二进制运算符

> 二进制运算符做为选讲内容

二进制运算符也叫位运算符，它按数值的位来进行运算，在`ECMAScript`的系统里面，**所有的数据都是用64位存储，32位运算**，如果进行二进制的位运算，需要先将原来的数值转换成2进制操作

> 位操作符用于在最基本的层次上，即按内存中表示数值的位来操作数值。ECMAScript 中的所有数值都以 IEEE-754 64 位格式存储，但位操作符并不直接操作 64 位的值。而是先将 64 位的值转换成 32 位的整数，然后执行操作，最后再将结果转换回 64 位。对于开发人员来说，由于 64 位存储格式是透明的，因此整个过程就像是只存在 32 位的整数一样。

按位非（NOT）

按位非操作符由一个波浪线（~）表示，执行按位非的结果就是返回数值的反码。按位非是 ECMAScript 操作符中少数几个与二进制计算有关的操作符之一。

```
var num1 = 25;
var num2 = ~num1;
```

现在来看它的计算过程

第一步：先将这个数转2进制

```
num1.toString(2);    //11001
```

第二步：补码

位数	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25的二进制的补码	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

第三步：取反

位数	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25的二进制的补码	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
反码	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0

取反以后，我们发现它的最高位31位是一个数字`1`,这就说明这个数是一个负数，负数的二进制是需要进行转码的

第三步：将负数的二进制转码

位数	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
25的二进制	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
反码	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0
减1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1
保留符号再取反	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0

将负数的二进制转码以后，得到的结果是`11010`,但是它的符号位是`1`说明它是一个负数

```
parseInt("11010",2);      //26
//因为符号位是负数，所以最终的结果是-26
```

通过上面计算，其实我们发现二进制的取反过程它就是将原来的数乘以`-1`再减去1就可以

按位与（AND）

按位与操作符使用的符号是`&`

```
var result = 18 & 3;
```

这个操作过程也很简单

位数	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18的二进制	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
3的二进制	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
按位与	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

第一个数值的位	第二个数值的位	结　果
1	1	1
1	0	0
0	1	0
0	0	0

按位或（OR）

按位或操作符由一个竖线符号（|）表示，同样也有两个操作数。按位或操作遵循下面这个真值表

第一个数值的位	第二个数值的位	结　果
1	1	1
1	0	1
0	1	1
0	0	0

```
var result = 23 | 71;
```

位数	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
23的二进制	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
71的二进制	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1
按位或	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1

得到结果87