# **Java基础面试题**

### **一: Java基础**

1: 简单说说Java中对象如何拷贝?

一、浅拷贝clone（）

如果对象中的所有数据域都是数值或者基本类型，使用clone（）即可满足需求，如：

Person p = new Person();

Person p1 = p.clone();

这样p和p1分别指向不同的对象。

二、深度拷贝

如果在对象中包含子对象的引用，拷贝的结果是使得两个域引用同一个对象，默认的拷贝是浅拷贝，没有拷贝包含在对象中的内部对象。

如果子对象是不可变的，如String，这没有什么问题；如果对象是可变的，必须重新定义clone方法；

三、序列化可克隆（深拷贝）

四、BeanUtils.copyProperties()

2: 伪代码快速实现一下深拷贝

3: Java是什么类型的语言?

一、你可以说它是编译型的：因为所有的Java代码都是要编译的，.java不经过编译就什么用都没有。

二、你可以说它是解释型的：因为java代码编译后不能直接运行，它是解释运行在JVM上的，所以它是解释运行的，那也就算是解释的了。

三、但是，现在的JVM为了效率，都有一些JIT优化。它又会把.class的二进制代码编译为本地的代码直接运行，所以，又是编译的。

定义：

（1）编译型语言：把做好的源程序全部编译成二进制代码的可运行程序。然后，可直接运行这个程序。

（2）解释型语言：把做好的源程序翻译一句，然后执行一句，直至结束！

区别：

（1）编译型语言，执行速度快、效率高；依靠编译器、跨平台性差些。

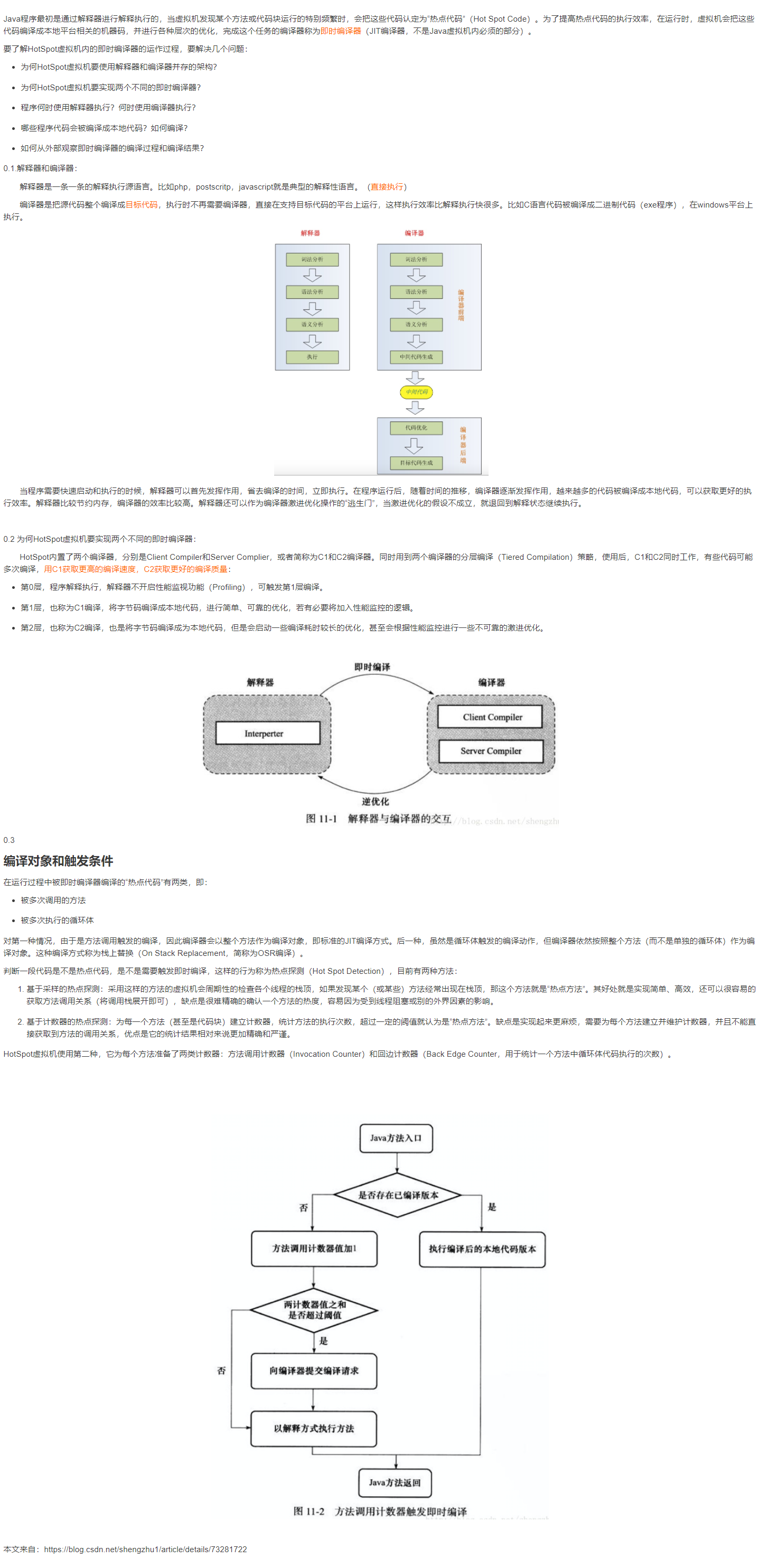
（2）解释型语言，执行速度慢、效率低；依靠解释器、跨平台性好。

java是解释型的语言，因为虽然java也需要编译，编译成.class文件，但是并不是机器可以识别的语言，而是字节码，最终还是需要 jvm的解释，才能在各个平台执行，这同时也是java跨平台的原因。所以可是说java即是编译型的，也是解释型，但是假如非要归类的话，从概念上的定义，恐怕java应该归到解释型的语言中。

附：

编译型的语言包括：C、C++、Delphi、Pascal、Fortran

解释型的语言包括：Java、Basic、javascript、python



GraalVM 即时编译 graal aot jit c1 c2

4: 什么是Object,有哪些常用的方法,怎么创建对象?

1、使用new关键字；

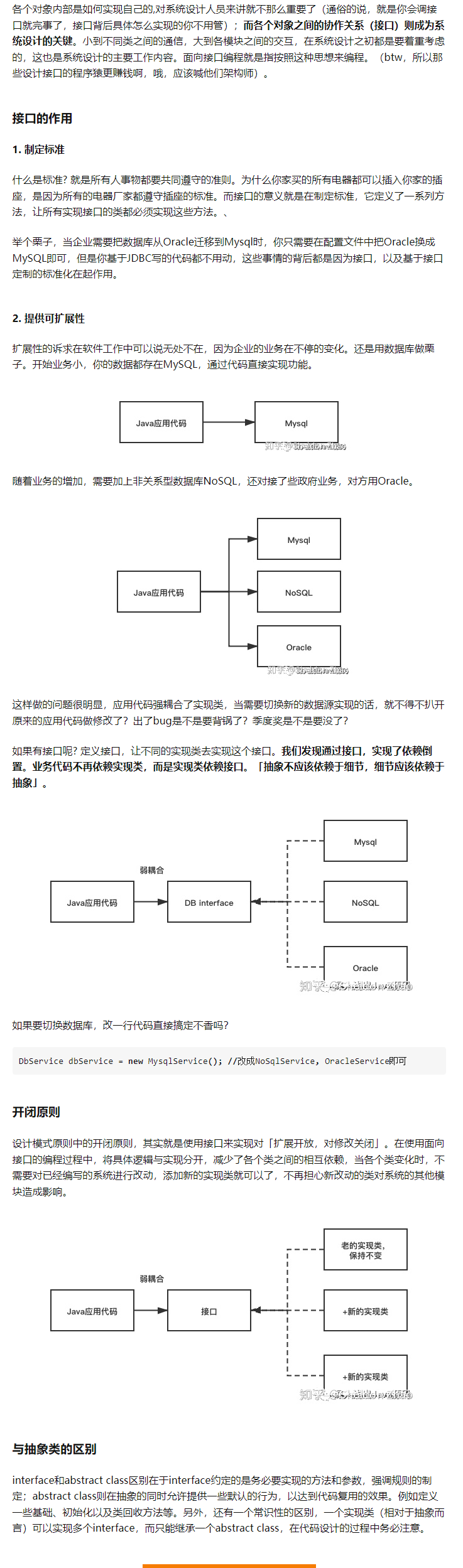
2、使用Class类的newInstance方法，可调用无参的构造函数创建对象；

3、使用Constructor类的newInstance方法；

4、使用clone方法；

5、使用反序列化。

5: 多态,面向接口编程?聊聊你的认知



6: 什么是内部类,说说你对他的理解以及实战场景

使用内部类最吸引人的原因是：每个内部类都能独立地继承一个（接口的）实现，所以无论外围类是否已经继承了某个（接口的）实现，对于内部类都没有影响。

在我们程序设计中有时候会存在一些使用接口很难解决的问题，这个时候我们可以利用内部类提供的、可以继承多个具体的或者抽象的类的能力来解决这些程序设计问题。可以这样说，接口只是解决了部分问题，而内部类使得多重继承的解决方案变得更加完整。

1、内部类可以用多个实例，每个实例都有自己的状态信息，并且与其他外围对象的信息相互独立。

2、在单个外围类中，可以让多个内部类以不同的方式实现同一个接口，或者继承同一个类。

3、创建内部类对象的时刻并不依赖于外围类对象的创建。

4、内部类并没有令人迷惑的“is-a”关系，他就是一个独立的实体。

5、内部类提供了更好的封装，除了该外围类，其他类都不能访问。

成员内部类

第一：成员内部类中不能存在任何static的变量和方法；

第二：成员内部类是依附于外围类的，所以只有先创建了外围类才能够创建内部类。

局部内部类

匿名内部类

1、 匿名内部类是没有访问修饰符的。

2、 new 匿名内部类，这个类首先是要存在的。如果我们将那个InnerClass接口注释掉，就会出现编译出错。

3、 注意getInnerClass()方法的形参，第一个形参是用final修饰的，而第二个却没有。同时我们也发现第二个形参在匿名内部类中没有使用过，所以当所在方法的形参需要被匿名内部类使用，那么这个形参就必须为final。

4、 匿名内部类是没有构造方法的。因为它连名字都没有何来构造方法。

静态内部类

1、 它的创建是不需要依赖于外围类的。

2、 它不能使用任何外围类的非static成员变量和方法。

https://www.cnblogs.com/chanchan/p/8402411.html 局部内部类只能使用final 局部变量

7: 说说 static 和 final 在Java中的意义

使用final的意义：

为方法上锁，防止任何继承类改变它的本来含义和实现。设计程序时，若希望一个方法的行为在继承期间保持不变，而且不可被覆盖或改写，就可以采取这种做法。

提高程序执行的效率，将一个方法设成final后，编译器就可以把对那个方法的所有调用都置入嵌入调用里（内嵌机制）。

static总结

static修饰成员函数:该成员函数不能使用this对象

static不能修饰构造函数

static不能修饰函数参数

static不能修饰局部成员变量

static修饰成员字段

当类被虚拟机加载时，首先按照字段声明的先后顺序对static成员字段进行初始化

static修饰语句块

当类被虚拟机加载时，按照声明顺序先后初始化static成员字段和static语句块

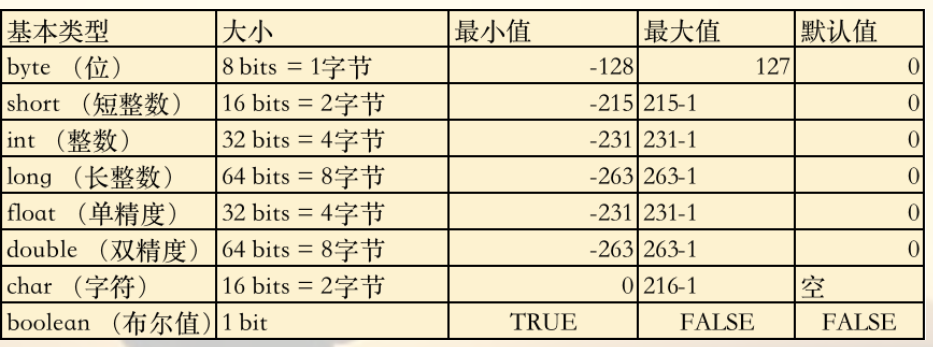
static所修饰的方法和字段是只属于类，所有对象共享。

在static所修饰的函数和语句块中不能使用非static成员字段。

在Java不能直接定义全局变量，是通过static来实现的

在Java中没有const，不能直接定义常量，通过static final来实现

8: Java中的基本数据类型占多少字节,不同的操作系统一样吗?



非基本类型BigDecimal引用的对象占用的字节数是可变的，具体大小依赖于你输入的数据量，要求的精度和选择的舍入方法。如果你问的是类型为BigDecimal的引用变量占用的字节数，那只有一个指针变量的大小，目前为4。

9: String,StringBuffer,StringBuild;以及String常用的方法

10: int和Integer有什么关联?为什么需要Integer?装箱拆箱

11: 什么是序列化,反序列化.说说运用场景

https://www.cnblogs.com/binarylei/p/10987540.html

12: 数组有length（）方法吗?

13: 构造器是否可以被重写

14: char是否可以存储一个汉字

15: 集合原理系列

ArrayList原理

LinkedList原理

为什么舍弃Vector

HashSet/TreeSet

HashMap/TreeMap 原理

为什么舍弃HashTable

Queue

　　add 增加一个元索 如果队列已满，则抛出一个IIIegaISlabEepeplian异常

　　remove 移除并返回队列头部的元素 如果队列为空，则抛出一个NoSuchElementException异常

　　element 返回队列头部的元素 如果队列为空，则抛出一个NoSuchElementException异常

　　offer 添加一个元素并返回true 如果队列已满，则返回false

　　poll 移除并返问队列头部的元素 如果队列为空，则返回null

　　peek 返回队列头部的元素 如果队列为空，则返回null

　　put 添加一个元素 如果队列满，则阻塞

　　take 移除并返回队列头部的元素 如果队列为空，则阻塞

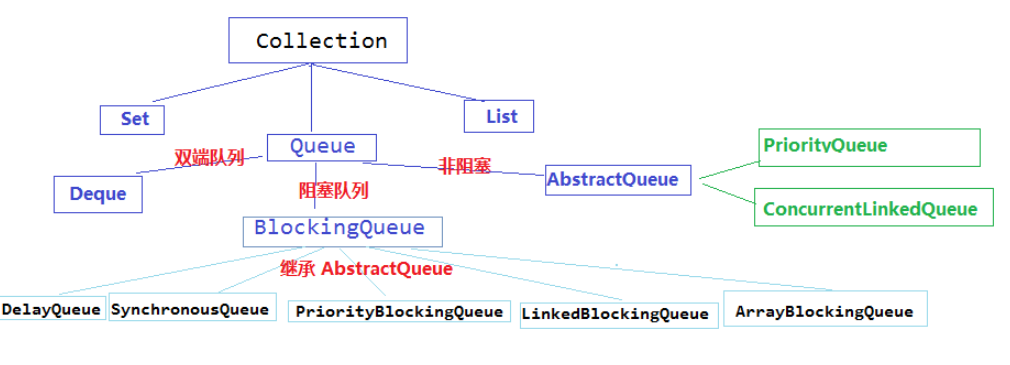
　　\* ArrayBlockingQueue ：一个由数组支持的有界队列。

　　\* LinkedBlockingQueue ：一个由链接节点支持的可选有界队列。

　　\* PriorityBlockingQueue ：一个由优先级堆支持的无界优先级队列。

　　\* DelayQueue ：一个由优先级堆支持的、基于时间的调度队列。

　　\* SynchronousQueue ：一个利用 BlockingQueue 接口的简单聚集（rendezvous）机制。



16: Enumeration接口和Iterator接口的区别有哪些?

Enumeration速度是Iterator的2倍，同时占用更少的内存。但是，Iterator远远比Enumeration安全，因为其他线程不能够修改正在被iterator遍历的集合里面的对象。同时，Iterator允许调用者删除底层集合里面的元素，这对Enumeration来说是不可能的。

17: 异常

受检查/不受检查

18: 什么是事务?为什么需要事务?如何实现事务?

通常的观念认为，事务仅与数据库相关。

事务必须服从ISO/IEC所制定的ACID原则。ACID是原子性（atomicity）、一致性（consistency）、隔离性 （isolation）和持久性（durability）的缩写。

事务的原子性：表示事务执行过程中的任何失败都将导致事务所做的任何修改失效。

事务的一致性：表示当事务执行失败时，所有被该事务影响的数据都应该恢复到事务执行前的状态。

事务的隔离性：表示在事务执行过程中对数据的修改，在事务提交之前对其他事务不可见。

事务的持久性：表示已提交的数据在事务执行失败时，数据的状态都应该正确。

通俗的理解，事务是一组原子操作单元，从数据库角度说，就是一组SQL指令，要么全部执行成功，若因为某个原因其中一条指令执行有错误，则撤销先前执行过的所有指令。更简答的说就是：要么全部执行成功，要么撤销不执行。

Java事务的类型有三种：JDBC事务、JTA（Java Transaction API）事务、容器事务。

1.JDBC事务

JDBC 事务是用 Connection 对象控制的。JDBC Connection 接口（ java.sql.Connection ）提供了两种事务模式：自动提交和手工提交。

复制代码

1 java.sql.Connection 提供了以下控制事务的方法：

2

3 public void setAutoCommit(boolean)

4 public boolean getAutoCommit()

5 public void commit()

6 public void rollback()

复制代码

使用 JDBC 事务界定时，您可以将多个 SQL 语句结合到一个事务中。

JDBC 事务的一个缺点是事务的范围局限于一个数据库连接。一个 JDBC 事务不能跨越多个数据库。

2.JTA（Java Transaction API）事务

JTA是一种高层的，与实现无关的，与协议无关的API，应用程序和应用服务器可以使用JTA来访问事务。

JTA允许应用程序执行分布式事务处理——在两个或多个网络计算机资源上访问并且更新数据，这些数据可以分布在多个数据库上。JDBC驱动程序的JTA支持极大地增强了数据访问能力。

如果计划用 JTA 界定事务，那么就需要有一个实现 javax.sql.XADataSource 、 javax.sql.XAConnection 和 javax.sql.XAResource 接口的 JDBC 驱动程序。一个实现了这些接口的驱动程序将可以参与 JTA 事务。一个 XADataSource 对象就是一个 XAConnection 对象的工厂。 XAConnection s 是参与 JTA 事务的 JDBC 连接。

您将需要用应用服务器的管理工具设置 XADataSource .从应用服务器和 JDBC 驱动程序的文档中可以了解到相关的指导。

J2EE应用程序用 JNDI 查询数据源。一旦应用程序找到了数据源对象，它就调用 javax.sql.DataSource.getConnection（） 以获得到数据库的连接。

XA 连接与非 XA 连接不同。一定要记住 XA 连接参与了 JTA 事务。这意味着 XA 连接不支持 JDBC 的自动提交功能。同时，应用程序一定不要对 XA 连接调用 java.sql.Connection.commit（） 或者 java.sql.Connection.rollback（） .

相反，应用程序应该使用 UserTransaction.begin（）、 UserTransaction.commit（） 和 serTransaction.rollback（） .

3.容器事务

容器事务主要是J2EE应用服务器提供的，容器事务大多是基于JTA完成，这是一个基于JNDI的，相当复杂的API实现。相对编码实现JTA事务管理， 我们可以通过EJB容器提供的容器事务管理机制（CMT）完成同一个功能，这项功能由J2EE应用服务器提供。这使得我们可以简单的指定将哪个方法加入事 务，一旦指定，容器将负责事务管理任务。这是我们土建的解决方式，因为通过这种方式我们可以将事务代码排除在逻辑编码之外，同时将所有困难交给J2EE容 器去解决。使用EJB CMT的另外一个好处就是程序员无需关心JTA API的编码，不过，理论上我们必须使用EJB.

三种Java事务差异?#

1、JDBC事务控制的局限性在一个数据库连接内，但是其使用简单。

2、JTA事务的功能强大，事务可以跨越多个数据库或多个DAO，使用也比较复杂。

3、容器事务，主要指的是J2EE应用服务器提供的事务管理，局限于EJB应用使用。

事务并发处理可能引起的问题

脏读(dirty read)：一个事务读取了另一个事务尚未提交的数据，

不可重复读(non-repeatable read) ：一个事务的操作导致另一个事务前后两次读取到不同的数据

幻读(phantom read) ：一个事务的操作导致另一个事务前后两次查询的结果数据量不同。

举例：

事务A、B并发执行时，当A事务update后，B事务select读取到A尚未提交的数据，此时A事务rollback，则B读到的数据是无效的”脏”数据。

当B事务select读取数据后，A事务update操作更改B事务select到的数据，此时B事务再次读去该数据，发现前后两次的数据不一样。

当B事务select读取数据后，A事务insert或delete了一条满足A事务的select条件的记录，此时B事务再次select，发现查询到前次不存在的记录(“幻影”)，或者前次的某个记录不见了。

19: java修饰符以及个字的作用范围

啊 访问权限 类 包 子类 其他包

　　　　 public ∨ ∨ ∨ ∨ （对任何人都是可用的）

　　　　 protect ∨ ∨ ∨ ×　　　 （继承的类可以访问以及和private一样的权限）

　　　　 default ∨ ∨ × ×　　　 （包访问权限，即在整个包内均可被访问）

　　　　 private ∨ × × ×　　　 （除类型创建者和类型的内部方法之外的任何人都不能访问的元素）

### **二: Java高级**

**linux部分**

1：什么是linux？如何安装使用linux

2：说说你常用的linux查看文件和文件夹命令

3：常用查看文件的命令有哪些?

4：如何递归删除文件夹和所有文件？

5：如何拷贝，移动一个文件？

6：如何查找一个文件？

7：你还是使用过哪些常用的linux命令？

8：grep是什么命令？man是什么命令？

9：chmod是什么命令？

10：VI、VIM是什么？

1: chmod命令如何使用？ r,w,x/u,g,o/+,-/1,2,4 各自代表什么含义？

2：什么是VI、VIM？说说你常用的VI命令和按键？

3：什么是编辑视图、预览视图？

4：netstat命令的作用？你如何使用？

5：kill命令如何使用？

6：linux下如何解压压缩文件？

7：ps和grep各自作用是什么？如何使用？

8：如何部署一个tomcat ， linux服务器？

9：如何部署自己的项目至linux服务器？

### **三: JavaWeb**

1:什么是servlet,jsp?

2: web.xml在web项目中有什么作用?

3: 说说servlet的原理

4: jsp里面的内置对象

5:什么是cookie?有什么作用?

6: 什么是session? 有什么作用?

7: 禁用客户端cookie,如何实现session?

8: dopost doget的区别?还有哪些请求类型?

9: Forward和Redirect的区别?

10: HTTP请求/响应的结构是怎么样的

HTTP响应由三个部分组成：

状态码(Status Code)：描述了响应的状态。可以用来检查是否成功的完成了请求。请求失败的情况下，状态码可用来找出失败的原因。如果Servlet没有返回状态码，默认会返回成功的状态码HttpServletResponse.SC\_OK。

HTTP头部(HTTP Header)：它们包含了更多关于响应的信息。比如：头部可以指定认为响应过期的过期日期，或者是指定用来给用户安全的传输实体内容的编码格式。如何在Serlet中检索HTTP的头部看这里。

主体(Body)：它包含了响应的内容。它可以包含HTML代码，图片，等等。主体是由传输在HTTP消息中紧跟在头部后面的数据字节组成的。

### **四: 前端技术**

选择器常用语法等

### **五: 框架技术**

1: 什么是框架?如何使用框架?

2: SSM,SSH,JPA,mybatisplus,及其实现原理?

3: 简单说一下Spring的IOC,以及DI

4: 简单说一下SpringAOP

5: 什么是ORM

6: 简要描述一下SSM如何进行整合

7: @RequestMapping及其系列注解

8: springmvc如何处理ajax请求?

9: Mybatis优缺点分析

10: #{}和${}的区别是什么?

11: 实体类属性和数据表字段不一样如何处理?

12: 为什么说Mybatis是半自动ORM映射工具？它与全自动的区别在哪里？

13: 什么是一对一,一对多,多对多? mybatis中如何实现?

### **六: 微服务部分**

nginx

1: 什么是nginx?

2: 什么是负载均衡?

3: 什么是动静分离?

4: 什么是反向代理?

5: nginx如何配置动静分离,反向代理?

6: 如何搭建伪集群?

7: 如何搭建集群?

8: 简要说一下nginx每个配置项的作用.

1: nginx\_http节点有什么作用？

2: nginx\_http\_upstream 节点有什么作用？

3: nginx\_http\_server节点有什么作用？

4: nginx\_http\_server\_location 节点有什么作用？

5：如何配置nginx伪集群，简要写一下nginx.conf的配置

6：如何配置nginx集群，简要写一下nginx.conf的配置

7：如何配置轮询策略，ip\_hash策略，权重策略，备份？

zookeeper+RPC+Dubbo

1：什么是dubbo？

2：简述一下dubbo的原理

3：什么是zookeeper

4：什么是zookeeper的leader和flower和observer？

5：什么是过半机制，为什么zookeeper集群部署奇数台服务器？

6：zookeeper如何选举？

7：zookeeper存储数据的特点？

8：如何实现服务的注册与消费？简要写一下伪代码

9：dubbo框架需要web容器吗

10：dubbo有哪些负载均衡策略