

包：

命名规则：智能包含数字字母下划线小圆点，但是不能用数字开头，不能是关键字与保留字

引入包：import 包

注意：

PkgDetail.java

1. package 的作用是声明当前类所在的包，需要放在类的最上面，一个类中最多只有一句package
2. import指令 位置放在package的下面，在类定义前面,可以有多句且没有顺序要求。

访问修饰符：

- 1) 公开级别:用 public 修饰,对外公开
- 2) 受保护级别:用 protected 修饰,对子类和同一个包中的类公开
- 3) 默认级别:没有修饰符号,向同一个包的类公开.
- 4) 私有级别:用 private 修饰,只有类本身可以访问,不对外公开.

- 1) 修饰符可以用来修饰类中的属性，成员方法以及类
- 2) 只有默认的和public才能修饰类！，并且遵循上述访问权限的特点。

面向对象编程有三大特征：封装、继承和多态。

封装：

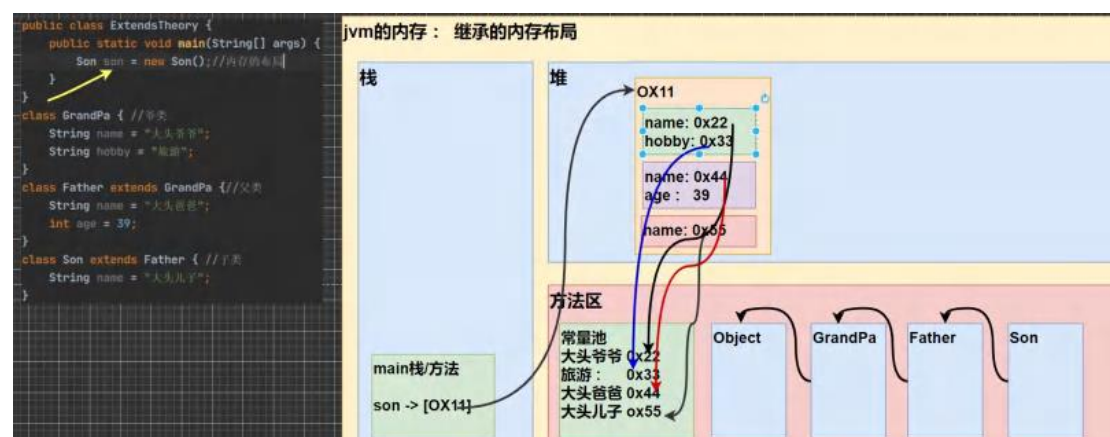
- 1) 将属性进行私有化private 【不能直接修改属性】
- 2) 提供一个公共的(public)set方法，用于对属性判断并赋值
public void setXxx(类型 参数名){ //Xxx 表示某个属性
 //加入数据验证的业务逻辑
 属性 = 参数名;
}
- 3) 提供一个公共的(public)get方法，用于获取属性的值
public 数据类型 getXxx(){ //权限判断,Xxx 某个属性
 return xx;
}

继承：

- 1) 子类继承了所有的属性和方法，非私有的属性和方法可以在子类直接访问，但是私有属性和方法不能在子类直接访问，要通过父类提供公共的方法去访问
- 2) 子类必须调用父类的构造器，完成父类的初始化

- 3) 当创建子类对象时，不管使用子类的哪个构造器，默认情况下总会去调用父类的无参构造器，如果父类没有提供无参构造器，则必须在子类的构造器中用 `super` 去指定使用父类的哪个构造器完成对父类的初始化工作，否则，编译不会通过
 - 4) 如果希望指定去调用父类的某个构造器，则显式的调用一下：`super(参数列表)`
 - 5) `super` 在使用时，必须放在构造器第一行(`super` 只能在构造器中使用)
 - 6) `super()` 和 `this()` 都只能放在构造器第一行，因此这两个方法不能共存存在一个构造器
 - 7) java 所有类都是 `Object` 类的子类，`Object` 是所有类的基类.
 - 8) 父类构造器的调用不限于直接父类！将一直往上追溯直到 `Object` 类(顶级父类)
 - 9) 子类最多只能继承一个父类(指直接继承)，即 java 中是单继承机制。
- 思考：如何让 A 类继承 B 类和 C 类？ **【A 继承 B， B 继承 C】**
- 10) 不能滥用继承，子类和父类之间必须满足 `is-a` 的逻辑关系

继承本质分析：



`Super` 关键字：

`super` 代表父类的引用，用于访问父类的属性、方法、构造器

1. 访问父类的属性，但不能访问父类的private属性 [案例]
super.属性名;
2. 访问父类的方法，不能访问父类的private方法
super.方法名(参数列表);
3. 访问父类的构造器(这点前面用过):
super(参数列表);只能放在构造器的第一句，只能出现一句!

No.	区别点	this	super
1	访问属性	访问本类中的属性，如果本类没有此属性则从父类中继续查找	从父类开始查找属性
2	调用方法	访问本类中的方法,如果本类没有此方法则从父类继续查找.	从父类开始查找方法
3	调用构造器	调用本类构造器，必须放在构造器的首行	调用父类构造器，必须放在子类构造器的首行
4	特殊	表示当前对象	子类中访问父类对象

方法重写:

1. 子类的方法的**形参列表,方法名称**,要和父类方法的**形参列表,方法名称**完全一样。【演示】
2. 子类方法的返回类型和父类方法返回类型一样，或者是父类返回类型的子类
比如 父类 返回类型是 Object ,子类方法返回类型是String 【演示】

```
public Object getInfo(){    public String getInfo(){
```
3. 子类方法不能缩小父类方法的访问权限 【演示】 public > protected > 默认>private

```
void sayOk(){    public void sayOk(){
```

名称	发生范围	方法名	形参列表	返回类型	修饰符
重载(overload)	本类	必须一样	类型，个数或者顺序至少有一个不同	无要求	无要求
重写(override)	父子类	必须一样	相同	子类重写的方法，返回的类型和父类返回的类型一致，或者是其子类	子类方法不能缩小父类方法的访问范围。

多态:

方法或对象具有多种形态。是面向对象的第三大特征，多态是建立在封装和继承基础之上的。

包括方法的多态与对象的多态。

方法的多态：方法重写、方法重载

对象的多态:

- (1) 一个对象的编译类型和运行类型可以不一致
- (2) 编译类型在定义对象时, 就确定了, 不能改变
- (3) 运行类型是可以变化的.
- (4) 编译类型看定义时 = 号的左边, 运行类型看 = 号的 右边

多态的前提是: 两个对象(类)存在继承关系

向上转型:

- 1) 本质: 父类的引用指向了子类的对象
- 2) 语法: 父类类型 引用名 = new 子类类型();
- 3) 特点: 编译类型看左边, 运行类型看右边。
可以调用父类中的所有成员(需遵守访问权限),
不能调用子类中特有成员;
最终运行效果看子类的具体实现!

向下转型:

- 1) 语法: 子类类型 引用名 = (子类类型) 父类引用;
- 2) 只能强转父类的引用, 不能强转父类的对象
- 3) 要求父类的引用必须指向的是当前目标类型的对象
- 4) 当向下转型后, 可以调用子类类型中所有的成员

动态绑定机制:

调用对象方法的时候, 该方法会和该对象的运行类型绑定。

调用对象属性的时候, 没有动态绑定机制, 哪里声明, 哪里使用

多态应用: 多态数组与多态参数。

Object 类:

HashCode 方法:

- 1) 提高具有哈希结构的容器的效率!
- 2) 两个引用, 如果指向的是同一个对象, 则哈希值肯定是一样的!
- 3) 两个引用, 如果指向的是不同对象, 则哈希值是不一样的
- 4) 哈希值主要根据地址号来的! 不能完全将哈希值等价于地址。

Tostring 方法:

1) 默认返回: 全类名+@+哈希值的十六进制,【查看 Object 的 toString 方法】

子类往往重写 toString 方法, 用于返回对象的属性信息

2) 重写 toString 方法, 打印对象或拼接对象时, 都会自动调用该对象的 toString 形式.

3) 当直接输出一个对象时, toString 方法会被默认的调用。

Finalize 方法:

1) 当对象被回收时, 系统自动调用该对象的 finalize 方法。子类可以重写该方法, 做一些释放资源的操作

2) 什么时候被回收: 当某个对象没有任何引用时, 则 jvm 就认为这个对象是一个垃圾对象, 就会使用垃圾回收机制来销毁该对象, 在销毁该对象前, 会先调用 finalize 方法。

3) 垃圾回收机制的调用, 是由系统来决定(即有自己的 GC 算法), 也可以通过 System.gc() 主动触发垃圾回收机制。

断点调试:

1. 断点调试是指在程序的某一行设置一个断点, 调试时, 程序运行到这一行就会停住, 然后你可以一步一步往下调试, 调试过程中可以看各个变量当前的值, 出错的话, 调试到出错的代码行即显示错误, 停下。进行分析从而找到这个Bug
2. 断点调试是程序员必须掌握的技能。
3. 断点调试也能帮助我们查看java底层源代码的执行过程, 提高程序员的Java水平。

F7(跳入)

F8(跳过)

shift+F8(跳出)

F9(resume, 执行到下一个断点)

F7: 跳入方法内

F8: 逐行执行代码.

shift+F8: 跳出方法