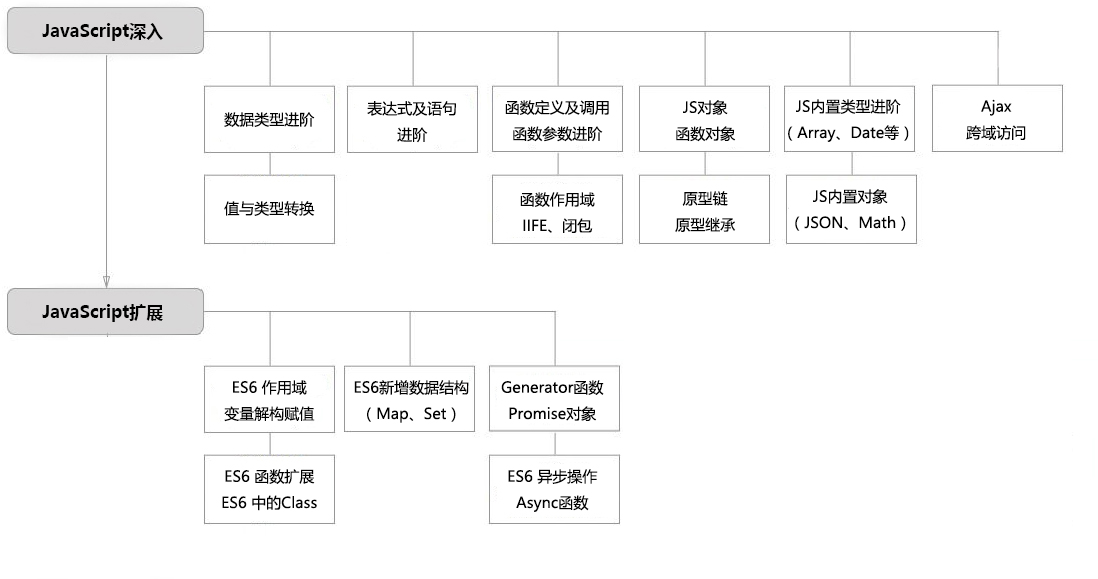
**Array.shift（）删除数组的第一个元素**

**JS笔记整理**

**第一章：JS发展特征及语言特点**

一、ES6=ES2015(年份比版本号小1)，15年以后每年更新一个版本

二、JS课程框架



三、JS的组成

1、JS只能在浏览器中执行 ×

DOM和BOM 是宿主对象，可变；但EC是核心语言，不可变。语言能够做什么取决于运行平台（宿主环境），当宿主环境改变后可未完成不同功能。

客户端JS= ES（核心语言）+BOM+DOM，能够做什么取决于浏览器，（BOM.DOM 是宿主对象，可换）

服务端（JSNode.js）= ES （核心）+ fs（操作文件）+ ……

2、JS在浏览器如何运行

JS是高级语言，及其无法识别，要通过浏览器内部的JS引擎（chrome是V8）转换成计算机能够识别的。

JS引擎（解释执行JS语言）

高级语言 计算机能识别的

Chrome（V8）

**四、JS语言特点**

（1）直译式脚本语言

在宿主（Node.Browser）中解释执行，边解释边执行，速度慢，（同传）

C语言： 编译执行，一次性解释完再运行，速度快，（交传）

（2）弱类型、动态类型语言→JS不适合大型项目的开发，易出错

弱类型：变量声明不用指定数据类型

动态类型：数据类型动态可修改

（3）ES5没有块作用域

{ } 外仍能访问

（4）基于原型链的继承方式

**第二章：数据存储和转换**

**一、数据类型**： JS 中变量是没有数据类型的，只有值才有，变量可以有任意类型的值

基本/原始数据类型（5种）：Number，String，Boolean, Null ,Undefined

引用/对象数据类型（Object）：Array，Function，Date ……

判断数据类型：通过typeof（a）/（typeof a）判断，typeof判断的是计算之前的数据类型，返回的是一个字符串，Null和引用数据类型返回Object，函数返回Function 但是无法区分null、date、数组到底是那个

Eg：var a = 1;

var b = ”hello”;

a+b;

typeof（a）； //Number

function student (name, age) {

this.name = name ;

this.age = age ;

}

Var stu = student(“lucy”, 20) ;

Console.log(type of stu) ; //打印出来的是stu是student的返回值，没有return是undefined



**二、数据的存储**

Var声明变量的同时开辟内存空间，赋值时将变量与开辟的内存空间联系起来。

堆栈内存≠C语言

每个值大小不定，随时可以修改。

堆内存：大小由代码分配，大小可控

空间大，运行效率低

每个值大小固定，不可改

 栈内存 大小系统自动分配

空间小，运行效率高

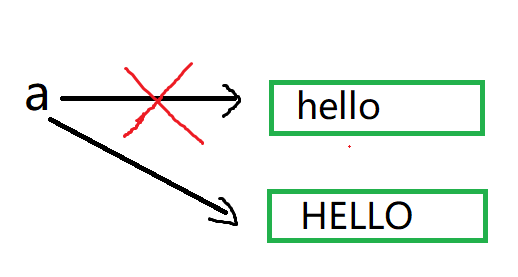
1、基本数据类型存在栈区：

其值不可变

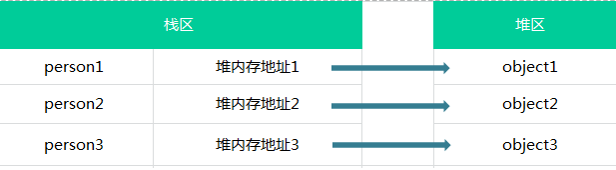
Eg： var a = “hello” ;

1. toUpperCase() ;

a的值还是hello不变，toUpperCase有返回值，在栈中开辟新的的空间存HELLO



2、引用数据类型同时存在堆区和栈区：

栈区：变量名+地址

堆区：值

通过修改堆区的值来修改值

3、基本数据类型和引用数据类型的区别

（1）访问机制不同

基本数据类型的值直接访问；引用数据类型通过地址引用访问，不能直接访问（地址→数据）

（2）变量复制不同

基本数据类型相互独立，不受影响；引用数据类型复制地址。

（3）变量比较不同

基本数据类型比值，引用数据类型比地址

（4）参数传递不同（ES函数传参都是值传递）

基本数据类型直接把值传给参数，参数和变量互不影响；引用数据类型第地址传给参数，参数和对象指向相同内存空间，操作相互影响

**三、数据类型的转换**

隐式类型转换：var a = 2 ; var b = “3” ; a + b =23(a : Number→String)

显示类型转换：强制类型转换

1、→Number（parseInt()，parseFloat()，Number()）

●Undefined：NaN（Not a Number）

●Null：0

●Boolean：true→1；false→0

空字符串：0

●字符串 纯数字：“123”→123

非空 非纯数字：“1a2b”→NaN

有多个小数点的只保留第一个，后面的省略：“13.456.35”→13.456

●Undefined：NaN

NaN： 不正确的，没有意义的数值。

Typeof NaN //Number

检测是否为NaN：isNaN（“12a”） // true

NaN == NaN //false

NaN和任何数运算结果都是NaN

2、→String（String()）

Null：“null”

Undefined：“undefined”

Number：123→“123”

+左右两侧只要一侧有字符串则为拼接运算符

3、→Boolean

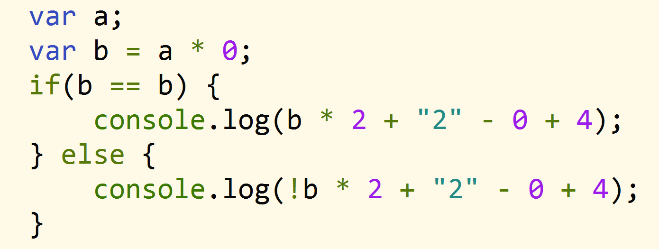
Undefined：false

Null：false

0：false

数字 NaN：false

其他的都是：true

Eg：

//26

**第五章：函数定义和调用形式**

**一．函数的定义**

**函数：**可以通过外部代码调用的一个“子程序”，值可以传递给一个函数，函数将返回一个值。

1、函数定义方法：

（1）函数声明：function fun（）{}

（2）函数表达式：var fun = function fun（）{ }，function前有东西的是函数表达式

（3）Function构造函数实例化：var max = Function（“a“，”b”，“return a>b ? a : b;”）

Function构造函数创建一个匿名函数，最后一个参数为函数体

2、函数定义三要素：函数名，函数的参数，函数的返回值

3、单独的匿名函数无法运行，必须赋值给变量或立即执行。

具名函数遇到错误是堆栈会显示函数名，易于寻找错误

4、代理函数名：可有可无，其作用域只在函数体内部，作用域外不能访问。

Eg: var f1 = function f2（）{} //f2就是代理函数名

f2 ( )； // f2 is not defined

f1 ( ) == f2 ( ) //true

5、name属性：返回函数名，用new Function（）的函数名为“anonymous”

f1.name //f1

6、length属性，返回形参个数

Function f2（a, b, c）{}

F2.length; //3

**二、arguments对象**

代表传入函数的实参；是函数中的局部变量；不能显示创建，只有在函数调用是才会创建；是个类数组对象本质是Object，和数组一样，有length、index属性