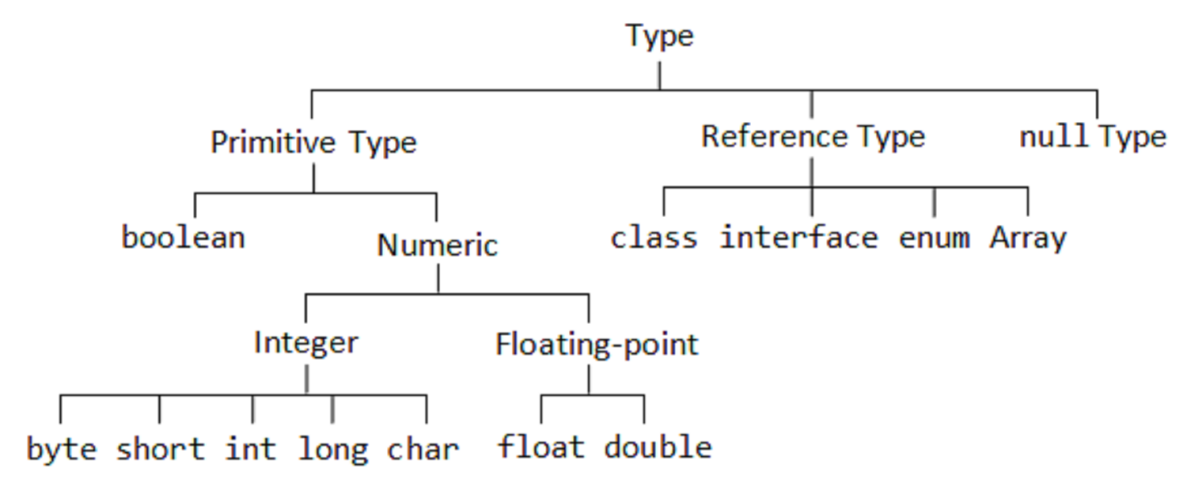
对象的内存空间在Heap区

Class是一种引用数据类型

Java Type类型的分类：



可变长参数的用法：

定义：

void showArgument(String... args){//可变长需放在最后面

for (int i=0;i < args.length; i++){

System.out.println("第"+(i+1)+"个参数为"+args[i]);

}

}

调用：

p.showArgument("Jack","Eton","Sunnie");

引用型

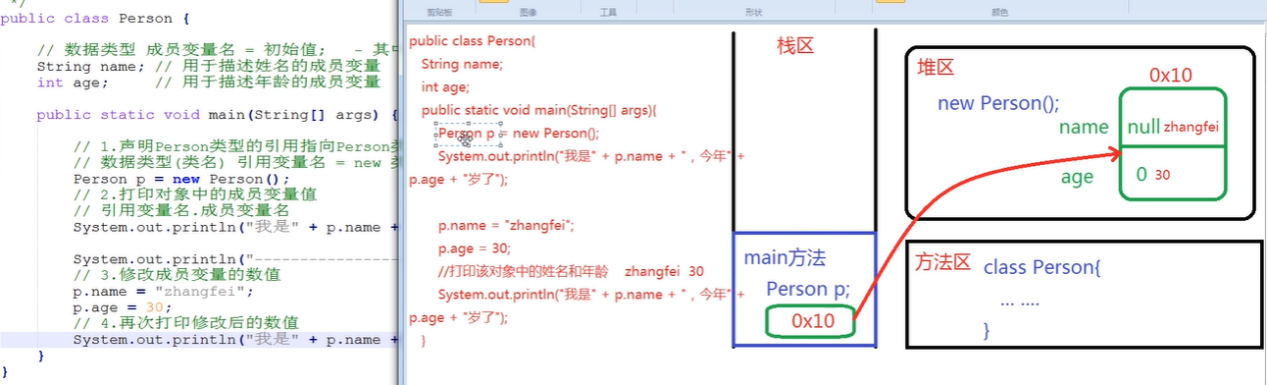
数据型成员变量默认值为0，boolean型成员变量默认值为false，引用类型成员变量默认值为null

java的引用类型只有三种，分别是类(class)、接口(interface)、数组。

栈区空间放对象变量名(对象的指针，即对象的引用)，堆区空间放对象的成员变量和方法

.class字节码文件在硬盘上，java person JVM加载字节码person.class到内存的方法区(非栈区Stack和堆区Heap)

栈区中为main方法申请一个空间(局部变量都在栈区)



Person p = new person();

栈区中分配空间给P 堆区中开辟空间给对象

地址指向对象的堆区空间

传参数的原理：

Int max(int ia, int ib){... ..}

~~~~~~~形参，max方法的局部变量，执行完后会释放空间，返回值会存到main方法 中的实参栈空间

Int a = 5; int b = 6; int res = max(a, b);

~~~~实参，main方法调用完毕，main方法中的变量栈空间也会释 放掉

Void show(int ia);{

}

引用变量调用成员方法

ArgumentTest at = new ArgumentTest ();

Int ib = 10;

at.show1(ib);

形参变量的数值改变，不影响实参变量

void show2(int[] arr1){

arr1[0] = 200;

System.out.println("In show2 method: arr1[0] = "+ arr1[0]);

}

....

int[] arr2 = new int[]{10,20};

at.show2(arr2);数组当形参时，传的是数组地址

arr1 = new int[2]; //每New一个新数组，就在堆区中新开辟一个内存空间存放

每调用一次方法，就开辟一个栈帧

Method Overload方法重载

构造方法中出现This，代表当前正在创建的对象

成员方法中出现This，代表当前正在调用的对象

this作为方法的返回值--Java源码中用得很多

this 就是本类类型的引用，是当前调用对象的本身

构造函数不允许用this关键字递归定义（即不允许当前方法内部调用自己）

Javac会把被依赖的类也自动一起编译掉

使用Private关键字可以把成员变量私有化，从而封装对象的数据。

类的构造块(Construction Block)的执行顺序在构造方法之前，可以做一些对象的初始化准备工作，例如对成员变量的统一初始化操作（所有对象都一样的初始化操作）。

静态代码块（只会先于创建对象执行一次，随着JVM对类的第一次加载就准备就绪）

static {

。。。。。

}

执行顺序：静态代码块--->类非静态代码块--->对象构造函数

私有化构造方法实现Singleton

对外只提供一个对象的类，成为单例类Singleton Class，应用场景：Windows任务管理器

三个步骤：

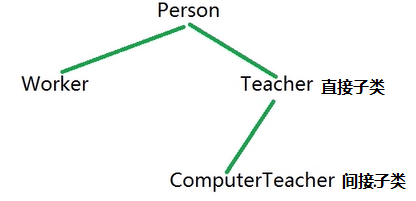
1. 私有化成员变量，私有化构造方法
2. 声明引用指向本对象
3. 提供public static公有get方法把对象return出去

//private static Singleton sin = new Singleton();//类层级,饿汉式，推荐饿汉式，目的是去抢占线程资源

private static Singleton sin = null;//类层级,懒汉式

子类初始化时，会调用父类的无参构造方法，相当于编译器在子类中自动运行super();

不同于C++，一个子类(儿子)只能有一个父类(父亲)



方法Overload和方法Override(重写)

类加载到方法区

对象New在Heap区

变量New在Stack区

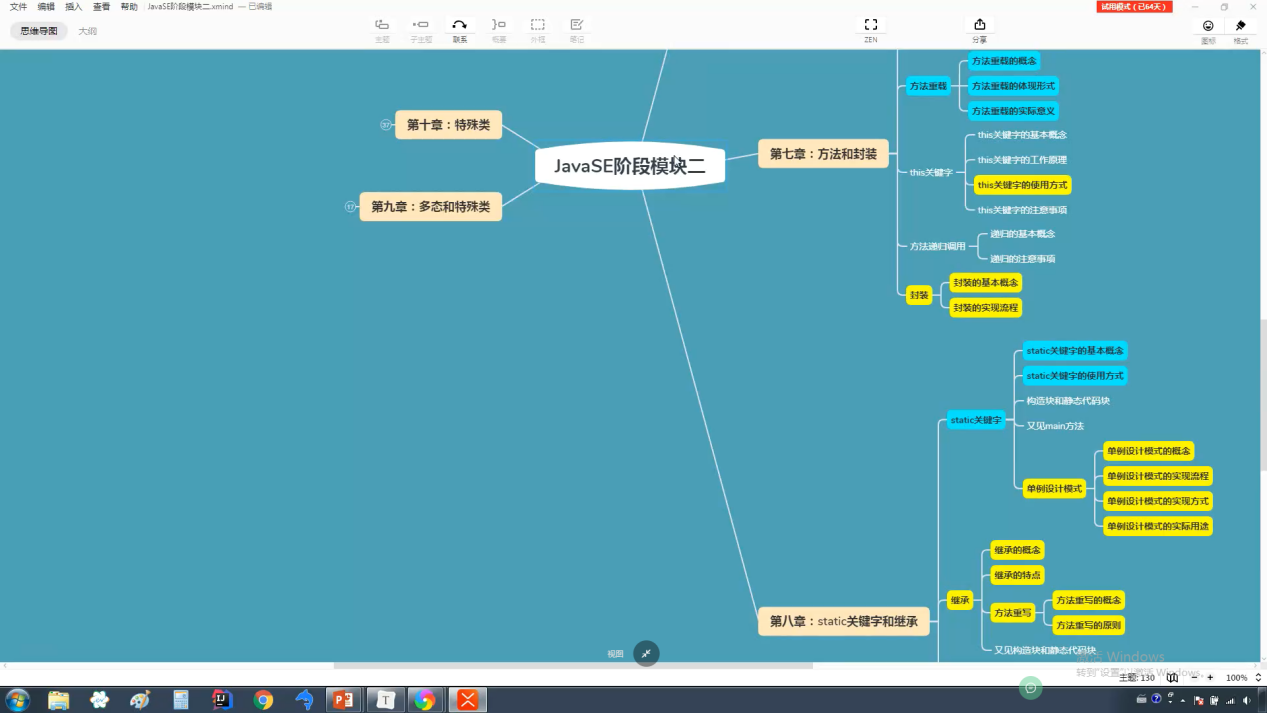
先加载父类，再加载子类--->构造父类对象，再构造子类对象

Public成员在任意位置使用，Private成员在本类使用，public修饰方法，Private修饰成员变量

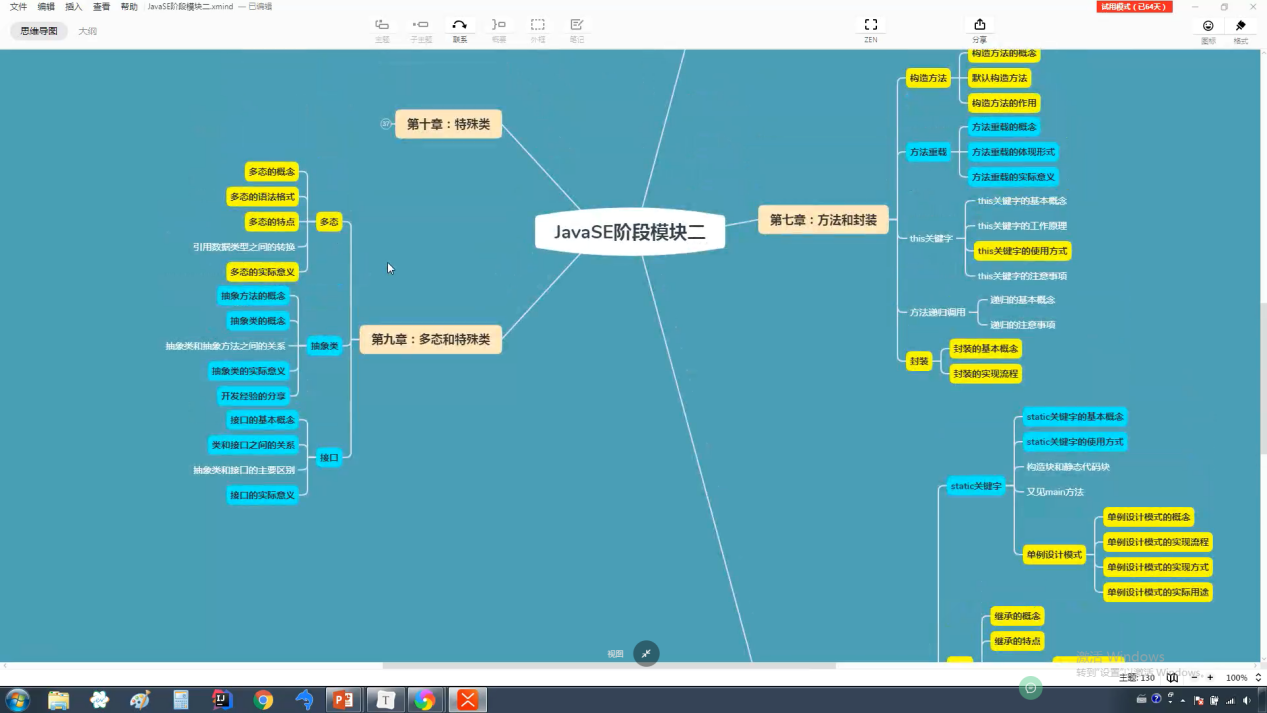
Final关键字修饰的方法不能被子类override，但是可以继承使用。

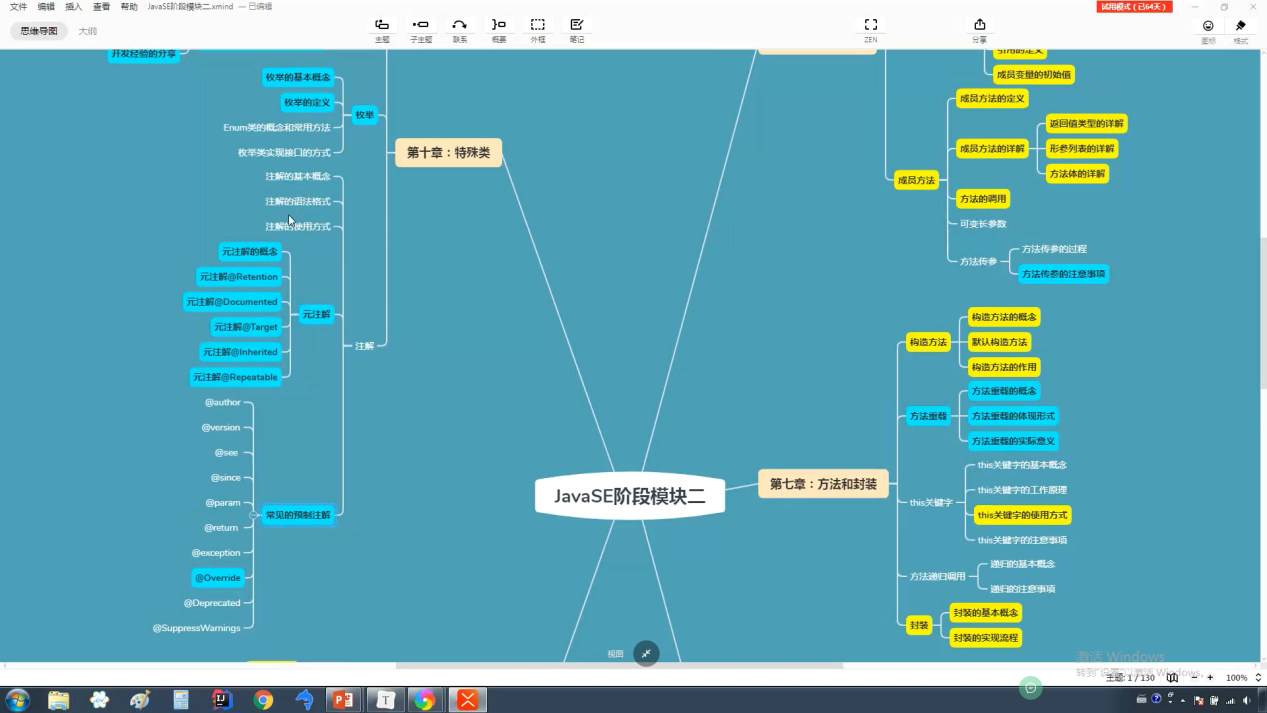
Final成员变量初始化后不能被再次赋值。通常用public static final修饰的变量作为常量.

常用访问控制符：



注解和枚举考得较少，枚举可能在面试和项目中用得较多，注解在框架中用得较多





父类引用可以调用子类对象

父类中继承下的方法不满足要求时，子类可以重写Override父类方法

//父类引用调用子类对象，编译阶段调用父类的方法，运行阶段调用子类的方法

Shape sr = new Rect(7,8,9,10); //一直披着羊皮（Shape外形）的狼（实际内容是Rect)

Ctrl-shif-/选择内容全文注释

父类引用可以调用父类独有方法

父类引用不可以直接调用子类方法

父子类都有的非静态方法，编译阶段调父类方法，运行阶段调子类重写Override的版本

但父类的静态方法不能被子类重写，不建议对象调用，static方法都调用父类版本

多态用法之一：父类类型的引用指向子类类型的对象

Abstract Class抽象类的抽象方法无方法体，所以不能New对象，它的意义是作为模板，被子类继承。

不能New对象，其余和普通类一致

抽象类的引用只能指向子类的对象，多态的应用，可以调用多个子类的功能，实现一个父类对象带来多种子类的功能

AbstractTest at = new SubAbstractTest();//抽象类的引用只能指向子类的对象

at.show();

子类不能继承父类的私有方法，所以private abstract无意义

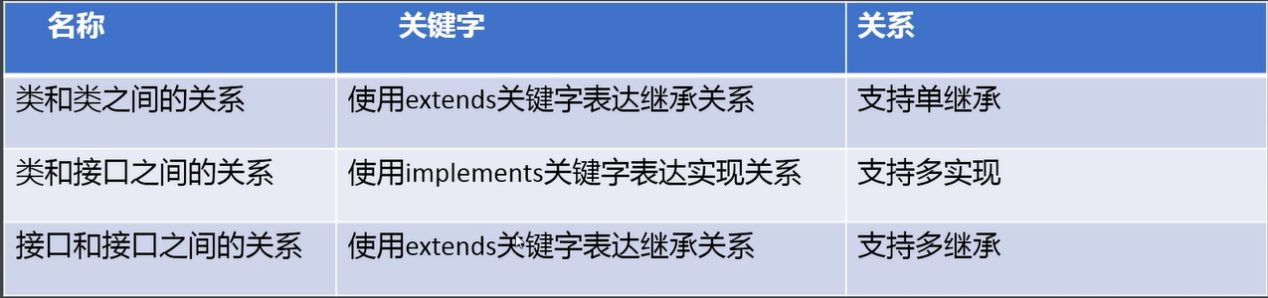
Static类不能new对象，而abstract方法只能通过类层级调用

**接口：所有的方法都是抽象方法**

JDK1.9以后允许接口中有私有方法

public class Gold implements Metal, Money{ //用多实现变相实现多继承

接口只能继承接口，不能继承普通类



public default void show1(){//方法体中是接口的默认实现

};

表示功能的方法可以用静态方法

public static Direction d1 = new Direction("向上");

枚举类常用方法：



打印引用变量时会自动调用ToString()方法