JavaScript笔记

JavaScript数据类型：

原始类型：数字，字符串，布尔值，null,undefined

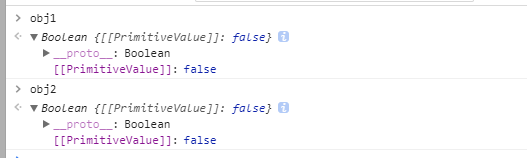
对象类型：数组，函数，普通对象，日期（Date），正则（RegExp），错误（Error）

JavaScript类型转换

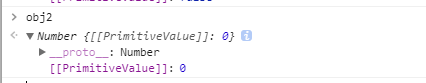
隐式类型转换：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 值 | 转换为 | | | |
| 字符串 | 数字 | 布尔值 | 对象 |
| undefined | “undefined” | **NaN** | False | Throw TypeError |
| null | “null” | **0** | False | Throw TypeError |
| true | “true” | **1** | True | New Boolean(true) |
| false | “false” | **0** | False | New Boolean(true) |
| “” | “” | **0** | **false** | New String(“”) |
| “1.2”非空，数字 | “1.2” | 1.2 | True | New String(“1.2”) |
| “one” | “one” | NaN | True | New String(“one”) |
| 0 | “0” | 0 | False | New Number(0) |
| -0 | “0” | -0 | False | New Number(-0) |
| NaN | “NaN” | NaN | False | New Number(NaN) |
| Infinity | “Infinity” |  | True | New Number(Infinity) |
| -Infinity | “-Infinity” |  | True | New Number(-Infinity) |
| Objects | | | | |
| {}任意对象 | **参考对象转换规则** | **参考对象转换规则** | True |  |
| []任意数组 | “” | 0 | True |  |
| [9]只包含一个数字类型的数组 | **“9”** | **9** | True |  |
| [‘a’]其他类型的数组 | 使用Join方法  “a,b,c” | NaN | True |  |
| Function (){}任意函数 | **参考函数转换规则** | NaN | True |  |

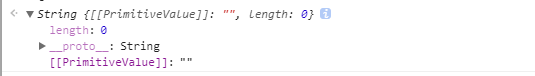
New Object(false)=New Boolean(false);这两个方法产生的结果一致，可以互用。



New Object(0) = new Number(0);



New Object(“”)=””;length=0的String类型的对象



New Object(null) = {}；转换成一个空对象



New Object(undefined) = {};转换成一个空对象



New Object([]) = [];还是一个空的数组

显式类型转换：

1. 通过类型封装函数Boolean(),Number(),String(),Object()
2. 通过运算符的类型转换
   1. +运算符： 数字+字符串=字符串
   2. +（一元）：+字符串=数字 +”123” = 123
   3. !（一元）：转换成布尔值并取反 !”123”=false !!”123” = true

Note:

1. 除了null和undefined之外的所有原始值和对象都有toString()方法，null和undefined是没有任何方法的原始值，如果试图调用方法，会抛出TypeError异常。

JavaScript对象转换成原始值：

1. 转换成布尔值：所有的对象都是true
   1. !!(New Object(false))=true
   2. !!(New Object(0))=true
   3. !!(New Object(“”))=true
   4. !!(New Object(null))=true
   5. !!(New Object(undefined))=true
2. toString() 和valueOf():所有的对象都继承了这两个方法。

常见类的toString()实现：

* 1. 数组：每个数组元素转换成字符串再用”,”链接起来
  2. 函数: 返回这个函数的源代码
  3. 日期: 可读的日期字符串格式，PST格式
  4. RegExp: 正则的可读形式

valueOf方法：如果有一个原始值能够表示这个对象，则转换成该原始值。大多数复合值的对象（包括数组，函数，正则）简单的返回对象本身。

日期类型的valueOf返回1970-1-1以来的毫秒表示方法

1. 对象到字符串的转换
   1. 如果对象有toString()方法（默认有，除非手工删除），则调用这个方法，如果这个方法返回一个原始值，js将这个值转换成字符串并返回。
   2. 如果没有toSting方法或者这个方法的返回值不是原始值，如果存在valueOf方法，并且返回原始值，js将这个原始值转换成字符串并返回。
   3. 否则，抛出TypeError
2. 对象到数字的转换
   1. 如果对象有valueOf方法，且这个方法返回一个原始值，则将这个对象转换成数字类型并返回
   2. 如果没有valueOf方法，或者返回的非原始值，则调用toString方法，同样如果返回值是原始值，转换成数字并返回
   3. 否则，抛出TypeError

举例：[]转换成数字：先调用valueOf返回非原始值，再调用toString，返回””，””再转换成数字0；

1. 对象到原始值的转换
   1. 对于二元运算符，+，==，!=，以及关系运算符没有明确对象类型需要转换成字符串函数数字，所以转换的规则稍微复杂。

对象分为两种:

* 一般对象（没有明确的toString和valueOf方法实现的），包括数组，函数，正则表达式
* 日期对象：日期类是JavaScript唯一预先定义的类，明确了toString和valueOf的实现方式。

一般对象的对于原始值的转换默认是指转换成数字。

日期对象的对于原始值转换默认是指转换成字符串。

* 1. 对于二元运算符-，默认都转换成数字。