数据类型

## 数据类型

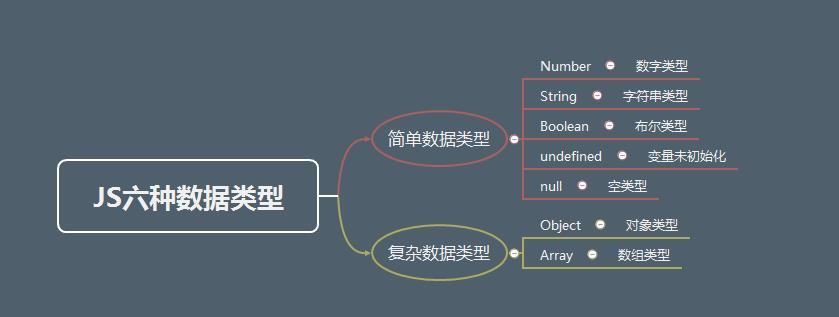
### 为什么会有数据类型？

* var a = 10;
* var b = '10';

思考：变量名a 和 变量名b 保存的数据是否一样？

### JS中常用的数据类型

* 程序在运行过程中会产生各种各样的临时数据, 为了方便数据的运算和操作, JavaScript对这些数据进行了分类, 提供了丰富的数据类型；
* 在JavaScript中一共有六种数据类型：



#### 数字类型（Number）

1: 最基本的数据类型  
2: 不区分整型数值和浮点型数值  
3: 能表示的最大值是±1.7976931348623157乘以10的308次方  
能表示的最小值是±5 乘以10的-324次方  
4：包含十六进制数据，以 0x开头 0到9之间的数字，a(A)-f(F)之间字母构成。 a-f对应的数字是10-15  
5： 八进制直接以数字0开始，有0-7之间的数字组成。

* 在JS中所有的数值都是Number类型(整数和小数)
* var num1= 123;
* var num2= 3.14;
* console.log(typeof num1); *// number*
* console.log(typeof num2); *// number*
* 由于内存的限制，ECMAScript 并不能保存世界上所有的数值
* *// 最大值：Number.MAX\_VALUE*
* console.log(Number.MAX\_VALUE); *// 1.7976931348623157e+308*
* *// 最小值：Number.MIN\_VALUE*
* console.log(Number.MIN\_VALUE); *// 5e-324*
* *// 无穷大：Infinity, 如果超过了最大值就会返回该值*
* console.log(Number.MAX\_VALUE + Number.MAX\_VALUE); *// Infinity*
* *// 无穷小：-Infinity, 如果超过了最小值就会返回该值*
* console.log(typeof Infinity); *// number*
* console.log(typeof -Infinity); *// number*

#### NaN 非法数字（Not A Number）

* JS中当对数值进行计算时没有结果返回，则返回NaN
* var str = NaN;
* console.log(typeof str ); *// number*
* Number类型注意点
* *// JS中整数的运算可以保证精确的结果*
* var counter = 120 + 20;
* console.log(counter ); *// 140*
* *// 在JS中浮点数的运算可能得到一个不精确的结果*
* var counter = 100.1 + 20.1;
* console.log(counter ); *// 120.200000000000003*

#### 字符串类型（string）

* 字符串由单引号或双引号括起
* 例如单独一个字母也被称为字符串（例如：‘a’）
* " '
* 使用字符串注意事项
* *// 正常的情况*
* var str1 = "hello";
* var str2 = `world`;
* console.log(typeof str1 ); *// string*
* console.log(typeof str2); *// string*
* *// 不能一单一双，必须成对出现*
* var str3 = `hello world"; // 错误
* // 相同引号不能嵌套，不同引号可以嵌套
* var str3 = "hello "world""; // 错误
* var str4 = `hello `world``; // 错误
* var str5 = "hello 'world'"; // 正确
* var str6 = `hello "world"`; // 正确

#### 布尔类型（Boolean）

* 布尔型也被称为逻辑值类型或者真假值类型
* 布尔型只能够取真（true）和假（false）两种数值, 也代表1和0，实际运算中true=1,false=0
* var bool1 = true;
* var bool2 = false;
* console.log(typeof bool1); *// boolean*
* console.log(typeof bool2); *// boolean*
* 虽然Boolean 类型的字面值只有两个，但 ECMAScript 中所有类型的值都有与这两个 Boolean 值等价的值

任何非零数值都是true, 包括正负无穷大, 只有0和NaN是false  
任何非空字符串都是true, 只有空字符串是false  
任何对象都是true, 只有null和undefined是false

var bool = Boolean(0);

console.log(bool ); *// false*

var bool = Boolean(1);

console.log(bool ); *// true*

var bool = Boolean(-1);

console.log(bool ); *// true*

var bool = Boolean("hello");

console.log(bool); *// true*

var bool = Boolean("");

console.log(bool ); *// false*

var bool = Boolean(undefined);

console.log(bool ); *// false*

var bool = Boolean(null);

console.log(bool8); *// bool*

#### Null和Undefined

* Undefined这是一种比较特殊的类型，表示变量未赋值，这种类型只有一种值就是undefined
* var num;
* console.log(num); *//结果是undefined*
* undefined是Undefined类型的字面量
  + 前者undefined和10, "abc"一样是一个常量
  + 后者Undefined和Number,Boolean一样是一个数据类型
* 需要注意的是typeof对没有初始化和没有声明的变量都会返回undefined。
* var str1 = undefined;
* console.log(typeof str1 ); *//结果是undefined*
* var str2;
* console.log(typeof str2); *//结果是undefined*

Null 类型是第二个只有一个值的数据类型，这个特殊的值是 null

* 从语义上看null表示的是一个空的对象。所以使用typeof检查null会返回一个Object
* var test1= null;
* console.log(typeof test1);

undefined值实际上是由null值衍生出来的，所以如果比较undefined和null是否相等，会返回true

var str1 = null;

var str2 = undefined;

console.log(str1 == str12);

console.log(str1=== str2);

### JS中如何查看数据类型?

typeof 操作符

* 会将检查的结果以字符串的形式返回给我们
* console.log(typeof 100); *// number*
* var name = "旋之华";
* console.log(typeof name ); *// string*

### 数据类型转换

#### 什么是数据类型转换?

将一个数据类型转换为其他的数据类型, 比如: 将String类型转换为Number类型, 将Number类型转换为Boolean类型... ..

#### toString()

将其它类型的数据转换为字符串类型

var num = 12306;

var str= num .toString();

console.log(str); *// 12306*

console.log(typeof str); *// string*

var num1 = true;

var str1 = num1.toString();

console.log(str1 ); *// true*

console.log(typeof str1 ); *// string*

注意:null和undefined没有toString()方法，如果强行调用,则会报错

var num2 = undefined;

var str2 = num2.toString(); *// 报错*

console.log(str2 );

var num3 = null;

var str3= num3.toString(); *// 报错*

console.log(res4);

var num4 = NaN;

var str4 = num4.toString();

console.log(str4); *// NaN*

console.log(typeof str4); *// String*

toString()不会影响到原变量，它会将转换的结果返回

var num5 = 10;

var str5 = str5.toString();

console.log(typeof num5 ); *// number*

console.log(typeof str5 ); *// string*

#### String()函数

有的时候, 某些值并没有toString()方法，这个时候可以使用String()。比如：undefined和null

* Number和Boolean的数据就是调用的toString()方法
* 对于null和undefined，就不会调用toString()方法, 因为没有这个方法, 会在内部生成一个新的字符串

#### +号链接

任何数据和 + 连接到一起都会转换为字符串  
, 其内部实现原理和String()函数一样

var num1 = 100;

var res1 = num1 + "";

console.log(res1); *// 100*

console.log(typeof res1); *// string*

var num2 = true;

var res2 = num2 + "";

console.log(res2); *// true*

console.log(typeof res2); *// string*

var num3 = undefined;

var res3 = num3 + "";

console.log(res3); *// undefined*

console.log(typeof res3); *// string*

var num4 = null;

var res4 = num4 + "";

console.log(res4); *// null*

console.log(typeof res4); *// string*

#### Number()函数（字符串 转 数字）

字符串 转 数字

* 如果是纯数字的字符串，则直接将其转换为数字
* var str1 = "666";
* var res1 = Number(str1);
* console.log(res1); *// 666*
* console.log(typeof res1); *// number*
* 如果字符串中有非数字的内容，则转换为NaN
* var str2 = "it666";
* var res2 = Number(str2);
* console.log(res2); *// NaN*
* 如果字符串是一个空串或者是一个全是空格的字符串，则转换为0
* var str3 = "";
* var res3 = Number(str3);
* console.log(res3); *// 0*
* var str4 = " ";
* var res4 = Number(str4);
* console.log(res4); *// 0*

undefined 转 数字

var str6 = undefined;

var res8 = Number(str6);

console.log(res8); *// NaN*

null 转 数字

var str5 = null;

var res7 = Number(str5);

console.log(res7); *// 0*

布尔 转 数字

* true 转成 1, false 转成 0
* var bool1 = true;
* var res5 = Number(bool1);
* console.log(res5); *// 1*
* var bool2 = false;
* var res6 = Number(bool2);
* console.log(res6); *// 0*

### parseInt()函数和parseFloat()函数

 Number()函数中无论混合字符串是否存在有效整数都会返回NaN, 利用parseInt()/parseFloat()可以提取字符串中的有效数值

#### parseInt()和parseFloat()的区别

 parseInt()和parseFloat()的区别是前者只能提取整数,后者可以提取小数

#### parseInt()提取字符串中的整数

* 从第一位有效数字开始, 直到遇到无效数字  
  如果第一位不是有效数字, 什么都提取不到, 会返回NaN
* 第一个参数是要转换的字符串，第二个参数是要转换的进制

var str7 = "300px";

var res9 = parseInt(str7);

console.log(res9); *// 300*

var str8 = "300px250";

var res10 = parseInt(str8);

console.log(res10); *// 300*

console.log(parseInt("abc123")); *//返回NaN，如果第一个字符不是数字或者符号就返回NaN*

console.log(parseInt("")); *//空字符串返回NaN，Number("")返回0*

#### parseFloat提取字符串中的小数

* 会解析第一个. 遇到第二个.或者非数字结束
* 如果第一位不是有效数字, 什么都提取不到
* 不支持第二个参数，只能解析10进制数
* 如果解析的内容里只有整数，解析成整数
* var str9 = "20.5px";
* var res11 = parseInt(str9);
* console.log(res11); *// 20*
* var str10 = "20.5.5.5px";
* var res12 = parseFloat(str10);
* console.log(res12); *// 20.5*

对非String使用parseInt()或parseFloat(), 会先将其转换为String然后再操作

var str11 = true;

var res13 = parseInt(str11); *// 这里相当于parseInt("true");*

console.log(res13); *// NaN*

var res14 = Number(str11);

console.log(res14); *// 1*