# Day03

#### IF判断

* **True 转换为 1**
* **False 转换为 0**
* **0 转换为 false**
* **非0 转换为 true**

***If条件都要转换成Boolean值***

* ***5大假值***
* **0**
* **空字符串**
* **Null**
* **Undefined**
* **False**
* ***If判断中不要去做小数的比较运算***
* ***小数运算有可能产生高精度的数***
* ***0.1 + 0.2 = 0.3 0000 0000 0000 0004***

***数值过大浏览器无法显示 ---->无限大 Infinity***

#### 对象

* 对象
* 什么是对象
* JavaScript中万物皆为对象
* 原始数据类型 严格上来说，属于对象，但是没有对象的特点
* 引用数据类型 有对象的特点
* 对象的创建方式 3种
* {} 对象字面量（对象直接量）
* new 构造函数new Object new系统构造函数 new出来以后是一个空对象 工作中很少使用
* new 自定义构造函数
* Object create() ES5中新增的
* () 函数执行符号
* 构造函数也属于函数
* 构造就是专门去出对象
* 人为规范（开发者制定出来的一套开发习惯）：
* 1.构造函数只能被new执行
* 2.写法：构造函数必须使用大驼峰命名（首字母大写）
* 对象的增删查改
* 增查改：对象打点
* 删除：delete bonly[k];
* 面试题：
* 函数在被new关键字执行以后发生了什么样的过程？
* 1.内部隐式的创建了一个this对象
* 2.在最后返回这个this对象
* 3.return值可以修改，但是只能去返回一个引用类型的值，如果改成原始类型的值，是没有任何反应的
* 在JavaScript中有几种输出方式
* 1.document.write
* 2.alert
* 3.console.log
* 前面的两种方式一定会调用toString(),无法查看类型,其次原型链的就近原则会导致toString被篡改
* 任何对象的最终原型都为object
* 正常来说是对的，但是仅有一种方法Object.create(null)

#### 包装类

* 原始数据类型 严格上来说，属于对象，但是没有对象的特点
* 原始类型添加了属性就马上销毁
* 引用数据类型 有对象的特点
* 构造函数出来的内容一定是一个引用类型
* 包装类 把原始类型进行一个包装 包装成一个引用类型 就有引用类型的特性

#### 原型、原型链

* **作用域链找不到就报错**
* **原型链找不到就为undefined**
* **作用域也是一个对象，但是这个对象看不到，只能通过执行的场景进行模拟**
* **原型也是一个对象，这个对象可以看到的对象**
* **原型是定义一个对象的公共祖先**
* **JavaScript基于对象，对象就是通过构造函数来的**
* **Prototype 构造函数的属性，负责定义**
* **\_\_proto\_\_ 对象的属性，负责查看（可以修改，千万不能用\_\_proto\_\_修改）**
* **自定义的构造函数可以设置原型对象**
* **系统的构造函数因为有原型对象，退一步，在它的基础上进行添加**
* 所有的dom获取元素出来全部都是类数组
* 类数组（伪数组）的本质是一个对象
* 类数组必须含有length属性
* 类数组调用了数组的push方法以后，可以沿着length属性进行扩展