Javascript中的内部对象按使用方式可分为动态对象和静态对象。在引用动态对象的属性和方法时，必须使用new关键字来创建一个对象实例，然后才能使用“对象实例名.成员”的方法来访问其属性和方法：引用静态对象的属性和方法时，不需要用new关键字创建对象实例，直接使用“对象名.成员”的方式来访问其属性和方法即可。

**1.Object对象**

语法：obj=new object([value])

Obj：必选项。要赋值为object对象的变量名。

Value：可选项。任意一种javascript基本数据类型(Number、Boolen、String)。如果value为一个对象，返回不做改动的该对象。如果value为null、undefined，或者没有给出，则产生没有内容的对象。

**2.Object对象的属性**

1. prototype属性：该属性返回对象类型原型的引用。

语法：objectName.prototype

用prototype属性可以提供对象的类的一组基本功能，对象的新实例“继承”赋予该对象原型的操作。

例：为Array对象添加返回数组中最大元素值的方法，声明该函数，将它加入Array.prototype，并使用它。

function array\_max()

{

var i,max=this[0];

for(i=1;i<this.length;i++)

{

if(max<this[i])

max=this[i];

}

return max;

}

Array.prototype.max=array\_max;

var x=new Array(1,2,3,4,5,6);

var y=x.max();

1. constructor属性：该属性表示创建对象的函数。

Object.constructor

Object：必选项，是对象或函数的名称。

constructor属性是所有具有prototype的对象的成员，它们包括除Global和Math对象以外的所有javascript固有对象。constructor属性保存了对构造特定对象实例的函数的引用。

**3.object对象的方法**

(1)toLocaleString()方法

该方法返回一个日期，该日期使用当前区域设置并已被转换为字符串。

声明：dateObj.toLocaleString()

dateObj：必选项，为任意的Date对象。

toLocaleString()方法返回一个String对象，该对象包含了用当前区域设置的默认格式表示的日期。

注意：此方法只是用来显示结果给用户，不要在脚本中用来做基本计算，因为返回的结果是随机器的不同而不同。

(2)toString()方法：返回对象的字符串表示

objectname.toString( [radix] )

objectname：必选项，要得到字符串表示的对象。

radix：可选项，指定将数字值转换为字符串时的进制。

(3)valueOf()方法：返回指定对象的原始值。

object.valueOf()

object：是任意固有javascript对象

**String对象**：是动态对象，需要创建对象实例后才能引用该对象的属性和方法。

1. **创建String对象**

语法：var newstr=new String( StringText )

newstr：创建的String对象名

StringText：可选，字符串文本

例子： var str=new String("welcome come to accel world!!!");

document.write(str);

事实上字符串常量(用单引号或双引号括起来的字符串)都是一个String，可直接做为对象来使用，在后面加“.”，便可以调用String对象的属性和方法。区别在于两者返回的typeof值的不同。

字符串返回的是String类型，

String对象返回的是Object类型。

字符串例子：

**var str="welcome come to accel world!!!";**

**document.write(str);**

String对象例子：

var str=new String("welcome come to accel world!!!");

document.write(str);

**2.String对象的属性**

(1)length属性：用于获得当前字符串的长度

语法：stringObject.length

var p=0;

var newstring=new String("abcdefg"); //实例化一个字符串对象

var p=newstring.length;

alert(p.toString(16));

var p=0;

var newStr="abcdefg"; //定义一个字符串变量

var p=newStr.length;

alert(p.toString(16));

1. constructor属性：用于对当前对象的函数的引用

语法：Object.constructor

var newName=new String("sdf"); //实例化一个字符串对象

if(newName.constructor==String)

alert("this is string");

1. prototype属性：为对象添加属性和方法

语法：object.prototype.name=value

Object：对象名或字符变量名

Name：要添加的属性名

给information对象添加一个自定义属性salary，并给该属性赋值(2035)。

function personnel(name,age)

{

this.name=name;

this.age=age;

}

var information=new personnel("张\*租",27);

personnel.prototype.salary=null;

information.salary=2035;

alert(information.salary);

**Date对象**

1. **创建Date对象**

日期对象是对一个对象数据类型求值，该对象主要负责处理与日期和时间有关的数据信息。

语法：

dateobj=new Date()

dateobj=new Date(dateVal)

dateobj=new Date(year, month, date[, hours[, minutes[, seconds[, ms]]]])

dateobj 必选项，要赋值为Date对象的变量名。

示例：

返回当前的日期和时间。

var newdate=new Date();

document.write(newdate);

用年、月、日、时、分、秒来创建日期对象

var newdate=new Date(2035,8,7,12,30,45);

document.write(newdate+"<br>");

以字符串形式创建日期对象

var newdate=new Date("jan 2,2050 23:45:20");

document.write(newdate);

1. **Date对象的属性**

与String对象中的属性语法相同，此处只介绍用法。

1. constructor属性

判断当前对象是否为日期对象

var newdate=new Date();

if(newdate.constructor==Date)

document.write("是日期型对象");

1. prototype属性

用自定义属性来记录当前日期是本周的周几

var newdate=new Date();

Date.prototype.mark=null;

newdate.mard=newdate.getDay();

alert(newdate.mard);

Date对象的方法：由于是js的一种内部数据类型，该对象没有可以直接读写的属性，所有对日期时间的操作都是通过方法完成的。

举其中一个作为例子好了。

var newdate=new Date();

alert(newdate.getDay()); //星期天是0

event对象代表事件状态

**event对象的属性**

altLeft, ctrlLeft, shiftLeft属性的值都是Boolean类型。返回值true表示关闭，false为不关闭。

语法：[ window .] event.altLeft

button属性：获取事件发生时用户所按的鼠标键。

|  |  |  |  |
| --- | --- | --- | --- |
| 值 | 说明 | 值 | 说明 |
| 0 | 表示没有按键 | 4 | 按下中间键 |
| 1 | 按下左键(即主键) | 5 | 同时按下左键和中间键 |
| 2 | 按下右键 | 6 | 同时按下右键和中间键 |
| 3 | 同时按下左右键 | 7 | 同时按下左键、右键和中间键 |

如果脚本同时按下两个以上按键执行特殊动作，那么应该忽略单一按键动作。

注意：button仅用于onmousedown, onmouseup, onmousemove事件，对于其他事件，无论鼠标状态如何，都返回0。

clientX, clientY：获取鼠标在浏览器窗口的X, Y 坐标，只能获取鼠标的当前位置，而不能改变鼠标的位置。是一个只读属性。

语法：[ window.]event.clientX

X,Y属性

该属性设置或获取鼠标指针位置相对于CSS属性中有position属性的上级元素的X轴坐标。如果CSS属性中没有position属性的上级元素，默认以body元素作为参考对象。

语法：[ window.]event. X

如果鼠标事件触发后，鼠标移出窗口外，则返回的值为-1，具有只读属性。

cancelBubble属性

检测是否接受上层元素的事件的控制。该属性默认值为false，允许被上层元素的事件控制；否则为true，不被上层元素的事件控制。

语法：[ window.]event.cancelBubble[ =cancelBubble]

srcElement属性

该属性设置或获取触发事件的对象。srcElement属性是事件初始目标的HTML元素对象引用。由于事件通过元素容器层次进行处理，且可以在任何一个层次进行处理。

1. **动态创建FileSystemObject对象**

语法：fso=new ActiveXObject(“Scripting.FileSystemObject”);

1. **FileSystemObject对象的方法**
2. GetAbsolutePathName()方法

该方法根据提供的路径返回完整的路径，如果路径指定的是映射驱动器的根文件夹，那么完整的路劲将只能由一个路径分隔符“\”结束。

语法：object.GetAbsolutePathName( pathspec )

Pathspec参数

|  |  |
| --- | --- |
| 参数 | 说明 |
| “ ” | 返回当前的完整路径 |
| “ .. ” | 返回当前路径的上一级路径 |
| “ \\ ” | 返回当前路径根目录 |
| “ myfile ” | 在当前路径后加上该字符 |

var aa=new ActiveXObject("Scripting.FileSystemObject");

document.write(aa.GetAbsolutePathName(""));

1. GetBaseName()方法

将以字符串的形式返回指定路径中最后成分中的基本名称，不包含文件扩展名。

语法：object. GetBaseName(path)

Path：必选，当指定路径与path参数不匹配时，将返回长度为0的空字符串。

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetBaseName(path);

alert(s);

}

file("i:\\accel\\world\\pig.js"); //返回pig

注意：该方法不试图解析路径，也不检查指定路径是否存在。

1. GetDriveName()：根据指定路径返回包含驱动器名称的字符串

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetDriveName(path);

alert(s);

}

file("i:\\accel\\world\\pig.js"); //返回i:

注意：该方法不试图解析路径，也不检查指定路径是否存在。也就是说该方法只会返回你指定的驱动器名称，哪怕不存在。

1. GetDrive()方法：返回指定路径中驱动器的Drive对象

语法：object.GetDrive(drivespec) 都是必选参数

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetDrive(aa.GetDriveName(path)); //搭配GetDriveName()使用

alert(s); //返回D:

}

file("d:\\word\\java");

(5)GetExtensionName()方法

返回指定路径中最后成分扩展名的字符串

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetExtensionName(path);

alert(s); //返回burst

}

file("d:\\word\\java.burst"); //即使你的拓展名是胡扯的也可以运行

1. GetfFileName()方法：返回指定路径的最后成分

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetFileName(path);

alert(s); //返回java.burst

}

file("d:\\word\\java.burst");

1. GetParentFolderName()方法

根据指定路径中的最后成分返回其所有父文件夹名称的字符串。

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetParentFolderName(path);

alert(s); //返回d:\\word\\accel

}

file("d:\\word\\accel\\java.burst");

1. GetSpecialFolder()方法：返回指定的特殊文件夹对象

Object. GetSpecialFolder(folderspec)

Folderspec参数设置

|  |  |  |
| --- | --- | --- |
| 常数 | 值 | 说明 |
| windowsfolder | 0 | Window文件夹，包含了由window操作系统安装的文件 |
| Systemfolder | 1 | 包含库、字体，以及设备驱动程序的Systrm文件夹 |
| temporaryfolder | 2 | 用于存储临时文件的Temp文件夹，可在TMP环境变量找到 |

function file(path)

{

var aa,s="";

aa=new ActiveXObject("Scripting.FileSystemObject");

s+=aa.GetSpecialFolder(path);

alert(s); //返回c:windows

}

file("0");

GetTempName()方法

该方法并不创建文件，只是返回一个随机产生的临时文件或文件夹名。

<script type="text/javascript">

var fso,tempfile;

fso=new ActiveXObject("Scripting.FileSystemObject");

function CreateTempFile()

{

var tfolder,tfile,tname,TemporaryFolder=2;

tfolder=fso.GetSpecialFolder(TemporaryFolder);

tname=fso.GetTempName();

tfile=tfolder.CreateTextFile(tname);

return tfile;

}

tempfile=CreateTempFile();

tempfile.close();

</script>

注意：此程序并不输出。

**Drive对象：**负责收集系统中的物理或逻辑驱动器资源内容，如驱动器的共享名和有多少可用空间，要说明的是驱动器可以是硬盘和RAM磁盘。

1. **动态创建Drive对象**

该对象通过FileSystemObject对象的GetDrive()方法来创建

例子：对c盘驱动器创建一个Drive对象。

var fso=new ActiveXObject(“Scripting.FileSystemObject”);

var s=fso.GetDrive(“c:\\”);

1. **Drive对象的属性**
2. FreeSpace属性：返回指定驱动器或网络共享上的可用空间大小，只读。
3. IsReady属性

判断当前驱动器是否可用，如果指定驱动器已就绪，则返回true，否则返回false。

1. TotalSize属性

该属性以字节为单位返回驱动器或网络共享的所有空间大小

实例：显示指定驱动器的大小以及可用空间

<script type="text/javascript">

function DriveSize(dn)

{

var fso=new ActiveXObject("Scripting.FileSystemObject");

var s=fso.GetDrive(dn.value);

if(s.IsReady)

{

var str,allsize=0.0;

str="当前驱动器的名称为："+s.DriveLetter+"\n";

allsize=s.TotalSize/1024/1024/1024;

str="当前驱动器的大小为："+parseInt(allsize\*10)/10+"\n";

allsize=s.FreeSpace/1024/1024/1024;

str="当前驱动器的可用空间为："+parseInt(allsize\*10)/10;

alert(str);

}

else

alert("该驱动器无效！！！");

}

</script>

<form name="form1" method="post" action="">

盘符：

<input type="text" name="text1">

&nbsp;&nbsp;&nbsp;

<input type="button" name="Button1" value="磁盘空间" onclick="DriveSize(document.form1.text1)">

</form>

1. DriveType属性：返回一个值，表示所指定驱动器的类型
2. SeriaNumber属性：返回连续十进制数字，用于唯一标识磁盘卷

案例：显示驱动器的类型及系列号

<script type="text/javascript">

function dtype(Drivename)

{

var fso=new ActiveXObject("Scripting.FileSystemObject");

var s=fso.GetDrive(Drivename.value);

var t="",n="";

switch(s.DriveType)

{

case 0:t='找不到服务器';break;

case 1:t='移动硬盘';break;

case 2:t='固定硬盘';break;

case 3:t='网络资源';break;

case 4:t='CD-ROM';break;

case 5:t='RAM';break;

}

if(s.IsReady)

n='系列号为：'+s.SerialNumber;

alert(t+'\n'+n);

}

</script>

</head>

<body>

<form name="form1" method="post" action="">

驱动器名称：

<input type="text" name="text1" >

&nbsp;&nbsp;

<input type="button" name="button1" value="驱动器类型" onclick="dtype(document.form1.text1)">

</form>

</body>

1. AvailableSpace属性

返回在指定的驱动器或网络共享上可用的空间的大小

1. FileSystem属性

返回指定驱动器所使用的文件系统的类型

1. Path属性

返回指定文件、文件夹或驱动器的路径。

说明：驱动器字母后不包括根驱动器。例如c驱动器的路径是c:，不会c:\

1. RootFolder属性

返回一个Folder对象，表示指定驱动器的根文件夹，只读。

(10)ShareName属性

返回指定驱动器的网络共享名。

(11)VolumeName属性

设置或返回指定驱动器的卷名。读/写

Object.VolumeName[=newname]

Newname：可选，为指定的object的新名称。

**File对象：获取服务器端指定文件的相关属性**

1. **动态创建File对象**

GetFile()方法：根据指定的路径中的文件返回相应的File对象。

Object.GetFile(filespec)

Filespec：必选，指定文件的路径(绝对或相对路径)

如果指定的文件不存在，则出错。

将burst.txt文件以File对象进行实例化。

var fso=new ActiveXObject("scripting.FileSystemObject");

var s=fso.GetFile("f:\\accelworld\\burst.txt");

alert(s);

1. **File对象的方法**
2. Copy()方法

Object.Copy( destination[, overwrite]);

destination：必选，复制文件或文件夹的目标位置，不允许含通配字符。

Overwrite：可选，如果要覆盖已有文件(夹)，则为true(默认)，false。

1. Delete()方法

Object.Delete( force)

force：可，若删除设置了只读属性的文件(夹)，true，否则false(默认)。

1. Move()方法

Object.Move(destination)

Destination：可，移动文件或文件夹的目标位置，不允许通配字符。

|  |  |
| --- | --- |
| Move(), FileSystemObject.MoveFile(), FileSystemObject.MoveFolder() | |
| 单个文件(夹) | 三者都能实现此功能 |
| 多个文件(夹) | 后两者能实现此功能 |

1. OpenAsTextStream()方法

打开指定的文件并返回一个TextStream对象，可以对文件进行读、写或追加。

Object.OpenAsTextStream( [iomode,[ format ]] )

1. **File对象的属性**
2. Attributes属性：设置或返回文件(夹)的属性。

Object.Attributes[=newattributes]

案例：将只读文件改为可写文件

function ShowFilelnfo(filespec)

{

var fso,f,s;

fso=new ActiveXObject("Scripting.FileSystemObject");

f=fso.GetFile(filespec);

switch(f.Attributes)

{

case 0:s='普通文件';break;

case 1:s='只读文件';break;

case 2:s='隐藏文件';break;

case 4:s='系统文件';break;

case 32:s='文件在上次备份后已经修改';break;

case 33:s='只读文件(已修改)';break;

case 34:s='隐藏文件(已修改)';break;

case 128:s='压缩文件';break;

}

if(f.Attributes==1||f.Attributes==33)

{

if(confirm('当前文件为'+s+'\n是否将其改为可写文件'))

{

f.Attributes=f.Attributes-1;

}

}

else

alert('当前文件为：'+s);

}

</script>

</head>

<body>

<form name="form1" method="post" action="">

文件路径：

<input type="text" name="text1" value="f:accelworld\burst.txt" />

<input type="button" name="Button1" value="文件类型" onclick="ShowFilelnfo(document.form1.text1.value)">

</form>

</body>

1. DateCreated属性：返回指定文件(夹)的创建日期和时间。只读
2. DateLastAccessed属性：

返回最后访问指定文件(夹)的日期和时间。只读

1. DateLastModified属性：

返回最后修改指定文件(夹)的日期和时间。只读

案例：获取指定文件的创建、修改及访问时间

<script type="text/javascript">

function ShowFileDate(filespec)

{

var fso,f,s;

fso=new ActiveXObject("Scripting.FileSystemObject");

f=fso.GetFile(filespec);

var d=f.DateCreated;

Cdate=new Date(d);

d=f.DateLastModified;

Mdate=new Date(d);

d=f.DateLastAccessed;

Adate=new Date(d);

s='当前文件的创建时间为：'+Cdate.toLocaleString()+'\n当前文件的修改时间为：'+Mdate.toLocaleString()+'\n当前文件的创建时间为：'+Adate.toLocaleString();

alert(s);

}

</script>

</head>

<body>

<form name="form1" method="post" action="">

文件路径：

<input type="text" name="text1" value="f:accelworld\burst.txt" />

<input type="button" name="Button1" value="文件的相关日期" onclick="ShowFileDate(document.form1.text1.value)">

</form>

</body>