**第二章JS数据类型、值与类型转换**

**JS（ES5）数据类型（6种）及其划分（2类）**

- 基本（原始）类型（Number、String、Boolean、Null、Undefined）（定义为基本类型的函数局部变量分配在栈区）  
- 引用（对象）类型（Object（Array、Function、Date、Error等））（定义为引用类型的变量，其引用分配在栈区或堆区，引用的对象分配在堆区）

数据类型检测方法（typeof、instanceof）

**基本类型与引用类型的区别**

内存分配方式不同

堆区与栈区、存值与存地址、影响变量的生命周期（自动清除、垃圾回收）  
 函数内定义的基本数据类型的临时变量分配在栈区  
 引用数据类型的变量的引用（地址）存储在栈区或堆区，被引用（指向）的对象存储在堆区

赋值时不同

赋值、赋引用（地址）、深拷贝与浅拷贝

判等时的不同

值类型是判断变量的值是否相等（值比较）  
 引用类型是判断所指向的内存空间是否相同（引用比较）

函数参数传递时的不同

按值传递(call by value)  
 按引用传递(call by reference)

真正决定这几种不同的是数据类型，而不是内存分配方式，内存分配方式决定的是变量的生命周期

**Number类型的值**

整数与浮点数  
 NaN、Infinity、-Infinity、+0、-0

**String类型的值**

空字符、字符和字符串、转义字符

**Boolean类型的值**

- true、false

**Null与Undefined**

null、undefined

**简单对象** 例：var obj = {name：“Jack”，age：20};

**数组**- 例：var arr = [1,2,true,“Hi”];

**函数对象**- 例：var foo = function(x,y){...};//函数也是对象（可执行的对象），也有属性和方法

**正则对象** 例：var reg = /^a+b+$/;

**包装对象**- 数字、布尔、字符串等基本数据类型都有对应的包装对象类型，可以将其包装成对象  
- 例：new Number(20)； new String('SomeStr');//装箱  
- 存储或读取基本类型（字符串、数字、布尔）值的属性时，会创建临时包装对象  
- 例: console.log('Hello，World'.length);  
- 基本类型其属性不能被改变、添加或删除（原始值不可变性）

**临时对象在使用之后立即释放**- 例：var str=”test”;  
 str.p = 4；//设置临时对象属性  
 var t = str.p; // 临时对象已释放，再输出t时为undefined

**数据类型转换**

**其他类型转换为Boolean类型**

* 图片包含 屏幕截图

  自动生成的说明 转换方式  
  - Boolean（）、value？true：false、！！value

**其他类型转换为Number类型**

图片包含 屏幕截图

自动生成的说明

其他类型转换为String类型

图片包含 屏幕截图

自动生成的说明

**隐式类型转换**- 使用关系运算符时的转换（==、>、<、引用类型和基本类型比较时）  
- 使用算数运算符时的转换（'img'+ 3 + '.jpg'; “25”-0;）  
- 使用逻辑运算符时的转换（ !!0; ）  
- 执行流程语句时的转换（if(obj){...}）

**显式类型转换（使代码更清晰）**- Boolean（）、Number（）、String（）、Object（）  
- 数转为字符串（toString()、toFixed()、toPrecision()、toExponential()）  
- 字符串转为数字（parseInt()、parseFloat()）  
- 对象转换为原始值（toString()、valueOf()）