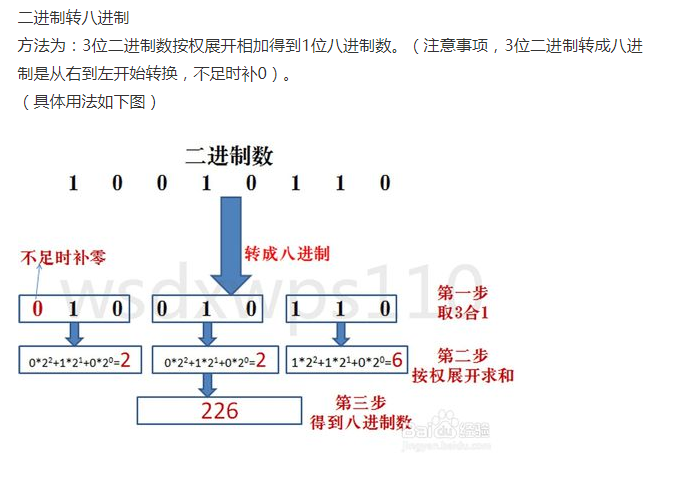
**Java位运算**

# 进制转换基础

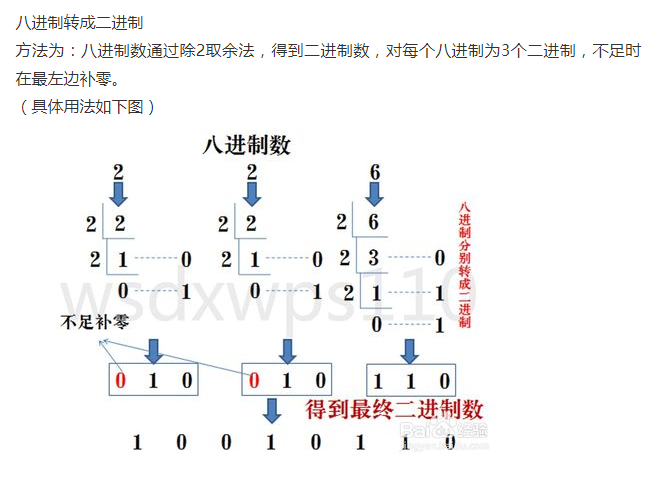
## 十进制转二进制

## 二进制转十进制

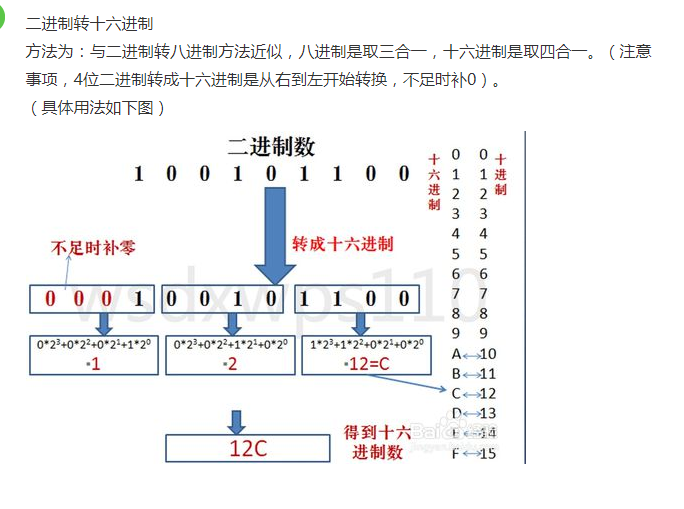
## 二进制转八进制



## 八进制转二进制



## 二进制转十六进制



## 十六进制转二进制

# Java基本数据类型所占字节

**整型：**

* **byte: 1个字节**
* **short : 2个字节**
* **int： 4个字节**
* **long ： 8个字节**

**浮点型：**

* + - **float：4个字节**
    - **double: 8个字节**

**char类型：**

* + - **char：2个字节**

**boolean类型：**

* + - **boolean：1个字节**

# Java源码、反码、补码

* **Java整数占4个字节，每个字节8位，一个共32位**
* **Java用补码表示整数**
* **最高位来表示符号位，1代表负数，0代表整数**

## **源码**

**就是将一个整型转化为二进制。**

**例如：**

**整数3和-3：**

**3： 0000 0000 0000 0000 0000 0000 0000 0011**

**-3：1000 0000 0000 0000 0000 0000 0000 0011**

## **反码**

**正数的反码与源码相同**

**负数的源码：符号位不变，其他取反**

**例如：**

**整数3和-3：**

**3： 0000 0000 0000 0000 0000 0000 0000 0011**

**-3： 1111 1111 1111 1111 1111 1111 1111 1100**

## **补码**

**正数的补码与源码相同**

**负数的补码：反码+1**

**例如：**

**整数3和-3：**

**3： 0000 0000 0000 0000 0000 0000 0000 0011**

**-3： 1111 1111 1111 1111 1111 1111 1111 1101**

# Java位运算

## 按位与（&）

**只有当对应位同为1时才是1否则为0**

**例如：**

**2&3**

**2: 0000 0000 0000 0000 0000 0000 0000 0010**

**3: 0000 0000 0000 0000 0000 0000 0000 0011**

**结果： 0000 0000 0000 0000 0000 0000 0000 0010 -> 2**

**-3&3**

**-3: 1111 1111 1111 1111 1111 1111 1111 1101**

**3: 0000 0000 0000 0000 0000 0000 0000 0011**

**结果: 0000 0000 0000 0000 0000 0000 0000 0001 -> 1**

**-3&-2**

**-3: 1111 1111 1111 1111 1111 1111 1111 1101**

**-2: 1111 1111 1111 1111 1111 1111 1111 1111**

**结果: 1111 1111 1111 1111 1111 1111 1111 1101**

**求源码： 补码-1后取反 (要带上符号)**

**0000 0000 0000 0000 0000 0000 0000 0011 -> -3**

## 2. 按位或（|）

**只有当对应位同为0时才是0否则为1**

## 3. 按位非（~）

**取反**

## 4. 按位异或（^）

**对应位不相同时为1，相同时为0**

**公式：**

**(a^b)^b 还是a**

**例子：**

**2^3**

**2: 0000 0000 0000 0000 0000 0000 0000 0010**

**3: 0000 0000 0000 0000 0000 0000 0000 0011**

**结果: 0000 0000 0000 0000 0000 0000 0000 0001 -> 1**

**用上面的结果再次 ^3**

**1^3**

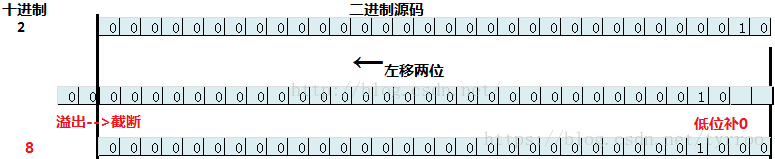
**1: 0000 0000 0000 0000 0000 0000 0000 0001**

**3: 0000 0000 0000 0000 0000 0000 0000 0011**

**结果: 0000 0000 0000 0000 0000 0000 0000 0010 -> 2**

## 5. 左位移（<<）

**算数左移：符号位不变，低位补0**



## 6. 右位移（>>）

**算数右移：符号位不变，高位补符号位**



## 无符号右位移（>>>）

**逻辑右移：高位补0，符号位也跟着变**

## 优先级

# 位运算应用