# 基本概念

### 组成：

ECMAScript，由ECMA-262定义，提供核心语言功能

语法、类型、语句、关键字、保留字、操作符、对象。

DOM文档对象类型，提供访问和操作网页内容的方法和接口

DOM视图、DOM事件、DOM样式、DOM遍历和范围、SVG、MathML、SMIL

BOM浏览器对象类型，提供与浏览器交互的方法和接口

浏览器窗口和框架

兼容：支持核心语言功能，支持Unicode字符标准。

### 联入：

<script type=”text/javascript”>……</script>

<script type=”text/javascript” src=”ecma.js”></script>

属性：async 异步脚本，立即下载，不影响其他加载，加载顺序不一定

<script type+”text/javascript” src=”url” async></script>

defer 延迟脚本，立即下载，延迟到页面加载和显示后执行，加载顺序不一定

<script type=”text/javascript” src=”url” defer></script>

注释：//开头，多行 /\*……\*/

一般放在页面内容后面，</body>之前，按先后执行

浏览器在不支持或不启用脚本时用<noscript></noscript>可以包含除<script>的所有HTML元素

兼容XHTML

<script type=”text/javascript”>

//<![CDATA[

…]

//]]>

</script>

### 语法：

代码块用“{ }“包裹，代码用” ；“分割

+ - \* /前后加空格

使用代码块

if （test） {

test = false;

}

#### 标识符

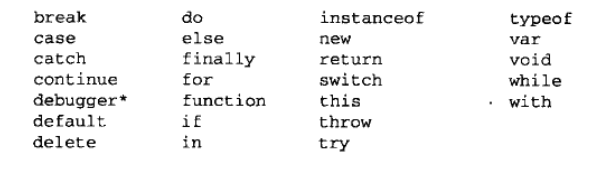
第一个字符是：字母，下划线 \_，美元字符 $。

后面可以有数字

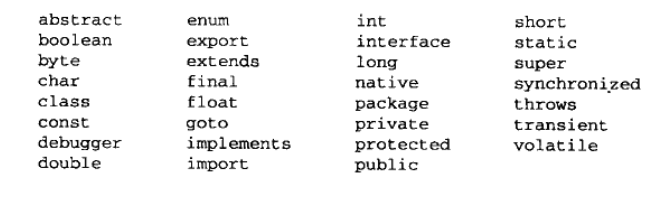
一般采取驼峰式

不能用：关键字、保留字、true、false、null。

##### 关键字



##### 保留字







#### 严格模式

“use strict” 切换为严格模式

Function doSomething() {

“us strict”;

//函数体

}

### 输出消息的五种方式

alert() 弹出对话框

confirm() 弹出带取消对话框，用于if

prompt() 接受用户信息

console() 在网页中控台输出消息

document.write() 在页面输出消息，可输入HTML标签

### 变量

##### Var 操作符

该操作符定义的变量属于该变量作用域中的局部变量。函数退出后就会被销毁

Function test() {

Var message = “hi’;//局部变量

}

Test ();

将var去除即为全局变量

Message = “hi”

不建议在局部作用域中使用全局变量

定义多个变量

Var message = “hi”,

Found = false,

Age = 29;

用逗号隔开

Var mess 声明： 在 环境/上下文 中指定一个变量的名字。

Var mess=1初始化：给一个声明后尚未初始化的变量一个有意义的初始值。

赋值 ： 销毁一个变量原来的值，并赋予一个新值。相当于改变了一个变量的状态。

注意：应注意区分初始化和赋值，在初始化之前不应该允许对变量进行赋值操作。

变量初始化之后不建议改变数据类型

Var mess=1,found=2,age=3;初始化多个变量用“ ，“隔开。

### 数据类型

#### 操作符

Typeof（） 返回数据类型

#### 简单数据类型

##### ****Undefined变量未初始化****

**var a1**

##### ****Null值为空****

**var mess=null**

**如果变量用于保存对象，将变量初始化为null，明确的让该变量保存null值**

##### ****布尔boolean****

**var x=true var x=false**

**转布尔值：Boolean( )**

|  |  |  |
| --- | --- | --- |
| Boolean | Ture | FALSE |
| string | 非空字符串 | “ ” 空字符串 |
| number | 任何非零数值，包括无穷大 | 0和NaN |
| Object | 任何对象 | null |
| Undefined | N/A（不适用） | undefined |

##### ****数字number****

**var a1=2**

**进制： 十进制，八进制（前面加0，严格模式下无效），十六进制（前面加0x）**

**浮点数：最高精确小数点后17位，（注意：不要测试特定的浮点数值0.1+0.2==0.3）**

**无穷大：Infinity**

**非数值：NaN ，（不等于任何值）**

**IsNaN( ) 判断是不是非数值**

**科学记数法：3.125e5 = 312500 0.000003=3e-6**

###### ****转换成数值****

* + - * 1. Number()

Undefined NaN

Null 0

Boolean,ture和false转换成1和0

number， 不变

string

纯数字转化为数值，16进制转化为10进制，（保留符号-+及小数，忽略前导0）

包含其他字符，转化为NaN

空字符串转化为0

如果是对象，调用valueOf()方法，如果结果是NaN，调用toString()方法

* + - * 1. ParseInt()

处理整数字符串

从第一个字符开始到第一个非数字字符结束，（认识符号，不认识小数，认识整数格式（各种进制））

空字符串和第一个字符非数字的字符串返回NaN

可使用转换基数 var num = parseInt(“af”,16) 16为转换基数（多少进制）

* + - * 1. ParseFloat ()

和parseInt()类似，区别：1 识别有效浮点数字字符，2 只解析10进制

##### ****字符串string****

**var a1=”3”;**

由0或多个16位Unicode字符组成的字符序列

###### 转义字符

|  |  |
| --- | --- |
| **代码** | **输出** |
| \' | 单引号 |
| \" | 双引号 |
| \& | 和号 |
| \\ | 反斜杠 |
| \n | 换行符 |
| \r | 回车符 |
| \t | 制表符 |
| \b | 退格符 |
| \f | 换页符 |

###### 转换成字符串

* + - * 1. .toString()

后面可以写基数，

* + - * 1. String()

几乎所有值都可以

有toString()方法，调用该方法

Null返回”null” undefined返回”undefined”

#### ****复杂****

##### Object

对象有时候被叫做关联数组，是属性和方法的集合，属性包含一个名和一个值，值为函数时，叫做方法。

###### New操作符

创建对象： **var o = new Object ();** new操作符后面跟创建的对象类型的名称，

Object

Object有的对象和方法会存在更具体的对象里

**constructor**：保存着用于创建当前对象的函数。对于前面的例子而言，构造函数（constructor）就是 Object()。

**hasOwnProperty(propertyName)**：用于检查给定的属性在当前对象实例中（而不是在实例的原型中）是否存在。其中，作为参数的属性名（propertyName）必须以字符串形式指定（例如：o.hasOwnProperty("name")）。

**isPrototypeOf(object)**：用于检查传入的对象是否是传入对象的原型

**propertyIsEnumerable(propertyName)**：用于检查给定的属性是否能够使用 for-in 语句来枚举。与 hasOwnProperty()方法一样，作为参数的属性名必须以字符 串形式指定。

**toLocaleString()**：返回对象的字符串表示，该字符串与执行环境的地区对应。

**toString()**：返回对象的字符串表示。

**valueOf()**：返回对象的字符串、数值或布尔值表示。通常与 toString()方法的返回值 相同。

### 操作符

用于操作数据值：算术操作符、位操作符、关系操作符、相等操作符

#### 一元操作符

只能操作一个值

包阔加减和递增，递减

++age age=age+1

age++ age=age+1

其他数据类型转化为数值变量，如数字值或NaN

###### 前置

Var age=34;

Var dse=2

Var cde = --age + dse //等于35

Var esd = age + dse //等于35

###### 后置

Var age=34;

Var dse=2

Var cde= age-- + dse //等于36

Var esd = age + dse //等于35

#### 位操作符

数值以64位格式储存。位操作符将64位转换位32位操作，再将值转换为64位。NaN和Infinity会被转化成0，非数值用Number()函数转换

有符号整数：前31位表示数值，第32位为符号位，0为正，1为负

无符号整数：32位数值，只有正数。

正数用纯二进制格式，负数用二进制补码

1 求绝对值 2 求反码 3 加1

##### 按位非(NOT)

由波浪线（~）表示，结果是返回数值的反码。一位操作数

Var num1=25;

Var num2=~num1; //输出-26

本质：操作数的负值减1。

##### 按位与(AND)

由和号字符表示(&)。两位操作数

将两个数的每一位对齐，同为1时返回1，否则返回0。

##### 按位或(OR)

由竖线符号表示（|）。两位操作数

至少有一位是1，返回1，否则返回零

##### 按位异或(XOR)

由一个插入符号表示（^）。两位操作数

只有一位是1，返回1，两位都是1或0，返回0

##### 左移

由两个小于号表示（<<），一位操作数

所有位向左移动指定位数，不会影响符号位，空白用0填充。

##### 有符号右移

由两个大于号表示（>>）,一位操作数

所有位向右移动指定位数，不会影响符号位，空白用0填充。

##### 无符号右移

由三个大于号表示（>>>）

所有位向右移动指定位数，（包括符号位），空白用0填充。

#### 布尔操作符

##### 逻辑非（NOT）

##### 逻辑与（AND）

##### 逻辑或（OR）

#### 乘性操作符

#### 加性操作符

#### 关系操作符

#### 相等操作符

#### 条件操作符

#### 赋值操作符

#### 逗号操作符

### 语句

#### If语句