一、关于学习

二、今日目标

1. 提高自己的学习积极性，正确对待学习
2. 了解JavaScript
3. 学会使用浏览器检查JS代码的执行情况
4. 能够说出ES5中的6种数据类型
5. 能够理解typeof和instanceof操作符的含义和作用（判断数据类型）
6. 能够理解null和undefined的区别
7. 能够理解什么是原始类型和引用类型
8. 能够定义JS中的数组
9. 能够定义JS中的直接量形式的对象
10. 能够用代码写出JS函数的定义形式和调用形式
11. 能够说出函数的形参和实参的区别
12. 能够理解匿名函数立即调用的定义形式和调用形式

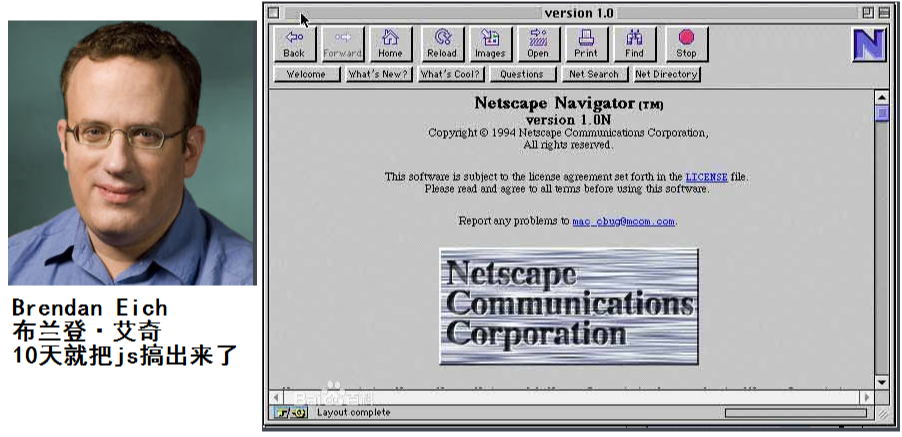
三、介绍JavaScript

1、介绍

JavaScript一种轻量级的脚本语言，也是一种嵌入式（embedded）语言，是一种 **对象模型** 语言，简称 **JS** ；JavaScript 的核心语法部分相当精简，也就是语言本身，只包括两个部分：基本的语法构造（比如操作符、控制结构、语句）和标准库（就是一系列具有各种功能的对象比如Array、Date、Math等）。想要实现其他复杂的操作和效果，都要依靠宿主环境提供API，目前，已经嵌入 JavaScript 的宿主环境有多种，最常见的环境就是 浏览器，另外还有服务器环境(操作系统)；

JavaScript被认为是客户端“脚本”，它不常叫“语言”，言外之意是比较简单，但是深入了解之后，你会发现，简单的外表下，蕴藏着丰富的内涵。

2、历史



1994年Netscape（网景）公司推出免费版本浏览器 Netscape Navigator(网景浏览器1.0)

1995年微软公司发布 Internet Explorer 1.0。

1995年网景公司为适应市场变化，需要开发一门专门在浏览器运行的脚本语言，这个任务交给了布兰登，为了应付公司安排的任务，

他只用10天时间就提交了工作，并将这门语言命名为 LiveScript;

后来为了蹭sun公司java的热度,与sun公司合作，将其临时改名为“JavaScript”;

1996年8月，微软模仿JavaScript开发了一种相近的语言，取名为JScript，首先内置于IE 3.0。

1997年7月，ECMA组织（欧洲计算机制造商协会）发布ECMAScript 1.0版；

此后，明争暗斗不断，1998年6月，ECMAScript 2.0版发布，1999年12月，ECMAScript 3.0版发布；

2007年10月，ECMAScript 4.0版草案发布，2008年7月中止ECMAScript 4.0的开发，并发布3.1版本；

会后不久，ECMAScript 3.1就改名为ECMAScript 5。

2011年6月，ECMAscript 5.1版发布，现在使用最为广泛的版本 版发布，现在使用最为广泛的版本；

2015年6月，ECMAScript 6正式发布，并且更名为“ECMAScript 2015”；

随后，ECMA组织决定，每年发布一个升级版本，以年号来代替版本号，如：ECMAScript 2016、ECMAScript 2017；

**另外**：

1996年，样式表标准CSS第一版发布；

1997年，DOM模式第一版正式应用，目前的通用版本是DOM3，下一代版本DOM 4正在拟定中。

1999年，IE5部署了XMLHttpRequest接口，允许JavaScript发出HTTP请求；

2001年，提出了JSON格式，用于取代XML格式。

2002年，Mozilla项目发布第一版Firefox。

2003年，苹果公司发布了Safari浏览器的第一版。

2006年，jQuery函数库诞生

2007年，Webkit引擎在iPhone手机中得到部署；

2008年，为Chrome浏览器而开发的V8编译器(解析引擎)诞生；

2009年，基于V8解析引擎的Node.js项目诞生,迎来前后端JS的霸权时代；

2009年，Google发布Chrome OS

2009年，Google发布Angular框架；

2013年，Mozilla基金会发布手机操作系统Firefox OS，该操作系统的整个用户界面都使用JavaScript；

2013年5月，Facebook发布UI框架库React；

2014年，尤雨溪发布开源前端开发库Vue.js；

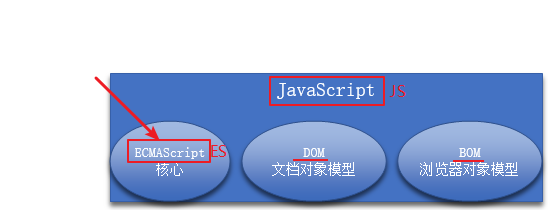
2015年3月，Facebook公司发布了 React Native项目；

3、ECMAScript和JavaScript

ECMA是一个组织，是欧洲计算机协会。是它制定了JavaScript的标准。

ECMAScript简称ES，比如ES5就表示ECMAScript5版本，ES6就表示ECMAScript6版本。JavaScript简称JS，JS的版本随ECMAScript的版本变化而变化。

ECMAScript只是JavaScript的核心，我们现在学习的JavaScript包括JS的核心ECMAScript，并且还包含DOM和BOM。



4、JS的强大之处

① 数据验证（表单验证）

② 网页特效（目前大多数的网页特效都是通过JS编写）

③ 编写网页小游戏

④ 服务器端编程，数据交互（Ajax、Node.js）

<http://impress.github.io/impress.js/>

<http://naotu.baidu.com/>

<http://echarts.baidu.com/index.html>

<https://blockly.uieee.com/>

四、如何学习JS

1、IDE

古人云，“工欲善其事，必先利其器”、“磨刀不误砍柴工”、“好的编辑器能提高你的开发效率”等。

* zendstudio 比较庞大
* Visio studio 微软的产品
* netbeans 默认就可以装中文版
* phpstrom 老师用的就是它
* webstrom
* HBuilder
* Sublime Text
* 等等。

2、调试工具

检测JavaScript执行结果最好的工具就是浏览器（推荐Google浏览器，火狐浏览器）。

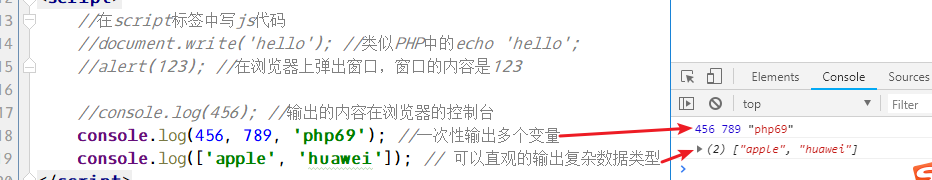
打开浏览器的控制台的方法：按F12

谷歌浏览器：右键🡪检查

火狐浏览器：右键🡪查看元素

*//在script标签中写js代码  
//document.write('hello'); //类似PHP中的echo 'hello';  
//alert(123); //在浏览器上弹出窗口，窗口的内容是123  
  
//console.log(456); //输出的内容在浏览器的控制台***console**.log(456, 789, **'php69'**); *//一次性输出多个变量***console**.log([**'apple'**, **'huawei'**]); *// 可以直观的输出复杂数据类型*

效果：



五、语言基础

1、代码位置

JS代码要写到一对script标签中。

script标签要放到哪里？一句话，哪都行，只要在html页面中就可以。



除此以外，还可以单独创建一个后缀为js的文件，在js文件中写js。需要html文件引入js文件：

**浏览器运行的是html文件，而不能直接是js文件**。

2、输出

介绍3种输出方式：

1、document.write(输出的内容); //这种输出的内容会显示在浏览器页面上

2、alert(输出的内容); //这种方法输出的内容会以提示框的形式显示

3、console.log(输出的内容); //输出的内容会显示在浏览器的控制台。推荐使用。

3、语句与表达式

JavaScript程序的执行单位为行（line），也就是一行一行地执行。一般情况下，每一行就是一个语句。

语句（statement）是为了完成某种任务而进行的操作，比如下面就是一行赋值语句：

var a = 1 + 3;

这条语句先用var命令，声明了变量a，然后将1 + 3的运算结果赋值给变量a。

1 + 3叫做表达式（expression），指一个为了得到返回值的计算式。

语句和表达式的区别在于，前者主要为了进行某种操作，一般情况下不需要返回值；后者则是为了得到返回值，一定会返回一个值。

凡是JavaScript语言中预期为值的地方，都可以使用表达式。比如，赋值语句的等号右边，预期是一个值，因此可以放置各种表达式。一条语句可以包含多个表达式。

语句以分号结尾，一个分号就表示一个语句结束。多个语句可以写在一行内。

var a = 1 + 3 ; var b = 'abc';

分号前面可以没有任何内容，JavaScript引擎将其视为空语句。

;;;

上面的代码就表示3个空语句。

JS语言中，一条语句结束，可以不用写分号，直接用回车即可。但是建议大家每条语句后面都加分号。

4、注释

单行注释：//这是一行注释

多行注释：/\*这是一段注释\*/

六、变量

1、什么是变量

* 什么是变量

变量是计算机内存中存储数据的标识符，根据变量名称可以获取到内存中存储的数据

* 为什么要使用变量

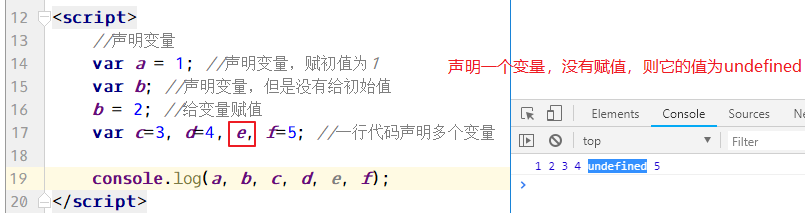
使用变量可以方便的获取或者修改内存中的数据

2、 如何使用变量

* var声明变量

*//声明变量***var *a*** = 1; *//声明变量，赋初值为1*

* 变量的赋值
* 同时声明多个变量
* 同时声明多个变量并赋值

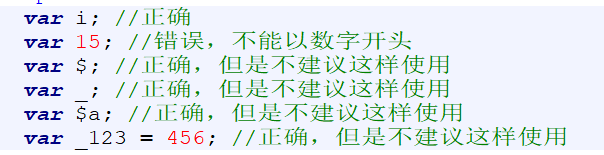
声明变量的时候，使用var，声明之后，再次使用变量的时候，就不要在写var了，直接就变量名即可。

3、变量在内存中的形式



4、 变量的命名规则和规范

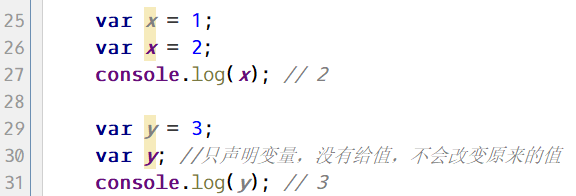
* 规则 - 必须遵守的，不遵守会报错
  + 由字母、数字、下划线、$符号组成，且不能以数字开头
  + 不能是关键字和保留字，例如：for、while、this、name。
  + 区分大小写
* 规范 - 建议遵守的，不遵守不会报错
  + 变量名必须有意义
  + 遵守驼峰命名法。(首字母小写，后面单词的首字母需要大写。例如：userName、userPassword)
* 下面哪些变量名不合法



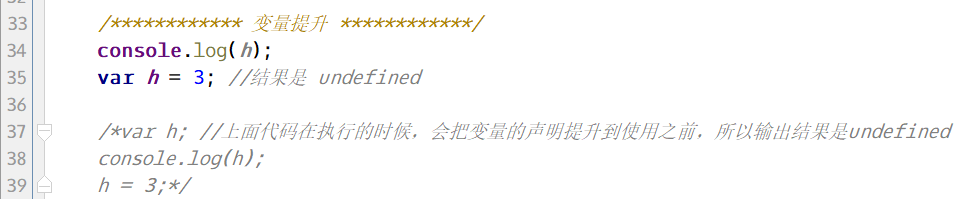
5、只声明变量没有赋初值



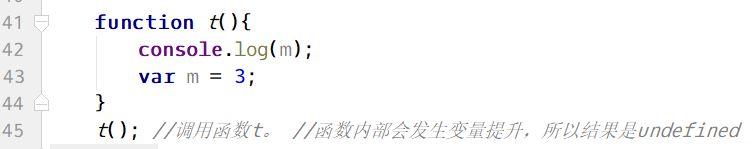
6、重新声明变量



7、变量提升（hoisting）



在同一个作用域中，都会发生变量提升，比如函数中。



七、常量

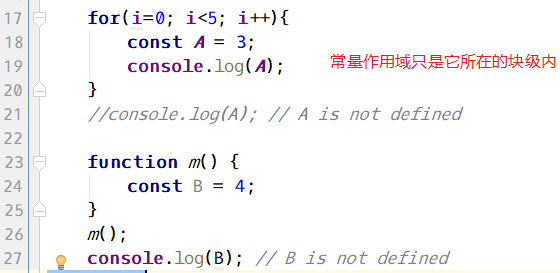
1、定义常量

JS中定义常量，只有const语法。

*//定义常量***const *A*** = 123;  
**console**.log(***A***); // 123

2、常量作用域

常量的作用域，是常量所在的块。（块可以理解为大括号）

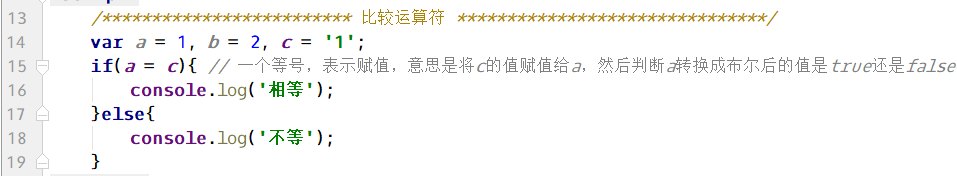


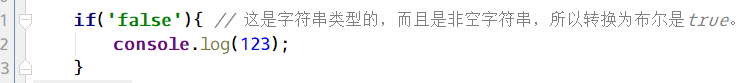
八、运算符

1、赋值运算符(Assignment operators)

赋值运算符是一个 “=”。表示将“=”后面的内容赋值给左边的内容。

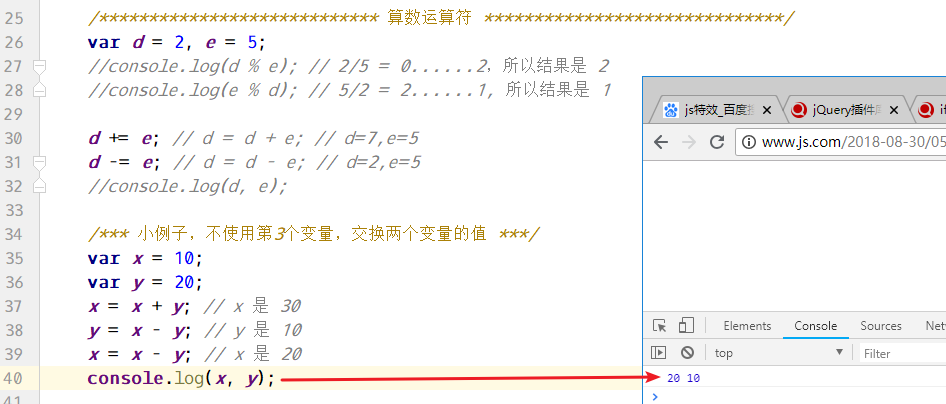
2、比较运算符(Comparison operators)





3、算数运算符(Arithmetic operators)

加减乘除取余



4、逻辑运算符(Logical operators)

或者、并且、非

*/\*if(0 && false){  
 console.log(123);  
}else{  
 console.log(456);  
}\*/***var *a*** = 1, ***b*** = 2, ***c*** = 0, ***d*** = **false**;  
**var *e*** = ***a***||***b***; *// e = a;  
//console.log(a||b); // 1  
//console.log(a||c); // 1  
//console.log(c||a); // 1  
//console.log(c||d); // false  
// || 关系，先看左边，左边为true，则直接等于左边；左边为false，直接等于右边***console**.log(***a***&&***b***); *// 2***console**.log(***a***&&***c***); *// 0***console**.log(***c***&&***a***); *// 0***console**.log(***c***&&***d***); *// 0  
// && 关系，先看左边，左边为true，直接等于右边；左边为false，直接等于左边。***console**.log(10||20); *// 10  
/\*var x = 10&&20;  
console.log(x||30); // 20\*/  
//console.log(0&&5); // 0*

5、字符串运算符(String operators)

PHP中用 点（.） 连接两个字符串。

JS中用 加号(+) 连接两个字符串。

如果使用+的是两个数字，则表示加法运算；如果使用+的有一方是字符串，则表示字符串连接。

**console**.log(2+3); *// 5，表示加法运算，因为参与运算的两个值都是数值型***console**.log(2+**'hello'**); *// 2hello，表示字符串相连***console**.log(**'hello '** + **'world'**); *//hello world***var *x*** = 35;  
**console**.log(**'hahaha'** + ***x***); *// hahaha35*

6、条件（三元）运算符(Conditional operator)

元，表示参与运算的参数个数。三元意思就是参与运算的有三个值。

var a = (b>c) ? x : y;

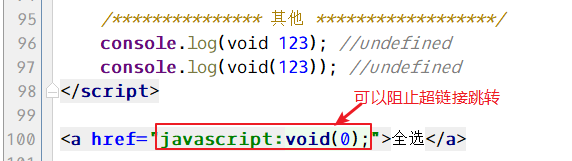
上述代码整体意思是给a赋值，值可能是x，也可能是y。关键取决于b是否大于c。如果b>c，则将x赋值给a；如果b不大于c，则将y赋值给a。

7、一元运算符(Unary operators)

**var *a*** = 2, ***b*** = 3;  
**var *c*** = ***a***++; *// var c = a; a = a+1***var *d*** = ++***b***; *// b = b+1; var d=b;***console**.log(***a***, ***b*** , ***c***, ***d***); *// a=3, b=4, c=2, d=4*

8、其他

- void 它是一个函数或是一个语言结构，返回值总是无效的undefined。经常用它来使得超链接变得无效。



- typeof它是一个函数或是一个语言结构。typeof(a)、 typeof a; 用于判断变量的数据类型。

**console**.log(typeof(123)); *// number*

**console**.log(typeof 123); *// number***console**.log(typeof 3.14); *// number***console**.log(typeof **'hello world'**);*// string***console**.log(typeof **true**); *// boolean*

- in -- 用于判断下标是否存在数组中，或判断对象中是否有哪个成员

- instanceof -- 判断对象的原型

九、数组

1、声明数组

在JS中，数组也是一种特殊的对象。

三种声明方式：

①、var arr = ['apple', 'pear']; // 推荐使用

②、var arr = new Array('apple', 'pear'); // 一般推荐

③、var arr = new Array(3); //表示数组中有三个单元

声明数组的数据，数组的单元不能指定下标，只能使用默认的数字。



2、获取数组的元素

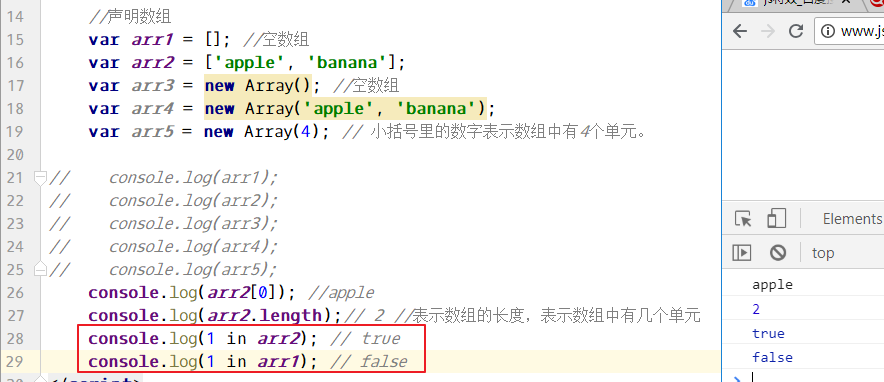
使用“数组[下标]”可以获取到数组中的值。JS中数组的下标一定是数字类型的。



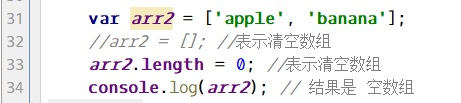
3、判断下标是否存在

使用 in 来判断下标是否存在。

语法： 下标 in 数组 如果存在返回true，不存在返回false。



4、清空数组



十、对象

1、创建对象

JS中要得到一个对象有很多种方式，今天学习的是直接量语法得到对象。

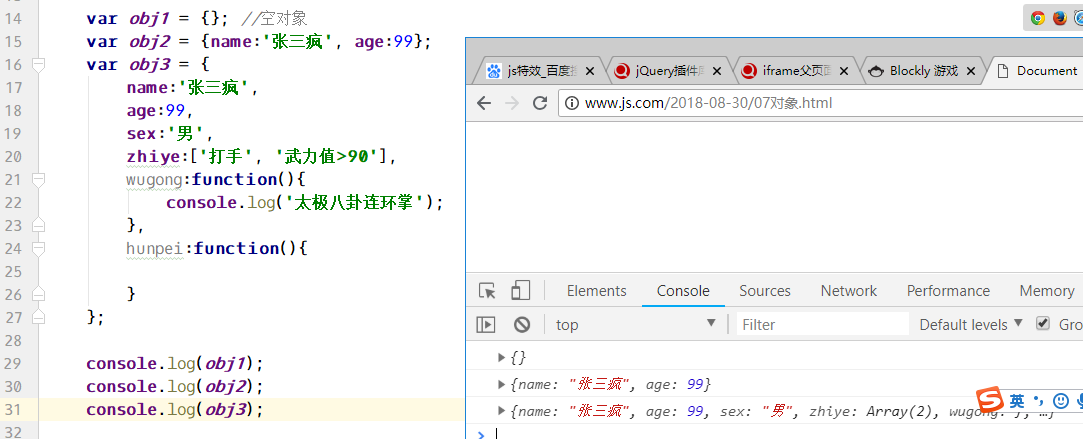
语法：

var obj = {}; //空对象  
var obj = {成员名:值, 成员名:值};   
var obj = {

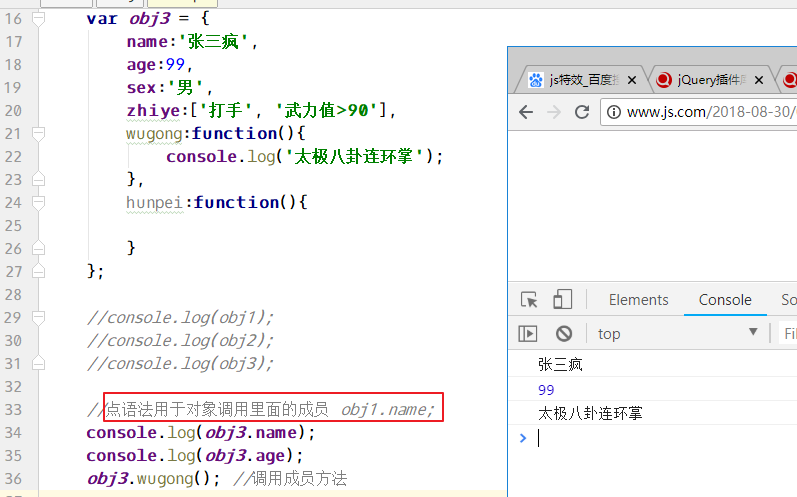
成员名:值,  
 成员名:值

};

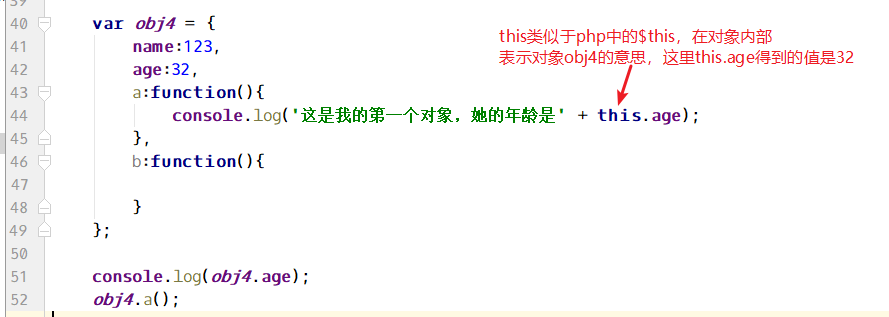
注释：值可以是数字型、字符串型等基本数据类型，也可以是数组或其他对象，还可以是函数。



2、获取对象的成员

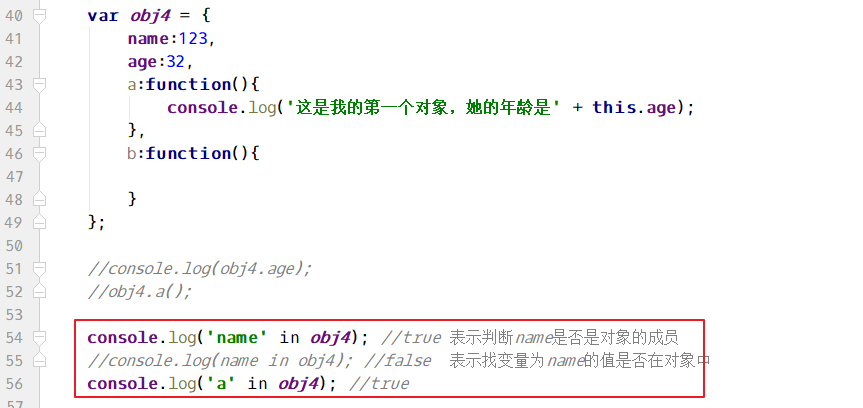


对象内部的this：理解它，对比php中的$this即可。

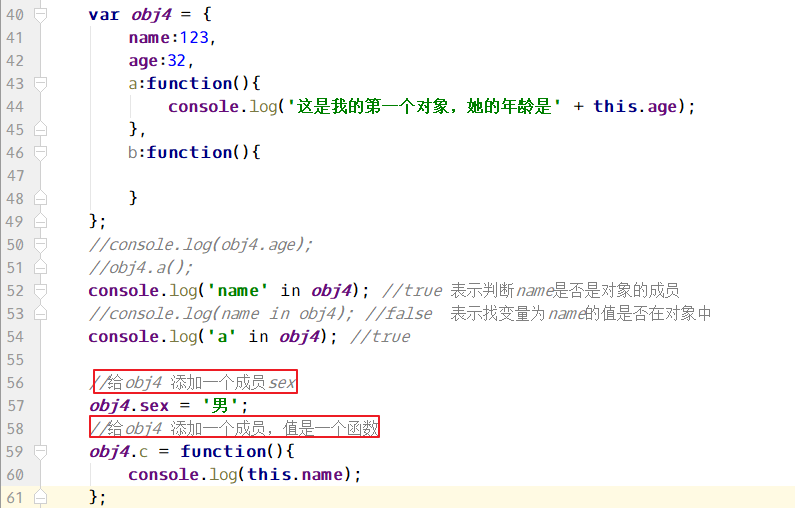


3、判断对象中的成员

用 in 判断对象中是否存在某个成员。

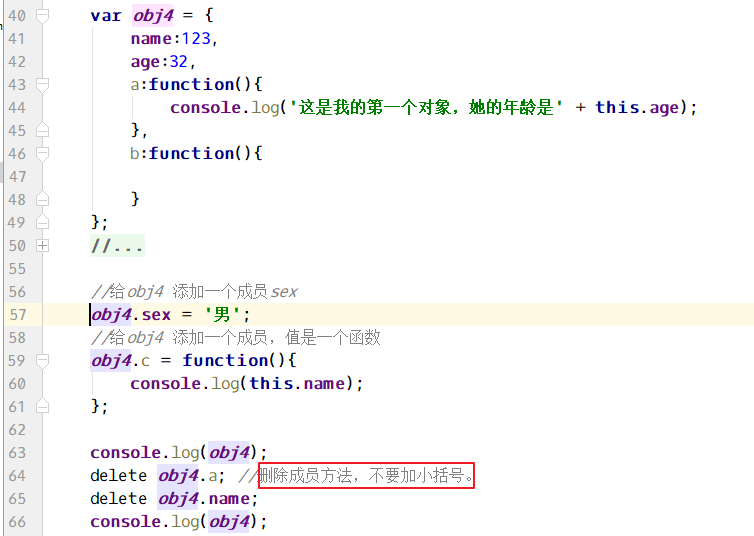


4、添加对象的成员



5、删除对象的成员

使用delete关键字来删除对象的成员。



十一、数据类型

1、数据类型

JavaScript 语言的每一个值，都属于某一种数据类型。JavaScript 的数据类型，共有六种。（ES6 又新增了第七种 Symbol 类型的值）

* number
  + int
  + float
* string
* bool
* null 空值，如果要设置一个变量的值为null，只能手动赋值为null。null也是一个对象。
* undefined 未定义，如果一个变量声明了，但是没有给值，则这个变量的值就是undefined
* object
  + 纯对象（狭义上的对象）
  + 数组
  + 函数
  + JS中任何数据类型都可以看做是对象

通常，我们将数值、字符串、布尔值称为原始类型（primitive type）的值，即它们是最基本的数据类型，不能再细分了。而将对象称为合成类型（complex type）的值或引用类型，因为一个对象往往是多个原始类型的值的合成，可以看作是一个存放各种值的容器。至于undefined和null，一般将它们看成两个特殊值。

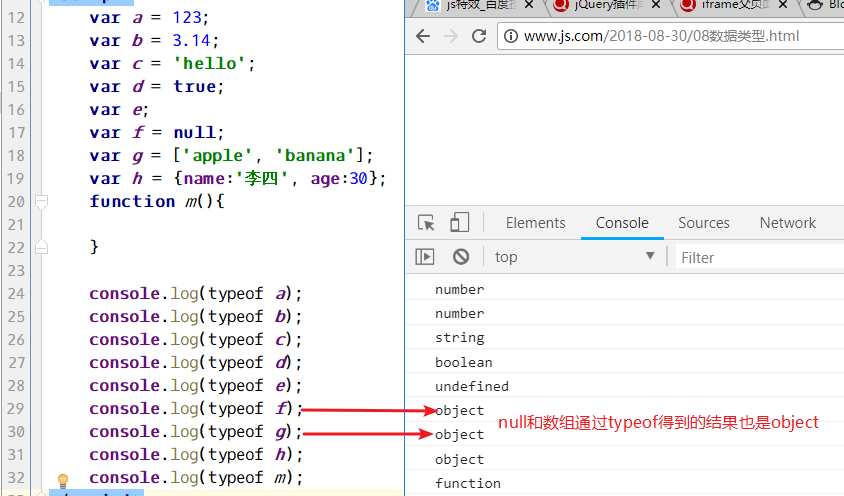
对象又可以分成三个子类型，即狭义的对象（object）、数组（array）、函数（function），其实，更加广义的讲，原始类型的值（数值、字符串、布尔）也可以看做对象。

2、数据类型示例

**var *a*** = 123;  
**var *b*** = 3.14;  
**var *c*** = **'hello'**;  
**var *d*** = **true**;  
**var *e***;  
**var *f*** = **null**;  
**var *g*** = [**'apple'**, **'banana'**];  
**var *h*** = {**name**:**'李四'**, **age**:30};  
**function** *m*(){  
  
}

可以输出查看效果。

3、判断数据类型



使用typeof无法区分null、数组、对象，要想区分他们三个，必须使用instanceof。

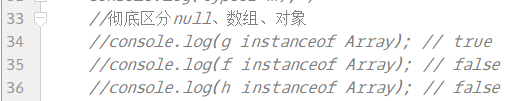
使用instanceof的语法：

变量 instanceof 类型

类型：包括Array、Object

如果一个变量instanceof Array之后，结果为true，则表示变量是数组类型；

如果一个变量instanceof Object之后，结果为true，则表示变量是对象类型；



下面一个函数可以判断数据类型到底是什么？

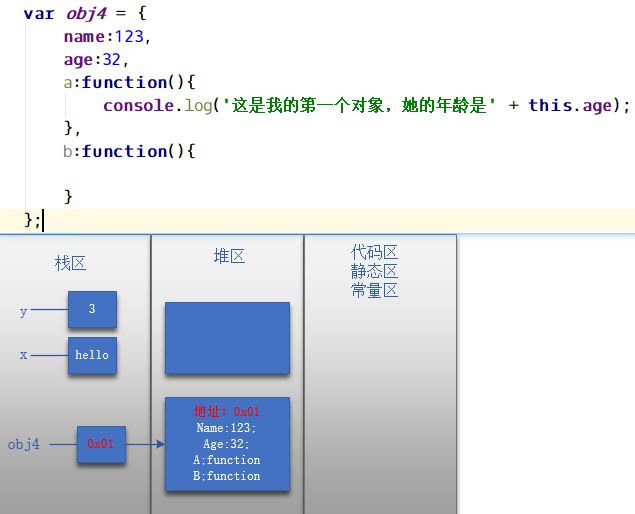
**function** *panduan*(x){  
 **if**(typeof(x) != **'object'**)  
 {  
 **return** typeof(x);  
 }  
 **else** {  
 **if**(x instanceof Array)  
 {  
 **return 'array'**;  
 }**else if**(x instanceof Object){  
 **return 'object'**;  
 }**else**{  
 **return 'null'**;  
 }  
 }  
}  
**console**.log(*panduan*(***h***));

4、原始类型和引用类型

引用类型指对象，指的是狭义上的对象。数组不应该算引用类型。

区分是否是引用类型，关键看内存图。

对象在内存的栈区只保存它的地址，在堆区保存它的实际内容，这种类型的变量就是引用类型。

引用类型的一些应用：



十二、流程控制

1、顺序结构

2、分支结构

3、循环结构

4、continue和break

十三、函数

1、定义与调用

2、函数也是值

3、函数预加载

注意不要在非函数的代码块中声明函数。

4、参数

形参、实参、个数、传值方式

5、立即调用模式