# 学习目标：

1. 能够使用jQuery的3种动画效果
2. 能够使用jQuery对象的遍历方法
3. 能够使用jQuery全局的遍历方法
4. 能够使用jQuery3.0的for-of遍历方法
5. 能够使用jQuery的绑定与解绑方法
6. 能够使用jQuery的完成案例-广告的显示与隐藏
7. 能够使用jQuery的完成案例-抽奖程序
8. 能够理解jQuery的插件机制及其相关API
9. 能够使用jQuery的表单校验插件进行表单校验
10. 能够使用表单校验插件的方法进行自定义校验规则

# 反馈

|  |
| --- |
| JQuery的加载事件里面写的方法要怎么写？  加载事件一般不写定义函数，建议写在加载事件外部，否则之后加载事件内部可用。  很多东西都需要在服务器启动的时候加载，具体有哪些？  Spring的配置文件，  过滤器，  ServletContext域，  ServletContextListener 监听器（监听域对象）  Web.xml  Servlet也可以在启动时加载（默认servlet是用户第一访问的时候），可以改变为启动时加载，只要注解设置loadOnStartup =1  Tomcat加载类和资源的顺序为（以项目应用为例）：  1、web项目 （/webapps/web项目 /下的web.xml，classes，lib子目录中\*.calss及\*.jar，仅当前项目可以加载）  2、Bootstrap （$JAVA\_HOME/jre/lib/ext/\*.jar）用于加载资源使用  3、System （$CLASSPATH/\*.class和CLASSPATH中指定的jar）  4、lib （$CATALINA\_HOME/lib/下的classes，lib， \*.class及\*.jar） |
| JQ这个不好消化啊.是要全部都记住 还是说能记多少是多少  我讲的都要记，利用好的复习方法进行有规律进行记忆。 |
| 宇哥，笔记中获取/设置css样式中方法：css(name[,value]) 这个方法好像是错误的，这里的css应该是attr(name[,value])吧  css(name[,value]) 设置样式的，具体设置标签内style样式的  attr(“style”,”background-color:red”); |
| crud为什么不写测试类啊，不应该后台测试没问题才写前台吗？  写测试是一个好的习惯，当代码足够复杂时建议写测试类。 |
| 如果要联合增删改查两张表怎么实现(一张主表,一张从表)?  查，多表查询，表连接，sql语句搞定。。。。List<Map<String,Object>>,项目实战第2天就会使用。  增、删、改。多表操作-----执行时是多条sql语句，使用事务进行控制，项目实战第3天就会使用。  数据库以后还会学吗?  学oracle  触发器什么的还会讲不?  会将的，将oracle的时候就会pl-sql编程 |

# jQuery动画

## 默认显示隐藏动画效果

### 方法

| **方法名称** | **解释** |
| --- | --- |
| show([speed,[easing],[fn]]) | 显示元素方法 |
| hide([speed,[easing],[fn]]) | 隐藏元素方法 |
| toggle([speed],[easing],[fn]) | 切换元素方法，显示的使之隐藏，隐藏的使之显示 |

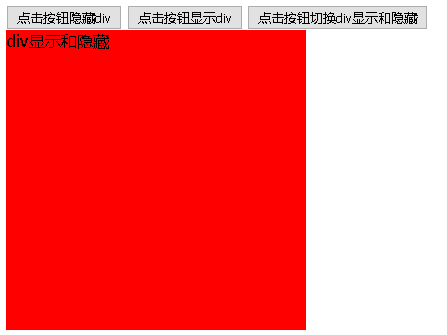
### 参数

| **参数名称** | **解释** |
| --- | --- |
| speed | 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000) |
| easing | 用来指定切换效果，默认是"swing"，可用参数"linear" |
| fn | 在动画完成时执行的函数，每个元素执行一次 |

### 示例

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Insert title here</**title**>  <**script type="text/javascript" src="../js/jquery-3.3.1.min.js"**></**script**>  <**script type="text/javascript"**>  */\*  \* jquery动画，主要是实现标签的显示与隐藏的动画方式不同  \* 1.基本动画  \* 2.滑动动画  \* 3.淡入与淡出  \* \*/   /\*  \* 基本动画  \* show(speed,fn); 显示动画  \* speed，动画执行的速度，这个参数有2种写法  \* 第一种：单词写法，slow/normal/fast  \* 第二种：毫秒数，3000，动画执行从开始到结束需要3秒  \* fn，动画执行完成后触发的函数，叫做回调函数  \* hide(speed,fn); 隐藏动画  \* toggle(speed,fn);切换动画  \* \*/   //隐藏* **function** *hideFn*() {  **$**(**"#showDiv"**).hide(2000);  }  *//显示* **function** *showFn*() {  **$**(**"#showDiv"**).show(2000);  }  *//切换显示与隐藏* **function** *toggleFn*() {  **$**(**"#showDiv"**).toggle(2000);  }    </**script**> </**head**> <**body**> <**input type="button" value="点击按钮隐藏div" onclick="***hideFn*()**"**> <**input type="button" value="点击按钮显示div" onclick="***showFn*()**"**> <**input type="button" value="点击按钮切换div显示和隐藏" onclick="***toggleFn*()**"**>  <**div id="showDiv" style="width**:300**px**;**height**:300**px**;**background**:**red"**>  div显示和隐藏 </**div**> </**body**> </**html**> |

### 示例效果图



## 滑动效果

### 方法

| **方法名称** | **解释** |
| --- | --- |
| slideDown([speed,[easing],[fn]]) | 向下滑动方法 |
| slideUp([speed,[easing],[fn]]) | 向上滑动方法 |
| slideToggle([speed],[easing],[fn]) | 切换元素方法，显示的使之隐藏，隐藏的使之显示 |

| **参数名称** | **解释** |
| --- | --- |
| speed | 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000) |
| easing | 用来指定切换效果，默认是"swing"，可用参数"linear" |
| fn | 在动画完成时执行的函数，每个元素执行一次 |

### 参数

### 示例

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Insert title here</**title**>  <**script type="text/javascript" src="../js/jquery-3.3.1.min.js"**></**script**>  <**script type="text/javascript"**>  */\*  \* jquery动画，主要是实现标签的显示与隐藏的动画方式不同  \* 1.基本动画  \* 2.滑动动画  \* 3.淡入与淡出  \* \*/   /\*  \* 滑动动画  \* slideDown(speed,fn); 向下展开滑动显示动画  \* speed，动画执行的速度，这个参数有2种写法  \* 第一种：单词写法，slow/normal/fast  \* 第二种：毫秒数，3000，动画执行从开始到结束需要3秒  \* fn，动画执行完成后触发的函数，叫做回调函数  \* slideUp(speed,fn); 向上收起滑动隐藏动画  \* slideToggle(speed,fn);切换滑动动画实现显示与隐藏  \* \*/   //隐藏* **function** *hideFn*() {  **$**(**"#showDiv"**).**slideUp**(2000);  }  *//显示* **function** *showFn*() {  **$**(**"#showDiv"**).**slideDown**(2000);  }  *//切换显示与隐藏* **function** *toggleFn*() {  **$**(**"#showDiv"**).**slideToggle**(2000);  }    </**script**> </**head**> <**body**> <**input type="button" value="点击按钮隐藏div" onclick="***hideFn*()**"**> <**input type="button" value="点击按钮显示div" onclick="***showFn*()**"**> <**input type="button" value="点击按钮切换div显示和隐藏" onclick="***toggleFn*()**"**>  <**div id="showDiv" style="width**:300**px**;**height**:300**px**;**background**:**red"**>  div显示和隐藏 </**div**> </**body**> </**html**> |

### 示例效果图



## 淡入淡出效果

### 方法

| **方法名称** | **解释** |
| --- | --- |
| fadeIn([speed,[easing],[fn]]) | 淡入显示方法 |
| fadeOut([speed,[easing],[fn]]) | 淡出隐藏方法 |
| fadeToggle([speed],[easing],[fn]) | 切换元素方法，显示的使之隐藏，隐藏的使之显示 |

### 参数

| **参数名称** | **解释** |
| --- | --- |
| speed | 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000) |
| easing | 用来指定切换效果，默认是"swing"，可用参数"linear" |
| fn | 在动画完成时执行的函数，每个元素执行一次 |

### 示例

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Insert title here</**title**>  <**script type="text/javascript" src="../js/jquery-3.3.1.min.js"**></**script**>  <**script type="text/javascript"**>  */\*  \* jquery动画，主要是实现标签的显示与隐藏的动画方式不同  \* 1.基本动画  \* 2.滑动动画  \* 3.淡入与淡出  \* \*/   /\*  \* 淡入与淡出，就是控制元素的透明度，执行的是底层css有一个控制透明度的样式opacity  \* fadeIn(speed,fn); 淡入显示动画  \* speed，动画执行的速度，这个参数有2种写法  \* 第一种：单词写法，slow/normal/fast  \* 第二种：毫秒数，3000，动画执行从开始到结束需要3秒  \* fn，动画执行完成后触发的函数，叫做回调函数  \* fadeOut(speed,fn); 淡出隐藏动画  \* fadeToggle(speed,fn);切换淡入淡出动画实现显示与隐藏  \* \*/   //隐藏* **function** *hideFn*() {  **$**(**"#showDiv"**).**fadeOut**(3000,**function** () {  *alert*(**"隐藏动画执行完毕"**);  });  }  *//显示* **function** *showFn*() {  **$**(**"#showDiv"**).**fadeIn**(3000);  }  *//切换显示与隐藏* **function** *toggleFn*() {  **$**(**"#showDiv"**).**fadeToggle**(**"fast"**);  }    </**script**> </**head**> <**body**> <**input type="button" value="点击按钮隐藏div" onclick="***hideFn*()**"**> <**input type="button" value="点击按钮显示div" onclick="***showFn*()**"**> <**input type="button" value="点击按钮切换div显示和隐藏" onclick="***toggleFn*()**"**>  <**div id="showDiv" style="width**:300**px**;**height**:300**px**;**background**:**red"**>  div显示和隐藏 </**div**> </**body**> </**html**> |

### 示例效果图



# jQuery的遍历

## 原始方式遍历

### 语法

for(var i=0;i<元素数组.length;i++){  
 元素数组[i];  
}

### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>   */\*  \* Jquery遍历一共有4种方式  \* 1.原始方式  \* 2.jquery对象方法方式  \* 3.jquery全局方法方式  \* 4.jquery3.0新特性for...of方法（js一直都有）  \* \*/* **$**(**function** () {  */\*原始方式\*/* **var** $citys = **$**(**"#city li"**);  **for**(**var** i=0;i<$citys.**length**;i++){  **var** city = $citys[i];*//city是js对象* **var** content= **$**(city).html();  *alert*(content);  }  });    </**script**> </**head**> <**body**> <**ul id="city"**>  <**li**>北京</**li**>  <**li**>上海</**li**>  <**li**>天津</**li**>  <**li**>重庆</**li**> </**ul**> </**body**> </**html**> |

### 代码效果



## jquery对象方法遍历

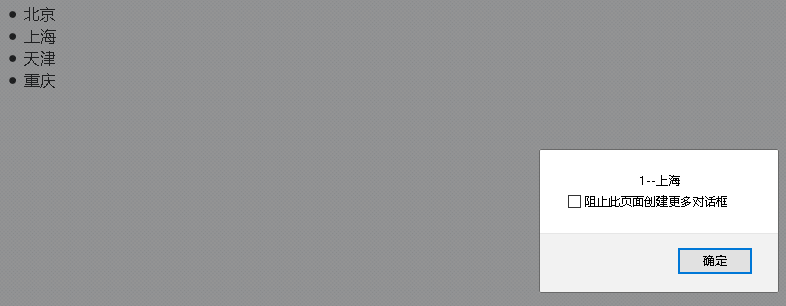
### 语法

jquery对象.each(function(index,element){});  
​  
其中，  
index:就是元素在集合中的索引  
element：就是集合中的每一个元素对象

### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>   */\*  \* Jquery遍历一共有4种方式  \* 1.原始方式  \* 2.jquery对象方法方式  \* 3.jquery全局方法方式  \* 4.jquery3.0新特性for...of方法（js一直都有）  \* \*/* **$**(**function** () {  */\*jquery对象方法方式  \* 格式：jq对象.each(fn(i,item){ //循环体代码 })  \* 参数i；循环的索引  \* 参数item:jq对象里面每个循环元素，这个元素是js对象  \* \*/* **var** $citys = **$**(**"#city li"**);  $citys.each(**function** (i,city) {  **var** content = **$**(city).html();  *alert*(i+content);  });  });    </**script**> </**head**> <**body**> <**ul id="city"**>  <**li**>北京</**li**>  <**li**>上海</**li**>  <**li**>天津</**li**>  <**li**>重庆</**li**> </**ul**> </**body**> </**html**> |

### 代码效果



## jquery的全局方法遍历

### 语法

$.each(jquery对象,function(index,element){});  
​  
其中，  
index:就是元素在集合中的索引  
element：就是集合中的每一个元素对象

### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>   */\*  \* Jquery遍历一共有4种方式  \* 1.原始方式  \* 2.jquery对象方法方式  \* 3.jquery全局方法方式  \* 4.jquery3.0新特性for...of方法（js一直都有）  \* \*/* **$**(**function** () {  */\*jquery全局方法方式  \* 格式：$.each(jq对象,fn(i,item){ //循环体代码 })  \* 参数i；循环的索引  \* 参数item:jq对象里面每个循环元素，这个元素是js对象  \* \*/* **var** $citys = **$**(**"#city li"**);  **$**.each($citys,**function** (i,city) {  **var** content = **$**(city).html();  *alert*(i+content);  });  });    </**script**> </**head**> <**body**> <**ul id="city"**>  <**li**>北京</**li**>  <**li**>上海</**li**>  <**li**>天津</**li**>  <**li**>重庆</**li**> </**ul**> </**body**> </**html**> |

### 代码效果



## jQuery3.0新特性for of语句遍历

### 语法

for(变量 of jquery对象){  
 变量；  
}  
​  
其中，  
变量:定义变量依次接受jquery数组中的每一个元素  
jquery对象：要被遍历的jquery对象

### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>   */\*  \* Jquery遍历一共有4种方式  \* 1.原始方式  \* 2.jquery对象方法方式  \* 3.jquery全局方法方式  \* 4.jquery3.0新特性for...of方法（js一直都有）  \* \*/* **$**(**function** () {  */\*jquery3.0新特性for...of方法（js一直都有）  \* 格式：for(循环元素 of jq对象/js数组){//循环体代码}  \* \*/* **var** $citys = **$**(**"#city li"**);  **for**(**var** city **of** $citys){  **var** content = **$**(city).html();  *alert*(content);  }   });      </**script**> </**head**> <**body**> <**ul id="city"**>  <**li**>北京</**li**>  <**li**>上海</**li**>  <**li**>天津</**li**>  <**li**>重庆</**li**> </**ul**> </**body**> </**html**> |

### 代码效果



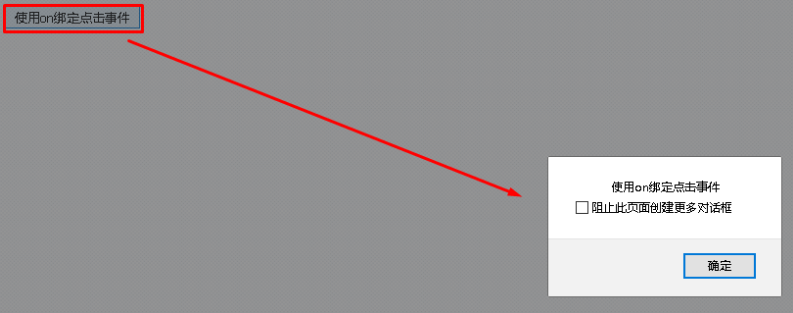
# jQuery的事件绑定与解绑

## on绑定事件

### 语法

jQuery元素对象.on(事件名称,function(){  
 //逻辑代码  
})  
​  
其中：事件名称是jQuery的事件方法的方法名称，例如：click、mouseover、mouseout、focus、blur等  
​

### 代码效果



### 代码

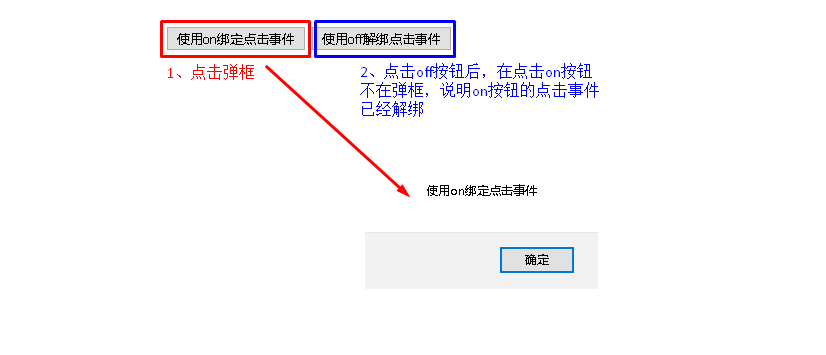
|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>  */\*  \* 事件的绑定  \* 1.jq对象.事件名(fn);//jq对象.click();  \* 2.on绑定事件方式  \* 第一种，绑定单个事件：jq对象.on("事件名",fn);  \* 第二种，绑定多个事件，jq对象.on({事件名1:fn1,事件名2:fn2....});  \*  \* js对象；var obj = {key1:value1,key2:value2...};  \* 例子：var obj = {age:20,name:"zhangsan"}  \* js对象里面属性名key，可以使用双引号括起来，也可以不括  \*  \* \*/* **$**(**function** () {   *//使用on绑定一个点击事件* **$**(**"#btn"**).on(**"click"**,**function** () {  *alert*(**"hello click"**);  });   *//使用on绑定多个事件，绑定鼠标悬浮mouseover和鼠标移出mouseout* **$**(**"#btn"**).on({  **"mouseover"**:**function** () {  *//将当前btn背景是设置为红色  //this，当前事件源对象，就是当前按钮对象，是一个js对象* **$**(**this**).css(**"background-color"**,**"red"**);  },  mouseout:**function** () {  *//将当前btn背景是设置为黄色* **$**(**this**).css(**"background-color"**,**"yellow"**);  }  });    });       *//js对象  // var obj = {age:20,name:"zhangsan"};  // alert(obj.age+","+obj.name);* </**script**> </**head**> <**body**> <**input id="btn" type="button" value="使用on绑定点击事件"**> </**body**> </**html**> |

## 解绑事件

### 语法

jQuery元素对象.off(事件名称);  
​  
其中：参数事件名称如果省略不写，可以解绑该jQuery对象上的所有事件  
​

### 代码效果



### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>  */\*  \* 事件的解绑,off，只针对动态绑定事件的解绑  \* 1.jq对象.off("事件名1")，解绑一个事件  \* 2.jq对象.off("事件名1 事件名2...")，解绑多个事件  \* 3.jq对象.off()，解绑所有事件  \*  \* 针对静态的事件绑定使用removeAttr("onclick")。。。等移出  \* \*/* **$**(**function** () {    *// 动态方式绑定事件,只要是通过js或jq代码实现绑定的事件都是动态的绑定  //使用on绑定一个点击事件  /\* $("#btn").on("click",function () {  alert("hello click");  });   //使用on绑定多个事件，绑定鼠标悬浮mouseover和鼠标移出mouseout  $("#btn").on({  "mouseover":function () {  //将当前btn背景是设置为红色  //this，当前事件源对象，就是当前按钮对象，是一个js对象  $(this).css("background-color","red");  },  mouseout:function () {  //将当前btn背景是设置为黄色  $(this).css("background-color","yellow");  }  }); \*/* });   *//静态方式绑定事件，直接写在标签上的事件为静态绑定* **function** *onclickFn*() {  *alert*(**"hello click"**);  }  **function** *onmouseoverFn*(obj) {  **$**(obj).css(**"background-color"**,**"red"**);  }    **function** *onmouseoutFn*(obj) {  **$**(obj).css(**"background-color"**,**"yellow"**);  }    */\*解绑btn一个click事件\*/* **function** *offOneEventFn*() {  *// $("#btn").off("click");//解绑动态事件* **$**(**"#btn"**).removeAttr(**"onclick"**);   }   */\*解绑btn鼠标移入与移出事件\*/* **function** *offMutiEventFn*() {  *// $("#btn").off("mouseover mouseout");* **$**(**"#btn"**).removeAttr(**"onmouseover"**).removeAttr(**"onmouseout"**);  }   */\*解绑btn所有事件\*/* **function** *offAllEventFn*() {  *//$("#btn").off();* **$**(**"#btn"**).removeAttr(**"onmouseover"**).removeAttr(**"onmouseout"**).removeAttr(**"onclick"**);  }    *//js对象  // var obj = {age:20,name:"zhangsan"};  // alert(obj.age+","+obj.name);* </**script**> </**head**> <**body**> *<!--onclick="onclickFn()" onmouseover="onmouseoverFn(this);" onmouseout="onmouseoutFn(this)"-->* <**input id="btn" type="button" value="使用on绑定点击事件" onclick="***onclickFn*()**" onmouseover="***onmouseoverFn*(**this**);**" onmouseout="***onmouseoutFn*(**this**)**"**> <**input id="btn2" type="button" value="使用off解绑点击事件" onclick="***offOneEventFn*();**"**> <**input id="btn3" type="button" value="使用off解绑鼠标移入与移除事件" onclick="***offMutiEventFn*()**"**> <**input id="btn4" type="button" value="使用off解绑所有事件" onclick="***offAllEventFn*();**"**> </**body**> </**html**> |

​

## 事件切换

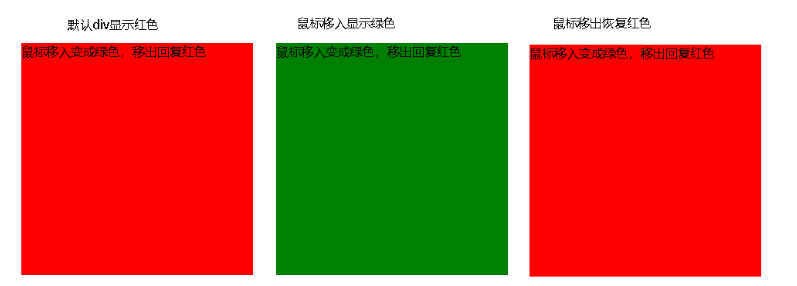
介绍：同时绑定多个事件的方式

### 普通写法

#### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**></**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>   */\*  \* 事件切换，主要是针对mouseover与mouseout事件的切换  \* 1.使用hover(fn1,fn2),fn1是mouseover事件，fn2是mouseout事件  \* 2.链式方式进行切换，jq对象.mouseover(fn1).mouseout(fn2);  \* 3.直接分别绑定mouseover和mouseout  \* \*/* **$**(**function** () {  */\*直接分别绑定mouseover和mouseout\*/  /\* $("#myDiv").mouseover(function () {  $(this).css("background-color","green");  });  $("#myDiv").mouseout(function () {  $(this).css("background-color","red");  }); \*/  /\*使用hover(fn1,fn2),fn1是mouseover事件，fn2是mouseout事件\*/  /\* $("#myDiv").hover(function () {  //移入事件  $(this).css("background-color","green");  },function () {  //移出事件  $(this).css("background-color","red");  })\*/   /\*链式方式进行切换，jq对象.mouseover(fn1).mouseout(fn2);\*/* **$**(**"#myDiv"**).mouseover(**function** () {  **$**(**this**).css(**"background-color"**,**"green"**);  }).mouseout(**function** () {  **$**(**this**).css(**"background-color"**,**"red"**);  });    });    </**script**> </**head**> <**body**>  <**div id="myDiv" style="width**:300**px**;**height**:300**px**;**background**:**red"**>  鼠标移入变成绿色，移出回复红色  </**div**> </**body**> </**html**> |

#### 效果

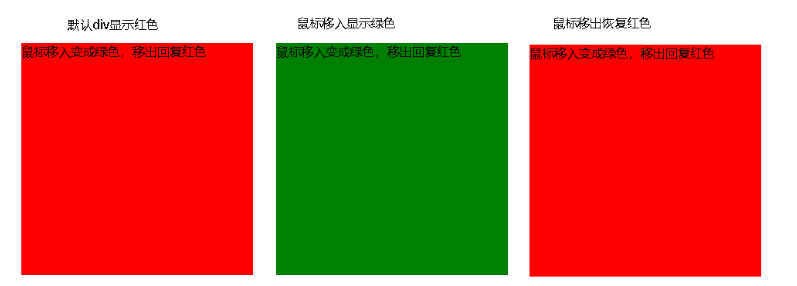


### 链式写法

#### 代码

|  |
| --- |
| <!DOCTYPE html> <html> <head>  <meta charset="UTF-8">  <title></title>  <script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"></script>  <script type="text/javascript">  */\*  \* 事件切换方式，同时绑定多个事件方式  \* 1.普通发方式  \* 2.链式方式  \* 3.切换方式  \* \*/  //2.链式方式,每个事件都会返回一个对象，这个对象就是事件源的jq对象* $(function () {  *//注册移入事件（鼠标悬浮）* $("#myDiv").mouseover(function () {  $(this).css("background-color","green");  }).mouseout(function () {  $(this).css("background-color","red");  });  });   </script> </head> <body>  <div id="myDiv" style="width:300px;height:300px;background:red">  鼠标移入变成绿色，移出回复红色  </div> </body> </html> |

#### 效果



### 切换函数写法

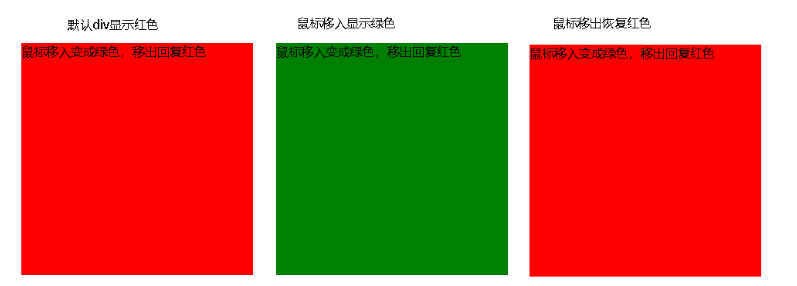
#### 语法

hover([over,]out)  
​  
其中，over代表鼠标移入事件触发的函数，out代表鼠标移出事件触发的函数  
​

#### 代码

|  |
| --- |
| <!DOCTYPE html> <html> <head>  <meta charset="UTF-8">  <title></title>  <script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"></script>  <script type="text/javascript">  */\*  \* 事件切换方式，同时绑定多个事件方式  \* 1.普通发方式  \* 2.链式方式  \* 3.切换方式（有限制，是真正的切换）  \* \*/  // 3.切换方式，使用hover(鼠标悬浮事件，鼠标移出事件)；* $(function () {  $("#myDiv").hover(  function () {  $(this).css("background-color","green");  },  function () {  $(this).css("background-color","red");  }  );  });   </script> </head> <body>  <div id="myDiv" style="width:300px;height:300px;background:red">  鼠标移入变成绿色，移出回复红色  </div> </body> </html> |

#### 效果



# jQuery综合案例

## 广告的自动显示与隐藏

### 需求描述

页面打开，延迟几秒钟，广告从上向下慢慢滑下显示，停留几秒，再从下向上慢慢滑动隐藏。

### 代码效果



### 代码实现

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>广告的自动显示与隐藏</**title**>  <**style**>  **#content**{**width**:100%;**height**:500**px**;**background**:**#999**}  </**style**>   *<!--引入jquery-->* <**script type="text/javascript" src="../js/jquery-3.3.1.min.js"**></**script**>  <**script**>  **$**(**function** () {  *//等3秒钟向下展开3秒显示广告，广告停留5秒钟后，向上收起3秒  setTimeout*(**function** () {   *//让ad向下展开滑动显示，3秒动画时间* **$**(**"#ad"**).**slideDown**(3000,**function** () {  *//广告显示后的回调函数   //让广告停留5秒，后让广告收起滑动隐藏  setTimeout*(**function** () {  *//让广告3秒收起* **$**(**"#ad"**).**slideUp**(3000);  },5000);  });   },3000);*//3秒以后执行js代码* });  </**script**> </**head**> <**body**> *<!-- 整体的DIV -->* <**div**>  *<!-- 广告DIV  display:控制元素显示的样式  none，隐藏  inline,显示后面不加换行  block，显示后面加换行  -->* <**div id="ad" style="display**: **none**;**"**>  <**img style="width**:100%**" src="../img/adv.jpg"** />  </**div**>   *<!-- 下方正文部分 -->* <**div id="content"**>  正文部分  </**div**> </**div**> </**body**> </**html**> |

## 抽奖程序

### 需求描述

* 当用户点击开始按钮时，小像框中的像片快速切换。
* 当用户点击停止按钮时，小像框中的像片停止切换，延时2秒后，大像框中也会显示与小像框相同的像片。

### 代码效果



### 代码实现

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>jquery案例之抽奖</**title**>  <**script type="text/javascript" src="../js/jquery-3.3.1.min.js"**></**script**> </**head**> <**body**>  *<!-- 小像框 -->* <**div style="border-style**:**dotted**;**width**:160**px**;**height**:100**px"**>  <**img id="img1ID" src="../img/man00.jpg" style="width**:160**px**;**height**:100**px"**/> </**div**>  *<!-- 大像框 -->* <**div  style="border-style**:**double**;**width**:800**px**;**height**:500**px**;**position**:**absolute**;**left**:500**px**;**top**:10**px"**>  <**img id="img2ID" src="../img/man00.jpg" width="800px" height="500px"**/> </**div**>  *<!-- 开始按钮 -->* <**input  id="startID"  type="button"  value="点击开始"  style="width**:150**px**;**height**:150**px**;**font-size**:22**px"  onclick="***imgStart*()**"**>  *<!-- 停止按钮 -->* <**input  id="stopID"  type="button"  value="点击停止"  style="width**:150**px**;**height**:150**px**;**font-size**:22**px"  onclick="***imgStop*()**"**>   <**script language='javascript' type='text/javascript'**>  *//准备一个一维数组，装用户的像片路径* **var *imgs*** = [  **"../img/man00.jpg"**,  **"../img/man01.jpg"**,  **"../img/man02.jpg"**,  **"../img/man03.jpg"**,  **"../img/man04.jpg"**,  **"../img/man05.jpg"**,  **"../img/man06.jpg"** ];   *//点击图片开始抽奖事件* **function** *imgStart*() {  *//让数组里面的图片轮换显示，每100毫秒轮换一张  changeImg*();  *//让按钮"点击开始"，禁用* **$**(**"#startID"**).prop(**"disabled"**,**true**);  *//让按钮"点击停止"，启用* **$**(**"#stopID"**).prop(**"disabled"**,**false**);  }   *//定义变量存储循环索引* **var *number*** = 0;    **function** *changeImg*() {  *//每100毫秒轮换一张* **taskId**= *setInterval*(**function** () {  *//计算出轮换第几张图片* **var** index = ***number***%***imgs***.length;  *//获取当前要显示的图片地址* **curImagPath** = ***imgs***[index];  *//将图片更新到页面小图片位置* **$**(**"#img1ID"**).attr(**"src"**,**curImagPath**);   ***number***++;  },100);  }   *//点击停止事件* **function** *imgStop*() {  *//停止定时器  clearInterval*(**taskId**);  *//让按钮"点击开始"，启用* **$**(**"#startID"**).prop(**"disabled"**,**false**);  *//让按钮"点击停止"，禁用* **$**(**"#stopID"**).prop(**"disabled"**,**true**);   *//将当前轮换的图片显示到大图片位置* **$**(**"#img2ID"**).hide();  **$**(**"#img2ID"**).attr(**"src"**,**curImagPath**);  **$**(**"#img2ID"**).**fadeIn**(3000);   *//延迟2秒后显示  //$("#img2ID").delay(2000).show(0);//只需要了解* }  </**script**> </**body**> </**html**> |

# jQuery的插件

## 插件概念

插件就是以jquery为基础进行扩展具有特定功能的代码，就是jquery插件。例如：图片轮播、选项卡等。

例如：jquery插件网：https://www.jq22.com/

## jQuery的插件机制

### jQuery插件机制概述

jQuery插件的机制很简单，就是利用jQuery提供的jQuery.fn.extend()和jQuery.extend()方法，扩展jQuery的功能。

### jQuery插件机制语法

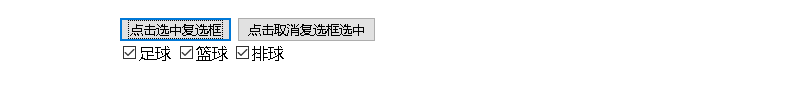
| **语法** | **解释** |
| --- | --- |
| jQuery.fn.extend(object) | 对jQuery对象进行方法扩展 |
| jQuery.extend(object) | 对jQuery全局进行方法扩展 |

### 对jQuery对象进行方法扩展

#### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>01-jQuery对象进行方法扩展</**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>  */\*  \* jquery插件机制  \* 作用，增强jquery，扩展jquery  \* $.fn.extend(object) 对jQuery对象进行方法扩展  $.extend(object) 对jQuery全局进行方法扩展  \* \*/   /\*$.fn.extend(object) 对jQuery对象进行方法扩展  \* object,是js对象，{key1：value1,key2:value2...}  \*  \* \*/* **$**(**function** () {  *//对jq对象扩展方法check(),uncheck();* **$**.**fn**.extend({  check:**function** () {  *//this,由于当前是扩展jq对象，所以this就是当前的jq对象* **this**.prop(**"checked"**,**true**);  },  uncheck:**function** () {  **this**.prop(**"checked"**,**false**);  }  });  });   **function** *checkFn*() {  *//$("input:checkbox").prop("checked",true); 优化扩展jquery对象，调用check()就可以实现这个效果* **$**(**"input:checkbox"**).check();  }   **function** *uncheckFn*() {  *//$("input:checkbox").prop("checked",false);优化扩展jquery对象，调用uncheck()就可以实现这个效果* **$**(**"input:checkbox"**).uncheck();  }    </**script**> </**head**> <**body**> <**input id="btn-check" type="button" value="点击选中复选框" onclick="***checkFn*()**"**> <**input id="btn-uncheck" type="button" value="点击取消复选框选中" onclick="***uncheckFn*()**"**> <**br**/> <**input type="checkbox" value="football"**>足球 <**input type="checkbox" value="basketball"**>篮球 <**input type="checkbox" value="volleyball"**>排球  </**body**> </**html**> |

#### 效果



### 对jQuery全局进行方法扩展

#### 代码

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>01-jQuery对象进行方法扩展</**title**>  <**script src="../js/jquery-3.3.1.min.js" type="text/javascript" charset="utf-8"**></**script**>  <**script type="text/javascript"**>  *//对全局方法扩展2个方法，扩展min方法：求2个值的最小值；扩展max方法：求2个值最大值  /\*$.extend(object) 对jQuery全局进行方法扩展\*/* **$**(**function** () {   **$**.extend({  min:**function** (a,b) {  *//this,这里没有this，因为这个方法不属于jq对象，是全局方法* **return** a<b?a:b;  },  max:**function** (a,b) {  **return** a<b?b:a;  }  });    *alert*(**$**.**min**(1,2));  *alert*(**$**.**max**(1,2));   });    </**script**> </**head**> <**body**> </**body**> </**html**> |

## 表单校验插件validator

### 表单校验插件概述

网络上有许多成熟的插件共使用者参考，插件就是将jquery实现的常用操作进行封装，封装后的“小框架“就可以叫做插件，按照插件的语法去操作可以实现很多复杂的功能，而我们需要做的是学会该插件的使用语法即可。表单校验插件是按照一定的语法编写代码，可以使用简单的代码完成复杂的表单校验工作。

### 表单校验插件的基本语法

#### 开发步骤

* 下载jquery-validation插件
* 将该插件（也就是一个js文件）导入到我们的工程中
* 在要使用校验插件的html中引入该js文件
* 编写表单校验的代码（语法）

$("form表单的选择器").validate({  
    rules:{  
      表单项name值:校验规则，  
      表单项name值:校验规则... ...  
   },  
    messages:{  
      表单项name值:错误提示信息，  
      表单项name值:错误提示信息... ...  
   }  
});

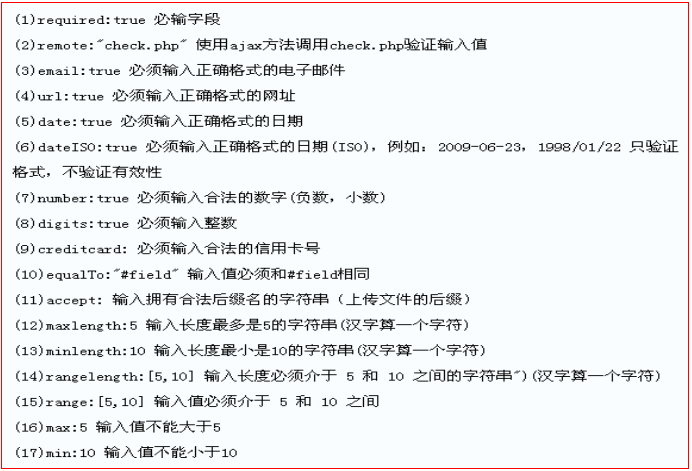
其中，rules是对表单项校验的规则，messages是对应的表单项校验失败后的错误提示信息

注意，当错误提示信息不按照我们预想的位置显示时，我们可以按照如下方式进行设置自定义错误显示标签放在我们需要显示的位置，当此表单项验证不通过时会将此信息自动显示出来，jQuery验证插件会自动帮助我们控制它的显示与隐藏

|  |
| --- |
| <lable for="html元素name值" class="error" style="display:none">错误信息</lable> |

如果设置了错误lable标签就不必在messages中设置此表单项错误提示信息了

#### 常用的校验规则



### 表单校验插件自定义校验方法

如果预定义的校验规则尚不能满足需求的话可以进行自定义校验规则：

自定义校验规则步骤如下：

* 使用$.validator.addMethod("校验规则名称",function(value,element,params)){}
* 在rules中通过校验规则名称使用校验规则
* 在messages中定义该规则对应的错误提示信息

其中， value是校验组件的value值，element是校验组件的节点对象，params是校验规则的参数。

### 代码实现

|  |
| --- |
| <**html**>  <**head**>  <**meta http-equiv="Content-Type" content="text/html; charset=utf-8"** />  <**title**>我的jquery表单校验页面</**title**>    <**style type="text/css"**>  **p**{**text-align**: **center**;**font-size**:24**px**;}  **table**{**margin**: 0 **auto**;**border**: 0;}  **table tr**{**height**:40**px**;**border**:0;}  **table tr td**{**padding**:0 14**px**;**border**:1**px solid #999**}  .**error**{**color**:**red**}  </**style**>    <**script type="text/javascript" src="../js/jquery-3.3.1.js"**></**script**>  <**script type="text/javascript" src="../js/jquery.validate.min.js"**></**script**>  *<!--默认的错误信息是英文的，可以使用国际化js文件转换为中文-->* <**script type="text/javascript" src="../js/messages\_cn.js"**></**script**>  <**script type="text/javascript"**>  **$**(**function** () {   */\*  \* 当前插件自定义校验规则  \* • 使用$.validator.addMethod("校验规则名称",function(value,element,params){})  \* //校验规则名称,自己定义  \* //function(value,element,params),验证的函数  \* value,当前验证的表单元素值  \* element,当前表单元素对象，js对象  \* params,是验证规则传递的参数  \* \*/   //规则1,身份证必须是15或18个字符长度* **$**.validator.addMethod(**"cartLength"**,  **function**(value,element,params){  *//params,[15,18]* **if**(value.**length**==params[0] || value.**length**==params[1]){  **return true**;*//验证通过* }  **return false**;*//验证不通过* }  );  *//规则2，如果15位，要求必须全部是数字* **$**.validator.addMethod(**"cart15"**,  **function**(value,element,params){  **if**(value.**length**==15){   **var** reg=/^\d{15}$/;  **if**(reg.test(value)){  **return true**;  }**else**{  **return false**;  }   }  **return true**;*//如果不是15为，让其他验证模式去验证，此时验证通过* }  );   *//规则3，如果是18位，要求前17位是数字最后一个是X或者全部是数字* **$**.validator.addMethod(**"cart18"**,  **function**(value,element,params){  **if**(value.**length**==18){   **var** reg=/^\d{17}(X|\d)$/;  **if**(reg.test(value)){  **return true**;  }**else**{  **return false**;  }   }  **return true**;*//如果不是18位，让其他验证模式去验证，此时验证通过* }  );    **$**(**"#empForm"**).validate({  */\*定义规则的  \* 表单元素name:验证模式  \* \*/* **rules**: {  **realname**:**"required"**,  **username**: {  **required**: **true**,  **minlength**: 5,  **maxlength**:8  },  **psw**: {  **required**: **true**,  **rangelength**:[6,12]  },  **psw2**: {  **required**: **true**,  **rangelength**:[6,12],  **equalTo**:**"#psw"** },  **gender**:**"required"**,  **age**:{  **required**: **true**,  **range**:[26,50]  },  **edu**:**"required"**,  **birthday**:{  **required**: **true**,  **date**:**true** },  **email**:{  **required**: **true**,  **email**:**true** },  **cart**:{  **required**: **true**,  **cartLength**:[15,18],  **cart15**:**true**,  **cart18**:**true** }   },  */\*验证失败的时候，显示错误消息\*/* **messages**: {  *// realname2:"真实姓名不能为空",如果不设置，会采用默认验证信息* **username**: {  **required**: **"请输入登录名"**,  **minlength**: **"至少5个字符"**,  **maxlength**:**"最多8个字符"** },  **psw**: {  **rangelength**:**"字符要求6~12个"** },  **gender**:**"请选择一个性别"**,  **cart**:{  **cartLength**:**"身份证必须是15或18位"**,  **cart15**:**"15位身份证号必须全部是数字"**,  **cart18**:**"格式错误"** }  }  });   })   </**script**> </**head**> <**body**>  <**p**>员工信息录入</**p**>  <**form name="empForm" id="empForm" method="post" action="test.html"**>  <**table border=1**>  <**tr**>  <**td**>真实姓名(不能为空 ,没有其他要求)</**td**>  <**td**><**input type="text" id="realname" name="realname"** />  </**td**>  </**tr**>  <**tr**>  <**td**>登录名(登录名不能为空,长度应该在5-8之间,可以包含中文字符(一个汉字算一个字符)):</**td**>  <**td**><**input type="text" id="username" name="username"** /></**td**>  </**tr**>  <**tr**>   <**td**>密码(不能为空,长度6-12字符或数字,不能包含中文字符):</**td**>  <**td**><**input type="password" id="psw" name="psw"** /></**td**>  </**tr**>  <**tr**>   <**td**>重复密码密码(不能为空,长度6-12字符或数字,不能包含中文字符):</**td**>  <**td**><**input type="password" id="psw2" name="psw2"** /></**td**>  </**tr**>  <**tr**>  <**td**>性别(必选其一)</**td**>  <**td**>  <**input type="radio" id="gender" value="m" name="gender"**/>**&nbsp;**男**&nbsp;&nbsp;** <**input type="radio" id="gender1" value="f" name="gender"**/>**&nbsp;**女  <**label for="gender" generated="true" class="error" style="display**: **none**;**"**>请选择一个性别</**label**>  </**td**>  </**tr**>  <**tr**>  <**td**>年龄(必填26-50):</**td**>  <**td**><**input type="text" id="age" name="age"** /></**td**>  </**tr**>    <**tr**>   <**td**>你的学历:</**td**>  <**td**> <**select name="edu" id="edu"**>  <**option value=""**>－请选择你的学历－</**option**>  <**option value="a"**>专科</**option**>  <**option value="b"**>本科</**option**>  <**option value="c"**>研究生</**option**>  <**option value="e"**>硕士</**option**>  <**option value="d"**>博士</**option**>  </**select**>  </**td**>  </**tr**>    <**tr**>   <**td**>出生日期(1982/09/21):</**td**>  <**td**><**input type="text" id="birthday" name="birthday" value=""** /></**td**>  </**tr**>    <**tr**>   <**td**>兴趣爱好:</**td**>  <**td colspan="2"**>   <**input type="checkbox" name="checkbox1" id="qq1"**/>**&nbsp;**乒乓球 **&nbsp;** <**input type="checkbox" name="checkbox1" id="qq2" value="1"** />**&nbsp;**羽毛球 **&nbsp;** <**input type="checkbox" name="checkbox1" id="qq3" value="2"** />**&nbsp;**上网 **&nbsp;** <**input type="checkbox" name="checkbox1" id="qq4" value="3"** />**&nbsp;**旅游 **&nbsp;** <**input type="checkbox" name="checkbox1" id="qq5" value="4"** />**&nbsp;**购物 **&nbsp;** </**td**>  </**tr**>  <**tr**>   <**td align="left"**>电子邮箱:</**td**>  <**td**><**input type="text" id="email" name="email"** /></**td**>  </**tr**>  <**tr**>   <**td align="left"**>身份证(15-18):</**td**>  <**td**><**input type="text" id="cart" name="cart"** /></**td**>  </**tr**>  <**tr**>  <**td**></**td**>  <**td**><**input type="submit" name="firstname" id="firstname" value="保存"**></**td**>  </**tr**>  </**table**>  </**form**>  <**script type="text/javascript"**>     </**script**>  </**body**> </**html**> |

### 代码效果

