# **[java中的 |=、&=、^=](https://www.cnblogs.com/qubaba/p/11558127.html)**

## ****|=****

关于 |= 运算符：|= 运算符和 += 这一类的运算符一样，拆解开就是 a = a | b；

代码如下：

public static strictfp void main(String[] args) {

int a = 5; // 0000 0101

int b = 3; // 0000 0011

a |= b; // 0000 00111

System.out.println(a);

}

具体规则为：两个二进制对应位为0时该位为0，否则为1。拿5的二进制 0000 0101 和 3的二进制 0000 0011 进行|运算，后三位的的对应位都不是同时等于0，所以最终结果为 0000 0111 也就是7的二进制。

## **&=**

&= 和 |= 基本是一样的，只不过对于比较同位方式不同。

代码如下：

public static strictfp void main(String[] args) {

int a = 5; // 0000 0101

int b = 3; // 0000 0011

a &= b; // 0000 0001 System.out.println(a);

}

具体规则为：两个二进制对应位都为1时，结果为1，否则结果为都0。拿5的二进制 0000 0101 和 3的二进制 0000 0011 进行&运算，只有最后一位都为1，则最终结果为 0000 0001 也就是1的二进制。

## ****^=****

运算规则还是同上。

代码如下：

public static strictfp void main(String[] args) {

int a = 5; // 0000 0101

int b = 3; // 0000 0011

a ^= b; // 0000 0110 System.out.println(a);

}

具体规则为：两个二进制对应位相同时，结果为0，否则结果为1。拿5的二进制 0000 0101 和 3的二进制 0000 0011 进行^运算，1-5位对应位都是0所以1-5位都为0，第8位都为1所以第8位也为0，其他的对应位都不相等所以为1，则最终结果为 0000 0110 也就是6的二进制。

很好理解  
比如：  
1^0=0;  
0^0=1;  
**也就是说对同一个字符进行两次[异或运算](https://www.baidu.com/s?wd=%E5%BC%82%E6%88%96%E8%BF%90%E7%AE%97&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)就会回到原来的值。**

总结：

|=：两个二进制对应位都为0时，结果等于0，否则结果等于1；

&=：两个二进制的对应位都为1时，结果为1，否则结果等于0；

^=：两个二进制的对应位相同，结果为0，否则结果为1。