

# 一、html

## 1. 什么是HTML

Hyper Text Markup Language 超文本标签语言，用于描述网页中存在哪些网页元素。

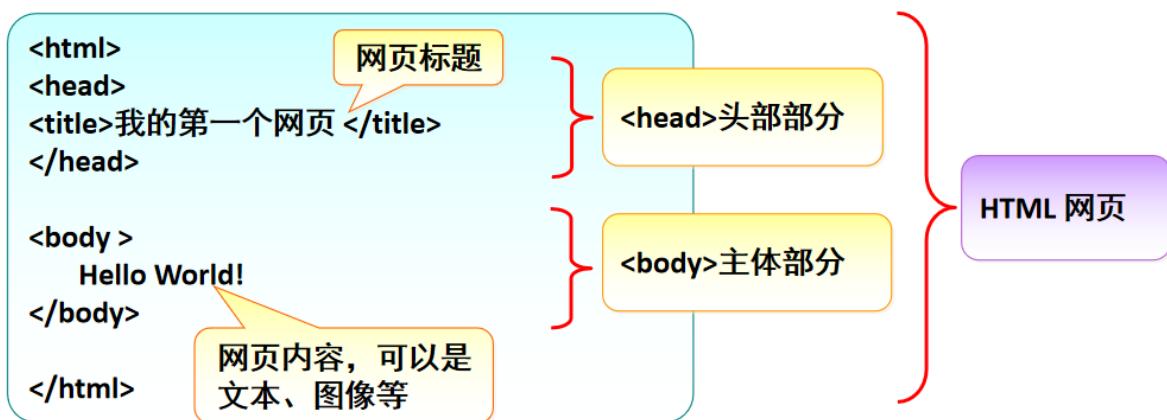
网页的“源码”。

浏览器：“解释和执行”HTML源码的工具

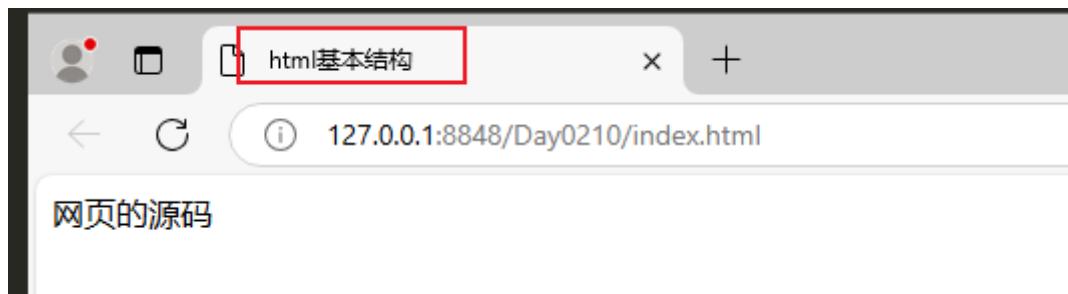


## 2. HTML文档的基本结构

html文件的后缀： .html



```
<!DOCTYPE html>
<html>
    <!-- head 是头部 -->
    <head>
        <meta charset="utf-8" />
        <!-- 网页的标题 -->
        <title>html基本结构</title>
    </head>
    <!-- body 主体 -->
    <body>
        网页的源码
    </body>
</html>
```



### 3. 网页的html的声明

Html5的声明: <!DOCTYPE html>

html4的声明:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

### 4. 标签的闭合特性

- 闭合标签：由开始标签和结束标签组成。开始标签是被括号包围的元素名，结束标签是被括号包围的斜杠和元素名。

代码示例: <p></p><a></a>

- 非闭合标签：是被括号包围的元素名和斜杠。

代码示例: <input/><img />

### 5. 标签的分类

- 块级标签

- 块级元素特点：
  - 总是以一个块的形式表现出来，占领一整行。若干同级块元素会从上之下依次排列（使用float属性除外）。
  - 可以设置高度、宽度、各个方向外补丁（margin）以及各个方向的内补丁（padding）。
  - 当宽度（width）缺省时，它的宽度时其容器的100%，除非我们给它设定了固定的宽度。

- 块级元素中可以容纳其他块级元素或行内元素。
- 常见的块级元素由

- **等等。**
- **块级元素的display属性值默认为block**

- 行级标签

- 行内元素特点：
- 它不会单独占据一整行，而是只占领自身的宽度和高度所在的空间。若干同级行内元素会从左到右（即某个行内元素可以和其他行内元素共处一行），从上到下依次排列。
- 行内元素不可以设置高度、宽度，其高度一般由其字体的大小来决定，其宽度由内容的长度控制。
- 行内元素只能设置左右的margin值和左右的padding值，而不能设置上下的margin值和上下padding值。因此我们可以通过设置左右的padding值来改变行内元素的宽度。
- 常见的行内元素由**等等**。
- **行内元素一般不可以包含块级元素。**
- **块级元素的display属性值默认为inline**

## 6. 基本块级标签

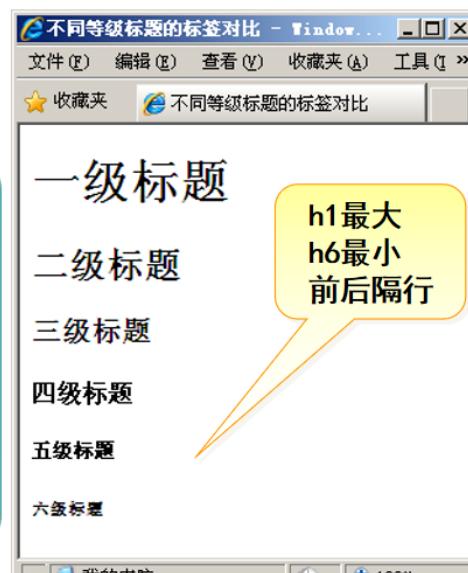
### 6.1 标题标签

 **语法**

```
<h1>标题</h1>
.....
<h6>标题</h6>
```

 **示例**

```
.....
<body>
  <h1>一级标题</h1>
  <h2>二级标题</h2>
  <h3>三级标题</h3>
  <h4>四级标题</h4>
  <h5>五级标题</h5>
  <h6>六级标题</h6>
</body>
.....
```



h1最大  
h6最小  
前后隔行

### 6.2 段落标签

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

## 6.3 水平线标签

```
<hr />
```

## 6.4 列表标记

- 有序列表

```
◦ <ol>  
    <li>列表项1</li>  
    ... ...  
    </ol>
```

- 无序列表

```
◦ <ul>  
    <li>列表项1</li>  
    .....  
    </ul>
```

## 6.5 定义描述标签

```
<dl>  
    <dt>标题</dt>  
    <dd>描述1</dd>  
    .....  
    </dl>
```

咖啡

一种黑色的热饮料，原料据说是咖啡豆，非洲盛产这类原料。可以提神，刺激神经。

## 6.6 表格

```
<table>
<tr>
    <td>百度</td>
    <td>新浪</td>
</tr>
.....
</table>
```

<table>--表格  
<tr> --行  
<td> --列（单元格）



```
<!-- 表格: table 表格  tr 行  td 列-->
<!-- border 边框 -->
<table border="1" width="500px" height="100px" align="center">
    <!-- 第一行 -->
    <!-- 对齐方式 :align="center/left/right"-->
    <tr align="center">
        <td>第1列</td>
        <td>第2列</td>
        <td>第3列</td>
    </tr>
    <!-- 第二行 -->
    <tr>
        <td>第1列</td>
        <td align="center">第2列</td>
        <td align="right">第3列</td>
    </tr>
</table>
```

第1列	第2列	第3列
第1列	第2列	第3列

- 表格的跨行跨列
  - 跨行 **rowspan**
  - 跨列 **colspan**
  - 无论是跨行还是跨列都是在td上设置的属性
- 表格的距离
  - **cellpadding** 内容与边框之间的距离

- **cellspacing** 单元格之间的距离

```
<!-- cellpadding 内容与边框之间的距离  
      cellspacing 单元格之间的距离  
-->  
<table border="1" width="300px" cellpadding="20px" cellspacing="20px">  
  <tr>  
    <!-- 跨列 -->  
    <td colspan="2">学生成绩</td>  
  </tr>  
  <tr>  
    <td>语文</td>  
    <td>98</td>  
  </tr>  
  <tr>  
    <td>数学</td>  
    <td>95</td>  
  </tr>  
</table>
```

学生成绩	
语文	98
数学	95 激活 Windows 转到“设置”以激活 Windows。

- 高级标签



## 6.7 表单

- *form* 表单
  - *method*
    - *get* 声明本次请求的目的是从服务器获取数据
    - *post* 声明本次请求的目的是向服务器传送数据。
    - *get* 与 *post* 的区别
      - 提交数据长度不同, *get*: 不超过255个字符, *post*理论上不受限制。
      - 安全性*get*在地址栏显示信息, 不安全。
      - 缓存机制不同, *get*请求的地址会保存到浏览器的访问历史记录中, *post*不会。
  - *action*
    - 规定当提交表单时, 表单数据的提交地址。

```

<!-- 表单标签 -->
<form action="02-练习.html" method="GET">
    <!-- type="text" 输入文本框
        placeholder 提示信息
        value 默认值
        size 设置输入框的宽度
        maxLength 最大长度
    -->
    用户名: <input type="text" name="uname" id="" placeholder="请输入用户名"
    value="admin" > <br />
    <!-- type="password" 密码框 -->
    密码: <input type="password" placeholder="请输入密码" size="10"
    maxLength="6" name="upass" > <br />

    <!-- type="radio" 单选按钮
        name 属性将多个单选按钮分为一组，实现只能选择其中一个的功能
        checked 默认被选中的
    -->
    性别:
    <input type="radio" name="sex" checked/> 男
    <input type="radio" name="sex"/> 女
    <br>

    <!-- type="checkbox" 复选框 -->
    兴趣爱好: <br />
    <input type="checkbox" name="hobby" checked/>运动
    <input type="checkbox" name="hobby"/>聊天
    <input type="checkbox" name="hobby"/>玩游戏
    <br>

    <!-- select>option 下拉列表 -->
    所在城市:
    <select>
        <option value="">大连</option>
        <option value="">沈阳</option>
        <option value="">锦州</option>
        <option value="">辽阳</option>
        <option value="">葫芦岛</option>
        <option value="">鞍山</option>
    </select>
    <br>

    <!-- 文本域
        cols 宽度
        rows 高度
    -->
    <input type="text" name="text" id="text" cols="30" rows="10" value="请输入内容" />
    <!-- 只读
    -->
    <input type="text" name="text" id="text" cols="30" rows="10" readonly="readonly" />
    同意以上协议
</textarea>
    <input type="checkbox" name="" id="" value="同意" />
    <br>
    <!-- 文件 -->
    <input type="file" name="" id="" />
    <br>
    <!-- 隐藏域 -->
    <input type="hidden" />

```

```

<br>
<!-- type="submit" 提交按钮
    disabled 禁用
-->
<input type="submit" value="注册" disabled>
<!-- type="reset" 重置按钮, 与提交按钮必须在form标签下才有效果 -->
<input type="reset" value="取消">
<!-- 普通按钮没有功能, 需要配合js一起使用 -->
<input type="button" value="普通按钮" onclick="alert('你好')">
<!-- 图片按钮可以提交 -->
<input type="image" src="img/a.jpg">
</form>

```

用户名:

密码:

性别:  男  女

兴趣爱好:

运动  聊天  玩游戏

所在城市:

同意以上协议  同意

未选择任何文件



## 6.8 跑马灯

```

<!-- 跑马灯 -->
<!-- direction属性: 走向(right, left, up, down) scrollamount属性: 速度-->
<marquee direction="right" scrollamount="20">hahahahahhaah...</marquee>
<marquee direction="left" scrollamount="20">hahahahahhaah...</marquee>
<marquee direction="up" scrollamount="5">hahahahahhaah...</marquee>
<marquee direction="down" scrollamount="5">hahahahahhaah...</marquee>

<!-- behavior属性(scroll:环绕走, slide:只走一次, alternate:来回走) -->
<marquee direction="right" scrollamount="20"
behavior="scroll">hahahahahhaah...</marquee>

```

```
<marquee direction="right" scrollamount="20"
behavior="slide">hahahahahhaah...</marquee>
<marquee direction="right" scrollamount="20"
behavior="alternate">hahahahahhaah...</marquee>

<!-- Loop属性 循环次数 -->
<marquee direction="right" scrollamount="20" behavior="alternate"
Loop="3">hahahahahhaah...</marquee>
```

hahahahahhaah...  
hahahahahhaah...  
.....  
hahahahahhaah...  
hahahahahhaah...  
hahahahahhaah...  
hahahahahhaah...

## 7、行级标签

### 7.1 图像标签

- 行内块元素display:inline-block
- 可以设置宽高、在一行显示
- src 图片的资源路径
- alt 当图片下载失败或不存在时的提示信息
- title 鼠标移入图片上方时的提示信息

```


```

### 7.2 超链接

- 锚点：跳转到页面的某个指定位置，使用id属性作为锚点的标记
- # 是id属性的标识符
- target属性值：
  - \_blank 在新窗口中打开被链接文档。
  - \_self 在被点击时的同一框架中打开被链接文档（默认）。
  - \_parent 在父框架中打开被链接文档。
  - \_top 在窗口主体中打开被链接文档。

```
<a href="06-marquee.html">超链接1</a>
```

```

<!-- 锚点：跳转到页面的某个指定位置，使用id属性作为锚点的标记
      # 是id属性的标识符
-->
<a href="#here">跳转到“来了！！！”位置</a>

```

## 7.3 特殊符号

### ❖ 特殊符号

- 空格： &nbsp;
- 大于(>)： &gt;
- 小于(<)： &lt;
- 引号 ("")： &quot;
- 版权号()： &copy;

1、因为<、>等符号在HTML中已使用，  
所以必须用其他符号来代替  
2、都以分号结束 ( ; )



```

<!-- 特殊符号
      &lt; <
      &gt; >
      &nbsp; 空格
-->
3&lt;5&gt;2
<br>
he11o&nbsp;&nbsp;&nbsp;&nbsp;world

<b>加粗</b>
<strong>加粗</strong>
<i>倾斜</i>
<em>倾斜</em>
<s>删除线</s>
<del>删除线</del>
<!-- 字体标签
      color 颜色
      size 字体大小 1-7
-->
<font color="red" size="6">这是一个字体标签</font>

<!-- 范围标签 -->
<span>范围标签</span>

<p>今日商品价格: <span style="color: red; font-size: 50px;">10</span>元</p>

<!-- 下角标 -->
H<sub>2</sub>O
<!-- 上角标 -->
X<sup>2</sup>

```

## 8、框架

- 1、显示多窗口页面--使用<frameset>框架集
- 2、页面复用--使用<iframe>内嵌框架

### ✿ 多个页面文件组成



### ○ 框架页面的基本语法



将窗口分割成左  
右3个部分,可选

将窗口分割成上  
下2个部分,可选

边框尺  
寸大小

```
<frameset cols="25%,50%,*" rows ="50%,*" border="5">  
    <frame src="the_first.html ">  
    ....  
</frameset>
```

引用各窗口要显  
示的网页文件



```
<!DOCTYPE html>  
<html>  
    <head>  
        <meta charset="utf-8">  
        <title></title>  
    </head>  
    <!-- <body>  
    </body> -->  
    <!-- frameset框架集，不能与body共同使用  
    noresize 不允许改变边框的大小  
    bordercolor 边框的颜色  
    border 边框的粗细  
    -->  
<frameset rows="20%,70%,*" bordercolor="blue" border="1">
```

```
<!-- 上 -->
<frame src="top.html" noresize>
<!-- 中间 -->
<frameset cols="25%, *">
    <frame src="left.html" noresize>
    <frame src="right.html" name="right">
</frameset>
<!-- 下 -->
<frame src="bottom.html">
</frameset>
</html>
```

上

超链接1  
超链接2

右

下

激活 Windows  
http://zhidao.baidu.com/question/363111371.html

和<frameset>不同，放在  
<body>标签内

.....

<body>

<iframe src="引用页面地址" name="框架标识名"  
frameborder="边框" scrolling="no" />

<body>

.....

指明引用的网页  
文件

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title></title>
    </head>
    <body>
        <iframe src="08-练习.html" frameborder="0" width="500px"></iframe>
        <p>hahahaha</p>
    </body>
</html>
```

## 申请表

姓名:

密码:

照片:  未选择任何文件

感兴趣的职位:

hahahaha

## 二、CSS

### 1、什么是css

层叠样式表，用来渲染HTML标签的外观样式。例如设定元素的位置、颜色、大小等。

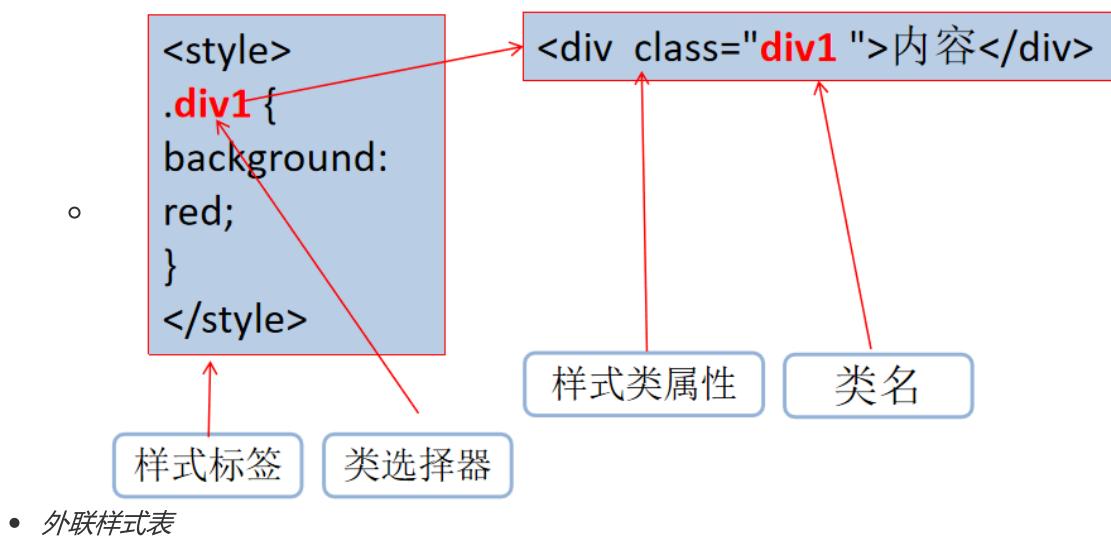
### 2、引入css样式有如下几种方式

- 内联样式表



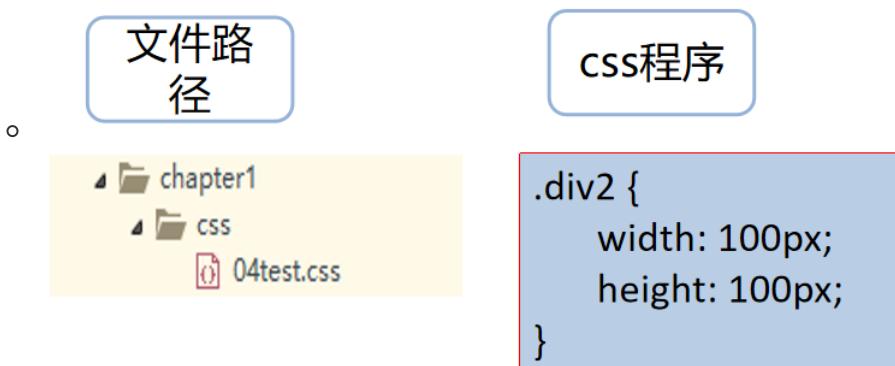
- 嵌入样式表

## - 基本语法



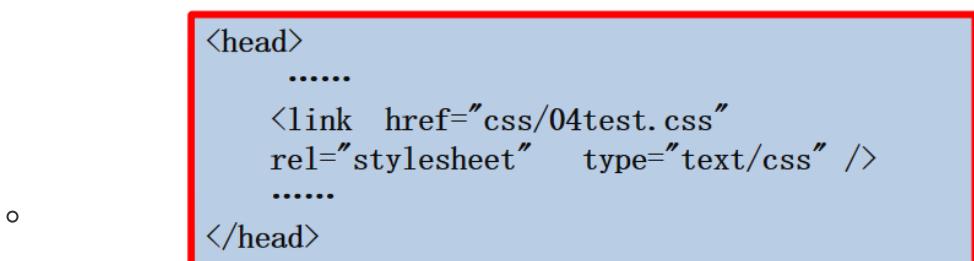
### • 外联样式表

- 外部文件：以css为后缀的文件名，内容为样式程序。



## ★ 外联式(Linking)

- 引入css文件声明



- link元素：连接元素，是head标签的子标签。
- href:设定引入外部文件的路径url。
- type: 设置或获取对象的 MIME 类型。

### • 导入样式表

```
<style type="text/css">  
    @import  
    url( "css/04test.css");  
  </style>
```

- @import: 导入样式的规则关键字。
- url( "css路径" ): 导入外部css文件的路径。

也可以在外部文件中导入css文件。

### 3、优先级别

内联式（行内样式）

内嵌式和外部样式考虑的执行顺序，最后执行的优先级高

## 4、选择器

选择器：用来找到要被渲染样式的元素。

### 4.1 标签选择器

使用html标签筛选需要渲染的网页元素。

```
div {  
    background-color: yellow;  
}
```



对当前网页所有div元素设定样式

### 4.2 Id选择器

根据标签的id属性筛选要被渲染的标签，标识符是：#

```
#test {  
    color: yellow;  
}
```

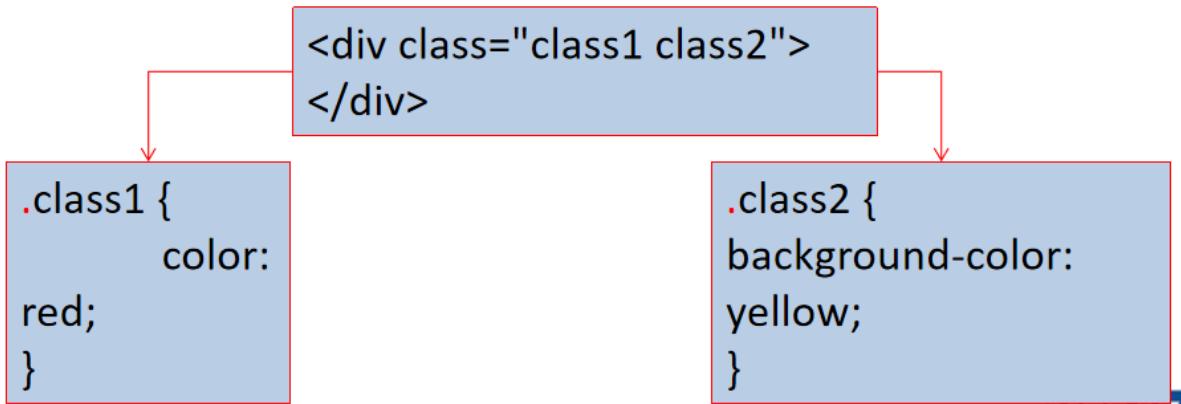


```
<div id = "test">  
id选择器  
</div>
```

## 4.3 类选择器

以class 属性筛选要被渲染的标签，标识符是：点.

class 属性可以有多个值，多个值用空格分割



*id、类、标签选择器的优先级：*

*id选择器 > 类选择器 > 标签选择器*

## 4.4 (多)元素选择器

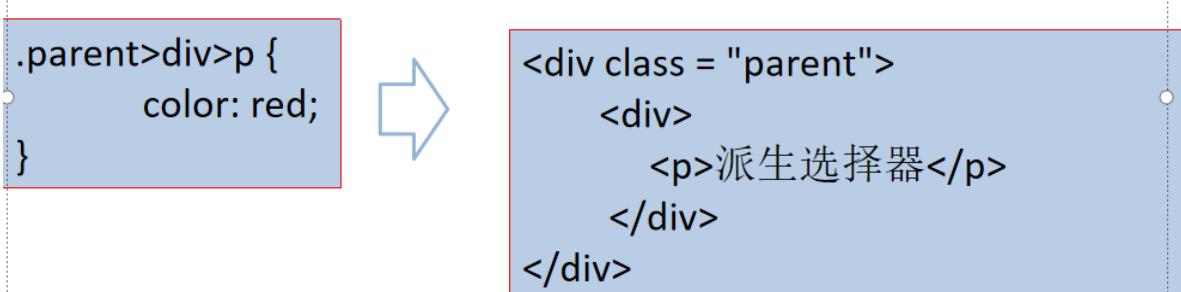
**#p,li,.a** 代表**p**为id的标签、**li**标签、**class**为**a**的标签都执行此样式

```
#p,li,.a{  
    color: red;  
    font-size: 20px;  
}
```

## 4.5 派生选择器

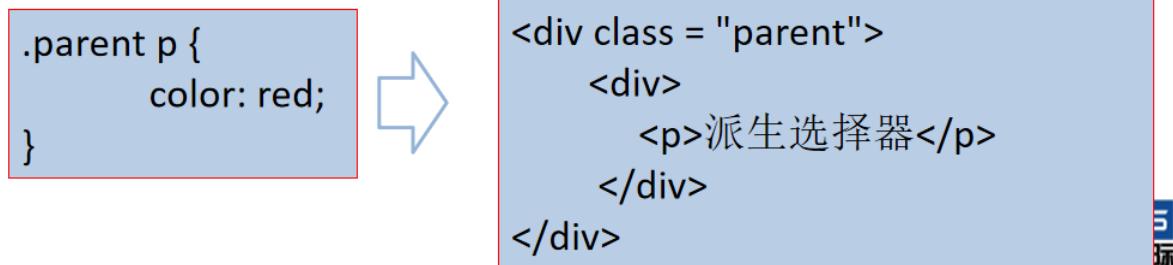
### 4.5.1 子元素选择器

子元素选择器，使用> 作为标识符，按照html标签的层级关系（直接后代），筛选要被渲染的标签。



#### 4.5.2 后代元素选择器

后代选择器，使用空格作为标识符，按照html标签的层级关系（全部后代子孙）筛选被渲染的标签。



#### 4.5.3 相邻元素选择器

紧邻其后的第一个元素被渲染样式，标识符 +

**div+p** 代表div标签的相邻标签p执行此样式  
(注意：只执行相邻的一个标签)

```
div+p{
  color: red;
}
```

```
<div>
  <p>123</p>
</div>
<p>456</p>
<p>789</p>
```

#### 4.5.4 同辈元素选择器

其后的所有同辈元素，标识符是 ~

The screenshot shows a code editor with two panes. The left pane displays a portion of a CSS file with the following content:

```
/*
 * 同辈选择器: ~ , 其后的所有同辈元素 */
#fu~p{
  /* 阴影 */
  text-shadow: 10px 10px pink;
}
/style>
```

The right pane shows an HTML document structure with the following code:

```
<body>
  <!-- div#fu -->
  <div id="fu">
    <p>p1</p>
    <span>span1</span>
    <div id="div2">
      <p>p2</p>
    </div>
  </div>
  <p>p3</p>
  <p>p4</p>
</body>
```

In the HTML structure, the <p> elements labeled "p3" and "p4" are highlighted with a red border, indicating they are selected by the sibling selector "#fu~p".

### 4.6 属性选择器

根据元素的html属性或属性值筛选要被渲染的元素。标识符是 【】

## - 基本语法

```
selector[attr][attr='value']{  
    此处为css属性以及取值  
}
```

```
<!DOCTYPE html>  
<html>  
    <head>  
        <meta charset="utf-8">  
        <title></title>  
        <style>  
            /* 属性选择器:[] */  
  
            /* a标签中包含href属性 */  
            a[href]{  
                color: red;  
            }  
  
            /* a标签中与class属性，并且class属性为aa */  
            a[class='aa']{  
                background-color: aqua;  
            }  
  
            /* a标签中有title属性和class属性 */  
            a[title][class]{  
                /* 去掉下划线 */  
                text-decoration: none;  
                /* 字体颜色 */  
                color: yellowgreen;  
            }  
  
            /* 有title属性，属性值以。。。开头 */  
            a[title^='t']{  
                font-size: 30px;  
            }  
  
            /* 有title属性，属性值以。。。结尾 */  
            a[title$='t']{  
                font-size: 50px;  
            }  
  
            /* 有title属性，属性包含指定的值 */  
            a[title*= 't']{  
                font-size: 10px;  
            }  
  
            /* 通用选择器 */
```

```

        /*
        /* 去掉外边距 */
        margin: 0;
    }

```

</head>

<body>

<a href="">第一个超链接</a>

<br>

<a href="" class="aa">第一个超链接</a>

<br>

<a href="" title="tt" class="aa">第一个超链接</a>

</body>

</html>

## 5、css属性

### 5.1 文本属性

属性	描述	取值
<b>text-transform</b>	检索或设置对象中的文本的大小写	none   capitalize   uppercase
<b>white-space</b>	检索或设置对象内空格的处理方式(空白处理)	normal   pre   nowrap   pre-wrap   pre-line
<b>text-align</b>	检索或设置对象中内容的水平对齐方式	left   right   center
<b>letter-spacing</b>	检索或设置对象中的字符之间的最小，最大和最佳间隙	normal   长度单位
<b>text-indent</b>	检索或设置对象中的文本的缩进	长度单位
text-align	检索或设置对象内容的对齐方式	center
line-height	行高	长度单位
text-decoration	文本修饰	None underline line-through
vertical-align	检索或设置对象内容的垂直对其方式	top   middle   bottom

### 5.2 字体属性

属性	描述	取值
font	复合属性。设置或检索对象中的文本特性	看独立属性
font-style	设置或检索对象中的字体样式	normal   italic   oblique
font-weight	设置或检索对象中的文本字体的粗细	normal   bold   bolder   lighter
font-size	设置或检索对象中的字体尺寸	
font-family	设置或检索用于对象中文本的字体名称序列	
font-variant	设置或检索对象中的文本是否为小型的大写字母	normal   small-caps

## 5.3 背景属性

属性	描述	取值
background	复合属性。设置或检索对象的背景特性	看独立属性
background-color	设置或检索对象的背景颜色	
background-image	设置或检索对象的背景图像	url(img/user.jpg)
background-repeat	设置或检索对象的背景图像如何平铺填充	repeat-x   repeat-y   [repeat   no-repeat]
background-attachment	设置或检索对象的背景图像是随对象内容滚动还是固定的	fixed   scroll
background-position	设置或检索对象的背景图像位置	[ ]   [ left   center   right   top   bottom ]

## 5.4 列表属性

*list-style* (列表风格)

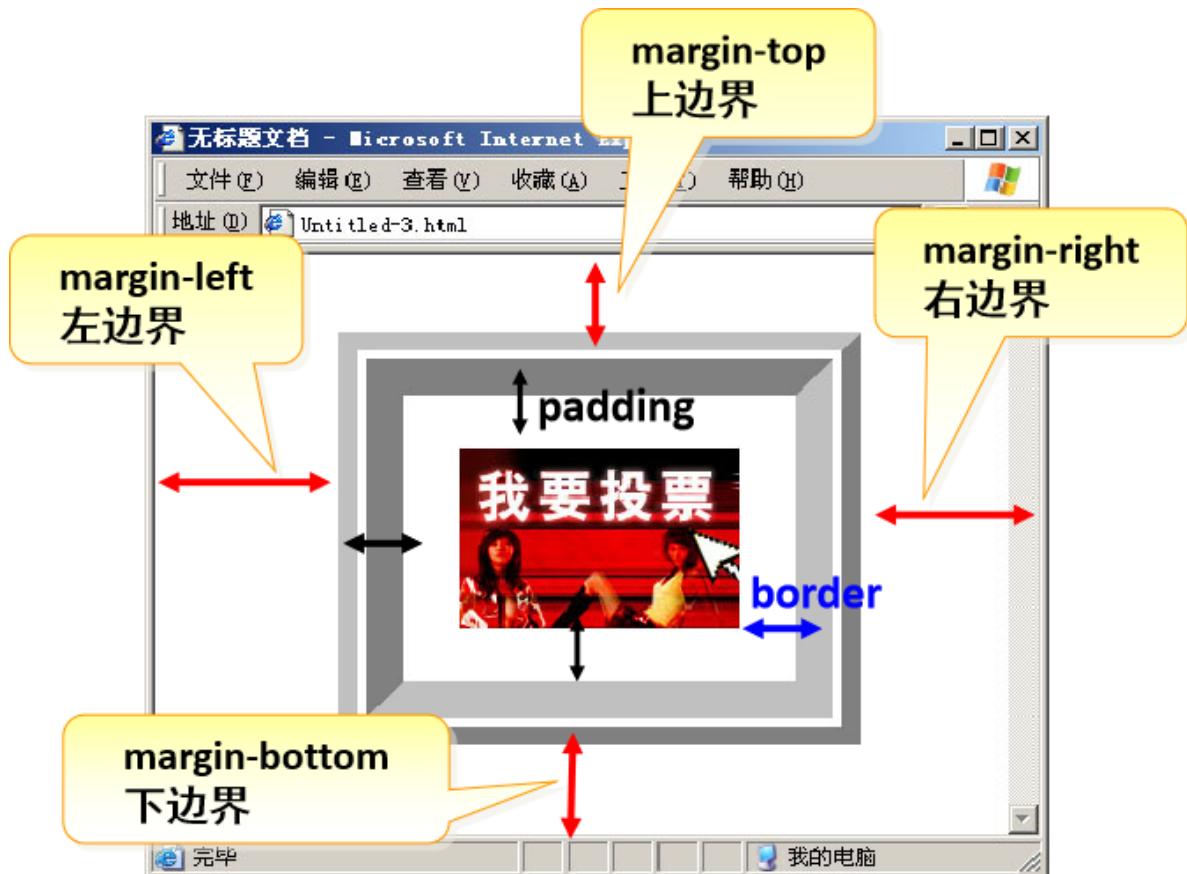
属性值	方式	语法实现	示例
none	无风格	list-style:none;	刷牙 洗脸
disc	实心圆 (** 默认类型) **	list-style:disc;	• 刷牙**• 洗脸**
circle	空心圆	list-style:circle;	○ 刷牙**○ 洗脸**
square	实心正方形	list-style:square;	■ 刷牙**■ 洗脸**
decimal	数字 (** 默认类型) **	list-style:decimal	1. 刷牙**2.** 洗脸

## 5.5 伪类样式

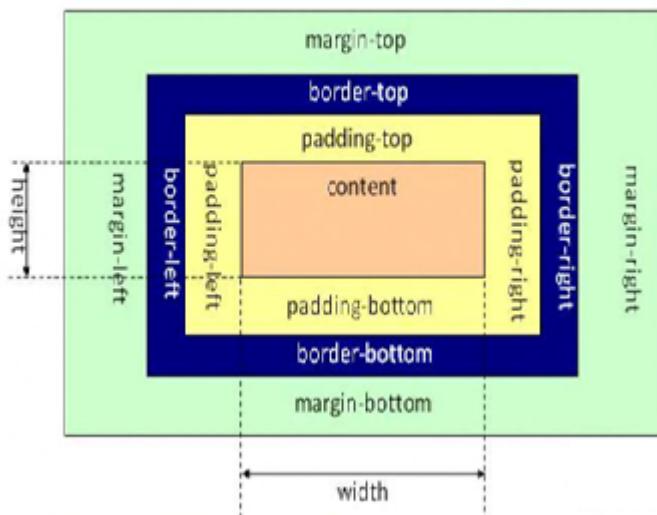
- `a:link {color:yellow;}/ * 未被访问的链接 */`
- `a:visited {color:red;}/ * 已被访问的链接 */`
- `a:hover {color:green;}/ * 鼠标指针移动到链接上 */`
- `a:active {color:blue;}/ * 正在被点击的链接 */`

## 6、盒模型

盒模型包含：外边距，边框，填充（内边距）和实际内容。



## ■ 标准盒子模型



### 6.1 外边距属性：设置对象四边的外部边距

属性	描述	取值
margin	检索或设置对象四边的外部边距	复合属性，看独立属性（可以为负值）
margin-top	检索或设置对象顶边的外部边距	长度单位
margin-bottom	检索或设置对象底边的外部边距	长度单位
margin-left	检索或设置对象左边的外部边距	长度单位
margin-right	检索或设置对象右边的外部边距	长度单位

外边距属性的特殊说明：

- 内联块级元素和块级元素的可以设置外边距。
- 内联元素可以设置左、右两边的外边距；若要设置上、下两边的外边距，必须先使该元素变为块级元素或内联块级元素。
- 如果提供全部四个参数值，将按上、右、下、左的顺序作用于四边。
- 如果提供两个，第一个用于上、下，第二个用于左、右。
- 如果提供三个，第一个用于上，第二个用于左、右，第三个用于下。

### 6.2 内边距属性：设置对象四边的内部边距

属性	描述	取值
padding	检索或设置对象四边的内部边距	
padding-top	检索或设置对象顶边的内部边距	
padding-bottom	检索或设置对象底边的内部边距	
padding-left	检索或设置对象左边的内部边距	
padding-right	检索或设置对象右边的内部边距	

### 内边距属性的特殊说明：

- 设置内联块级元素和块级元素的内边距。
- 行内元素可以设置左、右两边的内边距；若要设置上、下两边的内边距，必须先使该元素变为块级或内联块级元素。
- 改变padding的值，就改变了content的大小，  
而改变margin的值，则不改变content的大小

### ✿ 元素的实际占位（实际宽、高）

- 盒子高度 = height属性 + 上下填充高度 + 上下边框高度
- 盒子宽度 = width属性 + 左右填充宽度 + 左右边框宽度

## 6.3 边框属性：设置边框的特性

属性	描述	取值
border	复合属性。设置对象边框的特性	
border-width	设置或检索对象边框宽度	thin   medium   thick
border-style	设置或检索对象边框样式	none   hidden   dotted   dashed   solid   double   groove   ridge   inset   outset(详情请参见附录)
border-color	设置或检索对象边框颜色	
border-radius	设置或检索对象使用圆角边框	[ ]
border-top/bottom/left/right-width	设置或检索对象顶/底/左/右边样式	thin   medium   thick

## 7、浮动

## - float : 设定元素以浮动流方式显示。

- none: 设置对象不浮动
- left: 设置对象浮动方向向左
- right: 设置对象浮动方向右

```
.float {  
    float: left;  
}
```

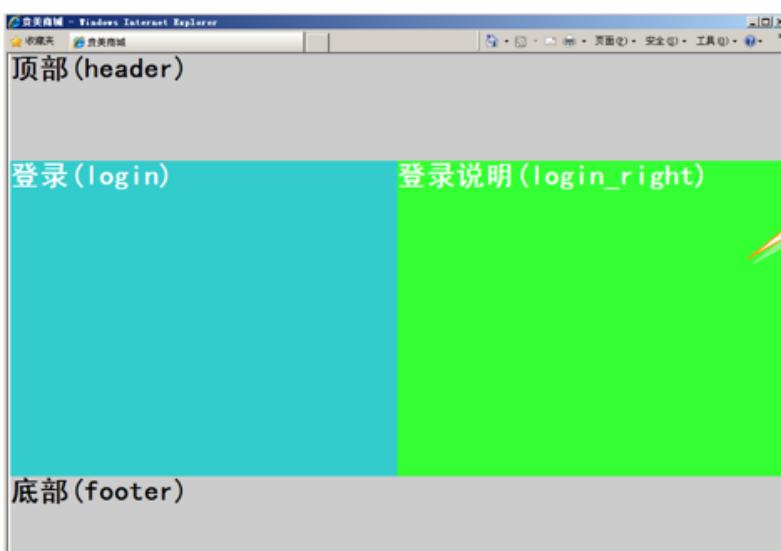
## - clear: 清除浮动。

- none: 不清除浮动。
- both: 清除两侧浮动。
- left: 清楚左侧浮动。
- right: 清除右侧浮动。

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

### 需求说明：

- 在前面上机练习基础上，实现页面中间布局



```
<!DOCTYPE html>  
<html>  
    <head>  
        <meta charset="utf-8">  
        <title></title>  
        <style>  
            *{  
                margin: 0;  
            }  
            #top{  
                height: 220px;  
                background-color: #cccccc;  
            }  
  
            #left{  
                width: 50%;  
                height: 500px;  
            }
```

```

background-color: aqua;
float: left;
}
#right{
width: 50%;
height: 500px;
background-color:#33ff33;
float: right;
}
#bottom{
/* 清除浮动 */
clear: both;
height: 200px;
background-color: #cccccc;
}

```

</style>

</head>

<body>

<div id="top">顶部</div>

<div id="left">登录</div>

<div id="right">登录说明</div>

<div id="bottom">底部</div>

</body>

</html>

## 8、定位

定位属性：使元素脱离正常的文档流，“漂浮”在指定的位置上的css属性。

属性	描述	取值
position	检索对象的定位方式	relative   absolute   fixed   static
z-index	检索对象的定位方式	auto   :用整数来定义堆叠级别，可以为负数
top	检索或设置对象参照相对物顶边界向下偏移位置。	长度单位
bottom	检索或设置对象参照相对物底边界向上偏移位置。	长度单位
left	检索或设置对象参照相对物底边界向右偏移位置。	长度单位
right	长度单位	长度单位

- **position:** 设置定位属性的方式。
  - **relative:** 相对定位，对象遵循普通流，当前元素参照父元素的左上角进行位置偏移。
  - **absolute:** 绝对定位，对象脱离普通流，当前元素偏移属性参照的是离自身最近的相对定位元素(父类元素)，如果没有相对定位的元素，则一直追溯到文档(浏览器)。
  - **fixed:** 与absolute一致，但偏移定位是以窗口为参考。当出现滚动条时，对象不会随着滚动。
  - **static:** 对象遵循常规流(默认值)。

## 9、控制显示类属性

– **display**: 设定元素的显示类型，常用取值如下。

- none: 隐藏对象，不占据空间。

- inline: 指定对象为内联元素。

- block: 指定对象为块元素。

- inline-block: 指定对象为内联块元素。

```
.display {  
    display: block;  
}
```

– **visibility** : 设定是否显示元素，常用取值如下

- visible: 设置对象可视，但占据空间

- hidden: 设置对象隐藏

```
.visibility {  
    visibility: hidden;  
}
```

## 三、JavaScript

### 1、什么是JavaScript

用来控制网页元素实现动态效果。

- Javascript的基本特点：

- 脚本语言：JavaScript是一种解释型的脚本语言。

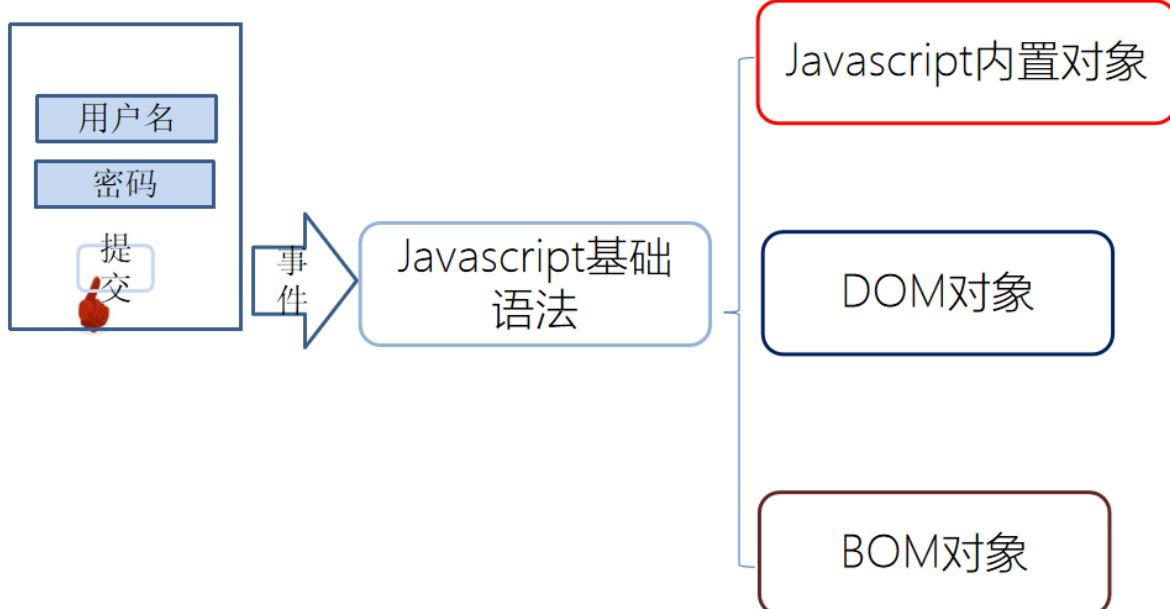
- 基于对象：JavaScript是一种基于对象的语言，而不是纯粹的面向对象的语言。

- 简单：采用的是弱类型的类型声明方式。

- 跨平台：依赖于浏览器而非操作系统。

- 嵌入式：开发中需要操作html元素与css样式，因此需要在html中引入。

### 2、javascript核心组成



*JavaScript内置对象：*

核心基础，变量、数据类型、流程控制语句。。

*DOM对象：*

*Document Object Model 文档对象模型*

对html中的标签进行操作的。读写操作、表单验证。。

*BOM对象：*

*Browser Object Model 浏览器对象模型*

上一步、下一步、关闭窗口、打开窗口、打印... 刷新

### 3、三种方式引入javascript程序

#### 3.1 内嵌方式引入

```
<script type="text/javascript" charset="UTF-8">
    内容
</script>
```

#### 3.2 外部文件方式引入

```
<script src="js/02outer.js" defer="defer"></script>
```

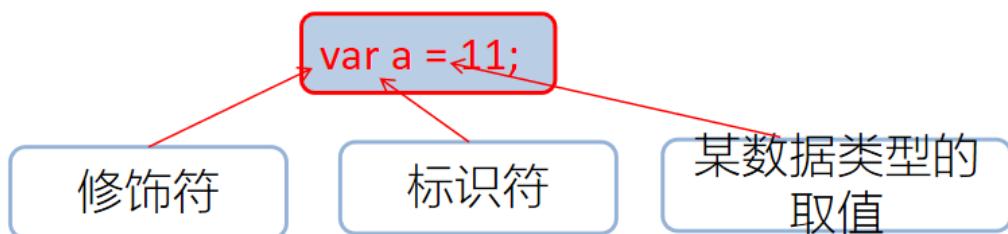
### 3.3 事件方式引入

```
<button onclick="document.write('1111')>按钮</button>
```

## 4、JavaScript核心语法

### 4.1 变量的概念

变量是存储以及访问临时信息的容器。



Js中使用var作为变量的修饰符，js是弱类型的语言

标识符就是函数、变量的名字，由开发者自己声明

Js中也具备基础数据类型与封装（对象）数据类型

修饰符：

`var` 全局，变量 在函数中是局部的，其他情况是全局的。

`let` 局部，变量

`const` 局部，常量，只能赋值一次。

### 4.2 数据类型分类

- 基础数据类型包含如下几种

1. undefined
2. null类型
3. boolean类型
4. number类型
5. string类型

两者区别



声明方式不同

内存存储方式不同

- 对象数据类型：非基础数据类型外都是对象（封装）数据类型

*undefined与null的区别：*

*null代表暂时没有赋值，undefined值未定义。*

*null分配内存空间，undefined不分配内存空间。*

*null可以参与运算，undefined无法参与运算。*

```
// undefined 不分配内存，不能参与运算
console.log(a+10); // NaN : not a number 不是一个数字
// null 分配内存，可以参与运算
console.log(b+10); // 10
```

- 数据类型的检测方式
- **typeof**: 获取目标变量的数据类型，以字符串形式返回，**typeof**是系统内置函数。
- **object**: javascript中的对象、数组和null

```
// 数据类型: 1、基本类型 2、对象类型
// 未定义的
let a;

// 空值
let b = null;

// 布尔boolean, true 和 false
let c = true;

// number 数值, 整数和小数(浮点数)
let d = 12.3;
let e = 30;

// string 字符串, "" / ''
let f = "abc";
let g = '你好';

// 打印到控制台
console.log(typeof a);
console.log(typeof b);
console.log(typeof c);
console.log(typeof d);
```

```
console.log(typeof e);
console.log(typeof f);
console.log(typeof g);
```

undefined

object

boolean

number

number

string

string

- **类型转换**

```
parseInt( '字符' ); // 将目标字符转换为数字，小数部分舍掉
parseFloat( '字符' ); // 将目标字符转换为浮点
```

```
<script>
    var a = "123";
    console.log(a + 6); // 1236
    // parseInt() 将字符串转成整数
    console.log(parseInt(a) + 6); // 129

    console.log(parseInt("你好123") + 6); // NaN
    console.log(parseInt("123你好") + 6); // 129
    console.log(parseInt("123.45") + 6); // 129

    // parseFloat() 将字符串转成浮点数
    console.log(parseFloat("123.45") + 6); // 129.45

    console.log(3 + true); // 4
    console.log(3 + false); // 3
</script>
```

## 4.3 三个对话框

- 提示框: `alert('title');`
  - 只有一个提示信息和一个确认按钮

127.0.0.1:8848 显示

○ 你好JavaScript!

确定

- 确认框: `confirm('title');`

- 返回值是`boolean`类型
- 一个提示信息, 一个确认按钮, 一个取消按钮
- 点击确认按钮, 返回`true`
- 点击取消按钮, 返回`false`

127.0.0.1:8848 显示

你确定要删除吗?

○

确定

取消

- 输入框: `prompt('title');`

- 返回值是`string`类型的
- `prompt('title', '默认值');`
- `prompt('title');`
- 一个提示信息, 一个输入框, 一个确认按钮, 一个取消按钮
- 点击确定按钮, 返回输入的框中的值
- 点击取消按钮, 返回`null`

127.0.0.1:8848 显示

请输入用户名

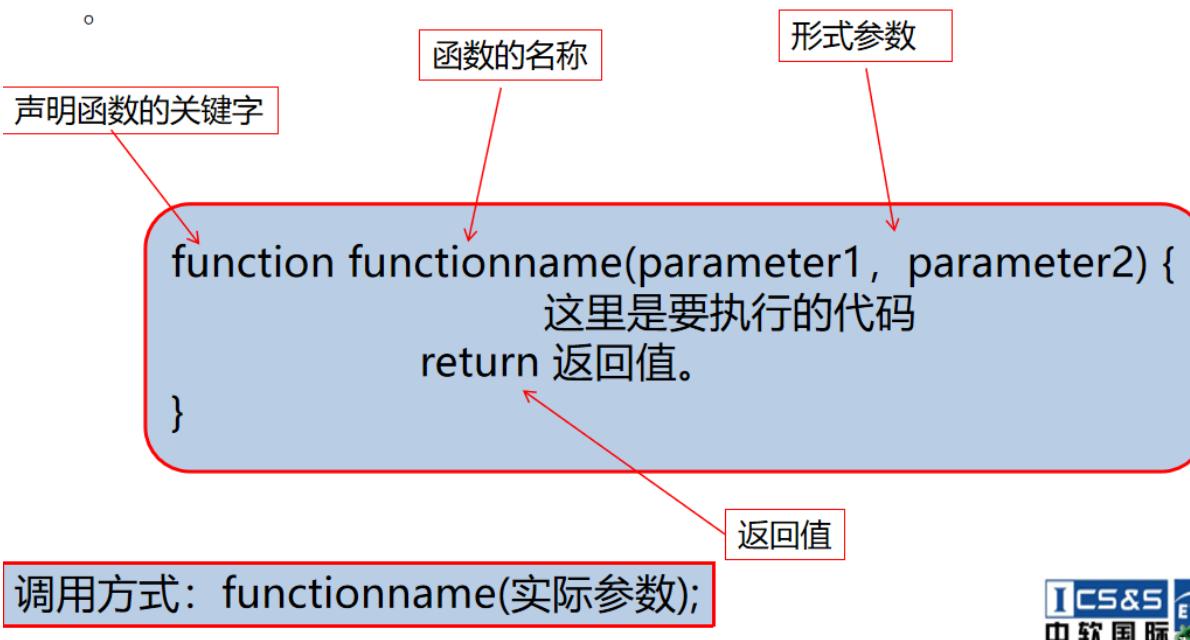
○

tom

确定

取消

## 5. 函数



## 6、对象

```
// 声明构造函数
function Student(name, age){
    this.name = name;
    this.age = age;
}

// 创建学生对象
let s1 = new Student("tom", 18);
let s2 = new Student("lisa", 20);

// 获取学生的信息
// 将内容写入到html网页上
document.write(s1.name + ":" + s1.age + "<br/>")
document.write(s2.name + ":" + s2.age)

// 字面量创建对象
// 语法结构: json格式 {key:value, key:value...}
let dog1 = {
    name: "旺财",
    age: 1,
    eat: function(){
        alert(this.name + "喜欢啃骨头")
    }
}

document.write(dog1.name + "--" + dog1.age)
dog1.eat()
```

## 7、数组

一组相同或不同数据类型数据的容器

- 数组的初始化方式

- new Array方式

```
var array = new Array();  
array[0] = 1;  
array[1] = 2;  
array[2] = 3;  
var array1 = new Array(1,2,3,4,5);  
var array2=new Array('大','家','好');  
for(var j=0;j<array2.length;j++){  
    document.writeln(array2[j]);  
}  
for(var index in array1){  
    document.writeln(array1[index]);  
}
```

- 使用[] 初始化并直接赋值

```
var array5 = [1,2,3,4,5];  
for(var j=0;j<array5.length;j++){  
    document.writeln(array5[j]);  
}
```

- 

```
var a = [1,2,3,4];  
var b = a;//数组赋值  
a[1] = 5;  
b[0] = 9;  
document.write(a + "<br>");// 9,5,3,4  
document.write(b );//9,5,3,4
```

- 数组的API方法

- 颠倒数组元素的顺序: `array.reverse()`
  - 对数组进行升序排序 数组本身发生变化: `array.sort()`
  - 将数组元素通过指定字符进行连接 返回值是String类型: `array.join("-")`

```

var array6 = [3,4,2,1,5,6,9,8];
//颠倒数组元素的顺序
document.writeln(array6.reverse());
//对数组进行升序排序 数组本身发生变化
document.writeln(array6.sort());
//将数组元素通过指定字符进行连接 返回值是String类型
var array4 = [1,2,3,true,5];
var str1 = array4.join("-");
document.write(str1);
document.write(array4.toString());//将数组转换为字符串

```

## 8、事件

指用户在通过外部设备与浏览器发生交互后，由浏览器调用本地程序响应客户操作的过程。

- 事件类型：用户与浏览器的交互类型，例如鼠标操作，键盘操作等，常用的事件类型如下：
  - 窗口事件类型
    - *onload* 页面加载
    - *onchange* 内容发生改变
  - 焦点事件类型
    - *focusout*:失去焦点事件。
    - *focusin*:得到焦点事件。
    - *blur* : 老版本的失去焦点事件，与*focusout*功能一致
    - *focus* : 老版本的得到焦点事件，与*focusin*功能一致
  - 鼠标事件类型
    - *click* : 鼠标点击后抬起事件。
    - *dblclick* : 鼠标双击事件。
    - *mousedown* : 鼠标左键被按下时触发的事件。
    - *mouseup* : 鼠标左键被抬起时触发的事件。
    - *mouseover* : 鼠标移动到某对象上方时触发的事件。
    - *mouseout* : 鼠标离开某对象范围时触发的事件。
    - *mousemove* : 鼠标移动时触发的事件，响应函数将不断被调用。
    - *contextmenu* : 鼠标右键菜单弹出前触发的事件。
  - 键盘事件类型
    - *keydown* : 当键盘按下时调用，如不松手则反复调用（非输入按键只调用一次），输入中文时，每次点击都会调用。
    - *keyup* : 当键盘松开时调用。
    - *keypress* : 与*keydown*相同，但对于非字符输入不调用。

## 9、Document对象

- *document*访问基本元素DOM对象的方式：
  - *document.documentElement* : 获取HTML标签的dom对象。

- `document.documentElement`: 获取HTML文档的文档头字符。
  - `document.body`: 获取body的dom对象。
  - `document.head`: 获取head标签的dom对象。
  - `document.title`: 获取title的文字标题。
- 子节点DOM对象的方式:
    - 根据ID获取DOM第一个对象，如果获取不到返回null。

`DomObj document.getElementById( 'id' )`
    - 根据标签名（忽略大小写）获取全部该标签的对象数组。

`HTMLCollection document.getElementsByTagName( 'tagName' )`
    - 根据name属性获取全部该标签的对象数组。
      - `HTMLCollection document.getElementsByName( 'name' )`
    - 根据样式类名属性获取全部该标签的对象数组。（非标准，但被主流浏览器兼容）

`HTMLCollection document.getElementsByClassName( 'cn' )`
    - 根据选择器获取

`DomObj document.querySelector("#div")`



## 10、节点操作

### 10.1 查看/修改属性节点

- `getAttribute("属性名")`
- `setAttribute("属性名", "属性值")`

### 10.2 设定样式

– 设定或获取CSS属性的方式：

`obj.style.backgroundColor = "yellow";`

– 设定或获取CSS样式类的方式：

`obj.className = "divClass";`

## 10.3 创建和增加节点的方法

- `createElement ()` : 创建节点
- `appendChild ()` : 末尾追加方式插入节点
- `insertBefore ()` : 在指定节点前插入新节点
- `cloneNode (boolean)` : 克隆节点(是否复制子节点)
- `removeChild ()` : 删除节点
- `replaceChild()` : 替换节点

## 10.4 Html属性

- `innerHTML` : 设置或获取当前元素内的html。
- `innerText` : 设置或获取当前元素内的文字内容。
- `value` : 设置或获取表单元素的内容