# day29-前端【javascript高级】

# 学习目标

01\_能够使用正则表达式进行表单的验证

02\_能够使用window对象常用的方法

03\_能够使用location对象常用的方法和属性

04\_能够使用history对象常用的方法

05\_能够使用DOM中来查找节点

06\_能够使用DOM来增删改节点

07\_能够使用JavaScript对CSS样式进行操作

# 项目清单

|  |
| --- |
|  |

# 回顾与提问

1\_JavaScript有哪几种数据类型？

|  |
| --- |
| number/boolean/undefined/string/ |

2\_以下代码是什么意思？

|  |
| --- |
| <input type=”button” value=”点我” onclick=”hello()”/> |

# 问题引入

HTML是做网页的结构和内容的，

CSS是对网页的样式进行更加精确性的修饰的，

但我们希望网页有时能够和用户进行直接的交互，从而引导用户进行操作，如：表单验证。

那么是什么技术可以实现网页的交互功能呢？

答案是：JavaScript，它是客户端网页脚本编程语言。

# 第01章 正则表达式

## 1.1正则表达式的规则

一个具有特殊意义的字符串。

正则表达式在多种语言中都可以使用，如：在Java中可以用正则表达式，在

JavaScript中也可以使用正则表达式。

|  |  |
| --- | --- |
| **符号** | **作用** |
| **\d** | **数字** |
| **\D** | **非数字** |
| **\w** | **单词：a-zA-Z0-9\_** |
| **\W** | **非单词** |
| **.** | **通配符，匹配任意字符** |
| **{n}** | **匹配n次** |
| **{n,}** | **大于或等于n次** |
| **{n,m}** | **在n和m之间** |
| **+** | **1~n次** |
| **\*** | **0~n次** |
| **?** | **0~1次** |
| **^** | **匹配开头** |
| **$** | **匹配结尾** |
| **[13579]** | **匹配1或3或5或7或9，即这5个数字之一** |
| **[0-9]** | **匹配1到 9，即0,1,2,3,4,5,6,7,8,9中的数字之一** |

你也可以参见JDK中：Pattern的说明。

|  |
| --- |
|  |

## 1.2正则表达式的创建方式

### 1.2.1方式一

|  |
| --- |
| **var reg = new RegExp("正则表达式");** |

说明：正则表达式在JS中是一个对象。Regular Expression。

### 1.2.2方式二

|  |
| --- |
| **var reg = /正则表达式/;** |

说明：以/开头，以/结尾，中间是正则表达式。

### 1.2.3两种方式的区别

1\_在JS中，正则表达式的两种声明方式对于“\d、\D”之类的匹配模式中，

方式一：需要转义；如：new RegExp(“\\d”)

方式二：无需转义。如：/\d/

2\_方式一：支持字符串拼接，支持变量，更加灵活；如：new RegExp(“\\d”+变量)

方式二：对于固定的表达式，书写起来方便快捷、更加直观。如：/\d{3}/

### 1.2.4匹配模式

|  |
| --- |
| **i ：忽略大小写比较** |

|  |
| --- |
| **匹配模式的两种写法** |
| **var reg = new RegExp("正则表达式", "匹配模式"); 如：new RegExp(“abc”,”i”)** |
| **var reg = /正则表达式/匹配模式; 如：/abc/i** |

## 1.3正则表达式中的常用方法

|  |  |
| --- | --- |
| **JS中正则表达式的方法** | **说明** |
| **boolean test("字符串")** | **如果正则表达式匹配字符串，返回true，否则返回false** |

示例

演示正则表达式的二种创建方式，匹配模式和常用方式。

|  |
| --- |
|  |

01\_regexp.html

|  |
| --- |
| <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>演示正则表达式的二种创建方式，匹配模式和常用方式</**title**> </**head**> <**body**>  <**script**>  **var *reg1*** = **new *RegExp***(**"\\d{3}"**);  **var *result1*** = ***reg1***.test(**"123"**);   **var *reg2*** = /\d{3}/;  **var *result2*** = ***reg2***.test(**"123"**);   **var *reg3*** = /cat/i;  **var *result3*** = ***reg3***.test(**"Cat"**);   ***window***.document.write(***result1***+**"<br/>"**);  ***window***.document.write(***result2***+**"<br/>"**);  ***window***.document.write(***result3***+**"<br/>"**);  </**script**> </**body**> </**html**> |

## 1.4 JS匹配上与Java中的不同

Java默认情况下必须要精确匹配，而在JS中默认是模糊匹配，只要字符串包含了正则表达式的内容就返回true。

|  |  |  |  |
| --- | --- | --- | --- |
| **正则表达式** | **匹配字符串** | **Java中匹配结果** | **JavaScript中匹配结果** |
| **\d{3}** | **a123b** | **false** | **true** |
| **^\d{3}** | **123b** | **false** | **true** |
| **\d{3}$** | **a123** | **false** | **true** |
| **^\d{3}$ 推荐** | **123** | **true** | **true** |

示例

|  |
| --- |
|  |

02\_regexp.html

|  |
| --- |
| <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>演示正则表达式的二种创建方式，匹配模式和常用方式。</**title**> </**head**> <**body**>  <**script**>  *//用new RegExp()来创建* **var *regexp1*** = **new *RegExp***(**"^\\d{11}$"**,**"i"**);  ***document***.write(***regexp1***.test(**"15876549180"**)+**"<br/>"**);   *//用//来创建* **var *regexp2*** = /^\d{11}$/;  ***document***.write(***regexp2***.test(**"158765491800"**)+**"<br/>"**);   *//用test()方法  //第1位必须是1，第2位可以是3/5/8之一，第3位到第11位，是0-9之间的任意值  //var regexp3 = /^1[358]\d{9}$/;* **var *regexp3*** = /^1[358][0-9]{9}$/;  ***document***.write(***regexp3***.test(**"18876549180"**)+**"<br/>"**);   ***document***.write(**"<hr/>"**);   *//用i进行大小写忽略比较* **var *regexp4*** = /^cat$/i;  ***document***.write(***regexp4***.test(**"cat"**)+**"<br/>"**);*//true* ***document***.write(***regexp4***.test(**"CAT"**)+**"<br/>"**);*//true* ***document***.write(***regexp4***.test(**"Cat"**)+**"<br/>"**);*//true* ***document***.write(***regexp4***.test(**"mycat"**)+**"<br/>"**);*//false* ***document***.write(***regexp4***.test(**"catmy"**)+**"<br/>"**);*//false* ***document***.write(***regexp4***.test(**"mycatmy"**)+**"<br/>"**);*//false* ***document***.write(***regexp4***.test(**"mycbtmy"**)+**"<br/>"**);*//false* </**script**> </**body**> </**html**> |

## 1.5案例：注册表单验证

注意，这里不用编写HTML页面，只需书写JS代码，

参见：素材/注册表单验证。

### 1.5.1案例需求

用户注册，需要进行如下验证，请在JS中使用正则表达式进行验证。

1\_用户名：只能由英文字母和数字组成，长度为4～16个字符，并且以英文字母开头

/^[a-zA-Z][a-zA-Z0-9]{3,15}$/

2\_密码： 数字6-20个字符

/^[0-9]{6,20}$/

3\_确认密码：两次密码要相同

---------------------------------------------------------------------1，2，3需求为课上完成，

---------------------------------------------------------------------4，5，6需求为课后完成。

4\_电子邮箱：符合邮箱地址的格式

/^\w+@\w+(.[a-zA-Z]{2,3}){1,2}$/

5\_手机号：符合手机号码的格式

/^1\d{10}$/

6\_生日：生日的年份在1900～2009之间，生日格式为1980-5-12或1988-05-04的形式，/^((19\d{2})|(200\d))-(0?[1-9]|1[0-2])-(0?[1-9]|[1-2]\d|3[0-1])$/

### 1.5.2案例效果

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

### 1.5.3案例分析

1\_每个验证项创建一个方法。

2\_创建正则表达式。

3\_得到文本框中输入的值。

4\_如果不匹配，返回false，并以报警框形式提示用户。

如果匹配，返回true。

5\_写一个验证表单中所有的项的方法check()，所有的方法都返回true，这个方法才返回true，

并提交表单，这里是<form onsubmit=”return check()”>。

### 1.5.4案例代码

03\_register.html

|  |
| --- |
| <!DOCTYPE **html** PUBLIC **"-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**> <**html xmlns="http://www.w3.org/1999/xhtml"**>  <**head**>  <**meta http-equiv="Content-Type" content="text/html; charset=utf-8"** />  <**title**>注册表单验证</**title**>  <**style type="text/css"**>  **body** {  **margin**: 0;  **padding**: 0;  **font-size**: 12**px**;  **line-height**: 20**px**;  }  .**main** {  **width**: 525**px**;  **margin-left**: **auto**;  **margin-right**: **auto**;  }  .**hr\_1** {  **font-size**: 14**px**;  **font-weight**: **bold**;  **color**: **#3275c3**;  **height**: 35**px**;  **border-bottom-width**: 2**px**;  **border-bottom-style**: **solid**;  **border-bottom-color**: **#3275c3**;  **vertical-align**: **bottom**;  **padding-left**: 12**px**;  }  .**left** {  **text-align**: **right**;  **width**: 80**px**;  **height**: 25**px**;  **padding-right**: 5**px**;  }  .**center** {  **width**: 280**px**;  }  .**in** {  **width**: 130**px**;  **height**: 16**px**;  **border**: **solid** 1**px #79abea**;  }  .**red** {  **color**: **#cc0000**;  **font-weight**: **bold**;  }  **div** {  **color**: **#F00**;  }  </**style**>  </**head**>  <**body**>  *<!--onsubmit是表单提交事件，如果返回值是true的话，提交表单，如果返回值为false的话，不提交表单  onsubmit和验证联系在一起，验证通过，返回true，没通过，返回false  -->* <**form action="01\_regexp.html" method="post" id="myform" onsubmit="return** *check*()**"**>  <**table class="main" border="0" cellspacing="0" cellpadding="0"**>  <**tr**>  <**td**><**img src="img/logo.jpg" alt="logo"** /><**img src="img/banner.jpg" alt="banner"** /></**td**>  </**tr**>  <**tr**>  <**td class="hr\_1"**>新用户注册</**td**>  </**tr**>  <**tr**>  <**td style="height**:10**px**;**"**></**td**>  </**tr**>  <**tr**>  <**td**>  <**table width="100%" border="0" cellspacing="0" cellpadding="0"**>  <**tr**>  <**td class="left"**>用户名：</**td**>  <**td class="center"**>  <**input id="user" name="user" type="text" class="in"** />  </**td**>  </**tr**>  <**tr**>  <**td class="left"**>密码：</**td**>  <**td class="center"**>  <**input id="pwd" name="pwd" type="password" class="in"** />  </**td**>  </**tr**>  <**tr**>  <**td class="left"**>确认密码：</**td**>  <**td class="center"**>  <**input id="repwd" name="repwd" type="password" class="in"**/>  </**td**>  </**tr**>  *<!--<tr>-->  <!--<td class="left">电子邮箱：</td>-->  <!--<td class="center">-->  <!--<input id="email" name="email" type="text" class="in"/>-->  <!--</td>-->  <!--</tr>-->  <!--<tr>-->  <!--<td class="left">手机号码：</td>-->  <!--<td class="center">-->  <!--<input id="mobile" name="mobile" type="text" class="in"/>-->  <!--</td>-->  <!--</tr>-->  <!--<tr>-->  <!--<td class="left">生日：</td>-->  <!--<td class="center">-->  <!--<input id="birth" name="birth" type="text" class="in"/>-->  <!--</td>-->  <!--</tr>-->* <**tr**>  <**td class="left"**>**&nbsp;**</**td**>  <**td class="center"**>  <**input name="" type="image" src="img/register.jpg"** />  </**td**>  </**tr**>  </**table**>  </**tr**>  </**table**>  </**form**>       <**script**>  *//针对用户名验证的函数* **function** *checkUsername*(username){*//" "* **var** flag = **false**;  *//去掉二边的空格* username = username.trim();  *//如果长度不为0* **if**(username.length > 0){  *//如果符合要求* **if**(/^[a-z][a-zA-Z0-9]{3,15}$/.test(username)){  flag = **true**;  }**else**{  ***window***.alert(**"用户名不合法"**);  }  *//如果长度为0* }**else**{  ***window***.alert(**"用户名必填"**);  }  **return** flag;  }   *//针对密码验证的函数* **function** *checkPassword*(password){  **var** flag = **false**;  password = password.trim();  **if**(password.length > 0){  **if**(/^\d{6,20}$/.test(password)){  flag = **true**;  }**else**{  ***window***.alert(**"密码不合法"**);  }  }**else**{  ***window***.alert(**"密码必填"**);  }  **return** flag;  }    *//针对确认密码验证的函数* **function** *checkRePassword*(repassword,password){  **var** flag = **false**;  repassword = repassword.trim();  **if**(repassword.length > 0){  **if**(repassword == password){  flag = **true**;  }**else**{  ***window***.alert(**"二次密码输入不相同"**);  *//清空确认密码框中的内容，即value=""* ***document***.getElementById(**"repwd"**).**value** = **""**;  *//将焦点定位于确认密码框中* ***document***.getElementById(**"repwd"**).focus();  }  }**else**{  ***window***.alert(**"确认密码必填"**);  }  **return** flag;  }   *//总的验证函数* **function** *check*(){  **var** flag = **false**;   *//获取用户名* **var** username = ***document***.getElementById(**"user"**).**value**;  *//获取密码* **var** password = ***document***.getElementById(**"pwd"**).**value**;  *//获取确认密码* **var** repassword = ***document***.getElementById(**"repwd"**).**value**;   *//验证用户名* **if**(*checkUsername*(username)){  **if**(*checkPassword*(password)){  **if**(*checkRePassword*(repassword,password)){  flag=**true**;  }  }  }   **return** flag;  }  </**script**>   </**body**> </**html**> |

# 第02章 BOM编程

## 2.1 BOM编程概述

### 2.1.1 BOM编程的作用

Browser Object Model为了便于对浏览器的操作，JavaScript封装了浏览器中各个对象，使得开发者可以方便的操作浏览器中的各个对象。

|  |
| --- |
|  |

### 2.1.2 BOM常用的对象

|  |  |
| --- | --- |
| **BOM常用对象** | **作用** |
| **window** | **代表浏览器窗口** |
| **location** | **代表浏览器中的地址栏对象** |
| **history** | **代表历史记录对象** |

## 2.2 window对象

### 2.2.1作用

表示整个浏览器窗口或者浏览器中一个选项卡窗口，如：弹出一个警告框。

它包含document、location、history等对象。

|  |
| --- |
|  |

### 2.2.2方法或属性

|  |  |
| --- | --- |
| **对话框有关的方法** | **作用** |
| **alert(“提示信息”)** | **弹出一个确认按钮的信息框** |
| **string prompt(“提示信息”,"默认值”)** | **弹出一个输入信息框，返回字符串类型。**  **注：IE中如果没有默认值，会显示一个undefined** |
| **boolean confirm("提示信息")** | **弹出一个信息框，有确定和取消按钮。如果点确定，返回true，点取消返回false** |

## 2.3 location对象

### 2.3.1作用

表示浏览器中的地址栏，如：跳转到其它网址。

### 2.3.2方法或属性

|  |  |
| --- | --- |
| **属性** | **作用** |
| **获取href属性** | **得到地址栏中访问地址** |
| **设置href属性** | **跳转到一个新的页面** |

## 2.4 history对象

### 2.4.1作用

表示浏览器中的前进和后退按钮，如：页面刷新。

### 2.4.2方法或属性

|  |  |
| --- | --- |
| **方法** | **描述** |
| **go(-1后退/0刷新/1前进)** | **重新加载当前页面，相当于浏览器中的刷新按钮** |

注意这种写法：window.history.go("http://www.baidu.com");

## 2.5 案例：倒计时跳转到另一个页面

注意，这里不用编写HTML页面，只需书写JS代码，

参见：素材/倒计时。

### 2.5.1案例需求

页面上显示一个倒计时5秒的数字，到了5秒以后跳转到另一个页面。

### 2.5.2案例效果

|  |
| --- |
|  |

### 2.5.3案例分析

1\_定义一个变量为5，每过1秒调用1次，refresh()函数。

2\_编写refresh()函数，修改span中的数字。

3\_判断变量是否为1，如果是1则跳转到新的页面。

### 2.5.4案例代码

04\_time.html

|  |
| --- |
| <!DOCTYPE **html** PUBLIC **"-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**> <**html xmlns="http://www.w3.org/1999/xhtml"**>  <**head**>  <**meta http-equiv="Content-Type" content="text/html; charset=utf-8"** />  <**title**>倒计时跳转到另一个页面</**title**>  </**head**>  <**body**>  <**div**><**span id="time"**>10</**span**>秒后跳转到其它页面</**div**>    <**script**>  *//读标签中的内容  //parseInt()将字符串转成number类型   //秒数* **var *time*** = 5;   **function** *refresh*(){  *//如果秒数为1* **if**(***time*** == 1){  *//关闭定时器,参数就是定时器返回的ID值* ***window***.clearInterval(***timeID***);  *//让浏览器跳转发百度首页* ***window***.location.href = **"http://www.baidu.com"**;  }**else**{  *//累减* ***time***--;  *//将time变量的值，再次设置到span标签之间的内容，用innerHTML/innerText* ***document***.getElementById(**"time"**).innerText = ***time***;  }  }   *//定时器，开启定时器* **var *timeID*** = ***window***.setInterval(**"***refresh***()"**,1000);  </**script**>  </**body**> </**html**> |

# 第03章 DOM编程

## 3.1 DOM编程概述

### 3.1.1 DOM编程的基本概念

文档对象模型（Document Object Model，简称DOM），可以让用户对网页中的元素进行操作。这里的元素等价于标签，如<div>标签，<span>标签。

### 3.1.2 DOM编程的作用

每个HTML页面在被浏览器解析的时候都会在内存中创建一棵倒状DOM树，我们通过编写JS代码就可以访问这棵树上任何一个节点，并且对节点进行操作。通过 DOM模型，可以访问所有的 HTML 元素，连同它们所包含的文本和属性。可以对其中的内容进行修改和删除，同时也可以创建新的元素。新创建的元素对象，要挂到DOM树上才可以在网页上显示出来。

|  |
| --- |
|  |

## 3.2查找节点的常用方法

|  |  |
| --- | --- |
| **获取元素的方法** | **作用** |
| **document.getElementById("id")** | **通过id属性找到唯一的元素**  **如果页面上有多个同名的id，则得到第1个元素** |
| **document.getElementsByName("name")** | **通过name属性得到一组标签，返回一个数组** |
| **document.getElementsByTagName ("标签名")** | **通过标签名字得到一组标签，返回一个数组** |
| **document.getElementsByClassName("类名")** | **通过类名得到一组标签，返回一个数组，class=”a”** |

## 3.3案例：商品结算的功能

注意，这里不用编写HTML页面，只需书写JS代码，

参见：素材/商品结算。

### 3.3.1案例需求

页面上有5件商品，前面都有复选框，名字叫item，value是商品的价格。下面有一个“全选/全不选”的复选框，id是"all"，点它实现全选或全不选的功能，还有个反选的按钮，点它实现反选的功能。下面有一个按钮，点它则计算选中商品的总金额。

### 3.3.2案例效果

|  |
| --- |
|  |

### 3.3.3案例分析

1\_复选框如果要选中，设置checked=true，取消设置checked=false。

2\_全选：通过name属性得到上面所有的复选框对象，遍历集合，将每一个元素的checked设置为true。

3\_全不选：将所有元素的checked属性设置为false。

4\_反选：原来选中的设置false，原来没选的设置为true。

5\_结账：将所有选中的元素的value转成数字，再累加，将累加的结果显示在后面的span中。

### 3.3.4案例代码

05\_goods.html

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>实现全选/全不选，商品结算的功能</**title**> </**head**> <**body**>  <**h3**>商品价格列表</**h3**>   <**input type="checkbox" name="item" value="1500"** /> 山地自行车1500<**br** />  <**input type="checkbox" name="item" value="200"** /> 时尚女装200<**br** />  <**input type="checkbox" name="item" value="3000"** /> 笔记本电脑3000元<**br** />  <**input type="checkbox" name="item" value="800"** /> 情侣手表800<**br** />  <**input type="checkbox" name="item" value="2000"** /> 桑塔纳2000<**br** />  <**input type="checkbox" name="item" value="2000"** /> 桑塔纳2000<**br** />   <**hr**/>   <**input type="checkbox" id="all" onclick="***selectAll*()**"** />全选/全不选  <**input type="button" id="reverse" onclick="***reverseSelect*()**" value="反选"**/>  <**input type="button" value=" 结账 " onclick="***total*()**"** />**&nbsp;&nbsp;&nbsp;** <**span id="result"**></**span**>   <**script**>  *//全选/全不选* **function** *selectAll*(){  *//定位全选/全不选复选框* **var** all = ***document***.getElementById(**"all"**);  *//如要选中了* **if**(all.checked){  *//定位所有name=item的复选框选* **var** items = ***document***.getElementsByName(**"item"**);  *//将上面所有商品复选框选中* **for**(**var** i=0;i<items.length;i++){  *//将每一个商品的复选框选中* items[i].checked = **true**;  }  }**else**{  *//定位所有name=item的复选框选* **var** items = ***document***.getElementsByName(**"item"**);  *//将上面所有商品复选框选中* **for**(**var** i=0;i<items.length;i++){  *//将每一个商品的复选框选中* items[i].checked = **false**;  }  }  }   *//反选* **function** *reverseSelect*(){  **var** items = ***document***.getElementsByName(**"item"**);  **for**(**var** i=0;i<items.length;i++){  items[i].checked = ! items[i].checked;  }  }   *//结账* **function** *total*(){  **var** sum = 0;  **var** items = ***document***.getElementsByName(**"item"**);  *//迭代* **for** (**var** i **in** items) {  *//如果某个商品选中了* **if**(items[i].checked){  *//取出该复选框的value属性值，是字符串类型* **var** strPrice = items[i].**value**;  *//将string转成number* **var** price = *parseInt*(strPrice);  *//累加* sum += price;  }  }  *//将sum变量的值，设置到span标签中，添加￥符号* ***document***.getElementById(**"result"**).innerHTML = **"<font color='red'>￥"** + sum+**"</font>"**;  }  </**script**>  </**body**> </**html**> |

## 3.4增删改节点的常用方法

在DOM树上创建元素分2步：

1\_创建这个元素。

2\_把元素挂到DOM树上。

### 3.4.1创建和修改元素的方法

|  |  |
| --- | --- |
| **创建元素的方法** | **作用** |
| **document.createElement("input")** | **在文档上创建一个元素对象** |
| **元素对象.setAttribute("type", "password")** | **给元素添加一个属性名和属性值**  **如果属性名不存在则是添加属性**  **如果属性名存在则是修改属性值** |
| **document.createTextNode("文本内容")** | **在文档上创建一个文本节点**  **可以使用简写形式innerHTML属性。** |

### 3.4.2挂到DOM树的方法

|  |  |
| --- | --- |
| **将元素挂到DOM树上的方法** | **作用** |
| **父元素.appendChild(子元素)** | **将元素追加成父元素的最后一个子元素** |
| **父元素.removeChild(子元素)** | **通过父元素删除一个子元素** |
| **元素.remove()** | **元素删除本身** |

### 3.4.3通过关系找节点的属性

|  |  |
| --- | --- |
| **遍历节点的属性** | **说明** |
| **childNodes** | **得到当前元素下所有的子节点** |
| **firstChild** | **得到当前元素的第一个子节点** |
| **lastChild** | **得到当前元素的最后一个子节点** |
| **parentNode** | **得到当前元素的父节点** |
| **nextSibling** | **得到当前元素的下一个兄弟节点** |
| **previousSibling** | **得到当前元素的上一个兄弟节点** |

## 3.5案例：学生信息管理

注意，这里不用编写HTML页面，只需书写CSS和JS代码，

参见：素材/学生管理系统。

### 3.5.1案例需求

1\_当鼠标移入时，该行的背景颜色为黄色，当鼠标移出时，该行的背景颜色还原。

2\_当点击添加按钮“添加一行数据”时，文本框中的数据添加到表格中且文本框清空。

3\_当点击表格中的“删除”时，该行数据被删除，删除前确认。

### 3.5.2案例效果

|  |
| --- |
|  |

### 3.5.3案例分析

1\_使用tr:hover{}选择器，让行背景变色。

2\_添加的实现：当点击按钮时，得到文本框中的文本，之后将文本框的值清空。

3\_创建tr，用innerHTML属性设置tr中的内容，最后将tr追加到tbody元素中，

注：tbody无论源代码中是否写了，在某些浏览器中始终存在。

4\_删除操作：当点击删除超链接时，用remove()方法从tbody中删除tr自己。

### 3.5.4案例代码

06\_students.html

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>学生信息管理</**title**>  <**style**>  **tbody tr**:**hover**{  **background-color** : **yellow**;  }  </**style**> </**head**> <**body**>  <**div**>  <**table border="1" cellspacing="0" cellpadding="3"**>  <**thead**>  <**tr**>  <**th**>学号</**th**>  <**th**>姓名</**th**>  <**th**>操作</**th**>  </**tr**>  </**thead**>  <**tbody id="data"**>  <**tr**>  <**td**>00001</**td**>  <**td**>潘金莲</**td**>  <**td**>  *<!--href=""-->  <!--href="#"-->  <!--href="javascript:;"-->  <!--href="javascript:void(0)"-->  <!--对a标签添加onclick事件-->  <!--this表示当前a标签-->* <**a href="javascript:void(0)" onclick="***deleteRow*(**this**)**"**>删除</**a**>  </**td**>  </**tr**>  <**tr**>  <**td**>00002</**td**>  <**td**>鲁智深</**td**>  <**td**>  <**a href="javascript:;" onclick="***deleteRow*(**this**)**"**>删除</**a**>  </**td**>  </**tr**>  </**tbody**>  </**table**>  <**br** />  学号：<**input type="text" id="no" value=""** />  姓名：<**input type="text" id="name" value=""** />  <**input type="button" id="addBtn" value="添加" onclick="***addRow*()**"**/>  </**div**>    <**script**>   *//添加一行* **function** *addRow*(){  *//获取学号和姓名* **var** no = ***document***.getElementById(**"no"**).**value**;  **var** name = ***document***.getElementById(**"name"**).**value**;   *//清空原来学号和姓名* ***document***.getElementById(**"no"**).**value** = **""**;  ***document***.getElementById(**"name"**).**value** = **""**;   *//创建tr标签/元素* **var** tr = ***document***.createElement(**"tr"**);  *//设置tr标签中的内容* tr.innerHTML = **"<td>"**+no+**"</td>"** +  **"<td>"**+name+**"</td>"** +  **"<td><a href='javascript:void(0);' onclick='***deleteRow***(this)'>删除</a></td>"**;   *//将新标签tr添加到tbody标签中，形式父子关系* ***document***.getElementById(**"data"**).appendChild(tr);  }   *//删除一行* **function** *deleteRow*(a){  *//提示* **var** flag = ***window***.confirm(**"你确认要删除本行吗？"**);  *//如果确认的话* **if**(flag){  *//获取a标签的父标签的父标签* **var** tr = a.parentNode.parentNode;  *//删除本身标签* tr.remove();  }  }  </**script**> </**body**> </**html**> |

# 第04章 操作CSS样式

## 4.1在JS中操作CSS属性命名上的区别

|  |  |  |
| --- | --- | --- |
| **CSS中写法** | **JS中的写法** | **说明** |
| **color** | **color** | **如果一个单词写法一样** |
| **font-size** | **fontSize** | **如果多个单词，使用驼峰命名法** |

### 4.1.1方式一

|  |
| --- |
| **元素.style.样式名 = "样式值";**  **每条语句只能修改一个样式** |

### 4.1.2方式二

|  |
| --- |
| **元素.className = "样式类名";**  **每条语句修改一批样式** |

## 4.2案例：点按钮，修改p标签的字体、颜色、大小

注意，这里不用编写HTML页面，只需书写CSS和JS代码，

参见：素材/改变样式。

### 4.2.1案例需求

1\_点击“改变几个样式”按钮，用方式一改变文本的字体，颜色，大小。

2\_点击“改变类样式”按钮，用方式二改变文本的字体，颜色，大小。

颜色：color

字体：font-size

字形：font-family

### 4.2.2案例效果

|  |
| --- |
|  |

### 4.2.3案例分析

1\_通过ID选择器定位p标签。

2\_用p.style一个个操作样式。

3\_用p.className一次性操作样式。

### 4.2.4案例代码

07\_changeStyle.html

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>改变样式</**title**>   <**style**>  */\*类样式的定义\*/* .**secondPStyle**{  **color**:**green**;  **font-size**: 66**px**;  **font-style**: **oblique**;  }  </**style**>  </**head**> <**body**>  <**p id="first"**>这是一个自然段</**p**>  <**p id="second"**>这是二个自然段</**p**>   <**input type="button" value="改变几个样式" onclick="***changeCss*()**"**/>  **&nbsp;&nbsp;&nbsp;&nbsp;** <**input type="button" value="改变类样式" onclick="***changeClass*()**"**/>    <**script**>  *//用JS来修改标签的样式* **function** *changeCss*(){  *//定位p标签* **var** pFirst = ***document***.getElementById(**"first"**);  *//修改color* pFirst.style.color = **"red"**;  *//修改fontSize* pFirst.style.fontSize = **"44px"**;  *//修改* pFirst.style.fontStyle = **"italic"**;  }   **function** *changeClass*(){  **var** pSecond = ***document***.getElementById(**"second"**);  pSecond.className = **"secondPStyle"**;  }  </**script**> </**body**> </**html**> |

## 4.3案例：使用JS修改表格行的背景色

注意，这里不用编写HTML页面，只需书写CSS和JS代码，

参见：素材/表格变色。

### 4.3.1案例需求

使用JS修改表格行的背景色，产生隔行变色的效果，鼠标移上去的时候这一行变成其它颜色，移出去的时候还原成之前的背景色。

### 4.3.2案例效果

|  |
| --- |
|  |

### 4.3.3案例分析

1\_创建三个类样式redStyle、yellowStyle、greenStyle，分别用于设置背景色为浅红lightpink，浅黄lightyellow，浅绿lightgreen。

2\_设置tr，td中的文本居中对齐。

3\_通过标签名得到所有的tr行。

4\_如果是偶数，则设置它的背景色为浅黄色，否则为浅红色。

5\_设置一个全局变量记录之前的类样式名。设置鼠标在上面和在外面的事件，鼠标在上面的事件中，记录没有换颜色之前的颜色放在全局变量中，再设置类样式为浅绿色。

如果鼠标移出之后，要回到原来的颜色，将全局变量记录的样式名赋值给当前行的类样式名。

### 4.3.4案例代码

08\_tableChange.html

|  |
| --- |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"**>  <**title**>表格变色</**title**>  <**style**>  **td**{  **text-align**: **center**;  }  .**redStyle**{  **background-color**:**lightpink**;  }  .**yellowStyle**{  **background-color**:**orange**;  }  .**greenStyle**{  **background-color**:**red**;  }  </**style**> </**head**> <**body**>  <**table id="tab1" border="1" width="800" align="center"**>  <**tr style="background-color**: **#ccc**;**"**>  <**th**>分类ID</**th**>  <**th**>分类名称</**th**>  <**th**>分类描述</**th**>  <**th**>操作</**th**>  </**tr**>  <**tr**>  <**td**>1</**td**>  <**td**>手机数码</**td**>  <**td**>手机数码类商品</**td**>  <**td**><**a href=""**>修改</**a**>|<**a href=""**>删除</**a**></**td**>  </**tr**>  <**tr**>  <**td**>2</**td**>  <**td**>电脑办公</**td**>  <**td**>电脑办公类商品</**td**>  <**td**><**a href=""**>修改</**a**>|<**a href=""**>删除</**a**></**td**>  </**tr**>  <**tr**>  <**td**>3</**td**>  <**td**>鞋靴箱包</**td**>  <**td**>鞋靴箱包类商品</**td**>  <**td**><**a href=""**>修改</**a**>|<**a href=""**>删除</**a**></**td**>  </**tr**>  <**tr**>  <**td**>4</**td**>  <**td**>家居饰品</**td**>  <**td**>家居饰品类商品</**td**>  <**td**><**a href=""**>修改</**a**>|<**a href=""**>删除</**a**></**td**>  </**tr**>  </**table**>    <**script**>  **var *globalTrClassName*** = **null**;   *//获取所有tr标签* **var *trs*** = ***document***.getElementsByTagName(**"tr"**);  *//除第1行外* **for**(**var *i***=1;***i***<***trs***.length;***i***++){   *//如果是偶数行的话* **if**(***i*** % 2 == 0){  *//修改tr标签的样式名* ***trs***[***i***].className = **"redStyle"**;  *//如果是奇数行的话* }**else**{  *//修改tr标签的样式名* ***trs***[***i***].className = **"yellowStyle"**;  }  *//为每一个tr标签添加鼠标移入事件* ***trs***[***i***].*onmouseover* = **function**(){  *//获取原tr标签的样式，保存起来  //this表示当前tr标签* ***globalTrClassName*** = **this**.className;  *//修改tr标签的样式* **this**.**className** = **"greenStyle"**;  };   *//为每一个tr标签添加鼠标移出事件* ***trs***[***i***].*onmouseout* = **function**(){  *//将原来的样式设置给当前tr的样式* **this**.**className** = ***globalTrClassName***;  };   }*//for结束* </**script**> </**body**> </**html**> |

# 随堂测试