1.cd进入文件

2.cls清屏

3.exit退出

4.cd..退一级

5.cd\返回根

6.c：更换盘

dir显示目录类容

chcp 65001将字符集设为 UTF-8

7.D:\jdk-17.0.6\bin\java -version

配置环境后：java -version

8.jshell

(1)/help

(2)/exit

(3) System.out.println("Hello,world") //有回车

System.out.print("Hello,world") //无回车

（$1) //变量

9. JVM //Java虚拟机

JRE //Java运行环境

JDK //Java开发环境

10.源文件.java->(javac编译)->A.class->JVM

java源码->(编译器)->class字节码->(解释器)->机器码

11.javac -encoding utf-8 文件.java \\指定编码格式为utf-8

12. 单行注释 \\

多行注释 \\* \*\

文档注释 \\*\* \*\

（写在方法前对方法的参数，返回值进行说明，\\*\*+enter(快捷)）

13.IDE //集成开发环境

14.eclipse

alt + / //快捷打出后面代码

15.vscode

main + enter=psvm + enter=public static void main(String[] args){}

sout + enter=Sysetem.out.println();

16.项目-模块-包-类

17.snipaste F1截屏 F3将粘贴内容转换为图片

18. System.in.read() //会将字母转化成数字，并且变量名称是自动取的

var（Scanner） 名字 = new Scanner(System.in) //创建新的对象

var (String) 名字 = 对象.nextLine() //读取

//名字不能以数字开头，包含关键字等

19. $1+$2 //将两个字符串拼接

转义字符：\' \" \\ \n \t

\b: 123\b4=124 \\退了一格将3覆盖了

\r: 123\r4=423 \\退到头

java11后：文本块：“”“

”“” //其中不再需要转义字符了（三个双引号后必须有换行）

20.数字类型：整数：byte,short,int,long 小数：float,double //var为某种类型

字符：char 字符串：String //大写



类型 变量名 = 值

String 变量名 = 对象.nextLine()

int 变量名 = 对象.nextInt()

Float

Double

Byte

21.整数/0 ->报错 小数/0 ->Infinity(无穷大)

22. 对象.方法名（参数）

类名.方法名（参数） Math.pow(底数，指数)

23. \\*\*+enter //写在方法前对方法的参数，返回值进行说明

Ctrl+shift+enter //无论光标在哪都换行

//在括号左侧时移到最后并且自动加上；

Ctrl+/ //注释掉这一行，再一次取消注释

Alt+enter+鼠标点击“引入局部变量” //自动补上变量类型，变量名，等号，分号

Alt+enter+鼠标点击““反转条件”” //将条件语句反转

Ctrl+Alt+L //格式化代码（自动补上空格）

Ctrl+Alt+M //抽取成方法（或选中段落，右键重构，提取方法）

Alt+Shift+上下箭头 //移动代码

Ctrl+D //复制该行到下一行

Tab //补充剩余名称

Ctrl+N, //搜索要查找的类

Ctrl+F //搜索要查找的词

Ctrl+R //替换词

Ctrl+Q, //显示选中方法信息

Ctrl+左键/Ctrl+B //前往声明或用法

Ctrl+F12 //显示类中所有的字段和方法

Shift+F6 //改名（会同时替换其他的所有）

Ctrl+O //补上子类重写方法

Ctrl+E //打开最近的文件

main+enter

fori+enter

数组名.fori+Enter //遍历数组

数组名.for+Enter //增强for循环遍历数组

Arrays.toString.sout //打印数组（会将数组转化成字符串）

24.（1）NumberFormat.getCurrencyInstance().format(数据) //将数据格式化为货币格式，默认为人民币，在第一个括号内修改地区（Locale.US），快捷方法：Nu+enter+get+enter+for+enter

（2） throw new 异常类型（“提示信息”）

throw new IllegalArgumentException()； //结束并报错括号内内容，(参数异常)快捷：thr+enter+I+enter

25.javadoc(javadocument):Ctrl+N,再搜索要查找的类

进入类后，右键翻译，Ctrl+F12,显示该类的方法

或在源文件中选中方法，再Ctrl+Q,直接显示该方法信息

26.与c区别：（1）if（1）在c中正确，在java中错误，因为这是int类型，而需要bool类型的参数

27.if-else语句优化：if(成立条件){\*\*；}else{\*\*；}=if（！成立条件）{\*\*；return；}\*\*；

28.方法内创建的对象只在方法内生效

29.创建package(软件包)：

Ctrl/Shift/Ctrl+Shift //多选拖入

com.\*\*\* //com文件下\*\*\*

（）

30.导入不同包的类：1.import 包.类 //java.lang包下不需要导入

后面每个写 类.方法

2.每个写 包.类.方法

导入同一包的类： 类.方法

31.jar（打包):文件-项目结构-工件-加号-jar-来自-模块-主类，构建-构建工件，左边的 artifact文件下

32.cmd终端运行：（1）cd 目录 （2）java -jar jar包名（包名中的空格要用双引号）（Tab）

33.C/S架构（Client/Server)(用浏览器的也叫B/S):

发送请求目的地URL（统一资源定位服）：协议://主机[:端口][/路径][？查询参数]

（1）https协议:加密，端口为443时端口可省略

（2）http协议：不加密，端口为80时端口可省略

（3）localhost主机：本机

34.开源框架SpringBoot

(1)官网：start.spring.io

(2)阿里云版：start.aliyun.com+java工程师脚手架

(3)项目基本信息：group:com.\*\*\* artifact:名称 组件依赖：Spring Web

(4)阿里云旧版修改：解压后找到pom.xml,拖到idea中打开

<java.version>??-><java.version>17

<spring-boot.version>???-><spring-boot.version>2.7.0

<source>??-><source>17

<target>??-><target>17

<version>??-><version>2.7.0

(5)Maven修改：idealC-2022.1.win\plgins\maven\lib\maven3\conf\settings.xml,拖到idea中打开

<mirror>

<id>central</id>

<mirrorOf>central</mirrorOf>

<url>https://maven.aliyun.com/repository/public</url>

</mirror>

加到网址后-->的下一排

（6）复制文件到project1目录下，文件-项目结构-模块-导入模块-选择project1下的pom.xml

或者直接在project1下找到pom.xml+右键添加为maven项目

（7）在src-main-java-com.\*\*下创建一个与\*\*Application同级（或下级）的类

类外：@Controller 类里：

@RequestMapping("/Hello")

@ResponseBody

String hi() {

return "Hello,lhx";

}

@RequestMapping("/hello")

@ResponseBody

public String hello(@RequestParam(name = "name", defaultValue = "unknown user") String name) {

return "Hello " + name;

}

34+.快速创建SpringBoot:新建Maven模块，复制网站中的pom到此模块中的pom（还要替换其中的所有module为新建的那个模块的名字），在模块下src下的java下新建包，包下新建类（作入口方法），加@SpringBootApplication ,main函数及其下的方法SpringApplication.run(类名.class,args);

35.js字符串拼接：`a=${变量}&b=${变量}`

36.数组：类型[] 变量名=new 类型[长度]

类型[] 变量名=new 类型[]{元素1，元素2，......}

类型[] 变量名={元素1，元素2，......}

类型[][] 变量名=new 类型[行数][列数]

数组名.length

38.报告具体错误信息：resources-application.properties

server.error.include-message=always

39.打包：右上角maven-package双击，包在target下

40.（1）类：是对象的模板，用字段描述对象将来长什么样，用构造给字段变量赋值

（2）class 类{

字段；

构造方法（参数）{

} //构造方法是给对象的字段赋初始值，和类同名，不加返回值类型声明；也可以无参构造：构造方法（）{}，也可以直接不写

（static）返回值类型 方法名（参数）{

}

} //控制对象的行为

（3）创建对象：new 构造方法名 （参数）；

\*等同类名\* \*按照类的字段\*

41.this用法：当想要将构造方法名取为和字段参数名一样时，为了区分，

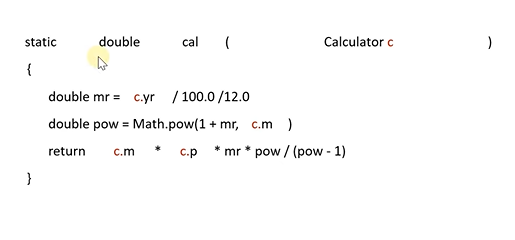
this.名字 表示字段参数名

名字 表示构造方法名

即构造方法中即可这样赋值：this.名字=名字

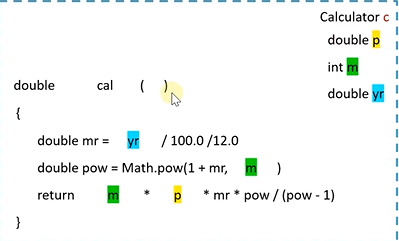
42.构造方法快速生成：右键生成-构造方法/Alt+Insert

43.static方法：数据全部来源于方法参数，调用时写 方法名（）



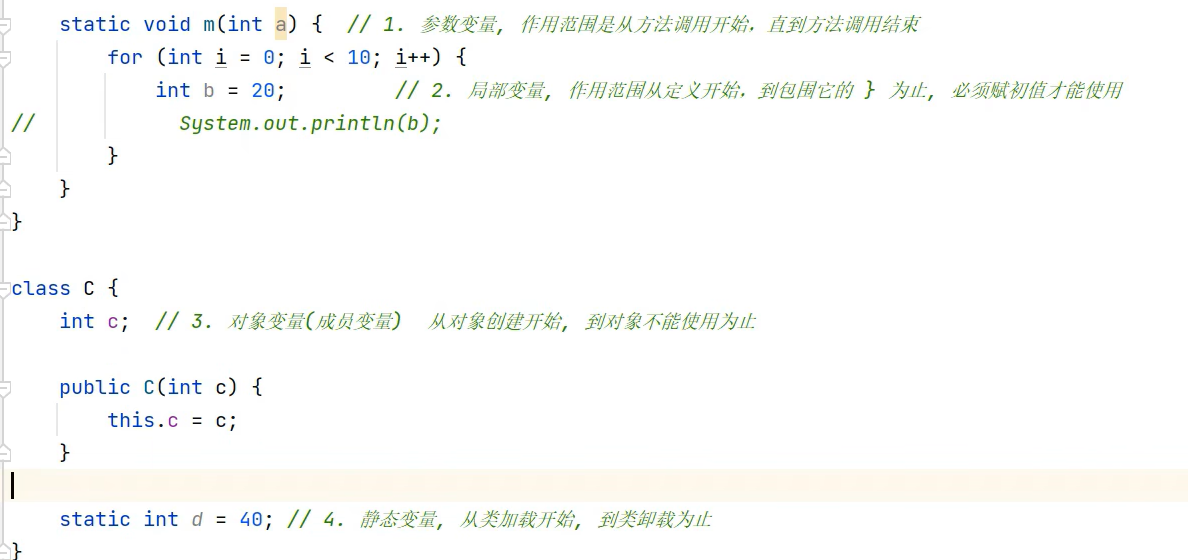
Calculator是对象

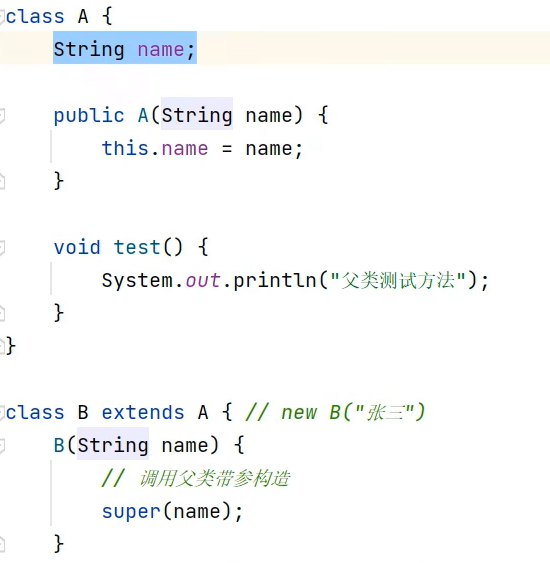
对象方法：数据来源于方法参数和对象内部，调用时写 对象.方法名()

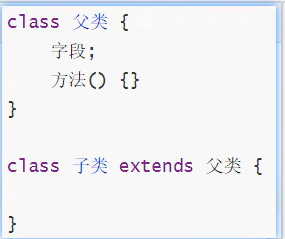
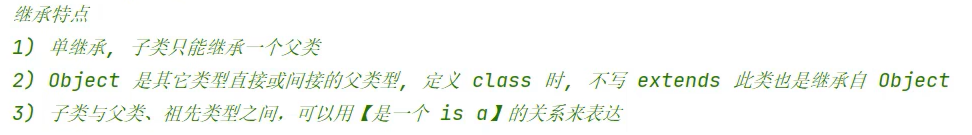
44.静态变量static:类中所有对象共享的变量，访问时最好用 类名.变量名（而不用对象名.变量名）

加final表示不可修改

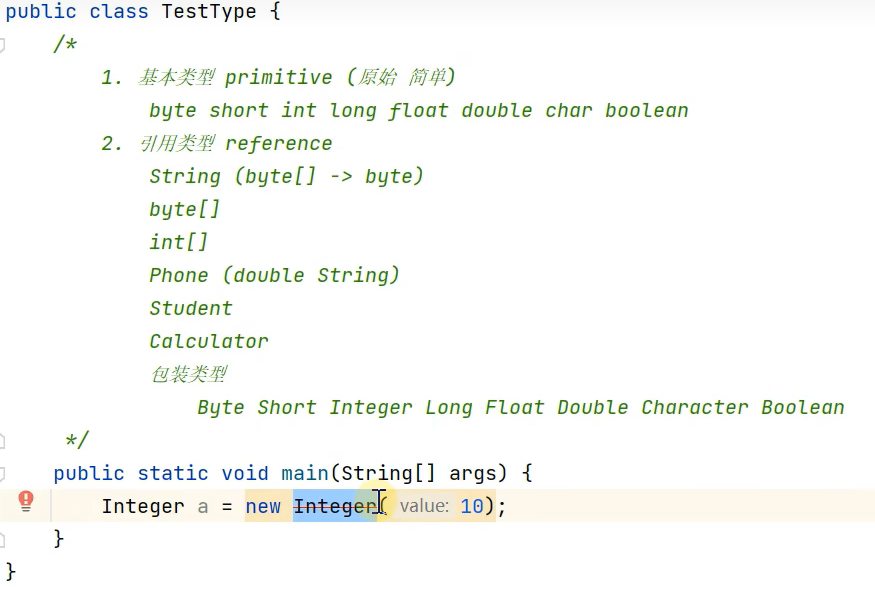
四种变量

1. 继承：子类可继承父类的字段和方法





46.两种类型

默认值：

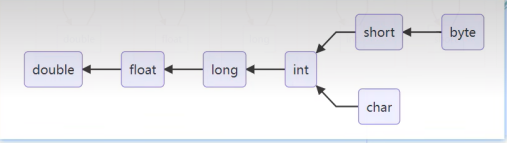
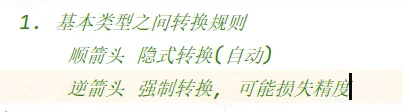
基本类型：对int数组，默认值为0，对double数组，默认值为0.0，对boolean数组，默认值为false

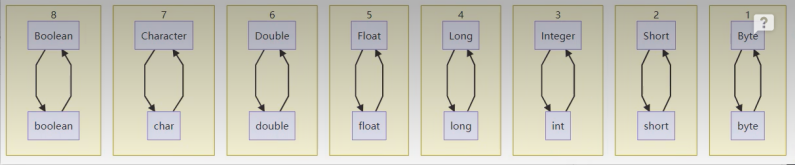
引用类型：默认值为NULL，例如String，数组，创建的对象，包装类型

引用类型有对象的性质，可以创建对象，引用方法（如创建了Student对象s1，可s1.name.length(),其中s1.name是String类型）

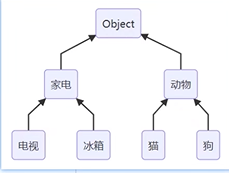
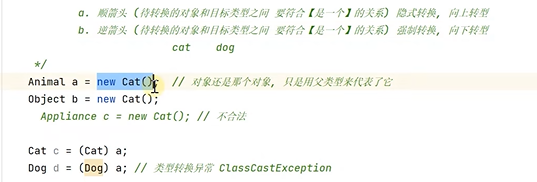
Snipaste_2023-04-12_17-26-17

1. 类型转换

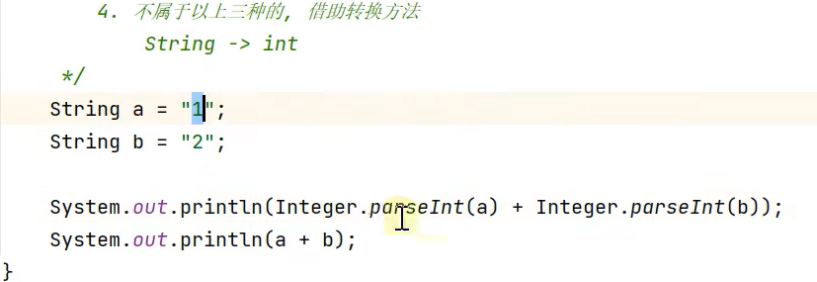


Snipaste_2023-04-12_17-51-18会互相隐式（自动）转换

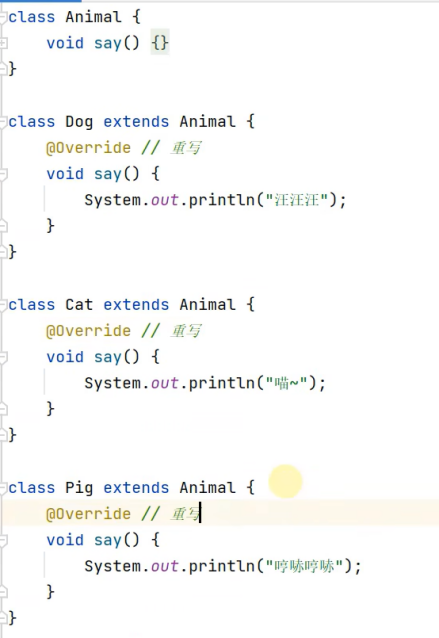
Snipaste_2023-04-12_18-21-03



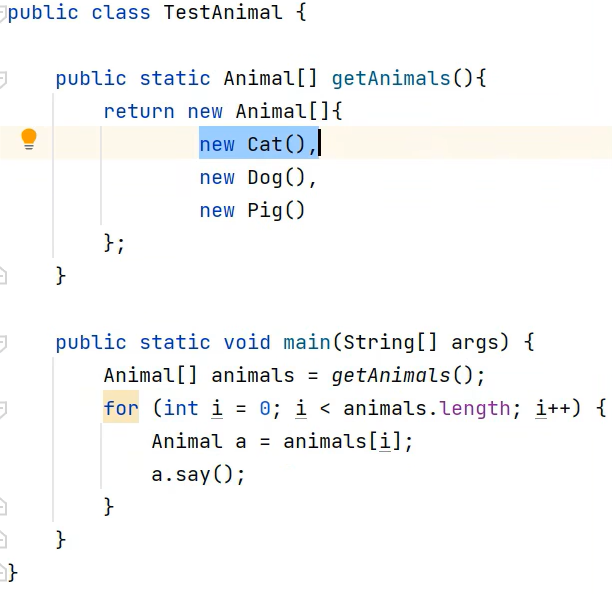
Snipaste_2023-04-12_18-44-18

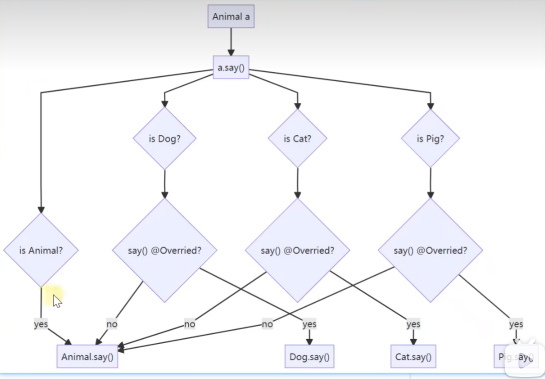


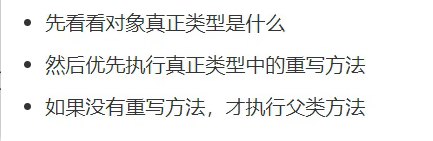
方法Integer.parseInt()将其他类型转化成int

1. 多态



@Override用于检查方法重写是否正确

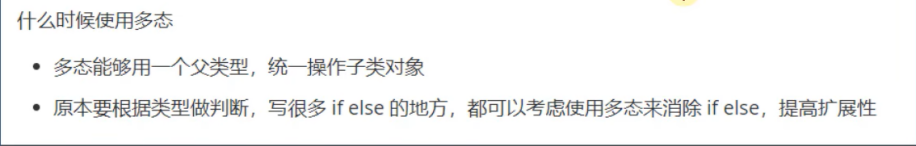




看真正类型，如Dog用了Animal代表，但它还是Dog

Alt+Enter补上子类构造方法

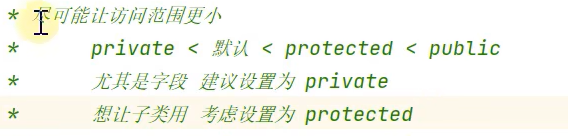
Ctrl+O 补上子类重写方法



48.封装

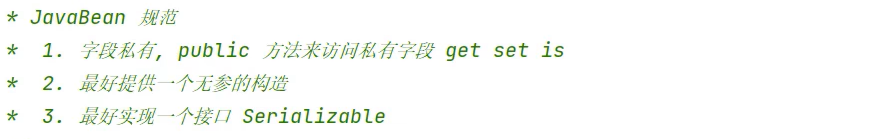
方法访问修饰符

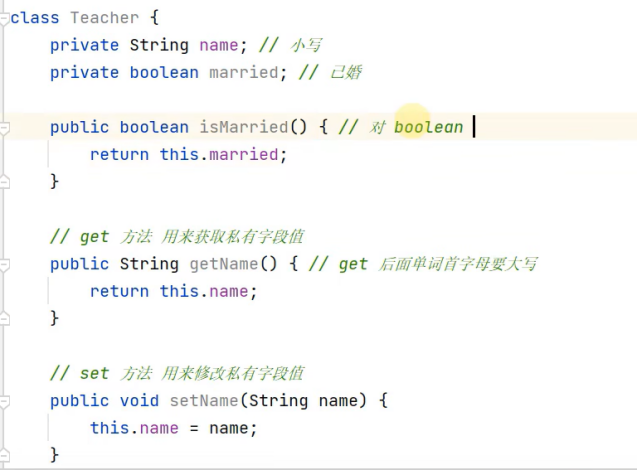




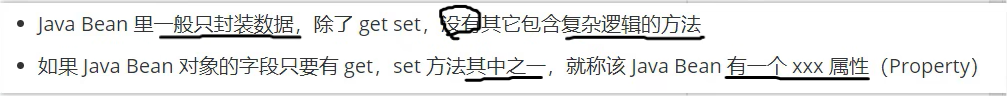
注：一个java类源文件中只能有一个类被public修饰，且类名必须与类源文件同名

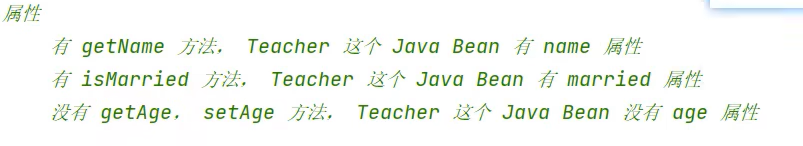
49.javabean





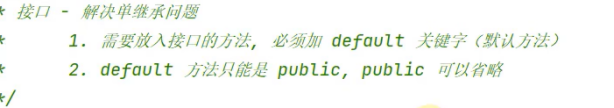
快捷：(右键+生成/Alt+Insert)+Setter+Getter

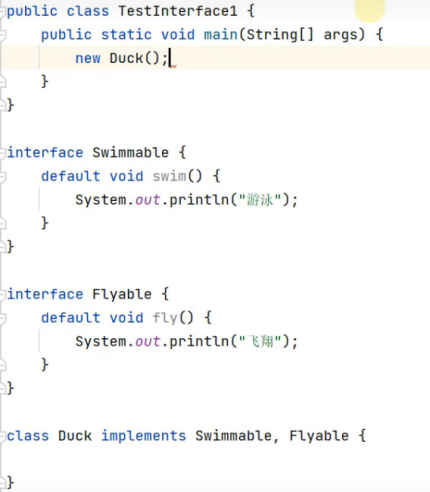
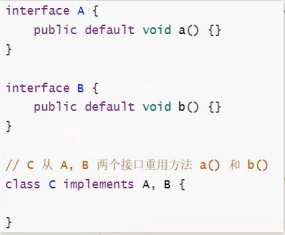




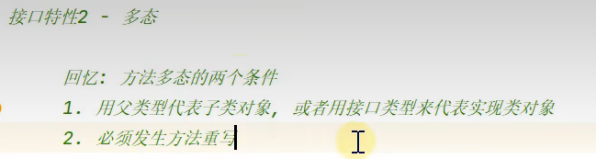
50.接口

1.默认方法





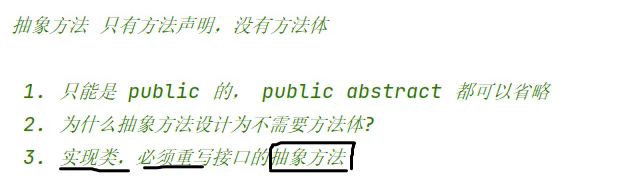
2.多态



屏幕截图 2023-04-23 085913

因为接口中的abstract,default方法只能是public，故实现类重写方法必须加public

3.抽象方法



屏幕截图 2023-04-23 091007

4.封装



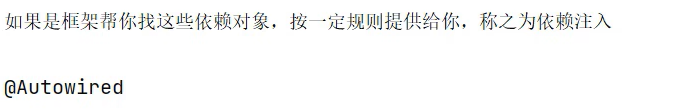
当用接口代表实现类时，无法通过接口访问实现类中的字段和非重写方法

父子类好像也是

51.Spring控制反转



(1)依赖注入（自动生成Service对象，只需去找到即可（会根据对象类型自动匹配））



例：

@Service

public class Service0 implements CalculatorService{ }

@Controller

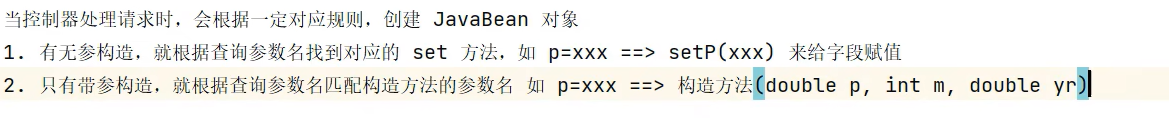
public class CalculatorController{

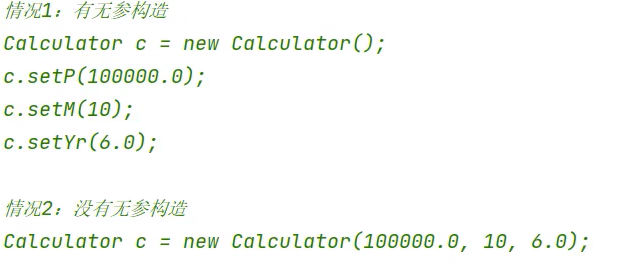
@Autowired

CalculatorService[ ] array;//等效于CalculatorService [ ] array=new 数组[ ]{new Service0,new Service1,new Service2}

}

（2）JavaBean自动生成，可直接使用





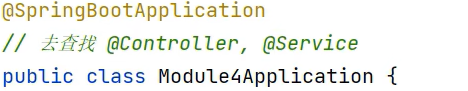
52.Spring包结构



控制器

JavaBean（dto：data transfer object）

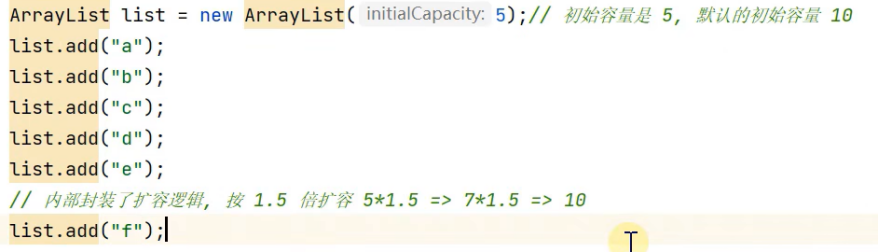
服务器{实现类，接口

入口方法

只能查找到与入口方法同级的范围

**53.核心类库-ArrayList**

（1）内容：解决数组不能自动扩容的问题

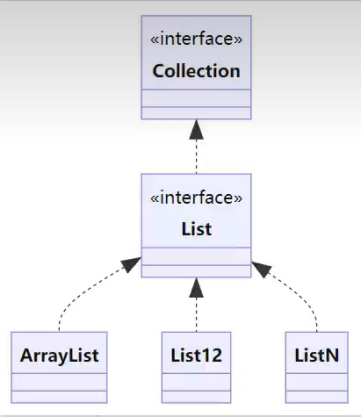


（2）内部数组数据类型：object数组 //object是所有引用类型的父类

（3）**泛型**：<>用于规定类型

ArrayList <Integer> list=new ArrayList<Integer>(5); //后面的<>中的可以省略

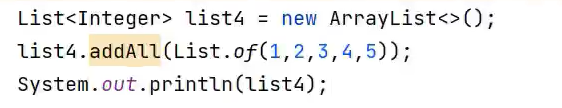
（4）List:可一次传入多个数据，但定义后不能再增加和删除数据

Snipaste_2023-07-11_16-57-53Snipaste_2023-07-11_16-57-23//List12是List数据数<=2时的数组数据类型，而ListN是>3

(5)两者结合让ArrayList实现加多个的操作

ArrayList的addAll方法：list1.addAll(list2);

其中addAll方法应传Collecton类型的参数（可用List向上转型）



54.debug

切换更详细的显示：在数据上右键，查看方式，object

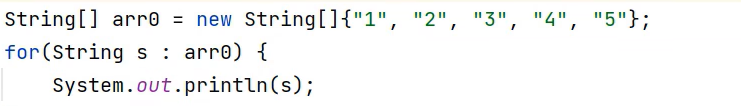
步过按钮：运行到下一行

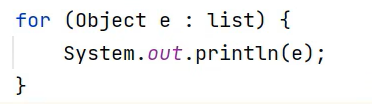
评估表达式按钮：在运行过程中动态调用方法

55.遍历：增强for循环

for(数据类型 新名：数组名) //也可以是Arraylist名

{ ....... }

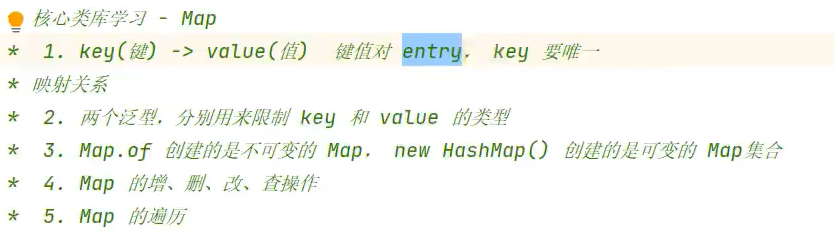


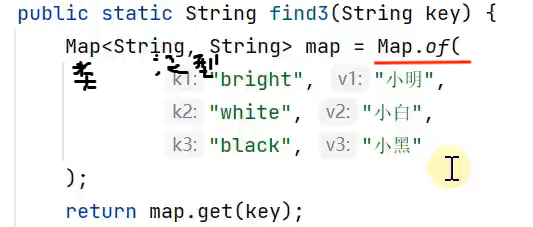


简法：数组名.for+Enter

Snipaste_2023-07-11_17-47-2456.字符串比较： 比较s和value字符串是否相同

Snipaste_2023-07-11_17-48-42

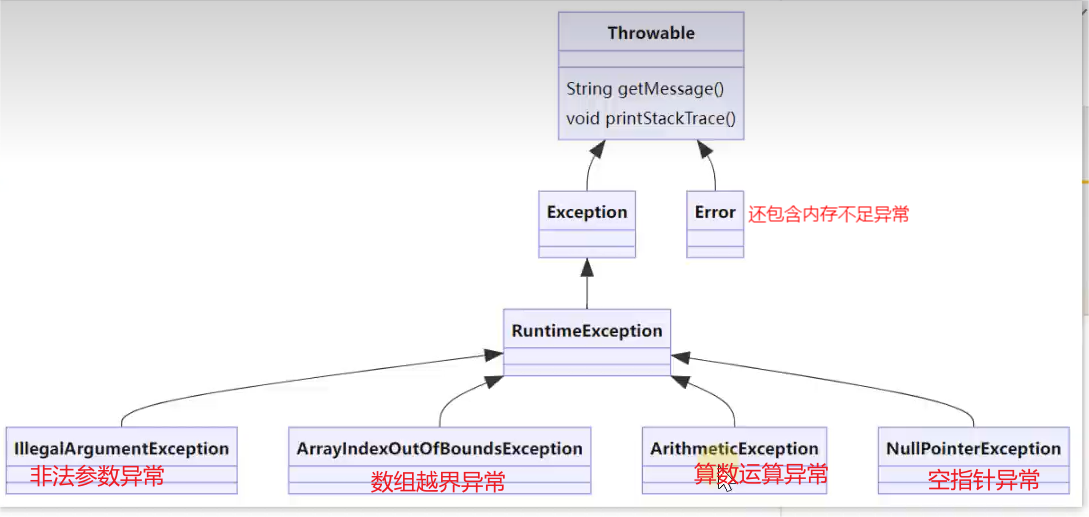
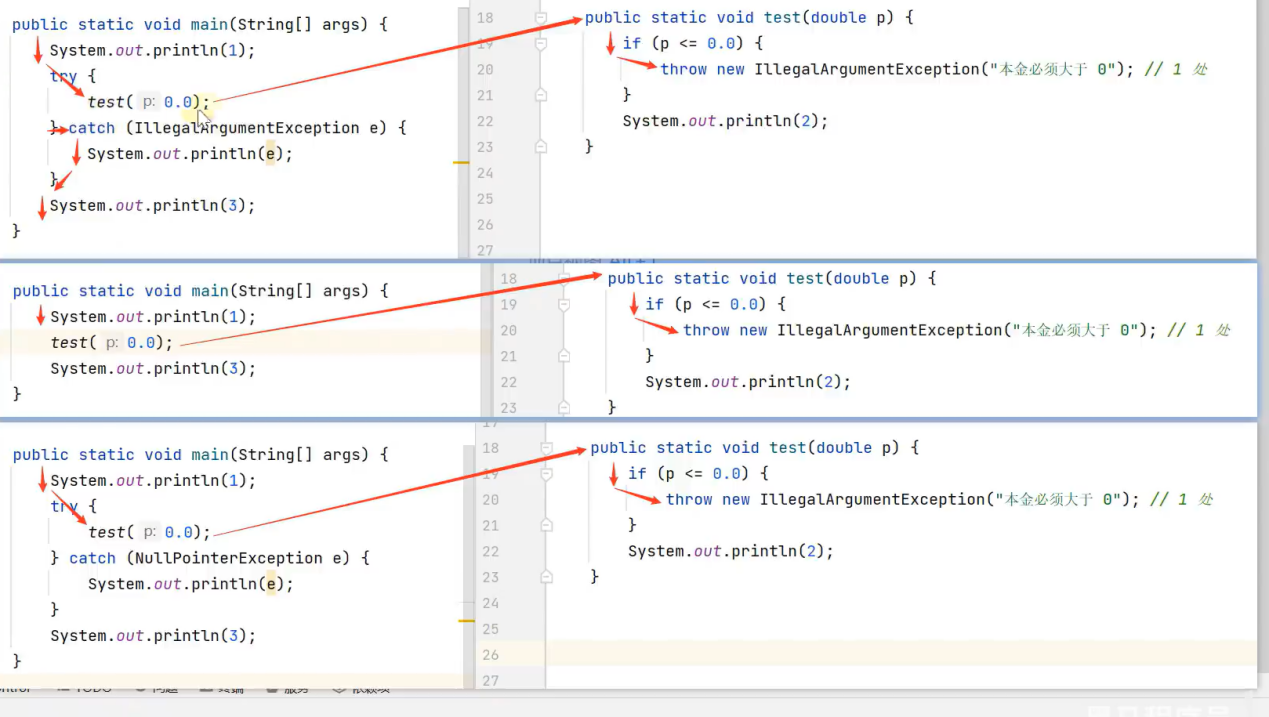
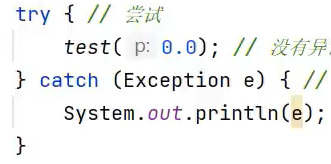
**57.核心类库-Map**





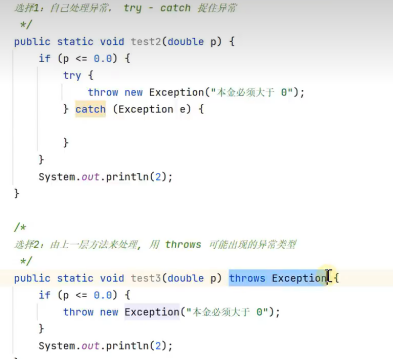
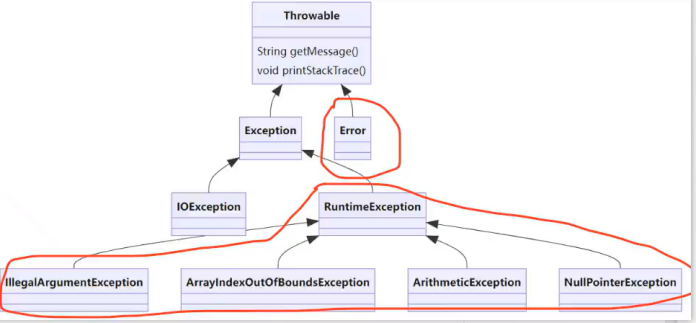
58.异常处理



yi

只要catch了它或其父类就行（一般用excption而非throwable）

59.编译异常



所有调用了test3的函数及向上的都要throw 异常

60.finally



简化语法

