目录

[jquery 1](#_Toc7826)

[复习： 3](#_Toc19960)

[一、 each 5](#_Toc8546)

[二、 animate函数 7](#_Toc1856)

[2.1 概述 7](#_Toc26245)

[2.2 能够参与动画的属性 7](#_Toc17030)

[2.3 动画的执行顺序 8](#_Toc31685)

[三、 回调函数 9](#_Toc30768)

[3.1 delay 10](#_Toc6828)

[3.2 stop 12](#_Toc31602)

[下午复习： 13](#_Toc4645)

[四、 防止动画积累 14](#_Toc11753)

[五、呼吸轮播图 14](#_Toc23091)

[5.1 概述 14](#_Toc14595)

[六、安徽龙溪 15](#_Toc2617)

[七、呼吸轮播变种 15](#_Toc17303)

[7.1右按钮的点击事件 15](#_Toc28330)

[7.2 左按钮的点击事件 16](#_Toc3425)

# 复习：

jquery引包：

引入jquery文件的script标签与书写jquery语句的script标签不能是同一个

$函数：

当引入jquery之后， 向全局暴露的是$变量， 所有我们书写的所有jquery语句都是以$开头， 里面有很多方法， 我们可以连续打点使用

jquery对象：

js与jquery互相转换

js转为jquery对象： 直接使用$

jquery对象转为js对象： 直接使用[]

常用方法：

size: 获取元素的个数

用length属性代替了size方法

show: 是显示的意思， 只有当元素的状态是display: none时候才可以使用，最终状态是display: block

hide: 隐藏的意思， 与show相反

toggle: 切换

slideDown: 慢慢展看， 只有当元素的状态是dislpaly: none的时候才可以使用， 当调用该方法的时候元素的height瞬间变为0， 然后慢慢的移动化的形式变到元素的初始height

slideUp: 徐徐合起， 与slideDown相反

slideToggle: 切换  
  
 fadeIn: 淡入， 只有当元素的状态是dislpaly: none的时候才可以使用， 当调用该方法的时候元素的opacity变为0， 然后慢慢的变到opacity: 1, 最终的状态是display: block

fadeOut: 淡出， 与fadeIn相反

fadeToggle: 切换

fadeTo(600, .5): 第一个参数表示完成动画的时间， 第二个参数指定的透明度

html: 获取元素的内部文本，没有参数

可以设置元素的内部文本， 参数就是要设置的内容

可以添加节点

attr:

可以设置元素的原有属性

$(dom).attr(原有属性， 属性值)

还可以读取元素的属性

addClass: 添加类型

removeClass: 移除类名

节点操作：

this: 在js中表示触发事件的元素

在jquery中有三个对象是不需要添加双引号的

$(this)、 $(document) 、$(window)

parent: 会选中父节点

siblings: 会选中兄弟节点

children: 会选中所有的子节点

next: 会选中下一个兄弟节点

nextAll: 会选中后面所有的兄弟节点

prev: 上一个兄弟节点

prevAll: 全面所有的兄弟节点

parents: 祖先节点

find: 后代节点

序号的问题：

eq: 表示选中元素大队列的排名

index: 表示亲兄弟的排名， 它的值非常稳定

# each

each方法用于遍历、循环所匹配到的所有元素， 接收一个参数是函数， 函数中也有一个this，表示遍历到的该对象

举例：

|  |
| --- |
| 1. <div class="box"> 2. <p></p> 3. <p></p> 4. <p></p> 5. <p></p> 6. </div> 7. <div class="box"> 8. <p></p> 9. <p></p> 10. <p></p> 11. <p></p> 12. </div> 13. <div class="box"> 14. <p></p> 15. <p></p> 16. <p></p> 17. <p></p> 18. </div> 19. <div class="box"> 20. <p></p> 21. <p></p> 22. <p></p> 23. <p></p> 24. </div> |

样式：

|  |
| --- |
|  |

执行代码：

|  |
| --- |
| 1. $("div").each(function() { 2. // console.log($(this)); 3. // console.log(this); 4. $(this).children().eq(0).css("background-color", "red"); 5. }) |

结果：

|  |
| --- |
|  |

each方法用于遍历， 以后只要看到每一个元素需要做的事件， 首先要想到each方法

隔列变色：

|  |
| --- |
| 1. $("tr").each(function() { 2. $(this).children("td:odd").css("backgroundColor", "orange"); 3. $(this).children("td:even").css("backgroundColor", "skyblue"); 4. }) |

结果：

|  |
| --- |
|  |

each方法中的函数，有一个参数i， 表示遍历到的该对象的序号

|  |
| --- |
| 1. // i的含义 2. $("tr.teshu").each(function(i) { 3. console.log(this, i); 4. }) |

结果：

|  |
| --- |
|  |

# animate函数

## 2.1 概述

animate可以接收两个参数

第一个参数： json

第二个参数： 是完成动画的时间

animate是基于终点的动画

|  |
| --- |
| 1. // 我们想要让box运动到1000的位置 2. $(".box").animate({"left": 500}, 2000); |

结果：

|  |
| --- |
|  |

## 2.2 能够参与动画的属性

所有数值型的属性都能够参与动画

width

height

border

不能参与动画的属性

backgroundColor

backgroundPosition

Css的一些复杂属性（transform）

border-radius可以参与动画的

|  |
| --- |
| 1. // 不能够参与动画的属性: backgroundColor, backgroundPosition 2. $(".box").animate({ 3. "width": "200px", 4. "height": "200px", 5. "backgroundColor": "blue", 6. "border-radius": "50%", 7. "left": 200, 8. "top": 200 9. }, 2000) |

结果：

|  |
| --- |
|  |

## 2.3 动画的执行顺序

* 同一元素的不同animate

执行顺序： 按照代码的绑定顺序

|  |
| --- |
| 1. // 第一次动画 2. $(".box").animate({"left": 500}, 2000); ① 3. // 第二次动画 4. $(".box").animate({"top": 500}, 2000); ② 6. $(".box").animate({ ③ 7. "left": 500, 8. "top": 500 9. }, 2000) |

结果：

|  |
| --- |
|  |

* 不同元素的animate

执行顺序: 是同时执行

|  |
| --- |
| 1. // 分别为两个元素添加animate函数 2. $(".box").animate({"left": 500}, 2000); 3. $(".box1").animate({"left": 500}, 2000); |

结果：

|  |
| --- |
|  |

* animate和css

animate是需要花费时间的

css是瞬间完成

|  |
| --- |
| 1. // 我们希望当box运动完毕之后改变它的背景颜色 2. $(".box").animate({"left": 600}, 2000); 3. // 改变背景颜色 4. $(".box").css("background-color", "skyblue"); |

结果：当没有动画之前已经改变了颜色

|  |
| --- |
|  |

# 回调函数

异步语句： 类似于animate这种需要花费时间的语句，比如后面还有其他的js语句， 这些语句不会死等着animate执行完之后去执行

回调函数： 异步语句一般都有一个回调函数，当异步语句执行完毕之后要做的事情就是写在回调函数中

比如：我们要做一个肉菜，第一步将肉取出来

然后是将肉放入微波炉中加热，这个过程需要花费10min, 在这期间我们还可以摘菜、洗菜、烧水， 等到微波炉叮一声之后（叮一声之后要做的事情就是写在回调函数中）， 将肉取出，开始做肉菜，做肉菜的这个行为就是写在回调函数中

实际上animate函数接收三个参数

第一个参数： json

第二个参数: time 完成动画的时间 单位ms 省略不写

第三个参数： 回调函数

animate()、 show(1000)、 hide(1000)、slideDown()、 slideUp()、fadeIn()、fadeOut()

举例：

|  |
| --- |
| 1. // 回调函数 2. $(".box").animate({"left": 1000}, 2000, function() { 3. // console.log("动画完毕了"); 4. // 运动完毕改变背景颜色 5. $(this).css("background-color", "blue"); 6. }) 7. </sc |

在动画执行之前：

|  |
| --- |
|  |

动画执行之后：

|  |
| --- |
|  |

例如show也有一个回调函数

|  |
| --- |
| 1. // show 2. $(".box1").show(2000, function() { 3. // console.log("显示成功"); 4. // 改变颜色 5. $(this).css("background-color", "blue"); 6. }) |

## 3.1 delay

delay: 延迟， 只能够在动画之前使用，接收一个参数就是要延迟的时间

animate()、 show(1000)、hide(1000)、 slideDown()、slideUp()、 fadeIn()、 fadeOut()

|  |
| --- |
| 1. // 当点击btn的时候让box延迟2s执行动画 2. $("#btn").click(function() { 3. $(".box").delay(2000).animate({"left": 500}, 1000); 4. }) |

结果：当点击了btn之后2s开始执行动画

|  |
| --- |
|  |

例如show也可以使用delay方法

|  |
| --- |
| 1. // show 2. $("#btn").click(function() { 3. $(".box1").delay(2000).show(2000, function() { 4. // 改变背景颜色 5. // console.log(this); 6. $(this).css("background-color", "red"); 7. }) 8. }) |

利用delay可以控制元素入场的顺序

|  |
| --- |
| 1. // 我们希望所有的p元素可以每500ms入场 2. $("p").each(function(i) { 3. // console.log(this); 4. // console.log(i); 5. // 利用delay 6. $(this).delay(1000 \* i).show(0); 7. }) |

结果：

|  |
| --- |
|  |

## 3.2 stop

该方法用于让运动的元素立即停止动画， 接收两个参数， 都是布尔值， 默认是fasle

第一个参数： 表示是否清空动画队列

第二个参数： 表示是否立即完成动画

|  |
| --- |
| 1. stop(false, false): 等价于stop() 立即进入下一个动画，立即停止当前动画 2. stop(false, true)： 立即进入下一个动画， 立即完成并停止当前动画 3. stop(true, true)： 清空当前动画队列， 立即完成并停止当前动画 4. stop(true, false): 等价于stop(true) 清空当前动画队列， 立即停止当前动画 |

|  |
| --- |
| 1. // stop(false, false) 2. $("#btn").click(function() { 3. $(".box").stop(); 4. }) 5. // stop(false, true) 6. $("#btn1").click(function() { 7. $(".box").stop(false, true); 8. }) 9. // stop(true, true) 10. $("#btn2").click(function() { 11. $(".box").stop(true, true); 12. }) 13. // stop(true, fasle) 14. $("#btn3").click(function() { 15. $(".box").stop(true); 16. }) |

## 下午复习：

each方法： 用于遍历、循环所匹配带的每一个元素，接收一个参数是函数， 函数中有一个参数i， 表示遍历到的该对象的序号， 函数中的this表示遍历到的该对象

animate函数：

接收三个参数

第一个参数： json

第二个参数： time

第三个参数： 回调函数

能够参与动画的属性：

所有数值型的属性都能够参与动画

不能够参与动画的属性：

backgroundColor

backgroundPosition

css3复杂属性 （transform）

border-radius可以参与动画

回调函数：

异步一般都有一个回调函数

异步语句执行完毕之后要做的事件就是写在回调函数中

delay: 表示延迟，只能书写在动画之前

stop: 用于执行动画的元素立即停止动画， 接收两个参数 ，都是布尔值 默认是false

第一个参数： 是否情况当前动画队列

第二个参数： 是否立即完成当前动画

stop(false, false): 等价方式stop(), 立即进入下一个动画， 立即停止当前动画

stop(false, true): 立即进入下一个动画， 立即完成并停止当前动画

stop(true, true): 情况当前动画队列，立即完成并停止当前动画

stop(true, false): 等价方式 stop(true) 清空当前动画队列， 立即停止当前动画

# 防止动画积累

一个元素的身上可能存在多个动画事件， 尤其是在不经意间添加上的，此时， 应该进行防止(防流氓)

比如： 有一支前往广州的军队， 在去往广州的路上，又接到命令，回到北京

此时， 这支队友有两个策略：

第一个策略： 放弃原有任务， 立即执行新的任务

利用stop(true)来实现

|  |
| --- |
| 1. //注册点击事件 2. $("#btn1").click(function() { 3. $(".box").stop(true).animate({"left": 1000}, 2000); 4. }) 5. // 为btn注册点击事件 6. $("#btn").click(function() { 7. $(".box").stop(true).animate({"left": 50}, 2000); 8. }) |

第二个策略：

只有当元素不运动的时候， 才接受新的任务

is(): 表示元素是否具备某种状态

|  |
| --- |
| 1. is(“:animated”): 当元素运动的时候返回true, 当元素不运动的时候返回false |

|  |
| --- |
| 1. // 需求: 当点击btn1的时候让元素去到1000的位置， 当点击btn的时候让元素回到50的位置 2. //注册点击事件 3. $("#btn1").click(function() { 4. $(".box").stop(true).animate({"left": 1000}, 2000); 5. }) 6. // 为btn注册点击事件 7. $("#btn").click(function() { 8. $(".box").stop(true).animate({"left": 50}, 2000); 9. }) |

# 五、呼吸轮播图

## 5.1 概述

轮播图：carousel

呼吸轮播变种布局重点：所有的图片摞一起。

jquery选择元素的能力非常好，但是我们习惯将用到的元素，提前保存到变量。通常我们使用id选中元素。一般我们以$box。

左右按钮防流氓的策略：当图片运动时，不进行任何操作。is()

小圆点的防流氓的策略：立即响应新事件。stop(true)

# 六、安徽龙溪

定时器就是右按钮点击事件

|  |
| --- |
| 1. // 定义函数， 等价于右按钮点击事件 2. function rightBtn() { 3. // 防流氓 4. if ($imgs.is(":animated")) { 5. // 什么也不做 6. return; 7. } 8. // 当前图片淡出 9. $imgs.eq(idx).fadeOut(600); 10. // 改变信号量 11. idx++; 12. // 边界判断 13. if (idx > length - 1) { 14. // 让idx变为0 15. idx = 0; 16. } 17. // 下一张图片要淡入 18. $imgs.eq(idx).fadeIn(600); 19. // 对应信号量的小圆点要改变 20. $circles.eq(idx).addClass("cur").siblings().removeClass("cur"); 21. } |

# 七、呼吸轮播变种

## 7.1右按钮的点击事件

当老图完全淡出之后新图再淡入。使用回调函数。

|  |
| --- |
| 1. // 右按钮点击事件 2. $rightBtn.click(function() { 3. // 防流氓 4. if ($imgs.is(":animated")) { 5. // 什么也不做 6. return; 7. } 8. // 当前图片淡出 9. $imgs.eq(idx).fadeOut(600, function() { 10. console.log("淡出完毕"); 11. // 当图片完全淡出之后，改变信号量 12. // 改变信号量 13. idx++; 14. // 边界判断 15. if (idx > length - 1) { 16. idx = 0; 17. } 18. // 下一张图片要进入 19. $imgs.eq(idx).fadeIn(600); 20. // 对应信号量的小圆点加上cur 21. $circles.eq(idx).addClass("cur").siblings().removeClass("cur"); 22. }); 23. }) |

## 7.2 左按钮的点击事件

|  |
| --- |
| 1. $leftBtn.click(function() { 2. // 防流氓 3. if (!$imgs.is(":animated")) { 4. // 当前图片淡出 5. $imgs.eq(idx).fadeOut(600, function() { 6. // console.log("淡出成功"); 7. // 当图片完全淡出之后， 验证信号量 8. idx--; 9. // 边界判断 10. if (idx < 0) { 11. idx = length - 1; 12. } 13. // 下一张图片要进入 14. $imgs.eq(idx).fadeIn(600); 15. // 对应信号量的小圆点要加上cur 16. $circles.eq(idx).addClass("cur").siblings().removeClass("cur"); 17. }) 18. } 19. }) |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |