编程思想:

面向过程->面向对象->面向接口-**>面向组**件->面向切面->面向服务

Java面向对象的高级部分:封装、继承、多态、接口

学习目标:

1. java面向对象高级概念
2. 初步形成面向对象设计
3. 更好读懂代码

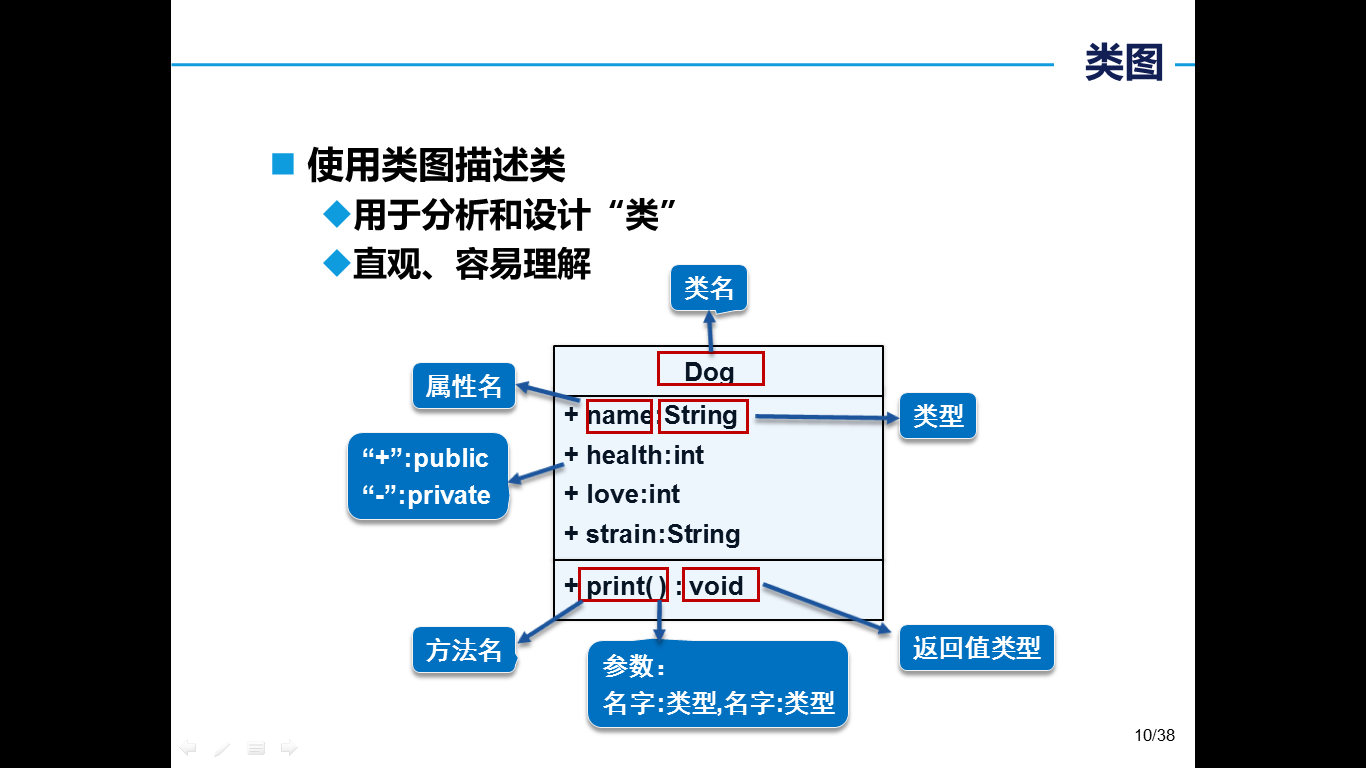
课程目标:

1. 理解面向对向的设计步骤
2. 方法重载 掌握
3. 构造方法 掌握
4. 封装的概念 掌握
5. 软件出现的目的
6. 用计算机的语言描述现实世界
7. 用计算机解决现实世界的问题

面向对象设计和开发程序好处:交流更加流畅、提高设计和开发的效率

1. 面向对象设计的步骤:
2. 确定类|发现类
3. 发现类的属性
4. 发现类的方法

注意:面向对象设计的产物:类图



1. 构造方法
2. 什么是构造方法:

当实例化对象时，自动执行的方法称为构造方法.

1. 构造方法定义:

public 类名(参数列表){

一般情况：完成属性的初始化工作…

}

注意:

1.每次创建对象时都会调用构造方法

2.每个类中默认都有一个不带参，不做任何操作的构造方法

Public 类名(){ }

当手动添加了构造方法，默认构造就不存在了,推荐把默认构造加上

3.一个类中可以定义多个构造(必需满足方法重载的条件)

示例:

|  |
| --- |
| *//第一步:定义类* **public class** Student {  *//第二步:确定类的属性* **public int xh**; *//学号* **public** String **name**; *//姓名* **public int age**; *//年龄* **public** String **sex**; *//性别   //第三步:确定类的方法* **public void** show(){  System.***out***.println(**"学号是："**+**this**.**xh**);  System.***out***.println(**"姓名是："**+**this**.**name**);  System.***out***.println(**"年龄是："**+**this**.**age**);  System.***out***.println(**"性别是："**+**this**.**sex**);  }  *//默认构造* **public** Student(){}  *//定义带参构造* **public** Student(**int** xh,String name,**int** age,String sex){  *//一般实现属性的初始化工作* **this**.**xh**=xh;  **this**.**name**=name;  **this**.**age**=age;  **this**.**sex**=sex;  } }  主函数:  **public static void main(String[] args) {**  **//1.创建学生对象 //先有对象再赋值**  **Student stu = new Student();**  **//赋值**  **stu.xh=101;**  **stu.name="张三";**  **stu.age=21;**  **stu.sex="男";**  **//调用方法**  **stu.show();**    **//2.创建学生对象 //创建对象时并初始化**  **Student stu2 = new Student(101, "罗总", 22, "女");**  **stu.show();**  **}** |

1. 方法重载
2. 实现方法重载的条件:

方法名相同，参数类型不同、参数个数不同、参数顺序不同即可构造方法重载.返回值类型不参与重载.

1. 作用:

将相似的功能归档(体现在方法名称上)在一起，便于调用者调用.

1. 访问修饰符
2. 作用:限制类成员(成员属性、成员方法)对外的访问级别.
3. 访问修饰符

private私有的:只能在类的内部使用.

默认:表示在同一个包下使用

**Protected受保护的:只能在同包下，或者子类中可以使用**

**Public公有的:无限制**

1. 封装(它是面向对象的三大特征之一)
2. 什么是封装:

**封装：将类的某些信息隐藏在类内部，不允许外部程序直接访问，而是通过该类提供的方法来实现对隐藏信息的操作和访问**

注意:封装是对类的私有属性进行封装

1. 实现封装

2.1将类的属性修改为私有

2.2使用setter和getter方法对类私有成员的封装

Setter方法用于赋值 getter方法用于取值

在类中-右键->找到generate🡪选中setter and getter

1. 封装方法的作用
2. 通过隐藏对象的属性来保护对象内部的状态
3. 可以对成员变量进行更精确的控制

如：只读、只写属性

1. 可以在封装方法添加控制语句