笔记

String.valueOf()方法的使用

1. 由 基本数据型态转换成 String String 类别中已经提供了将基本数据型态转换成 String 的 static 方法 也就是 String.valueOf() 这个参数多载的方法 有下列几种 String.valueOf(boolean b) : 将 boolean 变量 b 转换成字符串 String.valueOf(char c) : 将 char 变量 c 转换成字符串 String.valueOf(char[] data) : 将 char 数组 data 转换成字符串 String.valueOf(char[] data, int offset, int count) : 将 char 数组 data 中 由 data[offset] 开始取 count 个元素转换成字符串 String.valueOf(double d) : 将 double 变量 d 转换成字符串 String.valueOf(float f) : 将 float 变量 f 转换成字符串 String.valueOf(int i) : 将 int 变量 i 转换成字符串 String.valueOf(long l) : 将 long 变量 l 转换成字符串 String.valueOf(Object obj) : 将 obj 对象转换成 字符串, 等于 obj.toString() 用法如: int i = 10; String str = String.valueOf(i); 这时候 str 就会是 "10" 2. 由 String 转换成 数字的基本数据型态 要将 String 转换成基本数据型态转 大多需要使用基本数据型态的包装类别 比如说 String 转换成 byte 可以使用 Byte.parseByte(String s) 这一类的方法如果无法将 s 分析 则会丢出 NumberFormatException byte : Byte.parseByte(String s) : 将 s 转换成 byte Byte.parseByte(String s, int radix) : 以 radix 为基底 将 s 转换为 byte 比如说 Byte.parseByte("11", 16) 会得到 17 double : Double.parseDouble(String s) : 将 s 转换成 double float : Double.parseFloat(String s) : 将 s 转换成 float int : Integer.parseInt(String s) : 将 s 转换成 int long : Long.parseLong(String

Pattern类使用

//Pattern类的compile方法用于编译一个正则表达式并返回一个编译好的pattern对象 Pattern patt=Pattern.compile("[a-z[1-9]]{1,9}"); //Pattern类的matcher方法用于封装一个要操作的字符串并返回Matcher对象，用于操作字符串 Matcher matcher=patt.matcher("xzc"); //Matcher类的matches方法用判断字符串和正则表达式是否匹配 if(matcher.matches()) System.out.println("匹配"); else System.out.println("不匹配");

Integer.toHexString()

此方法返回的字符串表示的无符号整数参数所表示的值以十六进制（基数为16）

(int) Math.pow(2, i)

求2的i次方返回一个int

算术运算

A = 0011 1100

B = 0000 1101

A&b = 0000 1100 A | B = 0011 1101 A ^ B = 0011 0001 ~A= 1100 0011

操作符	描述	例子
&	如果相对应位都是1，则结果为1，否则为0	（A&B），得到12，即0000 1100
|	如果相对应位都是0，则结果为0，否则为1	（A | B）得到61，即0011 1101
^	如果相对应位值相同，则结果为0，否则为1	（A ^ B）得到49，即0011 0001
~	按位取反运算符翻转操作数的每一位，即0变成1，1变成0。	（~A）得到-61，即1100 0011
<<	按位左移运算符。左操作数按位左移右操作数指定的位数。	A << 2得到240，即 1111 0000
>>	按位右移运算符。左操作数按位右移右操作数指定的位数。	A >> 2得到15即 1111
>>>	按位右移补零操作符。左操作数的值按右操作数指定的位数右移，移动得到的空位以零填充。	A>>>2得到15即0000 1111

逻辑运算符

假设布尔变量A为真，变量B为假

操作符	描述	例子
&&	称为逻辑与运算符。当且仅当两个操作数都为真，条件才为真。	（A && B）为假。
||	称为逻辑或操作符。如果任何两个操作数任何一个为真，条件为真。	（A || B）为真。
!	称为逻辑非运算符。用来反转操作数的逻辑状态。如果条件为true，则逻辑非运算符将得到false。	！（A && B）为真。

赋值运算符

操作符	描述	例子
=	简单的赋值运算符，将右操作数的值赋给左侧操作数	C = A + B将把A + B得到的值赋给C
+ =	加和赋值操作符，它把左操作数和右操作数相加赋值给左操作数	C + = A等价于C = C + A
- =	减和赋值操作符，它把左操作数和右操作数相减赋值给左操作数	C - = A等价于C = C - A
* =	乘和赋值操作符，它把左操作数和右操作数相乘赋值给左操作数	C * = A等价于C = C * A
/ =	除和赋值操作符，它把左操作数和右操作数相除赋值给左操作数	C / = A等价于C = C / A
%=	取模和赋值操作符，它把左操作数和右操作数取模后赋值给左操作数	C%= A等价于C = C%A
<< =	左移位赋值运算符	C << = 2等价于C = C << 2
>> =	右移位赋值运算符	C >> = 2等价于C = C >> 2
&=	按位与赋值运算符	C&= 2等价于C = C&2
^ =	按位异或赋值操作符	C ^ = 2等价于C = C ^ 2
| =	按位或赋值操作符	C | = 2等价于C = C | 2