# java基础核心技术

- 三大特性
  - 封装
    - 1.方法
      - 方法名：小驼峰，后面的字母首字母大写
      - 构造方法必须与类名一样
        - 无参初始化 public name(){ }
        - 含参初始化 public name(int a,String b){this.a=;this.b=}
      - 成员方法自定义
        - 无参返回方法：public static void name( ){ }
        - 有参返回方法：public static name( ){return}
      - 要使用方法一定要先在mian里面创建该类的实例
    - 2.关键字 private 对输入进行把门
      - 使用方法：private int a
      - 赋值 public setXxx(数据){ }
      - 取值 public getXxx(){return}
      - 布尔型取值关键字为 isXxx(){return}
    - 优化
      - 避免重名尴尬 this
        - 使用方法：this.成员变量名
        - 区分：谁调用就是谁的this
      - JavaBean标准规范：private成员变量，无参、含参构造方法，Setter、Getter方法
  - 继承
    - 组合：class 父类{ }、class 子类 extends 父类{ }、public class{}执行main方法
    - 重写（子类方法与父类方法名一样）
      - 单纯：子类覆盖父类方法
      - 利用：super调用父类原方法，再增加性能提升多样性
    - 关键字 super
      - 构造函数，父类变量初始化供子类使用，super()
      - 可在子类中区分调用父子方法
        - 父：super.函数名()
        - 子：this.函数名()
    - 只能单类继承，但支持多层继承（同人类继承体系）
    - 避免重要类不被继承重写：关键字 final
      - 实现：final int age
      - final修饰基本类型变量，只能赋值一次
      - final修饰引用类型变量，不能改变地址
  - 多态
    - 理解：花木兰替父从征，父类的引用（花弧这个名字）指向了子类对象（花木兰）,,
      我访问花弧的属性，即看到了花木兰的武艺。但是我不能访问花弧没有的属性。
    - 意义
      - 借由父类实际访问子类的方法，利于方法调用
      - 不管我创建了多少个对象，都能通过父类直接调用子类的方法，体现代码的简便性。
    - 实现：父类 name = new 子类对象，Animal cat = new Cat();
    - 多态向上转型默认找爸爸：子类不能有自己的静态方法
    - 多态向下转型
      - 父类引用时，不能有自己的静态方法：我要做回我自己，调用自己的方法
      - 实现：Cat newname = (Cat ) cat
      - 优化，避免认错自己，c instanceof Cat，来判断强制转化有没有把类别搞错
    - 总结：编译看左边，运行看右边
- abstract抽象类
  - 抽象类
    - 定义：如图形、动物，没有具体无法定义面积怎么算、吃什么
    - 创建： abstract class name{ }
    - 不能new 抽象类，通过子类继承抽象类new，你创建图形我咋知道你要啥图形?
  - 抽象方法
    - 定义：给出某特性的一个框架，留具体的对象具体去填，如面积计算
    - 创建：public abstract void name()
    - 子类继承抽象父类必须重写父类所有抽象方法，不然报错，填具体实现
    - 抽象子类可以有普通方法
- 接口
  - 定义：实现了对方法的封装，可包含：抽象方法、静态方法、默认方法、私有方法
  - 意义：接口和实现分离，面向接口编程，只需知道接口能实现什么。代码依赖性低，
    可移植性高
  - 实现
    - 创建接口文件，public interface name{ 方法; },
    - 创建Implement类
      - 创建：新建以nameImpl为类名的类，重写方法，与继承要求类似。
      - 实现：class nameImpl implements name{重写方法;}
      - mian方法里新建对象，nameImpl 对象name = new nameImpl();
    - 接口实现类的使用
      - 直接当作你定义的普通类使用、调用方法
      - 实例化对象，nameImpl test = new nameImpl{ }
    - 方法实现
      - 抽象方法
        - 与继承类似
        - 在implements里面重写
      - 默认方法
        - 与成员方法类似
        - 在接口文件里定义，public default void name(){}
        - 在Implements里面先new 创建对象，不修改直接调用
        - 或者如abstract一样重写去掉default， public void name(){}
      - 静态方法
        - 创建：在接口文件中创建，public static void name(){}
        - 实现：在Implements文件中不能修改，只能通过接口名称调用
        - 方法调用：接口name.方法name()
      - 私有方法
  - 多接口继承，子类继承多接口
    - 多接口继承实现：public class C implements 接口A,接口B{ }
    - 继承父类多接口：public class C extends D Implements A,B
    - 多接口的abstract重名无影响，但是default会就近覆盖