学习方法

1. 建立逻辑思维
   1. 多分析
   2. 理解逻辑算法
   3. 多练、多敲、多读
2. 编程习惯
3. 大胆动手
4. 保持好奇好
5. 乐观面对错误

就业能力

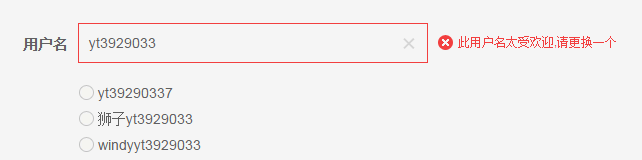
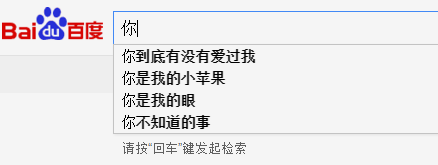
1. 分析能力
2. 独立思考能力
3. 程序阅读能力
4. 程序排错能力
5. 独立编写程序能力
6. 团队协作能力
7. 学习能力
8. 善于总结的能力

专业技能

Ajax上

1. Ajax的概念
   1. Ajax是一种在无需重新加载整个网页(刷新页面)的情况下，能够更新部分网页的技术。
   2. Ajax的全称是AsynchronousJavaScript and XML,即异步JavaScript+XML。它并不是新的编程语言，而是几种原有技术的结合体。
2. Ajax的优势
   1. 无刷新更新数据。
      1. Ajax最大优点就是能在不刷新整个页面的前提下与服务器通信维护数据。这使得Web应用程序更为迅捷地响应用户交互，并避免了在网络上发送那些没有改变的信息，减少用户等待时间，带来非常好的用户体验。
   2. 异步与服务器通信
      1. AJAX使用异步方式与服务器通信，不需要打断用户的操作，具有更加迅速的响应能力。优化了Browser和Server之间的沟通，减少不必要的数据传输、时间及降低网络上数据流量。
   3. 前端和后端负载平衡
      1. AJAX可以把以前一些服务器负担的工作转嫁到客户端，利用客户端闲置的能力来处理，减轻服务器和带宽的负担，节约空间和宽带租用成本。并且减轻服务器的负担，AJAX的原则是“按需取数据”，可以最大程度的减少冗余请求和响应对服务器造成的负担，提升站点性能。
3. 扩展：Ajax的工作原理
   1. Ajax的工作原理相当于在用户和服务器之间加了—个中间层(AJAX引擎),使用户操作与服务器响应异步化。并不是所有的用户请求都提交给服务器,像—些数据验证和数据处理等都交给Ajax引擎自己来做, 只有确定需要从服务器读取新数据时再由Ajax引擎代为向服务器提交请求。  
       Ajax其核心有JavaScript、XMLHTTPRequest、DOM对象组成，通过XmlHttpRequest对象来向服务器发异步请求，从服务器获得数据，然后用JavaScript来操作DOM而更新页面。这其中最关键的一步就是从服务器获得请求数据。
4. 扩展：同步和异步
   1. 异步：同时执行，也叫并发
   2. 同步：按步骤顺序执行，一条一条执行
5. 编写步骤
   1. 打电话： ajax请求：
      1. 买个手机 创建Ajax对象
      2. 拨号 连接服务器
      3. 说话 发送请求
      4. 听 接收返回
   2. function ajax(url,funWin,funFaild){  
       //1.创建ajax对象  
       var ajax = window.XMLHttpRequest ? new XMLHttpRequest() : new ActiveXObject(“Microsoft.XMLHTTP”);  
       //2.连接服务器  
       ajax.open(“GET”,url,true);  
       //3.发送请求  
       ajax.send();  
       //4.接收返回  
       ajax.onreadystatechange = function(){  
       if(ajax.readyState == 4){  
       if(ajax.status == 200){  
       if(funWin){  
       funWin(ajax.responseText);  
       }  
       }else{  
       if(funFaild){  
       funFaild();  
       }  
       }  
       }  
       }  
      }
   3. open方法：注意
      1. URL是相对于当前页面的路径，也可以使用绝对路径。
      2. Open方法不会向服务器发送真正请求，它相当于初始化请求并准备发送。
      3. 只能向同一个域中使用相同协议和端口的URL发送请求，否则会因为安全原因报错。
   4. 请求状态监控
      1. Onreadystatechange(状态改变事件)
         1. readyState属性：请求状态
         2. 0：请求未初始化（还没有调用open()）
         3. 1: 请求已经建立，但是还没有发送（还没有调用send（））
         4. 2：请求已发送，正在处理中（通常现在可以从响应中获取内容头）。
         5. 3：请求在处理中，通常响应中已有部分数据可用了，但是服务器还没有完成响应的生成。
         6. 4：响应已完成，您可以获取并使用服务器的响应了。
      2. Status属性：请求结果（200代表成功）
      3. ResponseText属性：服务器发回的信息（文件中的内容）。
6. 回调函数
7. 编写PHP服务器接口
   1. <?php  
       $username = $\_GET["user"];  
       print "欢迎你".$username.",你的登录时间是:".date("y/m/d");  
      ?>
   2. 然后我们在浏览器的地址栏输入这个 PHP的地址：http://localhost/ demo.php?user=zzl

第二节：应用

1. 验证用户名是否存在
2. 搜索下拉显示

第三节

1. JS解析JSON
   1. Eval()方法：解析JSON数据的最常用方法是使用javascript的eval()方法，代码如下：  
      function toJson(str){  
       var json = eval(‘(‘ + str + ‘)’);  
       return json;  
      }  
      该方法存在性能和安全方面的问题，不建议使用。
   2. JSON.parse()方法  
      这种方法只支持IE8/Firefox3.5+/Chrome4/Safari4/Opera10以上版本，这些浏览器都已经接近W3C标准，默认实现了toJSON方法。代码如下：  
      function toJson(str){  
       return JSON.parse(str);  
      }
   3. New Function方法：代码如下：  
      function toJson(str){  
       var json = (new Function(“return” + str))();  
       return json;  
      }
2. 局部数据刷新
   1. 请求并显示静态TXT文件
      1. 字符集编码（三个按钮分别读取三个不同的文件并存入div中）  
         var oBtn = document.getElementsByTagName("input");  
          var oDiv = document.getElementById("div1");  
          for(var i = 0; i < oBtn.length; i++){  
          oBtn[i].index = i;  
          oBtn[i].onclick = function(){  
          ajax(this.index + 1 + ".txt?t=" + new Date().getTime() ,function(str){  
          oDiv.innerHTML = str;  
          });  
          }  
          }  
           
         <input type="button" value="按钮一" />

<input type="button" value="按钮二" />  
 <input type="button" value="按钮三" />  
 <div id="div1"></div>

* + 1. 缓存、阻止缓存（?t = new Date().getTime()）  
       var oBtn = document.getElementsByTagName("input")[0]; oBtn.onclick = function(){  
        document.title = '1.txt?t=' + new Date().getTime();  
        ajax('1.txt?t=' + new Date().getTime(),function(str){  
        alert(str);  
        });  
        };  
         
       访问服务器，从服务器获取一个txt文件<br />  
       <input type="button" value="获取txt文件" />
  1. 动态数据：请求JS(或json)文件
     1. Eval的使用  
        var oBtn = document.getElementById("btn");  
         oBtn.onclick = function(){  
         ajax('data.txt',function(str){  
         //alert(str[0]);  
         //"[1,2,3,4,5]"  
         //alert(typeof str);  
         var arr = eval(str);  
         alert(arr[0]);  
         });  
         };  
          
        连接服务器，访问data中的一个数组<br />  
        <input type="button" value="访问数组" id = "btn" />
     2. DOM创建元素（加载更多按钮，分布）  
        var oBtn = document.getElementById("btn");  
         oBtn.onclick = function(){  
         ajax("data.json",function(str){  
         var arr = eval(str);  
         alert(arr[1].c);  
         });  
         };  
          
        通过ajax,访问一个json数据<br />  
        <input type="button" value="访问json数据" id="btn" />  
          
        json数据  
        [{a:1,b:2},{c:3,d:4}]  
          
        分页：  
        var oUl = document.getElementById("ul1");  
        var oBtn = document.getElementsByTagName("a");  
        for(var i = 0; i < oBtn.length; i++){  
         oBtn[i].index = i;  
         oBtn[i].onclick = function(){  
         ajax("page" + (this.index + 1) + '.txt',function(str){  
         var arr = eval(str);  
         oUl.innerHTML = "";  
         for(var j = 0; j < arr.length; j++){  
         var oLi = document.createElement("li");  
         oLi.innerHTML = "<strong>" + arr[j].user + "</strong><i>" + arr[j].pass + "</i>";  
         oUl.appendChild(oLi);  
         }  
         });  
         };  
        }  
          
        <ul id="ul1"></ul>  
        <a href="javascript:;">1</a>  
        <a href="javascript:;">2</a>  
        <a href="javascript:;">3</a>  
          
        json数据：  
        [{user:"aa",pass:"111"},{user:"bb",pass:"222"},{user:"cc",pass:"333"}]  
        [{user:"qq",pass:"444"},{user:"ww",pass:"555"},{user:"ee",pass:"666"}]  
        [{user:"xx",pass:"777"},{user:"yy",pass:"888"},{user:"zz",pass:"999"}]

1. 事件委托的应用
2. 前后端分离

第四节：应用

1. 加载更多
2. Ajax瀑布流

第五节：综合应用

1. 即时聊天