JavaScript进阶

---JS函数及函数参数

河北师范大学软件学院
Software College of Hebei Normal University

内容提纲

- **函数的定义与调用**
- **函数参数的数量问题**
- **参数类型与传递方式（值、引用）**

函数的定义与调用

- 函数定义方式（3种）
  - 通过函数声明的形式来定义（要有函数名）
  - 通过函数表达式的形式来定义（可以是没有函数名的匿名函数，有名的话方便调用栈追踪）
  - 通过Function构造函数实例化的形式来定义（JS中函数也是对象，函数对象）

```
function max(a,b){
    return a>b?a:b;
}
max(2,3);//3
```

```
var max = function(a,b){
    return a>b?a:b;
};
max(2,3);//3
```

```
var max = new Function("a","b","return a>b?a:b;");
max(2,3);//3
```


参见实例demo01

函数的定义与调用

- 函数调用方式（4种，注意调用函数的主体）
  - 作为函数直接调用（非严格模式下this为全局对象，严格模式下为undefined）
  - 作为方法调用（this为调用此方法的对象）
  - 通过call( )和apply( )间接调用（this为函数对象的call/apply方法的首个参数，移花接木）
  - 作为构造函数调用（this为实例化出来的对象）

```
function test(){
    console.log(this);
}
test();//window
```

```
var obj = {
    x:0,
    test:function(){
        console.log(this.x);
    }
};
obj.test();//0
```


参见实例demo02 Part1 函数嵌套情况下的调用

函数的定义与调用

- 函数调用方式（4种，注意调用函数的主体）
  - 作为函数直接调用（非严格模式下this为全局对象，严格模式下为undefined）
  - 作为方法调用（this指向调用此方法的对象）
  - 通过call( )和apply( )间接调用（this为函数对象的call/apply方法的首个参数，移花接木）
  - 作为构造函数调用（this指向实例化出来的对象）

```
objA = {name:"AA"};
objB = {name:"BB"};
objA.foo = function(){
    console.log(this.name);
};
objA.foo();//AA
objA.foo.call(objB);//BB
```

```
function Person(name){
    this.name = name;
}
Person.prototype.sayHi = function(){
    console.log("Hi,i'm "+this.name);
}
var p1 = new Person("Jack");
p1.sayHi();//Hi,i'm Jack
```

参见实例demo02 Part2 call/apply实例

内容提纲

- 函数的定义与调用
- **函数参数的数量问题**
- 参数类型与传递方式（值、引用）

调用参数的数量问题详解

- JS函数调用时实参数量可以与形参不一致
  - 实参数量大于形参的情况（通过函数对象属性arguments获得所有实参、类数组对象）
  - 实参数量小于形参的情况（少的参数值为undefined、可使用| |来给出默认值）

```
function test() {
    var s = "";
    for (var i = 0; i < arguments.length; i++) {
        s += arguments[i];
    }
    return s;
}
test("hello,", "world!");//"hello,world!"
```


参见实例demo03 实参数量大于形参数量

调用参数的数量问题详解

- JS函数调用时实参数量可以与形参不一致
  - 实参数量大于形参的情况（通过arguments获得所有实参、类数组对象、拥有对象属性）
  - 实参数量小于形参的情况（少的参数值为undefined、可使用| |来给出默认值）

```
var sum = function(a,b,c){
    b = b||4;
    c = c||5;
    return a+b+c;
}
console.log(sum(1,2,3));//1+2+3
console.log(sum(1,2));  //1+2+5
console.log(sum(1));    //1+4+5
```

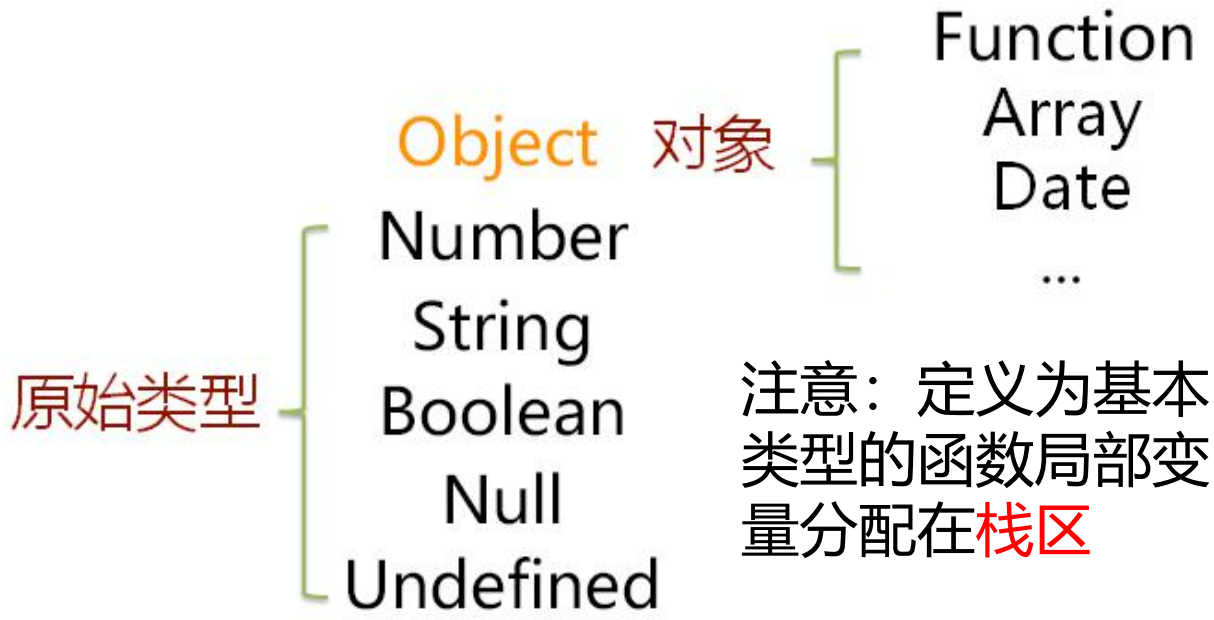

参见实例demo03 实参数量小于形参数量

内容提纲

- 函数的定义与调用
- 函数参数的数量问题
- 参数类型与传递方式（值、引用）

JavaScript数据类型-背景知识

- 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）
  - 引用（对象）类型（Object（Array、Function、Date、Error等））



注意：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

注意：定义为基本类型的函数局部变量分配在栈区

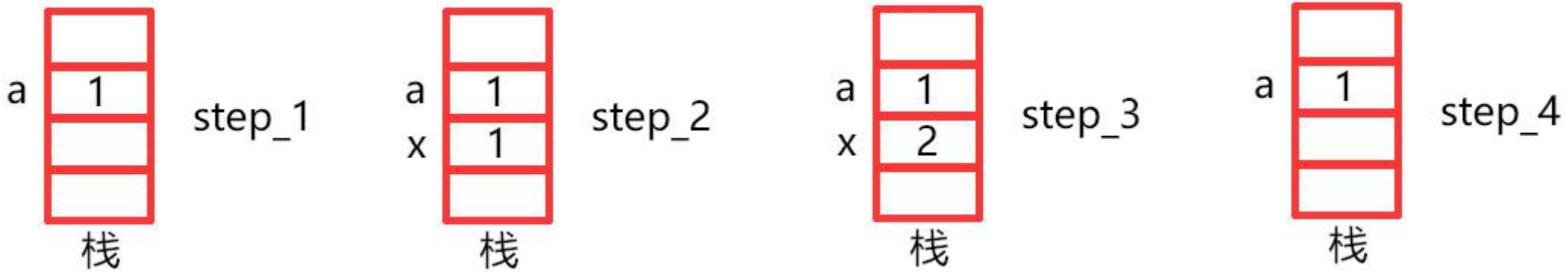
- 不同类型的数据，参数传递方式不同（值传递、引用传递）

参数类型与传递方式 - 值传递（基本数据类型的传递）

- 实参为基本数据类型时，形参改变不影响实参（值传递）

```
var a = 1;
function foo(x) {
    x = 2;  //step_2 此时a=1,x=1;
    console.log("a:",a,"x:",x); //step_3 此时a=1,x=2;
}

foo(a); //step_1 此时a=1
console.log(a); //step_4  a仍为1
```



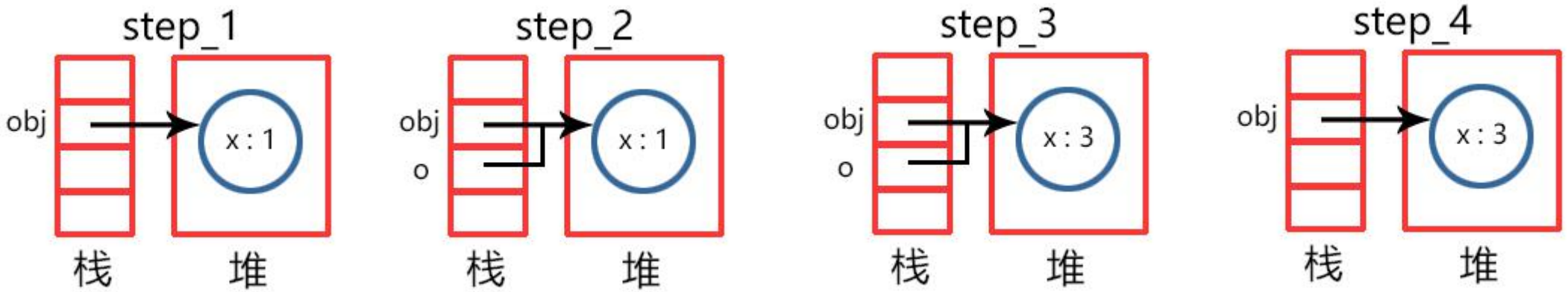
参见实例demo04 trace追踪 学会debug调试

参数类型与传递方式- 引用传递（引用数据类型的传递）

- 实参为引用类型时，形参改变影响实参（引用传递）

```
var obj = {x:1};
function fee(o){
    o.x = 3;     //step_2 此时obj.x为1，o.x为1
    console.log(obj.x,o.x); //step_3 此时obj.x为3，o.x为3
}

fee(obj);   //step_1 此时obj.x为1
console.log("obj.x :",obj.x);   //step_4 obj.x被改写为3
```



参见实例demo04 trace追踪 学会debug调试

总结

- 函数的定义与调用
- 调用参数的数量问题详解
- 参数类型与传递方式（值、引用）

Thank You!

河北師范大学软件学院
Software College of Hebei Normal University

作业

- 复习本章课件及练习
- 将自己的本章练习提交到个人仓库中
- 阅读《JavaScript权威指南》第8章中的8.1、8.2、8.3、8.4这4个小节中的内容
- 慕课网（第5章）
  https://www.imooc.com/learn/10