正课:

1. OOP:

1. OOP:

什么是对象: 使用: 描述现实中一个具体事物的程序结构

存储: 集中存储一个事物的属性和功能的存储空间

什么是面向对象: 程序都是先用对象描述出现实中的事物，再使用对象去操作现实中的事物。

为什么: 便于大量数据和功能的管理

何时: 今后所有程序都是用面向对象实现的

如何: 三大特点: 封装，继承，多态

1. 封装:

什么是: 创建一个对象，集中保存一个事物的属性和功能

为什么: 便于大量数据和功能的管理

何时: 只要使用面向对象思想编程，都要先创建对象

如何: 3种:

1. 直接量:

var obj={

属性名:值,

... : ...,

方法名:function(){

... ...

}

}

访问对象成员: 成员=属性+方法

访问属性: obj.属性名 ——属性其实就是保存在对象中的变量，用法和变量完全一样

访问方法: obj.方法名() ——方法其实就是保存在对对象中的功能函数，用法和函数完全一样

问题: 对象自己的方法内要使用对象自己的属性

错误: 直接使用属性名

为什么: 不加任何前缀的变量，默认只能在作用域链中查找，不能擅自进入对象中获得对象的成员。

正确: this.属性名

什么是this: 自动指向正在调用该函数的.前的对象

.前是谁，this就自动指谁

本质: 保存在函数作用域中的一个关键词

何时: 今后，只要对象自己的方法，要访问自己的成员，都必须加this.

2. 用new: 2步:

1. 创建空对象: var obj=new Object()

new可省略, ()也可省略，但必须保留其一

2. 为对象添加新成员: obj.成员=值

揭示本质: js中一切对象底层，都是关联数组！

vs 关联数组:

1. 即可用.也可用[""]访问成员

何时用. : 如果属性名是写死的

何时用[""] : 如果属性名是动态获得的

2. 访问不存在的成员不报错，返回undefiend

3. 给不存在的成员赋值不报错，强行添加该成员

4. 都能用for in遍历每个属性

问题: 一次只能创建一个对象

反复创建多个相同结构的对象时:

代码繁琐，不便于维护

解决: 构造函数！

3. 用构造函数:

何时: 反复创建多个同一类型相同结构的对象时

优点: 代码/结构重用

如何: 2步:

1. 用构造函数定义同一类型所有对象的统一结构

function 类型名(形参,...){

this.属性名=形参;

... ...

this.方法名=function(){

... this.属性 ...

}

}

2. 调用构造函数，反复创建多个相同结构的对象:

var obj=new 类型名();

new: 4件事:

1. 创建一个新的空对象:

2. ?

3. 调用构造函数:

new会让this自动指向新对象

执行构造函数过程中: 通过强行给新对象赋值新属性的方式，添加新属性

4. 返回新对象地址给变量

问题: 构造函数中的方法定义，每new一次，就会反复创建副本——浪费内存！也不便于维护

解决: 1. 构造函数中不应再包含方法定义！

2. 方法定义放哪儿？

2. 继承:

什么是: 父对象的成员，子对象无需重复创建，就可直接使用

为什么: 代码重用，节约内存

何时: 只要发现多个子对象拥有相同的成员时，都要用继承实现——只定义一次

如何: 共有的成员，只要定义在父对象上一份即可！

所有子对象，自动继承使用

如何使用父对象: ——原型对象

什么是: 集中存储所有子对象共有成员的父对象

创建: 不用创建，买一赠一

每创建一个构造函数，就附赠一个空的原型对象

确定继承关系: new的第2步: 自动让新对象继承构造函数的原型对象

利用继承关系: 只要放在原型对象中的成员，所有子对象自动就可继承使用

总结: 定义一种类型: 2部分:

1. 构造函数:

定义属性结构

用于创建子对象

2. 原型对象:

定义所有子对象共有的属性和方法

保存共有成员

内置类型的原型对象:

什么是内置类型: ES标准规定的，浏览器/平台已经实现的对象

包括: 11个:

Number String Boolean

Array Date RegExp Math

Error

Function Object

global (在浏览器中被window代替)

如何为内置类型扩展共有方法:

其实: 每种内置类型，同样也包含2部分:

1. 构造函数: 用于创建该类型的子对象

2. 原型对象: 保存该类型所有子对象共有的成员！

将共有方法添加到内置类型的原型对象中

this->将来调用该方法的.前的子对象

作业: 为String添加 ltrim() 和 rtrim()

原型链(prototype chain):

什么是: 由多级父对象，逐级继承，形成的链式结构

保存着所有对象的属性

控制着对象属性的使用顺序和共享范围

3. 多态:

什么是: 同一个函数在不同情况下表现出不同的状态

包括2种:

1. 重载(overload)

2. 重写(override)

什么是重写: 子对象觉得父对象的成员不好用，在本地定义同名成员来覆盖父对象成员

何时: 子对象觉得父对象的成员不好用

如何: 只要在子对象本地定义同名成员

作业: 如何判断一个对象是不是数组类型，有几种方法

typeof?

课后任务:

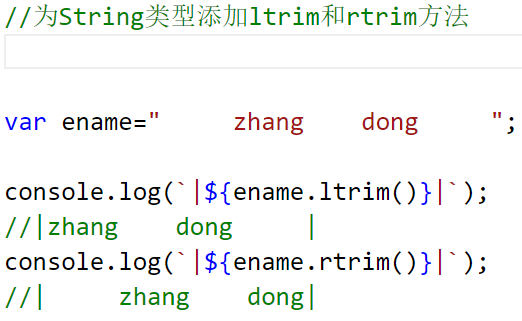
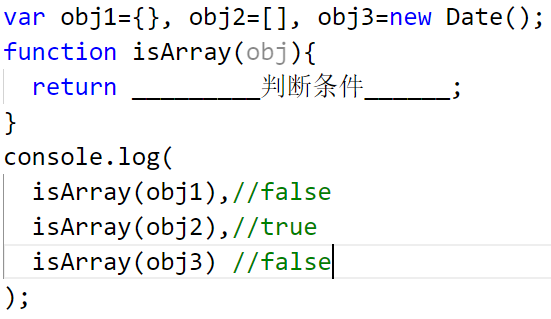
(1)复习: 面向对象三大特点

(2)作业: 完成课后练习:  
 题目要求:

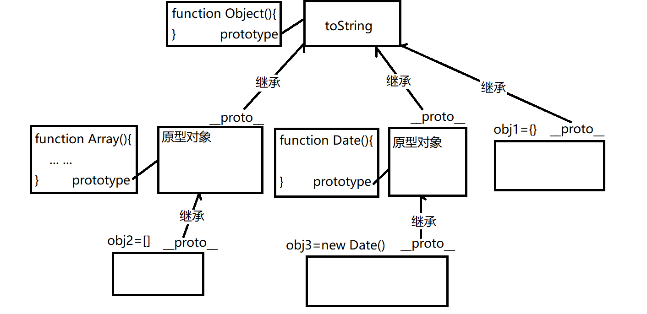
1. 为String类型添加共有方法ltrim和rtrim

2. 判断一个对象是不是数组，共有几种方法

运行效果:

提示:



(3)项目:

题目要求:

运行效果: