# javascript基础知识

## 1，js的使用格式

|  |  |
| --- | --- |
| C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(26).png | C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(27).png |
| **放在<head>部分**  最常用的方式是在页面中head部分放置<script>元素，浏览器解析head部分就会执行这个代码，然后才解析页面的其余部分。  进行页面显示初始化的js必须放在head里面，因为初始化都要求提前进行（如给页面body设置css等）；而如果是通过事件调用执行的function那么对位置没什么要求的。 | **放在<body>部分**  JavaScript代码在网页读取到该语句的时候就会执行。 |

## 2，语句和符号

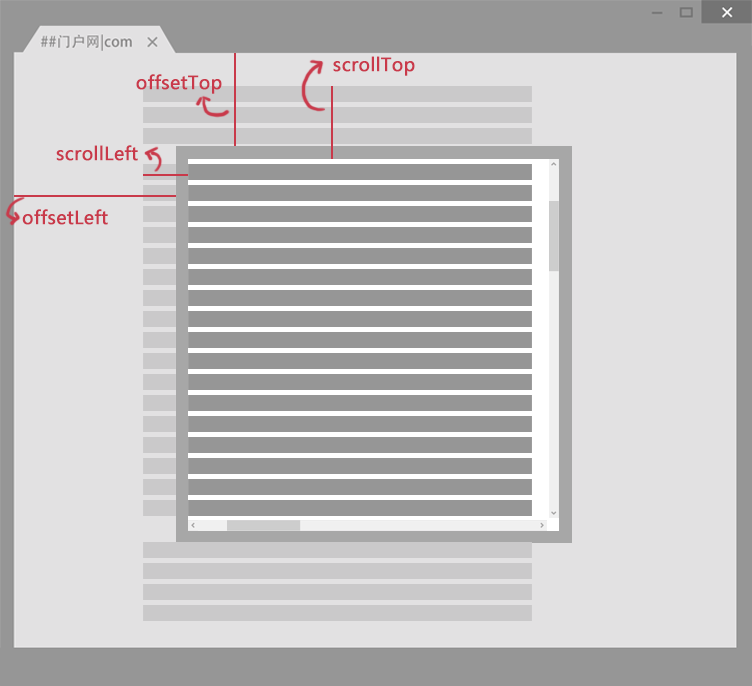
|  |  |  |  |
| --- | --- | --- | --- |
| **定义变量使用关键字var,语法如下：**  **var** 变量名 | **function 函数名(x,y,x)**  {       函数代码;  **return retVal**;  } | document.write()  **第一种:输出内容用“”括起，直接输出""号内的内容**  **第二种:通过变量，输出内容**  **第三种:输出多项内容，内容之间用+号连接**  **第四种:输出HTML标签，并起作用，标签使用“”括起来** | 输出警示框：  **alert(字符串或变量);** |
| 输出确认框  **confirm(str);** | 输出提问框:  **prompt(str1, str2);**  str1: 要显示在消息对话框中的文本，不可修改 str2：文本框中的内容，可以修改 | close()关闭窗口  **用法：**  window.close();   //关闭本窗口  <窗口对象>.close();   //关闭指定的窗口 |  |

|  |  |  |
| --- | --- | --- |
| open() 方法可以查找一个已经存在或者新建的浏览器窗口  window.open([URL], [窗口名称], [参数字符串]) | **URL：**可选参数，在窗口中要显示网页的网址或路径。如果省略这个参数，或者它的值是空字符串，那么窗口就不显示任何文档。 **窗口名称：**可选参数，被打开窗口的名称。     1.该名称由字母、数字和下划线字符组成。     2."\_top"、"\_blank"、"\_selft"具有特殊意义的名称。        \_blank：在新窗口显示目标网页        \_self：在当前窗口显示目标网页        \_top：框架网页中在上部窗口中显示目标网页     3.相同 name 的窗口只能创建一个，要想创建多个窗口则 name 不能相同。     4.name 不能包含有空格。 **参数字符串：**可选参数，设置窗口参数，各参数用逗号隔开。 | C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(28).png |

## 3，DOM操作

文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）。

|  |  |  |
| --- | --- | --- |
| **HTML文档可以说由节点构成的集合，三种常见的DOM节点:**  **1. 元素节点：**上图中<html>、<body>、<p>等都是元素节点，即标签。  **2. 文本节点:**向用户展示的内容，如<li>...</li>中的JavaScript、DOM、CSS等文本。  **3. 属性节点:**元素属性，如<a>标签的链接属性href="http://www.imooc.com"。 | <a href="http://www.imooc.com">JavaScript DOM</a>  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(29).png | **节点属性:**  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(1).jpg](http://img.mukewang.com/5375c953000117ee05240129.jpg)  **遍历节点树:**  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(2).jpg](http://img.mukewang.com/53f17a6400017d2905230219.jpg) |
| **DOM操作:**  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(3).jpg](http://img.mukewang.com/538d29da000152db05360278.jpg)  insertBefore(newnode,node);  如果要完全删除对象，给 x 赋 null 值，代码如下:  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(4).jpg](http://img.mukewang.com/539975a800017c8e04790082.jpg) | 通过id获取元素：  document. get**Elements**ByTagName(Tagname) 返回数组  document.get**Elements**ByName(name) 返回数组  **document.getElementById(“id”)**  **结果:null或[object HTMLParagraphElement]**  **object.className = classname**  1.获取元素的class 属性  2. 为网页内的某个元素指定一个css样式来更改该元素的外观  innerHTML 属性用于获取或替换 HTML 元素的内容。  **Object.innerHTML** | **设置CSS样式：**  **Object.style.property=“new style”;**  **property可以取右边表格所列，也可以是css样式中的其他属性**  Object.style.**display** =“none/block"  不显示/显示为块级元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(30).png |
| 设置/获取节点属性值：  elementNode.getAttribute(name)  elementNode.setAttribute(name,value) | 浏览器可视区域大小：  **一、对于IE9+、Chrome、Firefox、Opera 以及 Safari：**  •  window.innerHeight - 浏览器窗口的内部高度  •  window.innerWidth - 浏览器窗口的内部宽度 | 网页尺寸大小：  scrollHeight和scrollWidth，获取网页内容高度和宽度。  scrollHeight和scrollWidth还可获取Dom元素中内容实际占用的高度和宽度。  offsetHeight和offsetWidth，获取网页内容高度和宽度(包括滚动条等边线，会随窗口的显示大小改变)。  offsetHeight = clientHeight + 滚动条 + 边框。 |



## 4，JS事件驱动

|  |  |  |  |
| --- | --- | --- | --- |
| C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(32).png | onclick是鼠标单击事件，当在网页上单击鼠标时，就会发生该事件。同时onclick事件调用的程序块就会被执行，通常与按钮一起使用。    当鼠标移到一个对象上时，该对象就触发onmouseover事件，并执行onmouseover事件调用的程序。    鼠标移开事件，当鼠标移开当前对象时，执行onmouseout调用的程序    当网页中的对象获得聚点时，执行onfocus调用的程序就会被执行 | onblur事件与onfocus是相对事件，当光标离开当前获得聚焦对象的时候，触发onblur事件，同时执行被调用的程序。    选中事件，当文本框或者文本域中的文字被选中时，触发onselect事件，同时调用的程序就会被执行。    通过改变文本框的内容来触发onchange事件，同时执行被调用的程序。    当用户退出页面时（页面关闭、页面刷新等），触发onUnload事件，同时执行被调用的程序。 | 事件会在页面加载完成后，立即发生，同时执行被调用的程序。 注意：1. 加载页面时，触发onload事件，事件写在<body>标签内。        2. 此节的加载页面，可理解为打开一个新页面时  也可以使用window.onload和window.onunload代替在body标签中写 |

## 5，JS内置对象

|  |  |  |  |
| --- | --- | --- | --- |
| Date对象：  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(5).jpg](http://img.mukewang.com/555c650d0001ae7b04180297.jpg)  初始化Date:  var d = new Date(2012, 10, 1);  //2012年10月1日  var d = new Date('Oct 1, 2012'); //2012年10月1日  返回星期：  **getDay()**返回星期，返回的是0-6的数字，0 表示星期天。 | String对象：  stringObject.substr(startPos,length);与substring基本一致，注意length和stopPos  stringObject.substring(starPos,stopPos);获取子字符串  stringObject.split(separator,limit);返回分割后字符串数组  stringObject.indexOf(substring, startpos);返回指定字符串首次出现位置/-1  stringObject.charAt(index);返回指定字符/空字符  var myl=mystr.length;  var mynum=mystr.toUpperCase();  var mynum=mystr.toLowerCase(); | Math对象：  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(33).png | C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(34).png |
| C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(35).png | Array数组：  **创建数组语法：**  var myarray=**new Array();**  创建数组的同时，还可以为数组指定长度，长度可任意指定  **注意：**  1.创建的新数组是空数组，没有值，如输出，则显示undefined。  2.虽然创建数组时，指定了长度，但实际上数组都是变长的，也就是说即使指定了长度，仍然可以将元素存储在规定长度以外。  myarray.length; //获得数组myarray的长度  JavaScript数组的length属性是可变的，这一点需要特别注意。 | 第一种方法：  var myarray = new Array(66,80,90,77,59);//创建数组同时赋值  第二种方法：  var myarray = [66,80,90,77,59];//直接输入一个数组（称 “字面量数组”）  **注意：**数组存储的数据可以是任何类型（数字、字符、布尔值等） | **二维数组的定义方法一**  var myarr=new Array();  //先声明一维  for(var i=0;i<2;i++){   //一维长度为2    myarr[i]=new Array();  //再声明二维     for(var j=0;j<3;j++){   //二维长度为3    myarr[i][j]=i+j;   // 赋值，每个数组元素的值为i+j     }}  **二维数组的定义方法二**  var Myarr = [[0 , 1 , 2 ],[1 , 2 , 3, ]] |

## 6，浏览器对象

|  |  |  |  |
| --- | --- | --- | --- |
| window对象：当前浏览器窗口  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(36).png | window计时器：左列最后四个方法  **计时器类型：**  一次性计时器：仅在指定的延迟时间之后触发一次。  setTimeout(代码,延迟时间);  clearTimeout(id\_of\_setTimeout)  间隔性触发计时器：每隔一定的时间间隔就触发一次  setInterval(代码,交互时间);  clearInterval(id\_of\_setInterval) | History对象：浏览历史记录  **语法：**  window.history.[属性|方法]  **History 对象属性**  [C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(6).jpg](http://img.mukewang.com/53548c030001759e05840068.jpg)  **History 对象方法**  **[C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(7).jpg](http://img.mukewang.com/53548c200001228206210123.jpg)** | Location对象：获取或设置URL  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(37).png  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(38).png |
| navigator对象：浏览器和操作系统信息  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(39).png | screen对象：屏幕信息  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(40).png |  |  |

# Jquery基础知识

## 1, jquery元素选择器

|  |  |  |  |
| --- | --- | --- | --- |
| **Id选择器**：  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(41).png | **element选择器**：  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(42).png | **class选择器**：  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(43).png | **\*选择器**：表示选择全部元素，可以和其他选择器同时使用， 表示获取其他元素中的全部子元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(44).png |
| **逗号选择器**：多个选择器用逗号隔开，表示同时选定多个元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(45).png | **层次选择器**：空格隔开，表示树形结构下某个子元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(46).png | **parent > child选择器**：>表示父元素下第一层子元素  **prev + next选择器**：+表示prev相邻的下一元素  **prev ~  nexts选择器**：~表示prev相邻的所有同级元素 |  |

## 2，jquery过滤性选择器

|  |  |  |  |
| --- | --- | --- | --- |
| **:first选择器**：选择元素集合中第一个元素，同理**:last**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(3).png | **:eq(index)选择器**：选择元素集合中某一索引元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(4).png | **:contains(text)选择器**：元素集合中包含某一文本的元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(5).png | **:has(选择器）**:元素集合中满足另一选择器的元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(6).png |
| **:hidden选择器**：元素集合中所有不可见元素，同理**:visible**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(47).png | **[attribute=value]选择器**：元素集合中指定某属性值的元素。同理**[attribute!=value]**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(48).png | **[attribute\*=value]选择器**：元素集合中某属性值包含value的元素。  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(49).png | **：first-child选择器**：指定元素的首个子元素。同理**:last-child**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(50).png |

## 3，jquery表单选择器

|  |  |  |
| --- | --- | --- |
| **:input选择器**：返回全部表单元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(51).png | **:text选择器**：返回单行文本输入框元素  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(52).png | **:password选择器**：返回密码输入框元素  **:radio选择器**：返回全部单选框元素  **:checkbox选择器**：返回全部复选框元素  **input:submit选择器**: 返回提交按钮元素  **:image选择器**： 返回input type为image标签元素,不会返回img元素  **:button选择器**：返回input type为button的元素  **:checked选择器**： 返回当前被选中的元素  **:selected选择器**： 返回<select>下拉列表框被选中的元素 |

## 4，jquery操作dom

|  |  |  |  |
| --- | --- | --- | --- |
| **元素属性设置**  **attr()**方法的作用是设置或者返回元素的属性，其中attr(属性名)格式是获取元素属性名的值，attr(属性名，属性值)格式则是设置元素属性名的值  http://img.mukewang.com/52bb8ca300011b7203780146.jpg | **元素文本内容设置**：  **html()**和**text()**方法操作元素的内容，当两个方法的参数为空时，表示获取该元素的内容，而如果方法中包含参数，则表示将参数值设置为元素内容。 html()方法可以**获取元素的HTML内容**，因此，原文中的**格式代码**也被一起获取，而text()方法只是**获取元素中的文本内容**，并不包含HTML格式代码  **在元素前后插入内容**：  before()和after()方法可以在元素的前后插入内容，它们分别表示在整个元素的前面和后面插入指定的元素或内容，调用格式分别为：  **$(selector).before(content)**和**$(selector).after(content)** | **元素css样式设置**：  **addClass()**和**css()**方法可以方便地操作元素中的样式，前者括号中的参数为增加元素的样式名称，后者直接将样式的属性内容写在括号中。  http://img.mukewang.com/52bb94080001c89c03270369.jpg | **移除元素属性和样式**：  **removeAttr(name)**和**removeClass(class)**分别可以实现移除元素的属性和样式的功能，前者方法中参数表示移除属性名，后者方法中参数则表示移除的样式名  **追加元素内容**：  append(content)方法的功能是向指定的元素中追加内容，被追加的content参数，可以是字符、HTML元素标记，还可以是一个返回字符串内容的函数。  appendTo()方法也可以向指定的元素内插入内容，它的使用格式是**$(content).appendTo(selector)**  即把content内容插入selector元素内，默认是在尾部 |
| **clone()完整复制一个元素**：  clone()方法可以生成一个被选元素的副本，即复制了一个被选元素，包含它的节点、文本和属性，它的调用格式为：**$(selector).clone()**参数selector可以是一个元素或HTML内容  http://img.mukewang.com/52bb9da80001fede03660123.jpg | **替换元素或内容**：  replaceWith()和replaceAll()方法都可以用于替换元素或元素中的内容，但它们调用时，内容和被替换元素所在的位置不同，分别为如下所示：**$(selector).replaceWith(content)**和**$(content).replaceAll(selector)**  **包括元素或内容**：  wrap()和wrapInner()方法都可以进行元素的包裹，但前者用于包裹元素本身，后者则用于包裹元素中的内容，它们的调用格式分别为：**$(selector).wrap(wrapper)**和**$(selector).wrapInner(wrapper)** | **遍历元素集合**：  each()方法可以遍历指定的元素集合，在遍历时，通过回调函数返回遍历元素的序列号，它的调用格式为：  **$(selector).each(function(index))**  http://img.mukewang.com/52bba3720001175203180227.jpg | **删除元素**：  remove()方法删除**所选元素本身和子元素**，该方法可以通过添加过滤参数指定需要删除的某些元素，而empty()方法则只删除所选元素的**子元素**。  http://img.mukewang.com/52bceff500018e0702910171.jpg |

## 5，jquery事件处理

|  |  |  |  |
| --- | --- | --- | --- |
| **body页面加载事件**：  ready()事件类似于onLoad()事件，但前者只要页面的DOM结构加载后便触发，而后者必须在页面全部元素加载成功才触发，ready()可以写多个，按顺序执行。  http://img.mukewang.com/52d227fa000171db03160136.jpg | **为元素绑定事件处理**：  bind()方法绑定元素的事件非常方便，绑定前，需要知道被绑定的元素名，绑定的事件名称，事件中执行的函数内容就可以，它的绑定格式如下：**$(selector).bind(event,[data] function)**参数event为事件名称，多个事件用空格隔开  http://img.mukewang.com/52d22aeb0001ee2b04480177.jpg | **移除元素绑定事件**：  unbind()方法可以移除元素已绑定的事件，它的调用格式如下：  **$(selector).unbind(event,fun)**  http://img.mukewang.com/52d22feb0001c9ee04750306.jpg | 为元素绑定一次性事件：  one()方法可以绑定元素任何有效的事件，但这种方法绑定的事件只会触发一次，它的调用格式如下：  **$(selector).one(event,[data],fun)**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(53).png |
| **鼠标移入移出事件**：  hover()方法的功能是当鼠标移到所选元素上时，执行方法中的第一个函数，鼠标移出时，执行方法中的第二个函数，实现事件的切实效果，调用格式如下：**$(selector).hover(over，out);**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\52d22c9900013dbd03190241.jpg | **live绑定动态元素事件**：  与bind()方法相同，live()方法与可以绑定元素的可执行事件，除此相同功能之外，live()方法还可以绑定动态元素，即使用代码添加的元素事件，格式如下：**$(selector).live(event,[data],fun)**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(54).png | **手动触发元素某个事件**：  trigger()方法可以直接手动触发元素指定的事件，这些事件可以是元素自带事件，也可以是自定义的事件，总之，该事件必须能执行，它的调用格式为：**$(selector).trigger(event)**  C:\Users\BARRET~1\AppData\Local\Temp\enhtmlclip\Image(55).png |  |

## 6，jquery动画效果

|  |  |  |  |
| --- | --- | --- | --- |
| **显示/隐藏元素**：  show()和hide()方法用于显示或隐藏页面中的元素，它的调用格式分别为：  **$(selector).hide(speed,[callback]和$(selector).show(speed,[callback])**参数speed设置隐藏或显示时的速度值，可为“slow”、“fast”或毫秒数值，可选项参数callback为隐藏或显示动作执行完成后调用的函数名 | **Toggle动态显示隐藏元素**：  toggle()方法就可以很容易做到，即如果元素处于显示状态，调用该方法则隐藏该元素，反之，则显示该元素，它的调用格式是：  **$(selector).toggle(speed,[callback])**  http://img.mukewang.com/52dca25300010e0405830241.jpg | **Slide滑动效果**：  slideUp()和slideDown()方法在页面中滑动元素，前者用于向上滑动元素，后者用于向下滑动元素，它们的调用方法分别为：  **$(selector).slideUp(speed,[callback])和$(selector).slideDown(speed,[callback])**  slideDown()仅适用于**被隐藏的元素；**slideup()则**相反**  **slideToggle**()方法可以切换slideUp()和slideDown()，即调用该方法时，如果元素已向上滑动，则元素自动向下滑动，反之，则元素自动向上滑动，格式为：  **$(selector).slideToggle(speed,[callback])** | **Fade淡入淡出效果**：  fadeIn()和fadeOut()方法可以实现元素的淡入淡出效果，前者淡入隐藏的元素，后者可以淡出可见的元素，它们的调用格式分别为：  **$(selector).fadeIn(speed,[callback])和$(selector).fadeOut(speed,[callback])**  调用fadeTo()方法，可以将所选择元素的不透明度以淡入淡出的效果调整为指定的值，该方法的调用格式为：  **$(selector).fadeTo(speed,opacity,[callback])**  其中speed参数为效果的速度，opacity参数为指定的不透明值，它的取值范围是0.0~1.0 |
| **自定义动画效果：**  animate()方法可以创建自定义动画效果，它的调用格式为：  **$(selector).animate({params},speed,[back])**  其中，params参数为制作动画效果的CSS属性名与值，speed参数为动画的效果的速度，单位为毫秒，可选项callback参数为动画完成时执行的回调函数名  大小变化动画：  http://img.mukewang.com/52dcc3b30001fd5803210256.jpg | 位置移动动画**：**  被移元素的“position”属性值设为“absolute”或“relative”，否则，该元素移动不了  http://img.mukewang.com/52dcc4eb0001c08e03410291.jpg | **停止动画效果：**  stop()方法的功能是在动画完成之前，停止当前正在执行的动画效果，这些效果包括滑动、淡入淡出和自定义的动画，它的调用格式为：  **$(selector).stop([clearQueue],[goToEnd])**  其中，两个可选项参数clearQueue和goToEnd都是布尔类型值，前者表示是否停止正在执行的动画，后者表示是否完成正在执行的动画，默认为false。  **推迟动画效果：**  delay()方法的功能是设置一个延时值来推迟动画效果的执行，它的调用格式为：  **$(selector).delay(duration)**  其中参数duration为延时值，它的单位是毫秒，当超过延时值时，动画继续执行。 |  |

## 7，query实现AJAX

|  |  |
| --- | --- |
| **Load异步加载请求数据**：  load()方法通过Ajax请求加载服务器中的数据，并把返回的数据放置到指定的元素中，它的调用格式为：  **load(url,[data],[callback])**  参数url为加载服务器地址，可选项data参数为请求时发送的数据，callback参数为数据请求成功后，执行的回调函数 | **异步加载json请求数据**：  getJSON()方法可以通过Ajax异步请求的方式，获取服务器中的数据，并对获取的数据进行解析，显示在页面中，它的调用格式为：  **jQuery.getJSON(url,[data],[callback])或$.getJSON(url,[data],[callback])** |
| **异步加载JS文件**：  getScript()方法异步请求并执行服务器中的JavaScript格式的文件，它的调用格式如下所示：  jQuery.getScript(url,[callback])或$.getScript(url,[callback])  参数url为服务器请求地址，可选项callback参数为请求成功后执行的回调函数。 | **Get方式获取服务器数据：**  get()方法时，采用GET方式向服务器请求数据，并通过方法中回调函数的参数返回请求的数据，它的调用格式如下：  $.get(url,[callback]) |
| **Post方式获取服务器数据：**  post()方法多用于以POST方式向服务器发送数据，服务器接收到数据之后，进行处理，并将处理结果返回页面，调用格式如下：  $.post(url,[data],[callback])  参数url为服务器请求地址，可选项data为向服务器请求时发送的数据，可选项callback参数为请求成功后执行的回调函数。 | **Serialize序列化表单元素值：**  使用serialize()方法可以将表单中有name属性的元素值进行序列化，生成标准URL编码文本字符串，直接可用于ajax请求（一般用于get请求的？数据串），它的调用格式如下：  **$(selector).serialize()**  其中selector参数是一个或多个表单中的元素或表单元素本身。 |
| **ajax()原始方法请求服务器数据：**  ajax()方法是**最底层、功能最强大的请求服务器数据的方法**，它不仅可以获取服务器返回的数据，还能向服务器发送请求并传递数值，它的调用格式如下：  **jQuery.ajax([settings])或$.ajax([settings])**  其中参数settings为发送ajax请求时的配置对象，在该对象中，url表示服务器请求的路径，data为请求时传递的数据，dataType为服务器返回的数据类型，success为请求成功的执行的回调函数，type为发送数据请求的方式，默认为get。 | 使用**ajaxSetup()**方法可以设置Ajax请求的一些全局性选项值，设置完成后，后面的Ajax请求将不需要再添加这些选项值，它的调用格式为：  **jQuery.ajaxSetup([options])或$.ajaxSetup([options])**  **ajaxStart()和ajaxStop()**方法是绑定Ajax事件。ajaxStart()方法用于在Ajax请求发出前触发函数，ajaxStop()方法用于在Ajax请求完成后触发函数。它们的调用格式为：  **$(selector).ajaxStart(function())和$(selector).ajaxStop(function())** |