动画效果

学习要点：

1.显示、隐藏

2.滑动、卷动

3.淡入、淡出

4.自定义动画

5.列队动画方法

6.动画相关方法

7.动画全局属性

在以前很长一段时间里，网页上的各种特效还需要采用 flash在进行。但最近几年里，

我们已经很少看到这种情况了，绝大部分已经使用 JavaScript动画效果来取代 flash。这里

说的取代是网页特效部分，而不是动画。网页特效比如：渐变菜单、渐进显示、图片轮播等；

而动画比如：故事情节广告、MV等等。

一．显示、隐藏

jQuery中显示方法为：.show()，隐藏方法为：.hide()。在无参数的时候，只是硬性的显

示内容和隐藏内容。

//显示

$('.show').click(function () {

$('#box').show();

});

//隐藏

$('.hide').click(function () {

$('#box').hide();

});

注意：.hide()方法其实就是在行内设置 CSS代码：display:none;而.show()方法要根据原

来元素是区块还是内联来决定，如果是区块，则设置 CSS代码：display:block;如果是内联，

则设置 CSS代码：display:inline;。

在.show()和.hide()方法可以传递一个参数，这个参数以毫秒(1000毫秒等于 1秒钟)来控

制速度。并且里面富含了匀速变大变小，以及透明度变换。

$('.show').click(function () {

$('#box').show(1000);//显示用了 1秒

});

$('.hide').click(function () {

$('#box').hide(1000);//隐藏用了 1秒

});

除了直接使用毫秒来控制速度外，jQuery还提供了三种预设速度参数字符串：slow、

normal和 fast，分别对应 600毫秒、400毫秒和 200毫秒。

$('.show').click(function () {

$('#box').show('fast');//200毫秒

});

$('.hide').click(function () {

$('#box').hide('slow');//600毫秒

});

注意：不管是传递毫秒数还是传递预设字符串，如果不小心传递错误或者传递空字符串。

那么它将采用默认值：400毫秒。

$('.show').click(function () {

$('#box').show('');//默认 400毫秒

});

//使用.show()和.hide()的回调函数，可以实现列队动画效果。

$('.show').click(function () {

$('#box').show('slow', function () {

alert('动画持续完毕后，执行我！');

});

});

//列队动画，使用函数名调用自身

$('.show').click(function () {

$('div').first().show('fast', function showSpan() {$(this).next().show('fast', showSpan);

});

});

//列队动画，使用 arguments.callee匿名函数自调用

$('.hide').click(function () {

$('div').last().hide('fast', function() {

$(this).prev().hide('fast', arguments.callee);

});

});

我们在使用.show()和.hide()的时候，如果需要一个按钮切换操作，需要进行一些条件判

断。而 jQuery提供给我们一个类似功能的独立方法：.toggle()。

$('.toggle').click(function () {

$(this).toggle('slow');

});

二．滑动、卷动

jQuery提供了一组改变元素高度的方法：.slideUp()、.slideDown()和.slideToggle()。顾名

思义，向上收缩(卷动)和向下展开(滑动)。

$('.down').click(function () {

$('#box').slideDown();

});

$('.up').click(function () {

$('#box').slideUp();

});

$('.toggle').click(function () {

$('#box').slideToggle();

});

注意：滑动、卷动效果和显示、隐藏效果一样，具有相同的参数。

三．淡入、淡出

jQuery提供了一组专门用于透明度变化的方法：.fadeIn()和.fadeOut()，分别表示淡入、

淡出，当然还有一个自动切换的方法：.fadeToggle()。

$('.in').click(function () {

$('#box').fadeIn('slow');

});

$('.out').click(function () {

$('#box').fadeOut('slow');

});

$('.toggle').click(function () {

$('#box').fadeToggle();

});

上面三个透明度方法只能是从 0到 100，或者从 100到 0，如果我们想设置指定值就没

有办法了。而 jQuery为了解决这个问题提供了.fadeTo()方法。

$('.toggle').click(function () {

$('#box').fadeTo('slow', 0.33);//0.33表示值为 33

});

注意：淡入、淡出效果和显示、隐藏效果一样，具有相同的参数。对于.fadeTo()方法，

如果本身透明度大于指定值，会淡出，否则相反。

四．自定义动画

jQuery提供了几种简单常用的固定动画方面我们使用。但有些时候，这些简单动画无法

满足我们更加复杂的需求。这个时候，jQuery提供了一个.animate()方法来创建我们的自定

义动画，满足更多复杂多变的要求。

$('.animate').click(function () {

$('#box').animate({

'width' : '300px',

'height' : '200px',

'fontSize' : '50px',

'opacity' : 0.5

});

});

注意：一个 CSS变化就是一个动画效果，上面的例子中，已经有四个 CSS变化，已经

实现了多重动画同步运动的效果。

必传的参数只有一个，就是一个键值对 CSS变化样式的对象。还有两个可选参数分别

为速度和回调函数。

$('.animate').click(function () {

$('#box').animate({

'width' : '300px',

'height' : '200px'

}, 1000, function () {

alert('动画执行完毕执行我！');

});

});

到目前位置，我们都是创建的固定位置不动的动画。如果想要实现运动状态的位移动画，

那就必须使用自定义动画，并且结合 CSS的绝对定位功能。

$('.animate').click(function () {

$('#box').animate({

'top' : '300px',

'left' : '200px'

//先必须设置 CSS绝对定位

});

});

自定义动画中，每次开始运动都必须是初始位置或初始状态，而有时我们想通过当前位

置或状态下再进行动画。jQuery提供了自定义动画的累加、累减功能。

$('.animate').click(function () {

$('#box').animate({

'left' : '+=100px',

});

});

自定义实现列队动画的方式，有两种：1.在回调函数中再执行一个动画；2.通过连缀或

顺序来实现列队动画。

//通过依次顺序实现列队动画

$('.animate').click(function () {

$('#box').animate({'left' : '100px'});

$('#box').animate({'top' : '100px'});

$('#box').animate({'width' : '300px'});

});

注意：如果不是同一个元素，就会实现同步动画

//通过连缀实现列队动画

$('.animate').click(function () {

$('#box').animate({

'left' : '100px'

}).animate({

'top' : '100px'

}).animate({

'width' : '300px'

});

});

//通过回调函数实现列队动画

$('.animate').click(function () {

$('#box').animate({

'left' : '100px'

}, function () {

$('#box').animate({

'top' : '100px'

}, function () {

$('#box').animate({

'width' : '300px'

});

});

});

});

五．列队动画方法

之前我们已经可以实现列队动画了，如果是同一个元素，可以依次顺序或连缀调用。如

果是不同元素，可以使用回调函数。但有时列队动画太多，回调函数的可读性大大降低。为

此，jQuery提供了一组专门用于列队动画的方法。

//连缀无法实现按顺序列队

$('#box').slideUp('slow').slideDown('slow').css('background', 'orange');

注意：如果动画方法，连缀可以实依次列队，而.css()方法不是动画方法，会在一开始

传入列队之前。那么，可以采用动画方法的回调函数来解决。

//使用回调函数，强行将.css()方法排队到.slideDown()之后

$('#box').slideUp('slow').slideDown('slow', function () {

$(this).css('background', 'orange');

});

但如果这样的话，当列队动画繁多的时候，可读性不但下降，而原本的动画方法不够清

晰。所以，我们的想法是每个操作都是自己独立的方法。那么 jQuery提供了一个类似于回

调函数的方法：.queue()。

//使用.queue()方法模拟动画方法跟随动画方法之后

$('#box').slideUp('slow').slideDown('slow').queue(function () {

$(this).css('background', 'orange');

});

现在，我们想继续在.queue()方法后面再增加一个隐藏动画，这时发现居然无法实现。

这是.queue()特性导致的。有两种方法可以解决这个问题，jQuery的.queue()的回调函数可以

传递一个参数，这个参数是 next函数，在结尾处调用这个 next()方法即可再连缀执行列队动

画。

//使用 next参数来实现继续调用列队动画

$('#box').slideUp('slow').slideDown('slow').queue(function (next) {

$(this).css('background', 'orange');

next();

}).hide('slow');

因为next函数是jQuery1.4版本以后才出现的，而之前我们普遍使用的是.dequeue()方法。

意思为执行下一个元素列队中的函数。

//使用.dequeue()方法执行下一个函数动画

$('#box').slideUp('slow').slideDown('slow').queue(function () {

$(this).css('background', 'orange');

$(this).dequeue();

}).hide('slow');

如果采用顺序调用，那么使用列队动画方法，就非常清晰了，每一段代表一个列队，而

回调函数的嵌套就会杂乱无章。

//使用顺序调用的列队，逐个执行，非常清晰

$('#box').slideUp('slow');

$('#box').slideDown('slow');

$('#box').queue(function () {

$(this).css('background', 'orange');

$(this).dequeue();

})

$('#box').hide('slow');

.queue()方法还有一个功能，就是可以得到当前动画个列队的长度。当然，这个用法在

普通 Web开发中用的比较少，我们这里不做详细探讨。

//获取当前列队的长度，fx是默认列队的参数

function count() {

return $("#box").queue('fx').length;

}

//在某个动画处调用

$('#box').slideDown('slow', function () {alert(count());});

jQuery还提供了一个清理列队的功能方法：.clearQueue()。把它放入一个列队的回调函

数或.queue()方法里，就可以把剩下为执行的列队给移除。

//清理动画列队

$('#box').slideDown('slow', function () {$(this).clearQueue()});

六．动画相关方法

很多时候需要停止正在运行中的动画，jQuery为此提供了一个.stop()方法。它有两个可

选参数：.stop(clearQueue, gotoEnd)；clearQueue传递一个布尔值，代表是否清空未执行完的

动画列队，gotoEnd代表是否直接将正在执行的动画跳转到末状态。

//强制停止运行中的

$('.stop').click(function () {

$('#box').stop();

});

//带参数的强制运行

$('.animate').click(function () {

$('#box').animate({

'left' : '300px'

}, 1000);

$('#box').animate({

'bottom' : '300px'

}, 1000);

$('#box').animate({

'width' : '300px'

}, 1000);

$('#box').animate({

'height' : '300px'

}, 1000);

});

$('.stop').click(function () {

$('#box').stop(true ,true);

});

注意：第一个参数表示是否取消列队动画，默认为 false。如果参数为 true，当有列队动

画的时候，会取消后面的列队动画。第二参数表示是否到达当前动画结尾，默认为 false。

如果参数为 true，则停止后立即到达末尾处。

有时在执行动画或列队动画时，需要在运动之前有延迟执行，jQuery为此提供了.delay()

方法。这个方法可以在动画之前设置延迟，也可以在列队动画中间加上。

//开始延迟 1秒钟，中间延迟 1秒

$('.animate').click(function () {

$('#box').delay(1000).animate({

'left' : '300px'

}, 1000);

$('#box').animate({

'bottom' : '300px'

}, 1000);

$('#box').delay(1000).animate({

'width' : '300px'

}, 1000);

$('#box').animate({

'height' : '300px'

}, 1000);

});

在选择器的基础章节中，我们提到过一个过滤器:animated，这个过滤器可以判断出当前

运动的动画是哪个元素。通过这个特点，我们可以避免由于用户快速在某个元素执行动画时，

由于动画积累而导致的动画和用户的行为不一致。

//递归执行自我，无线循环播放

$('#box').slideToggle('slow', function () {

$(this).slideToggle('slow', arguments.callee);

});

//停止正在运动的动画，并且设置红色背景

$('.button').click(function(){

$('div:animated').stop().css('background', 'red');

});

六．动画全局属性

jQuery提供了两种全局设置的属性，分别为：$.fx.interval，设置每秒运行的帧数；$.fx.off，

关闭页面上所有的动画。

$.fx.interval属性可以调整动画每秒的运行帧数，默认为 13毫秒。数字越小越流畅，但

可能影响浏览器性能。

//设置运行帧数为 1000毫秒

$.fx.interval = 1000;

//默认为 13

$('.button').click(function () {

$('#box').toggle(3000);

});

$.fx.off属性可以关闭所有动画效果，在非常低端的浏览器，动画可能会出现各种异常

问题导致错误。而 jQuery设置这个属性，就是用于关闭动画效果的。

//设置动画为关闭 true

$.fx.off = true;

//默认为 false

补充：在.animate()方法中，还有一个参数，easing运动方式，这个参数，大部分参数值

需要通过插件来使用，在后面的课程中，会详细讲解。自带的参数有两个：swing(缓动)、

linear(匀速)，默认为 swing。

$('.button').click(function () {

$('#box').animate({

left : '800px'

}, 'slow', 'swing');

$('#pox').animate({

left : '800px'

}, 'slow', 'linear');

});