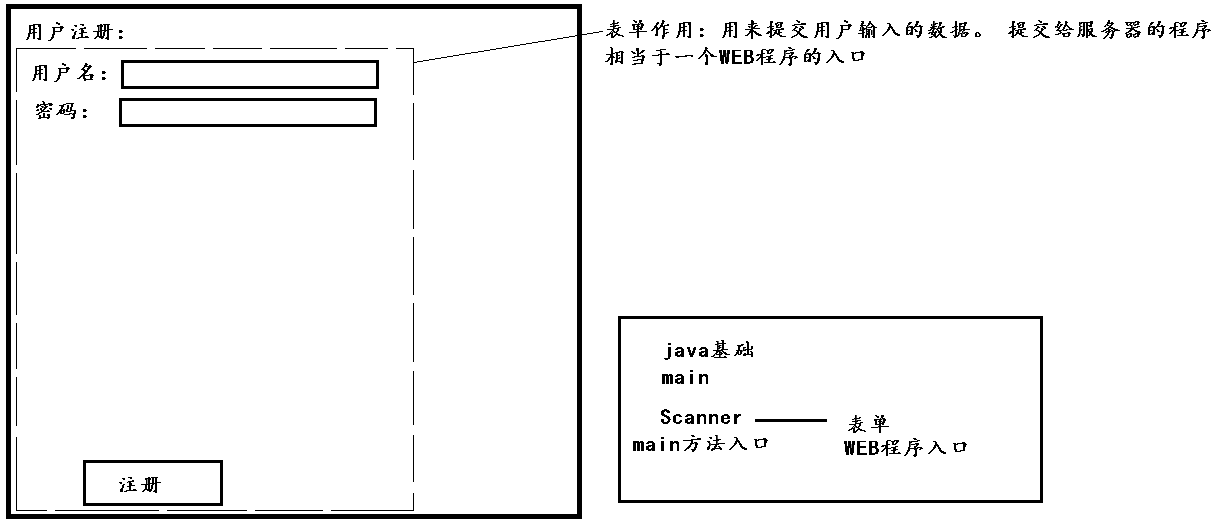
# HTML的表单标签

## 表单标签概述

### 什么是表单标签

我们去银行办理一些业务的时候，我们通常需要填写一些纸质单据，而如果我们在网页中需要填写一些单据呢？我们可以通过HTML的表单来实现。例如：



## 表单标签详解

### 输入项标签

<input/>标签

表单输入项标签之一，用户可以在该标签上 通过填写和选择 进行数据的输入。

* type:设置该标签的种类
* text:文本框。 默认
* password:密码框。 内容为非明文
* radio:单选框。 在同一组内有单选效果
* checkbox:复选框。 在同一组内有复选效果
* submit:提交按钮。用于控制表单提交数据
* reset:重置按钮。 用于将表单输入项恢复到默认状态
* file:附件框。用于文件上传。
* hidden:隐藏域。一般用作提交服务器需要拿到，但用户不需要看到的数据。
* button:普通按钮。需要和JS事件一起用
* name: 单选框、复选框进行数据的分组。/ 设置该标签对应的参数名

某个表单输入项需要通过参数列表提交，就必须设置name属性

* value:设置该标签对应的参数值。 / 作为按钮的名字

value属性的设置策略：

①文本框、密码框这样的表单输入项，可以不强制指定value，因为用户可以自由输入

②单选框、复选框这样的表单输入项，必须强制指定value，因为用户无法输入，只能选择，如果不指定value，那么提交上去的只有on

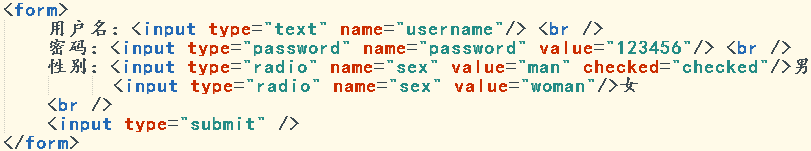
* checked:设置单选框/复选框的默认选中状态
* readonly:设置该标签的参数值只读，用户无法手动更改。数据是可以正常提交
* disabled:设置该标签不可用，参数值无法更改，且参数值也无法提交

参数列表的格式：

参数名1=参数值1&参数名2=参数值2&参数名3=参数值3…….

例如：username=zhangsan&password=123&sex=man

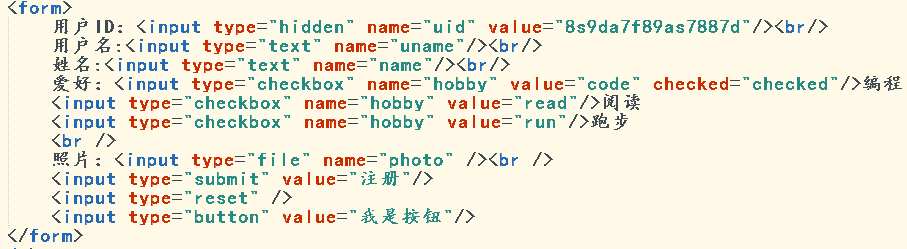
示例1：



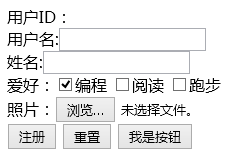
效果1：



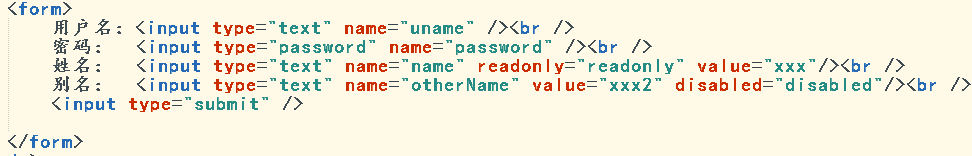
示例2：



效果2：



示例3：



效果3：



### 选择框标签

<select></select>标签 定义一个选择框

* name: 设置该标签对应的参数名
* multiple：设置该标签选项全部显示，并且可以进行多选提交。默认为单选。

<option></option>标签

选项标签，用于为一个选择框添加一个选项

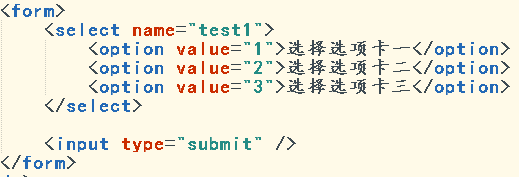
* value:设置需要提交的参数值。
* selected:设置选项的默认选中状态

注意事项：

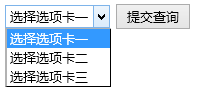
Option的内容体一般是用来进行展示

参数值 应该是option的value属性值

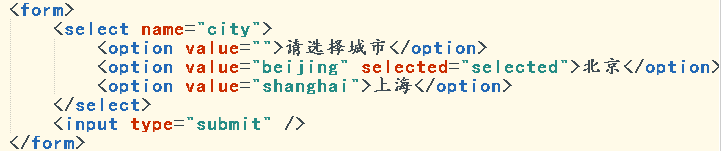
示例1：



效果1：



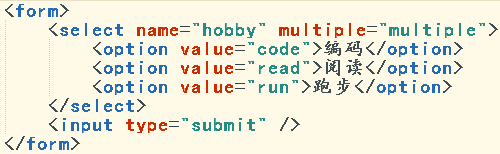
示例2：



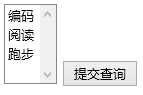
效果2：



示例3：



效果3：



### 文本域标签

<textarea></textarea>标签

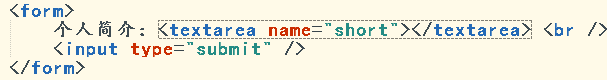
表单输入项标签之一，用户可以在该标签上 通过输入 进行数据的输入。

* name: 设置该标签对应的参数名

文本域和文本框区别：

1. 文本框不能换行，文本域可以
2. 文本框参数值是value属性，文本域参数值是标签的内容体

示例1：



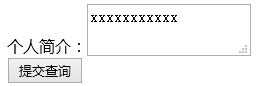
效果1：



示例2：



效果2：



### 表单标签

<form></form>标签。

用于在效果中定义一个表单，用于提交用户填写的数据。

* action:将数据提交到何处。

默认提交到本页。

本机内网路径：

相对路径：

绝对路径：

互联网路径：

http://www.baidu.com/xxx

* method:将数据以何种方式提交

默认为：get

提交方式可定义：get 或者 post

Get提交方式特点：把数据拼接到地址栏上

Post提交方式特点：没有把提交数据拼接到地址栏上。请求体

Get和post提交方式区别：

1. get提交的参数列表拼接到了地址栏后面

post方式不会 拼接地址栏

②get方式提交的数据 敏感信息不安全

Post方式提交的数据 相对安全

③get方式提交的数据量 有限的

Post方式从理论上提交的数据量 无限大

尽量使用post方式提交表单

### 提交表单注意事项

#### 提交中文问题

* **为什么要使用URL编码？**

正常提交数据：

?password=123&username=zhangsan

非正常提交数据：

?password=123#ohheih&dlajfk&dskfklas&username=zhangsan

URL编码解决方式：特殊符号，中文

例如：& -🡪 %26

?password=123#ohheih&dlajfk&dskfklas&username=zhangsan

编码完：?password=123#ohheih %26dlajfk %26dskfklas&username=zhangsan

为了保证表单数据传递时能更好区分出name和value。

保证数据传递的完整性

* **URL编码的表现形式&本质**

①你好，进行普通的编码，编码成字节数组。【使用的是页面规定的字符集，例如utf-8】

1. 字节数组中的每一个元素，都会从10进制，转为16进制
2. 把已经转为16进制的字节数组，以%进行拼接 。拼接出的字符串，就是URL编码后的结果

例如：你好，用UTF-8编码，编码的字节数组为：

[-28, -67, -96, -27, -91, -67]

URL编码后：

%E4%BD%A0%E5%A5%BD

# CSS

## CSS的概述

### CSS是什么

CSS 通常称为CSS样式或层叠样式表，主要用于设置HTML页面中的文本内容（字体、大小、对其方式等）、图片的外形（高宽、边框样式、边距等）以及版面的布局等外观显示样式。

CSS可以是HTML页面更好看，CSS色系的搭配可以让用户更舒服，CSS+DIV布局更佳灵活，更容易绘制出用户需要的结构。

### CSS名词解释

CSS (Cascading Style Sheets) ：指层叠样式表

样式：给HTML标签添加需要显示的效果。

层叠：使用不同的添加方式，给同一个HTML标签添加样式，最后所有的样式都叠加到一起，共同作用于该标签。

### CSS样式规则

使用HTML时，需要遵从一定的规范。CSS亦如此，要想熟练的使用CSS对网页进行修饰，首先需要了解CSS样式规则。具体格式如下

选择器{属性1:属性值;属性2:属性值;..}

在上面的样式规则中，“选择器”用于指定CSS样式作用的HTML对象，花括号内是对该对象设置的具体样式。属性和属性值以键值对方式出现，使用英文冒号“:”分隔。多个属性之间使用英文分号“;”分隔。例如：

<style>

h2{

color:red;

font-size:100px;

}

</style>

初学者在书写CSS样式时，除了要遵循CSS样式规则，还必须注意CSS代码结构中的几个特点，具体如下：

* CSS样式“选择器”严格区分大小写，“属性”和“属性值”不区分大小写。
  + 多个属性之间必须用英文状态下的分号隔开，最后一个属性后的分号可以省略，但是，为了便于增加新样式最好保留。
  + 如果属性的值由多个单词组成且中间包含空格，则必须为这个属性值加上英文状态下的引号。例如：

p { font-family:"Times New Roman";}

* + 在编写CSS代码时，为了提高代码的可读性，通常会加上CSS注释，例如：

/\* 这是CSS注释文本,此文本不会显示在浏览器窗口中 \*/

* + 在CSS代码中空格是不被解析的，花括号以及分号前后的空格可有可无。因此，可以使用空格键、Tab键、回车键等对样式代码进行排版，即所谓的格式化CSS代码，这样可以提高代码的可读性。例如：

h1{ font-size:20px; color:red; }

和

h1{

font-size:20px; /\* 定义字体大小属性 \*/

color:red; /\* 定义颜色属性 \*/

}

* + 上述两段代码所呈现的效果是一样的，但是，第二种书写方式的可读性更高。需要注意的是，属性的值和单位之间是不允许出现空格的，否则浏览器解析时会出错。例如，下面这行代码就是不正确的。

h1{ font-size:20 px; } /\* 20和单位px之间有空格 \*/

### 引入CSS样式

CSS使用非常灵活，及可以嵌入在HTML文档中，也可以是一个单独的文件，如果是单独的文件，则必须以.css为扩展名。CSS和HTML的结合3种常用方式：

1. 行内样式

行内样式，是通过标签的style属性来设置元素的样式。

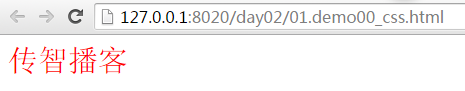
<!--方式1：行内样式

color : 表示字体颜色

font-size : 表示字体大小

-->

<a style="color: #f00; font-size: 30px;">传智播客</a>



行内样式通过标签的属性来控制样式，这样并没有做到结构与表现（HTML展示结构、CSS显示效果）相分离，所以一般很少使用。学习阶段有时候为了快速编程，偶有使用。

1. 内部样式

内部样式又称为内嵌式，是将CSS代码集中卸载HTML文档的<head>头部标签体中，并且使用<style>标签定义。

给当前html文件中的多个标签设置样式。

在html的<head>标签中使用<style>标签来定义CSS

<!--方式2：内部样式

background-color ： 表示背景色

-->

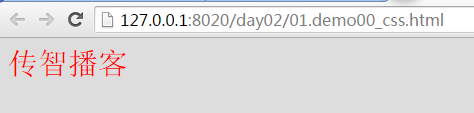
<style type="text/css">

body{

background-color: #ddd;

}

</style>



内嵌式CSS样式只对其所在的HTML页面有效，可以对多处标签统一设置样式，因此，仅设计一个页面时，使用内嵌式是个不错的选择。但如果是一个网站，不建议使用这种方式，因为他不能充分发挥CSS代码的重用优势。

1. 外部样式

外部样式又称为链入式，是将所有的样式放在一个或多个以.css为扩展名的外部样式表文件中，通过<link>标签将样式连接到HTML文档中。

<!--方式3：外部样式

rel="stylesheet" ,固定值，表示样式表

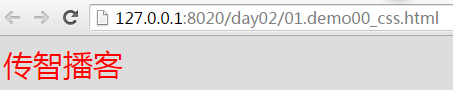
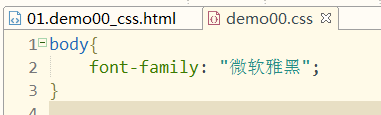
type="text/css"，固定值，表示css类型

href ，表示css文件位置

font-family 表示使用的字体系列，多个字体使用逗号分隔。例如：“字体1,字体2,字体3”，此时优先使用“字体1”，如果“字体1”系统不存在，再使用“字体2”，以此类推。

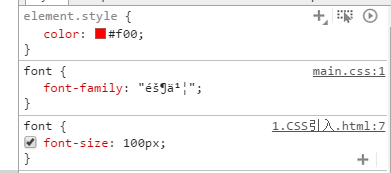
-->

<link rel="stylesheet" type="text/css" href="css/demo00.css"/>



链入式最大的好处是同一个CSS样式表可以被不同的HTML页面链接使用，同时一个HTML页面也可以通过多个<link />标记链接多个CSS样式表。

优先级



## CSS的选择器

要想将CSS样式应用于特定的HTML元素，首先需要找到该目标元素。在CSS中，执行这一任务的样式规则部分被称为选择器，本小节将对CSS基础选择器进行详细地讲解，具体如下：

### 元素选择器

标记选择器是指用HTML标记名称作为选择器，按标记名称分类，为页面中某一类标记指定统一的CSS样式。其基本语法格式如下：

标记名{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

该语法中，所有的HTML标记名都可以作为标记选择器，例如body、h1、p、strong等。用标记选择器定义的样式对页面中该类型的所有标记都有效。

例如：

h1{

color: #F00;

font-size: 50px;

}

<h1>传智播客</h1>



标记选择器最大的优点是能快速为页面中同类型的标记统一样式，同时这也是他的缺点，不能设计差异化样式。

### ID选择器

id选择器使用“#”进行标识，后面紧跟id名，其基本语法格式如下：

#id名{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

该语法中，id名即为HTML元素的id属性值，大多数HTML元素都可以定义id属性，元素的id值是唯一的，只能对应于文档中某一个具体的元素。

例如：

#demo1{

color:#0f0;

}

<h1 id="demo1">黑马训练营</h1>

“标签选择器”和“id选择器”共同作用的效果



### 类选择器

类选择器使用“.”（英文点号）进行标识，后面紧跟类名，其基本语法格式如下：

.类名{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

该语法中，类名即为HTML元素的class属性值，大多数HTML元素都可以定义class属性。类选择器最大的优势是可以为元素对象定义单独或相同的样式。

例如：

.myClass{

font-size: 25px;

}

<h1 class="myClass">Java培训</h1>

“标签选择器”和“类选择器”共同作用的效果



类选择器的高级用法：给指定的标签设置class样式

标签.类名{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

### 扩展：属性选择器

属性选择器，在标签后面使用中括号标记，其基本语法格式如下：

标签名[标签属性=’标签属性值’]{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

该选择器，是对“元素选择器”的扩展，对一组标签进一步过滤。

例如：

<style>

input[type="text"]{

background-color: yellow;

}

input[type="password"]{

background-color: green;

}

</style>

### 扩展：包含选择器

包含选择器，两个标签之间使用空格，给指定父标签的后代标签设置样式，可以方便在区域内编写样式。

父标签 后代标签{属性1:属性值1; 属性2:属性值2; 属性3:属性值3; }

该选择器，是对“元素选择器”的扩展，对一个标签内部所有后代标签进行过滤。

<style>

#d1 div{

color: red;

}

</style>

## CSS的样式

### 边框和尺寸：border、width、height

* border ：设置边框的样式
  + 格式：宽度 样式 颜色
  + 例如：style=”border:1px solid #f00” ，1像素实边红色。
    - 样式取值：solid 实线，none 无边，double 双线 等
* width、height：用于设置标签的宽度、高度。

<style type="text/css">

div{

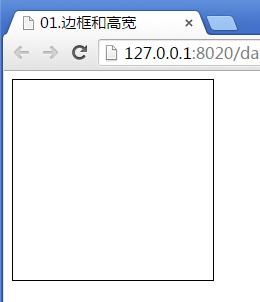
border:1px solid #000; /\*1像素，实边，黑色\*/

width:200px;

height:200px;

}

</style>



### 转换：display

HTML提供丰富的标签，这些标签被定义成了不同的类型，一般分为：块标签和行内标签。

* 块标签：以区域块方式出现。每个块标签独自占据一整行或多整行。
  + 常见的块元素：<h1>、<div>、<ul>等
* 行内元素：不必在新的一行开始，同时也不强迫其他元素在新的一行显示。
  + 常见的行内元素：<span>、<a> 等

在开发中，希望行内元素具有块元素的特性，需要使用display进行转换

选择器{display:属性值}

常用的属性值：

inline：此元素将显示为行内元素（行内元素默认的display属性值）

block：此元素将显为块元素（块元素默认的display属性值）

none：此元素将被隐藏，不显示，也不占用页面空间。

例如：

<html>

<head>

<meta charset="UTF-8">

<title></title>

<style type="text/css">

span{

border :1px solid #000;

width:100px;

height:40px;

}

</style>

</head>

<body>

<!--默认显示一行，边框环绕，高宽没有作用-->

<span>显示1-1</span>

<span>显示1-2</span>

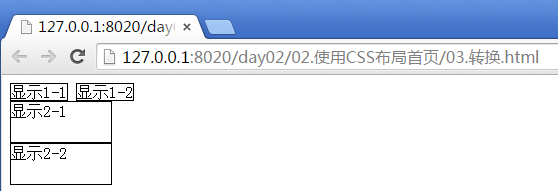
<!--每一行显示，高宽有作用-->

<span style="display: block;">显示2-1</span>

<span style="display: block;">显示2-2</span>

</body>

</html>



### 字体：color、font-size

color：颜色，字体颜色

例如：

<div style="height: 50px;width:200px;border:1px solid #000;">

<a href="">点击</a>

<a href="" style="text-decoration: none;">点击</a>

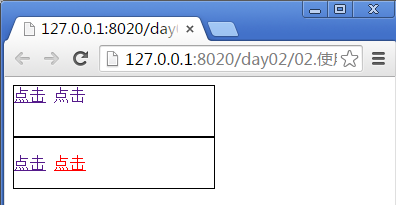
</div>

<div style="height: 50px;width:200px;border:1px solid #000;">

<a href="" style="line-height: 50px;">点击</a>

<a href="" style="color: red;">点击</a>

</div>



### 背景色：background-color

<ul style="background-color: #999;color: #FFF;">

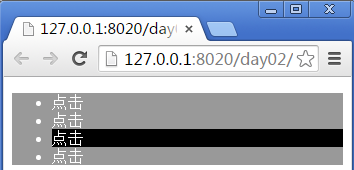
<li>点击</li>

<li>点击</li>

<li style="background-color: #000;">点击</li>

<li>点击</li>

</ul>



### 布局：float、clear

通常默认的排版方式，将页面中的元素从上到下一一罗列，而实际开发中，需要左右方式进行排版，就需要使用浮动

选择器{float:属性值;}

常用属性值：

left：元素向左浮动

right：元素向右浮动

none：元素不浮动（默认值）

由于浮动元素不再占用原文档流的位置，所以它会对页面中其他元素的排版产生影响。如果要避免影响，需要使用clear属性进行清除浮动。

选择器{clear:属性值;}

常用属性值：

left：不允许左侧有浮动元素（清除左侧浮动的影响）

right：不允许右侧有浮动元素（清除右侧浮动的影响）

both：同时清除左右两侧浮动的影响

例如：

<!--默认上下布局-->

<div>

<div>区域1-1</div>

<div>区域1-2</div>

</div>

<hr />

<!--浮动左右布局-->

<div>

<div style="float: left;">区域2-1</div>

<div style="float: left;">区域2-2</div>

</div>

<!--取消浮动，另起一行布局-->

<div style="clear:both"></div>

<hr />

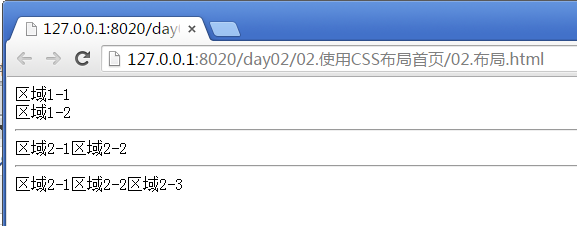
<div>

<div style="float: left;">区域2-1</div>

<div style="float: left;">区域2-2</div>

<div style="float: left;">区域2-3</div>

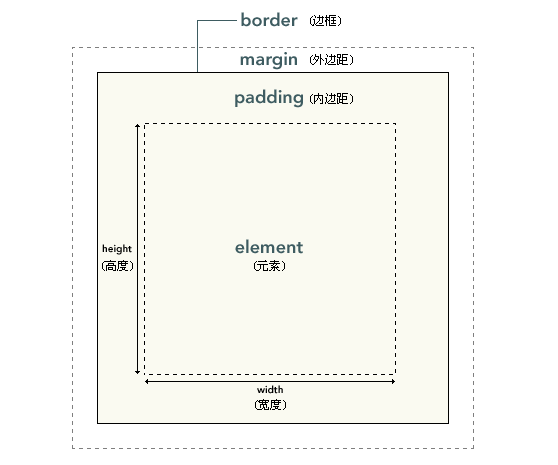
</div>



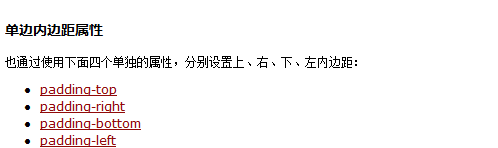
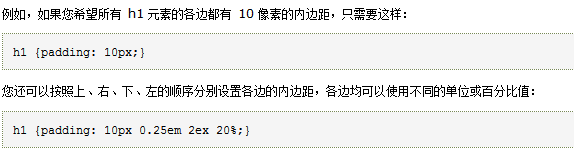
## CSS的盒子模型

### 什么是盒子模型

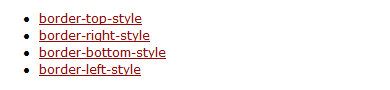
CSS 框模型 (Box Model) 规定了元素框处理元素内容、内边距、边框 和 外边距 的方式。



### 内边距：padding.



### 边框：border



### 外边距：margin

