http://www.w3school.com.cn/

这个网站提供大量的网站开发技术知识，包括W3C的标准技术：HTML、CSS、XML 。以及其他的技术，诸如JavaScript、PHP、SQL等等。还提供了还有非常实用的在线测试功能。

# HTML

页面跳转的几种方式：

# CSS

# JavaScript

## My97DatePicker时间控件详细使用说明

**onpicking事件**

实现日期选择联动：

<label class="date-title">就诊日期：</label>

<input id="d5221" class="Wdate" type="text" onFocus="var d5222=$dp.$('d5222');WdatePicker({onpicked:function(){d5222.focus();},minDate:'%y-%M-%d',maxDate:'#F{$dp.$D(\'d5222\')}'})"/>

至

<input id="d5222" class="Wdate" type="text" onFocus="WdatePicker({onpicked:doDateSearch, minDate:'#F{$dp.$D(\'d5221\')}'})"/>

选中日期后触发事件，调用doDateSearch方法：

<input id="d5222" class="Wdate" type="text" onFocus="WdatePicker({onpicked:doDateSearch, minDate:'#F{$dp.$D(\'d5221\')}'})"/>

# angularjs

# jquery

## jQuery 语法

jQuery 语法是为 HTML 元素的选取编制的，可以对元素执行某些操作。

基础语法是：$(selector).action()

美元符号定义 jQuery

选择符（selector）“查询”和“查找” HTML 元素

jQuery 的 action() 执行对元素的操作

示例

$(this).hide() - 隐藏当前元素

$("p").hide() - 隐藏所有段落

$(".test").hide() - 隐藏所有 class="test" 的所有元素（注意selector中不能有空格）

$("#test").hide() - 隐藏所有 id="test" 的元素

提示：jQuery 使用的语法是 XPath 与 CSS 选择器语法的组合。

**文档就绪函数：**

您也许已经注意到在我们的实例中的所有 jQuery 函数位于一个 document ready 函数中：

$(document).ready(function(){

--- jQuery functions go here ----

});

这是为了防止文档在完全加载（就绪）之前运行 jQuery 代码。

如果在文档没有完全加载之前就运行函数，操作可能失败。

## [jQuery取得select选中的值](http://www.cnblogs.com/greatverve/archive/2010/02/03/1662565.html)

获取选中的值：$("#select1").val();

获取选中的文本：$("#select1  option:selected").text();

## jQuery更改超链接的href

$(document).ready(function(){

$(".js-btn").click(function(){

var hospitalId = $(".js-hospital").val();

if(hospitalId == "" ){

alert("请选择医院");

return false;

}

this.href = this.href + "?hospitalId=" + hospitalId;

});

});

提交超链接是HTML本身就会自己完成的事情，JavaScript只是在提交之前更改一些属性的值，如href的值。另外，JavaScript也可以通过return false 来中断提交。

## jquery获取li列表中当前点击的元素下的各种值

**列表的HTML代码如下：**

<ul>

<li>

<a href="#1" class="case-info J\_CaseInfo js-meeting" >

<div class="expert">

<div class="date J\_Date">

<p>2015-01-26</p>

<p>星期一</p>

<input type="hidden" name="deptId" value="2001">

<input type="hidden" name="doctorId" value="5001">

</div>

<i class="icon pm">aa</i>

</div>

</a>

</li>

<li>

<a href="#2" class="case-info J\_CaseInfo js-meeting" deptid="aaa">

<div class="expert">

<div class="date J\_Date">

<p>2015-01-27</p>

<p>星期二</p>

<input type="hidden" name="deptId" value="2002">

<input type="hidden" name="doctorId" value="5002">

</div>

<i class="icon pm">bb</i>

</div>

</a>

</li>

</ul>

**jquery获取各个元素的代码示例如下：**

<script type="text/javascript">

$(document).ready(function(){

$(".js-meeting").click(function(){

alert($(this).attr('href')); //#1 (获取某个属性的值)

alert(this.href); //http://localhost:8080/opc/hospital/hospitalschedule?hospitalId=2000001#1

alert($(this).html()); //返回html文档： <div class="expert">…… </div>

alert($(this).find('div i').text()); //aa

alert($(this).find('div div p').text()); //2015-01-26星期一

alert($(this).find('div div p').eq(0).text()); //2015-01-26

alert($(this).find('div div p').eq(1).text()); //星期一

alert($(this).find("input[name='doctorId']").val()); //5001

});

});

</script>



## jquery与动态弹出层

## jquery与Json

以下例子演示jquery如何处理服务端返回的Json数据：

$(document).ready(function(){

$(".js-meeting").click(function(){

var hisPlanId = $(this).find("div div input").val();

var url = "${urlPath}" + "/opc/resource/getResourceByPlanId?hospitalId=${hospitalId}&hisPlanId=" + hisPlanId;

$.get(url, function(data, status){

//接收到的数据转化为JQuery对象，由JQuery为我们处理

var jsonData = eval(data);

if(jsonData.success == true){

$("tbody").html("<tr><th width=\"40%\">时间段</th><th width=\"20%\">就诊序号</th><th width=\"20%\">剩余号源</th><th width=\"20%\">操作</th></tr>");

//遍历对象数组，index是数组的索引号，objVal是遍历的一个对象。

$.each(jsonData.obj, function(index, objVal) {

//在table上追加一行

$("tbody").append("<tr><th height=\"40px\">"+objVal.startTime+"</th><th>"+objVal.orderNum+"</th><th>"+objVal.leaveNums+"</th><th><b>"+state+"</b></th></tr>");

});

}

});

});

});

public class Json {

private boolean success = false;

private String msg = "";

private Object obj = null; //实体列表，如 List<ResourceDto>

public boolean isSuccess() {

return success;

}

public void setSuccess(boolean success) {

this.success = success;

}

public String getMsg() {

return msg;

}

public void setMsg(String msg) {

this.msg = msg;

}

public Object getObj() {

return obj;

}

public void setObj(Object obj) {

this.obj = obj;

}

}

服务端的接口代码如下：

@RequestMapping("/getResourceByPlanId")

@ResponseBody

@Description("按条件查询排班")

public Json getResourceByPlanId(Long hospitalId,String hisPlanId){

Json j = new Json();

j.setSuccess(false);

try{

List<ResourceDto> resourceDtoList = infoService.getPlanResource(0, hospitalId, hisPlanId);

j.setSuccess(true);

j.setObj(resourceDtoList);

return j;

}

catch(Exception e){

e.printStackTrace();

}

return j;

}

## $.ajax()方法详解

虽然get()和post()函数非常简洁易用，但是对于更复杂的一些设计需求还是无法实现，比如在ajax发送的不同时段做出不同的动作等。jQuery提供一个更为具体的函数：ajax()。

jquery中的ajax方法参数总是记不住，这里记录一下。

**1.url**:   
要求为String类型的参数，（默认为当前页地址）发送请求的地址。

**2.type**:   
要求为String类型的参数，请求方式（post或get）默认为get。注意其他http请求方法，例如put和delete也可以使用，但仅部分浏览器支持。

**3.timeout**:   
要求为Number类型的参数，设置请求超时时间（毫秒）。此设置将覆盖$.ajaxSetup()方法的全局设置。

**4.async**:   
要求为Boolean类型的参数，默认设置为true，所有请求均为异步请求。如果需要发送同步请求，请将此选项设置为false。注意，同步请求将锁住浏览器，用户其他操作必须等待请求完成才可以执行。

**5.cache**:   
要求为Boolean类型的参数，默认为true（当dataType为script时，默认为false），设置为false将不会从浏览器缓存中加载请求信息。

**6.data**:   
要求为Object或String类型的参数，发送到服务器的数据。如果已经不是字符串，将自动转换为字符串格式。get请求中将附加在url后。防止这种自动转换，可以查看　　processData选项。对象必须为key/value格式，例如{foo1:"bar1",foo2:"bar2"}转换为&foo1=bar1&foo2=bar2。如果是数组，JQuery将自动为不同值对应同一个名称。例如{foo:["bar1","bar2"]}转换为&foo=bar1&foo=bar2。

**7.dataType**:   
要求为String类型的参数，预期服务器返回的数据类型。如果不指定，JQuery将自动根据http包mime信息返回responseXML或responseText，并作为回调函数参数传递。可用的类型如下：  
xml：返回XML文档，可用JQuery处理。  
html：返回纯文本HTML信息；包含的script标签会在插入DOM时执行。  
script：返回纯文本JavaScript代码。不会自动缓存结果。除非设置了cache参数。注意在远程请求时（不在同一个域下），所有post请求都将转为get请求。  
json：返回JSON数据。  
jsonp：JSONP格式。使用SONP形式调用函数时，例如myurl?callback=?，JQuery将自动替换后一个“?”为正确的函数名，以执行回调函数。  
text：返回纯文本字符串。

**8.beforeSend**：  
要求为Function类型的参数，发送请求前可以修改XMLHttpRequest对象的函数，例如添加自定义HTTP头。在beforeSend中如果返回false可以取消本次ajax请求。XMLHttpRequest对象是惟一的参数。  
            function(XMLHttpRequest){  
               this;   //调用本次ajax请求时传递的options参数  
            }  
**9.complete**：  
要求为Function类型的参数，请求完成后调用的回调函数（请求成功或失败时均调用）。参数：XMLHttpRequest对象和一个描述成功请求类型的字符串。  
          function(XMLHttpRequest, textStatus){  
             this;    //调用本次ajax请求时传递的options参数  
          }

**10.success**：要求为Function类型的参数，请求成功后调用的回调函数，有两个参数。  
         (1)由服务器返回，并根据dataType参数进行处理后的数据。  
         (2)描述状态的字符串。  
         function(data, textStatus){  
            //data可能是xmlDoc、jsonObj、html、text等等  
            this;  //调用本次ajax请求时传递的options参数  
         }

**11.error**:  
要求为Function类型的参数，请求失败时被调用的函数。该函数有3个参数，即XMLHttpRequest对象、错误信息、捕获的错误对象(可选)。ajax事件函数如下：  
       function(XMLHttpRequest, textStatus, errorThrown){  
          //通常情况下textStatus和errorThrown只有其中一个包含信息  
          this;   //调用本次ajax请求时传递的options参数  
       }

**12.contentType**：  
要求为String类型的参数，当发送信息至服务器时，内容编码类型默认为"application/x-www-form-urlencoded"。该默认值适合大多数应用场合。

**13.dataFilter**：  
要求为Function类型的参数，给Ajax返回的原始数据进行预处理的函数。提供data和type两个参数。data是Ajax返回的原始数据，type是调用jQuery.ajax时提供的dataType参数。函数返回的值将由jQuery进一步处理。  
            function(data, type){  
                //返回处理后的数据  
                return data;  
            }

**14.dataFilter**：  
要求为Function类型的参数，给Ajax返回的原始数据进行预处理的函数。提供data和type两个参数。data是Ajax返回的原始数据，type是调用jQuery.ajax时提供的dataType参数。函数返回的值将由jQuery进一步处理。  
            function(data, type){  
                //返回处理后的数据  
                return data;  
            }

**15.global**：  
要求为Boolean类型的参数，默认为true。表示是否触发全局ajax事件。设置为false将不会触发全局ajax事件，ajaxStart或ajaxStop可用于控制各种ajax事件。

**16.ifModified**：  
要求为Boolean类型的参数，默认为false。仅在服务器数据改变时获取新数据。服务器数据改变判断的依据是Last-Modified头信息。默认值是false，即忽略头信息。

**17.jsonp**：  
要求为String类型的参数，在一个jsonp请求中重写回调函数的名字。该值用来替代在"callback=?"这种GET或POST请求中URL参数里的"callback"部分，例如{jsonp:'onJsonPLoad'}会导致将"onJsonPLoad=?"传给服务器。

**18.username**：  
要求为String类型的参数，用于响应HTTP访问认证请求的用户名。

**19.password**：  
要求为String类型的参数，用于响应HTTP访问认证请求的密码。

**20.processData**：  
要求为Boolean类型的参数，默认为true。默认情况下，发送的数据将被转换为对象（从技术角度来讲并非字符串）以配合默认内容类型"application/x-www-form-urlencoded"。如果要发送DOM树信息或者其他不希望转换的信息，请设置为false。

**21.scriptCharset**：  
要求为String类型的参数，只有当请求时dataType为"jsonp"或者"script"，并且type是GET时才会用于强制修改字符集(charset)。通常在本地和远程的内容编码不同时使用。

案例代码：

[复制代码](javascript:void(0);)

$(function(){

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

$.ajax({

type: "GET",

url: "test.json",

data: {username:$("#username").val(), content:$("#content").val()},

dataType: "json",

success: function(data){

$('#resText').empty(); //清空resText里面的所有内容

var html = '';

$.each(data, function(commentIndex, comment){

html += '<div class="comment"><h6>' + comment['username']

+ ':</h6><p class="para"' + comment['content']

+ '</p></div>';

});

$('#resText').html(html);

}

});

});

});

[复制代码](javascript:void(0);)

**22.顺便说一下$.each()函数**:  
$.each()函数不同于JQuery对象的each()方法，它是一个全局函数，不操作JQuery对象，而是以一个数组或者对象作为第1个参数，以一个回调函数作为第2个参数。回调函数拥有两个参数：第1个为对象的成员或数组的索引，第2个为对应变量或内容。

另外可以参考：http://www.w3school.com.cn/jquery/ajax\_ajax.asp