**【基本概念】**

**标识符：**第一个字符为下划线、美元符号、字母；剩余的还可以为数字

**Undefined：**未初始化的值默认为undefined，未定义的值会报错；但typeof对未初始化和未定义的都会返回undefined；null==undefined

**Boolean**：Boolean()函数可以对任何类型进行转型， if判断时会自动执行Boolean

false，“”，undefined，null，0/NaN，会转换为false；

**Number**：浮点数内存空间是整数的两倍，浮点数最高精度为17位；ECMAScript保存的最大值为Number.MAX\_VALUE，最小值为Number.MIN\_VALUE

|  |  |
| --- | --- |
| isFinite() | 判断是否有穷 |
| isNaN() | 尝试将参数转换为数值，返回true/false |
| Number() | 用于任何类型；null/false/””都会被转换为0，“123b”被转换为NaN |
| parseInt | 两个参数，可以指定基数； |
| parseFloat | 不能设置基数，对整数会返回整数；只有第一个小数点有效 |

**String**：求字符串长度时，转义字符按一个字符算

|  |  |
| --- | --- |
| a.toString() | 除了null和undefined以外，其余类型均有此方法;可选参数：进制 |
| String(a) | 转型函数，将任何类型转为字符串（与Number()类似） |

**Object**：Object的每个实例都具有如下属性和方法

|  |  |
| --- | --- |
| constructor | 保存着创建当前对象的函数，默认为Object(); |
| hasOwnProperty(属性名a) | 检查属性是在实例中而不是在原型中，a必须为字符串 |
| isPrototypeOf(obj) | 检查obj是否是当前对象的原型 |
| propertyIsEnumerable(属性名a) | 检查属性是否可以用for-in枚举，a必须为字符串 |
| toLocaleString/toString/valueOf |  |

**操作符**：

|  |  |
| --- | --- |
| 一元+/- | 一元+与Number()作用一样、一元-先将进行Number(),再转换为负数 |
| ~a | 按位非，返回的为a的负值-1 |
| 位操作符 | &按位与、|按位或、^按位异或、<<有符号左移、>>有符号右移、<<<无符号….. |
| !逻辑非 | 返回布尔值，!!可与Boolean达到相同的效果 |
| a&&b逻辑与 | 短路操作，A或b不为布尔值时，不一定返回布尔值；a为obj时，返回b；a求值为TRUE时，返回b，a求值为假时，返回a |
| ||逻辑或 | 短路操作，与&&基本相反 |
|  | +、-、\*、/….. |

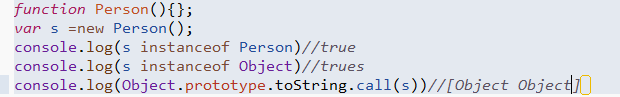
**With**语句：简化多次编写同意对象的工作；可使用保存局部变量代替

**Switch**语句中使用的是全等；

**函数**中执行完return后立即停止，后面的语句不再执行。

实参个数不受形参的影响，可以多于或少于

**Instanceof**:



**垃圾收集：**找到那些不再继续使用的变量，然后释放其占用的内存（按照固定时间间隔执行）

标记清除:垃圾收集器运行时给所有变量加上标记，然后去掉环境中的变量以及被引用的变量，然后在此之后再被加上标记的变量视为可删除的变量。

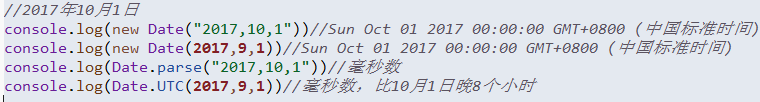
引用计数：

**【Object】**

创建方式

1. 字面量：属性名可以是字符串，数字会自动转换成字符串
2. new Object

**【Date】**





**【正则】**

**【函数】**

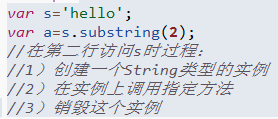
每个函数拥有两个属性：length和prototype；两个方法call和apply

Length表示形参的个数；

Prototype属性不可枚举，for-in无法发现

**【基本包装类型】（建议不使用）**

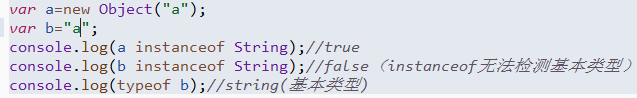
三个特殊的引用类型：Boolean,Number,String，每当读取一个此类基本类型值时，后台创建一个对应的包装类型对象，从而方便调用一些方法。



基本包装类型时引用类型，因此typeof会返回object ，转换为布尔值时是true

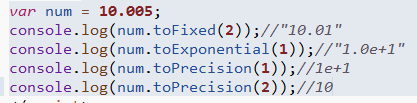


Object构造函数会根据出传入值返回相应的基本包装类型



【Number】

|  |  |
| --- | --- |
| **toFixed(a)** | 保留a位小数，自动实现四舍五入 |
| **toExponential(a)** | 返回以指数表示法表示的字符串形式，a指定输出结果的小数位数 |
| **toPrecision(a)** | 返回数值的最合适的格式，a表示总位数 |



【String】

|  |  |
| --- | --- |
| **s.charAt(a)** | 返回字符串s中a位置的字符（从0开始），**与使用s[a]效果一样** |
| **s.charCodeAt(a)** | 返回字符串s中a位置的字符编码（从0开始） |
| **concat()** | 不改变元字符串，接受任意多的字符（可以是数组），依次拼接；**使用+可达到效果** |

下面三个方法类似，截取字符串，当不指定第二个参数是作用一样，当a或b为负数时

|  |  |
| --- | --- |
| **slice(a,b)** | 不改变原字符串，a为开始位置，[b为结束位置] |
| **substr(a,b)** | 不改变原字符串，a为开始位置，[b为要返回的字符个数] |
| **substring(a,b)** | 不改变原字符串，a为开始位置，[b为结束位置] |

查找方法

|  |  |
| --- | --- |
| **indexOf(a,b)** | 从[b]开始查找a（子字符串）的位置，包括b位置的字符 |
| **lastIndexOf(a,b)** |  |

|  |  |
| --- | --- |
| **s.trim()** | 不改变原字符串，删除前置和后置空格 |
| **toLowerCase()** |  |
| **toUpperCase()** |  |

|  |  |
| --- | --- |
| **str.match(pattern)** | 与pattern.exec(str)相同，返回一个数组 |
| **str .search(pattern)** | 返回str中第一个匹配项的索引 |
| **replace** |  |

**【Math】**

**Math.max**接受任意多个数值参数（也可为数值字符串）



**Math.random**返回****的一个随机数，



**【数组】**

**join**：一个参数，作用：使用不同的分隔符来构建字符串

**push,unshift**——返回添加后的数组长度

**pop,shift—**—返回移除的项

**concat**：创建新数组，不改变原数组，将接受到的参数放在数组末尾；没有参数时复制当前数组并返回副本



**slice：**不改变原数组；两个参数，第一个为起始位置，第二个为结束位置；结束位置小于起始位置时，返回空数组



**Splice**：改变原数组，返回要删除的项（没有则返回空数组）；三个参数，起始位置、删除的项数、要插入的项（插入的时候若是数组以数组形式插入）

**Every**：函数对每一项都返回TRUE，则返回TRUE；

**Some**：函数对任意一项返回TRUE，则返回TRUE

**Filter：**返回函数返回TRUE的项组成的数组

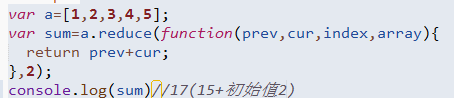
**forEach**：运行函数，没有返回值

**map**：返回每次函数调用的结果组成的数组

|  |  |
| --- | --- |
| 变 | Push/pop/shift/unshift/splice/ |
| 不变 | Concat/slice/五个迭代方法 |

**indexOf/lastIndexOf：**两个参数；要查找的项、[查找起点位置的索引]；返回要查找的项在数组中的位置

**reduce/reduceRight：**2个参数：函数、[归并初始值] ；函数有4个参数：前一个值、item、index、array；



【DOM】

文档节点是每个文档的根节点（document），文档节点的子节点称为文档元素（html）

【属性】

**nodeType：**每个节点都有，1为元素节点、2为属性节点、3为文本节点

**nodeName：**元素节点的nodeName返回其标签名（一般为大写）

**nodeValue：**只有文本节点和属性节点有此值，其他节点的nodeValue为null。

**childNodes：**每个节点都有，其中保存着一个NodeList对象（类数组对象），DOM结构的变化自动反应在NodeList对象中；（包含父节点中的所有类型节点）

**parentNode：**每个节点都有，指向父节点

**previousSibling/nextSibling：**第一个节点/最后一个节点的的previousSibling/nextSibling为null

**firstChild/lastChild：**指向父节点的第一个/最后一个节点。

**hasChildNode()：**判断节点是否包含子节点

**ownerDocument：**所有节点(document节点没有)都有，指向整个文档的文档节点

【操作节点】

**appendChild()：**在父节点末尾添加一个节点，返回新增的节点；若传入的节点已经是文档的一部分，那就是将该节点从原来的位置转移到新位置上；

**insertBefore(a,b)：**在b之前插入a节点，返回插入的节点a；若b为null，则执行与appendChild()相同的操作。

**replaceChild(a,b)：**用a替换b节点（注意：字符串的replace(a,b)方法是用b替换a），返回替换下来的节点b

**removeChild()：**移除节点，返回被移除的节点

（以上方法是对子节点的操作，父节点可通过parentNode获得，若不存在子节点是调用这些方法，将导致错误）

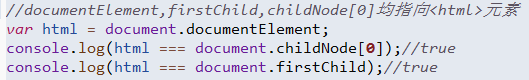
**a.cloneNode()**：所有类型节点都有，创建a的一个完全相同的副本；接受一个布尔值参数，若为true执行深复制（即复制节点及其整个子节点树），若为false执行浅复制，只复制节点本身。（此方法不会复制节点的js属性，例如事件处理程序）

normalize()：在操作DOM时，可能出现多个相邻文本节点/文本节点不包含文本；在某个节点上调用此方法，会在节点后代查找上述两种情况。若找到空文本节点，则删除，若找到相邻文本节点，则合并。

【Document类型】

nodeType=9，nodeName=“#document”，ownerDocument=null。

document对象表示整个HTML页面，是window对象的一个属性，因此可将其作为全局对象访问。**document.documentElement**指向<html>元素



**document.body**指向<body>元素，**document.title**指向<title>元素

**document.URL**包含页面的完整URL，**document.domain**包含页面的域名，**document.reference**保存着链接到当前页面的那个页面的URL。

**document.getElementById()**等：得到的是HTMLCollection对象，有一个nameItem()方法，通过元素的name特性去的集合中的项

**document.getElementsByName()**：此方法只有HTMLDocument类型才有；最常用在取得单选按钮（单选按钮的name值一样），这样可取得选中的单选按钮的值

另外，doument对象还有一些特殊集合

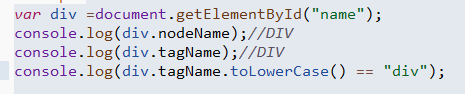
|  |  |
| --- | --- |
| document.anchors | 包含文档中所有带name特性的<a>元素 |
| document.forms | 包含文档中所有<form>元素 |
| document.images | 包含文档中所有<img>元素 |
| document.links | 包含文档中所有带href的<a>元素 |
| document.forms | 包含文档中所有<form>元素 |

文档写入

【Element类型】

nodeType=1，nodeName/tagName=标签名（大写）

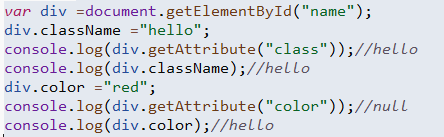
在HTML中，标签名始终以全部大写表示，因此比较之前最好经过大小写转换

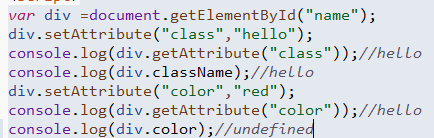


**特性和属性**

**setAttribute和getAttribute**可以设置/获得DOM 的本身属性和自定义设置的特性

当设置的是DOM本身存在的属性时，通过对象属性访问和getAttribute是同步的，但设置的是自定义属性，则不同步。





Attributes属性

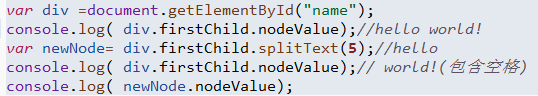
【Text类型】

nodeType=3，nodeName=“#text”,nodeValue=节点包含的文本

**createTextNode()**：创建文本节点

**normalize()**：合并文本节点

**splitText()**：分割文本节点



【comment类型】

nodeType=8，nodeName=“#comment”,nodeValue=注释内容

【DocumentFragment类型】

nodeType=11，nodeName=“#document-fragment”,nodeValue=null

**document.createDocumentFragment()**用于创建文档片段

【Attr类型】

nodeType=2，nodeName=特性的名称，nodeValue=特性的值

【DOM操作】

【动态脚本】【操作表格】

【使用NodeList】

NodeList、NamedNodeMap和HTMLCollection三者都是动态的（querySelectorAll返回的也是NodeList，但是是静态的）

NodeList：节点集合（包括元素节点、文本节点等）

HTMLCollection：元素集合

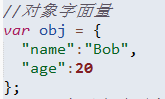
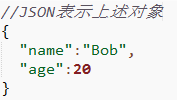
NamedNodeMap：存储着节点的“特性Attribute”集合

【JSON】

【语法】

JSON可以表示以下3种值（不支持变量、函数、对象实例）

1. 简单值：数值、字符串（字符串必须加**双**引号）、null、布尔值（不支持undefined）
2. 对象（属性名必须加引号、末尾没有分号、没有声明变量）

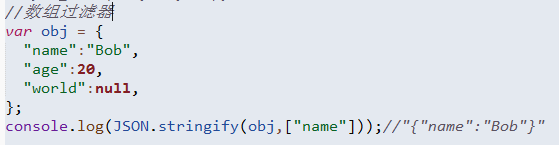
1. 数组

【序列化】

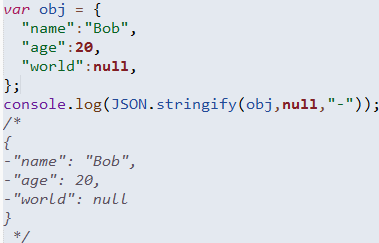
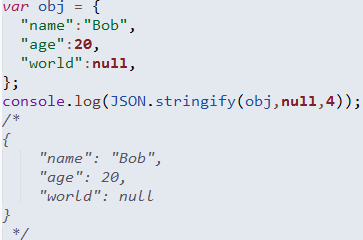
JSON.stringify(a,b,c)用于将js对象序列化为JSON字符串，所有函数及原型对象会被忽略，值为undefined的任何属性会被忽略，结果中最终都为有效JSON类型的实例属性。

a参数：待序列化的js对象

b参数：过滤器，可以是数组或者函数（函数的见高程p567）



C参数：缩进，可以为数字（缩进几个空白符）和字符串，只要传进有效的缩进，就会换行



【解析】

JSON.parse(a,b)将JSON字符串解析为原生js对象

a参数：待解析JSON字符串

b参数：函数（见高程p569）

【错误】

存在以下7种错误：Error、EvalError、RangeError、ReferenceError、SyntaxError、TypeError、URIError

【Error】

Error类型时基类型，其他错误类型继承自该类型

【EvalError】

EvalError类型会在使用eval()函数发生错误时抛出，各浏览器表现不太一样，极少出现

【RangeError】

RangeError在数值超出相应范围时抛出

【ReferenceError】

在找不到对象时发生

【SyntaxError】Syntax是语法的意思

【TypeError】

在变量中保存着意外的类型时

【URIError】

在使用encodeURI和decodeURI函数时，URI格式不正确会导致此错误。但发生概率很小，因为此两个函数容错率高

【DOM扩展】

【选择API】

querySelector：返回与选择符匹配的第一个元素

querySelectorAll：返回与选择符匹配的所有元素

（以上两个方法返回的是静态NodeList，ie8+）  
matchSelector：若找到可以匹配的元素则返回TRUE（不同的浏览器要加不同的前缀）

【元素遍历】(ie9+)

childElementCount：返回子元素的个数（不包含文本节点和注释）

firstElementChild：firstChild的元素版

lastElementChild：lastChild的元素版

previousElementSibling：previousSibling的元素版

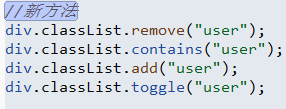
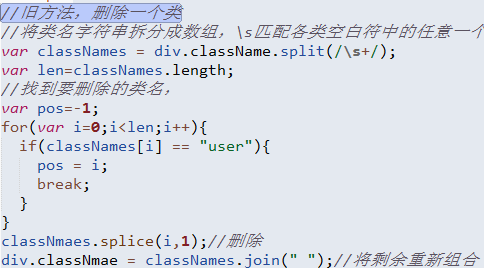
nextElementSibling：nextSibling的元素版

children：childNodes的元素版

【与类相关的扩充】

getElementsByClassName：返回动态NodeList，ie9+

classList属性：有4个方法add,contains,remove,toggle（classList属性firefox3.6+,chrome）



【焦点管理】ie4+

document.activeElement属性：等于DOM中当前获得焦点的元素，文档加载完时默认保存的是document.body

document.hasfocus():判断文档是否获得了焦点，可以知道用户是不是正在与用户交互

【HTMLDocument】

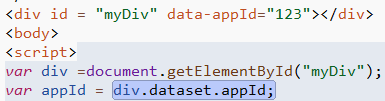
document的readyState：loading（正在加载文档），complete（已经加载完文档，图片和脚本都加载完，出发Window的load事件）

document.compatMode：在标准模式下等于“CSS1Compat”，在混杂模式下等于“BackCompat”

document.head：引用文档的head元素（chrome和Safari5）



【自定义属性】



【innerHTML】（已被html5纳入规范）

通过innerHTML插入的script不会执行，因为其被认为识“无作用域元素”（也就是在页面中看不到的元素），会被转换为空字符串。

解决方案：1）加上文本节点

2）包含非换行空格的元素

1. 隐藏的input域（推荐，因为上两种方法会影响布局，需要只会删除）

（不支持的innerHTML的元素有：table,tbody,thead,tr,tfoot,head,html,style,等

【outerHTML】（已被html5纳入规范）

div.outerHTML会包含div本身，即

实现p替换div元素

【insertAdjacentHTML方法】

【scrollIntoView方法】

可以在所有元素上调用，让调用元素出现在视口中。

传入参数true（或不传参数），让调用元素的顶部与视口顶部尽可能平齐

传入参数FALSE，让调用元素尽可能全部显示在视口中