爱创课堂前端培训

# javascript进阶

第3天课堂笔记（本课程共5天）

班级：北京前端训练营27期

讲师：蔡久明

日期：2019年10月12日

爱创课堂官网 ：[www.icketang.com](http://www.icketang.com)

# 复习

call与apply

相同点：改变this的指向

不通点：参数不同

call多个参数

第一个参数代表this指向

第二个参数开始为调用函数时传入的参数

apply两个参数

第一个参数代表this指向

第二个参数，是一个数组，为调用函数时传入参数的数组

事件三要素：

事件对象、事件类型、事件处理程序

事件的流程：

捕获---处理----冒泡

事件捕获：事件从最外层元素向目标元素传递对的过程

事件处理：事件位于目标元素

事件冒泡：事件从目标元素向外层元素传递的过程

DOM0和DOM2

DOM0: onclick dom.ontype = fn 只有事件冒泡 取消事件 dom.ontype = null

DOM2:

高级浏览器：

绑定：dom.addEventListener（type, fn, bool）

dom:绑定的元素对象

type：事件类型

fn：事件处理函数

bool：默认值为false 表示发生的阶段 true表示捕获 false 表示冒泡

移除：dom.removeEventListener(type, fn, bool)  
 dom:绑定的元素对象

type：事件类型

fn：事件处理函数

bool：默认值为false 表示发生的阶段 true表示捕获 false 表示冒泡

注意：如果要移除事件，则在添加的时候，将处理函数提炼到全局，绑定和移除的时候都使用函数名

IE浏览器：

绑定:dom.attachEvent(ontype, fn)

dom:绑定的元素对象

type：事件类型

fn：事件处理函数

移除：dom.detachEvent（ontype,fn）

dom:绑定的元素对象

type：事件类型

fn：事件处理函数

DOM0和DOM2的区别：

1. DOM0同元素的同类型事件只能存在一个，后面的覆盖前面，DOM2可以添加多个
2. DOM2在高级浏览器中可以将事件处理设置为事件捕获阶段  
   DOM0和DOM2共存时，事件执行的顺序:

高级浏览器：按照绑定的顺序依次执行

IE浏览器：

1. DOM0级事件
2. DOM2绑定顺序的逆序执行

事件对象

事件在发生的时候，会场一个对象，该对象记录着事件发生的详细信息，这个对象称之为事件对象

阻止冒泡：

1. e.cancelBubble = true
2. e.stopPropagation()

阻止默认行为：

DOM0 ： return false

IE:

returnValue = false

return false

高级浏览器：

e.preventDefault()

事件对象获得尺寸

offsetX和offsetY； 鼠标位于元素内的位置，受子元素影响

clientX和clientY: 鼠标位于视口中的位置

pageX和pageY: 鼠标相对于网页顶点的位置

screenX和screenY:鼠标相对于屏幕的位置

# 快捷的尺寸

## 1.1 clientWidth 和 clientHeight

值： content + padding

结构：

|  |
| --- |
| 1. div { 2. width: 200px; 3. height: 200px; 4. border: 10px solid #000; 5. background-color: orange; 6. padding: 16px; 7. margin: 20px; 8. } |

执行代码：

|  |
| --- |
| 1. // 获得元素对象 2. var div = document.getElementById('box'); 3. // 获得尺寸 4. console.log(div.clientWidth, div.clientHeight); |

结果：

|  |
| --- |
|  |

## 1.2 offsetWidth 和 offsetHeight

值 = content + padding + border

结构：

|  |
| --- |
| 1. div { 2. width: 200px; 3. height: 200px; 4. border: 15px solid #000; 5. background-color: orange; 6. padding: 16px; 7. margin: 20px; 8. } |

执行代码：

|  |
| --- |
| 1. // 获得元素对象 2. var div = document.getElementById('box'); 3. // 获得尺寸 4. console.log(div.offsetWidth, div.offsetHeight); |

结果：

|  |
| --- |
|  |

## 1.3 clientLeft 和 clientTop

clientLeft: 左边框的宽度

clientTop: 上边框的宽度

结构：

|  |
| --- |
| 1. div { 2. width: 200px; 3. height: 200px; 4. border: 15px solid #000; 5. background-color: orange; 6. padding: 16px; 7. margin: 20px; 8. position: relative; 9. left: 10px; 10. top: 30px; 11. border-left: 40px solid red; 12. border-top: 55px solid blue; 13. } |

执行代码：

|  |
| --- |
| 1. // 获得元素对象 2. var div = document.getElementById('box'); 3. // 获得尺寸 4. console.log(div.clientLeft, div.clientTop); |

结果：

|  |
| --- |
|  |

## 1.4 jquery的快捷尺寸

盒模型：

|  |
| --- |
|  |

width和 height

|  |
| --- |
|  |

innerWidth和innerHeight

|  |
| --- |
|  |

outerWidth和outerHeight

默认值为false

|  |
| --- |
|  |

参数设置为true

|  |
| --- |
|  |

执行代码：

|  |
| --- |
| 1. // 获得元素 2. var $box = $('#box'); 3. /\* 4. width和height： content 5. width:元素的宽度 6. height:元素的高度 7. \*/ 8. console.log($box.width(), $box.height()); 9. /\* 10. innerHeight和innerWidth： content + padding 11. \*/ 12. console.log($box.innerWidth(), $box.innerHeight()); 13. /\* 14. outerWidth（false）和outerHeight（false） 默认值为false 15. false:content + padding + border 16. true:content + padding + border + margin 17. \*/ 18. console.log($box.outerWidth(), $box.outerHeight()); 19. console.log($box.outerWidth(true), $box.outerHeight(true)); |

结果：

|  |
| --- |
|  |

结论：

width和height： content

innerHeight和innerWidth： content + padding

outerWidth（false）和outerHeight（false） 默认值为false

false:content + padding + border

true:content + padding + border + margin

## 1.5 定位元素

offsetParent获得定位的父元素,定位父元素不一定是parentNode

结果：

|  |
| --- |
| 1. #carousel { 2. position: relative; 3. width: 200px; 4. height: 200px; 5. border: 1px solid red; 6. margin: 0 auto; 7. } 8. #box { 9. position: absolute; 10. left: 20px; 11. top: 30px; 12. width: 50px; 13. height: 50px; 14. background-color: orange; 15. } |

执行代码：

|  |
| --- |
| 1. <script> 2. // 获得box元素 3. var box = document.getElementById('box'); 4. // 获得定位的父元素 5. console.log(box.offsetParent); 6. console.log(box.parentNode); |

结果：

|  |
| --- |
|  |

## 1.6 offsetLeft和offsetTop

获得定位的相关信息: left 和 top

结构：

|  |
| --- |
|  |

执行代码：

|  |
| --- |
| 1. // 获得定位的相关信息 2. console.log(box.offsetLeft, box.offsetTop); |

结果：

|  |
| --- |
|  |

结论： 高级浏览器在计算left和top值的时候，不会将边框计算在内。但是有一些IE的版本会将border计算在内。

## 1.7 jquery中的相关定位

结构：

|  |
| --- |
|  |

执行代码：

|  |
| --- |
| 1. // 获得定位信息 2. console.log($('#box').position()); |

结果：

|  |
| --- |
|  |

## 1.8 获得元素到页面之间的距离

结构：

|  |
| --- |
|  |

执行代码：

|  |
| --- |
| 1. // 获得box到页面的距离 2. console.log($('#box').offset()); |

结果：

|  |
| --- |
|  |

需求：自定义实现jquery中的offset()

|  |
| --- |
|  |

## 1.9 IE

offsetLeft和offsetTop计算的时候，会计算当前元素到父元素之间的距离 + 父元素的边框

# 滚动条事件

## onscroll

滚动条滚动的事件，鼠标的滚轮、上下键、空格、PgUp、PgDn

执行代码：

|  |
| --- |
| document.onscroll = function() {  console.log('我被触发了');  } |

结果：

|  |
| --- |
|  |

## 获得页面滚动条的卷动值

垂直方向：document.documentElement.scrollTop;

水平方向：document.documentElement.scrollLeft;

执行代码：

|  |
| --- |
| 1. document.onscroll = function() { 2. console.log(document.documentElement.scrollTop); 3. console.log(document.documentElement.scrollLeft); 5. } |

结果：

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |