

JavaScript

JavaScript

JavaScript介绍

特点

JavaScript和HTML代码的结合方式

第一种方式

第二种方式

JavaScript的变量和数据类型

JavaScript的变量类型

JavaScript里的特殊值

JavaScript定义变量格式

JavaScript的运算

JavaScript的关系（比较）运算

JavaScript的逻辑运算

&& 与运算

|| 或运算

JavaScript的数组

数组定义方式

函数

函数的两种定义方式

第一种

第二种

函数不允许重载

函数的arguments隐形参数

需求

JavaScript中的自定义对象

Object形式的自定义对象

{ }花括号形式的自定义对象

JavaScript的事件

常用的事件

事件的注册

静态注册事件

动态注册事件

动态注册基本步骤:

onload事件

静态注册onload事件

动态注册onload事件

onclick事件

静态注册onclick事件

动态注册onclick事件

onblur失去焦点事件

静态注册失去焦点事件

动态失去焦点事件

onchange事件

静态注册onchange事件

动态注册onchange事件

onsubmit表单提交事件

静态注册onsubmit事件

动态注册onsubmit事件

DOM模型

Document对象

Document对象的理解

Document对象中的方法介绍

需求

正则表达式对象

两种常见的验证提示效果

第一种

第二种

getElementsByTagName方法

getElementsByTagName方法

document对象三个查询方法的使用注意事项

节点的常用属性和方法

方法

属性

DOM查询练习

createElement() && appendChild()

JavaScript介绍

JavaScript语言诞生主要是完成页面的数据验证。因此它运行在客户端，需要运行浏览器来解析执行JavaScript代码。

JavaScript是弱类型，Java是强类型。弱类型指类型可变。强类型指定义变量时，类型已确定，且不可变。

特点

- 交互性（信息交互）
- 安全性（不允许直接访问本地硬盘）
- 跨平台性（只要是可解释JavaScript的浏览器都可以执行，与平台无关）

JavaScript和HTML代码的结合方式

第一种方式

在head标签中，或者在body标签中，使用script标签来书写JavaScript代码。

```
1 <script type = "text/javascript">
2     alert("Hello JavaScript!");
3 </script>
```

第二种方式

使用script标签引入单独的JavaScript代码文件

```
1 <script type = "text/javascript" src = "路径.js"></script>
```

JavaScript的变量和数据类型

JavaScript的变量类型

- 数值类型 number
- 字符串类型 string
- 对象类型 object
- 布尔类型 boolean
- 函数类型 function

JavaScript里的特殊值

- undefined: 未定义，所有JavaScript变量为赋予初始值时，默认值为undefined
- null
- NaN: Not a Number - 非数值

```
1 var a = 12
2 var b = "abc";
3 alert(a * b); //NaN 非数值 非数字
```

JavaScript定义变量格式

```
1 var 变量名 = 值;
```

JavaScript的运算

JavaScript的关系（比较）运算

- 等于: == 等于是简单的做字面的比较
- 全等于: === 除了做字面值的比较之外，还会比较两个变量的数据类型

```
1 var a = "12"
2 var b = 12;
3 alert(a == b); //true
4 alert(a === b); //false
```

JavaScript的逻辑运算

在JavaScript中，所有变量都可以作为一个Boolean类型的变量去使用，且0、null、undefined、""（空串）被认为是false。

&& 与运算

- 第一，当表达式全为真时，返回最后一个表达式的值。
- 第二，当表达式有一个为假时，返回第一个为假的表达式的值。

|| 或运算

- 第一，当表达式全为假时，返回最后一个表达式的值
- 第二，只要有一个表达式为真。就会把第一个为真的表达式的值返回

JavaScript的数组

数组定义方式

```
1 var 数组名 = [];  
2 var 数组名 = [1, 'abc', true];
```

注意：JavaScript中的数组，只要我们通过数组下标赋值，那么最大的下标值，就会自动的给数组做扩容操作。

```
1 for(var i = 0; i < arr.length; i++){  
2     alert(arr[i]);  
3 }
```

函数

函数的两种定义方式

第一种

可以使用function关键字定义函数

```
1 function 函数名(形参列表){  
2     函数体  
3 }
```

```
1 function fun(a, b){  
2     alert(a + b);  
3 }  
4 fun("Hello", "JavaScript");
```

注意：JavaScript中定义带有返回值的函数，直接在函数体内加上**return**语句即可。

第二种

```
1 var 函数名 = function(形参列表){函数体}
```

函数不允许重载

JavaScript 中的函数重载 JavaScript 中没有真正意义上的函数重载，因为javascript 中同一作用域下的同名函数，前者会被后者覆盖，但是可通过其他方法间接实现重载同样的效果，JavaScript中的函数没有签名，它的参数是由包含零的多个数组来表示的。

函数的arguments隐形参数

- 只在function函数内
- 定义：在function函数中不需要定义，但却可以直接用来获取所有参数的变量。称为隐形参数。
- 隐形参数特别像Java中的可变长参数
- `public void function(Object... args)`
- JavaScript中隐形参数与Java的可变长参数一样，操作类似数组。

```
1 function fun(){
2     alert(arguments.length);
3
4     for(var i = 0; i < arguments.length; i++){
5         alert(arguments[i]);
6     }
7 }
```

需求

要求编写一个函数，用于计算所有参数相加的和并返回、

```

1  function sum(num1, num2){
2      var result = 0;
3      for(var i = 0; i < arguments.length; i++){
4          //避免字符串或者其他非number类型造成除了参数相加的其他操作
5          if(typeof(arguments[i]) == "number"){
6              result += arguments[i];
7          }
8      }
9      return result;
10 }
11 alert(sum(num1, num2, ...));

```

JavaScript中的自定义对象

Object形式的自定义对象

```

1  var 变量名 = new Object();//对象实例（空对象）
2  变量名.属性名 = 值;//定义一个属性
3  变量名.函数名 = function(){}//定义一个函数

```

{ }花括号形式的自定义对象

```

1  var 变量名 = {
2      属性名:值,
3      属性名:值,
4      函数名:function(){}//最后一个不需要加逗号
5  };

```

JavaScript的事件

事件是电脑输入设备与页面进行交互的响应

常用的事件

- **onload** 加载完成事件：页面加载完成之后，常用于做页面JavaScript代码初始化操作。

- **onclick** 单击事件：常用于按钮的点击响应操作。
- **onblur** 失去焦点事件：常用于输入框失去焦点后验证其输入内容是否合法。
- **onchange** 内容发送改变事件：常用于下拉列表和输入框发送改变后操作。
- **onsubmit** 表单提交事件：常用于表单提交前，验证所有表单项是否合法。

事件的注册

告诉浏览器，当事件响应后要执行哪些操作代码，称为事件注册或事件绑定。

静态注册事件

通过HTML标签的事件属性直接赋予事件响应后的代码，这种方式被称为静态注册。

动态注册事件

是指先通过JavaScript代码得到标签的dom对象. 事件名=function(){}，这种形式赋予事件响应后的代码，被称为动态注册。

动态注册基本步骤：

1. 获取标签对象
2. 标签对象.事件名 = function(){}

onload事件

静态注册onload事件

onload 事件是浏览器解析完页面就会自动触发的事件

onload事件的静态注册是在标签中

```
1 <body onload = "alert('静态注册onload事件')">
```


动态注册onload事件

```
1 <script type = "text/javascript">
2     //onload事件动态注册，固定写法
3     window.onload = function(){
4         alert("动态注册的onload事件");
5     }
6 </script>
```

onclick事件

静态注册onclick事件

```
1 <button onclick = "onclickFun();"></button>
```

动态注册onclick事件

```
1 <script type = "text/javascript">
2     //动态注册onclick事件
3     //1. 获取标签对象
4     //document对象是文档的根节点
5     var btnObj = document.getElementById("btn01");
6     //2. 通过标签对象.事件名 = function(){}
7     btnObj.onclick = function(){
8         alert("动态注册onclick事件");
9     }
10 </script>
11 </head>
12 <body>
13     <button id = "btn01">按钮</button>
```

onblur失去焦点事件

静态注册失去焦点事件

```
1 <script type = "text/javascript">
2     function onBlurFun(){
3         //console是控制台对象，是由JavaScript语言提供，专门用来向浏览器的
           控制器打印输出，用于测试使用
4         //log()是打印的方法
5         console.log("静态注册失去焦点事件");
6     }
7 </script>
8 </head>
9 <body>
10     用户名: <input type="text" onBlur="onBlurFun();"><br/>
11     密码: <input type="text"><br/>
```

动态失去焦点事件

```
1 <script type = "text/javascript">
2     window.onload = function(){
3         var passwordObj = document.getElementById("password");
4         passwordObj.onblur = function(){
5             console.log("动态注册失去焦点事件");
6         }
7     }
8 </script>
9 </head>
10 <body>
11     用户名: <input type="text" onBlur="onBlurFun();"><br/>
12     密码: <input id="password" type="text"><br/>
```

onchange事件

静态注册onchange事件

```

1 <script type="text/javascript">
2     function onchangeFun(){
3         alert("Number is changed");
4     }
5 </script>
6 </head>
7 <body>
8     <select onchange="onchangeFun();">
9         <option>Number</option>
10     ...

```

动态注册onchange事件

```

1 <script type="text/javascript">
2     //1. 获取标签对象
3     window.onload = function(){
4         var selObj = document.getElementById("se101"){
5             //2. 通过标签对象.事件名 = function(){}
6             selObj.onchange = function(){
7                 alert("Number is changed");
8             }
9         }
10    </script>
11 </head>
12 <body>
13     <select id = "se101">
14         <option>Number</option>
15     ...

```

onsubmit表单提交事件

静态注册onsubmit事件

```

1  <script type = "text/javascript">
2      function onsubmitFun(){
3          if( ){//验证其合法性，如表单项不合法，则阻止提交
4              alert("静态注册表单提交事件---不合法");
5              return false;
6          }
7      }
8  </script>
9  </head>
10 <body>
11     <form action = "http://localhost:8080" method = "post"
12     onsubmit = "return onsubmitFun()">
13         <input type = "submit" value = "静态注册"/>
14     </form>

```

动态注册onsubmit事件

```

1  <script type = "text/javascript">
2      window.onload = function(){
3          var formObj = document.getElementById("form01");
4          formObj.onsubmit = function(){
5              alert("动态注册onsubmit事件");
6              return false;//阻止
7          }
8      }
9  </script>
10 </head>
11 <body>
12     <form action = "http://localhost:8080" method = "post" id =
13     "form01">
14         <input type = "submit" value = "静态注册"/>
15     </form>

```

DOM模型

*DOM*全称是*Document Object Model* 文档对象模型。是把文档中的标签，属性，文本，转换成为对象来管理。

Document对象

Document文档树内存结构及使用方法的[参考网址](#)（点击可跳转）

Document对象的理解

- Document管理了所有的HTML文档内容
- document是一种树结构文档，有层级关系
- document让所有标签都对象化
- 可以通过document访问所有的标签对象

Document对象中的方法介绍

- document.getElementById(elementId)
- document.getElementsByName(elementName)
- document.getElementsByTagName(tagname)
- document.createElement(tagName)

需求

当用户点击了校验按钮，要获取输出框中的内容。然后验证其是否合法。

验证的规则是，必须有字母、数字、下划线组成，并且长度为5到12位

```
1 <script type="text/javascript">
2     function onclickFun(){
3         //获取标签对象
4         var usernameObj = document.getElementById("username");
5         var usernameText = usernameObj.value;
6         var patt = /^w{5, 12}$/;
7         //test()方法用于测试输入字符串是否符合输入规则
8         if(patt.test(usernameText)){
9             alert("用户名合法");
10        }else{
11            alert("用户名不合法");
12        }
13    }
14 </script>
```

```

15 </head>
16 <body>
17     用户名: <input type="text" id="username" value="username"/>
18     <button onclick="onclickFun()">校验</button>

```

正则表达式对象

参考资料[菜鸟教程-正则表达式](#)（点击可跳转）

两种常见的验证提示效果

第一种

```

1 <script type="text/javascript">
2     function onclickFun(){
3         //获取标签对象
4         var usernameObj = document.getElementById("username");
5         var usernameText = usernameObj.value;
6         var patt = /^w{5, 12}$/;
7         //test()方法用于测试输入字符串是否符合输入规则
8         var usernameSpanObj =
document.getElementById("usernameSpan");
9         //innerHTML 表示起始标签和结束标签中的内容
10        //innerHTML 此属性可读，可写
11        if(patt.test(usernameText)){
12            usernameSpanObj.innerHTML = "用户名合法";
13        }else{
14            usernameSpanObj.innerHTML = "用户名不合法";
15        }
16    }
17 </script>
18 </head>
19 <body>
20     用户名: <input type="text" id="username" value="username"/>
21         <span id="usernameSpan" style="color:red;"></span>
22     <button onclick="onclickFun()">校验</button>

```

第二种

- 1 在第一种的情况下，将 `usernameSpanObj.innerHTML = ""`

getElementsByTagName方法

返回带有指定名称的对象集合

```

1      <script type="text/javascript">
2          function checkALL(){
3              //document.getElementsByTagName();根据指定的name属性查询返回多个标签对象集合
4              //集合的操作与数组一样
5              //集合中每个元素都为dom对象
6              //集合中元素顺序位HTML页面中从上到下的顺序
7              var hobbies = document.getElementsByTagName("hobby");
8              //checked表示复选框的选中状态，选中则为true，反之则为false
9              //checked属性可读，可写
10             for(var i = 0; i < hobbies.length; i++){
11                 hobbies[i].checked = true;
12             }
13         }
14         function checkNo(){
15             var hobbies = document.getElementsByTagName("hobby");
16             for(var i = 0; i < hobbies.length; i++){
17                 hobbies[i].checked = false;
18             }
19         }
20         function checkReverse(){
21             var hobbies = document.getElementsByTagName("hobby");
22             for(var i =0; i < hobbies.length; i++){
23                 if(hobbies[i] == true)
24                     hobbies[i] = false;
25                 if(hobbies[i] == false)
26                     hobbies[i] = true;
27             }
28         }
29     </script>

```

```

30 </head>
31 <body>
32     兴趣爱好:
33     <input type="checkbox" name="hobby" value="cpp">C++
34     <input type="checkbox" name="hobby" value="java">Java
35     <input type="checkbox" name="hobby"
value="js">JavaScript
36     <button onclick="checkALL">全选</button>
37     <button onclick="checkNo">全不选</button>
38     <button onclick="checkReverse">反选</button>

```

getElementsByTagName方法

返回带有指定签名的对象集合

```

1 <script type="text/javascript">
2     function checkALL(){
3         //document.getElementsByTagName();根据指定标签名来进行查
        询并返回集合
4         //集合的操作与数组一样
5         //集合中每个元素都为dom对象
6         //集合中元素顺序位HTML页面中从上到下的顺序
7         var inputs = document.getElementsByTagName("input");
8         //checked表示复选框的选中状态，选中则为true，反之则为false
9         //checked属性可读，可写
10        for(var i = 0; i < inputs.length; i++){
11            inputs[i].checked = true;
12        }
13    }
14 </script>
15 </head>
16 <body>
17     兴趣爱好:
18     <input type="checkbox" value="cpp">C++
19     <input type="checkbox" value="java">Java
20     <input type="checkbox" value="js">JavaScript
21     <button onclick="checkALL">全选</button>

```


document对象三个查询方法的使用注意事项

- document对象的三个查询方法，如果有id属性，优先使用**getElementById**方法进行查询。如果没有id属性，则优先使用**getElementsByName**。如果id属性和name属性都没有最后再按标签名查**getElementsByTagName**。
- 三个方法都一定要在页面加载完成之后执行，才能查询到标签对象。

节点的常用属性和方法

节点是标签对象

方法

getElementsByTagName()

获取当前节点的指定标签名孩子节点

appendChild(oChildNode)

可以添加一个子节点，oChildNode是要添加的孩子节点

属性

childNodes

获取当前节点的所有节点

firstNode

获取当前节点的第一个子节点

lastNode

获取当前节点的最后一个子节点

parentNode

获取当前节点的父节点

nextSibling

获取当前节点的下一个节点

previousSibling

获取当前节点的上一个节点

className

用于获取或设置标签的class属性值

innerHTML

获取/设置起始标签和结束标签中的内容

innerText

获取/设置起始标签和结束标签中的文本

DOM查询练习

createElement() && appendChild()

```
1 <script type="text/javascript">
2     window.onload = function(){
3         //使用JavaScript代码创建HTML标签，并显示在页面上
4         var divObj = document.createElement("div");
5         var divObj.innerHTML = "创建div标签";
6         document.body.appendChild(divObj);
7     }
8 </script>
```

文末¹

[返回文首](#)

1. 点击到达文末。↩