数据类型存储和转换

- 数据类型
  - ES5数据类型（6种）及其划分（2类）
    - 基本（原始）数据类型
      - Number、String、Boolean、Null、Undefined
    - 引用（对象）数据类型
      - Object（Array、Function、Datedeng）
  - typeof：判断数据类型
    - typeof操作符，返回一个字符串，表示未经计算的操作数的类型
    - Null 返回object
    - 函数对象，返回function；其他对象返回object
  - 数据、变量和类型
    - 变量没有类型；数据值有类型
    - 变量可以随时持有任何类型的值
    - 在对变量执行typeof操作时，得到的结果并不是该变量的类型，而是该变量持有的值的类型
- 数据类型存储
  - 变量与内存
    - 变量声明
      - 使用方便的标识符，用于引用计算机内存地址；变量声明指向一块内存，用于保存数据
    - 变量赋值
      - 向变量指向的内存空间中存放数据
    - 系统划分两种不同的内存空间
      - 栈内存
      - 堆内存
  - 堆栈内存
    - 栈内存
      - 存储的值大小固定；由系统自动分配内存空间；空间小，运行效率高
    - 堆内存
      - 存储的值大小不定，可动态调整；有程序员通过代码进行分配；空间大，运行效率相对较低
  - 基本类型的存储
    - 基本类型的变量是存放在栈区的
    - 基本类型的值是不可变的
  - 引用类型的存储
    - 引用类型的值是同时保存在栈内存和堆内存中的对象 ，栈区内存堆内存地址
    - 引用类型的值可以改变
  - 基本类型与引用类型的区别
    - 访问机制不同
      - 基本数据类型的值直接访问
      - 引用类型的值通过引用访问，不能直接访问。
    - 复制变量不同
      - 基本类型，相互独立互不影响
      - 引用类型，改变指向
    - 比较变量不同
      - 基本数据类型判断变量的值是否相等（值比较）
      - 引用数据类型判断所指向的内存空间（地址）是否相同（引用比较）
    - 参数传递不同
      - ECMAScript中所有函数的参数都是按值来传递的
      - 引用类型值：把对象的引用（地址）值传递给参数，参数和对象都指向同一个对象，相互影响
      - 基本类型值：把变量里的数据值传递给参数，之后参数和变量互不影响
- 类型转换
  - 类型转换
    - 将值从一种类型转换为另一种类型
    - 隐式类型转换：通常是某些操作的副作用，不易看出
    - 显示类型转换：可以在代码中明显看出（强制类型转换）
  - 转换为Number类型规则
    - Undefined--NaN;空字符--0
    - parseInt（）、parseFloat（）、Number（）
    - NaN：表示一个没有意义不正确的数值，与自身不相等
      - isNaN（）检测是否为NaN值
  - 转换为String类型规则
    - String（）
  - 转换为Boolean类型规则
    - undefined、null、0、NaN、 ‘’ 、都为False
    - Boolean（）