

2022年Java工程师面试题目汇总(附答案详解)

2022-07-14 14:43:13 By 二维码防伪系统

1.什么是B/S架构？什么是C/S架构

B/S(Browser/Server)，浏览器/服务器程序

C/S(Client/Server)，客户端/服务端，桌面应用程序

2.你所知道网络协议有那些？

HTTP：超文本传输协议

FTP：文件传输协议

SMTP：简单邮件协议

TELNET：远程终端协议

POP3：邮件读取协议

3.Java都有那些开发平台？

JAVASE：主要用在客户端开发JVAEE：主要用在web应用程序开发JAVAME：主要用在嵌入式应用程序开发

4.什么是JVM？Java虚拟机包括什么？

JVM：Java虚拟机，运用硬件或软件手段实现的虚拟的计算机，Java虚拟机包括：寄存器，堆栈，处理器

5.Java是否需要开发人员回收内存垃圾吗？

大多情况下是不需要的。Java提供了一个系统级的线程来跟踪内存分配，不再使用的内存区将会自动回收

6.什么是JDK？什么是JRE？

JDK：Javadevelopmentkit：Java开发工具包，是开发人员所需要安装的环境

JRE：Javaruntimeenvironment：Java运行环境，Java程序运行所需要安装的环境

7.什么是数据结构？

计算机保存，组织数据的方式

8.Java的数据结构有那些？

线性表(ArrayList)链表(LinkedList)栈(Stack)队列(Queue)图(Map)树(Tree)

9.什么是OOP？

面向对象编程

10.什么是面向对象？

世间万物都可以看成一个对象。每个物体包括动态的行为和静态的属性，这些就构成了一个对象。

11.类与对象的关系？

类是对象的抽象，对象是类的具体，类是对象的模板，对象是类的实例

12.Java中有几种数据类型

整形：byte,short,int,long浮点型：float,double字符型：char布尔型：boolean

13.什么是隐式转换，什么是显式转换

显示转换就是类型强转，把一个大类型的数据强制赋值给小类型的数据;隐式转换就是大范围的变量能够接受小范围的数据;隐式转换和显式转换其实就是自动类型转换和强制类型转换。

14.Char类型能不能转成int类型？能不能转化成string类型，能不能转成double类型

Char在Java中也是比较特殊的类型，它的int值从1开始，一共有2的16次方个数据;Char<int<long<float<double;Char类型可以隐式转成int,double类型，但是不能隐式转换成string;如果char类型转成byte，short类型的时候，需要强转。

15.什么是拆装箱？

拆箱：把包装类型转成基本数据类型装箱：把基本数据类型转成包装类型

16.Java中的包装类都是那些？

byte：Byte short：Short int：Integer long：Long float：Float double：Double char：Character boolean：Boolean

17.一个Java类中包含那些内容？

属性、方法、内部类、构造方法、代码块。

18.例如：if(a+1.0=4.0)，这样做好吗？

不好，因为计算机在浮点型数据运算的时候，会有误差，尽量在布尔表达式中不使用浮点型数据(if,while,switch中判断条件不使用浮点型)

19.那针对浮点型数据运算出现的误差的问题，你怎么解决？

使用BigDecimal类进行浮点型数据的运算

20.++i与i++的区别

++i：先赋值，后计算 i++：先计算，后赋值

21.程序的结构有那些？

顺序结构选择结构循环结构

22.数组实例化有几种方式？

静态实例化：创建数组的时候已经指定数组中的元素，

```
int[] a=new int[]{1,3,3}
```

动态实例化：实例化数组的时候，只指定了数组程度，数组中所有元素都是数组类型的默认值

23.Java中各种数据默认值

Byte,short,int,long默认是都是0 Boolean默认值是false Char类型的默认值是" Float与double类型的默认是0.0 对象类型的默认值是null

24.Java常用包有那些？

Java.lang Java.io Java.sql Java.util Java.awt Java.net Java.math

25.Java最顶级的父类是哪个？

Object

26.Object类常用方法有那些？

Equals Hashcode toString wait notify clone getClass

27.Java中有没有指针？

有指针，但是隐藏了，开发人员无法直接操作指针，由jvm来操作指针

28.Java中是值传递引用传递？

理论上说，Java都是引用传递，对于基本数据类型，传递是值的副本，而不是值本身。对于对象类型，传递是对象的引用，当在一个方法操作操作参数的时候，其实操作的是引用所指向的对象。

29.假设把实例化的数组的变量当成方法参数，当方法执行的时候改变了数组内的元素，那么在方法外，数组元素有发生改变吗？

改变了，因为传递是对象的引用，操作的是引用所指向的对象

30.实例化数组后，能不能改变数组长度呢？

不能，数组一旦实例化，它的长度就是固定的

31.假设数组内有5个元素，如果对数组进行反序，该如何做？

创建一个新数组，从后到前循环遍历每个元素，将取出的元素依次顺序放入新数组中

32.形参与实参

形参：全称为“形式参数”，是在定义方法名和方法体的时候使用的参数，用于接收调用该方法时传入的实际值;实参：全称为“实际参数”，是在调用方法时传递给该方法的实际值。

33.构造方法能不能显式调用？

不能构造方法当成普通方法调用，只有在创建对象的时候它才会被系统调用

34.构造方法能不能重写？能不能重载？

可以重写，也可以重载

35.什么是方法重载？

方法的重载就是在同一个类中允许同时存在一个以上的同名方法，只要它们的参数个数或者类型不同即可。在这种情况下，该方法就叫被重载了，这个过程称为方法的重载(override)

36.内部类与静态内部类的区别？

静态内部类相对与外部类是独立存在的，在静态内部类中无法直接访问外部类中变量、方法。如果要访问的话，必须要new一个外部类的对象，使用new出来的对象来访问。但是可以直接访问静态的变量、调用静态的方法；

普通内部类作为外部类一个成员而存在，在普通内部类中可以直接访问外部类属性，调用外部类的方法。

如果外部类要访问内部类的属性或者调用内部类的方法，必须要创建一个内部类的对象，使用该对象访问属性或者调用方法。

如果其他的类要访问普通内部类的属性或者调用普通内部类的方法，必须要在外部类中创建一个普通内部类的对象作为一个属性，外同类可以通过该属性调用普通内部类的方法或者访问普通内部类的属性

如果其他的类要访问静态内部类的属性或者调用静态内部类的方法，直接创建一个静态内部类对象即可。

37.Static关键字有什么作用？

Static可以修饰内部类、方法、变量、代码块

Static修饰的类是静态内部类

Static修饰的方法是静态方法，表示该方法属于当前类的，而不属于某个对象的，静态方法也不能被重写，可以直接使用类名来调用。在static方法中不能使用this或者super关键字。

Static修饰变量是静态变量或者叫类变量，静态变量被所有实例所共享，不会依赖于对象。静态变量在内存中只有一份拷贝，在JVM加载类的时候，只为静态分配一次内存。

Static修饰的代码块叫静态代码块，通常用来做程序优化的。静态代码块中的代码在整个类加载的时候只会执行一次。静态代码块可以有多个，如果有多个，按照先后顺序依次执行。

38.Final在Java中的作用

Final可以修饰类，修饰方法，修饰变量。修饰的类叫最终类。该类不能被继承。修饰的方法不能被重写。修饰的变量叫常量，常量必须初始化，一旦初始化后，常量的值不能发生改变。

39.Java中操作字符串使用哪个类？

String，StringBuffer，StringBuilder

40.StringBuffer，StringBuilder有什么区别？

StringBuffer与StringBuilder都继承了AbstractStringBulder类，而AbstractStringBuilder又实现了CharSequence接口，两个类都是用来进行字符串操作的。

在做字符串拼接修改删除替换时，效率比string更高。

StringBuffer是线程安全的，Stringbuilder是非线程安全的。所以Stringbuilder比stringbuffer效率更高，StringBuffer的方法大多都加了synchronized关键字

41.Stringstr="aaa",与Stringstr=newString("aaa")一样吗？

不一样的。因为内存分配的方式不一样。第一种，创建的“aaa”是常量，jvm都将其分配在常量池中。第二种创建的

是一个对象，jvm将其值分配在堆内存中。

42.Stringstr="aa",Strings="bb",Stringaa=aa+s;一种创建了几个对象？

一共有两个引用，三个对象。因为"aa"与"bb"都是常量，常量的值不能改变，当执行字符串拼接时候，会创建一个新的常量是"aabbb",有将其存到常量池中。

43.将下Java中的math类有那些常用方法？

Pow()：幂运算Sqrt()：平方根Round()：四舍五入Abs()：求绝对值Random()：生成一个0-1的随机数，包括0不包括1

44.String类的常用方法有那些？

charAt：返回指定索引处的字符indexOf()：返回指定字符的索引replace()：字符串替换trim()：去除字符串两端空白split()：分割字符串，返回一个分割后的字符串数组getBytes()：返回字符串的byte类型数组length()：返回字符串长度toLowerCase()：将字符串转成小写字母toUpperCase()：将字符串转成大写字母substring()：截取字符串format()：格式化字符串equals()：字符串比较

45.判断两个对象是否相同，能使用equals比较吗？

不能。Equals大多用来做字符串比较，要判断基本数据类型或者对象类型，需要使用==

46.==与equals有什么区别？

\==可以判断基本数据类型值是否相等，也可以判断两个对象指向的内存地址是否相同，也就是说判断两个对象是否是同一个对象，Equals通常用来做字符串比较。

47.如何将字符串反转？

StringBuilder或者StringBuffer的reverse方法

48.面向对象的语言有那些特征？

封装、继承、多态

49.Java中的继承是单继承还是多继承

Java中既有单继承，又有多继承。对于Java类来说只能有一个父类，对于接口来说可以同时继承多个接口

50.什么是重写？什么是重载？

重载和重写都是Java多态的表现。

重载叫override，在同一个类中多态的表现。当一个类中出现了多个相同名称的方法，但参数个数和参数类型不同，方法重载与返回值无关

重写叫overwrite，是子类中多态的表现。当子类出现与父类相同的方法，那么这就是方法重写。方法重写时，子类的返回值必须与父类的一致。如果父类方法抛出一个异常，子类重写的方法抛出的异常类型不能小于父类抛出的异常类型。

51.构造方法能不能重载？能不能重写？

可以重载，必须重写

52.如果父类只有有参构造方法，那么子类必须要重写父类的构造方法吗？

必须重写

53.创建一个子类对象的时候，那么父类的构造方法会执行吗？

会执行。当创建一个子类对象，调用子类构造方法的时候，子类构造方法会默认调用父类的构造方法。

54.什么是父类引用指向子类对象？

是Java多态一种特殊的表现形式。创建父类引用，让该引用指向一个子类的对象

55.当父类引用指向子类对象的时候，子类重写了父类方法和属性，那么当访问属性的时候，访问是谁的属性？调用方法时，调用的是谁的方法？

子类重写了父类方法和属性，访问的是父类的属性，调用的是子类的方法

56.Super与this表示什么？

Super表示当前类的父类对象This表示当前类的对象

57.抽象的关键字是什么？

Abstract

58.抽象类必须要有抽象方法吗

不是必须。抽象类可以没有抽象方法。

59.如果一个类中有抽象方法，那么这个一定是抽象类？

包含抽象方法的类一定是抽象类

60.抽象类可以使用final修饰吗？

不可以。定义抽象类就是让其他继承的，而final修饰类表示该类不能被继承，与抽象类的理念违背了

61.普通类与抽象类有什么区别？

普通类不能包含抽象方法，抽象类可以包含抽象方法抽象类不能直接实例化，普通类可以直接实例化

62.什么是接口？

接口就是某个事物对外提供的一些功能的声明，是一种特殊的Java类

63.JAVA为什么需要接口？

接口弥补了Java单继承的缺点

64.接口有什么特点？

接口中声明全是publicstaticfinal修饰的常量接口中所有方法都是抽象方法接口是没有构造方法的接口也不能直接实例化接口可以多继承

65.接口与抽象类有什么区别？

抽象类有构造方法，接口没有构造方法抽象类只能单继承，接口可以多继承抽象类可以有普通方法，接口中的所有方法都是抽象方法接口的属性都是publicstaticfinal修饰的，而抽象的不是

66.Java中异常分为哪两种？

编译时异常运行时异常

67.说几个常见的编译时异常类？

NullPointerException：空指针异常ArrayIndexOutOfBoundsException：数组下标越界NumberFormatException：数字转换异常IllegalArgumentException：参数不匹配异常InstantiationException：对象初始化异常ArithmeticException：算术异常

68.异常的处理机制有几种？

异常捕捉：try...catch...finally，异常抛出：throws。

69.如何自定义一个异常

继承一个异常类，通常是RuntimeException或者Exception

70.在异常捕捉时，如果发生异常，那么try.catch.finally块外的return语句会执行吗？

会执行，如果有finally，在finally之后被执行，如果没有finally，在catch之后被执行

71.Try.catch.finally是必须要存在的吗？

Try块必须存在，catch和finally可以不存在，但不能同时不存在

72.Throw与throws区别

Throw写在代码块内，throw后面跟的是一个具体的异常实例Throw写在方法前面后面，throws后面跟的是异常类，异常类可以出现多个

73.Error与Exception区别？

Error和Exception都是Java错误处理机制的一部分，都继承了Throwable类。

Exception表示的异常，异常可以通过程序来捕捉，或者优化程序来避免。

Error表示的是系统错误，不能通过程序来进行错误处理。

74.使用Log4j对程序有影响吗？

有，log4j是用来日志记录的，记录一些关键敏感的信息，通常会将日志记录到本地文件或者数据库中。记录在本地文件中，会有频繁的io操作，会耗费一些系统资源。记录在数据库中，会频繁地操作数据库表，对系统性能也有一定的影响。但是为了程序安全以及数据的恢复或者bug的跟踪，这点资源消耗是可以承受的。

75.Log4j日志有几个级别？

由低到高：debug、info、warn、error

76.除了使用new创建对象之外，还可以用什么方法创建对象？

Java反射

77.Java反射创建对象效率高还是通过new创建对象的效率高？

通过new创建对象的效率比较高。通过反射时，先找查找类资源，使用类加载器创建，过程比较繁琐，所以效率较低

78.Java中集合框架的有几个？

Coillection、Map。

79.Collection接口下有那些集合框架？

List：线性表、Set：无序集合。

80.List接口有什么特点？

顺序存储、可以有重复值。

81.Set接口有什么特点

无须存储、不能有重复值。

82.ArrayList与LinkedList有什么区别？

ArrayList与LinkedList都实现了List接口。ArrayList是线性表，底层是使用数组实现的，它在尾端插入和访问数据时效率较高，Linked是双向链表，他在中间插入或者头部插入时效率较高，在访问数据时效率较低

83.Array与ArrayList有什么不一样？

Array与ArrayList都是用来存储数据的集合。ArrayList底层是使用数组实现的，但是arrayList对数组进行了封装和功能扩展，拥有许多原生数组没有的一些功能。我们可以理解成ArrayList是Array的一个升级版。

84.Map有什么特点

以键值对存储数据元素存储循序是无须的不允许出现重复键

85.JDBC操作的步骤

加载数据库驱动类打开数据库连接执行sql语句处理返回结果关闭资源

86.在使用jdbc的时候，如何防止出现sql注入的问题。

使用PreparedStatement类，而不是使用Statement类

87.怎么在JDBC内调用一个存储过程

使用CallableStatement

88.是否了解连接池，使用连接池有什么好处？

数据库连接是非常消耗资源的，影响到程序的性能指标。连接池是用来分配、管理、释放数据库连接的，可以使应用程序重复使用同一个数据库连接，而不是每次都创建一个新的数据库连接。通过释放空闲时间较长的数据库连接避免数据库因为创建太多的连接而造成的连接遗漏问题，提高了程序性能。

89.你所了解的数据源技术有那些？使用数据源有什么好处？

Dbcp,c3p0等,用的最多还是c3p0,因为c3p0比dbcp更加稳定,安全;通过配置文件的形式来维护数据库信息,而不是通过硬编码。当连接的数据库信息发生改变时,不需要再更改程序代码就实现了数据库信息的更新。

90.Java的io流分为哪两种?

按功能来分

输入流(input), 输出流(output)

按类型来分

字节流, 字符流

91.常用io类有那些?

FileFileInputSteam, FileOutputStreamBufferInputStream, BufferedOutputSreamPrintWriteFileReader, FileWriterBufferReader, BufferedWriterObjectInputStream, ObjectOutputSream

92.字节流与字符流的区别

以字节为单位输入输出数据, 字节流按照8位传输以字符为单位输入输出数据, 字符流按照16位传输

93.final、finalize()、finally

性质不同

final为关键字;

finalize()为方法;

finally为区块标志, 用于try语句中;作用

final为用于标识常量的关键字, final标识的关键字存储在常量池中(在这里final常量的具体用法将在下面进行介绍);

finalize()方法在Object中进行了定义, 用于在对象“消失”时, 由JVM进行调用用于对对象进行垃圾回收, 类似于C++中的析构函数;用户自定义时, 用于释放对象占用的资源(比如进行I/O操作);

finally{}用于标识代码块, 与try{}进行配合, 不论try中的代码执行完或没有执行完(这里指有异常), 该代码块之中的程序必定会进行;94.抽象类和接口的区别?

抽象类:

抽象方法, 只有行为的概念, 没有具体的行为实现。使用abstract关键字修饰, 没有方法体。子类必须重写这些抽象方法。

包含抽象方法的类, 一定是抽象类。

抽象类只能被继承, 一个类只能继承一个抽象类。接口:

全部的方法都是抽象方法, 属型都是常量

不能实例化, 可以定义变量。

接口变量可以引用具体实现类的实例

接口只能被实现，一个具体类实现接口，必须实现全部的抽象方法

接口之间可以多实现

一个具体类可以实现多个接口，实现多继承现象95.线程同步的方法

wait():让线程等待。将线程存储到一个线程池中。

notify():唤醒被等待的线程。通常都唤醒线程池中的第一个。让被唤醒的线程处于临时阻塞状态。

notifyAll():唤醒所有的等待线程。将线程池中的所有线程都唤醒。96.线程与进程的区别

进程是系统进行资源分配和调度的一个独立单位，线程是CPU调度和分派的基本单位

进程和线程的关系：

一个线程只能属于一个进程，而一个进程可以有多个线程，但至少有一个线程。

资源分配给进程，同一进程的所有线程共享该进程的所有资源。

线程在执行过程中，需要协作同步。不同进程的线程间要利用消息通信的办法实现同步。

线程是指进程内的一个执行单元，也是进程内的可调度实体。线程与进程的区别：

调度：线程作为调度和分配的基本单位，进程作为拥有资源的基本单位。

并发性：不仅进程之间可以并发执行，同一个进程的多个线程之间也可以并发执行。

拥有资源：进程是拥有资源的一个独立单位，线程不拥有系统资源，但可以访问隶属于进程的资源。

系统开销：在创建或撤销进程的时候，由于系统都要为之分配和回收资源，导致系统的明显大于创建或撤销线程时的开销。但进程有独立的地址空间，进程崩溃后，在保护模式下不会对其他的进程产生影响，而线程只是一个进程中的不同的执行路径。线程有自己的堆栈和局部变量，但线程之间没有单独的地址空间，一个线程死掉就等于整个进程死掉，所以多进程的程序要比多线程的程序健壮，但是在进程切换时，耗费的资源较大，效率要差些。97.&和&&的区别

&是位运算符。&&是布尔逻辑运算符，在进行逻辑判断时用&处理的前面为false后面的内容仍需处理，用&&处理的前面为false不再处理后面的内容。

98.重载与重写

Overload为重载，Override为重写方法的重写和重载是Java多态性的不同表现。重写是父类与子类之间多态性的一种表现，重载是一个类中多态性的一种表现。

如果在子类中定义某方法与其父类有相同的名称和参数，我们说该方法被重写(Override)。子类的对象使用这个方法时，将调用子类中的定义，对它而言，父类中的定义如同被“屏蔽”了。

如果在一个类中定义了多个同名的方法，它们或有不同的参数个数或有不同的参数类型，则称为方法的重载(Overload)。

重载的方法是可以改变返回值的类型。99.如果对象的引用被置为null，垃圾收集器是否会立即释放对象占用的内存？

不会，在下一个垃圾回收周期中，这个对象将是可被回收的。

100.串行(serial)收集器和吞吐量(throughput)收集器的区别是什么？

吞吐量收集器使用并行版本的新生代垃圾收集器，它用于中等规模和大规模数据的应用程序。而串行收集器对大多数的小应用(在现代处理器上需要大概100M左右的内存)就足够了。

101、面向对象和面向过程的区别

面向过程：面向过程性能比面向对象高。因为类调用时需要实例化，开销比较大，比较消耗资源，所以当性能是最重要的考量因素的时候，比如单片机、嵌入式开发、Linux/Unix等一般采用面向过程开发。但是，面向过程没有面向对象易维护、易复用、易扩展。

面向对象：面向对象易维护、易复用、易扩展。因为面向对象有封装、继承、多态性的特性，所以可以设计出低耦合的系统，使系统更加灵活、更加易于维护。但是，面向对象性能比面向过程低。

简单易学;

面向对象(封装，继承，多态);

平台无关性(java虚拟机实现平台无关性);

可靠性;

安全性;

支持多线程(C++语言没有内置的多线程机制，因此必须调用操作系统的多线程功能来进行多线程程序设计，而Java语言却提供了多线程支持);

支持网络编程并且很方便(java语言诞生本身就是为简化网络编程设计的，因此Java语言不仅支持网络编程而且很方便);

编译与解释并存;

103、JVM、JDK、JRE

JVM：Java虚拟机(JVM)是运行Java字节码的虚拟机。JVM有针对不同系统的特定实现(Windows，Linux，macOS)，目的是使用相同的字节码，它们都会给出相同的结果。

JDK：(JavaDevelopmentKit)Java开发工具包，它拥有JRE所拥有的一切，还有编译器(Javac)和工具(如Javadoc和jdb)。它能够创建和编译程序。

JRE：(JavaRuntimeEnvironment)Java运行时环境，它是运行已编译Java程序所需的所有内容的集合，包括Java虚拟机(JVM)，Java类库，Java命令和其他的一些基础构件。但是，它不能用于创建新程序。

104、Java和C++的区别？

都是面向对象的语言，都支持封装、继承和多态。

Java不提供指针来直接访问内存，程序内存更加安全。

Java的类是单继承的，C++支持多重继承;虽然Java的类不可以多继承，但是接口可以多继承。

Java有自动内存管理机制，不需要程序员手动释放无用内存。

形式上：字符常量是单引号引起的一个字符;字符串常量是双引号引起的若干个字符。

含义上：字符常量相当于一个整型值(ASCII值)，可以参加表达式运算;字符串常量代表一个地址值(该字符串在内存中存放位置)。

占内存大小：字符常量只占2个字节;字符串常量占若干个字节(至少一个字符结束标志)。(注意：char在Java中占两个字节)

8大基本数据类型及其字节数？

byte：1字节 short：2字节

int：4字节 long：8字节

float：4字节精确到7位有效数字 double：8字节

char：2字节 boolean：1位

引用类型：4字节，1个字节表示8位

106、构造器Constructor是否可被override？

父类的私有属性和构造方法并不能被继承，所以Constructor也就不能被override(重写),但是可以overload(重载),所以你可以看到一个类中有多个构造函数的情况。

107、重载和重写的区别？

重载：发生在同一个类中，方法名必须相同，实质表现就是多个具有不同的参数个数或者类型的同名函数(返回值类型可随意，不能以返回类型作为重载函数的区分标准)，返回值类型、访问修饰符可以不同，发生在编译时。

重写：发生在父子类中，方法名、参数列表必须相同，是父类与子类之间的多态性，实质是对父类的函数进行重新定义。返回值范围小于等于父类，抛出的异常范围小于等于父类，访问修饰符范围大于等于父类;如果父类方法访问修饰符为private则子类就不能重写该方法。问：Java构造方法能否被重写和重载？

重写是子类方法重写父类的方法，重写的方法名不变，而类的构造方法名必须与类名一致，假设父类的构造方法如果能够被子类重写则子类类名必须与父类类名一致才行，所以Java的构造方法是不能被重写的。而重载是针对同一个的，所以构造方法可以被重载。

108、Java面向对象编程三大特性？

封装继承多态

封装：

封装就是把抽象的数据和对数据进行的操作封装在一起，数据被保存在内部，程序的其他部分只有通过被授权的操作(成员方法)才能对数据进行操作。

Java提供了四种控制修饰符控制方法和变量访问的权限：

public：对外公开

protected：对子类和同一包中的类公开

没有修饰符号：向同一个包的类公开

private：只有类本身可以访问，不对外公开

继承：(extends) 继承是使用已存在的类的定义作为基础建立新类的技术。继承可以解决代码复用问题，当多个类存在相同的属性(变量)和方法时，可以从这些类中抽象出父类，在父类中定义这些相同的属性和方法，所有的子类不需要重新定义这些属性和方法，只需要通过extend语句来声明继承父类。

关于继承如下3点请记住：

子类拥有父类对象所有的属性和方法(包括私有属性和私有方法)，但是父类中的私有属性和方法子类是无法访问，只是拥有。

子类可以拥有自己属性和方法，即子类可以对父类进行扩展。

子类可以用自己的方式实现父类的方法。多态：

所谓多态，就是指一个引用(类型)在不同情况下的多种状态，你也可以这样理解：父类型的引用指向子类型的对象。

多态有两个好处：

1.应用程序不必为每一个派生类编写功能调用，只需要对抽象基类进行处理即可。大大提高程序的可复用性。//继承

2.派生类的功能可以被基类的方法或引用变量所调用，这叫向后兼容，可以提高可扩充性和可维护性。//多态的真正作用，

109、String、StringBuffer和StringBuilder的区别是什么？String为什么是不可变的？(重要)

一、区别

1、String是字符串常量，而StringBuffer和StringBuilder是字符串变量。由String创建的字符内容是不可改变的，而由StringBuffer和StringBuidler创建的字符内容是可以改变的。

2、StringBuffer是线程安全的，而StringBuilder是非线程安全的。StringBuilder是从JDK5开始，为StringBuffer类补充的一个单线程的等价类。我们在使用时应优先考虑使用StringBuilder，因为它支持StringBuffer的所有操作，但是因为它不执行同步，不会有线程安全带来额外的系统消耗，所以速度更快。

二、String为什么是不可变的？

String类中使用final关键字修饰字符数组来保存字符串，private final char value[]，所以String对象是不可变的。而StringBuilder与StringBuffer都继承自AbstractStringBuilder类，在AbstractStringBuilder中也是使用字符数组保存字符串char[]value但是没有用final关键字修饰，所以这两种对象都是可变的。

String是被声明为finalclass，除了hash这个属性其它属性都声明为final。因为它的不可变性，所以例如拼接字符串时候会产生很多无用的中间对象，如果频繁的进行这样的操作对性能有所影响。

StringBuffer就是为了解决大量拼接字符串时产生很多中间对象问题而提供的一个类，提供append和add方法，可以将字符串添加到已有序列的末尾或指定位置，它的本质是一个线程安全的可修改的字符序列，把所有修改数据的方法都加上了synchronized。但是保证了线程安全是需要性能的代价的。

在很多情况下我们的字符串拼接操作不需要线程安全，这时候StringBuilder登场了，StringBuilder是JDK1.5发布的，它和StringBuffer本质上没什么区别，就是去掉了保证线程安全的那部分，减少了开销。

StringBuffer和StringBuilder二者都继承了AbstractStringBuilder，底层都是利用可修改的char数组(JDK9以后是byte数组)。

再说说字符串常量池：

Java为了避免在一个系统中产生大量的String对象，引入了字符串常量池。

创建一个字符串时，首先会检查池中是否有值相同的字符串对象，如果有就直接返回引用，不会创建字符串对象;如果没有则新建字符串对象，返回对象引用，并且将新创建的对象放入池中。但是，通过new方法创建的String对象是不检查字符串常量池的，而是直接在堆中创建新对象，也不会把对象放入池中。上述原则只适用于直接给String对象引用赋值的情况。

```
Stringstr1=newString("a");//不检查字符串常量池的 Stringstr2="bb";//检查字符串常量池的
```

String还提供了intern()方法。调用该方法时，如果字符串常量池中包括了一个等于此String对象的字符串(由equals方法确定)，则返回池中的字符串的引用。否则，将此String对象添加到池中，并且返回此池中对象的引用。

在JDK6中，不推荐大量使用intern方法，因为这个版本字符串缓存在永久代中，这个空间是有限了，除了FullGC之外不会被清楚，所以大量的缓存在这容易OutOfMemoryError。

之后的版本把字符串放入了堆中，避免了永久代被挤满。

参考：<https://baijiahao.baidu.com/s?id=1629804867201303563&wfr=spider&for=pc>

对于三者使用的总结：

操作少量的数据:适用String

单线程操作字符串缓冲区下操作大量数据:适用StringBuilder

多线程操作字符串缓冲区下操作大量数据:适用StringBuffer110、自动装箱与拆箱

装箱：将基本类型用它们对应的引用类型包装起来;

拆箱：将包装类型转换为基本数据类型;//自动装箱Integertotal=99;//系统自动执行了Integertotal=Integer.valueOf(99);//自定拆箱inttotalprim=total;//系统自动执行了inttotalprim=total.intValue();

问题一：int和Integer的区别？

#是基本数据类型，Integer是int的包装类就是将int类型包装成Object对象;

2.Integer变量必须实例化后才能使用;int变量不需要;

3.Integer实际是对象的引用，指向此new的Integer对象;int是直接存储数据值;

4.Integer的默认值是null;int的默认值是0。

深入：

两个通过new生成的Integer变量永远是不相等的。因为new生成的是两个对象，其内存地址不同。

Integer与newInteger不会相等。因为非new生成的Integer变量指向的是Java常量池中的对象，而newInteger()生成的变量指向堆中新建的对象，两者在内存中的地址不同。

两个都是非new出来的Integer，如果数在-128到127之间，则是true,否则为false。

Java在编译Integeri=127的时候,被翻译成Integeri=Integer.valueOf(127);JavaAPI中对Integer类型的valueOf的定义如下，对于-128到127之间的数，会进行缓存，Integeri=127时，会将127这个Integer对象进行缓存，下次再写Integerj=127时，就会直接从缓存中取，就不会new了。Integer变量和int变量比较时，只要两个变量的值是向等的，则结果为true。(因为包装类Integer和基本数据类型int比较时，Java会自动拆箱为int，然后进行比较，实际上就变为两

个int变量的比较)Integer.valueOf函数源码：

```
public static Integer valueOf(int i) { return i >= 128 || i < -128 ? new Integer(i) : SMALL_VALUES[i + 128]; }
```

它会首先判断i的大小：如果i小于-128或者大于等于128，就创建一个Integer对象，否则执行SMALL_VALUES[i+128]。

SMALL_VALUES[i+128]是什么东西：

```
private static final Integer[] SMALL_VALUES = new Integer[256];
```

SMALL_VALUES本来已经被创建好，也就是说在i>=128 || i<-128是会创建不同的对象，在i<128&& i>=-128会根据i的值返回已经创建好的指定的对象。

问题二：为什么有了int还要有设计Integer？

对象封装有很多好处，可以把属性也就是数据跟处理这些数据的方法结合在一起，比如Integer就有parseInt()等方法来专门处理int型相关的数据。

另一个非常重要的原因就是Java中绝大部分方法或类都是用来处理类类型对象的，如ArrayList集合类就只能以类作为他的存储对象，而这时如果想把一个int型的数据存入list是不可能的，必须把它包装成类，也就是Integer才能被List所接受。所以Integer的存在是很必要的。

111、为什么不能从静态的方法里调用非静态的方法或变量？

非静态的方法可以调用静态的方法，但是静态的方法不可以调用非静态的方法。

类的静态成员(变量和方法)属于类本身，在类加载的时候就会分配内存，可以通过类名直接去访问;非静态成员(变量和方法)属于类的对象，所以只有在类的对象产生(创建类的实例)时才会分配内存，然后通过类的对象(实例)去访问。

在一个类的静态成员中去访问其非静态成员之所以会出错是因为在类的非静态成员不存在的时候类的静态成员就已经存在了，访问一个内存中不存在的东西当然会出错。

参考<https://blog.csdn.net/l18649805795/article/details/48939487>

112、静态方法和实例方法有何不同？

在外部调用静态方法时，可以使用“类名.方法名”的方式，也可以使用“对象名.方法名”的方式。而实例方法只有后面这种方式。也就是说，调用静态方法可以无需创建对象。

静态方法在访问本类的成员时，只允许访问静态成员(即静态成员变量和静态方法)，而不允许访问实例成员变量和实例方法;实例方法则无此限制。问题：static修饰变量、代码块时什么时候执行？执行几次？

在类加载的init阶段，类的类构造器中会收集所有的static块和字段并执行;static块只执行一次。

【注】：

static语句块,不是在实例化的时候被执行的;

在调用类中任何一个方法时，jvm进行类加载，static语句块是在类加载器加载该类的最后阶段进行初始化的。并且只会被初始化一次。(注：若一次性调用多个方法，则只会执行一次static代码块。)113、在Java中定义一个不做事且没有参数的构造方法的作用？

Java程序在执行子类的构造方法之前，如果没有用super()来调用父类特定的构造方法，则会调用父类中“没有参数的

构造方法”。因此，如果父类中只定义了有参数的构造方法，而在子类的构造方法中又没有用super()来调用父类中特定的构造方法，则编译时将发生错误，因为Java程序在父类中找不到没有参数的构造方法可供执行。解决办法是在父类里加上一个不做事且没有参数的构造方法。

问题：子类继承父类时，父类的构造方法什么时候调用？

实例化一个子类对象时会先执行其父类的构造函数，然后再执行子类的构造函数。

super()必须先被调用;如果子类构造方法里没有写super()，编译器会自动调用super()方法，即调用父类的默认无参构造方法。所以不可以父类中只定义了有参数的构造方法(在Java中，如果一个类没有定义构造方法，编译器会默认插入一个无参数的构造方法;但是如果一个构造方法在父类中已定义，在这种情况下，编译器是不会自动插入一个默认无参构造方法)

114、构造方法有哪些特性？

名字与类名相同。

没有返回值，但不能用void声明构造函数。

生成类的对象时自动执行，无需调用。

115、接口(interface)和抽象类(abstractclass)的区别是什么？

接口中的所有方法必须都是抽象的，不能有非抽象的普通方法(所有方法在接口中不能有实现);而抽象类中可以包含非抽象的普通方法。

接口中不能有构造方法，抽象类可以有构造方法。

接口中除了static、final变量，不能有其他变量，而抽象类中则不一定。

一个类可以实现多个接口，但只能继承一个抽象类。接口自己本身可以通过extends关键字扩展多个接口。

接口中的抽象方法只能是public类型的，并且默认修饰符是public;抽象方法可以有public、protected和default这些修饰符(抽象方法就是为了被重写所以不能使用private关键字修饰！)。

接口中不能包含静态方法;抽象类中可以包含静态方法。

抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是publicstaticfinal类型，并且默认即为publicstaticfinal类型。接口和抽象类的相同点：

都可以被继承

都不能被实例化

都可以包含方法声明

派生类必须实现未实现的方法

116、成员变量与局部变量的区别有那些？

从语法形式上看:成员变量是属于类的，而局部变量是在方法中定义的变量或是方法的参数;成员变量可以被public,private,static等修饰符所修饰，而局部变量不能被访问控制修饰符及static所修饰;但是，成员变量和局部变量都能被final所修饰。

从变量在内存中的存储方式来看:如果成员变量是使用static修饰的，那么这个成员变量是属于类的，如果没有使用st

atic修饰，这个成员变量是属于实例的。而对象存在于堆内存，局部变量则存在于栈内存。

从变量在内存中的生存时间上看:成员变量是对象的一部分，它随着对象的创建而存在，而局部变量随着方法的调用而自动消失。

成员变量如果没有被赋初值:则会自动以类型的默认值而赋值(一种情况例外:被final修饰的成员变量也必须显式地赋值)，而局部变量则不会自动赋值。117、==与equals的区别？(重要)

1)\==比较的是值是否相等

****如果作用于基本数据类型的变量，则直接比较其存储的“值”是否相等;****

如果作用于引用类型的变量，则比较的是所指向的对象的地址。

2)equals方法不能作用于基本数据类型的变量，只能用于类变量。(对于基本数据类型要用其包装类)

如果没有对equals方法进行重写，则比较的是引用类型的变量所指向的对象的地址;

诸如String、Date等类对equals方法进行了重写的话，比较的是所指向的对象的内存地址。

【注】Object类中的equals方法和“==”是一样的，没有区别，而String类，Integer类等等一些类，是重写了equals方法，才使得equals和“==”不同，所以，当自己创建类时，自动继承了Object的equals方法，要想实现不同的等于比较，必须重写equals方法。

```
Stringa=newString("ab");//a为一个引用Stringb=newString("ab");//b为另一个引用,对象的内容一样Stringaa="ab";//
放在常量池中Stringbb="ab";//从常量池中查找System.out.println(aa==bb);//trueSystem.out.println(a==b);//false
,非同对象System.out.println((a.equals(b)));//trueSystem.out.println((42==42.0))//true
```

String中的equals方法是被重写过的，因为object的equals方法是比较的对象的内存地址，而String的equals方法比较的是对象的值。

当创建String类型的对象时，虚拟机会在常量池中查找有没有已经存在的值和要创建的值相同的对象，如果有就把它赋给当前引用。如果没有就在常量池中重新创建一个String对象。118、为什么重写equals时必须重写hashCode方法？(重要)

当对象的equals()方法被重写时，通常有必要重写hashCode()方法，以维护hashCode方法的常规协定，该协定声明相等对象必须具有相等的哈希码。

(1)两个对象相等，hashcode一定相等(2)两个对象不等，hashcode不一定不等(3)hashcode相等，两个对象不一定相等(4)hashcode不等，两个对象一定不等hashcode是用于散列数据的快速存取，如利用HashSet/HashMap/Hashtable类来存储数据时，都是根据存储对象的hashcode值来进行判断是否相同的。这样**如果我们对一个对象重写了equals，意思是只要对象的成员变量值都相等那么equals就等于true，但不重写hashcode，那么我们再new一个新的对象，当原对象.equals(新对象)等于true时，两者的hashcode却是不一样的，由此将产生了理解的不一致**。

[#/wang-meng/p/7501378.html](#)

HashMap中的get与put：

(1)put：

1.首先根据put元素的key获取hashcode，然后根据hashcode算出数组的下标位置，如果下标位置没有元素，直接放入元素即可。2.如果该下标位置有元素，则需要已有元素和put元素的key对象比较equals方法，如果equals不一样，则说明可以放入进map中，会在该数组位置创建一个链表，后put进入的元素到放链表头，原来的元素向后移动。

(2)get :

根据元素的key获取hashCode, 然后根据hashCode获取数组下标位置, 如果只有一个元素则直接取出。如果该位置是一个链表, 则需要调用equals方法遍历链表中的所有元素与当前的元素比较, 得到真正想要的对象。(当两个对象的hashCode不同的话, 肯定他们不能equals。)

阿里2022年的Java面试题总结

19、Java中final、finally、finalize的区别？

(1)final是一个修饰符，

如果一个类被声明为final则其不能再派生出新的子类，所以一个类不能既被声明为abstract又被声明为final[所谓不可同时出现]的;

将变量或方法声明为final可以保证它们在使用中不被改变(对于对象变量来说其引用不可变，即不能再指向其他的对象，但是对象的值可变)。注：

被声明为final的变量必须在声明时给定初值，而在以后的引用中只能读取不可修改，被声明为final的方法也同样只能使用不能重载。

使用final关键字如果编译器能够在编译阶段确定某变量的值则编译器就会把该变量当做编译期常量来使用，如果需要在运行时确定(譬如方法调用)则编译器就不会优化相关代码;将类、方法、变量声明为final能够提高性能，这样JVM就有机会进行估计并进行优化;接口中的变量都是publicstaticfinal的。final关键字主要用在三个地方：变量、方法、类。

对于一个final变量，如果是基本数据类型的变量，则其数值一旦在初始化之后便不能更改;如果是引用类型的变量，则在对其初始化之后便不能再让其指向另一个对象。

当用final修饰一个类时，表明这个类不能被继承。final类中的所有成员方法都会被隐式地指定为final方法。

使用final方法的原因有两个。第一个原因是把方法锁定，以防任何继承类修改它的含义;第二个原因是效率。在早期的Java实现版本中，会将final方法转为内嵌调用。但是如果方法过于庞大，可能看不到内嵌调用带来的任何性能提升(现在的Java版本已经不需要使用final方法进行这些优化了)。类中所有的private方法都隐式地指定为final。(2)finally：用来在异常处理中，如果抛出一个异常，则相匹配的catch子句就会执行，然后控制就会进入finally块

finally是对Java异常处理模型的最佳补充。finally结构使代码总会执行，而不管无异常发生。使用finally可以维护对象的内部状态，并可以清理非内存资源。特别是在关闭数据库连接这方面，如果程序员把数据库连接的close()方法放到finally中，就会大大降低程序出错的几率。

异常处理：

try块：用于捕获异常。其后可接零个或多个catch块，如果没有catch块，则必须跟一个finally块。

catch块：用于处理try捕获到的异常。

finally块：无论是否捕获或处理异常，finally块里的语句都会被执行。当在try块或catch块中遇到return语句时，finally语句块将在方法返回之前被执行。在以下4种特殊情况下，finally块不会被执行：

在finally语句块第一行发生了异常。因为在其他行，finally块还是会得到执行

在前面的代码中用了System.exit(int)已退出程序。exit是带参函数;若该语句在异常语句之后，finally会执行

程序所在的线程死亡。

关闭CPU。(3)finalize(): 是一个方法,它是在对象被垃圾回收之前由Java虚拟机来调用的。

finalize()方法是GC运行机制的一部分,finalize()方法是在GC清理它所从属的对象时被调用的,如果执行它的过程中抛出了无法捕获的异常,GC将终止对改对象的清理,并且该异常会被忽略;直到下一次GC开始清理这个对象时,它的finalize()会被再次调用。

120、this、super

(1)this: 代表对象本身,可以理解为:指向对象本身的一个指针。

this的用法在Java中大体可以分为3种:

1)普通的直接引用

这种就不用讲了,this相当于是指向当前对象本身:

2)形参与成员名字重名,用this来区分:

3)引用构造函数

this(参数): 调用本类中另一种形式的构造方法(应该为构造方法中的第一条语句)

(2)super: 代指父类,可以用于调用父类的普通方法和构造方法。

调用父类构造方法: super()(无参构造方法)或super(参数)(有参构造方法)

调用父类普通方法: super.方法名(参数)

121、Java序列化中如果有些字段不想进行序列化,怎么办?

对于不想进行序列化的变量,使用transient关键字修饰。

transient关键字的作用是:阻止实例中那些用此关键字修饰的的变量序列化;当对象被反序列化时,被transient修饰的变量值不会被持久化和恢复。transient只能修饰变量,不能修饰类和方法。

122、获取用键盘输入常用的的两种方法

方法1: 通过Scanner

```
Scanner sc = new Scanner(System.in); String s = sc.nextLine(); input.close();
```

方法2: 通过BufferedReader

```
BufferedReader input = new BufferedReader(new InputStreamReader(System.in)); String s = input.readLine();
```

版权声明:

本站遵循 [署名-非商业性使用-相同方式共享 2.5](#) 共享协议。

转载请注明转自[悠久防伪防窜货追溯系统](#)并标明URL。

本文链接: <https://www.fwxt.cn/1867.html>