

数据类型转换

2020年02月17日, 星期一 09:32

1. 类型转换——将值从一种类型转换为另一种类型

- 隐式类型转换：通常是某些操作的副作用，不易看出
- 显示类型转换：可以在代码中明显看出

```
var a = 2; //number
var b = "3"; //string
var c = a + b;
console.log(c, typeof c); //?
```

隐式类型转换

强制类型转换

```
var a = 2; //number
var b = String(a);
console.log(a, typeof a); //?
console.log(b, typeof b); //?
```

河北师范大学软件学院

➤ 转换为 Number 类型

- 转换为 Number 类型规则：

值	结 果
undefined	NaN
null	0
布尔值	false 转换成 0, true 转换成 1
数字	保持不变（没什么好转换的）
字符串	解析字符串中的数字（忽略开头和结尾的空格）；空字符串转换成 0。比如 '3.141' 转换成 3.141

- 强制转换为 Number 类型：

parseInt()、parseFloat()、Number()

➤ 转换为string类型

- 转换为string类型规则：

值	结 果
undefined	'undefined'
null	'null'
布尔值	false->'false' true->'true'
数字	(例如, 3.141->'3.141')
字符串	输出即输入（无须转换）

- 强制转换为String类型：

string()

➤ 转换为Boolean类型

- 转换为Boolean类型规则：

值	转换成的布尔值
undefined	False
null	False
布尔值	与输入相同（不用转换）
数字	0, NaN 转换成 false, 其他数字转换成 true
字符串	"转换成 false, 其他字符串转换成 true

- 强制转换为Boolean类型：

Boolean()

2. NaN! =NaN (not a number, 没有意义的数字, NaN不等于他自身)

```
console.log( typeof NaN); //number
```

3.转换为Number类型习题:

```
console.log(1 + true); //2
console.log(1 + undefined); // NaN
console.log(1 + null); //1
console.log(1 * "123.4"); //123.4
console.log(1 * "123.4abc"); //NaN
console.log("2" > 10); //false
console.log(NaN == NaN); //false
```

true 转换为数字是1

null 转换为数字是0

```
var a;
if(a == a) {
    console.log(a * 1);
} else {
    console.log(a + 1);
}
```

请问在控制台输出结果为 ()

- ☐ A undefined ☐ B null
- ☒ C NaN ☐ D 1

4.转换为String类型习题:

```
console.log("" + true); //true
console.log("" + false); //false
console.log("" + 123); //123
console.log("" + NaN); //NaN
console.log("" + undefined); //undefined
console.log("" + null); //null
```

```
> var a = 1;
   var b = "2";
   var c = 3;
   console.log(a + b + c);
   console.log(a + c + b);
   console.log(b + c + a);
```

123

42

231

< undefined

结果均为String类型

- "+" 运算符左右两侧有字符串时为拼接运算符。
- 运算符等级相同时，从左往右计算。

5.转换为Boolean类型习题:

逻辑运算符会将数据类型转换为布尔类型之后再运算

```
var a;  
console.log(a+1);///  
console.log(!a+1);///  
console.log(!!a+1);///  

```

a是Undefined类型，所以a+1是NaN

! 是一个逻辑运算符，取非

!a先把a进行隐式类型转换，转换成布尔类型，转换后为false，然后再取非变成true，因此 !a+1=1+1=2

!!a相当于!a再次取非，这是!!a为true，因此!!a+1=1+1=2

把一个数据转换成布尔类型最简单的方法就是两次取非!

把一个布尔类型转换成Number类型最简单的方法就是*1，eg: a=true; a*1=1.

6.综合练习

```
var a;  
var b = a * 0;  
if(b == b) {  
    console.log(b * 2 + "2" - 0 + 4);  
} else {  
    console.log(!b * 2 + "2" - 0 + 4);  
}
```

a是Undefined，所以b结果为NaN

NaN != NaN,所以进入到 !b * 2 + "2" - 0 + 4的运算中,

根据优先级，b转换为布尔类型为false，所以 !b就为true，所以 !b * 2 = 1 * 2 = 2

接着2 + "2" = "22"

"22" - 0 = 22 - 0 = 22 ("22" 转换为Number类型 22)

22 + 4 = 26

因此最终结果为26.