链滴

javascript 学习笔记 - 类型和变量

作者：wthfeng

原文链接：https://ld246.com/article/1480077318453

来源网站：链滴

js是前端的基础。最近又对着《javascript权威指南》重头学了一遍，感慨挺多于是记录下看书过程中的关键点以备忘，相当于一个精简版的javascript权威南。该笔记主要以代码为主，捎带讲解，因为我相信代码即是最好的说明。

一. 类型和变量

1.数据类型

关键点：

- **javascript数据类型分为两类：原始类型和对象类型**
- **原始类型包括：字符串、数值、布尔、null和undefined**
- **null(空)、undefined(未定义)是它们各自类型的唯一成员**
- **对象是属性的集合，每个属性由键值对组成。**
- **数组可看为是一种特殊的对象。普通对象是属性的无序集合，而它是属性的有序集合，且属性为数字**
- **函数可认为是具有可执行代码的特殊对象**
- **js变量是无类型的，可以被赋给任何类型的值**

笔记：

类可看做是对象的子类型，是一类对象的集合。如数组和函数就是一类对象的集合。可以这样理解：符合属性为数字且有length属性的一类对象叫做数组，这些对象又赋予了各种方便数组操作的特殊行。

除数组和函数外，js还有三大内置类：Date()日期类、RegExp(正则类)、Error(错误类)。

```
//js的数值类型，用一个全局变量Infinity表示无穷大，NaN表示非数值
//js在算术运算时，发生溢出或被0整除时不会报错。会返回Infinity或NaN
//NaN表示非数值，它和任何值都不相等，包括它自身。判断一个数是否为NaN使用isNaN()
var num = 4/0;
console.log(num); //Infinity
console.log(0/0); //NaN
var n1 = 'abc';
var n2 = 0/0;
console.log(isNaN(n1)); //true ,表示非数值
console.log(isNaN(n2)); //true
console.log(NaN==NaN); //false
```

笔记（重要）：

关于js中的假值是：0、-0、NaN、undefined、null、''。其他为真值。包括字符串'0'。

2.关于null和undefined

关键点：

- **null是关键字，而undefined是全局变量**

- **null==undefined为true，若比较需用严格相等符===**
- **两者均为假值**
- * **两者均无方法或属性，任何对它们的调用都会报错**

```
//null和undefined都没有任何属性和方法。用它们调用会有类型错误。
//null为特殊的对象，意为非对象，是关键字。
//null用来表示数字、字符串、对象是非值的。是null类型唯一成员。
console.log(typeof null); //object ,说明是一种特殊对象类型，

//undefined是全局变量。表示变量没有初始化，说明属性或元素不存在或方法没有返回值
//是全局变量，是undefined类型唯一成员。
console.log(typeof undefined); //undefined
console.log(null==undefined); //true ， 两者都表示空值
console.log(null===undefined); //false, 需用===严格相等运算符计算
```

3. 全局对象

关键点：

- **全局对象无需声明即可使用**

```
// 当页面重新加载时，会创建一个全局对象，这个全局对象有许多属性和全局方法，如：
//全局属性： undefined，NaN * 全局函数：isNaN(),parseInt(),eval();
// 全局对象：Math、JSON * 以上这些可直接使用。
console.log(isNaN('dfg')); // true ， 表明是非数字
console.log(parseInt('23',10)); //23
```

4.字符串、布尔和数值类型

关键点：

- **字符串、数值、布尔的字面量不是对象，却有属性和方法**
- **当它们的字面量发生属性、方法调用时，会创建一个临时对象负责，调用完成，临时对象销毁**

```
//关于字符串、数值、布尔类型
//它们不是对象，但却有方法和属性，原因是：如字符串，当调用s.subString()时，
//会通过new String()构建一个临时对象，方法的调用都是来自这个临时对象。
//null和undefined没有包装对象，若访问会报错

 var str = 'abc';  // 字面量可直接调用方法
 console.log(str.charAt(0)); // a
 console.log(str.substring(1,3)); // ab,包括起始边界不包括终止边界,[a,b)

 var s1 = null;
 //console.log(s1.charAt(0)); //TypeError: Cannot read property 'charAt' of null
```

```
//str为字符串类型，当调用其属性时，创建一个String的临时对象，
//将属性len赋值为4，执行后临时对象销毁
str.len = 4;
//再次访问，又重新创建临时对象并访问String对象的len,String没有len属性。
//修改只存在临时对象本身，并没有保存下来
console.log(str.len); // undefined

//注意： str2是用构造函数创建的String对象，与字面量的字符串不同，它是真正的对象。
var str2 = new String();
str2.len =4; //为对象添加一个新属性
console.log(str2.len); //4 ,输出属性
```

笔记：

可通过String()、Boolean()、Number()创建的相应的对象，它们与用字面量创建的有所不同。前者对象（包装对象），而后者是原始类型。类似于java中int和Integer的概念。

```
var s1='abc';
var s2 = new String('abc');
console.log(typeof s1); // string
console.log(typeof s2); // object,可认为是string对应的包装类型，它是一种对象。
console.log(s1==s2); //true ,两者值相等
console.log(s1===s2); //false ， 两者类型不相等
```

5. 关于基本类型及对象

关键点：

- **基本类型不可变，对象可变。这是它们重要的区别。**
- **基本类型是值的比较，而对象是引用地址的比较**

部分对象与基本类型转换表:

值	转为字符串	转为数字	转为布尔	为对象
undefined	"undefined"	NaN	false	hrows TypeError
null	"null"	0	false	th ows TypeError
true	"true"	1	本身	n w Boolean(true)
false	"false"	0	本身	oolean(false)
空串('')	本身	0	false	ew String("")
非空数字(1)	1	本身	true	ew String("1")

非空非数字('a') 本身 NaN true
ew String("a")

0 "0" 本身 false new
Number(0)

NaN "NaN" 本身 false
ew Number(NaN)

Infinity "Infinity" 本身 true
ew Number(Infinity)

999(常数,非零) "999" 本身 true
ew Number(999)

```
var s = '';
console.log(new Number(s)); // [Number:0]

var n = null;
console.log(new Number(n)); //[Number:0]
console.log(new String(n)); //[String:'null']

var u = undefined;
console.log(new Number(u)); //[Number:NaN]
console.log(new String(u)); //[String:undefined]

var f = false;
console.log(new Number(f)); //[Number:0]
console.log(new String(f)); //[String:'false']
```

6.对象与原始值的转换

前面提到原始值与对象的转换，对象转原始值较复杂，所以抽出一节来讲。

关键点：

- **对象转布尔值全为真值。**
- **对象转字符串通过转换方法实现：toString()和valueOf()**

1. toString()函数

我们将对象分为包装对象（即字符串、布尔、数值的包装类）、特殊对象（即内嵌特殊行为的对象，一类对象，如数组、日期、函数）、和普通对象。所有对象 都继承了此方法。**默认实现只是简单输出象的类型信息，而许多特定的对象（包装对象和类对象）又重新实现了它的行为，使其包含更多对象息。**

```
//普通对象的toString方法
//固定输出[object Object]
var obj ={
 a:'abc',
 b:'bcd',
```

```
c:12,
d:function(){
  console.log(this.a);
 }
};
console.log(obj.toString()); //[object Object]
var obj2 ={  a:[1,2,3]  }
console.log(obj2.toString()); // [object Object]


  //包装对象的toString方法
  //输出其原始值
 var b = new Boolean(true);
 console.log(b.toString()); //true
 var num = new Number(10);
 console.log(num.toString()); //10

 /**
  * 特殊对象，数组、函数、日期等的toString()方法
  * 各种类型均有特定实现
  */

 //数组toString()，每个元素添加','组成字符串，与join(',')相同
 var arr = [1,2,3,4];
 console.log(arr.toString()); // 1,2,3,4
 console.log(arr.join(',')); //1,2,3,4

 // 日期toString()，输出表示日期时间的可读字符串
 var date = new Date();
 console.log(date.toString()); //Fri Oct 21 2016 08:11:00 GMT+0800 (中国标准时间)
 // 函数toString(),输出函数定义源码
 console.log(obj.d.toString()); // function (){console.log(this.a);}
```

2. valueOf()函数

如果对象可以转为原始值，如包装类，则输出原始值，否则返回对象本身

```
//若对象有原始值，valueOf()输出原始值

 var b = new Boolean(true);
 console.log(b.valueOf()); //true
 console.log(b); //[Boolean: true]
 var num = new Number(10);
 console.log(num.valueOf()); //10
 console.log(num); //[Number: 10]
 var str = new String('abc');
 console.log(str.valueOf()); //abc
 console.log(str); //[String: 'abc']

 /*
  * 若对象无原始值，则简单输出对象本身。需注意日期类输出的是它内部的表示值。
  */
```

```
var obj ={  a:'abc',  b:'bcd',  c:12,  d:function(){  console.log(this.a);  }  };
console.log(obj.valueOf()); //{ a: 'abc', b: 'bcd', c: 12, d: [Function] }
console.log(obj); // { a: 'abc', b: 'bcd', c: 12, d: [Function] }
var obj2 ={  a:[1,2,3]  }
console.log(obj2.valueOf()); // { a: [ 1, 2, 3 ] }
console.log(obj2); //{ a: [ 1, 2, 3 ] }


//特殊对象，数组、函数、日期等的valueOf()方法


//数组
var arr = [1,2,3,4];
console.log(arr.valueOf()); //[ 1, 2, 3, 4 ]
console.log(arr); //[ 1, 2, 3, 4 ]

// 日期  var date = new Date();
console.log(date.valueOf()); //1477011865082
console.log(date); // Fri Oct 21 2016 09:04:25 GMT+0800 (中国标准时间)

// 函数  console.log(obj.d.valueOf());
// [Function]  console.log(obj.d); //[Function]
```

提示： 可以将上述两段toString()和valueOf()代码结合起来，会有更深认识。

全文完