

JavaScript BOM

Window对象的常用属性

语法：

Window对象的常用方法

语法：

open()和close()方法

History对象

常用属性和方法

示例

Location对象

常用属性

常用方法

Document对象的常用方法

定时函数

超时调用：setTimeout()

间歇调用：setInterval()

JavaScript内置对象

Array对象

String对象

Math对象

Date对象

Window对象的常用属性

属性名称	说明
history	有关客户访问过的URL的信息
location	有关当前 URL 的信息
screen	只读属性，包含客户端显示屏幕的信息

语法：

```
window.属性名= "属性值";  
window.location="";
```

Window对象的常用方法

方法名称	说明
prompt()	显示可提示用户输入的对话框
alert()	显示带有一个提示信息和一个确定按钮的警示框
confirm()	显示一个带有提示信息、确定和取消按钮的对话框
close()	关闭浏览器窗口
open()	打开一个新的浏览器窗口，加载给定 URL 所指定的文档
setTimeout()	在指定的毫秒数后调用函数或计算表达式
setInterval()	按照指定的周期（以毫秒计）来调用函数或表达式
clearTimeout()	用于停止执行setTimeout()方法的函数代码
clearInterval()	用于停止 setInterval() 方法执行的函数代码

语法：

```
window.open( "弹窗内容的url", "窗口名称", "窗口特征", true/false );  
window.close();
```

open()和close()方法

属性名称	说明
height、width	窗口文档显示区的高度、宽度，以像素计
left、top	窗口的x坐标、y坐标，以像素计
toolbar=yes no 1 0	是否显示浏览器的工具栏，默认是yes

scrollbars=yes no 1 0	是否显示滚动条，默认是yes
location=yes no 1 0	是否显示地址地段，默认是yes
status=yes no 1 0	是否添加状态栏，默认是yes
menubar=yes no 1 0	是否显示菜单栏，默认是yes
resizable=yes no 1 0	窗口是否可调节尺寸，默认是yes
titlebar=yes no 1 0	是否显示标题栏，默认是yes
fullscreen=yes no 1 0	是否使用全屏模式显示浏览器，默认是no。 处于全屏模式的窗口必须同时处于剧院模式

History对象

保存用户上网的历史记录，可通过window.history属性访问

常用属性和方法

类别	名称	说明
属性	length	返回历史记录列表中的网址数
方法	back()	加载 History 对象列表中的前一个URL
	forward()	加载 History 对象列表中的下一个URL
	go()	加载 History 对象列表中的某个具体URL

示例

history.back()等价于，history.go(-1)浏览器中的“后退”

history.forward()等价于，等价于浏览器中的“前进”

Location对象

包含有关当前URL的信息，可通过window.location属性访问

常用属性

名称	说明
host	设置或返回主机名和当前URL的端口号
hostname	设置或返回当前URL的主机名
href	设置或返回完整的URL

常用方法

名称	说明
reload()	重新加载当前文档
replace()	用新的文档替换当前文档

Document对象的常用方法

Document对象代表整个HTML文档

名 称	说明
getElementById()	返回对拥有指定id的第一个对象的引用
getElementsByName()	返回带有指定名称的对象的集合
getElementsByTagName()	返回带有指定标签名的对象的集合
write()	向文档写文本、HTML表达式或JavaScript代码

定时函数

超时调用：setTimeout()

语法：window.setTimeout("调用的函数", 等待的毫秒数);

示例：

JavaScript | 复制代码

```
1 var myTime=setTimeout("disptime()", 1000);
2
3 //1秒(1000毫秒)后执行disptime()函数一次
```

间歇调用：setInterval()

语法：window.setInterval("调用的函数", 间隔的毫秒数);

示例：

JavaScript | 复制代码

```
1 var myTime=setInterval("disptime()", 1000);
2
3 //每隔1秒(1000毫秒)执行一次disptime()函数
```

JavaScript内置对象

Array：用于在单独的变量名中存储一系列的值

String：用于支持对字符串的处理

Math：用于执行常用的数学任务，包含若干个数字常量和函数

Date：用于操作日期和时间

Array对象

创建数组：

JavaScript | 复制代码

```
1 var arr = []; // 空数组
2 var arr2 = [1, 2, 3]; // 包含三个元素的数组
```

访问数组元素：



JavaScript

复制代码

```
1 var arr = [1, 2, 3];
2 console.log(arr[0]); // 输出: 1
3 console.log(arr[1]); // 输出: 2
4 console.log(arr[2]); // 输出: 3
```

修改数组元素:



JavaScript

复制代码

```
1 var arr = [1, 2, 3];
2 arr[1] = 4; // 把第二个元素改为 4
3 console.log(arr); // 输出: [1, 4, 3]
```

String对象

创建字符串:



JavaScript

复制代码

```
1 var str = ""; // 空字符串
2 var str2 = "Hello, world!"; // 包含字符串的双引号
3 var str3 = 'Hello, world!'; // 包含字符串的单引号
```

字符串连接:



JavaScript

复制代码

```
1 var str1 = "Hello";
2 var str2 = "world";
3 var str3 = str1 + " " + str2;
4 console.log(str3); // 输出: Hello world
```

字符串替换:



JavaScript

复制代码

```
1 var str = "Hello, world!";
2 console.log(str.replace("world", "Universe")); // 输出: Hello, Universe!
3 console.log(str.replace(/o/g, "a")); // 输出: Hella, warld!
```

Math对象

随机数：

`Math.random()`：返回一个0到1之间的随机浮点数。

以下代码生成一个0到100之间的随机整数：

▼ JavaScript 复制代码

```
1 int result = (int) (Math.random() * 100);
2 System.out.println(result);
```

Date对象

创建Date对象：

▼ JavaScript 复制代码

```
1 Date date = new Date();
2 System.out.println(date); // 输出: Sat Feb 27 19:55:22 CST 2023
```

格式化日期和时间：

▼ JavaScript 复制代码

```
1 Date date = new Date();
2 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
3 String dateString = sdf.format(date);
4 System.out.println(dateString); // 输出: 2023-02-27 20:09:01
```