面向对象

毕向东

3 面向对象

3.1 面向对象概念

3.2 类与对象的关系

3.3 封装

3.4 构造函数

3.5 this关键字

3.6 static关键字

3.7 单例设计模式

3.1 面向对象概念

3.1.1 理解面向对象

3.1.2 面向对象的特点

3.1.1理解面向对象

- 面向对象是相对面向过程而言
- 面向对象和面向过程都是一种思想
- 面向过程
  - 强调的是功能行为
- 面向对象
  - 将功能封装进对象，强调具备了功能的对象。
- 面向对象是基于面向过程的。

3.1.2 面向对象的特点

- 是一种符合人们思考习惯的思想
- 可以将复杂的事情简单化
- 将程序员从执行者转换成了指挥者
- 完成需求时：
  - 先要去找具有所需的功能的对象来用。
  - 如果该对象不存在，那么创建一个具有所需功能的对象。
  - 这样简化开发并提高复用。

3.1.3 面向对象开发，设计，特征

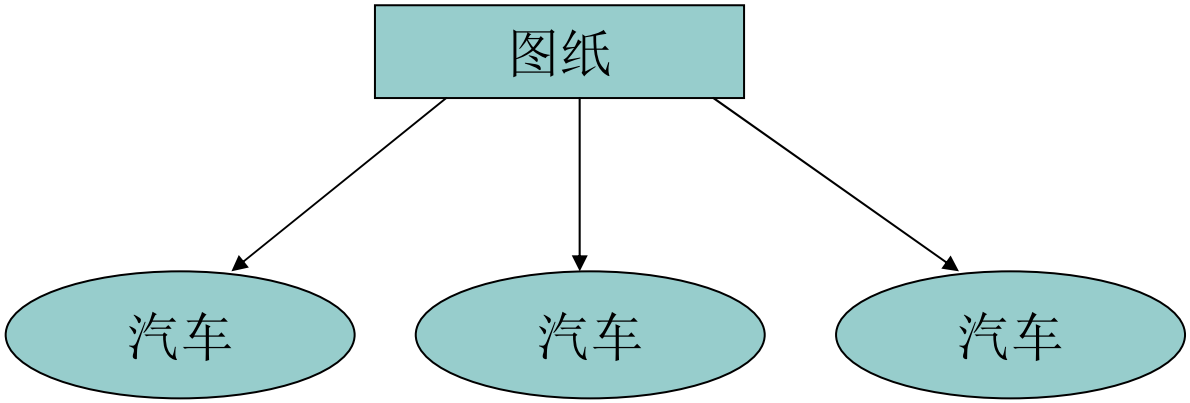
- 开发的过程：其实就是不断的创建对象，使用对象，指挥对象做事情。
- 设计的过程：其实就是在管理和维护对象之间的关系。
- 面向对象的特征：
  - 封装(encapsulation)
  - 继承(inheritance)
  - 多态(polymorphism)

3.2 类与对象的关系

- 使用计算机语言就是不断的在描述现实生活中的事物。
- java中描述事物通过类的形式体现，类是具体事物的抽象，概念上的定义。
- 对象即是该类事物实实在在存在的个体。

3.2.1 类与对象(图例)

- 类与对象的关系如图



- 可以理解为：
  - 类就是图纸
  - 汽车就是堆内存中的对象

3.2.2 类的定义

- 生活中描述事物无非就是描述事物的属性和行为。
  - 如：人有身高，体重等属性，有说话，打球等行为。
- Java中用类class来描述事物也是如此
  - 属性：对应类中的成员变量。
  - 行为：对应类中的成员函数。
- 定义类其实在定义类中的成员(成员变量和成员函数)。

3.2.3 成员变量和局部变量的区别？

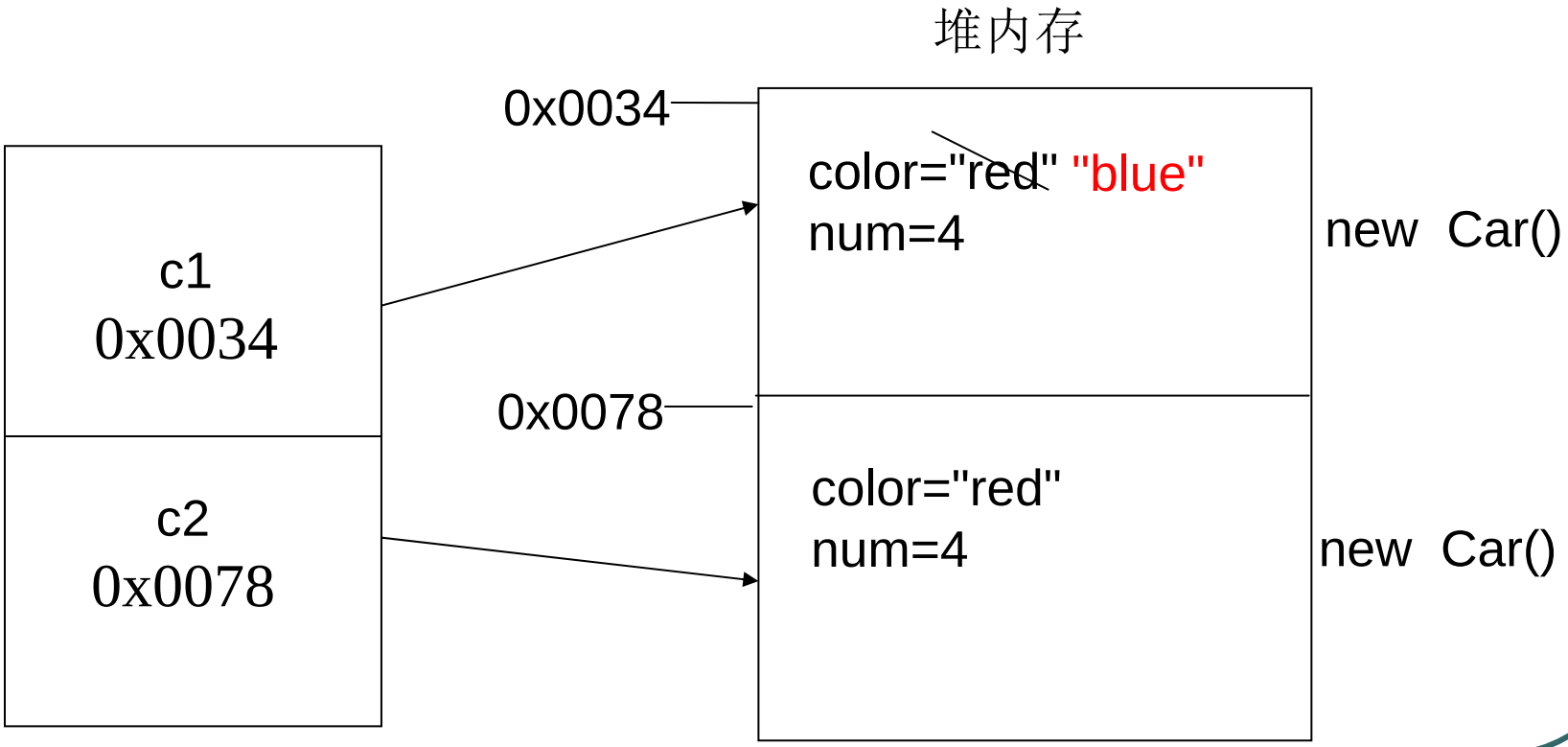
- 成员变量：
  - 成员变量定义在类中，在整个类中都可以被访问。
  - 成员变量随着对象的建立而建立，存在于对象所在的堆内存中。
  - 成员变量有默认初始化值。
- 局部变量：
  - 局部变量只定义在局部范围内，如：函数内，语句内等。
  - 局部变量存在于栈内存中。
  - 作用的范围结束，变量空间会自动释放。
  - 局部变量没有默认初始化值。

3.2.4 创建对象，使用对象

```
class Car//对Car这类事物进行描述
{
   String color = "red";
   int num = 4;
   void show()
   {
        System.out.println("color="+color+"..num="+num);
   }
}
class CarDemo
{
   public static void main(String[] args)
   {
        Car c = new Car();//建立对象
        c.color = "black";//对对象的属性进行修改
        c.show();//使用对象的功能。
   }
}
```

3.2.5 对象内存结构

```
Car c1 = new Car();c1.color="blue";
Car c2 = new Car();
```



3.2.6 匿名对象

- 匿名对象是对象的简化形式
- 匿名对象两种使用情况
  - 当对对象方法仅进行一次调用的时
  - 匿名对象可以作为实际参数进行传递

3.3 封装(Encapsulation)

- 封装：是指隐藏对象的属性和实现细节，仅对外提供公共访问方式。
- 好处：
  - 将变化隔离。
  - 便于使用。
  - 提高重用性。
  - 提高安全性。
- 封装原则：
  - 将不需要对外提供的内容都隐藏起来。
  - 把属性都隐藏，提供公共方法对其访问。

3.4.1 private(私有)关键字

- **private**关键字：
  - 是一个权限修饰符。
  - 用于修饰成员(成员变量和成员函数)
  - 被私有化的成员只在本类中有效。
- 常用之一：
  - 将成员变量私有化，对外提供对应的set ， get 方法对其进行访问。提高对数据访问的安全性。

3.4 构造函数

特点：

1. 函数名与类名相同
2. 不用定义返回值类型
3. 不可以写return语句

作用：

给对象进行初始化。

注意：

1. 默认构造函数的特点。
2. 多个构造函数是以重载的形式存在的。

3.5 this关键字

特点：this代表其所在函数所属对象的引用。
换言之：this代本类对象的引用。

什么时候使用this关键字呢？
当在函数内需要用到调用该函数的对象时，就用this。
例程。

3.6 static(静态)关键字

- static关键字：
  - 用于修饰成员（成员变量和成员函数）
- 被修饰后的成员具备以下特点：
  - 随着类的加载而加载
  - 优先于对象存在
  - 被所有对象所共享
  - 可以直接被类名调用
- 使用注意
  - 静态方法只能访问静态成员
  - 静态方法中不可以写this，super关键字
  - 主函数是静态的