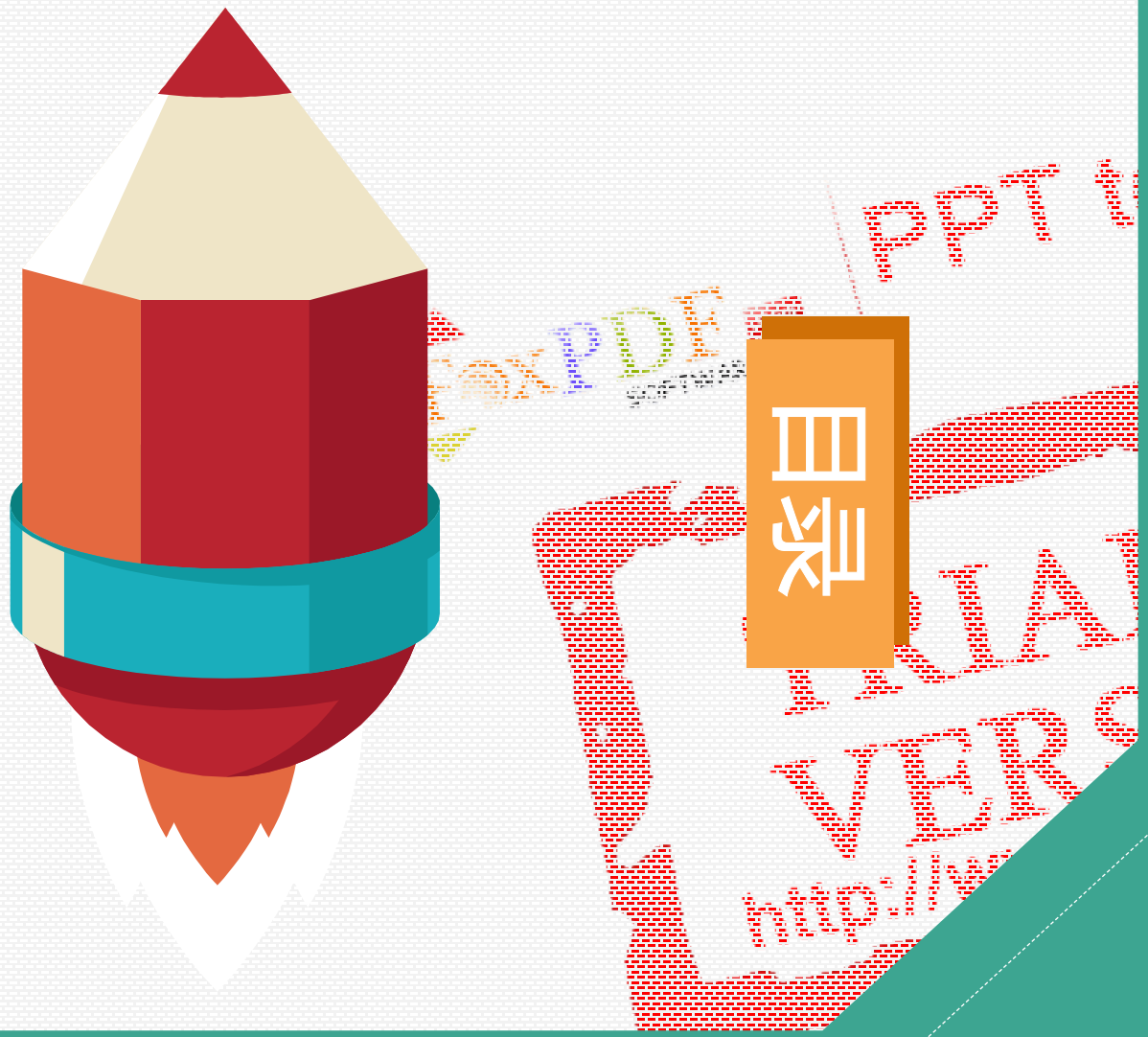

第八课：运算符详解第二节
主讲老师：万章



- 相等操作符
- 条件操作符
- 赋值操作符
- 逗号操作符

相等操作符

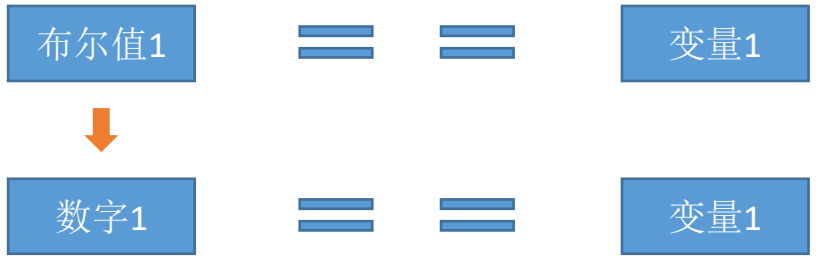
相等操作符之相等

相等和不相等
先转换再比较

相等操作符由两个等于号（==）表示，如果两个操作数相等，则返回true。
不相等操作符由叹号后跟等于号（!=）表示，如果两个操作数不相等，则返回true。
这两个操作符都会先转换操作数（通常称为强制转型），然后再比较它们的相等性。

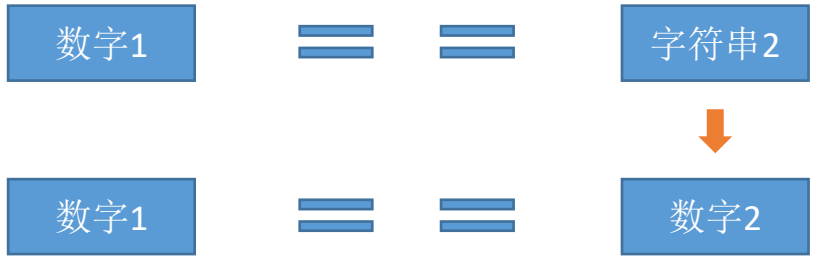
在转换不同的数据类型时，相等和不相等操作符遵循下列基本规则：

如果有一个操作数是布尔值，则在比较相等性之前先将其转换为数值——false 转换为0，而true 转换为1；




```
> false==1
< false
> false==0
< true
>
```

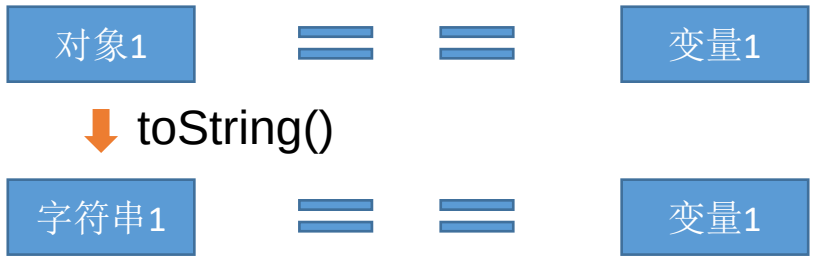
如果一个操作数是字符串，另一个操作数是数值，在比较相等性之前先将字符串转换为数值；




```
> false=="0"
< true
> false=="1"
< false
>
```

相等操作符之相等

如果一个操作数是对象，另一个操作数不是，则调用toString()方法，用得到的基本类型值按照前面的规则进行比较；




```
> var o={x:"2",y:"3"};
undefined
> var a=[1,2];
undefined
> o.toString();
"[object Object]"
> a.toString();
"1,2"
> o=="[object Object]"
true
> a=="1,2"
true
>
```

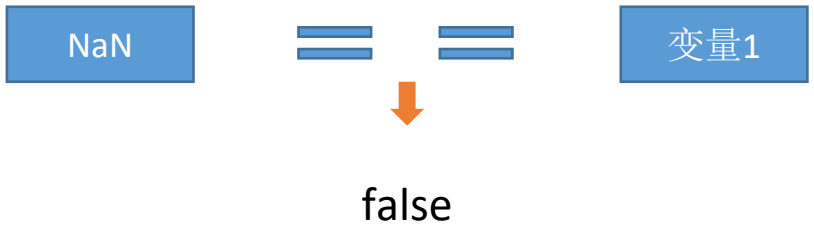
要比较相等性之前，不能将null 和undefined 转换成其他任何值。

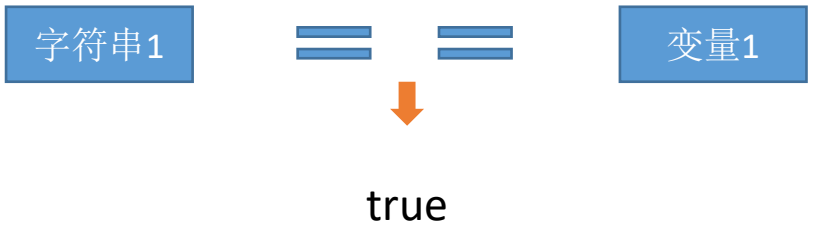
null 和undefined 是相等的。

```
> null==undefined
true
> undefined==0
false
> undefined==false
false
> null==null
true
> undefined==undefined
true
>
```

相等操作符之相等

NaN与任何数相比都不等, 包括和自己本身. 所以如果是相等判断, 那么只要有一个操作数是NaN, 结果一定是false. 如果是不相等判断, 那么只要有一个操作数是NaN,结果一定是true





如果两个操作数都是对象，则比较它们是不是同一个对象。如果两个操作数都指向同一个对象，则相等操作符返回true；否则，返回false。

相等操作符之相等

表 达 式	值	表 达 式	值
null == undefined	true	true == 1	true
"NaN" == NaN	false	true == 2	false
5 == NaN	false	undefined == 0	false
NaN == NaN	false	null == 0	false
NaN != NaN	true	"5"==5	true
false == 0	true		

某些特殊情况下的判断

相等操作符之全等

全等和不全等
仅比较而不转换

除了在比较之前不转换操作数之外，全等和不全等操作符与相等和不相等操作符没有什么区别。全等操作符由3 个等于号（===）表示, 不全等操作符由一个感叹号和两个等号(!==)表示，它只在两个操作数未经转换就相等的情况下返回true;

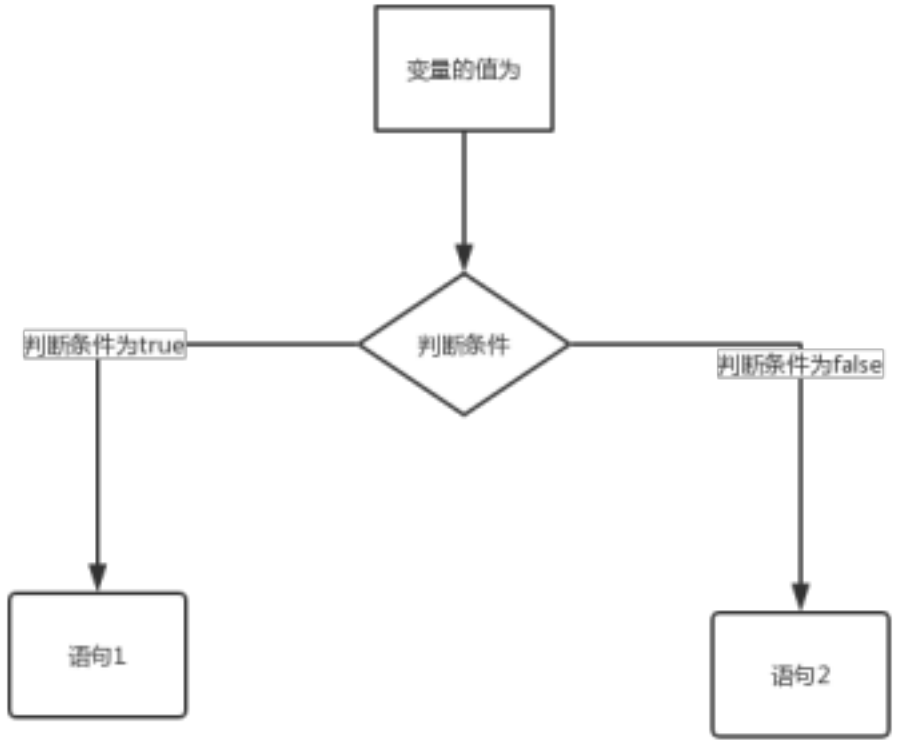
简而言之, 想要全等得满足以下条件:
1:数据类型一致(比如都是number或是string或是object等)
2:值完全一致

```
> null===undefined
< false
> false=="0"
< true
> false==="0"
< false
> [1,2,3]=="1,2,3"
< true
> [1,2,3]==="1,2,3"
< false
>
```

条件操作符

条件操作符(即三目运算符)

```
变量 = 判断条件?语句1:语句2;
```




```
var a=4>3?6:10;
```

```
> var a=4>3?6:10;
< undefined
> a
< 6
>
```

赋值操作符

赋值操作符

简单的赋值操作符由等于号（=）表示，其作用就是把右侧的值赋给左侧的变量

变量 = 值;

在等于号（=）前面再添加乘性操作符、加性操作符，就可以完成**复合赋值操作**

```
var num = 10;
num = num + 10;
```

等价

```
var num = 10;
num += 10;
```

每个主要算术操作符（以及个别的其他操作符）都有对应的复合赋值操作符。这些操作符如下所示：

乘/赋值（*=）； 例：num=num*10; -> num*=10;
除/赋值（/=）； 例：num=num/10; -> num/=10;
模/赋值（%=）； 例：num=num%10; -> num%=10;
加/赋值（+=）； 例：num=num+10; -> num+=10;
减/赋值（ =）； 例：num=num-10; -> num-=10;

逗号操作符

逗号操作符

逗号在CSS中被称为并列选择器，作用是同时使用多个选择器代码

选择器1，选择器2,选择器3{ CSS代码 }

在js中也是如此，我们可以在一个语句中执行多个操作

使用逗号操作符可以在一条语句中执行多个操作，如下面的例子所示：

```
var num1=1, num2=2, num3=3;
```

这行代码的意思是同时创造三个变量并进行赋值

相当于共用了一个 var