**位运算符**

Java定义了位运算符，应用于整数类型(int)，长整型(long)，短整型(short)，字符型(char)，和字节型(byte)等类型。

位运算符作用在所有的位上，并且按位运算。假设a = 60，b = 13;它们的二进制格式表示将如下：

A = 0011 1100

B = 0000 1101

-----------------

A&B = 0000 1100

A | B = 0011 1101

A ^ B = 0011 0001

~A= 1100 0011

下表列出了位运算符的基本运算，假设整数变量 A 的值为 60 和变量 B 的值为 13：

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

实例

下面的简单示例程序演示了位运算符。复制并粘贴下面的Java程序并保存为Test.java文件，然后编译并运行这个程序：

Test.java 文件代码：

public class Test {

public static void main(String[] args) {

int a = 60; /\* 60 = 0011 1100 \*/

int b = 13; /\* 13 = 0000 1101 \*/

int c = 0;

c = a & b; /\* 12 = 0000 1100 \*/

System.out.println("a & b = " + c );

c = a | b; /\* 61 = 0011 1101 \*/

System.out.println("a | b = " + c );

c = a ^ b; /\* 49 = 0011 0001 \*/

System.out.println("a ^ b = " + c );

c = ~a; /\*-61 = 1100 0011 \*/

System.out.println("~a = " + c );

c = a << 2; /\* 240 = 1111 0000 \*/

System.out.println("a << 2 = " + c );

c = a >> 2; /\* 15 = 1111 \*/

System.out.println("a >> 2 = " + c );

c = a >>> 2; /\* 15 = 0000 1111 \*/

System.out.println("a >>> 2 = " + c );

}

}

以上实例编译运行结果如下：

a & b = 12

a | b = 61

a ^ b = 49

~a = -61

a << 2 = 240

a >> 2 = 15

a >>> 2 = 15