



- 
- 
- 
- 
- 
- 
- 
- 
- 
- 



700

-  or  |    
-  and  &    
-  not  ~ 
-  xor  ^   
-  left shift  << 
-  right shift  >> 
-  zero filled right shift  >>> 



bit



JavaScript 64

32



```
i = i | 0;
```

32

--	--	--	--

```
function toInt32(x) {
  return x | 0;
}
```

```
toInt32(1.001) // 1
toInt32(1.999) // 1
toInt32(1) // 1
toInt32(-1) // -1
toInt32(Math.pow(2, 32) + 1) // 1
toInt32(Math.pow(2, 32) - 1) // -1
```

[illegible]

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



```
0 | 3 // 3
```

[illegible]

```
2.9 | 0 // 2
-2.9 | 0 // -2
```

2147483647 ☐☐☐

```
2147483649.4 | 0;  
// -2147483647
```



0 & 3 // 0



10

-10



```
~~2.9 // 2
~~47.11 // 47
~~1.9999 // 1
~~3 // 3
```



Number



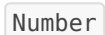
```
// 000~Number('011')
~'011' // -12

// 000~Number('42 cats')
~'42 cats' // -1

// 000~Number('0xcafebabe')
~'0xcafebabe' // 889275713

// 000~Number('deadbeef')
~'deadbeef' // -1
```

Number



```
// [] ~Number([])
~[] // -1

// [] ~Number(NaN)
~NaN // -1

// [] ~Number(null)
~null // -1
```



```
0 ^ 3 // 3
```


// 13

// -13

```
// (1 << 24)□□□□□□□□6□□
```

}

```
// "#bada55"
```

RGB HEX

--	--	--	--	--

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

[illegible]0[illegible]

1

[illegible]

2

[illegible]
$$/^{*}$$

```
// □□□□□ 0000000000000000000000000000000000000000000000010□
```

 $\ast/$

// -2

$$/^{*}$$

```
// 00-40000000 1111111111111111111111111111100
// 000000001000 1111111111111111111111111111110,
// 000000-2
*/
```

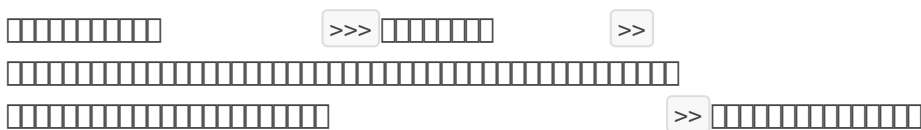
00000000 2 000000

```
5 >> 1
// 2
// 000 5 / 2 = 2

21 >> 2
// 5
// 000 21 / 4 = 5

21 >> 3
// 2
// 000 21 / 8 = 2

21 >> 4
// 1
// 000 21 / 16 = 1
```



```
4 >>> 1
// 2

-4 >>> 1
// 2147483646
/*
// 00-40000000111111111111111111111111100
// 0000000000000000111111111111111111111110
// 0000002147483646
*/
```

32

```
-1 >>> 0 // 4294967295
```

$(2^{32}) - 1$ 11111111111111111111111111111111 4294967295



```
var FLAG_A = 1; // 0001
var FLAG_B = 2; // 0010
var FLAG_C = 4; // 0100
var FLAG_D = 8; // 1000
```

A B C D

```
var flags = 5; // 0101

if (flags & FLAG_C) {
  // ...
}

// 0101 & 0100 => 0100 => true
```

C true false

A B D

```
var mask = FLAG_A | FLAG_B | FLAG_D;
// 0001 | 0010 | 1000 => 1011
```

A B D 1011

```
flags = flags | mask;
```



flags



```
flags = flags & mask;
```



toggle



```
flags = flags ^ mask;
```



0

1

1

0

```
flags = ~flags;
```



- Michal Budzynski, [JavaScript: The less known parts. Bitwise Operators](#)
- Axel Rauschmayer, [Basic JavaScript for the impatient programmer](#)
- Mozilla Developer Network, [Bitwise Operators](#)

#2

2018-01-11 16:01:15 UTC

2022-03-01 13:14:29 UTC