

变量

变量是程序的基本组成单位

变量定义

变量相当于内存中一个数据存储空间的表示

变量的使用注意

1. 变量表示内存中的一个存储区域 [不同的变量, 类型不同, 占用的空间大小不同, 比如: int 4 个字节, double 就是 8个字节, 先有基本印象, 后面说字节]
2. 该区域有自己的名称[变量名]和类型[数据类型]
3. 变量必须先声明, 后使用, 即有顺序
4. 该区域的数据/值可以在**同一类型**范围内不断变化
5. 变量在同一个作用域内不能重名
6. 变量=变量名+值+数据类型, 这一点请大家注意。变量**三要素**

程序中+号的使用

- 1) 两边都是数值型时, 做加法运算。
- 2) 左右两边有一方为字符串时, 做拼接运算。
- 3) 运算顺序: 从左至右。

数据类型

基本数据类型: 包括整数类型、浮点类型、字符类型、布尔类型。

引用数据类型

整数类型:

Java 的整数类型就是用于存放整数值的, 比如 12 , 30, 3456 等等

类 型	占用存储空间	范围
byte [字节]	1字节	-128 ~ 127 为啥存放的范围是这个=>二进制(二进制我们详解)
short [短整型]	2字节	$-(2^{15}) \sim 2^{15}-1$ -32768 ~ 32767
int [整型]	4字节	$-2^{31} \sim 2^{31}-1$ -2147483648 - 2147483647
long [长整型]	8字节	$-2^{63} \sim 2^{63}-1$

1. Java各整数类型有固定的范围和字段长度，不受具体OS[操作系统]的影响，以保证java程序的可移植性。
2. Java的整型常量（具体值）默认为 int 型，声明long型常量须后加 'l' 或 'L'
3. java程序中变量常声明为int型，除非不足以表示大数，才使用long
4. bit: 计算机中的最小存储单位。byte:计算机中基本存储单元,1byte = 8 bit。

浮点类型

Java 的浮点类型可以表示一个小数，比如 123.4 ， 7.8 ， 0.12 等等

类 型	占用存储空间	范围
单精度float	4字节	-3.403E38 ~ 3.403E38
双精度double	8字节	-1.798E308 ~ 1.798E308

- 1) 关于浮点数在机器中存放形式的简单说明, 浮点数=符号位+指数位+尾数位
- 2) 尾数部分可能丢失，造成精度损失(小数都是近似值)。

字符类型

字符类型可以表示单个字符, 字符类型是 char，char 是两个字节(可以存放汉字)，多个字符我们用字符串 String

- 1) 字符常量是用单引号引起来的单个字符。
- 2) Char 的本质是一个整数，输出时，是 unicode 对应的字符。
- 3) Char 类型可以进行运算，相当于一个整数。

布尔类型

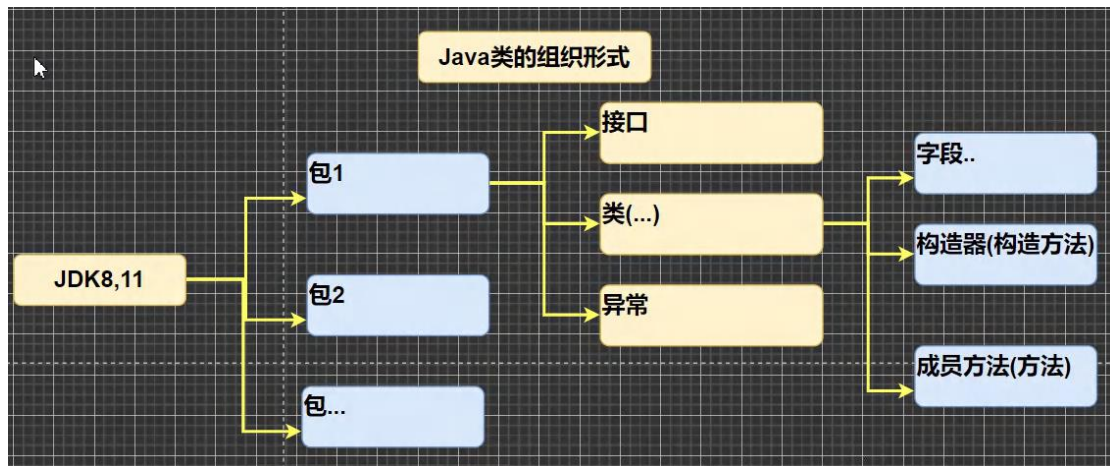
只允许取值 true 或者 false，占用 1 字节。

适用于逻辑运算，一般用于程序流程控制。

API 文档

API 是 java 提供的基本编程接口，提供了大量的基础类。

Java 类的组织形式



字符编码表

ASCII 码

1. ASCII码：上个世纪60年代，美国制定了一套字符编码(使用一个字节)，对英语字符与二进制位之间的关系，做了统一规定。这被称为ASCII码。ASCII码一共规定了128个字符的编码，只占了一个字节的后面7位，最前面的1位统一规定为0。特别提示：一个字节可以表示256个字符，ASCII码只用了128个字符。
2. 看一个完整的ASCII码表 [资料中]
3. 缺点：不能表示所有字符。

Unicode 码

1. Unicode的好处：一种编码，将世界上所有的符号都纳入其中。每一个符号都给予一个独一无二的编码，使用 Unicode 没有乱码的问题。
2. Unicode 的缺点：一个英文字母和一个汉字都占用2个字节，这对于存储空间来说是浪费。
3. 2的16次方是 65536，所以最多编码是65536个字符。
4. 编码0-127的字符是与ASCII的编码一样。比如 'a' 在ASCII码是 0x61，在 unicode码是 0x0061，都对应97。因此 Unicode码兼容 ASCII码。

Utf-8

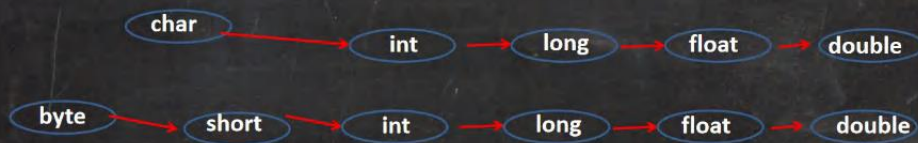
1. UTF-8 是在互联网上使用最广的一种 Unicode 的实现方式 (改进)
2. UTF-8 是一种变长的编码方式。它可以使用 1-6 个字节表示一个符号，根据不同的符号而变化字节长度。
3. 使用 大小可变的编码 字母占1个字节，汉字占3个字节

自动类型转换

✓ 介绍

当java程序在进行赋值或者运算时，精度小的类型自动转换为精度大的数据类型，这个就是**自动类型转换**。

✓ 数据类型按精度(容量)大小排序为(背，规则)



1. 有多种类型的数据混合运算时，系统首先自动将所有数据转换成容量最大的那种数据类型，然后再进行计算。

2. 当我们把精度(容量)大 的数据类型赋值给精度(容量)小 的数据类型时，就会报错，反之就会进行自动类型转换。

3. (byte, short) 和 char之间不会相互自动转换。

```
byte b = 10;
char c = b;
```

```
int num2 = 10;
//float f = num2 + 1.2;
double d2 = num2 + 1.2;
```

```
double d3 = 100;
int num3 = 1.1;
```

4. byte, short, char 他们三者可以计算，在计算时首先转换为int类型。

5. boolean 不参与转换

6. 自动提升原则：表达式结果的类型自动提升为 操作数中最大的类型

```
byte b = 10;
char c = 90;
short s = b+c;
```

强制类型转换

自动类型转换的逆过程，将容量大的数据类型转换为容量小的数据类型。使用时要加上强制转换符 ()，但可能造成精度降低或溢出，格外要注意。

```
public class ForceConvertDetail {
    //编写一个 main 方法
    public static void main(String[] args) {
        //演示强制类型转换
        //强转符号只针对于最近的操作数有效，往往会使用小括号提升优先级
        //int x = (int)10*3.5+6*1.5;//编译错误： double -> int
        int x = (int)(10*3.5+6*1.5);// (int)44.0 -> 44
        System.out.println(x);//44
        char c1 = 100; //ok
        int m = 100; //ok
        //char c2 = m; //错误
        char c3 = (char)m; //ok
        System.out.println(c3);//100 对应的字符，d 字符
    }
}
```

