## DOM Document Object Model 文档对象模型

Dom是html和xml的应用程序接口，DOM把整个页面映射为一个多层节点结构。借助DOM提供的API，开发人员可以轻松地删除、添加、替换或修改任何节点。

HTML文档中的所有内容都可以称之为节点。

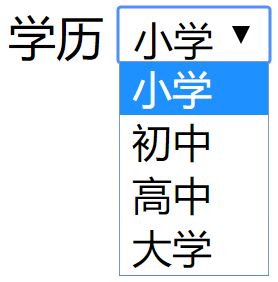
|  |  |  |  |
| --- | --- | --- | --- |
| 节点属性  常见节点 | nodetype | nodeValue | nodeName |
| 元素节点 | 1 | null | 大写的标签名 |
| 属性节点 | 2 | 属性值 | 属性名 |
| 文本节点 | 3 | 文本 | #text |
| 注释节点 | 8 | 注释中的内容 | #comment |
| 元素节点属性：innerHTML和innerText | | | |

## 获取节点的方法（找对象）

|  |  |  |
| --- | --- | --- |
| getElementsByTagName() | getElementsByClassName() | 获取一个数组 |
| getElementsByName() | querySelectorAll()H5 |
| querySelector()H5 | getElementById() | 只获取一个 |

|  |  |  |  |
| --- | --- | --- | --- |
| **DOM属性——节点与节点的关系** | | | |
| parentNode | 获取父节点 | 老爹 |
| childNodes | 获取所有的子节点，返回一个节点数组 | 所有子女 |
| children | 只获取子元素，不是DOM标准方法 | IE6-8不兼容 |
| firstChild | 获取第一个子节点 | 老大 |
| lastChild | 获取最后一个子节点 | 老幺 |
| previousSibling | 获取上一个兄弟节点 | 哥 |
| nextSibling | 获取下一个兄弟节点 | 弟 |
| 属性节点不是这个标签的子节点，类似“经纪人”。 | | |

|  |  |  |
| --- | --- | --- |
| **节 点 操 作** | | |
|  | 作用 | 备注 |
| createElement() | 增加新元素  创建  添加  联合使用 | 只能被document调用 |
| createTextNode() |  |
| appendChild() | 必须通过父元素来添加子元素 |
| insertBefore(new, ref) |
| cloneNode(boolean) | 自我复制 | 深拷备、浅拷备 |
| removeChild() | 删除 | 必须通过父元素来删除子元素 |
| removeAttribute() |  |  |
| replaceChild(new, old） |  | 必须通过父元素替换子元素 |
| getAttribute() |  |  |
| setAttribute(name, value) |  |  |

练习

1. 获取下拉框中所有选项的值
2. 在下拉框中增加一项“硕士”

## 事件

例：点击按钮，关闭广告。

#### 事件三要素：

1. 事件源（DOM元素）
2. 事件
3. 事件处理程序

|  |  |  |
| --- | --- | --- |
| **JS事件** | | |
| 鼠标事件 | onclick |  |
| onmouseover |  |
| onmouseout |  |
| 键盘事件 | onkeydown |  |
| onkeyup |  |
| 表单 | onfocus |  |
| onblur |  |
| onchange | 下拉框 |
| onsubmit | 提交按钮被点击 |
|  | onload |  |

1. 鼠标移上，箭头出现
2. 表单验证
3. **事件绑定**
4. 直接绑定：on...
5. 高级绑定：addEventListener("click", fn)

#### 事件解绑

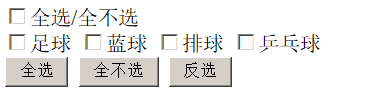
1. 直接解绑：null
2. 事件移除：removeEventListener

#### **事件委托**

1. 原理
2. 优点

### 练习2





需求：

1. 当选择 全选/全不选时，下面的四个爱好，都会与我们全选/全不选一样。
2. 当选择全选按钮时，要求四个爱好项全都选择。
3. 当选择全不选按钮时，要求四个爱好项全都不选择。
4. 当选择反选时，要求四个爱好项，选择的取消，没有选择的选中。

怎样控制checkbox被选择了？

只需要通过一个属性checked设置其值为true/false就可以。

<!DOCTYPE html>

<html>

<head>

<title>dom练习2</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">

<script type="text/javascript">

//当id=main的checkbox被点击后执行

function selectCk() {

//1.得到id=main的checkbox的checked的值.

var flag = document.getElementById("main").checked;

//2.获取所有的name=hobby的checkbox元素

var hobby = document.getElementsByName("hobby");

//3.将name=hobby的checkbox元素的checked属性值设置为flag.

for ( var i = 0; i < hobby.length; i++) {

hobby[i].checked = flag;

}

}

//全选/全不选按钮的中全选

function selectAll(flag) {

//1.获取所有的name=hobby的checkbox元素

var hobby = document.getElementsByName("hobby");

//2.将name=hobby的checkbox元素的checked属性值设置为flag.

for ( var i = 0; i < hobby.length; i++) {

hobby[i].checked = flag;

}

}

/\*

//全不选

function selectNotAll(){

//1.获取所有的name=hobby的checkbox元素

var hobby = document.getElementsByName("hobby");

//2.将name=hobby的checkbox元素的checked属性值设置为flag.

for ( var i = 0; i < hobby.length; i++) {

hobby[i].checked = false;

}

}\*/

//反选操作

function selectReverse(){

//1.获取所有的name=hobby的checkbox元素

var hobby = document.getElementsByName("hobby");

//2.将name=hobby的checkbox元素的checked属性值设置为flag.

for ( var i = 0; i < hobby.length; i++) {

hobby[i].checked =!hobby[i].checked ;

}

}

</script>

</head>

<body>

<input type="checkbox" id="main" onclick="selectCk()">全选/全不选

<br>

<input type="checkbox" name="hobby">足球&nbsp;

<input type="checkbox" name="hobby">蓝球&nbsp;

<input type="checkbox" name="hobby">排球&nbsp;

<input type="checkbox" name="hobby">乒乓球&nbsp;

<br>

<input type="button" value="全选" onclick="selectAll(true)">&nbsp;

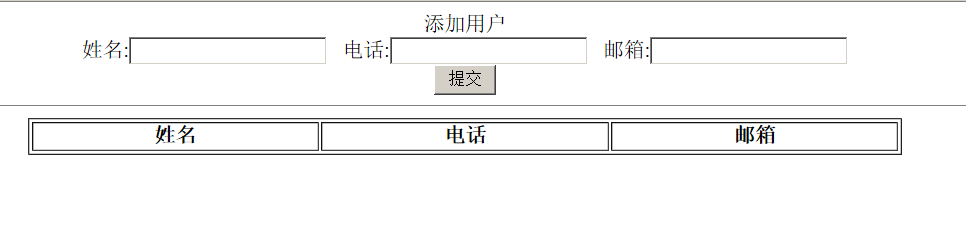
<input type="button" value="全不选" onclick="selectAll(false)">&nbsp;

<input type="button" value="反选" onclick="selectReverse()">&nbsp;

</body>

</html>

### 练习3



需求:

在文本框中输入信息后，点击提交按钮，会将信息动态生成表格中。

<!DOCTYPE html>

<html>

<head>

<title>dom练习3</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">

<script type="text/javascript">

function fun() {

//1.获取姓名，邮箱，电话信息

var username = document.getElementById("username").value;

var telephone = document.getElementById("telephone").value;

var email = document.getElementById("email").value;

/\*

//2.1使用xmldom操作

//通过createElement来创建tr td在td中添加上username,telephone,email信息。

//通过appendChild方法组装。

//2.1.1创建一个tr元素

var tr= document.createElement("tr");

//2.1.2创建三个td元素

var td1=document.createElement("td");

var td2=document.createElement("td");

var td3=document.createElement("td");

td1.innerHTML=username;

td2.innerHTML=telephone;

td3.innerHTML=email;

//2.1.3将三个td添加到tr中。

tr.appendChild(td1);

tr.appendChild(td2);

tr.appendChild(td3);

//2.1.4将tr添加到table中。

document.getElementById("tab").appendChild(tr);

\*/

//2.2.使用htmldom操作 这种方案也不是特别方便 可以使用table tablerow tablecell对象。

//2.3可以直接使用innerHTML来组装数组。

var trmsg = "<tr><td>" + username + "</td><td>" + telephone

+ "</td><td>" + email + "</td></tr>";

document.getElementById("tab").innerHTML += trmsg;

//2.4 文本框清空

document.getElementById("username").value = "";

document.getElementById("telephone").value = "";

document.getElementById("email").value = "";

}

</script>

</head>

<body>

<div align="center">

添加用户<br> 姓名:<input type="text" id="username"> &nbsp; 电话:<input

type="text" id="telephone"> &nbsp; 邮箱:<input type="text"

id="email"><br> <input type="button" value="提交"

onclick="fun()">

</div>

<hr>

<div align="center">

<table border='1' width="65%" id="tab">

<tr>

<th>姓名</th>

<th>电话</th>

<th>邮箱</th>

</tr>

</table>

</div>

</body>

</html>

## Javascript中的bom