# 通过animate制作简单动画效果

javascript实现动画的原理跟动画片的制作一样.动画片是把一些差距不大的原画以一定帧数播放.js动画是靠连续改变元素的某个css属性值,比如left, top.达到视觉的动画效果.如果是靠js来实现一个动画其实不是很复杂，但是一个动画需要有一堆的兼容与限制条件，所以jQuery 通过 animate 这个方法把动画的操作给包装简单化了，我们只需要自定义配置好一些参数就可以了

**简单动画**

通过animate方法让一个元素先动起来，我们先看一个最简单的用法

在一个HTML放一个按钮，点击按钮后，对应的这个红色区块就会发动动画变化

我们给HTML附上一些初始化的值，首先我们要知道动画要动起来必备的一些条件，元素的布局是需要定位处理的，这样等于把元素单独拿出来做一个动画的处理，所以就需要赋予position属性，值可以是absolute或者relative，然后可以根据请客赋予元素在页面的坐标top或者left

#book{

width: 100px;

height: 150px;

position: relative;

background: red;

left: 5px;

}

通过调用animate方法，设定一个时间（2秒），我们在规定的时间内改变元素的宽度、高度跟坐标left 的值

$('#book').animate({

left : '100px',

width :'200px',

height :'300px'

}, 2000);

我们测试下这个代码，元素就会神奇的执行了，所以通过jQuery中设置一个动画就是这么简单的事情

**语法分析**

animate方法有2种参数处理，第一种是比较常规的配置，第二种就是相对比较复杂点，我们先熟悉第一种动画处理。注意：中括号[]的意思就是选项是可选择性的)

1 .animate( properties [, duration ] [, easing ] [, complete ] )

2 .animate( properties, options )

通过上面简单动画案例我们发现animate方法中

第一个是传递一个对象properties，这个对象包含着我们需要做动画改变的属性

第二个参数就是一个时间[, duration ]（时间是采用的毫秒计算）

其格式如下：

.animate( properties [, duration ] )

那么中括号其实是可选的，意味着这个时间其实也不是必须的，如果不传递jQuery内部用取一个默认的时间400毫秒，

第三个参数[, easing ] 表示过渡使用哪种缓动函数，简单的来说就是一个变化的算法，easing的作用可以让元素在不同的时间内产生的距离有变化。比如我们让一个元素在2秒钟运行100px，那么我们可以在第一秒运行80px 剩下一秒运行20px，jQuery自身提供"linear" 和 "swing"

第四个参数[, complete ] 这个也是一个可选项，我们一般在做动画的时候，动画完成之后我们就需要干点别的事情，这个就是一个告诉你动画啥时候会完成的方法了

**参数[, complete ] - 回调函数的作用**

我们知道这几个参数的作用了，我们继续实现一个稍微强化一点的效果，在动画结束之后给页面新增一个文字提示

$('#book').animate({

left :'100',

width :'200',

height :'300px'

},2000,'linear',function(){

$('body').append('<div>动画执行完毕</div>')

});

效果跟之前的区别就是加了'linear'与complete函数了，那么这个动画结束之后我们页面就会出现一个文字的div元素。

**动画参数的自动增减的用法**

动画属性也可以是一个相对值。如果提供一个以+= 或 -=开始的值，那么目标值就是以这个属性的当前值加上或者减去给定的数字来计算的。

# book {

left:50px

}

$('#book').animate({

left :'100'

}

这个意思就是设置id为book的元素移动到left为100的位置，那么实际上这个元素只移动了50px就足够了，因为元素本身是left为50px的初始化位置

设置+= 或 -=

$('#book').animate({

Left :'+=100' //在50px的位置上再移动100px

}

这个意思就是在元素本身的位置上在移动100px，所以实际上的位置是 50+100 = 150px，递减也是一样，在本身的元素的偏移量上在减去设置的值

**动画的快捷方式**

除了给动画指定的变化属性外，jQuery还提供了几个便携的参数，分别是'show', 'hide', 和 'toggle'

简单的来说这个几个参数的作用就是直接指定元素的最终的状态

#book{

width : 200px;

height : 200px;

position : relative;

background : red;

opacity :1;

}

开始给元素设置一个宽高，然后给一个opacity的透明度是1（默认就是1）

$('#book').animate({

opacity : 'hide',

width : 'hide',

height : 'toggle'

});

那么这个代码的意思就是，把元素的opacity改为0，width改为0，height改为0，

所以show与hide比较容易理解，就是显示与隐藏，toggle是一个相对值，就是相对元素的初始状态，如果开始显示那么通过toggle的效果就是隐藏，反之亦然

**多个连续动画**

如果我们要做不同元素的连续动画，我们就可以在一个动画结束之后调用另一个动画了

**1 同一个元素的连续动画**

$('#book1').animate({

left: '100px'

}, 1000)

$('#book1').animate({

width :'200px',

height : '200px'

}, 3000)

这里设置了2个动画，因为2个动画是有不同的时间的，所以我们不能写在一个animate方法里面了，针对同一个元素上的动画，所以我们需要针对针对left的移动， width与height的改变来写2个animate方法，第一个执行完毕后第二个又会开始执行了

jQuery的提倡链式的写法，所以我们可以合并方法调用

$('#book1')

.animate({

left: '100px'

}, 1000)

.animate({

width: '200px',

height: '200px'

}, 3000)

**2 不同元素的连续动画**

很多时候，我们设置一组元素动画完成了，又想接着有一组元素的动画开始，所以这时候就需要用到**[, complete ]** 这个参数了，我们要监听一个动画什么时候结束的

$('#book1').animate({

left: '+=100px'

}, 1000, function() {

//#book1元素的动画结束后，#book2的动画才开始

$('#book2').animate({

left: '+=100px',

}, 2000)

})

这样就可以形成多个不同元素的连续动画