN: 0px;padding:0px;WIDTH: 200px;HEIGHT:80px;}

页面代码: <a title="网页设计师" href="http://www.w3cn.org/" ><img height="80" alt="链接到 w3cn.org 首页" src="images/logo_w3cn_200x80.gif" width="200" /></a>

以上代码现在应该容易理解。先在 CSS 定义了一个 logo 的层，然后在页面中调用它。需要说明的是，为了使网页有更好的易用性，web 标准要求大家给所有的、属于正式内容的图片，加一个 alt 属性。这个 alt 属性是用来说明图片的作用(当图片不能显示的时候就显示替换文字)，所以不要只写成无意义的图片名称。

接下来是定义菜单。

## 1.不用表格的菜单(纵向)

我们先来看菜单的最终效果：

- **什么是网站标准**
- **使用标准的好处**
- **怎样过渡**
- **相关教程**
- **工具**
- **资源及链接**

通常方法我们至少嵌套 2 层表格来实现这样的菜单，间隔线采用在 td 中设置背景色并插入 1px 高的透明 GIF 图片实现；背景色的交替效果采用 td 的 onmouseover 事件实现。但查看本菜单的页面代码，你会看到只有如下几句：

```
<div id="menu">
<ul>
<li><a title="网站标准" href="http://www.w3cn.org/webstandards.html">什么是网站标准</a></li>
<li><a title="标准的好处" href="http://www.w3cn.org/benefits.html">使用标准的好处</a></li>
<li><a title="怎样过渡" href="http://www.w3cn.org/howto.html">怎样过渡</a></li>
<li><a title="相关教程" href="http://www.w3cn.org/tutorial.html">相关教程</a></li>
<li><a title="工具" href="http://www.w3cn.org/tools.html">工具</a></li>
<li><a title="资源及链接" href="http://www.w3cn.org/resources.html">资源及链接</a></li>
</ul>
</div>
```

没有用任何 table,而用的是无序列<li>，整个菜单的效果实现的秘密完全在于 id="menu"，我们再来看 CSS 中关于 menu 的定义：

(1)首先定义了 menu 层的主要样式:

```
#menu {
MARGIN: 15px 20px 0px 15px;  /*定义层的外边框距离*/
PADDING:15px;    /*定义层的内边框为 15px*/
BACKGROUND: #dfdfdf;   /*定义背景颜色*/
COLOR: #666;    /*定义字体颜色*/
BORDER:#fff 2px solid;  /*定义边框为 2px 白色线条*/
WIDTH:160px;    /*定义内容的宽度为 160px*/
}
```

(2)其次定义无序列表的样式：

```
#menu ul {
```

```
MARGIN: 0px;
PADDING: 0px;
BORDER: medium none; /*不显示边框*/
LINE-HEIGHT: normal;
LIST-STYLE-TYPE: none;

}
#menu li {BORDER-TOP: #FFF 1px solid; MARGIN: 0px;}
```

说明：这里用的是 id 选择器的派生方法定义(参考第 7 天：CSS 入门的介绍)了在 menu 层中的子元素<ul>和<li>的样式。LIST-STYLE-TYPE: none 一句表示不采用无序列表的默认样式，即：不显示小圆点（我们后面用自己的图标来代替小圆点）。BORDER-TOP: #FFF 1px solid;则定义了菜单之间的 1px 间隔线。

(3)定义 onmouseover 效果

```
#menu li a {
PADDING:5px 0px 5px 15px;
DISPLAY: block;
FONT-WEIGHT: bold;
BACKGROUND: url(images/icon_dot_lmenu.gif) transparent no-repeat 2px 8px;
WIDTH: 100%;
COLOR: #444;
TEXT-DECORATION: none;
}
#menu li a:hover { BACKGROUND: url(images/icon_dot_lmenu2.gif) #C61C18 no-repeat 2px 8px;
COLOR: #fff; }
```

解释如下：

- "display:block;"表示将标签 a 当作块级元素来显示，使得链接变成一个按钮；
- "BACKGROUND: url(images/icon_dot_lmenu.gif) transparent no-repeat 2px 8px;"这一句定义了替代 li 的小圆点的图标。"transparent"指背景为透明，"2px 8px"指定图标的位置是距左边 2px，距上边 8px。这一句也可以拆分写成四句：BACKGROUND-IMAGE: url(images/icon_dot_lmenu.gif); BACKGROUND-POSITION: 2px 8px; BACKGROUND-REPEAT: no-repeat; BACKGROUND-COLOR: transparent;"
- "#menu li a:hover"定义了当鼠标移动到链接上以后的颜色变化和小图标变化。

ok，不用表格的菜单就这样实现了。大家可以明显感觉到，原来写在 HTML 里的表现样式全部剥离放到 CSS 文件里去了。页面代码节约了大半。通过 CSS 要修改菜单样式就很简单了。

# 2.不用表格的菜单(横向)

上面是纵向的菜单,如果要显示横向菜单，用 li 也可以吗？当然是可以的，下面给出代码，效果就在本页顶部:

页面代码

```
<div id="submenu">
<ul>
<li id="one"><a title="首页" href="http://www.w3cn.org/">Home</a></li>
<li id="two"><a title="关于我们" href="http://www.w3cn.org/aboutus.html">关于我们</a></li>
<li id="three"><a title="网站标准" href="http://www.w3cn.org/webstandards.html">网站标准</a></li>
<li id="four"><a title="标准的好处" href="http://www.w3cn.org/benefits.html">标准的好处</a></li>
<li id="five"><a title="怎样过渡" href="http://www.w3cn.org/howto.html">怎样过渡</a></li>
<li id="six"><a title="相关教程" href="http://www.w3cn.org/tutorial.html">相关教程</a></li>
```

```
<li id="seven"><a title="工具" href="http://www.w3cn.org/tools.html">工具</a></li>
<li id="eight"><a title="资源及链接" href="http://www.w3cn.org/resources.html">资源及链接</a></li>
<li id="nine"><a title="常见问题" href="http://www.w3cn.org/faq.html">常见问题</a></li>
</ul>
</div>
```

样式表代码

```
#submenu {
MARGIN: 0px 8px 0px 8px;
PADDING: 4px 0px 0px 0px;
BORDER: #fff 1px solid;
BACKGROUND: #dfdfdf;
COLOR: #666;
HEIGHT:25px; }

#submenu ul {
CLEAR: left;
MARGIN: 0px;
PADDING:0px;
BORDER: 0px;
LIST-STYLE-TYPE: none;
TEXT-ALIGN: center;
DISPLAY:inline;
}

#submenu li {
FLOAT: left;
DISPLAY: block;
MARGIN: 0px;
PADDING: 0px;
TEXT-ALIGN: center}

#submenu li a {
DISPLAY: block;
PADDING:2px 3px 2px 3px;
BACKGROUND: url(images/icon_dot_lmenu.gif) transparent no-repeat 2px 8px;
FONT-WEIGHT: bold;
WIDTH: 100%;
COLOR: #444;
TEXT-DECORATION: none;
}

#submenu li a:hover {
BACKGROUND: url(images/icon_dot_lmenu2.gif) #C61C18 no-repeat 2px 8px;
COLOR: #fff; }
```

```
#submenu ul li#one A { WIDTH: 60px}
#submenu ul li#two A { WIDTH: 80px}
#submenu ul li#three A { WIDTH: 80px}
#submenu ul li#four A { WIDTH: 90px}
#submenu ul li#five A { WIDTH: 80px}
#submenu ul li#six A { WIDTH: 80px}
#submenu ul li#seven A { WIDTH: 60px}
#submenu ul li#eight A { WIDTH: 90px}
#submenu ul li#nine A { WIDTH: 80px}
```

以上代码不逐一分析了。横向菜单的关键在于:定义<li>样式时的"FLOAT: left;"语句。另外注意 UL 定义中的 DISPLAY:inline;一句表示将 li 强制作为内联对象呈递，从对象中删除行，通俗讲就是 li 不换行。实现横向排列。你也可以象例子中定义每个子菜单的宽度，控制菜单的间隔。好了，你也可以动手试试，用 li 实现各种各样的菜单样式。

Tips:如果你子菜单的宽度总和大于层的宽度，菜单会自动折行，利用这个原理可以实现单个无序列表的 2 列或者 3 列排版，这是原来 HTML 很难实现的。

感谢 zhuweiwei 指出横向菜单的 bug，本文 7 月 6 日修正。

# 网页设计 DIV+CSS——第 12 天:校验及常见错误

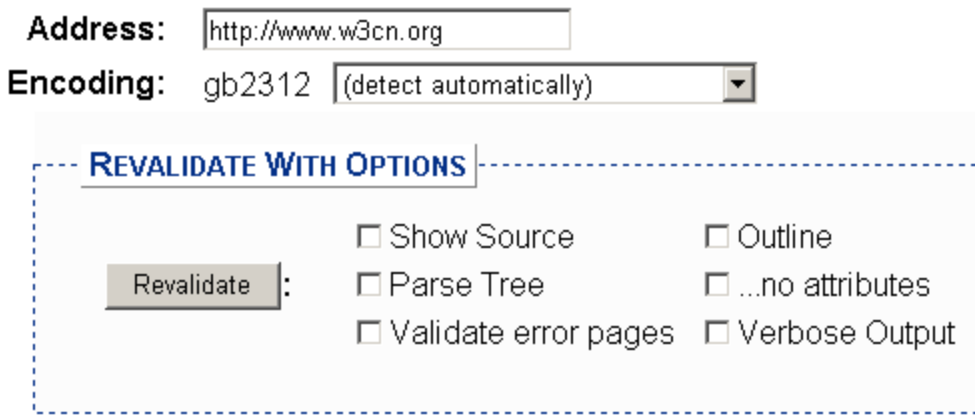
辛苦了好多天，我们努力学习使用 XHTML+CSS 来重新设计我们的网站。那么我们如何知道自己制作的页面真的符合 web 标准？W3C 和一些志愿者网站提供了在线校验程序，来帮助我们检查页面是否符合标准，并提供了修正错误的帮助信息。这些校验非常有用，是我调试页面第一步要做的事情。

## 1. XHTML 校验

- 校验网址：http://validator.w3.org/
- 校验方式:网址校验、文件上传校验

校验成功，会显示"This Page Is Valid XHTML 1.0 Transitional!",如图：


校验失败，会显示更多校验选项和错误信息，如图：




一般选择"Show Source"和"Verbose Output"可以帮助你找到错误代码所在行和错误原因。

## XHTML 校验常见错误原因对照表

- No DOCTYPE Found! Falling Back to HTML 4.01 Transitional--未定义 DOCTYPE。
- No Character Encoding Found! Falling back to UTF-8.--未定义语言编码。
- end tag for "img" omitted, but OMITTAG NO was specified--图片标签没有加"/"关闭。
- an attribute value specification must be an attribute value literal unless SHORTTAG YES is specified--属性值必须加引号。
- element "DIV" undefined---DIV 标签不能用大写，要改成小写 div。
- required attribute "alt" not specified---图片需要加 alt 属性。
- required attribute "type" not specified---JS 或者 CSS 调用的标签漏了 type 属性。

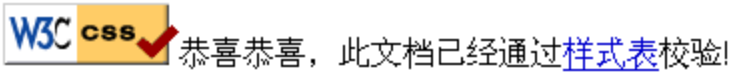
其中最最常见的错误就是标签的大小写问题了。通常这些错误都是关联的，比如忘记了一个</li>其他<li>标签都会报错，所以不要看到一堆的错误害怕，通常解决了一个错误，其他的错误也都没有了。如果你的页面通过 XHTML1.0 校验，可以在页面上放置这么一个图标：  代码如下：

```
<p> <a href="http://validator.w3.org/check/referer"><img src="http://www.w3.org/Icons/valid-xhtml10" alt="Valid XHTML 1.0!" height="31" width="88" /></a> </p>
```

## 2. CSS2 校验

- 校验网址：http://jigsaw.w3.org/css-validator/
- 校验方式:网址校验、文件上传校验、直接贴入代码校验

校验成功，会显示"恭喜恭喜，此文档已经通过样式表校验！",hoho,校验信息支持中文噢。如图：




校验失败，会显示两类错误：错误和警告。错误表示一定要修正，否则无法通过校验；警告表示有代码不被 W3C 推荐,建议修改。

## CSS2 校验常见错误原因对照表

- (错误)无效数字 : color909090 不是一个 color 值 : 909090 ---十六进制颜色值必须加"#"号，即#909090
- (错误)无效数字 : margin-topUnknown dimension : 6pixels ---pixels 不是一个单位值，正确写法 6px
- (错误)属性 scrollbar-face-color 不存在 : #eeeeee --- 定义滚动条颜色是非标准的属性
- (错误)值 cursorhand 不存在 : hand 是非标准属性值，修改为 cursor:pointer
- (警告)Line : 0 font-family: 建议你指定一个种类族科作为最后的选择 --W3C 建议字体定义的时候，最后以一个类别的字体结束，例如"sans-serif"，以保证在不同操作系统下，网页字体都能被显示。
- (警告)Line : 0 can't find the warning message for otherprofile --表示在代码中有非标准属性或值，校验程序无法判断和提供相应的警告信息。

同样，通过检验后，可以放置一个 CSS 校验通过图标，代码如下：

```
<p> <a href="http://jigsaw.w3.org/css-validator/"> <img style="border:0;width:88px;height:31px" src="http://jigsaw.w3.org/css-validator/images/vcss" alt="Valid CSS!" /> </a> </p>
```

# CSS 的十八般技巧

最近,经常有朋友问我一些工作中遇到的 CSS 问题。他们总是不能很好的控制 CSS，影响 CSS 的效率发挥。我来分析总结一下错误所在，帮助大家更加容易使用 CSS。

本文总结了我开始使用 CSS 布局方法以来所有的技巧和兼容方案，我愿意把这些与你分享，我会重点解释一些新手容易犯的错误(包括我自己也犯过的)，如果你已经是 CSS 高手，这些经验技巧可能已经都知道，如果你有更多的，希望可以帮我补充。

# 一. 使用 css 缩写

使用缩写可以帮助减少你 CSS 文件的大小，更加容易阅读。css 缩写的主要规则请参看《常用 css 缩写语法总结》，这里就不展开描述。

# 二. 明确定义单位，除非值为 0

忘记定义尺寸的单位是 CSS 新手普遍的错误。在 HTML 中你可以只写 width="100"，但是在 CSS 中，你必须给一个准确的单位，比如：width:100px width:100em。只有两个例外情况可以不定义单位：行高和 0 值。除此以外，其他值都必须紧跟单位，注意，不要在数值和单位之间加空格。

# 三. 区分大小写

当在 XHTML 中使用 CSS，CSS 里定义的元素名称是区分大小写的。为了避免这种错误，我建议所有的定义名称都采用小写。class 和 id 的值在 HTML 和 XHTML 中也是区分大小写的，如果你一定要大小写混合写，请仔细确认你在 CSS 的定义和 XHTML 里的标签是一致的。

# 四. 取消 class 和 id 前的元素限定

当你写给一个元素定义 class 或者 id，你可以省略前面的元素限定，因为 ID 在一个页面里是唯一的，而 clas s 可以在页面中多次使用。你限定某个元素毫无意义。例如：

```
div#content { /* declarations */ }
fieldset.details { /* declarations */ }
```

可以写成

```
#content { /* declarations */ }
.details { /* declarations */ }
```

这样可以节省一些字节。

# 五. 默认值

通常 padding 的默认值为 0，background-color 的默认值是 transparent。但是在不同的浏览器默认值可能不同。如果怕有冲突，可以在样式表一开始就先定义所有元素的 margin 和 padding 值都为 0，象这样：

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

# 六. 不需要重复定义可继承的值

CSS 中，子元素自动继承父元素的属性值，象颜色、字体等，已经在父元素中定义过的，在子元素中可以直接继承，不需要重复定义。但是要注意，浏览器可能用一些默认值覆盖你的定义。

# 七. 最近优先原则

如果对同一个元素的定义有多种，以最接近(最小一级)的定义为最优先，例如有这么一段代码

Update: Lorem ipsum dolor set

在 CSS 文件中，你已经定义了元素 p，又定义了一个 class"update"

```
p {
```

```
margin:1em 0;
font-size:1em;
color:#333;
}
.update {
font-weight:bold;
color:#600;
}
```

这两个定义中,class="update"将被使用,因为 class 比 p 更近。你可以查阅 W3C 的《 Calculating a selector’ s specificity》了解更多。

# 八.多重 class 定义

一个标签可以同时定义多个 class。例如：我们先定义两个样式，第一个样式背景为#666；第二个样式有 10 px 的边框。

```
.one{width:200px;background:#666;}
.two{border:10px solid #F00;}
```

在页面代码中，我们可以这样调用

```
<div class="one two"></div>
```

这样最终的显示效果是这个 div 既有#666 的背景，也有 10px 的边框。是的，这样做是可以的，你可以尝试一下。

# 九.使用子选择器(descendant selectors)

CSS 初学者不知道使用子选择器是影响他们效率的原因之一。子选择器可以帮助你节约大量的 class 定义。我们来看下面这段代码：

```
<div id="subnav">
<ul>
<li class="subnavitem"> <a href="#" class="subnavitem">Item 1</a></li>>
<li class="subnavitemselected"> <a href="#" class="subnavitemselected"> Item 1</a> </li>
<li class="subnavitem"> <a href="#" class="subnavitem"> Item 1</a> </li>
</ul>
</div>
```

这段代码的 CSS 定义是：

```
div#subnav ul { /* Some styling */ }
div#subnav ul li.subnavitem { /* Some styling */ }
div#subnav ul li.subnavitem a.subnavitem { /* Some styling */ }
div#subnav ul li.subnavitemselected { /* Some styling */ }
div#subnav ul li.subnavitemselected a.subnavitemselected { /* Some styling */ }
```

你可以用下面的方法替代上面的代码

```
<ul id="subnav">
<li> <a href="#"> Item 1</a> </li>
<li class="sel"> <a href="#"> Item 1</a> </li>
<li> <a href="#"> Item 1</a> </li>
</ul>
```

样式定义是：

```
#subnav { /* Some styling */ }
#subnav li { /* Some styling */ }
```

```
#subnav a { /* Some styling */ }
#subnav .sel { /* Some styling */ }
#subnav .sel a { /* Some styling */ }
```

用子选择器可以使你的代码和 CSS 更加简洁、更加容易阅读。

# 十.不需要给背景图片路径加引号

为了节省字节，我建议不要给背景图片路径加引号，因为引号不是必须的。例如：

```
background:url("images/***.gif") #333;
```

可以写为

```
background:url(images/***.gif) #333;
```

如果你加了引号，反而会引起一些浏览器的错误。

# 十一.组选择器(Group selectors)

当一些元素类型、class 或者 id 都有共同的一些属性，你就可以使用组选择器来避免多次的重复定义。这可以节省不少字节。例如：定义所有标题的字体、颜色和 margin，你可以这样写：

```
h1,h2,h3,h4,h5,h6 {
font-family:"Lucida Grande",Lucida,Arial,Helvetica,sans-serif;
color:#333;
margin:1em 0;
}
```

如果在使用时，有个别元素需要定义独立样式，你可以再加上新的定义，可以覆盖老的定义，例如：

```
h1 { font-size:2em; }
h2 { font-size:1.6em; }
```

# 十二.用正确的顺序指定链接的样式

当你用 CSS 来定义链接的多个状态样式时，要注意它们书写的顺序，正确的顺序是：:link :visited :hover :active。抽取第一个字母是"LVHA"，你可以记忆成"LoVe HAte"(喜欢讨厌)。为什么这么定义，可以参考 Eric Meyer 的《Link Specificity》。如果你的用户需要用键盘来控制，需要知道当前链接的焦点，你还可以定义:focus 属性。:focus 属性的效果也取决与你书写的位置，如果你希望聚焦元素显示:hover 效果，你就把:focus 写在:hover 前面；如果你希望聚焦效果替代:hover 效果，你就把:focus 放在:hover 后面。

# 十三.清除浮动

一个非常常见的 CSS 问题，定位使用浮动的时候，下面的层被浮动的层所覆盖，或者层里嵌套的子层超出了外层的范围。通常的解决办法是在浮动层后面添加一个额外元素，例如一个 div 或者一个 br，并且定义它的样式为 clear: both。这个办法有一点牵强，幸运的是还有一个好办法可以解决，参看这篇文章《How To Clear Floats Without Structural Markup》(注：本站将尽快翻译此文)。

上面 2 种方法可以很好解决浮动超出的问题，但是如果当你真的需要对层或者层里的对象进行 clear 的时候怎么办？一种简单的方法就是用 overflow 属性，这个方法最初的发表在《Simple Clearing of Floats》，又在《Clearance》和《Super simple clearing floats》中被广泛讨论。

上面那一种 clear 方法更适合你，要看具体的情况，这里不再展开论述。另外关于 float 的应用，一些优秀的文章已经说得很清楚，推荐你阅读：《Floatutorial》、《Containing Floats》和《Float Layouts》

# 十四.横向居中(centering)

这是一个简单的技巧，但是值得再说一遍，因为我看见太多的新手问题都是问这个：CSS 如何横向居中？你需要定义元素的宽，并且定义横向的 margin，如果你的布局包含在一个层(容器)中，就象这样：

你可以这样定义使它横向居中：

```
#wrap {
width:760px; /* 修改为你的层的宽度 */
margin:0 auto;
}
```

但是 IE5/Win 不能正确显示这个定义，我们采用一个非常有用的技巧来解决：用 text-align 属性。就象这样：

```
body {
text-align:center;
}
#wrap {
width:760px; /* 修改为你的层的宽度 */
margin:0 auto;
text-align:left;
}
```

第一个 body 的 text-align:center; 规则定义 IE5/Win 中 body 的所有元素居中(其他浏览器只是将文字居中)，第二个 text-align:left;是将#warp 中的文字居左。

# 十五.导入(Import)和隐藏 CSS

因为老版本浏览器不支持 CSS，一个通常的做法是使用@import 技巧来把 CSS 隐藏起来。例如：

```
@import url("main.css");
```

然而，这个方法对 IE4 不起作用，这让我很是头疼了一阵子。后来我用这样的写法：

```
@import "main.css";
```

这样就可以在 IE4 中也隐藏 CSS 了，呵呵，还节省了 5 个字节呢。想了解@import 语法的详细说明，可以看这里《centricle's css filter chart》

# 十六.针对 IE 的优化

有些时候，你需要对 IE 浏览器的 bug 定义一些特别的规则，这里有太多的 CSS 技巧(hacks)，我只使用其中的两种方法，不管微软在即将发布的 IE7 beta 版里是否更好的支持 CSS，这两种方法都是最安全的。

- 1.注释的方法
  - (a)在 IE 中隐藏一个 CSS 定义，你可以使用子选择器(child selector):

    ```
    html>body p {
    /* 定义内容 */
    }
    ```

  - (b)下面这个写法只有 IE 浏览器可以理解(对其他浏览器都隐藏)

    ```
    * html p {
    /* declarations */
    }
    ```

  - (c)还有些时候，你希望 IE/Win 有效而 IE/Mac 隐藏，你可以使用"反斜线"技巧：

    ```
    /* \*/
    * html p {
    ```

```
declarations
}
/* */
```

- 2.条件注释(conditional comments)的方法
  另外一种方法，我认为比 CSS Hacks 更加经得起考验就是采用微软的私有属性条件注释(conditional comments)。用这个方法你可以给 IE 单独定义一些样式，而不影响主样式表的定义。就象这样：

```
<!--[if IE]>
<link rel="stylesheet" type="text/css" href="ie.css" />
<![endif]-->
```

# 十七.调试技巧：层有多大?

当调试 CSS 发生错误，你就要象排版工人，逐行分析 CSS 代码。我通常在出问题的层上定义一个背景颜色，这样就能很明显看到层占据多大空间。有些人建议用 border，一般情况也是可以的，但问题是，有时候 border 会增加元素的尺寸，border-top 和 boeder-bottom 会破坏纵向 margin 的值，所以使用 background 更加安全些。
另外一个经常出问题的属性是 outline。outline 看起来象 boeder，但不会影响元素的尺寸或者位置。只有少数浏览器支持 outline 属性，我所知道的只有 Safari、OmniWeb、和 Opera。

# 十八.CSS 代码书写样式

在写 CSS 代码的时候，对于缩进、断行、空格，每个人有每个人的书写习惯。在经过不断实践后，我决定采用下面这样的书写样式：

```
selector1,
selector2 {
property:value;
}
```

当使用联合定义时，我通常将每个选择器单独写一行，这样方便在 CSS 文件中找到它们。在最后一个选择器和大括号{之间加一个空格，每个定义也单独写一行，分号直接在属性值后，不要加空格。
我习惯在每个属性值后面都加分号，虽然规则上允许最后一个属性值后面可以不写分号，但是如果你要加新样式时容易忘记补上分号而产生错误，所以还是都加比较好。
最后，关闭的大括号}单独写一行。

# WEB 打印实例教程

做 Web 开发的人员一定都会面临一个共同的难题，那就是打印。的确，相对于 Windows 桌面应用程序来讲，Web 应用程序的打印有种种限制，技术人员在项目开发过程中经常会遇到用户这样或那样的需求. 做过桌面应用开发的人都会非常熟悉水晶报表、Active Report 之类的报表控件，它们不仅有简单灵活的设计界面，更具有非常强大的报表功能，能满足各种报表的打印需求。而 Web 应用则因为其特殊的呈现方式，只能寻求其他的解决方案。现在我们来分析一下目前已经成形的 Web 打印方案：

现有的 Web 打印控制技术分成几种方案：

一. 自定义控件完成打印

利用 IE 自带的 WebBrowser 控件实现打印

利用第三方控件实现打印

1、自定义控件方式

自定义控件方式就是利用 VB 或 VC 等工具生成 COM 组件，用定义好的打印格式来分析打印源文件从而实现打印。只有将生成的组件下载并注册到客户机上，才能实现在客户端的

打印。

难点主要是定义打印格式、如何来分析打印源文件。现有的比较好的方法是利用 XML 技术来全面的解决问题，利用 XML 可以非常容易地定义打印目标的文本、表格等内容的格式。

但对程序员的开发要求高，难度比较大。

2、利用 WebBrowser 实现 Web 打印

WebBrowser 是 IE 内置的浏览器控件，无需用户下载。本文档所讨论的是有关 IE6.0 版本 的 WebBrowser 控件技术内容。与其相关的技术要求有：打印文档的生成、页面设置、打印操作的实现等几个环节。

（一）、打印文档的生成

1、客户端脚本方式

客户端脚本分为 VBScript、JavaScript、JScript 几种脚本语言。在 IE 下开发应用使用的语法为 JScript 的语法，由于它和 JavaScript 几乎没有什么区别，所以也可以称其为 JavaScript（下面简写为 JS）。一般情况下，主要使用 JS 来实现 DOM 文档的分析，DOM 为微软提出的一种 Web 文档模型，主要用来实现 Web 脚本编程。

利用 JS 可以分析源页面的内容，将欲打印的页面元素提取出来，实现打印。通过分析源文档的内容，可以生成打印目标文档。

优点：客户端独立完成打印目标文档的生成，减轻服务器负荷；

缺点：源文档的分析操作复杂，并且源文档中的打印内容要有约定；

2、服务器端程序方式

服务器端程序方式，主要是利用后台代码从数据库中读取打印源，生成打印目标文档。当的页面生成时，还应适当考虑使用 CSS 来实现强制分页控制。

优点：可以生成内容非常的丰富的打印目标文档，目标文档的内容的可控性强。由于打印内容是从数据库中获取的，所以生成操作相对简单；

缺点：服务器端负载比较大；

（二）、页面设置

页面设置主要是指设置打印文档的页边距、页眉、页脚、纸张等内容。页面设置将直接影响到打印文档版面的生成效果，所以它和打印文档的生成有着密切的关系。比如：表格的

行数、大小、位置、字体的大小等。

现有的技术是利用 IE6.0 内置的打印模板方式来控制页面设置，其可以对打印目标文档产生非常大的影响。打印模板可以控制页边距、页眉、页脚、奇偶页等内容，并可以将用户的设置取得，还可以将设置发送到服务器端。

打印模板技术可以自定预览窗口和打印格式，最大限度地影响目标文档和打印效果。

（三）、打印操作的实现

此功能的实现主要是利用 WebBrowser 控件的函数接口来实现打印、打印预览（默认的）、

页面设置（默认的）。

```
<object ID='WebBrowser1' WIDTH=0 HEIGHT=0
CLASSID='CLSID:8856F961-340A-11D0-A96B-00C04FD705A2'>
//打印
WebBrowser1.ExecWB(6,1);
//打印设置
WebBrowser1.ExecWB(8,1);
//打印预览
WebBrowser1.ExecWB(7,1);
```

3、一个实例项目采用的打印方案

服务器端程序方式、打印预览接口调用，下面为例， 主要参考项目中的：

pageErrorPrint.aspx.vb 文件

主调用页

```
function PrintPage(iPageIndex,strQuery)
{
var strURL;
strURL = "PageErrorPrint.aspx?PageIndex=" + iPageIndex + "&QueryString=" +
strQuery;
winPrint=window.open(strURL,"","left=2000,top=2000,fullscreen=3");
}
```

打印页 HTML 源中的预览控制

```
<SCRIPT language="javascript">
document.write("<object ID='WebBrowser' WIDTH=0 HEIGHT=0
CLASSID='CLSID:8856F961-340A-11D0-A96B-00C04FD705A2'></object>");
WebBrowser.ExecWB(7,1);
window.opener=null;
window.close();
</SCRIPT>
```

程序头

```
'首先声明表格容器
Protected WithEvents phContainer As System.Web.UI.WebControls.PlaceHolder
'每个表格中的记录数量
Private Const ItemPerTable As Integer = 20
```

关键的实现部分

```
'创建一个符合打印要求的表格
tabPagePrint = NewPrintTable()
'将表头添加到此表格中
Call AddTableTitle(tabPagePrint)
'初始化记录器
i = 0
iItemIndex = iStartPoint
For Each clsItem In clsAllData.ErrorCollection
If i > 0 And i Mod ItemPerTable = 0 Then
'添加表格控件到页面中
phContainer.Controls.Add(tabPagePrint)
'在页面中添加一个换行符
Call AddPageBreak()
'创建新一轮的表格
tabPagePrint = NewPrintTable()
Call AddTableTitle(tabPagePrint)
End If
'将记录添加到表格中
Call AddItemToTable(iItemIndex, tabPagePrint, clsItem)
iItemIndex = iItemIndex + 1
i = i + 1
Next
'添加表格控件到页面中
```

```
phContainer.Controls.Add(tabPagePrint)
支持函数
'功能:添加页的换行符
Private Sub AddPageBreak()
Dim ltBreak As LiteralControl
ltBreak = New LiteralControl("<p style='page-break-before:always'>")
phContainer.Controls.Add(ltBreak)
End Sub
```

二、利用 IE 自身打印

这种方式比较简单，也常用的打印方式，只需要将报表页面设计好，用户通过 IE 菜单中的打印功能完成打印。优点是简单，容易实现，缺点是不灵活，不能控制分页，不能控制好页眉和页脚。

三、将报表导出成 Word,Excel 或 PDF 形式打印

这种方式需要将页面导出成 Office 文档或 pdf,最低的要求是客户端已经安装用以打开 Word、Excel 或 Pdf 文档的软件。这种方式可以通过水晶报表组件或其他一些第三方控件非常容易地实现。导出成 Pdf 形式后打印质量和效果都很好，导出成 Word 或 Excel 后用户可以自定义打印的内容和格式。

总之，现有的打印方案各有所长，在开发过程中应根据用户的需求作选择，利用 IE 打印简单，容易实现，在用户需求简单或打印内容较少的情况下采用此方案比较适宜。利用自定义控件打印可以实现完全自定义，但需要较高的技术要求和开发周期。利用导出的方式则可以满足用户需要一点自定义或打印内容有多页的需求。

---

1、控制"纵打"、横打”和“页面的边距。

（1）<script defer>

```
function SetPrintSettings() {
  // -- advanced features
  factory.printing.SetMarginMeasure(2) // measure margins in inches
  factory.SetPageRange(false, 1, 3) // need pages from 1 to 3
  factory.printing.printer = "HP DeskJet 870C"
  factory.printing.copies = 2
  factory.printing.collate = true
  factory.printing.paperSize = "A4"
  factory.printing.paperSource = "Manual feed"
  // -- basic features
  factory.printing.header = "This is MeadCo"
  factory.printing.footer = "Advanced Printing by ScriptX"
  factory.printing.portrait = false
  factory.printing.leftMargin = 1.0
  factory.printing.topMargin = 1.0
  factory.printing.rightMargin = 1.0
  factory.printing.bottomMargin = 1.0
}
</script>
```

（2）

```
<script language="javascript">
    function printsetup(){
    // 打印页面设置
```

```
    wb.execwb(8,1);
    }
    function printpreview(){
    // 打印页面预览

    wb.execwb(7,1);



    }
    function printit()
    {
    if (confirm('确定打印吗?')) {
    wb.execwb(6,6)
    }
    }
    </script>
</head>
<body>
<OBJECT classid="CLSID:8856F961-340A-11D0-A96B-00C04FD705A2"
height=0 id=wb name=wb width=0></OBJECT>
<input type=button name=button_print value="打印"
onclick="javascript:printit()">
<input type=button  name=button_setup value="打印页面设置"
onclick="javascript:printsetup();">
<input type=button  name=button_show value="打印预览"
onclick="javascript:printpreview();">
<input type=button name=button_fh value="关闭"
onclick="javascript:window.close();">
```

---

关于这个组件还有其他的用法，列举如下：

WebBrowser.ExecWB(1,1) 打开

Web.ExecWB(2,1) 关闭现在所有的 IE 窗口，并打开一个新窗口

Web.ExecWB(4,1) 保存网页

Web.ExecWB(6,1) 打印

Web.ExecWB(7,1) 打印预览

Web.ExecWB(8,1) 打印页面设置

Web.ExecWB(10,1) 查看页面属性

Web.ExecWB(15,1) 好像是撤销，有待确认

Web.ExecWB(17,1) 全选

Web.ExecWB(22,1) 刷新

Web.ExecWB(45,1) 关闭窗体无提示

2、分页打印

```
<HTML>
<HEAD>
```

```
<STYLE>
    P {page-break-after: always}
</STYLE>
</HEAD>
<BODY>
<%while not rs.eof%>
<P><%=rs(0)%></P>
<%rs.movenext%>
<%wend%>
</BODY>
</HTML>
```

3、ASP 页面打印时如何去掉页面底部的路径和顶端的页码编号

（1）ie 的文件-〉页面设置-〉讲里面的页眉和页脚里面的东西都去掉，打印就不出来了。

（2）

```
<HTML>
<HEAD>
<TITLE> New Document </TITLE>
<META NAME="Generator" CONTENT="EditPlus">
<META NAME="Author" CONTENT="YC">
<script language="VBScript">
dim hkey_root,hkey_path,hkey_key
hkey_root="HKEY_CURRENT_USER"
hkey_path="\Software\Microsoft\Internet Explorer\PageSetup"
'//设置网页打印的页眉页脚为空
function pagesetup_null()
    on error resume next
    Set RegWsh = CreateObject("WScript.Shell")
    hkey_key="\header"
    RegWsh.RegWrite hkey_root+hkey_path+hkey_key,""
    hkey_key="\footer"
    RegWsh.RegWrite hkey_root+hkey_path+hkey_key,""
end function
'//设置网页打印的页眉页脚为默认值
function pagesetup_default()
    on error resume next
    Set RegWsh = CreateObject("WScript.Shell")
    hkey_key="\header"
    RegWsh.RegWrite hkey_root+hkey_path+hkey_key,"&w&b 页码, &p/&P"
    hkey_key="\footer"
    RegWsh.RegWrite hkey_root+hkey_path+hkey_key,"&u&b&d"
end function
</script>
</HEAD>
<BODY>
<br/>
```

```
<br/>
<br/>
<br/>
<br/>
<br/><p align=center>
<input type="button" value="清空页码" onclick=pagesetup_null()> <input type="button" value="恢复页吗"
onclick=pagesetup_default()><br/>
</p>
</BODY>
</HTML>
```

4、浮动帧打印

```
<SCRIPT LANGUAGE=javascript>
function button1_onclick() {
    var odoc=window.iframe1.document;
    var r=odoc.body.createTextRange();
    var stxt=r.htmlText;
    alert(stxt)
    var pwin=window.open("","print");
    pwin.document.write(stxt);
    pwin.print();
}
</SCRIPT>
```

4、用 FileSystem 组件实现 WEB 应用中的本地特定打印

```
<script Language=VBScript>
function print_onclick //打印函数
dim label
label=document.printinfo.label.value //获得 HTML 页面的数据
set objfs=CreateObject("Scripting.FileSystemObject") //创建 FileSystem 组件对象的实例
set objprinter=objfs.CreateTextFile ("LPT1:",true) //建立与打印机的连接
objprinter.Writeline("_____") //输出打印的内容
objprinter.Writeline("| |")
objprinter.Writeline("| 您打印的数据是："&label& " |")
objprinter.Writeline("| |")
objprinter.Writeline("|_____|")
objprinter.close //断开与打印机的连接
set objprinter=nothing
set objfs=nothing // 关闭 FileSystem 组件对象
end function
</script>
```

服务器端脚本：

```
<%……
set conn=server.CreateObject ("adodb.connection")
conn.Open "DSN=name;UID=XXXX;PWD=XXXX;"
set rs=server.CreateObject("adodb.recordset")
```

```
rs.Open(“select ……”),conn,1,1
…….%> //与数据库进行交互
HTML 页面编码：
<HTML>
……
<FORM ID=printinfo NAME="printinfo" >
<INPUT type="button" value="打印>>" id=print name=print > //调用打印函数
<INPUT type=hidden id=text1 name=label value=<%=……%>> //保存服务器端传来的数据
……
</HTML>
```

## Div+CSS 布局入门教程

在网页制作中，有许多的术语，例如：CSS、HTML、DHTML、XHTML 等等。在下面的文章中我们将会用到一些有关于 HTML 的基本知识，而在你学习这篇入门教程之前，请确定你已经具有了一定的 HTML 基础。下面我们就开始一步一步使用 DIV+CSS 进行网页布局设计吧。

所有的设计第一步就是构思，构思好了，一般来说还需要用 PhotoShop 或 FireWorks(以下简称 PS 或 FW)等图片处理软件将需要制作的界面布局简单的构画出来，以下是我构思好的界面布局图。

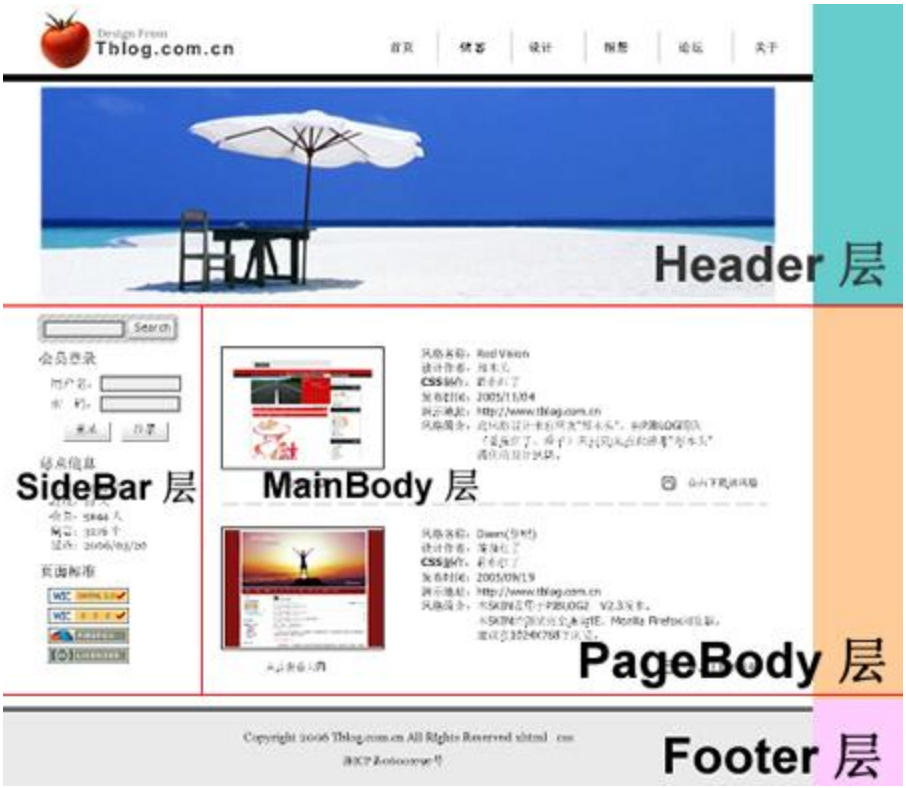
下面，我们需要根据构思图来规划一下页面的布局，仔细分析一下该图，我们不难发现，图片大致分为以下几个部分：

1、顶部部分，其中又包括了 LOGO、MENU 和一幅 Banner 图片；

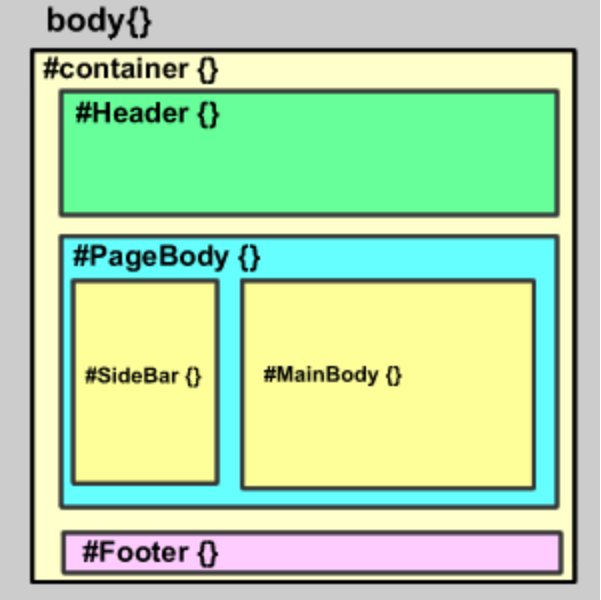
2、内容部分又可分为侧边栏、主体内容；

3、底部，包括一些版权信息。

有了以上的分析，我们就可以很容易的布局了，我们设计层如下图:




根据上图，我再画了一个实际的页面布局图，说明一下层的嵌套关系，这样理解起来就会更简单了。




DIV 结构如下：

```
|body {}  /*这是一个 HTML 元素，具体我就不说明了*/
└#Container {}  /*页面层容器*/
    ├#Header {}  /*页面头部*/
    ├#PageBody {}  /*页面主体*/
    |  ├#Sidebar {}  /*侧边栏*/
    |  └#MainBody {}  /*主体内容*/
    └#Footer {}  /*页面底部*/
```

至此，页面布局与规划已经完成，接下来我们要做的就是开始书写 HTML 代码和 CSS。

接下来我们在桌面新建一个文件夹，命名为“DIV+CSS 布局练习”，在文件夹下新建两个空的记事本文档，输入以下内容：

```
<!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>无标题文档</title>
<link href="css.css" rel="stylesheet" type="text/css" />
</head>

<body>
</body>
</html>
```

这是 XHTML 的基本结构，将其命名为 index.htm，另一个记事本文档则命名为 css.css。

下面，我们在<body></body>标签对中写入 DIV 的基本结构，代码如下：

```
<div id="container">[color=#aaaaaa]<!--页面层容器-->[/color]
    <div id="Header">[color=#aaaaaa]<!--页面头部-->[/color]
    </div>
    <div id="PageBody">[color=#aaaaaa]<!--页面主体-->[/color]
        <div id="Sidebar">[color=#aaaaaa]<!--侧边栏-->[/color]
        </div>
        <div id="MainBody">[color=#aaaaaa]<!--主体内容-->[/color]
        </div>
    </div>
    <div id="Footer">[color=#aaaaaa]<!--页面底部-->[/color]
    </div>
</div>
```

为了使以后阅读代码更简易，我们应该添加相关注释，接下来打开 css.css 文件，写入 CSS 信息，代码如下：

```
/*基本信息*/
body {font:12px Tahoma;margin:0px;text-align:center;background:#FFF;}

/*页面层容器*/
#container {width:100%}

/*页面头部*/
#Header {width:800px;margin:0 auto;height:100px;background:#FFCC99}

/*页面主体*/
#PageBody {width:800px;margin:0 auto;height:400px;background:#CCFF00}

/*页面底部*/
#Footer {width:800px;margin:0 auto;height:50px;background:#00FFFF}
```

把以上文件保存，用浏览器打开，这时我们已经可以看到基础结构了，这个就是页面的框架了。

关于以上 CSS 的说明（详细请参考 CSS2.0 中文手册，网上有下载）：

1、请养成良好的注释习惯，这是非常重要的；

2、body 是一个 HTML 元素，页面中所有的内容都应该写在这标签对之内，我就不多说了；

3、讲解一些常用的 CSS 代码的含义：

**font:12px Tahoma**；
这里使用了缩写，完整的代码应该是：font-size:12px;font-family:Tahoma；说明字体为 12 像素大小，字体为 Tahoma 格式；

**margin:0px**；

也使用了缩写，完整的应该是：

*margin-top:0px;margin-right:0px;margin-bottom:0px;margin-left:0px*

*或*

*margin:0px 0px 0px 0px*

顺序是 上 / 右 / 下 / 左，你也可以书写为 margin:0(缩写)；

以上样式说明 body 部分对上右下左边距为 0 像素，如果使用 auto 则是自动调整边距，

另外还有以下几种写法：

*margin:0px auto；*

说明上下边距为 0px，左右为自动调整；

我们以后将使用到的 padding 属性和 margin 有许多相似之处，他们的参数是一样的，

只不过各自表示的含义不相同，margin 是外部距离，而 padding 则是内部距离。

**text-align:center**

文字对齐方式，可以设置为左、右、中，这里我将它设置为居中对齐。

**background:#FFF**

设置背景色为白色，这里颜色使用了缩写，完整的应该是 background:#FFFFFF。

background 可以用来给指定的层填充背景色、背景图片，以后我们将用到如下格式：

*background:#ccc url('bg.gif') top left no-repeat；*

表示：使用#CCC(灰度色)填充整个层，使用 bg.gif 做为背景图片，

*top left*

表示图片位于当前层的左上端，no-repeat 表示仅显示图片大小而不填充满整个层。

*top/right/left/bottom/center*

用于定位背景图片，分别表示 上 / 右 / 下 / 左 / 中；还可以使用

*background:url('bg.gif') 20px 100px;*

表示 X 座标为 20 像素，Y 座标为 100 像素的精确定位；

*repeat/no-repeat/repeat-x/repeat-y*

分别表示 填充满整个层 / 不填充 / 沿 X 轴填充 / 沿 Y 轴填充。

**height / width / color**

分别表示高度(px)、宽度(px)、字体颜色(HTML 色系表)。

4、如何使页面居中?

大家将代码保存后可以看到，整个页面是居中显示的，那么究竟是什么原因使得页面居中显示呢？

是因为我们在#container 中使用了以下属性：

*margin:0 auto;*

按照前面的说明，可以知道，表示上下边距为 0，左右为自动，因此该层就会自动居中了。

如果要让页面居左，则取消掉 auto 值就可以了，因为默认就是居左显示的。

通过 margin:auto 我们就可以轻易地使层自动居中了。

5、这里我只介绍这些常用的 CSS 属性了，其他的请参看 CSS2.0 中文手册。

当我们写好了页面大致的 DIV 结构后，我们就可以开始细致地对每一个部分进行制作了。

在上一章中我们写入了一些样式，那些样式是为了预览结构而写入的，我们把 css.css 中的样式全部清除掉，重新写入以下样式代码：

```
/*基本信息*/

body {font:12px Tahoma;margin:0px;text-align:center;background:#FFF;}

a:link,a:visited {font-size:12px;text-decoration:none;}

a:hover{}

/*页面层容器*/

#container {width:800px;margin:10px auto}
```

**样式说明：**

*a:link,a:visited {font-size:12px;text-decoration:none;}*

*a:hover {}*

这两项分别是控制页面中超链接的样式，具体我就不说明了，请大家参阅手册。

*#container {width:800px;margin:10px auto}*

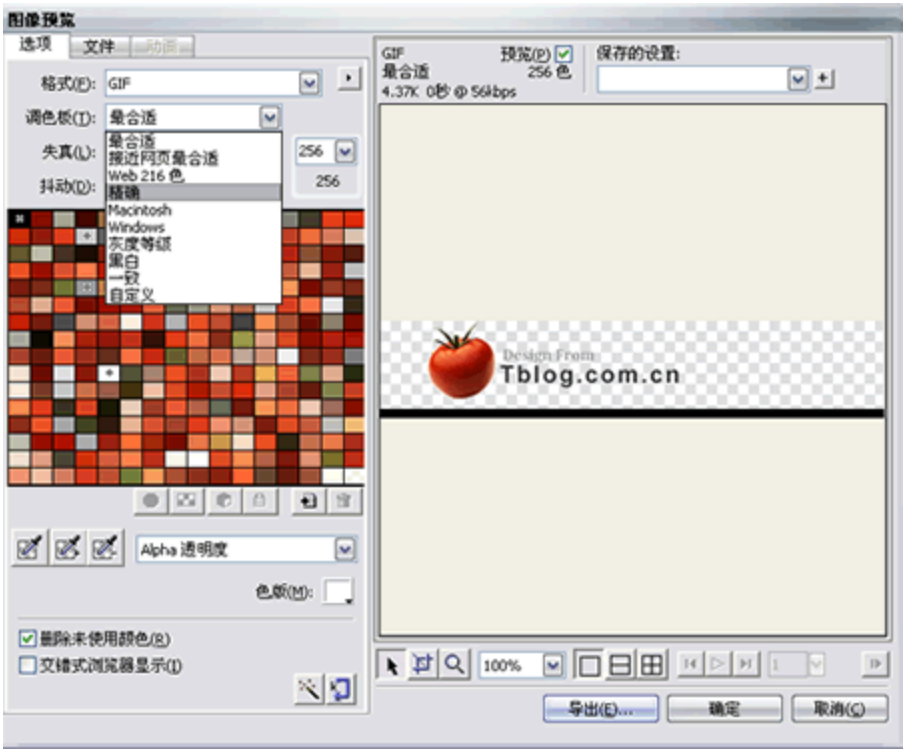
指定整个页面的显示区域。

*width:800px* 指定宽度为 800 像素，这里根据实际所需设定。

*margin:10px auto*，则是页面上、下边距为 10 个像素，并且居中显示。

上一章中我们讲过，对层的 margin 属性的左右边距设置为 auto 可以让层居中显示。

接下来，我们开始制作 TOP 部分，TOP 部分包括了 **LOGO、菜单和 Banner**，首先我们要做的就是对设计好的图片进行切片，以下是在 FW 下完成的切片：




我将 TOP 部分切片为两部分，第一部分包括了 LOGO 和一条横线。由于 LOGO 图片并没有太多的颜色，这里我于是将这一部分保存为 GIF 格式，调色板选择为精确，选择 Alpha 透明度，色版为白色(此处颜色应与背景色相同)，导出为 logo.gif，图像宽度为 800px。

到这里，有的朋友就说了，*** 为什么要使用 GIF 格式？使用 JPEG 不是更好吗？**

因为 GIF 格式的图片文件更小，这样能使页面载入的速度更快，当然使用此格式之前必须确定图片并没有使用太多的颜色，当我们使用了 GIF 格式时，从肉眼上并不能看出图片有什么太大的变化，因此这是可行的。

*** 接下来的 Banner 部分还能使用 GIF 格式吗？**

答案是不能，因为 Banner 部分是一个细致的图片，如果使用 GIF 格式颜色会有太大的损失，所以必须使用 JPEG 格式，将文件导出为 banner.jpg。

*** 合理的切片是非常之重要的，因为切片的方法正确与否决定了 CSS 书写的简易程度以及页面载入速度。**

切好片后，我们还需要对 TOP 部分进行分析并将 DIV 结构写入 Header 中代码如下：

```
<div id="menu">
<ul>
<li><a href="#">首页</a></li>
<li class="menuDiv"></li>
<li><a href="#">博客</a></li>
<li class="menuDiv"></li>
<li><a href="#">设计</a></li>
<li class="menuDiv"></li>
<li><a href="#">相册</a></li>
<li class="menuDiv"></li>
<li><a href="#">论坛</a></li>
<li class="menuDiv"></li>
<li><a href="#">关于</a></li>
</ul>
</div>
```

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

为什么要这么写呢，因为对菜单使用列表<li>形式，可以在以后方便对菜单定制样式。

而为什么要添加以下代码呢？

*<li class="menuDiv"></li>*

插入这一段代码是可以方便地对菜单选项之间插入一些分隔样式，例如预览图中的竖线分隔。

然后我们在 css.css 中再写入以下样式：

```
/*页面头部*/
#header {background:url(logo.gif) no-repeat}
```

**样式说明：**

*#header {background:url(logo.gif) no-repeat}*

给页面头部分加入一个背景图片 LOGO，并且不作填充。

**这里，我们没有指定 header 层的高度，为什么不指定呢？**

因为 header 层中还有菜单和 banner 项，所以层的高度暂时是未知的，而层的属性又可以让层根据内容自动设定调整，因此我们并不需要指定高度。

**使用列表<li>制作菜单**

开始此节的学习前，请确认你已经参照之前的几节内容写入了 DIV、CSS 到 index.htm 和 css.css 文件中。

这一节我将告诉大家如何用列表<li>来制作菜单。

```
<div id="menu">
  <ul>
    <li><a href="#">首页</a></li>
```

```
<li class="menuDiv"></li>
<li><a href="#">博客</a></li>
<li class="menuDiv"></li>
<li><a href="#">设计</a></li>
<li class="menuDiv"></li>
<li><a href="#">相册</a></li>
<li class="menuDiv"></li>
<li><a href="#">论坛</a></li>
<li class="menuDiv"></li>
<li><a href="#">关于</a></li>
</ul>
</div>
```

以上是这部分的结构，有关于**<ul></ul>**、**<li></li>**这两个 HTML 元素大家自己去参考相关的内容吧，它们最主要的作用就是在 HTML 中以列表的形式来显示一些信息。

还有一点需要大家一定要分清楚的，当在 HTML 中定义为 **id="divID**"时，在 CSS 对应的设置语法则是 **#divID{}** ，如果在 HTML 中定义为 **class="divID"**时，则在 CSS 中对应的设置语法是.divID。

如果 **id="divID"**这个层中包括了一个<img></img>，则这个 img 在 CSS 中对应的设置语法应该是**#divID img {}**，同样，如果是包含在 **class="divID"**这个层中时，则设置语法应该是.**divID img {}**，这一点希望大家要分清楚了。

另外，HTML 中的一切元素都是可以定义的，例如 table、tr、td、th、form、img、input...等等，如果你要在 CSS 中设置它们，则直接写入元素的名称加上一对大括号{}就可以了。所有的 CSS 代码都应该写在大括号**{}**中。

按照上面的介绍，我们先在 css.css 中写入以下代码：

```
#menu ul {list-style:none;margin:0px;}
```

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

**解释一下：**

*#menu ul {list-style:none;margin:0px;}*

list-style:none，这一句是取消列表前点，因为我们不需要这些点。

margin:0px，这一句是删除 UL 的缩进，这样做可以使所有的列表内容都不缩进。

*#menu ul li {float:left;}*

这里的 float:left 的左右是让内容都在同一行显示，因此使用了浮动属性(float)。

到这一步，建议大家先保存预览一下效果，我们再添加下面的内容，效果如下：

首页博客设计相册论坛关于

这时，列表内容是排列在一行，我们在#menu ul li {}再加入代码 **margin:0 10px**

```
#menu ul {list-style:none;margin:0px;}
```

```
#menu ul li {float:left;margin:0 10px}
```

margin:0 10px 的作用就是让列表内容之间产生一个 20 像素的距离(左：10px，右：10px)，预览的效果如下：

首页 博客 设计 相册 论坛 关于

现在，雏形已经出来了，我们再来固定菜单的位置，把代码改成如下：

```
#menu {padding:20px 20px 0 0}
```

```
/*利用 padding:20px 20px 0 0 来固定菜单位置*/
```

```
#menu ul {float:right;list-style:none;margin:0px;}
```

```
/*添加了 float:right 使得菜单位于页面右侧*/
```

```
#menu ul li {float:left;margin:0 10px}
```

这时，位置已经确定了，可是构思图中，菜单选项之间还有一条竖线，怎么办呢？

别忘了，我们早就已经留好了一个空的<li class="menuDiv"></li>，要想加入竖线就使用它了。

按照上面说的方法，我们再添加以下代码：

```
.menuDiv {width:1px;height:28px;background:#999}
```

保存预览一下，竖线是否已经出来了？关于这段代码就不多讲了，应该是很容易理解的。

首页 | 博客 | 设计 | 相册 | 论坛 | 关于

不过，菜单选项的文字却在顶部，我们再修改成以下代码：

```
#menu ul li {float:left;margin:0 10px;display:block;line-height:28px}
```

关于 **display:block;line-height:28px** 大家可以去参阅一下手册，我就不多讲了。

效果基本上已经实现了，剩下的就是修改菜单的超链接样式，在 css.css 中添加以下代码：

```
#menu ul li a:link,#menu ul li a:visited {font-weight:bold;color:#666}
```

```
#menu ul li a:hover{}
```

这个也不多说了，没什么好说的了，最后的效果如下：

首页 | 博客 | 设计 | 相册 | 论坛 | 关于

这一节到这里就完毕了

这一节里面，主要就是想告诉大家如何使用好 **border** 和 **clear** 这两个属性。

首先，如果你曾用过 table 制作网页，你就应该知道，如果要在表格中绘制一条虚线该如何做，那需要制作一个很小的图片来填充，其实我们还有更简单的办法，只要在<td></td>中加入这么一段就可以了，你可以试试：

* *

大家可以再次参考手册，然后你就能明白 dashed、solid、dotted...等的作用，利用它们你可以制作出许多效果来，实线、虚线、双线、阴影线等等。

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

以上代码便可以实现设计草图中的 banner，在 css.css 中加入以下样式：

```
#banner {
background:url(banner.jpg) 0 30px no-repeat; /*加入背景图片*/
width:730px; /*设定层的宽度*/
margin:auto; /*层居中*/
height:240px; /*设定高度*/
border-bottom:5px solid #EFEFEF; /*画一条浅灰色实线*/
clear:both /*清除浮动*/
}
```

通过 border 很容易就绘制出一条实线了，并且减少了图片下载所占用的网络资源，使得页面载入速度变得更快。

另一个要说明的就是 clear:both，表示清除左、右所有的浮动，在接下来的布局中我们还会用这个属性：clear:left/right。在这里添加 clear:both 是由于之前的 ul、li 元素设置了浮动，如果不清除则会影响 banner 层位置的设定。

```
<div id="pagebody"><!--页面主体-->
<div id="sidebar"><!--侧边栏-->
</div>
<div id="mainbody"><!--主体内容-->
</div>
</div>
```

以上是页面主体部分，我们在 css.css 中添加以下样式：

```
#pagebody {
width:730px; /*设定宽度*/
margin:8px auto; /*居中*/
}
#sidebar {
width:160px; /*设定宽度*/
text-align:left; /*文字左对齐*/
float:left; /*浮动居左*/
clear:left; /*不允许左侧存在浮动*/
overflow:hidden; /*超出宽度部分隐藏*/
}
#mainbody {
width:570px;
text-align:left;
```

```
float:right; /*浮动居右*/
clear:right; /*不允许右侧存在浮动*/
overflow:hidden
}
```

为了可以查看到效果，建议在#sidebar 和#mainbody 中加入以下代码，预览完成后可以删除这段代码：

```
border:1px solid #E00;
height:200px
```

保存预览效果，可以发现这两个层完美的浮动，在达到了我们布局的要求，而两个层的实际宽度应该 160+2(border)+570+2=734px，已经超出了父层的宽度，由于 clear 的原因，这两个层才不会出现错位的情况，这样可以使我们布局的页面不会因为内容太长（例如图片）而导致错位。

而之后添加的 overflow:hidden 则可以**使内容太长（例如图片）的部份自动被隐藏**。通常我们会看到一些网页在载入时，由于图片太大，导致布局被撑开，直到页面下载完成才恢复正常，通过添加 overflow:hidden 就可以解决这个问题。

CSS 中每一个属性运用得当，就可以解决许多问题，或许它们与你在布局的页并没有太大的关系，但是你必须知道这些属性的作用，在遇到难题的时候，可以尝试使用这些属性去解决问题。