## Ajax

|  |
| --- |
| $.ajax({url: ""}); |
| $.ajax({  type: "POST",  async: **false**, // 默认true:异步, 同步false 注意: 同步请求将锁住浏览器，用户其它操作必须等待请求完成才可以执行。  dataType: "json",//返回json格式的数据  url: "<c:url value='/session/accountIsUsed' />",  data : {  account:account,  time : new Date().getTime()  },  success: **function**(data){  **if**(data.result){  $("#account-error").html("登录名已被占用！");  }**else**{  $("#account-error").html("");  flag = **true**;  }  },  error: **function**(msg) {  alert("Data not found");  }  }); |
| java输出json对象  **public** **void** writeDataToResponse(Object data, HttpServletResponse response) {  response.setHeader("Cache-Control", "no-cache");  response.setContentType("text/html; charset=UTF-8");  PrintWriter out = **null**;  **try** {  out = response.getWriter();  out.println(data == **null** ? "" : data.toString());  } **catch** (IOException e) {  log.error("write data to response failed:" + e);  } **finally** {  **if** (out != **null**) {  out.flush();  out.close();  }  }  } |

### jQuery迭代

|  |
| --- |
| **var** aa = [0, 1, 2];  $.each(aa,**function**(i, n) {  alert(i + ": " + n);  }); |
| **var** bb = {  name: "John",  lang: "JS"  };  $.each(bb,**function**(i, n) {  alert(i + ":" + n);  }); |

### jquery对象和javascript对象互转

|  |
| --- |
| DOM**对象转jquery对象**  var v = document.getElementById("v");        //DOM对象  var $v = $(v);        //jQuery 对象 |
| **jquery对象转**DOM**对象**  var $v = $("#v");          //jQuery 对象  var v = $v[0]; 或者$v.get(0)               //DOM 对象 |

## get和post

|  |
| --- |
| **get和post的区别**  //构造要传的参数和参数值  **function** createString(){  **var** username = encodeURI($("#username").val());  **var** password = encodeURI($("#password").val());  **var** parameterString = {username:username,  password:password};  **return** parameterString;  }  //get方式  **function** doRequestUsingGET(){  $.get("http://service.swu.edu.cn/ipaddr/remote\_logout2.jsp", createString(), **function**(data){  $("#serverResponse").html(decodeURI(data));  })  }  //post方式  **function** doRequestUsingPOST(){  $.post("http://service.swu.edu.cn/ipaddr/remote\_logout2.jsp", createString(), **function**(data){  $("#serverResponse").html(decodeURI(data));  })  } |
| **设置全局Ajax**  //构造要传的参数和参数值  **function** createString(){  **var** username = encodeURI($("#username").val());  **var** password = encodeURI($("#password").val());  **var** parameterString = {username:username,  password:password};  **return** parameterString;  }  //全局设置Ajax  $.ajaxSetup({  url:"http://service.swu.edu.cn/ipaddr/remote\_logout2.jsp",  data:createString(),  success:**function**(data){  $("#serverResponse").html(decodeURI(data));  }  })    //get方式  **function** doRequestUsingGET(){  $.ajax({  type:"GET"  })  }  //post方式  **function** doRequestUsingPOST(){  $.ajax({  type:"POST"  })  } |

## Jquery异步传输纯文本

|  |
| --- |
| js/verify.js  **function** verify(){  //1.获取文本框中的内容  //用js做:document.getElementById("username");  //Jquery的查找节点的方法: jquery方法返回的都是jquery对象, 可以继续在上面执行其他的jquery方法.  **var** jqueryObj = $("#username");  **var** username = jqueryObj.val();    //2.将文本框中的诗句发送给服务器的servlet  //第一个参数: 发送给服务器的url地址"servlet/LoginServlet" , 和web.xml中servlet的url-pattern值一样. 它会自动加上项目名.  $.get("servlet/LoginServlet?username=" + username, **null**, callback)    //回调函数  **function** callback(data){  //3.接受服务器端返回的数据    //4.接收服务器端传来的数据, 并将数据显示在页面上.  **var** resultObj = $("#result");  resultObj.html(data); // 给节点赋值  }  } |
| HTML页面代码  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  <html>  <head>  <title>login.html</title>  <meta http-equiv=*"keywords"* content=*"keyword1,keyword2,keyword3"*>  <meta http-equiv=*"description"* content=*"this is my page"*>  <meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"*>  <!--<link rel="stylesheet" type="text/css" href="./styles.css">-->  <script type=*"text/javascript"* src=*"js/jquery.js"*></script>  <script type=*"text/javascript"* src=*"js/verify.js"*></script>  </head>  <body>  <input type=*"text"* id=*"username"*>  <input type=*"button"* id=*"test"* value=*"检查用户名是否可用"* onclick="verify()"/>  <span id=*"result"*></span>  </body>  </html> |
| Servlet.java里面的代码:  package servlet;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class LoginServlet extends HttpServlet {  public void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  this.doPost(request, response);  }  public void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  response.setContentType("text/html;charset=utf-8");  PrintWriter out = response.getWriter();    // 1. 取参数  String username = request.getParameter("username");  System.out.println("username: " + username);  // 2. 检查参数是否有问题(容错处理)  if(username == null || username.length() == 0){  out.println("用户名不能为空!");  // 3. 对参数做校验  }else if(username.equalsIgnoreCase("jinwanlin")){  // 4. 和传统应用不同, 这一步需要将用户感兴趣的数据返回给页面, 而不是将一个新的页面发送给用户.  out.println("用户名"+ username +"已存在, 请使用其它用户名");  }else{  out.println("用户名"+ username +"尚未存在, 可以使用该用户名注册.");  }  }  } |

## Jquery异步传输XML文件

|  |
| --- |
| **js/verify.js**  **function** verify(){  //1.获取文本框中的内容  //用js做:document.getElementById("username");  //Jquery的查找节点的方法: jquery方法返回的都是jquery对象, 可以继续在上面执行其他的jquery方法.  **var** jqueryObj = $("#username");  **var** username = jqueryObj.val();  //2.将文本框中的诗句发送给服务器的servlet  //第一个参数: 发送给服务器的url地址"servlet/LoginServlet" , 和web.xml中servlet的url-pattern值一样. 它会自动加上项目名.  // $.get("servlet/LoginServlet?username=" + username, null, callback)  **var** url = "servlet/LoginServlet";  $.ajax({  type: "GET", //get 或 post  url: convertURL(url) , //url加上时间戳  dataType: "script",    //解决中文乱码, 做一次encodeURI  // 或者做两次encodeURI, 服务器端手动做一次decode(uaername, "utf-8"), 因为服务器接收到参数后会自动做一次decode;  data: "username="+ encodeURI(username),  dataType: "xml", //告诉jquery返回的数据格式  success: callback //回调函数名  });  }  //回调函数 , 接受服务器端返回的数据:data  **function** callback(data){  //3.  //将data这个dom对象转换成jquery对象.  **var** jqueryObj = $(data);  //查找message这个节点  **var** message = jqueryObj.children(); //obj.children 返回的是所有子节点的文本拼接成的字符串  //取message节点的文本内容  **var** text = message.text();  //将数据显示在页面上.  **var** resultObj = $("#result");  resultObj.html(text); // 给节点赋值  }  //给url地址增加时间戳, 骗过浏览器, 不读取缓存  **function** convertURL(url){  //获取时间戳  **var** timstamp = (**new** Date()).valueOf();  //将时间戳信息拼接在url后面  **if**(url.indexOf("?") >= 0){  url = url + "&t=" + timstamp;  }**else** {  url = url + "?t=" + timstamp;  }  **return** url;  } |
| **Servlet.java里面的代码:**  package servlet;  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public class LoginServlet extends HttpServlet {  private static final long serialVersionUID = 6264381047408967619L;  public void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  this.doPost(request, response);  }  public void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  //修改contentType为text/xml  response.setContentType("text/xml;charset=utf-8");  PrintWriter out = response.getWriter();  StringBuffer builder = new StringBuffer();  builder.append("<message>");  // 1. 取参数  String username = request.getParameter("username");  // 2. 检查参数是否有问题(容错处理)  if(username == null || username.length() == 0){  builder.append("用户名不能为空!").append("</message>");  // 3. 对参数做校验  }else if(username.equalsIgnoreCase("jinwanlin")){  // 4. 和传统应用不同, 这一步需要将用户感兴趣的数据返回给页面, 而不是将一个新的页面发送给用户.  builder.append("用户名【"+ username +"】已存在, 请使用其它用户名").append("</message>");  }else{  builder.append("用户名【"+ username +"】尚未存在, 可以使用该用户名注册.").append("</message>");  }  out.println(builder.toString());  System.out.println(builder.toString());  }  } |

## AJAX中使用DOM对象

|  |
| --- |
|  |

# HTML对象转jquery对象

|  |
| --- |
| **var** appObj = $(app); //HTML对象转jquery对象 |

# jQuery基本语法

$符号用法

|  |
| --- |
| **$选择器\_1----html元素类型, 父子关系**  window.onload = **function**(){  **var** oElements = $("h2 a"); //选择<h2><a>111</a></h2>  **for**(**var** i=0; i<oElements.length; i++){  alert(oElements[i]);  }  } |
| **选择器\_2-----html元素类型**  **var** oElements = $("div"); //查找一个 DIV 节点元素。 |
| **$选择器\_3----ID**  $("#myDiv"); // ID="myDiv"的元素。 |
| **$选择器\_3----class**  $(".myClass"); // class="myClass"的元素。 |
| **$选择器\_3----通配符**  $("\*"); //每一个元素 |
| **$选择器\_4----** li:has(a) // <li>下含<a>的元素  $("ul li ul li:has(a)").addClass("altrow"); //给<ul>下的<li>下的<ul>下的<li>下包含<a>的元素添加class属性. |
| **$选择器\_5----**  $("p:lt(2)").addClass("myClass"); //<p>元素数组中, 脚标小于2的元素, 添加class属性. |
| **去掉字符串前后空格**  **var** sString = " 13 24 ";  sString = **$**.**trim**(sString); //去掉字符串前后空格, 中间的空格这个函数去不掉  alert(sString.length); |
| **代替window.onload**  $(document).ready(**function**(){ //可以写任意多个  //...  })  //或者简写成  $(**function**(){  //...  }) |
| **创建DOM元素**  $(**function**(){  **var** oNewP = $("<p>这是新创建的段落</p>")  oNewP.insertAfter("#myTarget"); //插入到id="myTarget"的元素后面  }) |
| **获取节点属性的值**  $(**function**(){  **var** sTitle = $("a:eq(1)").attr("href"); //获取第二个<a>元素节点的href属性值  $("span").text(sTitle); //把 sTitle 赋值给节点$("span")的文本  }) |
| **追加内容**  $("#logbox").append("<p>Hello</p>"); |
| **设置节点属性的值**  **function** DisableBack(){  $("button:gt(0)").attr("disabled", "disabled"); //修改button数组中大于脚标大于0的元素的disabled属性等于disabled  } |
| **给节点同时添加多个class**  <script type=*"text/javascript"*>  <!--  $(**function**(){  $("div").addClass("class1 class2");  });  //-->  </script>  <style type=*"text/css"*>  .class1 {  border: *1px solid #750037*;  width: *120px*;  height: *80px*;  }    .class2 {  background-color: *#ffcdfc*;  }  </style> |
| **动态切换绑定和解除绑定样式**  $(**function**(){  $("div").addClass("class1");  $("div").click(**function**(){  $(**this**).toggleClass("class2"); //在绑定class2和解除绑定class2之间切换  });  }); |
| **获取节点内容的两种方法的区别**  $(**function**(){  **var** sString1 = $("p:first").text(); //获取$("p:first")节点的纯文本, 过滤掉各种各样的显示样式  $("div").html(sString1);  **var** sString2 = $("p:first").html(); //获取$("p:first")节点的纯文本, 保留显示样式  $("span").html(sString2);  })  <p><b>文本1</b>分隔<em>文本2</em>尾巴</p>  <div></div><p />  <span></span> |
| **克隆元素**  $(**function**(){  $("img:eq(1)").clone().appendTo($("p")); //克隆第2张图片, 然后将它添加到$("p")节点后面  $("img:eq(0)").clone().appendTo($("p:eq(0)")); //克隆第1张图片, 然后将它添加到$("p:eq(0)")后面  }); |
| **把多个事件绑定到一个节点**  $(**function**(){  $("img")  .bind("click", fnMyFunc1 = **function**(){  $("#show").append("<div>点击事件1</div>");  })  .bind("click", **function**(){  $("#show").append("<div>点击事件2</div>");  })  .bind("click", **function**(){  $("#show").append("<div>点击事件3</div>");  });  $("input[type=button]").click(**function**(){  $("img").unbind("click", fnMyFunc1); //移除对象fnMyFunc1  })  }); |
| **显示和隐藏**  $(**function**(){  $("input:first").click(**function**(){  $("p").hide(); //还可以是hide(3000), 表示3000毫秒; 右下往左上收起  //也可以用fadeOut(3000);//整体模糊渐变  //可用slideUp(speed, [callback])  })  $("input:last").click(**function**(){  $("p").show();//还可以是show(1000), 表示3000毫秒; 左上往右下展开  //也可以用fadeIn(1000);// 整体模糊渐变  //也可用slideDown(speed, [callback])  })  }) |
| **broswer对象**    **function** detect(){  **if**($.browser.msie) **return** "IE";  **if**($.browser.mozilla) **return** "Mozilla";  **if**($.browser.safari) **return** "Safari";  **if**($.browser.opera) **return** "Opera";  }  **var** sBrowser = detect();  document.write("名称: " + sBrowser);  document.write("<br>");  document.write("版本: " + $.browser.version); |
| **盒子模型**      **var** sBox = $.boxModel ? "标准W3C盒子模型" : "IE盒子模型";  document.write(sBox); |
| **each遍历**  $each(obj, fn) /obj:要遍历的对象; fn:遍历的每次循环都要执行这个函数. |
| **数据过滤**(像纱布一样, 滤掉不符合条件的数据, 留下符合条件的数据)  **var** aArray = [1, 2, 3, 4, 5, 6, 7, 8, 9];  **var** aResult = $.grep(aArray, **function**(value){ //$.grep(对象, 函数(变量)) 说明: 对象:是等待过滤的源数据, 函数:是过滤函数  **return** value > 4; //过滤条件, 其中value是对象中的一个值  })  document.write("aArray: " + aArray.join() + "<br>");  document.write("aResult: " + aResult.join()); |
| **数组的转化** |

# jQuery读写cookie

|  |
| --- |
| /\*\*设置cookie\*/  **function** **setCookie**(name,value){  **var** exp = **new** Date();  exp.setTime(exp.getTime() + 30\*24\*60\*60\*1000);  document.cookie = name + "="+ escape (value) + ";expires=" + exp.toGMTString();  }    /\*\*获取cookie\*/  **function** **getCookie**(name){  **var** arr = document.cookie.match(**new** RegExp("(^| )"+name+"=([^;]\*)(;|$)"));  **if**(arr != **null**){  **return** unescape(arr[2]);  }**else**{  **return** "";  }  }  /\*\*删除cookie\*/  **function** **delCookie**(name){  **var** exp = **new** Date();  exp.setTime(exp.getTime() - 1);  **var** cval=getCookie(name);  **if**(cval!=**null**) document.cookie= name + "="+cval+";expires="+exp.toGMTString();  } |

# jQuery插件

## 表单插件(Form Plugin)

|  |
| --- |
| 需要引入 **jquery.form.js**  **功能:** 获取表单数据, 重置表单项目, 使用Ajax提交数据等. |
| **获取表单数据**  **function** checkFiledValue(){  **var** aFieldValue = $("#myForm \*").fieldValue(); //返回myForm这个form中所有填写了数据的表单value值.  alert(aFieldValue.join());  } |
| **清除 和 重置**  $(**function**(){  $("input[type=button]:eq(0)").click(**function**(){  $("#myForm").clearForm();  });  $("input[type=button]:eq(1)").click(**function**(){  $("#myForm").resetForm();  });  }) |
| **Ajax提交表单**  $(**function**(){  $("input[type=button]:eq(0)").click(**function**(){  **var** options = {  target:"#myTargetDiv" //数据返回后插入到目标元内  }  $("#myForm").ajaxSubmit(options); //提交myForm这个表单, 数据和目标utl地址都中form表单里面定义了..  })  }); |

## UI插件

|  |
| --- |
| **鼠标拖拽特效**  $(**function**(){  **for**(**var** i=0; i<3; i++){  $(document.body).append($("<div class='block'>Div" + i.toString() + "</div>").css("opactity", 0.6)); //添加三个div块  }  $(".block").draggable(); //直接调用拖拽方法  }) |
| **拖入购物车**  $(**function**(){  $(".draggable").draggable({helper:"clone"}); //给class="draggable"的元素绑定可拖拽事件  $("#droppable-accept").droppable({  accept:**function**(draggable){  **return** $(draggable).hasClass("green"); //如果接收的元素class="green", 则返回true  },  drop:**function**(){ //当鼠标弹起后执行这个方法  $(**this**).append($("<div></div>").html("drop!"));  }  })  })  <div class=*"draggable red"*></div>  <div class=*"draggable green"*></div>  <div class=*"draggable-accept"* id=*"draggable-accept"*></div> |

## jQuery validator

### 自定义校验条件

|  |
| --- |
| jQuery.validator.addMethod("stringCheck", **function**(value, element) {  **return** **this**.optional(element) || /^[\u-\uFFE\w]+$/.test(value);  }, "只能包括中文字、英文字母、数字和下划线"); |
| jQuery.validator.addMethod("startCheck", **function**(value, element) {  **return** **this**.optional(element) || /^[\u4E00-\u9FA5|[a-z]|[A-Z]]$/.test(value);  }, "只能以汉字或者字母开头"); |
| //字母数字  jQuery.validator.addMethod("alnum", **function**(val, element) {  **return** **this**.optional(element) || /^[a-zA-Z0-9]+$/.test(val);  },"只能包括英文字母和数字"); |
| // 邮政编码验证  jQuery.validator.addMethod("zipcode", **function**(val, element) {  **var** tel = /^[0-9]{6}$/;  **return** **this**.optional(element) || (tel.test(val));  }, "请正确填写邮政编码"); |
| // 手机号码验证  jQuery.validator.addMethod("cellphone", **function**(val, element) {  **var** length = val.length;  **return** **this**.optional(element) || (length == 11 && /^(1\d{10})$/.test(val));  }, "请正确填写手机号码"); |

### 汉化提示文字

|  |
| --- |
| **使用方法: 复制到一个js文件中, 在网页中引入即可**  jQuery.extend(jQuery.validator.messages, {  required: "必选字段",  remote: "请修正该字段",  email: "请输入正确格式的电子邮件",  url: "请输入合法的网址",  date: "请输入合法的日期",  dateISO: "请输入合法的日期 (ISO).",  number: "请输入合法的数字",  digits: "只能输入整数",  creditcard: "请输入合法的信用卡号",  equalTo: "请再次输入相同的值",  accept: "请输入拥有合法后缀名的字符串",  maxlength: jQuery.validator.format("请输入一个长度最多是 {0} 的字符串"),  minlength: jQuery.validator.format("请输入一个长度最少是 {0} 的字符串"),  rangelength: jQuery.validator.format("请输入一个长度介于 {0} 和 {1} 之间的字符串"),  range: jQuery.validator.format("请输入一个介于 {0} 和 {1} 之间的值"),  max: jQuery.validator.format("请输入一个最大为 {0} 的值"),  min: jQuery.validator.format("请输入一个最小为 {0} 的值")  }); |

# jQuery事件

|  |
| --- |
| $("#cookImgDiv").mousemove(floatFile);  $("#submitFirst").click(confirmForm) |

# jQuery实例

## 淡入淡出浮动窗口

|  |
| --- |
|  |
| **JS代码**  //淡入淡出浮动窗口  function showwin(){  //alert("dddddddd");  //1.找到对应的div节点  var winNode = $("#win");  //2.让div显示出来  //方法1: 修改css里的 display:block;  //winNode.css("display", "block");  //方法2: 用jquery的show方法 , 缓慢打开窗口  //winNode.show("slow");  //方法3: 利用jquery的fadeIn方法, 淡入淡出  winNode.fadeIn("slow");  }  //隐藏窗口的方法  function hide(){  //找到对应的div节点  var winNode = $("#win");  //方法1: 修改css  //winNode.css("display", "none");  //方法2: 利用hide方法  // winNode.hide("slow");  //方法3: 淡出  winNode.fadeOut("slow");  } |
| **css代码**  @CHARSET "UTF-8";  #win{  border: 1px red solid;/\*定义上下左右所有边框\*/  width: 300px;  height: 200px;  position: absolute;/\*绝对定位, 需要首先设置 position: absolute; 然后\*/  left: 350px; /\*然后设置左边界和上边界\*/  top:100px;  display:none;/\*隐藏窗口\*/  }  #title{  background-color: #bffbaa;/\*背景颜色\*/  color:red; /\*字体颜色\*/  padding-left:3px /\*左内边距\*/    }  #content{  padding-top:5px;  padding-left:3px;  }  /\*关闭按钮的样式\*/  #close{  margin-left:158px; /\*向右移动\*/  cursor: pointer;  } |
| **HTML代码**  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  <html>  <head>  <title>淡入淡出浮动窗口</title>    <meta http-equiv=*"keywords"* content=*"keyword1,keyword2,keyword3"*>  <meta http-equiv=*"description"* content=*"this is my page"*>  <meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"*>    <link rel=*"stylesheet"* type=*"text/css"* href=*"css/win.css"*>  <script type=*"text/javascript"* src=*"js/jquery.js"*></script>  <script type=*"text/javascript"* src=*"js/jquerywin.js"*></script>    </head>    <body>  <a onclick="showwin()" href=*"#"*>显示浮动窗口</a><br>    <!-- 淡入淡出浮动窗口 -->  <div id=*"win"*>  <div id=*"title"*>我的标题栏!<span　id=*"close"* onclick="hide()">Ｘ</span></div>  <div id=*"content"*>我的窗口主体!</div>  </div>      </body>  </html> |

## 可编辑的表格

|  |
| --- |
| **HTML代码**  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  <html>  <head>  <title>可编辑的表格</title>    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">  <meta http-equiv="description" content="this is my page">  <meta http-equiv="content-type" content="text/html; charset=UTF-8">    <link rel="stylesheet" type="text/css" href="css/edit.css">  <script type="text/javascript" src="js/jquery.js"></script>  <script type="text/javascript" src="js/jqueryEdit.js"></script>    </head>    <body>    <table>  <tr>  <td>33333333</td>  <td>33333333</td>  <td>33333333</td>  <td>33333333</td>  </tr>  <tr>  <td>22222222</td>  <td>22222222</td>  <td>33333333</td>  <td>33333333</td>  </tr>  <tr>  <td>22222222</td>  <td>22222222</td>  <td>33333333</td>  <td>33333333</td>  </tr>  </table>  </body>  </html> |
| **CSS代码**  table, td{  border-collapse: collapse; /\*让相邻边框合并\*/  border:1px solid black;  }  td{  width:100px;  height:20px;  }  input {  width:100px;  height:18px;  } |
| **JS代码**  //在页面装载时, 让所有的td都拥有一个点击事件  $(document).ready(  **function**() {  **var** tds = $("td");//找到所有的td节点  tds.click(tdclick);//给所有的td节点增加点击事件  });  //td被点击的事件  **function** tdclick() {  **var** td = $(**this**);//把当前td保存起来  **var** text = td.text();//1.取出当前td中的文本内容保存起来  td.html(""); //清空td里面的内容, 也可以用td.empty();  **var** input = $("<input>");//3.建立一个文本框, 也就是input的元素节点  input.attr("value", text); //4.设置input的value等于原来保存的文本内容.  input.focus(**function**() { //文本框获得焦点事件  input.blur(**function** () { //文本框失去焦点事件, 保存修改  **var** inputnode = $(**this**);  **var** inputtext = inputnode.val();//2.获取当前文本框的内容  **var** tdNode = inputnode.parent();//3.找到td节点  tdNode.html(inputtext); //4.将保存的文本框的内容填充到td中  tdNode.click(tdclick);//5.让td重新拥有点击事件  })//4.5.2 让文本框响应失去焦点事件  input.keyup(**function**(event) { //4.5.1 让文本框可以响应键盘事件 /\*event是Firefox获取键值需要的参数\*/  **var** myEvent = event || window.event; //获取当前用户按下的键值, 解决不同浏览器获取事件对象的差异  **var** kcode = myEvent.keyCode;  **if** (kcode == 13) { //按下了回车键, 将保存修改  **var** inputnode = $(**this**);  **var** inputtext = inputnode.val();//2.获取当前文本框的内容  **var** tdNode = inputnode.parent();//3.清空td里面的内容  tdNode.html(inputtext); //4.将保存的文本框的内容填充到td中  tdNode.click(tdclick);//5.让td重新拥有点击事件  } **else** **if** (kcode == 27) { //按下了esc键, 将不保存修改  **var** inputnode = $(**this**);  **var** tdNode = inputnode.parent();//3.清空td里面的内容  tdNode.html(text); //4.将保存的表格文本内容填充到td中  tdNode.click(tdclick); //5.让td重新拥有点击事件  }  $("#name").focus(**function**() {  $("#name").addClass("focus");  });  });  }  );  td.append(input);//5.把文本框加入到td中  **var** inputdom = input.get(0); //5.5 使用js让文本框里面的文字被选中, 首先需要把jquery对象转换成js对象//0代表第一个节点  inputdom.select();  td.unbind("click");//6.需要清除td上的点击事件  }; |

## 显示/隐藏 子菜单

|  |
| --- |
| **js代码**  //页面装载时执行的方法.  $(document).ready(**function**(){//这里需要首先找到所有的主菜单  **var** as = $("ul > a");//然后给所有的主菜单注册点击事件  as.click(**function**(){ //这里需要找到当前dt中的dd, 然后让dd显示出来  **var** aNode = $(**this**); //获取当前被点击的a节点  **var** lis = aNode.nextAll("li");//找到当前dt节点的所有dd子节点  lis.toggle("show");//让子节点显示或隐藏  })  }) |
| **CSS代码**  li {  margin-left: *32px*;  display: *none*; /\* \*/  }  a {  list-style: *none*;  margin-left: *0px*;  }  ul.a:link{text-decoration:*none*; color:*Green*;}  ul.a:visited{text-decoration:*none*; color:*Green*;}  ul.a:hover{text-decoration:*none*; color:*Green*;}  ul.a:active{text-decoration:*none*; color:*Green*;}  /\*  dd{  margin-left:32px;  display:none; /\* \*/  }  \*/ |
| **html代码**  <ul><a href=*"#"*>出题</a>  <li><a href="${pageContext.request.contextPath}*/process/thesis/makeProject/searchProject.do?method=doSearch&department.pkDepartmentId=0&professional.pkProfessionalId=0&user.pkUserId=0&project.title=''*" target=*"right"*>查询已出的题目</a></li>  <li><a href="${pageContext.request.contextPath}*/process/thesis/makeProject/addNewProject.jsp*" target=*"right"*>我要出新的题目</a></li>  </ul>  <ul><a href=*"#"*>审核</a>  <li><a href="${pageContext.request.contextPath}*/process/thesis/check/waitChick.jsp*" target=*"right"*>待审核的题目</a></li>  <li><a href="${pageContext.request.contextPath}*/process/thesis/check/checked.jsp*" target=*"right"*>已审核的论题</a></li>  </ul> |

## 模仿 百度 google 的自动补全搜索框

|  |
| --- |
| **HTML页面**  <body>  <input type=*"text"* id=*"word"* />  <input type=*"button"* value=*"提交"* />  <div id=*"auto"*></div>  </body> |
| **servlet代码**  **package** **servlet**;  **import** java.io.IOException;  **import** javax.servlet.ServletException;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **public** **class** AutoComplete **extends** HttpServlet {  **public** **void** doGet(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  **this**.doPost(request, response);  }  **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  System.*out*.println("进入servlet了");  String word = request.getParameter("word");  request.setAttribute("word", word);  request.getRequestDispatcher("**wordxml.jsp**").forward(request, response);  }  } |
| **wordxml.jsp页面**  <%@ page contentType=*"text/xml; charset=utf-8"* language=*"java"*%>  <%  String word = (String)request.getAttribute("word");    %>  <words>  <%**if**("absolute".startsWith(word)){%>  <word>absolute</word>  <%} %>  <%**if**("aasdf".startsWith(word)){%>  <word>aasdf</word>  <%} %>  <%**if**("aew".startsWith(word)){%>  <word>aew</word>  <%} %>  <%**if**("bdsaf".startsWith(word)){%>  <word>bdsaf</word>  <%} %>  <%**if**("bsa".startsWith(word)){%>  <word>bsa</word>  <%} %>  <%**if**("bdssf".startsWith(word)){%>  <word>bdssf</word>  <%} %>  <%**if**("bsafas".startsWith(word)){%>  <word>bsafas</word>  <%} %>  </words> |
| **JS代码**  **var** timeoutId;  **var** highlightindex = -1;// 定义一个全局的高亮节点, 初始值为-1, 代表没有任何一个节点被高亮  **var** autoNodes;// autoNodes是一个数组  //var wordNode = $("#word");  $(document).ready(  **function**() {  **var** wordInput = $("#word");  **var** offset = wordInput.offset();  $("#auto")// 定义弹出框的属性  .hide()//  .css("border", "1px black solid")// 自动补全框最开始是隐藏的  .css("position", "absolute") // 绝对定位  .css("top", offset.top + wordInput.height() + 6 + "px")//  .css("left", offset.left + 1 + "px")//  .width(wordInput.width());  $("#word").keyup( **function**(event) { // <<<<<<输入文本框键盘弹起时的事件>>>>>>  **var** myEvent = event || window.event;  **var** keyCode = myEvent.keyCode;  **if** (keyCode >= 65 && keyCode <= 90// 如果按下的是字母键,  || keyCode == 8// delete键,  || keyCode == 46) { // backSpace键  **var** wordText = $("#word").val();// 获取文本框里的值  **if** (wordText != "") {// 如果文本内容不为空,  clearTimeout(timeoutId);//清除上次未完成的延时  timeoutId = setTimeout(**function**(){// 才发送到服务器  $.post("AutoComplete",{word :wordText},**function**(data) {  **var** jqueryObj = $(data);// 将dom对象data转换成jquery对象  **var** wordNodes = jqueryObj.find("word");// 找到所有的word节点  **var** autoNode = $("#auto");// 弹出框节点  autoNode.html("");  wordNodes.each( **function**(i) {// wordNodes.each遍历wordNodes,  // wordNodes是一个对象  **var** wordNode = $(**this**);// 取出单词内容  **var** newDivNode = $("<div>").attr("id", i);// 新建div节点,  newDivNode.html(wordNode.text()).appendTo(autoNode);// 将单词内容加到新建的div节点中, 将新建的节点加入到弹出框节点中  newDivNode.mouseover(**function**(){//鼠标经过事件  **if**(highlightindex != -1){  $("auto").children("div").eq(highlightindex).css("background-color", "white");  }  highlightindex = $(**this**).attr("id");  $(**this**).css("background-color", "red");  });  newDivNode.mouseout(**function**(){//鼠标离开事件  $(**this**).css("background-color", "white");  });  newDivNode.click(**function**(){//鼠标点击事件  **var** comText = $(**this**).text();//取出高亮文本  highlightindex = -1;  $("#auto").hide();  $("#word").val(comText);//把高亮处的文本赋值给输入框  });  })  **if** (wordNodes.length > 0) {  $("#auto").show();  } **else** {  $("#auto").hide();  highlightindex = -1;  }  }, "xml");  }, 500);  } **else** {  $("#auto").hide();  highlightindex = -1;  }  } **else** **if**(keyCode == 38 || keyCode == 40){  **if** (keyCode == 38) { // 如果按下的是上箭头,  autoNodes = $("#auto").children("div");// autoNodes是一个数组    **if**(highlightindex != -1){// 如果原来存在高亮节点  autoNodes.eq(highlightindex).css("background-color", "white");// 则将背景色改成白色  highlightindex--;  }    **if**(highlightindex == -1){//如果原来不存在高亮点  highlightindex = autoNodes.length - 1;//按下上箭头, highlightindex修改为数组最大值  }  autoNodes.eq(highlightindex).css("background-color", "red");    }**if** (keyCode == 40) { // 如果按下的是下箭头,    autoNodes = $("#auto").children("div");  **if**(highlightindex != -1){  autoNodes.eq(highlightindex).css("background-color", "white");  }  highlightindex++; // 数组元素脚标增大  **if**(highlightindex == autoNodes.length){  highlightindex = 0;  }  autoNodes.eq(highlightindex).css("background-color", "red");  }  } **else** **if** (keyCode == 13) { // 如果按下的是回车  **if**(highlightindex != -1){// 下拉框有高亮内容  **var** comText = autoNodes.eq(highlightindex).text();//取出高亮文本  highlightindex = -1;  $("#auto").hide();  $("#word").val(comText);//把高亮处的文本赋值给输入框  }**else**{// 下拉框没有高亮内容  alert("文本框中的【" + $("#word").val() + "】被提交了");  $("#word").get(0).blur();//让文本框失去焦点  }  }  })  $("input[type='button']").click( **function**() {// <<<<<<提交按钮>>>>>>  alert("文本框中的【" + $("#word").val() + "】被提交了");  })  }) |

## 模拟股票动态信息

|  |
| --- |
| **eval(**  [  {name: "上证指数",id: "300001",yes: 3000.0,tod: 2990.1,now: 3006.6,},  {name: "浦发银行",id: "000001",yes: 23.22,tod: 23.5,now: 23.04,}  ]  **);**  说明: eval('需要验证的js代码'); |
| //采用json的数据格式返回股票信息. |
| //方法一: 采用数组方式回传两个股票对象  //[{}, {}] //JS数组用[]表示, 里面包含两个对象, 一个对象有5个属性值.  // 以下是伪代码  // [{  // name: "szzs.getName()", //字符串需要加双引号, 数值就不用加双引号了.  // id: "szzs.getId()",  // yes: szzs.getYesterday(),  // tod: szzs.getToday(),  // now: szzs.getNow(),  // }, {  // name: "pfyh.getName()",  // id: "pfyh.getId()",  // yes: pfyh.getYesterday(),  // tod: pfyh.getToday(),  // now: pfyh.getNow(),  // }]    StringBuilder builder = **new** StringBuilder();  builder.append("[{")  .append("name: \"" + szzs.getName()+ "\",")  .append("id: \"" + szzs.getId() + "\",")  .append("yes: " + szzs.getYesterday()+ ",")  .append("tod: " + szzs.getToday()+ ",")  .append("now: " + szzs.getNow()+ ",")  .append("}, {")  .append("name: \"" + pfyh.getName()+ "\",")  .append("id: \"" + pfyh.getId()+ "\",")  .append("yes: " + pfyh.getYesterday()+ ",")  .append("tod: " + pfyh.getToday()+ ",")  .append("now: " + pfyh.getNow()+ ",")  .append("}]");  out.print(builder.toString()); |
| //方法二: 采用json对象方式回传两个股票对象  //如果回传表示对象的json, 需要在最外层加上一个括号, 否则页面会解析出错, 或者修改println为print  // {szzs.getId():{  // name: szzs.getName(),  // yes: szzs.getYesterday(),  // tod: szzs.getToday(),  // now: szzs.getNow(),  // },  // pfyh.getId():{  // name: pfyh.getName(),  // yes: pfyh.getYesterday(),  // tod: pfyh.getToday(),  // now: pfyh.getNow(),  // }  // }  StringBuilder builder = **new** StringBuilder();  builder.append("")  .append("({")    .append(pfyh.getId() + ":{")  .append("name: \"" + pfyh.getName() + "\",")  .append("yes: " + pfyh.getYesterday() + ",")  .append("tod: " + pfyh.getToday() + ",")  .append("now: " + pfyh.getNow())  .append("}")    .append(",")    .append(szzs.getId() + ":{")  .append("name: \"" + szzs.getName() + "\",")  .append("yes: " + szzs.getYesterday() + ",")  .append("tod: " + szzs.getToday() + ",")  .append("now: " + szzs.getNow())  .append("}")    .append("})");  out.print(builder.toString()); |
| **JS代码**  **var** obj;// 股票对象  **var** sid;  // 期望进入页面后就可以开始从服务器断获取数据, 然后显示股票价格  $(document).ready( **function**() {  setInterval(getInfo, 1000); // 每隔一秒钟和服务器交互一次, 用户不用刷新页面.  **var** stockNode = $("#stock").css("border", "1px solid black")// 边框属性  .width("110px") // 宽度  .css("position", "absolute") // 绝对定位  .css("z-index", "99").css("background-color", "#fff");  $("#stock").hide();// 页面载入时隐藏弹出框  **var** as = $("a"); // 查找所有超链接 节点  as.mouseover(**function**(event){  **var** aNode = $(**this**);  **var** divNode = aNode.parent();// 当前鼠标下的a节点的父节点  sid = divNode.attr("id");// 返回 ID属性的值  updatediv();  // 控制弹出框的位置  // 效果(一):显示在当前链接的正下方  // var offset = aNode.offset();// 返回当前链接左边界和上边界简单对象, 这个对象有两个属性,top和left  // stockNode.css("left", offset.left + "px").css("top", offset.top + aNode.height() + "px");    // 效果(二):显示在鼠标的正下方, 横纵都加5像素  **var** myEvent = event || window.event;  stockNode.css("left", myEvent.clientX + 10 + "px").css("top", myEvent.clientY + 20 + "px");    stockNode.show();  });// 给所有的a节点注册mouseover事件  as.mouseout( **function**() {  stockNode.hide();  });  })  // 同步更新弹出框里的内容  **function** updatediv() {  **var** stockobj = obj[sid];// 当前股票对象  **for** ( **var** proname **in** stockobj) {// 把股票对象里的数据填充到弹出div中去  **if** (proname != "name") { // 股票名不用取出来  $("#" + proname).children("span").html(stockobj[proname]);  }  }  }  // 从服务器端获取数据并显示在页面上  **function** getInfo() {  // 1. 向服务器发起请求, 获取数据.  $.get("GetStocksInfo", **null**, **function**(data) {  // 2. 接收并解析数据  obj = eval(data);//把服务器返回的json对象转换成js对象  //这里还可以写成 obj=data; 然后在本函数花括号后面写上 ,"json" 意思是给get方法添加了一个可选参数, 这样jquery会自动将服务器返回的json类型的对象转换成js对象, 说的就是data  // 2.1 获取两只股票的当前指数信息.  **var** szzs = obj["333"];  **var** pfyh = obj["555"];  // 2.2 找到页面中对应的节点, 然后天聪最新的股票价格  **var** span3 = $("#333").children("span").html(szzs.now);  **if** (szzs.now > szzs.yes) {  // 当前价格大于昨天收盘, 则显示为红色  span3.css("color", "red");  } **else** {  span3.css("color", "green");  }  **var** span4 = $("#555").children("span").html(pfyh.now);  **if** (pfyh.now > pfyh.yes) { // 当前价格大于昨天收盘, 则显示为红色  span4.css("color", "red");  } **else** {  span4.css("color", "green");  }  updatediv(); // 数据获取完毕后, 同步更新弹出框里的内容  })  }  } |
| **HTML代码**  <div id=*"333"*><a href=*"#"*>上证指数:</a><span></span></div>  <div id=*"555"*><a href=*"#"*>浦发银行:</a><span></span></div>  <br> |

## 在DOM载入后开始执行事件

|  |
| --- |
| $(document).ready(function() {  // do stuff when DOM is ready  }); |
| $("a") 是一个jQuery选择器(selector),在这里，它选择所有的a标签（译者Keel注：即<a></a>），$号 是 jQuery “类”(jQuery "class")的一个别称，因此$()构造了一个新的jQuery 对象(jQuery object)。函数 click() 是这个jQuery对象的一个方法，它绑定了一个单击事件到所有选中的标签(这里是所有的a标签),并在事件触发时执行了它所提供的alert方法. |

## 选择html的 elements

|  |
| --- |
| jquery的写法:  第一种: 用CSS和Xpath选择器联合起来形成一个字符串来传送到jQuery的构造器（如：$("div > ul a")）；  第二种: 用jQuery对象的几个methods(方法) |
| javascript写法:  document.getElementById("orderedlist") |

## 隔行变色的表格

|  |
| --- |
| <script type=*"text/javascript"* src=*"jquery.js"*></script>  <script type=*"text/javascript"*>  <!--  window.onload = **function**(){  **var** oTable = document.getElementById("oTable");  **for**(**var** i=0; i<oTable.rows.length; i++){  **if**(i%2 == 0){  oTable.rows[i].className = "event";  }**else**{  oTable.rows[i].className = "odd";  }  }  }  //-->  </script>  <style type=*"text/css"*>  <!--  .event {  background-color: *#a5e5aa*;  }  .odd {  background-color: *#aaaaaa*;  }  -->  </style> |

## 给某个节点注册事件

|  |
| --- |
| $(**function**(){    $("#form1").submit(**function**(){  **var** str=escape($("#str").val());  alert(str);  dialog("我的标题","url:post?http://www.xij.cn/blog/wp-content/uploads/2008/08/dialog/test.html?str="+str+"","200px","auto","form");  **return** **false**;  });  }); |

## dtree(树)

|  |
| --- |
| **html页面代码**  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>  <%@ include file=*"/WEB-INF/jsp/common/include.jsp"*%>  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  <html>  <head>  <title>选择部门</title>  <script src='<c:url value=*"/javascripts/dtree.js"*/>'></script>  <script src="<c:url value=*'/javascripts/jquery-1.4.2.js'* />" type=*"text/javascript"*></script>  <link rel=*"StyleSheet"* href="<c:url value=*"/stylesheets/dtree.css"*/>" type=*"text/css"* />  <script type=*"text/javascript"*>  **var** id;  **var** name;  **function** set(id,name){  **this**.id=id;  **this**.name=name;  }  tree = **new** dTree('tree');//创建一个对象.  $.ajax({  url:'<c:url value="/department/load.do"/>',  type:'post', //数据发送方式  dataType:'xml', //接受数据格式  error:**function**(json){  alert( "not lived!");  },  async: **false** ,//同步方式  success: **function**(xml){  $(xml).find("node").each(**function**(){  **var** nodeId=$(**this**).attr("nodeId");  **var** parentId=$(**this**).attr("parentId");  **var** nodeName=$(**this**).text();  **var** clickStr = "javascript:set('+nodeId+','+nodeName+')" ;  tree.add(nodeId,parentId,nodeName+"<input style='width:20px;' type='radio' name='custDeptTree' value='"+nodeId+"' onclick='javascript:set(\""+nodeId+"\",\""+nodeName+"\")'>","","","workspace","","",**false**);  });  }  });  document.write(tree);  **function** setDepartment(){  opener.document.forms[0].departmentName.value = name;  opener.document.forms[0].departmentId.value = id;  window.close();  }  </script>  </head>  <body>  <input type=*"button"* value=*"确定"* onclick="setDepartment();">  <input type=*"button"* value=*"取消"* onclick="window.close();">  </body>  </html> |
| **Java代码**  **public** ModelAndView load(HttpServletRequest request, HttpServletResponse response) **throws** Exception {  request.setCharacterEncoding("utf-8");  // 取得当前企业  Long orgId = Long.*parseLong*((String) request.getSession().getAttribute(WebConfig.*ORG\_ID*));  EnterpriseOrg enterpriseOrg = (EnterpriseOrg) **this**.enterpriseOrgService.getByPrimaryKey(EnterpriseOrg.**class**, orgId);  // 取得当前企业下的部门  List<EnterpriseDepartment> departments = **this**.enterpriseDepartmentService.listDepartmentByOrg(orgId);  response.setContentType("text/xml;charset=utf-8");  PrintWriter pw = response.getWriter();  pw.println("<?xml version=\"1.0\" encoding=\"UTF-8\"?>");  pw.println("<nodes>");  pw.println("<node nodeId='0' parentId='-1' >" + enterpriseOrg.getName() + "</node>");  **if** (departments.size() > 0) {  **for** (**int** i = 0; i < departments.size(); i++) {  EnterpriseDepartment enterpriseDepartment = departments.get(i);  Long parentId = **new** Long(0);  String name = "未命名";  **if** (enterpriseDepartment.getParent() != **null** && !"".equals(enterpriseDepartment.getParent())) {  parentId = enterpriseDepartment.getParent().getId();  }  **if** (enterpriseDepartment.getName() != **null** && !"".equals(enterpriseDepartment.getName())) {  name = enterpriseDepartment.getName();  }  pw.println("<node nodeId='" + enterpriseDepartment.getId() + "' parentId='" + parentId + "' >" + name + "</node>");  }  }  pw.println("</nodes>");  **return** **null**;  } |

## JQuery获取浏览器窗口宽高，文档宽高

|  |
| --- |
| $(document).ready(function()  {  alert($(window).height()); //浏览器当前窗口可视区域高度  alert($(window).width()); //浏览器当前窗口可视区域宽度  alert($(document).height()); //浏览器当前窗口文档的高度  alert($(document).width());//浏览器当前窗口文档对象宽度  alert($(document.body).height());//浏览器当前窗口文档body的高度  alert($(document.body).width());//浏览器当前窗口文档body的高度  alert($(document.body).outerHeight(true));//浏览器当前窗口文档body的总高度 包括border padding margin  alert($(document.body).outerWidth(true));//浏览器当前窗口文档body的总宽度 包括border padding margin    } |

)

</script>

## 倒序选择多选框

|  |
| --- |
| <html>  <head>  <title></title>  <script type=*"text/javascript"* src=*"jquery-1.4.2.js"*></script>  <script type=*"text/javascript"*>  <!--  $(**function**(){  **var** haha = $("#haha").prev(".ds");  //alert(haha.html());    //1. 载入页面后, 除了最后一个checkbox, 其它都设置不可操作  $(".cooking").not($(".cooking:last")).attr("disabled","disabled");    $(".cooking").change(**function**(){  **if**($(**this**) .is(":checked")){ //或者$(this).prop("checked")  //2. 当选中一个checkbox, 上面的checkbox才可操作  **var** index = 0;  **var** thisValue = $(**this**).val();  $(".cooking").each(**function**(i){  **if**(**this**.value == thisValue){  index--;  $(".cooking:eq("+index+")").removeAttr("disabled");  **return**;  }  index++;  });    }**else**{  //3. 当取消一个checkbox, 上面的checkbox都被取消切, 且都不可操作    **var** index = 0;  **var** thisValue = $(**this**).val();  $(".cooking").each(**function**(i){  **if**(**this**.value == thisValue){  //取消选择  $(".cooking:lt("+index+")").removeAttr("checked");  //不可操作  $(".cooking:lt("+index+")").attr("disabled","disabled");  }  index++;  });    }  });  })    //-->  </script>  </head>  <body>  <table>  <tr>  <td><input type=*"checkbox"* class=*"cooking"* name=*"cooking"* value=*"4"*/></td>  <td>洗碗</td>  </tr>  <tr>  <td><input type=*"checkbox"* class=*"cooking"* name=*"cooking"* value=*"3"*/></td>  <td>吃饭</td>  </tr>  <tr>  <td><input type=*"checkbox"* class=*"cooking"* name=*"cooking"* value=*"2"*/></td>  <td>做饭</td>  </tr>  <tr>  <td><input type=*"checkbox"* class=*"cooking"* name=*"cooking"* value=*"1"*/></td>  <td>买菜</td>  </tr>  </table>  </body>  </html> |

## 淡入淡出提示div(自己封装的)

|  |
| --- |
| /\*\*  \* 淡入淡出提示div  \* 使用方法: JS调用notice(message)方法, 参数message为提示文字.  \*/  **function** **notice**(message){  $("body").prepend('<div id="noticeDiv" style="color:red; font-weight:bolder; text-align:center; margin:auto; vertical-align:bottom; padding-top:5px; display:block; height:20px; letter-spacing:4; width:300px; background-color:#FFF68F; position:absolute; top:2px; z-index:999;"></div>');    **var** noticeDiv = $("#noticeDiv");  **var** left = (document.body.clientWidth+noticeDiv.width() - noticeDiv.width())/2 + document.body.scrollLeft - 180 + 'px';    noticeDiv.css("left", left);  noticeDiv.html(message);  noticeDiv.fadeIn(500);  noticeDiv.fadeOut(1000000);  }  $(**function**(){  **if** (document.cookie == "") {  setCookie("isBack", "0");  }  **var** isBack = getCookie("isBack");//读取cookie中名为isBack的值  **if**(isBack == 0){  **var** message = $("#messageDiv").html();  **if**(message != **null**){  notice(message);  }  }**else**{  setCookie("isBack", "0"); //每次进入页面, 显示完提示信息后. 都设置标志isBack为0, 表示不是返回的页面  }  })  /\*\*返回上一个页面调用的方法\*/  **function** **getBack**(){  setCookie("isBack", "1"); //设置cookie名为isBack的值为1, 表示执行了返回动作)  history.go(-1);  }  /\*\*设置cookie\*/  **function** **setCookie**(name,value){  **var** exp = **new** Date();  exp.setTime(exp.getTime() + 30\*24\*60\*60\*1000);  document.cookie = name + "="+ escape (value) + ";expires=" + exp.toGMTString();  }    /\*\*获取cookie\*/  **function** **getCookie**(name){  **var** arr = document.cookie.match(**new** RegExp("(^| )"+name+"=([^;]\*)(;|$)"));  **if**(arr != **null**){  **return** unescape(arr[2]);  }**else**{  **return** "";  }  }  /\*\*删除cookie\*/  **function** **delCookie**(name){  **var** exp = **new** Date();  exp.setTime(exp.getTime() - 1);  **var** cval=getCookie(name);  **if**(cval!=**null**) document.cookie= name + "="+cval+";expires="+exp.toGMTString();  } |

## jquery 判断 checkbox 选中

|  |
| --- |
| **jquery对象方法**  **if** ($("#check").is(":checked")) { //或者 $("#check").prop("checked")  //alert("选中了");  } |
| **DOM对象方法**  var v = document.getElimentById("man");  alert(v.checked); |

## jquery 让 checkbox 选中和取消选中

|  |
| --- |
| $("#select\_all").change(**function**(){  **if**($(**this**).is(":checked")){  $(".checkbox\_item").prop("checked", **true**);  }**else**{  $(".checkbox\_item").prop("checked", **false**);  }  }); |

## 单行文本框恢复默认值

|  |
| --- |
| $('#hpt\_change\_order\_region\_first').val($('#hpt\_change\_order\_region\_first')[0].defaultValue) |

## 单行文本框disabled切换

|  |
| --- |
| $('#hpt\_change\_order\_region\_first').removeAttr('disabled'); |
| $('#hpt\_change\_order\_region\_first').attr('disabled', true); |

## 单选按钮切换事件

|  |
| --- |
| $(".bindCardCheckbox").change(function() {  var selectedvalue = $(".bindCardCheckbox:checked").val();  if (selectedvalue == 1) {    } else {  }  }); |

## 获取/设置 下拉选择框值

|  |  |
| --- | --- |
| **获取value** | **var** opennessCode = $("#openness option:selected").val();  **var** opennessCode = $("#openness").val(); |
| **获取select选中的文字描述** | **var** opennessCode = $("#openness option:selected").text(); |
| 设置 | $("#openness").val(1); |
| 重置select的值 | $('#package\_id').val( $('#package\_id').prop('defaultSelected') ); |

## 好友左移右移

|  |
| --- |
| /\*\*双击右移\*/  $("#left").children().dblclick(**function**(){  $("#right").append($(**this**));  });  /\*\*双击左移\*/  $("#right").children().dblclick(**function**(){  $("#left").append($(**this**));  });  /\*\*右移\*/  $("#toRight").click(**function**(){  **var** stus = $("#left option:selected");  $("#right").append(stus);  });  /\*\*左移\*/  $("#toLeft").click(**function**(){  **var** stus = $("#right option:selected");  $("#left").append(stus);  });  /\*\*全部右移\*/  $("#allToRight").click(**function**(){  **var** stus = $("#left option");  $("#right").append(stus);  });  /\*\*全部左移\*/  $("#allToLeft").click(**function**(){  **var** stus = $("#right option");  $("#left").append(stus);  }); |

## append/appendTo/prepend/prependTo的区别

|  |  |  |
| --- | --- | --- |
| 追加 | $("p").append("<b>Hello</b>"); | 内部追加 |
| $("p").appendTo("div"); | 把节点$("p")追加到节点"div" |
|  |  |  |
| 前面插入 | $("p").prepend("<b>Hello</b>"); | 内部前方插入 |
| $("p").prependTo("#foo"); | 把节点$("p")插入到节点"div"内部前方 |

## 外部插入

|  |
| --- |
| （1）after()  　　$(selector).after(content);  　　$(selector).after(function(index [,html]){...}); |
| （2）insetAfter()  　　$(content).insertAfter(selector); |
| （3）before()  　　$(selector).before(content);  　　$(selector).before(function(index [,html]){...}) |
| （4）insertBefore()  　　$(content).insertBefore(selector); |

## 点击过的菜单链接设置特别的样式

|  |
| --- |
| <script type=*"text/javascript"*>  $(**function**(){  **var** apps = $(".sidebar\_app li a");  **var** url = document.location+"";  **for**(**var** i=0;i<apps.length;i++){  **var** app = apps[i];  **if** (url.indexOf(app)>=0){  **var** appObj = $(app); //HTML对象转jquery对象  appObj.addClass("blod");  }  }  })  </script> |
| 效果 |

## 重置form

|  |
| --- |
| $(':input','#myform')  .not(':button, :submit, :reset, :hidden')  .val('')  .removeAttr('checked')  .removeAttr('selected'); |

## 校验图片文件类型

|  |
| --- |
| types = /(\.|\/)(gif|jpe?g|png)$/i  if types.test(file.type) || types.test(file.name){  alert("1");  } |