《JavaWeb课程》教案

第3章 js

[1. 回顾上一章: [10分钟] 3](#_Toc489947987)

[2. 课程知识点讲解： 3](#_Toc489947988)

[2.1. 知识点1：javascript简介[70分钟] 4](#_Toc489947989)

[2.2. 知识点2：核心语法[40分钟] 6](#_Toc489947990)

2.3 知识点3：DOM[80分钟]...........................................................................................................................................9

2.4 知识点4：BOM[60分钟].........................................................................................................................................12

2.5 知识点5：正则表达式 表单验证[80分钟]............................................................................................................15

[3. 本章总结 [20分钟] 2](#_Toc489947993)0

[4. 测试题 [30分钟] 2](#_Toc489947994)0

[5. 作业 [30分钟] 2](#_Toc489947995)0

[6. 扩展部分：](#_Toc489947996) 20

[7. 学员问题汇总： 2](#_Toc489947997)1

编创教师:

课时：420分钟

本章主要目标

掌握js简介： 定义、js特点、js组成(ECMAscript DOM BOM)、引入方式、执行原理

熟练运用js的语法：数据类型、变量、内置对象Math,Date,Array,String、运算符、逻辑控制语句（简单介绍，java中讲过）、注释、函数、系统函数（parseInt(),parseFloat(),isNaN()）、自定义函数、事件

DOM：定义、DOM与网页的关系、DOM的常用获取节点的方法（做表格）

掌握js的ＤＯＭ操作及常用事件：

onload事件

onclick事件

onmouseover和onmouseout事件

onfocus、onblur和onsubmit事件

BOM：BOM定义（强调与DOM的区别）、BOM的应用场景、window对象的方法

表单验证：

onsubmit

onfocus,onblur

正则

本章重点

重点1：变量声明，函数，事件和定时器的使用

重点2：运用正则表达式验证表单

本章难点

难点1：函数的理解

难点2：事件的运用

本章授课注意事项

1、本章采用由浅入深，逐层深入的的授课思路，教学过程中注意控制进度，避免出现前一阶段没有跟上，后一阶段只能放弃学习的情况

2、本章知识点讲解中主要使用先讲解理论再结合代码，并与之前学过的java做对比，帮助学员理解js语法和对js的运用。

# 回顾上一章: [10分钟]

作业点评和上章主要内容回顾

回顾css样式,提问：鼠标悬停时如何改变样式。几种定位方式？分别是什么？不同定位的区别有？

# 课程知识点讲解：

知识点1：[70分钟]

知识点1文字说明

Javascript简介：定义、javascript特点、javascript组成、基本结构(script标签)、javascript执行原理、 在网页中引入javascript的3种方式。

2.1.1定义

导入（演示网页中用到js的例子，这些是只用html和css无法实现的）

js定义：**基于对象**和**事件驱动**的**嵌入式**脚本语言。

基于对象： 使用对象的属性和方法。有js内置对象，也可以自定义对象。

事件驱动： 例如：当用户点击的时候，执行函数。点击就是触发了事件

嵌入式： 不能单独执行，需要嵌入到html中使用。

2.1.2javascript特点

JS特点（语法和java类似）：JavaScript一种直译式脚本语言，是一种动态类型、弱类型（java强类型）、基于原型（java基于类）的语言。它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言。js是一种**解释性**脚本语言（代码不进行预编译）

2.1.3javascript组成

**ECMAScript**，描述了该语言的语法和基本对象。

**文档对象模型（DOM）**，描述处理网页内容的方法和接口。

**浏览器对象模型（BOM）**，描述与浏览器进行交互的方法和接口。

2.1.4基本结构

1、script标签可以在的位置? 放在Html的任何位置都可以

2、script标签的数量? 数量可以有多个，没有限制。但最好使用一个，一般放在页面底部。

1. 为什么使用注释? JS注释： 单行注释// 多行注释/\* \*/
2. script标签中的type属性。type 属性规定脚本的 MIME 类型。对于 JavaScript，其 MIME 类型是 "text/javascript"。默认的是javascript，一般省略不写。

案例：输出‘Hello， world！’

2.1.5 javascript执行原理：在页面中引入js后，打开浏览器进行查看调试。

2.1.6在网页中引入javascript的3种方式：

1、 <button onclick="alert(123);">点击</button>

2、 在页面中加入<script>标签

3、 从外部引入js文件 <script src=’js/abc.js’></script>



**案例描述**：

任务描述

**案例代码**：（源文件名称）

1、先感受一下js代码：

<script>

document.write("<h1>这是一个标题</h1>");

</script>

2、网页中引入javascript的3种方式：

1)在标签内添加： <button onclick="alert(123);">点击</button>

2)在页面中加入标签：

<script>

// js代码

</script>

3)在页面中引入js文件，首先创建js文件，例如：abc.js。在js文件内部写js代码。在html中加入script标签，使用src属性，引入外部js文件。

<script src=’js/abc.js’></script>

**案例说明**：

1）案例说明：简单的代码体验一下。

2）案例说明：主要说明js的几种引入方式。



提问或设问

1）问题1：js的组成？

2）问题2: 什么叫事件驱动



1）关于本知识点的注意事项1：理解js的定义

2）关于本知识点的注意事项2：区分js的引入方式



练习1：

参考代码：使用document.write在页面中输出内容。并使用3种js的引入方式来实现。

知识点2：[40分钟]

知识点2 文字说明：

2.2.1数据类型

2.2.2变量（弱数据类型，typeof）

2.2.3内置对象Math,Date,Array,String

2.2.3运算符

2.2.4逻辑控制语句（简单介绍，java中讲过）

2.2.5注释

练习（多个）

2.2.6调试

2.2.7函数

定义：

系统函数

parseInt(),parseFloat(),isNaN()

自定义函数

函数定义 函数调用

匿名函数（定义，调用，用于回调）

2.2.8事件

onclick

onload

**数据类型：** 5 种简单数据类型（也称为基本数据类型）：Undefined、Null、Boolean、Number和 String。

还有 1种复杂数据类型——Object，Object 本质上是由一组无序的名值对组成的。

**变量：**使用关键字 var 。可以把变量看做存储数据的容器。

**ECMAScript 定义变量函数的语法**：

1、变量名 函数名 避免关键字和保留字

2、ECMAScript 中的一切（变量、函数名和操作符）都区分大小写。

变量名 test 和变量名 Test 分别表示两个不同的变量

3、标识符，就是指变量、函数、属性的名字，或者函数的参数。

标识符可以是按照下列格式规则组合起来的一或多个字符：

 **第一个字符**必须是一个字母、下划线（\_）或一个美元符号（$）；

 其他字符可以是字母、下划线、美元符号或数字。

标识符中的字母也可以包含扩展的 ASCII 或 Unicode 字母字符（如 À 和 Æ），但我们不推荐这样做。

按照惯例，ECMAScript 标识符采用**驼峰大小写格式**，也就是第一个字母小写，剩下的每个单词的首字母大写，

例如：firstSecond myCar doSomethingImportant

关键字：用于表示控制语句的开始或结束，或者用于执行特定操作等。按照规则，关键字也是语言保留的，不能用作标识符。

保留字： 尽管保留字在js语言中还没有任何特定的用途。但它们有可能在将来被用作关键字。

ECMAScript的关键字:

|  |  |  |  |
| --- | --- | --- | --- |
| break | do | instanceof | typeof |
| case | else | new | var |
| catch | finally | return | void |
| continue | for | switch | while |
| debugger | function | this | with |
| default | if | throw | delete |
| in | try |  |  |

ECMAScript的保留字:

|  |  |  |  |
| --- | --- | --- | --- |
| abstract | enum | int | short |
| boolean | export | interface | static |
| byte | extends | long | super |
| char | final | native | synchronized |
| class | float | package | throws |
| const | goto | private | transient |
| debugger | implements | protected | volatile |
| double | import | public |  |



**案例描述**：

任务描述

**案例代码**：（源文件名称）

1、typeof 操作符：

对一个值使用 typeof 操作符可能返回下列某个字符串：

 "undefined"——如果这个值未定义；

 "boolean"——如果这个值是布尔值；

 "string"——如果这个值是字符串；

 "number"——如果这个值是数值；

 "object"——如果这个值是对象或 null；

"function"——如果这个值是函数。

2、ECMAScript 中的语句以一个分号结尾;

var num; // 声明变量

console.log(num); // undefined未定义

var num = 123; // 声明变量并赋值

console.log(num); // 123

3、定义不同的变量，在控制台打印。来理解不同类型的值

var a = true;

var b = "123";

var c = 123;

var d = null;

var e;

var f = function () {

}

console.log(typeof(a));

console.log(typeof(b));

console.log(typeof(c));

console.log(typeof(d));

console.log(typeof(e));

console.log(typeof(f));

console.log(typeof(g));

1. 内置对象：

**数组**：

var arr = [1,’2’,3,’a’];

console.log(typeof(arr));

console.log(arr.length); // 数组长度

创建一个数组

创建一个数组，有三种方法。

1: 常规方式:

var week = new Array();

week [0]="Monday";

week [1]="Tuesday";

week [2]="Wednesday";

2: 简洁方式:

var week =new Array("Monday","Tuesday","Wednesday");

3: 字面:

var week =["Monday","Tuesday","Wednesday"];

数组的属性和方法：

var x=week.length // week中元素的数量

删除数组的最后一个元素 - pop()

数组的末尾添加新的元素 - push()

删除数组的第一个元素 - shift()

在数组的开头添加新元素 - unshift()

join() 把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。

reverse() 颠倒数组中元素的顺序。

**字符串：**

**var str = ‘123’；**

**console.log(typeof （str）); // string**

indexOf() 方法

使用 indexOf() 来定位字符串中某一个指定的字符首次出现的位置。没有则返回-1

var str="Hello world!"

document.write(str.indexOf("Hello")) // 0

document.write(str.indexOf("World")) // -1

document.write(str.indexOf("world")) //6

match() 方法

使用 match() 来查找字符串中特定的字符，并且如果找到的话，则返回这个字符。

var str="Hello world!"

document.write(str.match("world") + "<br />") // world

document.write(str.match("World") + "<br />") // null

document.write(str.match("worlld") + "<br />") //null

document.write(str.match("world!")) //world!

//split() – 通过将字符串划分成子串，将一个字符串做成一个字符串数组。

var str03 = "aa.bb.cc"

document.write(str03.split("."));

console.log(str03.split(".")); //使用打印看数组结构

**日期**：

通过new关键字来定义日期对象。

注释：Date 对象自动使用当前的日期和时间作为其初始值。

var myDate=new Date()

**运算符**：

算术运算符用于执行变量与/或值之间的算术运算。

给定 y=5，下面的表格解释了这些算术运算符：



赋值运算符用于给 JavaScript 变量赋值。

给定 x=10 和 y=5，下面的表格解释了赋值运算符：



**逻辑控制语句**：

if (条件 1)

{

当条件 1 为 true 时执行的代码

}

else if (条件 2)

{

当条件 2 为 true 时执行的代码

}

else

{

当条件 1 和 条件 2 都不为 true 时执行的代码

}

switch 语句：选择要执行的多个代码块之一

var day=new Date().getDay();

switch (day)

{

case 0:

x="Today it's Sunday";

break;

case 1:

x="Today it's Monday";

break;

case 2:

x="Today it's Tuesday";

break;

case 3:

x="Today it's Wednesday";

break;

case 4:

x="Today it's Thursday";

break;

case 5:

x="Today it's Friday";

break;

case 6:

x="Today it's Saturday";

break;

}

while 循环：while 循环会在指定条件为真时循环执行代码块。

while (条件)

{

需要执行的代码

}

while (i<5)

{

x=x + "The number is " + i + "<br>";

i++;

}

do/while 循环会至少执行一次，即使条件是 false。

do/while 循环会执行一次代码块，在检查条件是否为真之前，然后如果条件为真的话，就会重复这个循环。

语法:

do

{

需要执行的代码

}

while (条件);

do

{

x=x + "The number is " + i + "<br>";

i++;

}

while (i<5);

for循环：

var cars=["BMW","Volvo","Saab","Ford"];

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

console.log(cars[i]);

}

打开控制台，调试代码。

**函数**：

全局函数

**parseInt():**parseInt() 函数可解析一个字符串，并返回一个整数。

<script>

document.write(parseInt("10") + "<br>");

document.write(parseInt("10.33") + "<br>");

document.write(parseInt("34 45 66") + "<br>");

document.write(parseInt(" 60 ") + "<br>");

document.write(parseInt("40 years") + "<br>");

document.write(parseInt("He was 40") + "<br>");

</script>

**parseFloat()：**parseFloat() 函数可解析一个字符串，并返回一个浮点数。

注意： 字符串中只返回第一个数字。开头和结尾的空格是允许的。如果字符串的第一个字符不能被转换为数字，那么 parseFloat() 会返回 NaN非数字。

使用 parseFloat() 来解析不同的字符串：

<script>

document.write(parseFloat("10") + "<br>");

document.write(parseFloat("10.33") + "<br>");

document.write(parseFloat("34 45 66") + "<br>");

document.write(parseFloat(" 60 ") + "<br>");

document.write(parseFloat("40 years") + "<br>");

document.write(parseFloat("He was 40") + "<br>");

</script>

**isNaN()：**

isNaN() 函数用于检查其参数是否是非数字值。

如果参数值为 NaN 或字符串、对象、undefined等非数字值则返回 true, 否则返回 false。

<script>

document.write(isNaN(123)+ "<br>");

document.write(isNaN(-1.23)+ "<br>");

document.write(isNaN(5-2)+ "<br>");

document.write(isNaN(0)+ "<br>");

document.write(isNaN("Hello")+ "<br>");

document.write(isNaN(NaN)+ "<br>");

</script>

**自定义函数**：

1、// 声明函数

function abc() {

alert('123');

}

// 函数调用 abc();

2、**function play (time) {**

**alert(time + '点了,去玩耍了！');**

**}**

**function eat (time) {**

**alert(time + '点了,吃早饭了！');**

**}**

**function study (time) {**

**alert(time + '点了,好好学习！');**

**}**

**eat(8); // 函数调用**

**study (10);**

**play (19);**

**3、function myFunction(){**

**var x=5;**

**return x;**

**}**

**var myVar=myFunction();**

**console.log(myVar);**

**事件：**

函数是由事件驱动的或者当它被调用时执行的可重复使用的代码块。

onclick点击事件：

**<button onclick="message()">点击这里</button>**

onload事件：

**<body onload="SomeJavaScriptCode">**

**onload 事件会在页面或图像加载完成后立即发生。**

**onload 通常用于 <body> 元素，在页面完全载入后(包括图片、css文件等等。)执行脚本代码。**

给img加onload事件：

**<img src="logo.png" onload="loadImage()" />**

**综合onload事件:**

**<body>**

**<img id="myImg" width="1000" src="http://img2.niutuku.com/desk/1208/1401/ntk-1401-8806.jpg" />**

**</body>**

**<script>**

**window.onload = function(){**

**alert("页面dom加载完成");**

**}**

**document.getElementById("myImg").onload = function(){**

**alert("图片已经加载完成");**

**}**

**</script>**

**案例说明**：

1）案例说明：匿名函数和有名函数的区别。以及函数的调用。

2）案例说明：函数调用时，传入参数。time是参数，形参。在用调用方法时，传入实参。



1. 问题1：定义函数的关键字？

2）问题2：关于函数的参数，形参和实参的区别？



1）关于本知识点的注意事项1： 注意区分形参和实参

2）关于本知识点的注意事项2： 事件和函数的概念要分清楚



练习1： 练习声明变量，将所有的数据类型的都练习，并在控制台输出。

练习2： 练习/2.2数组方法/彩色条纹，练习/2.2数组方法/切换彩色条纹

练习3： 练习/2.2字符串方法/反向排列输入的值

练习4：

参考代码：结合if判断

var time = 20;

if (7<time && time<9) {

eat(time);

}else if (9<=time && time<=18) {

study (time);

}else if (18< time < 22) {

play (time);

}

function eat (time) {

console.log(time + '点了,吃早饭了！');

}

function study (time) {

console.log(time + '点了,好好学习！');

}

function play (time) {

console.log(time + '点了,去玩耍了！');

}

练习5：练习/2.2 TAB切换效果

知识点1：[80分钟]

当网页被加载时，浏览器会创建页面的文档对象模型（Document Object Model）。

HTML DOM 模型被构造为对象的树：



知识点1文字说明:

DOM定义：

定义了访问和操作 HTML 文档的标准方法。

DOM就是是关于如何获取、修改、添加或删除文档 元素的标准。

DOM与网页的关系： html文档中的元素都是一个个节点。JS可以获取修改、添加或删除DOM 节点，来改变页面结构。

常用获取DOM 的方法：

|  |  |  |
| --- | --- | --- |
| 方法 | 区别 | 获取到的元素 |
| getElementById(‘’) | 通过id名获取 | 单个元素 |
| document.getElementsByTagName(‘’) | 通过标签名获取 | 一组元素 |
| document.getElementsByClassName(‘’) | 通过class名获取 | 一组元素 |



**案例描述**：

任务描述

**案例代码**：（源文件名称）案例/2.3实现全选及全不选效果、点击导航下拉菜单.html、鼠标移入移出导航下拉菜单.html

获取DOM： document.getElementById("myBtn")

document.getElementByTagName("p")

document.getElementByClassName("btn")

1、// 可以在标签上添加事件

**<button onclick="message()">点击这里</button>**

**2、 通过document.getElementById("myBtn")获取到DOM元素**

**<body>**

**<button id=“myBtn”></button>**

**<div id=“box”></div>**

**</body>**

**<script>**

**document.getElementById("myBtn").onclick=function(){message()};**

**function message(){**

**document.getElementById("box").innerHTML=‘点击事件’;**

**}**

**</script>**

**鼠标移入、移出事件：**

**onmouseover和onmouseout事件**

**// this代表当前对象。这里就是div**

**<div onmouseover="mOver(this)" onmouseout="mOut(this)" style="background-color:#D94A38;width:120px;height:20px;padding:40px;">**

**Mouse Over Me**

**</div>**

**<script>**

**function mOver(obj){**

**obj.innerHTML="Thank You"**

**}**

**function mOut(obj){**

**obj.innerHTML="Mouse Over Me"**

**}**

**</script>**

**案例说明**：

1）案例说明：获取页面中的DOM

2）案例说明：列举常见事件的用法



1）问题1：获取页面中的DOM

2）问题2：学习的2种触发事件的方式？

****

1）关于本知识点的注意事项1：js要写在标签之后，否则获取不到DOM。

2）关于本知识点的注意事项2：练习理解常见的事件，以及如何触发事件。



练习1：

参考代码：见 : 练习/2.3 DOM事件

练习2：

参考代码：练习/2.3制作树形菜单

知识点1：[60分钟]

2.4.1 知识点4内的第1小知识点

知识点1文字说明:

2.4.1BOM定义：浏览器对象模型 (BOM) 是 JavaScript 与 浏览器 ‘交流’的桥梁。它提供了很多对象,用于访问浏览器的功能;这些功能与任何网页内容无关。

2.4.2BOM的应用场景：网页的前进后退，查看浏览历史记录，查看浏览器信息等。

2.4.3window对象的方法：

alert(),confirm(),prompt();

网页的打开关闭：

open(),close();

定时器：setTimeout(),setInterval()

2.4.4 history对象和location对象

历史记录history：

history.back() - 与在浏览器点击后退按钮相同

history.forward() - 与在浏览器中点击按钮向前相同

Location：用于获得当前页面的地址 (URL)，并把浏览器重定向到新的页面。

location.hostname 返回 web 主机的域名

location.pathname 返回当前页面的路径和文件名

location.port 返回 web 主机的端口 （80 或 443）

location.protocol 返回所使用的 web 协议（http: 或 https:）



**案例描述**：

任务描述

**案例代码**：（源文件名称）案例/2.4BOM

**open（）close（）**：

<input type="button" value="新窗口的新文档" onclick="createNew()">

<script>

function createNew(){

var w=window.open();

w.document.open();

w.document.write("<h1>Hello World!</h1>");

w.document.close();

}

</script>

**location:**

location.hostname 返回 web 主机的域名

location.pathname 返回当前页面的路径和文件名

location.port 返回 web 主机的端口 （80 或 443）

location.protocol 返回所使用的 web 协议（http: 或 [https:）](https://）)

<script>

document.write(location.href);

</script>

<script>

function newDoc()

{

window.location.assign("http://www.baidu.com")； // 加载新文档

}

</script>

**history:**

history.back() - 与在浏览器点击后退按钮相同,加载历史列表中的前一个 URL。

history.forward() - 与在浏览器中点击按钮向前相同,加载历史列表中的下一个 URL。

<script>

function goBack(){

window.history.back()

}

</script>

<script>

function goForward(){

window.history.forward()

}

</script>

**开启setInterval：**

<p id="demo"></p>

<script>

var myVar=setInterval(function(){myTimer()},1000);

function myTimer(){

var d=new Date();

var t=d.toLocaleString();

document.getElementById("demo").innerHTML=t;

}

</script>

**停止setInterval：**

<p id="demo"></p>

<button onclick="myStopFunction()">停止时钟</button>

<script>

var myVar=setInterval(function(){myTimer()},1000);

function myTimer(){

var d=new Date();

var t=d.toLocaleTimeString();

document.getElementById("demo").innerHTML=t;

}

function myStopFunction(){

clearInterval(myVar);

}

</script>

**开启setTimeout()：**

3秒后弹出‘Hello’

<button onclick="myFunction()">点击</button>

<script>

function myFunction(){

setTimeout(function(){alert("Hello")},3000);

}

</script>

**停止setTimeout()：**

<p>点击第二个按钮来阻止第一个函数运行。（你必须在3秒之前点击它）。</p>

<button onclick="myFunction()">点我</button>

<button onclick="myStopFunction()">停止弹框</button>

<script>

var myVar;

function myFunction(){

myVar=setTimeout(function(){alert("Hello")},3000);

}

function myStopFunction(){

clearTimeout(myVar);

}

</script>

**案例说明**：

1）案例说明： window对象的属性

2）案例说明： window对象的方法。



1）问题1：window对象的属性有？

2）问题2：window对象的方法可以如何简写？

****

1. 关于本知识点的注意事项1： window对象的几个属性和方法，可以省略 window. ,直接使用。

2）关于本知识点的注意事项2：两种定时器的区别。



练习1：

参考代码：见 练习/2.4定时器-图片切换

知识点1：[80分钟]

2.5.1 知识点5内的第1小知识点

知识点1文字说明:表单事件、正则表达式。

2.5.1 onsubmit：表单提交事件

2.5.2 onfocus,onblur：表单获取焦点事件、失去焦点事件

2.5.3正则 **正则表达式**：正则表达式描述了字符的模式对象。当您检索某个文本时，可以使用一种模式来描述要检索的内容。



**案例描述**：

任务描述

**案例代码**：（源文件名称） 案例/2.5表单、正则

代码

以下这两种写法是等价的。

var re = new RegExp("\\w+");

var re = /\w+/;

RegExp 修饰符：

修饰符用于执行不区分大小写和全文的搜索。

|  |  |
| --- | --- |
| i | 执行对大小写不敏感的匹配。 |
| g | 执行全局匹配（查找所有匹配而非在找到第一个匹配后停止）。 |
| m | 执行多行匹配。 |

方括号用于查找某个范围内的字符：

|  |  |
| --- | --- |
| **表达式** | **描述** |
| [[abc]](http://www.w3school.com.cn/jsref/jsref_regexp_charset.asp) | 查找方括号之间的任何字符。 |
| [[^abc]](http://www.w3school.com.cn/jsref/jsref_regexp_charset_not.asp) | 查找任何不在方括号之间的字符。 |
| [0-9] | 查找任何从 0 至 9 的数字。 |
| [a-z] | 查找任何从小写 a 到小写 z 的字符。 |
| [A-Z] | 查找任何从大写 A 到大写 Z 的字符。 |
| [A-z] | 查找任何从大写 A 到小写 z 的字符。 |
| [adgk] | 查找给定集合内的任何字符。 |
| [^adgk] | 查找给定集合外的任何字符。 |
| (red|blue|green) | 查找任何指定的选项。 |

元字符（Metacharacter）是拥有特殊含义的字符：

|  |  |
| --- | --- |
| **元字符** | **描述** |
| [.](http://www.w3school.com.cn/jsref/jsref_regexp_dot.asp) | 查找单个字符，除了换行和行结束符。 |
| [\w](http://www.w3school.com.cn/jsref/jsref_regexp_wordchar.asp) | 查找单词字符。 |
| [\W](http://www.w3school.com.cn/jsref/jsref_regexp_wordchar_non.asp) | 查找非单词字符。 |
| [\d](http://www.w3school.com.cn/jsref/jsref_regexp_digit.asp) | 查找数字。 |
| [\D](http://www.w3school.com.cn/jsref/jsref_regexp_digit_non.asp) | 查找非数字字符。 |
| [\s](http://www.w3school.com.cn/jsref/jsref_regexp_whitespace.asp) | 查找空白字符。 |
| [\S](http://www.w3school.com.cn/jsref/jsref_regexp_whitespace_non.asp) | 查找非空白字符。 |
| [\b](http://www.w3school.com.cn/jsref/jsref_regexp_begin.asp) | 匹配单词边界。 |
| [\B](http://www.w3school.com.cn/jsref/jsref_regexp_begin_not.asp) | 匹配非单词边界。 |
| \0 | 查找 NUL 字符。 |
| [\n](http://www.w3school.com.cn/jsref/jsref_regexp_newline.asp) | 查找换行符。 |

量词：

|  |  |
| --- | --- |
| n+ | 匹配任何包含至少一个 n 的字符串。 |
| n\* | 匹配任何包含零个或多个 n 的字符串。 |
| n? | 匹配任何包含零个或一个 n 的字符串。 |

<script>

var str="Is this all there is?";

var patt1=/is/g;

document.write(str.match(patt1));

</script>

test()：test()方法搜索字符串指定的值，根据结果并返回真或假。

var patt1=new RegExp("e");

document.write(patt1.test("The best things in life are free"));

exec()：exec() 方法检索字符串中的指定值。返回值是被找到的值。如果没有发现匹配，则返回 null。

var patt1=new RegExp("e");

document.write(patt1.exec("The best things in life are free"));

**案例说明**：

1）案例说明:

2）案例说明：



1）问题1：正则表达式的定义方式？修饰符的作用？

2）问题2：正则的两个方法？如何使用

****

1. 关于本知识点的注意事项1：

2）关于本知识点的注意事项2



练习1：

参考代码：(文件路径：练习/2.5正则表达式/邮箱格式)

练习2：

参考代码：(文件路径：练习/2.5正则表达式/02表单验证)

# 本章总结 [20分钟]

# 测试题 [分钟]

Key1 测试题1：

Key1 测试题2：

# 作业 [30分钟]

writer01 习题1：轮播图 （见： 案例/轮播图）

writer01 习题2：省市县三级联动 （见： 案例/省市县联动）

# 扩展部分（选讲内容）：

* 扩展部分1：
* 扩展部分2：

# 学员问题汇总（经验积累）：

* 学员听课中提问的问题：
* 学员练习中遇到的问题：