# 运算符和表达式及数据类型转换

## 运算符和表达式

### 1、运算符和表达式分类

#### 2）赋值运算符和表达式

= 赋值运算 是把= 右边的值 赋值给 = 左边的变量；

=左边一定是变量（后期还有属性） =右边一定是有值的东西； \*\*\*\*\*\*\*\*\*\*\*

var a = 10;

var b = 20;

a = a + b;

a = b;

+= -= \*= /= %= 复合赋值运算符

a += b; <====> a = a + b;

a += 1; a = a + 1;

#### 3）条件（比较）运算符 和 表达式

> < >= <= == != ===全等 !== 不全等

比较运算符和表达式；最终表达式的值都是布尔值；

#### 4）逻辑运算符 常用在多个条件表达式的连接

&&与 ||或 ！非

1、通用：

&& 整个表达式的值：如果&&前面为真就取后面的值作为整个表达式的值如果&&前面为假就取前面的值作为整个表达式的值,后面的表达式根本不执行;

|| 整个表达式的值：如果||前面为真就取前面的值作为整个表达式的值，后面的表达式根本不执行;如果||前面为假就取后面的值作为整个表达式的值;

! 非真即假 非假即真 和它构成的表达式 最终都会是一个布尔值；

出现了！，！要求后面必须值是布尔值，不是就转化为布尔值，必须是布尔值；

var a = 10;

var b = 20;

var c = a || b;

2、常用：

多个条件表达式的连接的时候 ,两遍都是条件表达式才能使用

&& 一假则假

|| 一真则真

! 非真即假 非假即真

var a = 10;

var b = 20;

var c = a> b && a < 30;

案例：

var a = 10;

a = 0;

var b = '二狗子';

console.log(a && b);//对于数字，非0即为真

var a = 0.5;

a = 0;

var b = 2;

console.log(a && b++);

console.log(b);

      var str = 'i love you~';

      str = '';

      var c = 10;

      console.log(str || ++c);//对于字符串，非空(空串)即为真

      console.log(c);var c= 'a'&&'' || (true || (3<1 && 5<6))

    console.log(c)

#### 5）三目运算符 三元 问号冒号表达式

? :

一元 ++ -- ！ 一元运算符 a++ ++a !a !(a>b) +5 -5

一元运算符，优先级是最高的

var c = a + b++

二元 + - \* / % 二元运算符

&& || =(在二元运算符优先级很低的)

var c = a + b;

三元 ?: 是运算符当中唯一的一个三元运算符

var a = 10;

var b = 10?++a:a--;

问号冒号表达式：先看第一个表达式问号前面的值是否为真(不是布尔要转化为布尔)？ 如果为真，则取冒号前面的值作为整个表达式的值，后面的不执行，如果为假,则取冒号后面的值作为整个表达式的值,前面的不执行

案例：

var a = 10;

      var b = 20;

      console.log(a + b > 50 ? a++ : ++b);

      console.log(a,b);

      var a = 100;

      var b = 200;

      console.log('heihei'?a+b++:b--);

      console.log(a,b);

var str = '666';

      var a = 0;

      var c = (a && str)? str : (a++ && str);

      console.log(c,a);

var a = 5

var b = a<6?(a>7?a--:a>3?a++:--a):a=8

console.log(a,b)

## 数据类型转换

### 1、数据类型强制转换（显式）

#### 1）其它类型值转数字

Number() 强制将一个其它类型数据转化为数字类型

转不了就是NaN not a number

转化字符串

如果字符串整体来看是一个数字，那么就转化为这个数字

如果字符串整体来看不是一个数字，那么就转化为NaN

如果字符串是一个特殊的空字符串或者空白字符串，那么转化为0

转化boolean true会转化为1 false 会转化为0

转化undefined undefined 会转化为NaN

转化null null 会转化为0

#### 2）其它类型值转字符串

String() 强制将一个其它类型数据转化为字符串类型

转化字符串没有什么特殊，只要给我值，那么就会把这个值原样变为字符串输出；

#### 3）其它类型值转布尔值

Boolean() 强制将一个其它类型数据转化为boolean类型

转化数字的时候，除了0和NaN是false,其余都是true;

转化字符串的时候，除了空字符串是false,其余都是true

转化undefined和null都是false;

### 2、数据类型隐式转换

各种类型在适当的场合会发生隐式转换（隐式转换的规则和强制转换一样）

主要是运算、比较大小及判等的过程当中

判等的时候：先看两边是不是同种数据类型，如果是直接看是不是一样就完了；

如果不是同种数据类型，那么两边都转数字

当遇到null的时候;会有特殊情况发生,

特殊情况：空串和null不相等

特殊情况：false和null不相等

特殊情况: 0和null不相等

特殊情况：undefined 和 null 相等；

运算和比较情况下：

第一步：先看是不是 +

如果是+ 看有没有字符串，如果有 就是拼接字符串

第二步：再看是不是比较

如果是比较 看是不是两边都是字符串 如果是 比较的是字符串的ascll码

最后： 全部转数字

NaN: 所有的东西和NaN进行算术运算都是NaN 所有的东西和NaN进行比较大小都是false 所有的东西和NaN都不相等（包括自己）

NaN == NaN//fasle

全等和不全等： 不会出现类型转换，他们在判等的时候，先判断类型是不是一样，如果类型一样再判断值是不是相同，如果都相同才全等；如果有一个不一样，那么就不全等；

1 == true; //true

1 === true；//false

== 是用来判值的，最后的值相等就相等（它是可以发生隐式转化的）

=== 是类型和值都要判，必须全部相等才全等；

### 3、数据类型手动转换（其实就是在字符串当中提取数字）

字符串：数字字符必须在字符串的前面（前面可以有空白字符）；

否则就是NaN

parseInt()

parseFloat()

## 语句结构分类：

顺序结构 分支结构 循环结构

## if判断

### 1、单分支

if(一般都是一个表达式，但是最终只要有值就行，并且这个值最终会转化为boolean){

代码块；

}

if执行过程： 1、先计算小括号当中的值

2、根据小括号当中的值，转化为布尔值

3、根据转化出来的布尔值，决定是否需要执行花括号当中的代码块（如果是true就执行，如果是false就不执行，但是最终都还会执行下面的代码）

### 2、双分支（三元表达式）

if(一般都是一个表达式，但是最终只要有值就行，并且这个值最终会转化为boolean){

代码块；

}else{

代码块；

}

单分支和双分支其实都可以理解为双分支，也就是把一件事分成两种情况去处理，符合条件和不符合条件,单分支只是处理了符合条件的，不符合条件的不管

双分支是都要处理；

如果以后我们碰见if...else双分支语句，而且每个分支当中只有一条语句，就最好改成三元表达式（问号冒号表达式）；

### 3、多分支

多分支是把一件事分成多种情况去考虑，每一种情况都要去处理；

if(){

代码块

}else if(){

代码块

}else if(){

代码块

}else{

代码块

}

### 4、if判断案例

单分支

输入钱数，如果超过100万我就去环游世界

双分支

天气好就去看电影，不好就写代码

输入一个数，这个数如果大于0就-1，不大于0就+1

多分支：

输入体重判断属于什么样的体型

## switch....case分支语句

### 1、语法：

switch(有值的东西，最后这里面是一个值，不会进行转化){

case 值：

代码块；

break;

case 值：

代码块；

break;

default:

代码块；

break;

}

switch 语句执行过程： 先求出小括号当中的值 接着会拿着这个值从上到下和所有的标号后面的值进行对比，如果对比（是判断是否全等）成功，就执行对比成功这个标号下面的代码块；

## 练习题：

1. 输入一个数字，判断是星期几

2. 输入分数，判定优良差和不及格

90 ~ 100 A

80~90 B

70~80 C

60~70 D

0~ 60 不及格

其它 提示

强化练习

求出下面表达式的值

Number(''); 0

    Number('     ');0

    Number('123iloveyou');NAN

    Number('ilove123dd');NAN

    Number(true);1

    Number(false);0

    Number(undefined);NAN

    Number(null);0

    Number(NaN) NAN

    String(null); NULL

    String(undefined); undefined

    String(1234.56); 1234.56

    Boolean(NaN) ; false

    Boolean('') ;false

    NaN + NaN        nan

    null == ''       false

    null == null     true

    null == undefined true

    null == false false

    NaN == NaN   FALSE

    '100' + NaN 100NaN

    true == '    '; FALSE

    true + '123'; true123

    +'45'-false;

    parseInt('  12.345.67sdjfg'); 12

    parseFloat('  12.345.67sdjfg'); 12.345

    parseFloat('  iloveyou12.345.67sdjfg'); NaN