# 接口

类 方法都是实现的 实例化对象就可以调用

抽象类 部分方法是实现的，但是还有部分方法没有实现，让子类去实现，成为一个标准。子类中实现的方法统一。

接口 更加抽象了，里面方法都是定义，没有实现。所以让实现类去实现。更加统一更加标准了。但是注意在jdk8以后接口可以通过default实现方法体，接口和抽象类合并了。

1.2 1.5 1.8 3个版本是更新比较大的

j2se java5 java8

static关键字

继承只能是单继承，而接口可以多实现，而且接口继承可以是多个接口。弥补了java中不能多继承的功能。

**implements**

实现接口，并且实现接口中的所有方法

接口中都是标准

固定的，如果被修改了，那就不是标准

所以 接口中定义的属性都是 静态的常量

static 静态的

作用在属性上和方法上

引用

int a=byte b

接口可以引用实现类

接口配合工厂模式才能发挥作用

接口可以引用实现类，父类可以引用子类，都是安全的

同种类型之间可以转换 ，自动 强制

低精度-》高 自动

反之就强制

向上自动转型 父 上 子 下

向下 强制转换 注意 强制，可定会有问题的

高精度转低精度 可以安全转换，也可能数据溢出

明确 转可以 但是注意，强制

类中其实也是同种类型 ，都是继承object

A 类

B 类

不 是同种

前面基本类型 最多就是数据丢失 丢失了就不准了

对象中，就不允许这样装换

不同种类型就不能引用

多继承多实现会导致类型的不确定

java放弃多继承的一个原因之一

又不想失去这个功能，所以在接口中用了一下

静态 static

区别于成员

实例化对象之后才能被调用的成员

成员属性 成员函数

通过对象去访问

static是可以不用通过对象也能访问的那种 类属性，类函数

静态属性 静态方法

注意点，静态的不能访问非静态

非静态的可以访问静态的资源

最后从内存上面分析

静态的数据是每个对象所共有的。只要有一个改变，所有的对象都发生改变。使用的是同一个数据

静态资源 共享数据

非静态的资源中可以访问静态的

接口，静态

抽象类

类

语法比较简单。重点以后我们怎么去使用它们，什么时候用类。使用时候用抽象类，用接口

站在 设计的角度 考虑

目前只要掌握基础语法

工厂模式，反射机制

集合框架 通过接口设计的。

在1.8以后 接口

之前 只有定义，没有实现

之后，有定义，也有实现，关键字 default

抽象类

只要有一个接口（实现和非实现）

思维角度分析

没有主动去学习jdk8

在后面分布式 引入jar 比较新的

进入源码

发现jar 里面接口 居然有实现

改版了

研究1.8特性

jar稳定 ，后面最新发现问题

新版版兼容以前版本

旧版本不能新版本

框架是新版本

但是用旧版本jdk

看版本

非静态的调用静态的

对象.静态的资源

可以

ic接口 show2

IC2 show4

接口中已经实现的代码

那么如果在实现类中再次实现了。语法允许吗？调用的是接口中的还是实现类中？

c2 show2

c2 show4

实现类必须实现接口中未实现的方法，有可以实现接口中已经实现的方法，用这个实现类中的方法了

接口中 ，静态的不能被继承

也不能实现静态的方法

不能通过对象去访问的

只能通过接口名字直接访问静态

类

接口

静态

成员

多继承（类，接口）

传递继承

定义类

定义接口

动物 狗 猫

在做什么功能定义类

做什么定义接口

定义里面的属性 使用继承关系

定义方法的时候，使用接口

接口中一般都是方法调用

即使放属性，静态的常量 ，配置信息

使用static 之后的类加载

{}

静态代码块，只要jvm加载就运行一次。整个项目就只运行一次。

比如：加载数据库驱动

驱动只要加载一次就可以了。所以设计在静态代码块中

{ } 成员代码块

构造函数 区别 功能是一样但是顺序是不一样的

加载类的时候，就调用了静态代码块

不管调用哪个构造函数。代码块始终运行的

而且先与构造函数之前运行。

static类的加载熟顺序

先执行静态变量 静态代码块 构造函数 初始换成员变量 代码块 构造函数下面的代码

加上继承关系

先调用父类的静态变量，静态代码块，子类的静态变量，静态代码块，子类的构造函数，父类的构造函数，父类的成员属性，父类的静态块，父类的构造函数其他代码 子类的成员属性，子类的代码块 子类的构造函数其他代码 结束了

沟通技巧

答案 一通的讲出来 没问题，对的

没有被录取

技术 沟通 表达能力

大公司 需要猎头 500强 合格的人才

# 封装

把属性方法放在一个类中，过程就是封装

访问的权限

参数传递封装

private私有的只能在一个类中使用

public 可以在不同包中，也不是子类之间能使用

默认的 同包中能使用  
受保护的 子类中使用，可以不用在同包中

封装+访问修饰 控制数据的安全性 保护