爱创课堂前端培训

jquery第五天

班级：爱创课堂十八期

讲师：彭帅伟

日期：2018年11月6日

# 函数节流

函数节流：可以对一些频繁触发的函数事件进行限制， 可以使函数在规定的时间内或者当满足某些条件之后再去执行。

设置一个变量 取值是布尔值

var lock = true 函数是可以执行的

lock = false 函数是不能被执行的

举例：

|  |
| --- |
| 1. // 定义锁 2. var lock = true; 3. function a() { 4. if (true) { 5. return; 6. } 7. // 关闭锁  lock = false; 8. console.log(123); 9. // 定义延时器 10. setTimeout(function() { 11. // 开锁 12. lock = true; 14. }, 3000) 16. } |

# 延时器

可以把延时器当做一个定时炸弹， 在规定的时间内只会执行一次

使用方式：

setTimeout(fn, time)

fn: 执行的函数

time: 要延迟的事件

举例：

|  |
| --- |
| 1. // 定义锁 2. var lock = true; 3. // 给btn添加点击事件 4. $("#btn").click(function() { 5. if (!lock) { 6. return; 7. } 8. // 进入函数中， 关闭锁 9. lock = false; 10. console.log("函数执行了"); 11. // 在s3之后， 开锁 12. setTimeout(function() { 13. console.log("延时器执行了"); 14. lock = true; 15. }, 3000) 16. }) |

# this

this表示触发事件的对象

each方法中接受一个函数， 函数中的this表示遍历到的该对象

函数中也有this, 当函数自执行的时候，指向window

举例：

|  |
| --- |
| 1. // 定义一个函数 2. function fun() { 3. console.log(this); 4. } 5. // 执行 6. fun(); |

输出：

|  |
| --- |
|  |

改变函数中的this指向， 可以使用call()方法

使用方式：

函数.call() 接受的参数就是要改变的this指向

举例：

|  |
| --- |
| 1. // 定义一个函数 2. function fun() { 3. console.log(this); 4. } 5. // 使用call方法改变函数执行时候的this指向 6. fun.call($("#btn")); 7. fun.call(document); 8. fun.call(document.body); |

输出：

|  |
| --- |
|  |

# each方法

each方法有两种方式：

第一种使用方式：

对象.each(function() {})

举例：

|  |
| --- |
| 1. <ul> 2. <li></li> 3. <li></li> 4. <li></li> 5. <li></li> 6. <li></li> 7. <li></li> 8. <li></li> 9. <li></li> 10. <li></li> 11. <li></li> 12. </ul> 13. <script type="text/javascript" src="js/jquery-1.12.3.min.js"></script> 14. <script type="text/javascript"> 15. // 使用第一种方式 16. $("li").each(function(i) { 17. console.log(i); 18. }) |

第二种使用方式：

一般用于遍历数组或者是对象的时候

$.each(target, fn)

target: 要遍历的目标

fn: 要执行的函数

函数中有两个参数

第一个参数： 索引值

第二个参数：成员值

举例：

|  |
| --- |
| 1. // 定义数组 2. var arr = ["a", "b", "c", "d"]; 3. // 遍历数组 4. $.each(arr, function() { 5. console.log(arguments); 6. }) |

|  |
| --- |
|  |

遍历对象：

|  |
| --- |
| 1. // 定义对象 2. var obj = { 3. a: 1, 4. b: 2, 5. c: 3 6. } 7. // 遍历对象 8. $.each(obj, function() { 9. console.log(arguments); 10. }) |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |