# 概述

## 如何描述现实世界事物

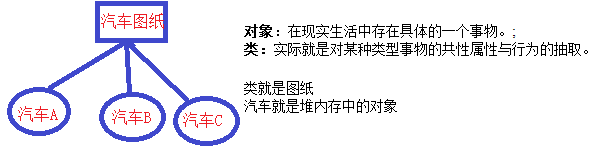
属性 就是该事物的描述信息(事物身上的名词) --成员变量

行为 就是该事物能够做什么(事物身上的动词) --成员方法

**Field = 属性 = 成员变量**

**Method = 行为 = 成员方法**

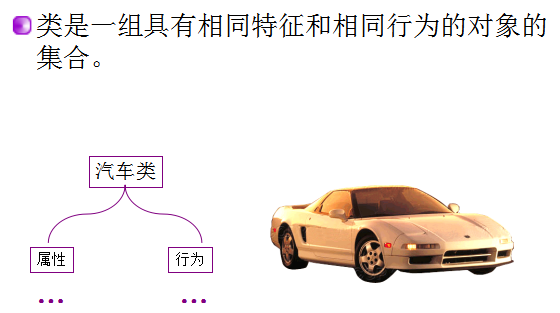
## 类和对象的概念



## 什么是对象



## 什么是类

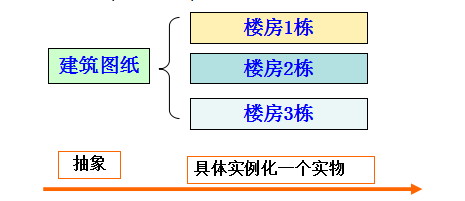


## 类与对象的关系

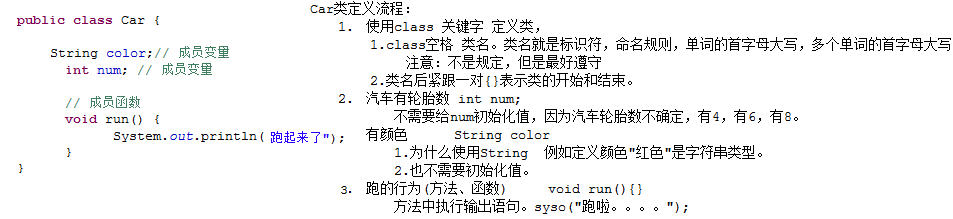
类与对象什么关系

类（Class）是对一组具有相同属性、操作和关系的同类事物的描述，是抽象的、概念上的定义。

对象（Object）是实际存在的该类事物的每个个体，因而也称实例(instance)。



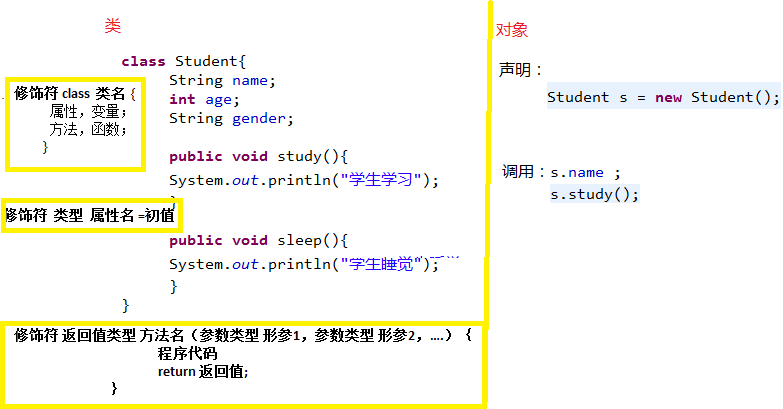
## 使用java来描述事物的属性和行为



举例：

\* 类 学生

\* 对象 具体的某个学生就是一个对象



## 类和对象的区别？

* + 类是集合、是抽象的、是制造对象的模板、是数据结构和行为的定义；
  + 对象是个体、是具体的、是类的实例、是数据。

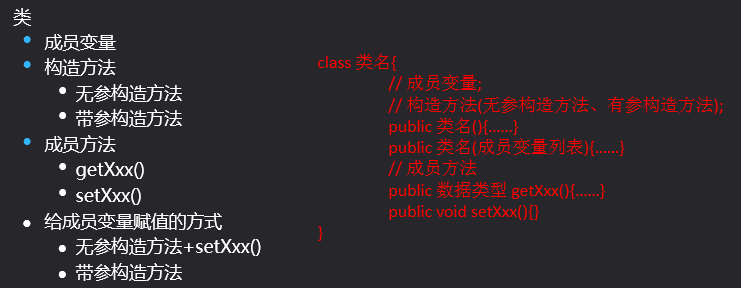
# 类class

## 什么是类？

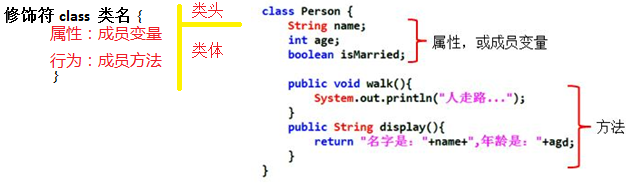
* + 具有相同特征和行为的同一类事物的集合，是抽象的概念，是模板。

## 如何设计一个类？如何得到类的实例？

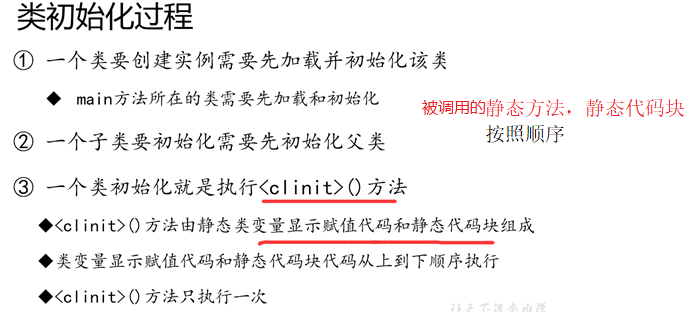
* + 共同的特征→属性，共同的行为→方法；【抽象】
  + 使用new 类名()；得到类的实例。 类名 引用名 = new 类名();
  + 由类的得到类的对象(实例)的过程，成为“实例化”。

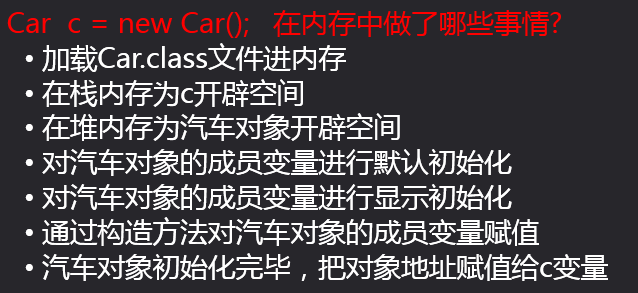


## 类的定义



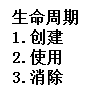
## 类的初始化过程（静态）





# 对象（实例化）

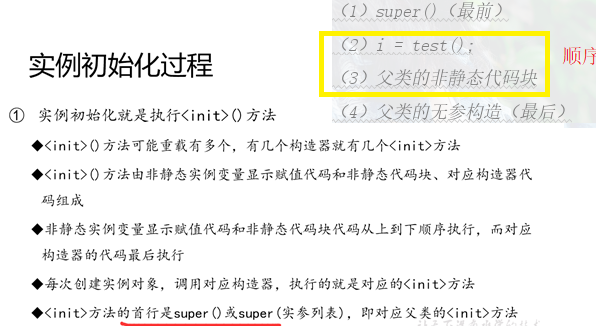
类是对象的抽象，即对象的类型。对象则是类的实例

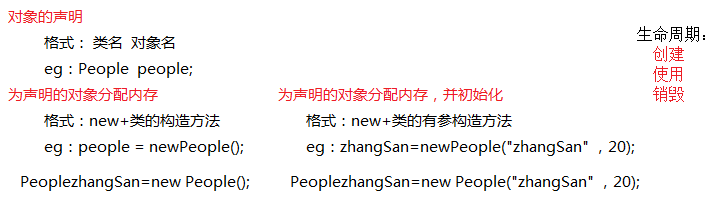


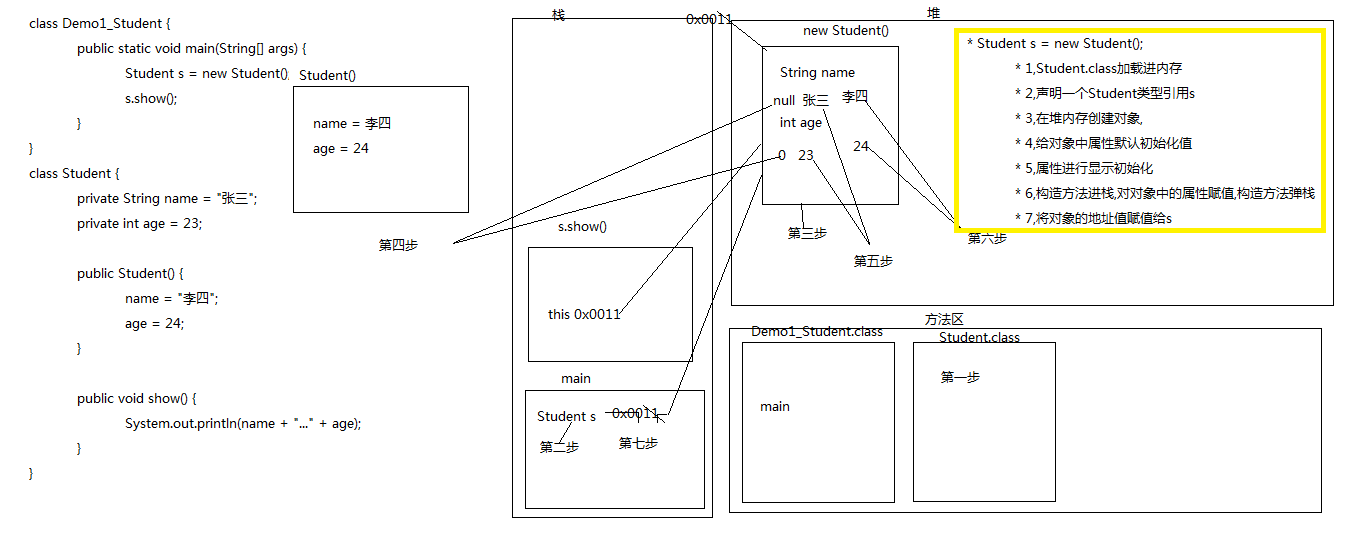
## 什么是对象？

* + 实实在在存在的个体，现实世界中一切都可以看做对象。

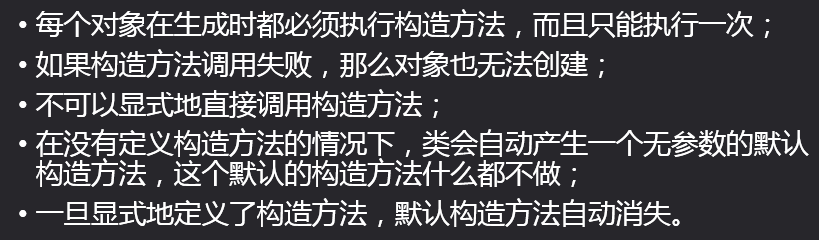
## 对象的实例化过程（非静态）







## 构造方法



## 使用对象

使用对象的成员变量： 对象名. 变量名

调用对象的方法： 对象名.方法名（）

体现封装：类中方法可以操作成员变量。当对象调用方法时方法中出现的成员变量就是分配给该对象的成员变量

## 匿名对象

### 什么是匿名对象

没有名字的实体，也就是该实体没有对应的变量名引用

### 匿名对象应用场景

调用方法，仅仅只调用一次的时候。

那么，这种匿名调用有什么好处吗? 节省代码

注意：调用多次的时候，不适合。匿名对象调用完毕就是垃圾。可以被垃圾回收器回收。

匿名对象可以作为实际参数传递

### 案例演示

匿名对象应用场景

**class** Demo2\_Car {

**public** **static** **void** main(String[] args) {

/\*Car c1 = new Car(); //创建有名字的对象

c1.run();

c1.run();

new Car().run(); //匿名对象调用方法

new Car().run(); \*/ //匿名对象只适合对方法的一次调用,因为调用多次就会产生多个对象,不如用有名字的对象

//匿名对象是否可以调用属性并赋值?有什么意义?

/\*

匿名对象可以调用属性,但是没有意义,因为调用后就变成垃圾

如果需要赋值还是用有名字对象

\*/

**new** Car().color = "red";

**new** Car().num = 8;

**new** Car().run();

}

}

**class** Car {

String color; //颜色

**int** num; //轮胎数

**public** **void** run() {

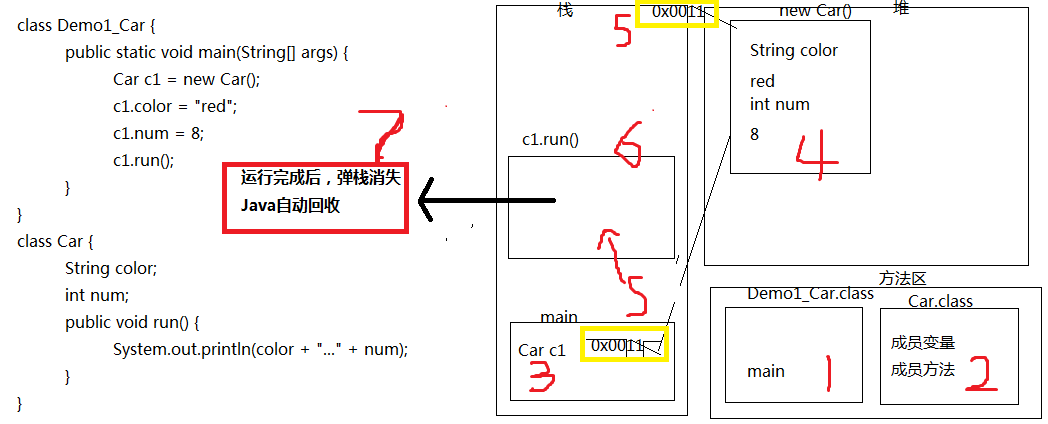
System.*out*.println(color + "..." + num);

}

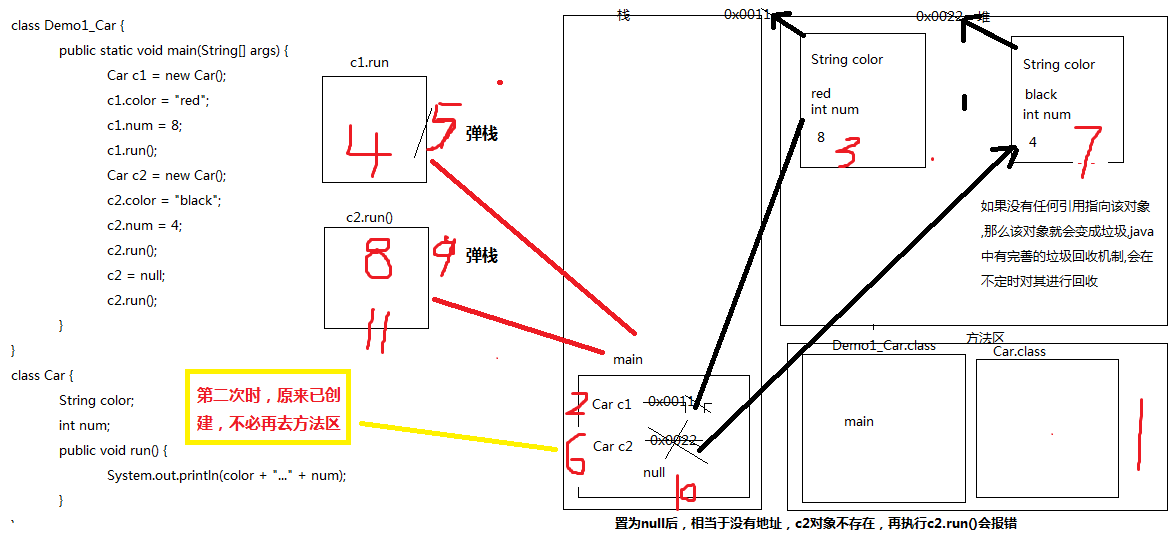
}

## 对象的内存图

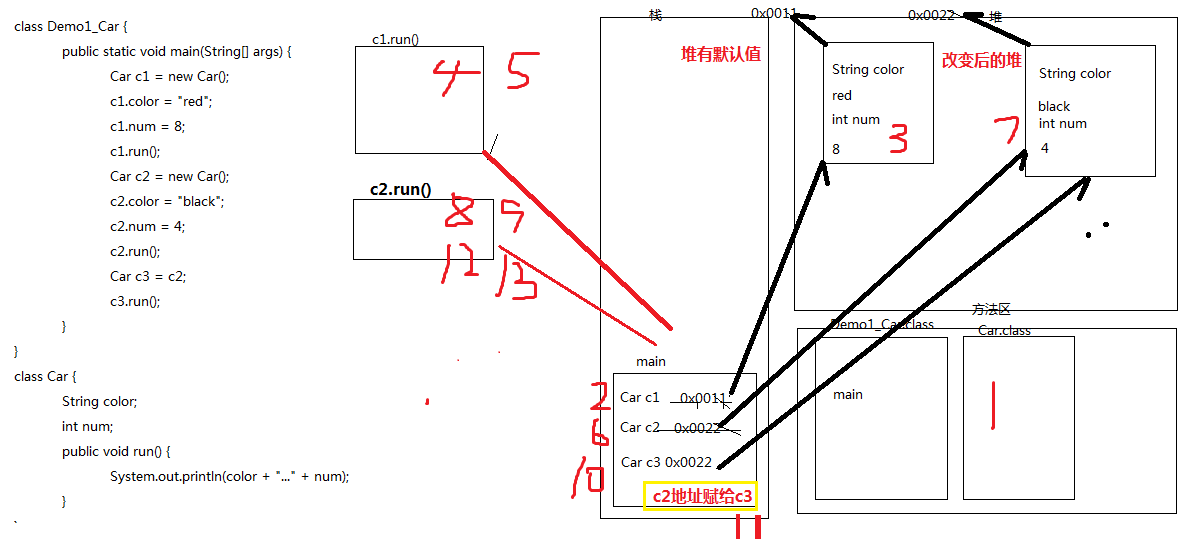
### 一个对象的内存图



### 两个对象的内存图



### 三个引用两个对象内存图



### 匿名对象

