1. 下载jquery.ui ：<http://jqueryui.com/>

在这里还可以选择下载不同样式的css效果

2、在界面里引入jquery.ui，以及css，但是要先引入原生jquery，记得把己写的js和css放在后面

<script type="text/javascript" src="js/jquery-1.10.1.js"></script>

<script type="text/javascript" src="js/jquery-ui.js"></script>

<script type="text/javascript" src="js/index.js"></script>

<link rel="stylesheet" type="text/css" href="css/redmond/jquery-ui.css"/>

**Dialog（）弹出框方法**

Dialog方法是弹出对话框方法，下面是一些解析

**简单弹窗**

<div id="reg" title="会员注册"> //title会变成标题

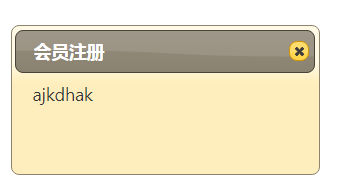
表单区--会员注册

</div>

$('#reg').dialog();

$('#reg').dialog()意味着将id为reg的div制造成弹窗，内容变成“表单区--会员注册”

头变成tittle部分的内容，并且无条件弹出

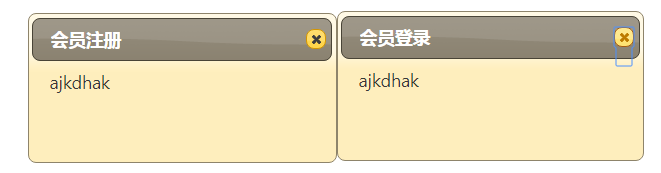


**同时弹窗、**

$('#reg').dialog();

$('#login').dialog();

同时弹出两个窗口，把id为reg的div制造成弹窗，id为login的div制造成弹窗



**二． 修改 dialog 样式**

在弹出的 dialog 对话框中，在火狐浏览器中打开 Firebug 或者右击->查看元素。这样，

我们可以看看 dialog 的样式，根据样式进行修改。我们为了和网站主题符合，对 dialog 的标

题背景进行修改。

//无须修改 ui 里的 CSS，直接用 style.css 替代掉

.ui-widget-header {

background:url(../img/ui\_header\_bg.png);

}

注意：其他修改方案类似。

**事件触发弹窗**

$(‘reag’).click(function()

$('#reg').dialog()

});

**Dialog键值对方式**

$('#reg\_a').click(function(){

$('.zhuce').dialog({

title : '注册知问',

//设置弹窗的标题，这里的级别大于在html行内的title

buttons : {'按钮' : function () {alert('注册知问')}},

//在弹窗里添加按钮，文字为‘按钮’，点击按钮后执行函数

height:500,

width:500,

//设置弹出框的长度和宽度

minWidth : 300,

minHeight : 300,

maxWidth : 800,

maxHeight : 600,

//设置弹出框的最大长度和最大宽度

show : 'blind',

hide : 'blind',

//设置弹出和消失的样式

modal : true,

//设置锁屏遮罩

});

});

**Show和hide的样式表**

blind 对话框从顶部显示或消失

bounce 对话框断断续续地显示或消失，垂直运动

clip 对话框从中心垂直地显示或消失

slide 对话框从左边显示或消失

drop 对话框从左边显示或消失，有透明度变化

fold 对话框从左上角显示或消失

highlight 对话框显示或消失，伴随着透明度和背景色的变化

puff 对话框从中心开始缩放。显示时“收缩”，消失时“生长”

scale 对话框从中心开始缩放。显示时“生长”，消失时“收缩”

pulsate 对话框以闪烁形式显示或消失

**隐藏弹窗**

$('#reg').dialog({

autoOpen : false,

});

默认为 true，调用 dialog()方法时就会打开对话框；

如果为 false，对话框不可见，但对话框已创建，可

以通过 dialog('open')才能可见。

**弹窗事件**

dialogfocus 得到焦点时触发

dialogopen 显示时触发

dialogbeforeclose 将要关闭时触发

dialogclose 关闭时触发

dialogdrag 每一次移动时触发

dialogdragstart 开始移动时触发

dialogdragstop 移动结束后触发

dialogresize 每次调整大小时触发

dialogresizestart 开始调整大小时触发

dialogresizestop 结束调整大小时触发

$('#reg').on('dialogclose', function () {

alert('关闭');

});

dialog('open') jQuery 对象 打开对话框

dialog('close') jQuery 对象 关闭对话框

dialog('destroy') jQuery 对象 删除对话框，直接阻断了 dialog。

dialog('isOpen') 布尔值 判断对话框是否打开状态

dialog('widget') jQuery 对象 获取对话框的 jQuery 对象

**按钮（.button()）**

按钮（button），可以给生硬的原生按钮或者文本提供更多丰富多彩的外观。它不单单可以设置按钮或文本，还可以设置单选按钮和多选按钮。

$('#search\_button').button({

disabled : false,//禁用按钮，默认为true

icons : {

primary : 'ui-icon-search',//修改按钮前的图标，可以在jquery的官网查看api文档来获取

secondary : 'ui-icon-search' //文字后面加图标

},label : '查找',//修改按钮上的文字

text : false,//把按钮上的字去掉

});

button('disable') jQuery 对象 禁用按钮

button('enable') jQuery 对象 启用按钮

button('destroy') jQuery 对象 删除按钮，直接阻断了 button。

button('refresh') jQuery 对象 更新按钮布局。

**单选框、复选框（用得少未加实验）**

button 按钮不但可以设置普通的按钮，对于单选框、复选框同样有效。

//HTML 单选框

<input type="radio" name="sex" value="male" id="male">

<label for="male">男</label>

</input>

<input type="radio" name="sex" value="female" id="female">

<label for="female">女</label>

</input>

//jQuery 单选框

$('#reg input[type=radio]').button();

//jQuery 单选框改

$('#reg').buttonset(); //HTML部分做成一行即可

//HTML 复选框

<input type="checkbox" name="color" value="red" id="red">

<label for="red">红</label>

</input>

<input type="checkbox" name="color" value="green" id="green">

<label for="green">绿</label>

</input>

<input type="checkbox" name="color" value="yellow" id="yellow">

<label for="yellow">黄</label></input>

<input type="checkbox" name="color" value="orange" id="orange">

<label for="orange">橙</label>

</input>

//jQuery 复选框

$('#reg input[type=radio]').button();

//jQuery 复选框改

$('#reg').buttonset();

将单选复选框做成方框

$('#reg').buttonset();将单选多选按钮进行设置

**应用日历方法datepicker()（先在html用readonly="readonly"）**

$('#date').datepicker();（日期选择期）将日历卡引入，一般在行内加readonly，让这个框不能填只能选

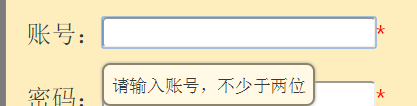
<input type="text" id="date" readonly="readonly"/>



**将tittle用.tooltip()做成提示**

1、在html文件对reg下的input加入一个tittle属性，

2、$('#reg input[title]').tooltip();然后tittle上的文字会设置成提示信息



**调用 tooltip() 方法提示**

在调用 tooltip()方法之前，首先需要针对元素设置相应 title 属性。

Html

<input type="text" name="user" class="text" id="user" title="请输入帐号，不小于 2 位！" />

Js

$('#user').tooltip();

$(function () {

$('input').tooltip({

content : '改变文字', 提示的文字

items : 'input',

tooltipClass : 'reg\_tooltip',

position:{ //设置提示框位置

my:'left center',

at:'right+5 center',

},

show : 'blind', 提示框的效果

hide : 'blind'

});

});

dialogopen 显示时触发

tooltipclose 每一次移动时触发

$('#reg').on('tooltipopen', function () {

alert('打开时触发！');

});

**调用 autocomplete() 方法 自动补全**

$('#email').autocomplete({

source : ['aaa@163.com', 'bbb@163.com', ['ccc@163.com'],](mailto:'ccc@163.com'],) 提示库

disabled : false, 是否禁用提示

minLength : 2, 输入最小长度

delay : 50, 默认为 300 毫秒，延迟显示设置

autoFocus : true, 第一个默认选中

})

**可能会出现提示框被盖住，那么就用**

.ui-autocomplete {

z-index: 9999999;

}

**自动补全的整体方法**

var hosts = ["qq.com", "126.com", "yahu.com", "yahoo.com", "cnblogs.com"]; //补全库

$("#email").autocomplete({

autoFocus: true,

source: function(request, response) {

var term = request.term, //request.term为输入的字符串

ix = term.indexOf("@"),

name = term, // 用户名

host = "", // 域名

result = []; // 结果

result.push(term);

// result.push({ label: term, value: term }); // json格式

if (ix > -1) {

name = term.slice(0, ix);

host = term.slice(ix + 1);

}

if (name) {

var findedHosts = (host ? $.grep(hosts, function(value) {

return value.indexOf(host) > -1;

}) : hosts),

findedResults = $.map(findedHosts, function(value) {

return name + "@" + value; //返回字符串格式

// return { label: name + " @ " + value, value: name + "@" + value }; // json格式

});

result = result.concat($.makeArray(findedResults));

}

response(result);//呈现结果

}

});

**一些字符串操作方法**

String.slice(5);截取第五个字符串以后的字符

String.slice(0, ix);截取0到ix内的字符串

$.grep(array, function (value, index) {})；

Array是待匹配数组，等于array中的遍历，index是对应的下标

String.indexOf(host)在字符串里匹配是否有带有host的字符，如果匹配成功，返回相应的位置，否则返回-1，indexOf(host)本身可作为匹配成功的元字符串输出

$.map(array, function (value index){}),value是array的遍历，index是下标，function用于筛选想要的东西

**日历ui，datepicker（）**

$('#date').datepicker();（日期选择期）将日历卡引入，一般在行内加readonly，让这个框不能填只能选

<input type="text" id="date" readonly="readonly"/>

**如果显示英文格式，那么在jquery的源文件**

Query UI\jquery-ui-1.10.3.custom\development-bundle\ui\i18n下找到jquery.ui.datepicker-zh-CN.js然后把里面的代码复制到juqery里面



**选项卡tabs(jquery内置方法)**

选项卡（tab），是一种能提供给用户在同一个页面切换不同内容的 UI。 尤其是在页面布局紧凑的页面上，提供了非常好的用户体验



选项卡ui对html和js要相互配合

**写在一个html文件的选项卡ui**

<div id="tabs">

<ul>

<li><a href="#tabs1">tab1</a></li>

<li><a href="#tabs2">tab2</a></li>

<li><a href="#tabs3">tab3</a></li>

</ul>

<div id="tabs1">tab1-content</div>

<div id="tabs2">tab2-content</div>

<div id="tabs3">tab3-content</div>

</div>

//通过href="#tabs1与d="tabs1相连接,实现功能

//jQuery 部分

$('#tabs').tabs();

**通过引入别的html文件(内部ajax会自动实现)**

<div id="tabs">

<ul>

<li><a href="tabs1.html">tab1</a></li>

<li><a href="tabs2.html">tab2</a></li>

<li><a href="tabs3.html">tab3</a></li>

</ul>

</div>

通过主文件分别引入了三个html文件，他会把东西引进一个默认的div里面,通过<a href="tabs1.html">tab1</a>连接

//jQuery 部分

$('#tabs').tabs();

其他功能

$('#tabs').tabs({

collapsible : true,//当设置为true时为允许折叠选项卡内容

disabled : [0], //禁用选项卡

event : 'mouseover',//将触发事件由click改为mouseover

active : false, //选择那个为页面初始时的页面

heightStyle : 'content',//自动填充一部分空白

hide : true,

show : true,

});

//获取选项卡 jQuery 对象

$('#tabs').tabs('widget');

//重载指定选项卡内容、、非常重要，页面自动ajax重载

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

$('#tabs').tabs('load', 0);

});

tabsload Ajax 加载后触发

tabsbeforeload Ajax 加载前触发

tabsactivate 选项卡切换时触发

tabsbeforeactivate 选项卡切换前触发

**折叠菜单（accordion jquery.ui自带）**

折叠菜单（accordion），和选项卡一样也是一种在同一个页面上切换不同内容的功能

UI。它和选项卡的使用几乎没有什么太大区别，只是显示的效果有所差异罢了。

<div id="accordion">

<h1>菜单 1</h1>

<div>内容 1</div>

<h1>菜单 2</h1>

<div>内容 2</div>

<h1>菜单 3</h1>

<div>内容 3</div>

</div>

//jQuery 部分

$('#accordion').accordion();

$('#accordion').accordion({

collapsible : true, //允许折叠,自我关闭

event : 'mouseover',//改变触发事件

active : 1,// 默认初始显示

active : true,//

heightStyle : 'content',//默认填充

//改变图表

icons: {

"header": "ui-icon-plus",

"activeHeader": "ui-icon-minus",

},

});

accordion('disable') jQuery 对象 禁用折叠菜单

accordion('enable') jQuery 对象 启用折叠菜单

accordion('widget') jQuery 对象 获取折叠菜单的 jQuery 对象

accordion('destroy') jQuery 对象

删 除 折 叠 菜 单 ， 直 接 阻 断 了

accordion。

accordion('refresh') jQuery 对象 更新折叠菜单，比如高度。

accordion('option', param) 一般值 获取 options 属性的值

accordion('option', param,

value)

jQuery 对象 设置 options 属性的值

**验证插件 validate（）**

这个插件是表单的简单验证插件，共两个文件包，

分别是jquery.validate.js和他的中文化插件jquery.validate.messages\_zh，

使用的时候找到jquery.validate.messages\_zh并打开，选取message里的内容，然后去覆盖jquery.validate.js的message里的内容

**具有的功能**

required: "必选字段"

remote: "请修正该字段", //远程提交 ，验证账号密码

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: $.validator.format("请输入一个长度最多是 {0} 的字符串"),

minlength: $.validator.format("请输入一个长度最少是 {0} 的字符串"),

rangelength: $.validator.format("请输入一个长度介于 {0} 和 {1} 之间的字符串")

range: $.validator.format("请输入一个介于 {0} 和 {1} 之间的值"),

max: $.validator.format("请输入一个最大为 {0} 的值"),

min: $.validator.format("请输入一个最小为 {0} 的值")

**用法:**

//第一步，

引入jquery文件

//第二步，

引入validate.js

//第三步，html文件

<form action="http://www.baidu.com" id="reg">

<P>账号：<input type="text" name="user" /></P>

<P>密码：<input type="text" name="password" id="pass" /></P>

<P>密码：<input type="text" name="text" /></P>

<P>邮箱：<input type="text" name="email" /></P>

<P>生日：<input type="text" name="day" /></P>

<button>tijiao</button>

</form>

//第四步，加载validate

js文件：在rules里以name值作为连接点，里面写入相应的键值对，messages里面对应着相应的提示，若不写，则提示系统默认的

$('#reg').validate({

rules:{ //规则库，想要的功能可以参照上面的“具有的功 能”板块里面的功能字段

user:{ //”user”为上面的name值

required:true,

minlength:2,

//将user的值ajax远程提交到user.php去验m证，只返回true或者flase

remote : 'user.php',

},

password:{

rangelength:[6,15],

required:true,

},

email:{

email:true,

required:true,

},

day:{

required:true,

date:true,

},

text:{

equalTo : '#pass', //与输入与id为pass相同的值

},

},

messages:{ //提示信息字段与上面的ruler相对应，在操作错误时就会显示出来

user:{

required:'必填项目',

minlength:'至少输入两位',

},

password:{

rangelength:'输入6-15位密码',

required:'必填项目'

},

},

//加入这两行则可以将错误逐条显示在class为ol.myerror里面

//在表单form下面加入<ol class="myerror"></ol>

errorLabelContainer : 'ol.myerror',

wrapper : 'li',

//输入全部正确，本地验证完成后执行的函数，在点击提交按钮后执行，并且会阻止原来表单的提交

submitHandler : function (form) {

alert(‘可以执行 ajax 提交，并且无须 debug:true 阻止提交了,

Form为整个表单的所有字段(object HTMLFormElement)，可利用form

获取数据’)

},

});

**Remote里面提交多条信息时（比如在登陆时同时提交账号和密码来提前静态验证）**

    remote: {

        url: "../user/validateUserName",

        type: "post",

        data: {

            username: function() {

                return $("#userInfoForm input[name='username']").val();

            }

        }

    }

**将提示信息全部显示在下面**

**Html**：

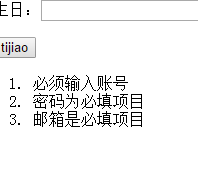
在form的最后加

<ol class="myerror"></ol>

**Js**

errorLabelContainer : 'ol.myerror',

wrapper : 'li',



**//使用方法加载 class 并添加文本，在成功元素的背后**

success : function (label) {

label.addClass('success').text('ok');

},



**//高亮显示有错误的元素，变色式，第一个参数代表错误的元素，**

highlight: function(element, errorClass) {

$(element).css('border', '1px solid red');

},

**//成功的元素移出错误高亮**

unhighlight : function (element, errorClass) {

$(element).css('border', '1px solid #ccc');

},

**//////////////remote提交，ajax静态验证输入的东西是否有效**

rules : {

user : {

required : true,

remote : 'an url',

//将user的值ajax远程提交到user.php去验m证，只返回true或者flase，默认get发送},

},

提交到服务器验证

以servlet为例：会自动捕获out对象输出的（out.print(true);）或者（out.print(false);），若为true表示验证成功，false表示 验证失败，

Messages{

User:{remote:’账户被占用’,}//当返回flase时这句话执行

}，

**一次提交多个数据**

rules : {

user : {

required : true,

remote : {

Url：‘an url’

Date:{

password: ‘’’’

}

}

//将user(默认会提交自己所以date里不用写)和password的值ajax远程提交到user.php去验m证，只返回true或者flase，默认get发送},

},

**单独验证法**

$('#user').rules('add', {

required : true,

minlength : 2,

messages : {

required : '帐号不得为空！',

minlength : jQuery.format('帐号不得小于{0}位！'),

}

});

**创建自己的验证**

/添加自定义验证

$.validator.addMethod('code', function (value, element) {

var tel = /^[0-9]{6}$/;

return this.optional(element) || (tel.test(value));//死的一句话

}, '请正确填写您的邮政编码');

//调用自定义验证

rules : {

code : {

required : true,

code : true,

}

},

**这是一个测试工具，验证成功后会执行函数**

submitHandler : function (form) {

//可以执行 ajax 提交，并且无须 debug:true 阻止提交了

},

1804r7429b.iok.la

**jquery.form.js插件，ajax提交插件**

**先引入从插件**

**只对表单有效**

$('#reg').submit(function () {//提交reg表

$(this).ajaxSubmit({

url : 'test.php',//设置提交的 url，可覆盖 action 属性

target : '#box',//服务器返回的内容存放在#box 里

type : 'GET',//GET,POST

dataType : null, //xml,json,script，默认为 null

//clearForm : true,//成功提交后，清空表单

//resetForm : true,//成功提交后，重置表单

data : {//增加额外的数据提交

aaa : 'bbb',

},

success : function (responseText, statusText) {

alert(responseText + statusText);

},//成功后回调

error : function (event, errorText, errorType) {

alert(errorText + errorType);

},//错误时调用

});

return false; //阻止默认提交

});

**jquery自带的ajax方法，好用**

**（这是一个下拉列表框提交的例子，可参照jquery笔记，可用于非表单提交）**

$('#se').change(function(){

var opt\_values = $("#se option:selected").val();

alert(opt\_values);

$.ajax({

type:"post", //提价类型

url:"http://asdajhsdjakshdka.com", //提交地址

data: {

select:opt\_values,

},

success:function(request,status,xhr){ //提交成功后执行的方法，response为返回的数据，status为状态

},

error : function (xhr, errorText, errorStatus) {

alert(xhr.status + ':' + xhr.statusText);

}

});

**提交程序书写方法**

**放在validate里面，在验证成功后会执行这一段，如果注册后不想跳转可以用这一段，然后ajax静态提交，如果默认表单直接提交，这一段可以不写,必须引入jquery.form.Js,ajaxsubmit()是里面的一个函数**

submitHandler : function (form) {

$('form').ajaxSubmit({

url:'add.php', //提交到哪里

type:'post', //以什么方式提交

beforeSubmit : function (formData, jqForm, options) {

$('#loading').dialog('open');

$('#reg').dialog('widget').find('button').eq(1).button('disable');

},//提交前出现一个小漩涡



success:function(responseText,statusText){

$('#reg').dialog('widget').find('button').eq(1).button('enable');

$('#loading').css('background', 'url(img/success.gif) no-repeat 20px center').html('数据新增成功...');

//提交成功后把圈圈改为勾

setTimeout(function () {

$('#loading').dialog('close'); //关闭等待框

$('#reg').dialog('close'); //关闭注册表单

$('#reg').resetForm(); //重置表单

$('#reg span.star').html('\*').removeClass('succ');

$('#loading').css('background', 'url(img/loading.gif) no-repeat 20px center').html('数据交互中...');//重置等待框

}, 1000);

},

});

},

**Cookie插件**

**使用前要先引入cookie插件**

$.cookie('user', 'bnbbs', {

expires : 7, //过期时间，7 天后

path : '/', //设置路径，上一层

domain : 'www.ycku.com', //设置域名

secure : true, //默认为 false，需要使用安全协议 https

});

//读取 cookie 数据

alert($.cookie('user'));

//读取 cookie 数据

alert($.cookie('user'));

**each循环遍历节点对象**

$.each($(‘.editor’),function(index,value){

arr[index] = $(value).height();

$(value).height(150)

$(value).next('.bottom').find('.up').hide();

});

//对于每个class为editor的元素，都执行函数,index为对应的索引，相当于数组下标

Value为对应的element元素节点，$(value)则为对应的jquery对象

**each循环遍历json对象**

$.each(json,function(index,value){

html += '<h4>' + value.users + ' 发表于 ' + value.date + '</h4><h3>' + value.tittle + '</h3><div class="editor">' + value.content + '</div><div class="bottom">0条评论 <span class="down"> 显示全部 &nbsp&nbsp</span><span class="up"> 收起 </span></div><hr noshade="noshade" size="1" />';

});

**后端发送json及jquery处理json的过程**

1、后端从数据库获取数据放入arraylist集合

2、应用json-lib.Jar和其他一些jar包（放在lib下）来讲arraylist转为json数组

3、通过out.Print()方法返回json

4、juery通过

$.ajax({

url: 'servlet/returnjson',

type:'POST',

success:function(response,status,xhr){ //response里面存着返回值

var json = $.parseJSON(response)；

});

5、通过each循环遍历json