



5.6 HTML5的WebStorage

- ❖ 解决本不应该却不得不用Cookie进行本地存储的应用场景。
 - Cookie大小限制在4KB
 - HTTP请求中Cookie是明文传递的（HTTPS不是）
- ❖ 在每个HTTP请求的HttpRequest和HttpResponse的header中都要传输Cookie。HTML5提供了两个存储数据的内置对象，都是window的成员。
 - sessionStorage: 存储在会话Session中，保存至浏览器关闭。
 - localStorage: 不限时间地存储在客户端本地。
- ❖ 只能存储字符串，可结合JSON对象（stringify和parse方法）一起。



❖ 操作sessionStorage对象变量

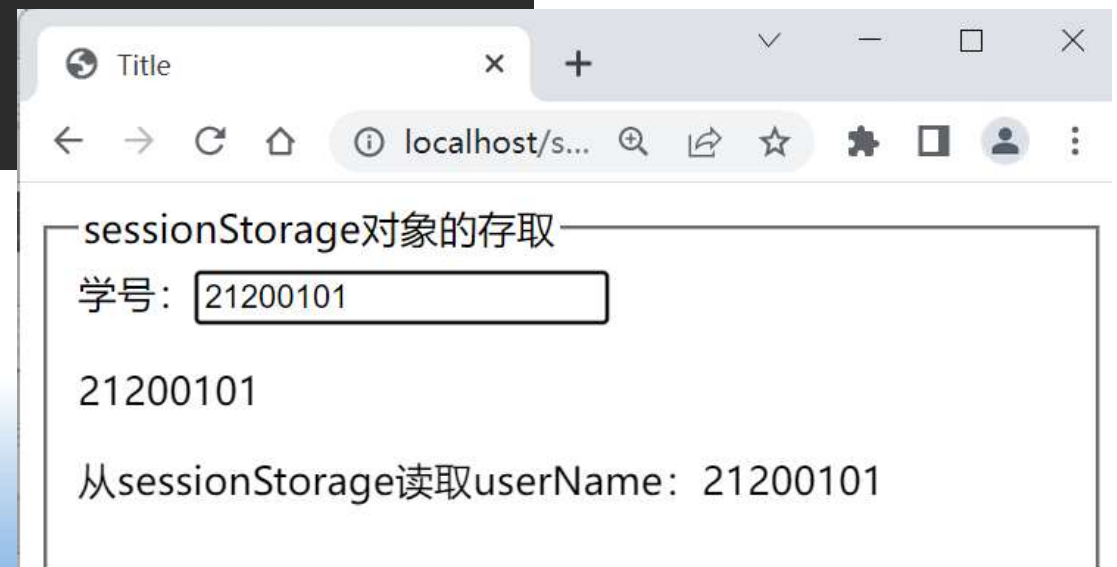
- `sessionStorage.keyName = value;`
- `sessionStorage.setItem(keyName, value);` //没有变量keyName , 生成新的
- `sessionStorage.getItem(keyName);` //没有变量keyName , 返回null
- `keyName = sessionStorage.key(n);`
- `keyNum = sessionStorage.length;` //返回sessionStorage中数据项数
- Session结束时, 自动销毁变量。



```
<body>
  <fieldset>
    <legend>sessionStorage对象的存取</legend>
    学号: <input type="text" name="userName" onchange="userNameChange(this);">
    <p id="status1"></p>
    <p id="status2"></p>
  </fieldset>
  <script type="text/javascript">
    function userNameChange(obj){
      var userName = obj.value;
      sessionStorage.setItem("userName", userName);
      document.getElementById("status1").innerHTML = userName;
      document.getElementById("status2").innerHTML = "从sessionStorage读取userName: "
        + sessionStorage.getItem("userName");
    }
  </script>
</body>
```



```
<body>
  <fieldset>
    <legend>sessionStorage对象的存取</legend>
    学号: <input type="text" name="userName" onchange="userNameChange(this);">
    <p id="status1"></p>
    <p id="status2"></p>
  </fieldset>
  <script type="text/javascript">
    function userNameChange(obj){
      var userName = obj.value;
      sessionStorage.setItem("userName", userName);
      document.getElementById("status1").innerHTML = userName;
      document.getElementById("status2").innerHTML = "从sessionStorage读取userName: "
        + sessionStorage.getItem("userName");
    }
  </script>
</body>
```





❖ 操作localStorage对象变量

- `localStorage.keyName = value;`
- `localStorage.setItem(keyName, value);` //没有变量keyName , 生成新的
- `localStorage.getItem(keyName);` //没有变量keyName , 返回null
- `keyName = localStorage.key(n);`
- `keyNum = localStorage.length;` //返回localStorage中数据项数
- `localStorage.clear();` //清除全部数据
- `localStorage.removeItem(keyName);`

❖ 建议使用localStorage, 可将数据长期保存在客户端, 直至人工清除。



```
<form id="frmLogin" action="#">
  <fieldset>
    <legend>登录</legend>
    <ul>
      <li><span id="spnStatus"></span></li>
      <li>账号: <input type="text" id="userName" onchange="userName_change(this);"></li>
      <li>密码: <input type="password" id="userPwd"></li>
      <li><input type="checkbox" id="chkSave">是否保存密码</li>
      <li>
        <input type="button" name="btnLogin" value="登录" onclick="btnLogin_click();">
      </li>
    </ul>
  </fieldset>
</form>
```



```
<script type="text/javascript">
    function $(id){
        return document.getElementById(id);
    }
    function userName_change(obj){
        var userPwd = localStorage.getItem(obj.value);
        if (userPwd){
            $$("userPwd").value = userPwd;
            $$("spnStatus").innerHTML = "密码是: " + userPwd;
        }else {
            $$("spnStatus").innerHTML = "";
        }
    }
    function btnLogin_click(){
        if ($$("chkSave").checked){
            localStorage.setItem($$("userName").value, $$("userPwd").value);
        }else {
            localStorage.removeItem($$("userName").value);
        }
    }
</script>
<div>
```




localhost/localStorage.html

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

密钥	值
选择一个值以	

localhost/localStorage.html

登录

- 账号:
- 密码:
- ☒ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

密钥	值
21200101	0101pwd
选择一个值以	



localhost/localStorage.html

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
- http://localhost
- 会话存储空间
- http://localhost
- IndexedDB

密钥	值
21200101	0101pwd

选择一个值以

登录

- 密码是: 0101pwd
- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
- http://localhost
- 会话存储空间
- http://localhost
- IndexedDB

密钥	值
21200101	0101pwd

选择一个值以

登录

- 密码是: 0101pwd
- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

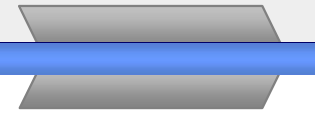
- 清单
- Service Workers
- 存储

存储

- 本地存储空间
- http://localhost
- 会话存储空间
- http://localhost
- IndexedDB

密钥	值
----	---

选择一个值以



```
function btnLogin_click(){
    if ($("#chkSave").checked){
        localStorage.setItem($("#userName").value, $("#userPwd").value);
        sessionStorage.setItem($("#userName").value, $("#userPwd").value);
    }else {
        localStorage.removeItem($("#userName").value);
        sessionStorage.removeItem($("#userName").value);
    }
}
</script>
```



localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☒ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

密钥	值
userName	userName

1 userName

localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☒ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

密钥	值
userName	userName

1 userName



localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

过滤

密钥	值
userName	userName

1 userName

localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

过滤

密钥	值
userName	userName

1 userName



localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

过滤

密钥	值
userName	userName

1 userName

localhost/localStorage.html

更新

登录

- 账号:
- 密码:
- ☐ 是否保存密码
-

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
 - http://localhost
- 会话存储空间
 - http://localhost

过滤

密钥	值
----	---

选择一个值
以预览



```
<p id="pInfo">请输入信息...</p>
<table id="tblMessage" border="1px solid"></table>
<textarea id="txtContent" cols="40" rows="2"></textarea><br>
<input id="btnAdd" type="button" value="发表">
<input id="btnClear" type="button" value="清空">
<script type="text/javascript">
```

```
function $(id){
    return document.getElementById(id);
}
window.onload = function (){
    $$("#btnAdd").addEventListener("click", btnAdd_click);
    $$("#btnClear").addEventListener("click", btnClear_click);
}
```

```
function RetRndNum(n){
    var strRnd = "";
    for (var i = 0; i < n; i++){
        strRnd += Math.floor(Math.random() * 10);
    }
    return strRnd;
}
```

```
function btnClear_click(){
    localStorage.clear();
    $$("#tblMessage").innerHTML = "";
    $$("#pInfo").innerHTML = "localStorage已清空! ";
}
```



字符串格式

```
function btnAdd_click(){
    var strContent = $("#txtContent").value;
    var strTime = new Date();
    if (strContent.length > 0){
        var strKey = "cnt" + RetRndNum(4);
        var strVal = strContent + "," + strTime.toLocaleTimeString();
        localStorage.setItem(strKey, strVal);
    }
    loadLocalData();
    $("#txtContent").value = "";
}
```

```
function loadLocalData(){
    var strHTML = "<tr><td>编号</td><td>内容</td><td>时间</td></tr>";
    var strArr = new Array();
    for(var i = 0; i < localStorage.length; i++){
        var strKey = localStorage.key(i);
        if(strKey.substring(0, 3) == "cnt"){
            strArr = localStorage.getItem(strKey).split(",");
            strHTML += "<tr><td>" + strKey + "</td>";
            strHTML += "<td>" + strArr[0] + "</td>";
            strHTML += "<td>" + strArr[1] + "</td>";
            strHTML += "</tr>";
        }
    }
    $("#tblMessage").innerHTML = strHTML;
    $("#pInfo").innerHTML = "localStorage存储的信息为";
}
```



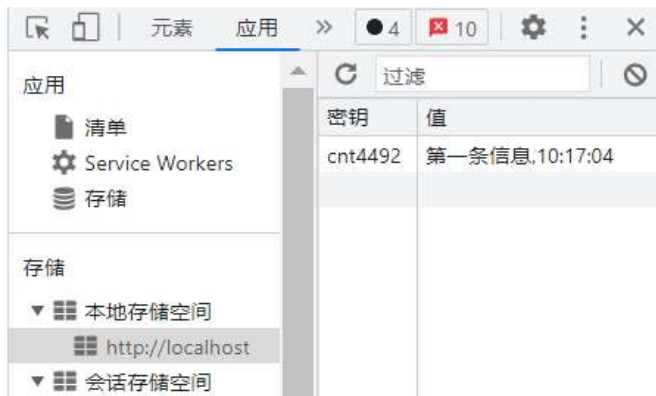
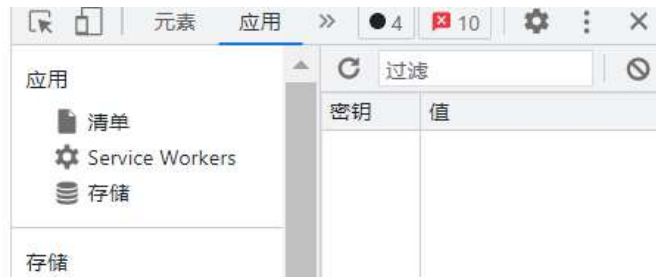
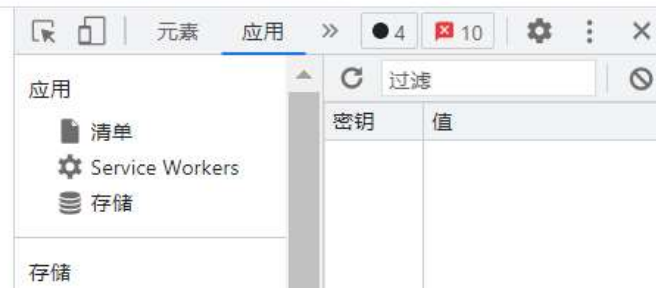

清空

第一条信息

清空

编号	内容	时间
cnt4492	第一条信息	10:17:04

清空





localStorage存储的信息为

编号	内容	时间
cnt6414	第二条信息	10:17:30
cnt4492	第一条信息	10:17:04

发表

清空

应用

应用

清单

Service Workers

存储

存储

本地存储空间

http://localhost

会话存储空间

http://localhost

IndexedDB

过滤

密钥

值

cnt4492

第一条信息,10:17:04

cnt6414

第二条信息,10:17:30

localStorage已清空!

发表

清空

应用

应用

清单

Service Workers

存储

存储

本地存储空间

http://localhost

会话存储空间

http://localhost

IndexedDB

过滤

密钥

值



JSON格式

```
function btnAdd_click(){  
    var strContent = $("#txtContent").value;  
    var strTime = new Date();  
    if (strContent.length > 0){  
        var strKey = "cnt" + RetRndNum(4);  
        var setData = new Object();  
        setData.key = strKey;  
        setData.content = strContent;  
        setData.time = strTime.toLocaleTimeString();  
        var setDataStr = JSON.stringify(setData);  
        localStorage.setItem(strKey, setDataStr);  
    }  
    loadLocalData();  
    $("#txtContent").value = "";  
}
```

```
function loadLocalData(){  
    var strHTML = "<tr><td>编号</td><td>内容</td><td>时间</td></tr>";  
    for(var i = 0; i < localStorage.length; i++){  
        var strKey = localStorage.key(i);  
        if(strKey.substring(0, 3) == "cnt"){  
            // strArr = localStorage.getItem(strKey).split(",");  
            var getData = JSON.parse(localStorage.getItem(strKey));  
            strHTML += "<tr><td>" + strKey + "</td>";  
            strHTML += "<td>" + getData.content + "</td>";  
            strHTML += "<td>" + getData.time + "</td>";  
            strHTML += "</tr>";  
        }  
    }  
    $("#tblMessage").innerHTML = strHTML;  
    $("#pInfo").innerHTML = "localStorage存储的信息为";  
}
```



localhost/traverseLocalStorage.html

请输入信息...

发表 清空

应用

- 清单
- Service Workers
- 存储

存储

过滤

密钥	值
----	---

localStorage存储的信息为

编号	内容	时间
cnt0519	第一条信息	10:50:57

发表 清空

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
- http://localhost
- 会话存储空间

过滤

密钥	值
cnt0519	{"key":"cnt0519","content":"第一条信息","time":"10:50:57"}

localStorage存储的信息为

编号	内容	时间
cnt0519	第一条信息	10:50:57
cnt6814	第二条信息	10:51:39

发表 清空

应用

- 清单
- Service Workers
- 存储

存储

- 本地存储空间
- http://localhost
- 会话存储空间
- http://localhost
- IndexedDB

过滤

密钥	值
cnt0519	{"key":"cnt0519","content":"第一条信息","time":"10:50:57"}
cnt6814	{"key":"cnt6814","content":"第二条信息","time":"10:51:39"}

6、AJAX

第4章 JavaScript



6.1 为什么有AJAX

❖ 传统Web交互

- 客户端向服务器发送HTTP请求；服务器收到请求后做出响应，返回一个新页面。
- 在服务器处理客户端请求的时间里，客户端只能空闲等待。
- 即便只需从服务器端得到很简单的一个数据，都要返回一个完整的页面，浪费用户的时间和带宽。
- 应用的响应时间依赖于服务器的响应时间，导致用户界面的响应比本地应用慢得多。

❖ AJAX

- 是一种快速创建动态网页的技术，用于解决传统方法的缺陷。通过在后台与服务器进行少量数据交换，实现网页的异步更新，即可以在不重新加载整个网页的情况下，对网页的某部分进行更新。



6.2 XMLHttpRequest对象

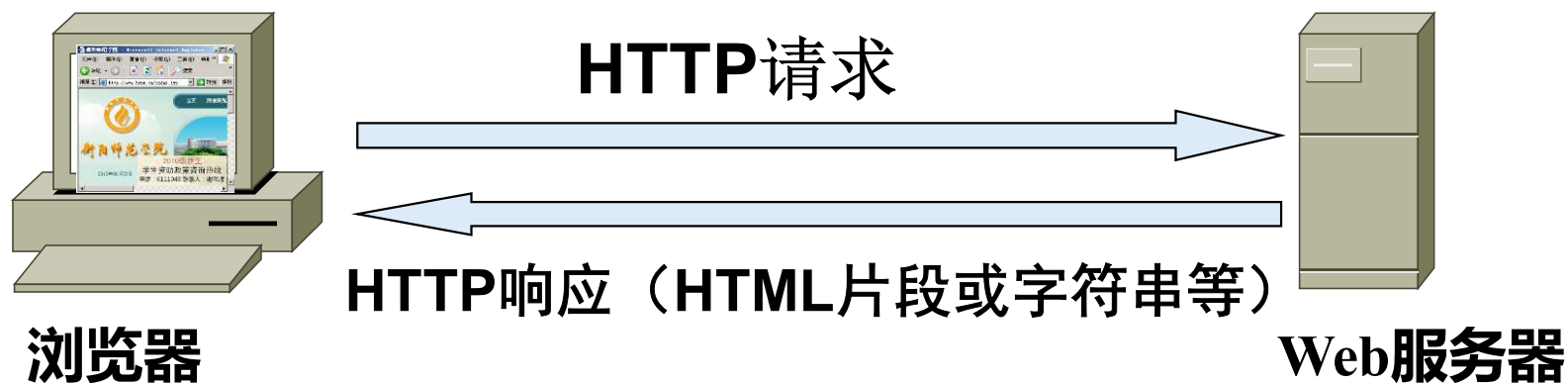
- ❖ XMLHttpRequest对象能在网页加载后与服务器进行通信、交换数据，实现对网页的部分更新。（**AJAX的核心概念**）
 - 在不重新加载整个页面的情况下更新页面的局部
 - 在页面已加载后向服务器请求数据
 - 在页面已加载后从服务器接收数据
 - 向服务器发送数据
- ❖ 现在，几乎所有的浏览器都支持 XMLHttpRequest 对象。

```
> typeof(XMLHttpRequest);  
< 'function'
```

```
> var xhr = new XMLHttpRequest();  
  xhr instanceof XMLHttpRequestEventTarget;  
< true  
  
> var xhr = new XMLHttpRequest();  
  xhr instanceof EventTarget;  
< true
```



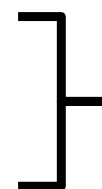

浏览器发送HTTP请求的三种方式



输入网址（请求载入页面）

提交表单（发送数据并载入页面）

用XMLHttpRequest对象发送异步HTTP请求



页面会刷新

页面不刷新



XMLHttpRequest对象的方法

方法	描述
setRequestHeader(header, value)	添加 HTTP请求头 <ul style="list-style-type: none">header: 报头标签value: 报头值
open(method, url, async, user, psw)	设置请求的类型、URL、异/同步方式 <ul style="list-style-type: none">method: 请求类型 (GET/POST)url: 服务器上文件的URLasync: true (异步) 或 false (同步)user: 用户名 (可选)psw: 密码 (可选)
send(string)	将POST请求发送到服务器 <ul style="list-style-type: none">string: 仅用于 POST 请求
send()	将GET请求发送到服务器



XMLHttpRequest对象的属性

属性	描述	
readyState	XMLHttpRequest 的状态 <ul style="list-style-type: none">0: 请求未初始化1: 服务器连接已建立2: 请求已接收3: 请求处理中4: 请求已完成且响应已就绪	
onreadystatechange	当 readyState 属性发生变化时，就会调用该函数。	
responseText	字符串形式的响应数据	
responseXML	XML DOM形式的响应数据	
status和statusText	<ul style="list-style-type: none">200: "OK"304: "Not Modified"	<ul style="list-style-type: none">403: "Forbidden"404: "Not Found"



XMLHttpRequest与服务器通信的步骤

1、创建XMLHttpRequest对象。

```
var xmlhttp = new XMLHttpRequest();
```

2、使用open()方法设置请求的URL、HTTP请求的发送方式、是否为异步模式。

3、使用send()方法发送HTTP请求。

- GET方式

```
xmlhttp.open("GET", "getHint.php?q=A&sid=0.850", true);
```

```
xmlhttp.send(null);
```



- POST方式

先使用setRequestHeader() 添加HTTP头，然后在send()方法中发送数据。

```
xmlHttp.open("POST", "getHint.php", true);
```

```
xmlHttp.setRequestHeader("Content-Type",  
    "application/x-www-form-urlencoded");
```

```
xmlHttp.send("q=A&sid=0.8507425065126537");
```



4、使用onreadystatechange事件监听服务器端的反馈，根据readyState属性来判断服务器是否已经对请求处理完成，一旦完成就接收服务器端传回的数据。

```
xmlHttp.onreadystatechange = function () {  
    if (xmlHttp.readyState == 4 && xmlHttp.status == 200) {  
        document.getElementById("txtHint").innerHTML = xmlHttp.responseText;  
    }  
}
```



XMLHttpRequest代码实例（GET）

```
<> 4-xmlHttpRequest.html > script
1  <!DOCTYPE html>
2  <head>
3  |   <meta charset="UTF-8">
4  </head>
5  <body>
6  |   <form onsubmit="return false;">
7  |   |  关键字(A、B、C、D) : <input type="text" id="txt">
8  |   </form>
9  |   <p id="p">可查询到的姓名为: <span id="txtHint"></span></p>
10 </body>
11 </html>
```

关键字 (A、B、C、D) :

可查询到的姓名为:

```
array("Anna", "Brittany", "Cinderella", "Dianna", "Amanda", "Cindy", "Doris");
```





```
> 4-xmlHttpRequest.html > ...
13 <script type="text/javascript">
14   var xmlhttp = new XMLHttpRequest();
15   document.getElementById("txt").addEventListener("keyup", showHint);
16   var cnt = 0;
17   xmlhttp.onreadystatechange = function () { //定义服务器监听函数
18       console.log(++cnt + ": " + "readyState--" + xmlhttp.readyState + " status--" + xmlhttp.status);
19       if (xmlhttp.readyState == 4 && xmlhttp.status == 200) { //正确接受服务器响应信息
20           console.log("responseText--" + xmlhttp.responseText);
21           document.getElementById("txtHint").innerHTML = xmlhttp.responseText;
22       }
23   }
24   function showHint(){
25       var str = document.getElementById("txt").value;
26       if (str.length == 0){
27           document.getElementById("txtHint").innerHTML = "";
28           return;
29       }
30       var url = "4-getHint.php";
31       var param = "q=" + str + "&sid=" + Math.random(); //阻止从缓存中读取数据
32       console.log(param);
33       xmlhttp.open("POST", url, true); //准备发送数据
34       xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
35       xmlhttp.send(param); //数据发送到服务器，开始监听
36       console.log("关键字母: " + str + " URL:" + url);
37   }
38 </script>
```



4-getHint.php

```
1  <?php
2      $hintArr = array("Anna", "Brittany", "Cinderella",
3          "Dianna", "Amanda", "Cindy", "Doris");
4      $q = $_POST["q"];
5      $hint = "";
6      foreach ($hintArr as $name){
7          if (stripos($name, $q) === 0){
8              $hint .= $name." ";
9          }
10     }
11     echo "<br>$hint<br>";
12     ?>
```



localhost/xmlHttpRequest.html

← → ↻ 🏠 ⓘ

🔗 ☆ ⚙️ 🖱️ 👤 ⋮

关键字母 (A、B、C、D) :

可查询到的姓名为:

元素 控制台 源代码 网络 性能 内存 应用 >>

🔍 top 过滤 默认级别 11 个问题: 11 414 条已隐藏

>

关键字母 (A、B、C、D) :

可查询到的姓名为:
Anna Amanda

元素 控制台 源代码 网络 性能 内存 应用

🔍 top 过滤 默认级别 1

1: readyState--1 status--0
关键字母: A URL: getHint.php?q=A&sid=0.4530054155056733
2: readyState--2 status--200
3: readyState--3 status--200
4: readyState--4 status--200
responseText--
Anna Amanda

关键字母 (A、B、C、D) :

可查询到的姓名为:
Anna

元素 控制台 源代码 网络 性能 内存 应用

🔍 top 过滤 默认级别 11

3: readyState--3 status--200
4: readyState--4 status--200
responseText--
Anna Amanda

5: readyState--1 status--0
关键字母: An URL: getHint.php?q=An&sid=0.9220924305047482
6: readyState--2 status--200
7: readyState--3 status--200
8: readyState--4 status--200
responseText--
Anna



XMLHttpRequest代码实例（POST）

```
var url = "getHint.php";  
var param = "q=" + str + "&sid=" + Math.random(); //阻止从缓存中读取数据  
console.log(param);  
xmlHttp.open("POST", url, true); //准备发送数据  
xmlHttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");  
xmlHttp.send(param); //数据发送到服务器，开始监听  
console.log("关键字: " + str + " URL:" + url);
```

```
<?php  
$hintArr = array("Anna", "Brittany", "Cinderella", "Dianna", "Amanda", "Cindy", "Doris");  
$q = $_POST["q"];  
$hint = "";  
foreach ($hintArr as $name){  
    if (stripos($name, $q) === 0){  
        $hint .= $name." ";  
    }  
}  
echo "<br>$hint<br>";
```



XMLHttpRequest读取JSON文件

```
<script type="text/javascript">
  var ajax = new XMLHttpRequest();
  ajax.open("get", "4-student.json");
  ajax.send();
  console.log("readyState--" + ajax.readyState +
    " status--"+ ajax.status);
  ajax.onreadystatechange = function () {
    console.log("readyState--" + ajax.readyState +
      " status--"+ ajax.status);
    if (ajax.readyState == 4 && ajax.status == 200) {
      console.log("OK");
      console.log(ajax.responseText);
    }
  };
</script>
```

```
[
  {"name": "张一", "id": "21210101", "age": 18, "gender": "男"},
  {"name": "王二", "id": "21210102", "age": 19, "gender": "男"},
  {"name": "李三", "id": "21210103", "age": 20, "gender": "男"},
  {"name": "赵四", "id": "21210104", "age": 21, "gender": "女"},
  {"name": "刘五", "id": "21210105", "age": 22, "gender": "女"}
]
```




元素 控制台 源代码 网络 性能 内存 应用 安

top ▼

过滤

readyState--1	status--0
readyState--2	status--200
readyState--3	status--200
readyState--4	status--200
OK	
[{"name": "张一", "id": "21210101", "age": 18, "gender": "男"}, {"name": "王二", "id": "21210102", "age": 19, "gender": "男"}, {"name": "李三", "id": "21210103", "age": 20, "gender": "男"}, {"name": "赵四", "id": "21210104", "age": 21, "gender": "女"}, {"name": "刘五", "id": "21210105", "age": 22, "gender": "女"}]	

7、事件处理

第4章 JavaScript



7.1 事件处理

- ❖ 事件是用户和Web页面交互时产生的各种操作。
- ❖ 浏览器运行的大部分时间都是在等待事件的发生，并在事件发生时调用相应的事件处理函数，完成事件处理。
- ❖ 发生事件的地方称为事件源，发生事件时调用的处理函数称为事件处理器。
- ❖ 事件是JavaScript和DOM之间进行交互的桥梁，当某个事件发生时，通过它的处理函数执行相应的JavaScript代码。



7.2 事件处理器的两种注册方法

❖ 将事件处理器连接到事件源的过程称为注册

- 在HTML标签中使用标签属性绑定事件处理器

```
<input type = "button" id = "myBtn" onclick = "func( );" />
```

- 在JavaScript脚本中使用对象的属性动态地绑定事件处理器

```
document.getElementById("myBtn").addEventListener("click", func);
```



7.3 浏览器中的常用事件

❖ 窗口事件（仅在 body 元素中有效）

- onload: 当文档载入时
- onunload: 当文档卸载时

❖ 表单元素事件（仅在表单元素中有效）

- onchange: 当元素改变时
- onsubmit: 当表单被提交时
- onreset: 当表单被重置时
- onselect: 当元素被选取时
- onblur: 当元素失去焦点时
- onfocus: 当元素获得焦点时



❖ 键盘事件

- onkeydown: 当键盘被按下时
- onkeypress: 当键盘被按下后又松开时
- onkeyup: 当键盘被松开时

❖ 鼠标事件

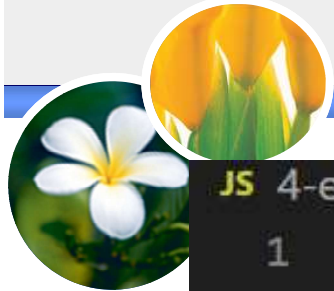
- onclick: 当鼠标被单击时
- ondblclick: 当鼠标被双击时
- onmousedown: 当鼠标按钮被按下时
- onmouseup: 当鼠标按钮被松开时
- onmousemove: 当鼠标指针移动时
- onmouseover: 当鼠标指针悬停在元素上时
- onmouseout: 当鼠标指针移出元素时



7.4 事件的应用举例

4-eventListener.html > ...

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>表单事件处理</title>
  <link rel="stylesheet" type="text/css" href="4-eventListener_style.css">
</head>
<body>
  <form id="register" name="register" action="4-register_submit.php" method="POST">
    <table><tr><td>用户名: </td>
      <td><input type="text" id="userName" name="userName" value="用户名"></td>
      <td><span id="userNameSpan"></span></td></tr>
      <tr><td>输入密码: </td>
      <td><input type="password" id="password" name="password" value="1111"></td>
      <td></td></tr>
      <tr><td>确认密码: </td>
      <td><input type="password" id="passwordConfirm" name="passwordConfirm" value="1112"></td>
      <td><span id="passwordConfirmSpan"></span></td></tr>
      <tr><td colspan="3" align="center">
        <input type="submit" id="submit" value="注册">
        <input type="reset" value="重置"></td></tr></table>
    </form>
    <script charset="UTF-8" type="text/javascript" src="4-eventListener.js"></script>
  </body>
</html>
```

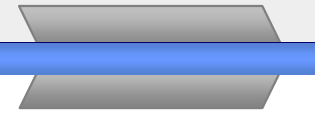
JS 4-eventListener.js > ...

```
1  function $(id){
2      return document.getElementById(id);
3  }
4  window.onload = function (){
5      $$("#passwordConfirm").addEventListener("blur", checkPassword);
6      $$("#submit").addEventListener("click", submitForm);
7  }
8  function checkPassword(){
9      if ($$("#password").value != $$("#passwordConfirm").value){
10         $$("#passwordConfirm" + "Span").innerHTML = "密码不一致! ";
11         $$("#passwordConfirm" + "Span").className = "failure";
12     }else {
13         $$("#passwordConfirm" + "Span").innerHTML = "密码验证一致! ";
14         $$("#passwordConfirm" + "Span").className = "success";
15     }
16 }
17 function submitForm(){
18     // 表单验证后提交!
19     // $$("#register").submit();
20     $$("#register").onsubmit;
21 }
```



4-eventListener_style.css > ...

```
1  input{border: 1px solid #0066CC;}
2  form{font-size: 12px;}
3  table{border: 0;margin: 0;padding: 0;}
4  .success{
5      background: #f5f5f5;
6      font-weight: bold;
7      color: #000000;
8      border: 1px #009900 solid;
9  }
10 .failure{
11     background: #f5f5f5;
12     font-weight: bold;
13     color: #000000;
14     border: 1px #990000 solid;
15 }
```

🐼 4-register_submit.php

```
1  <?php
2      header("Content-Type:text/html;charset=UTF-8");
3      echo "用户名: " . $_REQUEST["userName"] . "<br>";
4      echo "密码: " . $_REQUEST["password"] . "<br>";
5      echo "确认密码: " . $_REQUEST["passwordConfirm"] . "<br>";
6  ?>
```



← → ↻ 🏠 ⓘ localhost/... 🔗 ☆ 🧩 📺 👤

用户名:

输入密码:

确认密码:

用户名:

输入密码:

确认密码: 密码不一致!

用户名:

输入密码:

确认密码: 密码验证一致!

← → ↻ 🏠

📘 ⓘ localhost/register_submit.php?userName=用户名&password=1111&passwordConfi...

用户名: 用户名
密码: 1111
确认密码: 1111

← → ↻ 🏠

📘 ⓘ localhost/register_submit.php

用户名: 用户名
密码: 1111
确认密码: 1112