**注意： string(“”) => number null=> number number(0,NaN,-1,Infinity) => boolean**

**JavaScript中原始数据类型的转换**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **目标类型** | | **null** | **undefined** | **string** | **number** | **boolean** |
| **string** | **隐式** | **"" + null** | **"" + undefined** | **所有的原生对象包括String / Number / Boolean都各自了实现toString，此方法通常和 String 方法返回一致** | **"" + 1.2** | **"" + false** |
| **显式** | **String(null)** | **String(undefined)** | **String(1.2)**  **String(1.7\*10^21)** | **String(false)**  **String(true)** |
| **值** | **“null”** | **“undefined”** | **“1.2” / “1.7e21”** | **“false”/“true”** |
| **number** | **隐式** | **+/-/\*/==/>=** | **-undefined** | **+“1.2”** | **parseInt/parseFloat**  **只针对 string, 传入其它类型都返回 NaN** | **false-0 / true-0** |
| **显式** | **Number(null)** | **Number(undefined)** | **Number(“”)**  **Number(“1.2”)**  **Number(“X2”)** | **Number(false)**  **Number(true)** |
| **值** | **0** | **NaN** | **0 / 1.2 / NaN** | **0 / 1** |
| **boolean** | **隐式** | **! / || / &&** | **if (undefined) !undefined**  **|| / &&** | **if (“”) / if (“test”)**  **!(“”) / !(“test”)** | **if (0) / if (1)**  **!0 / !1** | **常见的隐式转换：**  **if ()**  **while(..) / do..while**  **for (.. ; .. ; ..) //第二**  **? : //三元表达式**  **|| &&！//逻辑表达式** |
| **显式** | **Boolean(null)** | **Boolean(undefined)** | **Boolean(“”)**  **Boolean(“test”)** | **Boolean(0)/Boolean(NaN)**  **Boolean(-1)/Boolean(Infi)** |
| **值** | **false** | **false** | **false / true** | **false / false / true / true** |

**JavaScript中引用数据类型的转换**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **目标类型** | | **Object** | **Array** | **Function** | **Date** |
| **ToPrimitive操作** | | **valueOf //返回自身对象**  **toString//“[object Object]”** | **valueOf //继承,返回自身对象**  **toString //“1,2,3”** | **valueOf //继承,返回自身对象**  **toString //“function(){}”** | **toString //“Mon May 2.....”**  **valueOf //14954551049** |
| **string** | **求值** | **String({})** | **String([1,2,3])** | **String(function(){})** | **String(new Date{})** |
| **求原始值调用** | **先toString后valueOf** | **先toString后valueOf** | **先toString后valueOf** | **先toString后valueOf** |
| **结果** | **“[object Object]”** | **“1,2,3”** | **“function() {…}”** | **“Mon May 22 2017 19:38:13**  **GMT+0800 (中国标准时间)”** |
| **number** | **求值** | **Number({})** | **Number([]) Number([“t”])**  **Number([8]) Number([8,9])** | **Number(function(){})** | **Number(new Date())** |
| **求原始值调用** | **先valueOf后toString** | **先toString后valueOf** | **先valueOf后toString** | **先valueOf后toString** |
| **结果** | **NaN** | **0 / NaN / 8 / NaN** | **NaN** | **1495455104698** |
| **boolean** | **结果** | **引用对象的boolean 值都是 true** | | | |

**具体运算过程如下:**

1. **对于大部分的运算符来说，都有明确的作用类型，比如 “-” 和 “\*”，要求参与运算的数据都是number类型。那么针对这些运算符，直接通过ToValue操作获得number或者 ToString 操作获得string。但有些运算符例外，比如“+”和比较运算符“==”、“!=”、“>”、“<=”等，这些运算符同时适用于多种类型值的运算，实际参与计算的类型需要通过判断两边具体的原始值类型来决定。这种情况下就需要ToPrimitive操作来获得原始值。**
2. **对引用类型数据进行ToPrimitive操作可以获得一个原始值。****ToPrimitive操作与 ToValue 操作十分像似。会先尝试调用对象的 valueOf 方法，如果该方法返回一个原始值，则将原始值返回并结束 ToPromitive操作。如果valueOf 方法不存在或返回的不是一个原始值，则继续尝试调用 toString 方法，如果toString 方法也不存在或没有返回原始值则操作报错。另一种ToPrimitive操作是优先调用toString方法，这种操作只适用于Date 对象**
3. **通过类型转化后最终会得到两个原始值，如果这两个原始值的类型不一致，还需要把他们转换成一致的原始类型。比如： ”+” 运算符会优先转化成string，如果没有一个 string 值存在则转换成number。对于其他运算符（“==”、“!==”、“>”、”<=” 等）来说原始值会优先转化为 number。**

|  |  |
| --- | --- |
| **运算符** | **运算步骤** |
| **“==”** | **1. 引用类型转换为原始值**  **2. 两个原始值相同或者两个值分别是null和undefined，则返回true; 若存在单个undefined 或者 null 则返回 false**  **3. 两边的值，统一转换成数字比较** |
| **“+”** | **1. 引用类型转换为原始值**  **2. 如果有字符串，则优先转换为字符串并返回拼接结果**  **3. 两边的值，统一转换成数字比较** |
| **”-” / “\*” / “/”**  **“>” / ”<”**  **”>=” / “<=”** | **1. 引用类型转换为原始值**  **2. 两边的值，统一转换成数字比较** |
| **“===”** | **不进行类型转换** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **原生构造方法** | **不同参数的处理方式** | | | **目标类型** |
| **String()** | **基本类型** | **直接转换成目标类型 (参见基本数据类型的转换表)** | | **string** |
| **引用类型** | **先通过ToPrimitive操作获得原始值，再转换成目标类型** | |
| **Number()** | **基本类型** | **直接转换成目标类型 (参见基本数据类型的转换表)** | | **number / NaN / Infinity** |
| **引用类型** | **先通过ToPrimitive操作获得原始值，再转换成目标类型** | |
| **Boolean()** | **基本类型** | **直接转换成目标类型 (参见基本数据类型的转换表)** | | **boolean** |
| **引用类型** | **都返回 true** | |
| **Object()**  **等于**  **new Object()** | **基本类型** | **返回相应的包装类型** | | **对象** |
| **引用类型** | **返回自身对象** | |
| **Array()**  **等于**  **new Array()** | **基本类型** | **方式一：Array(length) length必须为数字，否则按第二种方式处理**  **方式二： Array(“arg1”, ”arg2”.....)** | | **array** |
| **引用类型** | **按方式二处理** | |
| **Date()** | **方法一：Date(year, month-1, date, hours, minites, seconds) //指定时间**  **方法二：Date(millisecond) //指定毫秒数 方法三：Date() //当前时间** | | | **返回Date对象的字符串类型** |
| **new Date()** | **由于 Date方法return字符串，所以用 new 调用该方法时，会忽略基本类型的返回值，并且返回新的Date对象** | | **Date.now()** | **获取当前时间毫秒数** |