2015 腾讯高级软件工程师、项目经理面试题

1、 类、对象的概念？

2、 简述抽象？

3、 接口与抽象类？

4、内部类（Inner Class）？

5、访问修饰符限制？

6、Static 关键字的使用？

7、final 关键字？

8、abstract 关键字？

9、native 关键字？

10、synchronized 关键字？

11、运行时异常与一般异常有何异同？

12、编程题： 写一个 Singleton 出来。

13、分类列举服务器和组件技术？

14、Http 与 Https ？

15、OSI（Open System Interconnection）网络抽象模型？

16、J2EE 的容器与服务器？

17、继承限制？

18、逻辑操作：c=(a>b)?a:b;等同于下式？

19、列举常见集合框架类型？

20、面向对象的特征？

21、Java 命名规范？

22、Java 语言共包含 47 个关键字？

23、设计模式？

24、简述 MVC 的应用(如 STRUTS1.x、STRUTS2.x 架构等)？

25、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI,WSDL 解释。

26、存储过程和函数的区别？

27、 游标的作用？如何知道游标已经到了最后？

28、你认为一个项目如何进行才正确？

29、你经常看或仔细研读过的书有哪些？

30、你认为你应聘我们公司的项目经理，你自身的优势在哪？

31、你认为项目中最重要的是哪些过程

32、如果给你一个 4－6 个人的 Team，那么你怎么分配他们、管理他们？

33、简述常用的软件开发文档

34、简述类的关系

注：下面是面试题答案，答案根据自己所学知识整理，仅供参考，不保证正确性。

1、 类、对象的概念：

1） 类：具有共同属性和行为的对象的抽象。类是创建对象的模板。

2） 对象：现实世界中的实体。在计算机中，是指可标识的存储区域。

3） 类是对象的抽象、对象是类的实例。

2、 抽象：是从特定的实例中抽取共同性质形成一般化概念的过程。

3、 接口与抽象类：

1）接口和抽象类都用于抽象，接口是抽象类的抽象。

2）接口中只有方法声明，没有实现（无方法体）；在接口中声明的方法具有 public 和 abstract 属性，一个类可以实现多个接口（即多继承），接口以‘，’分隔；接口中的方法必须全部实现。

3）抽象类可以有部分方法实现，抽象类必须通过继承才能使用。

4、 内部类（Inner Class）：

1）内部类是嵌套在另一个类中的类。

2）内部类用于名称隐藏和程序代码的组织，另外内部类拥有直接访问其外部类所有成员（包括 private 的）的权限（无需任何关键字修饰）。

3）内部类不可以在其他类或 main 方法里实例化，必须使用如下方法（非静态内部类）

外部类.内部类 对象名=new 外部类().new 内部类();

静态内部类调用方式：

外部类.内部类 对象名=new 外部类.内部类();

4）非静态内部类不可以声明静态成员；静态内部类的非静态成员可以访问其外部类的静态成员，声明为静态的成员不可以访问外部的非静态成员。

5、 访问修饰符限制：

Private protected friendly(default) public

同类 Y Y Y Y

同包不同类 N Y Y Y

同包子类 N Y Y Y

不同包不同类 N N N Y

不同包子类 N Y N Y

6、 Static 关键字的使用：

1）类成员，直接使用 类名.成员 调用。

2）静态方法只能访问静态成员。

3）静态方法不能使用 this、super 关键字。

4）静态方法不能被非静态方法重写或重载。

7、 final 关键字：

1）被 final 修饰的变量为常量不能改变。

2）被 final 修饰的方法不可以重写。

3）被 final 修饰的类不能被继承。

8、 abstract 关键字：

1）被 abstract 修饰的类不能实例化。

2）被 abstract 修饰的方法只能在子类中实现。

9、 native 关键字：非 Java 语言的编写，例如 JNI 技术。

10、 synchronized 关键字：多线程的同步访问控制。

11、运行时异常与一般异常有何异同？

异常表示程序运行过程中可能出现的非正常状态，运行时异常表示虚拟机的通常操作中可能遇到的异常，是一种常见运行错误。java 编译器要求方法必须声明抛出可能发生的非运行时异常，但是并不要求必须声明抛出未被捕获的运行时异常。

（Throwable 是所有 Java 程序中错误处理的父类，有两种资类：Error 和 Exception。

Error：表示由 JVM 所侦测到的无法预期的错误，由于这是属于 JVM 层次的严重错误，导致 JVM 无法继续执行，因此，这是不可捕捉到的，无法采取任何恢复的操作，顶多只能显示错误信息。

Exception：表示可恢复的例外，这是可捕捉到的。

Java 提供了两类主要的异常:runtime exception 和 checked exception。

checked 异常也就是我们经常遇到的 IO 异常，以及 SQL 异常都是这种异

常。对于这种异常，JAVA 编译器强制要求我们必需对出现的这些异常进行 catch。所以，面对这种异常不管我们是否愿意，只能自己去写一大堆 catch 块去处理可能的异常。

但是另外一种异常：runtime exception，也称运行时异常，我们可以不处理。当出现这样的异常时，总是由虚拟机接管。比如：我们从来没有人去处理过 NullPointerException 异常，它就是运行时异常，并且这种异常还是最常见的异常之一。

出现运行时异常后，系统会把异常一直往上层抛，一直遇到处理代码。如果没有处理块，到最上层，如果是多线程就由 Thread.run()抛出，如果是单线程就被 main()抛出。抛出之后，如果是线程，这个线程也就退出了。如果是主程序抛出的异常，那么这整个程序也就退出了。运行时异常是 Exception 的子类，也有一般异常的特点，是可以被 Catch 块处理的。只不过往往我们不对他处理罢了。也就是说，你如果不对运行时异常进行处理，那么出现运行时异常之后，要么是线程中止，要么是主程序终止。

如果不想终止，则必须扑捉所有的运行时异常，决不让这个处理线程退出。队列里面出现异常数据了，正常的处理应该是把异常数据舍弃，然后记录日志。不应该由于异常数据而影响下面对正常数据的处理。在这个场景这样处理可能是一个比较好的应用，但并不代表在所有的场景你都应该如此。如果在其它场景，遇到了一些错误，如果退出程序比较好，这时你就可以不太理会运行时异常，或者是通过对异常的处理显式的控制程序退出。

异常处理的目标之一就是为了把程序从异常中恢复出来。

）

12、编程题： 写一个 Singleton 出来。

Singleton 模式主要作用是保证在 Java 应用程序中，一个类 Class 只有一个实例存在。

一般 Singleton 模式通常有几种种形式:

第一种形式： 定义一个类，它的构造函数为 private 的，它有一个 static 的 private 的该类变量，在类初始化时实例话，通过一个 public 的 getInstance 方法获取对它的引用,继而调用其中的方法。

```
public  class  Singleton  {
private  Singleton(){}
            //在自己内部定义自己一个实例，是不是很奇怪？
            //注意这是 private  只供内部调用
            private  static  Singleton  instance  =  new  Singleton()
;
            //这里提供了一个供外部访问本 class 的静态方法，可以直接访
问
            public  static  Singleton  getInstance()  {
                return  instance;
            }
        }
        第二种形式:
public  class  Singleton  {
    private  static  Singleton  instance  =  null;
    public  static  synchronized  Singleton  getInstance()  {
    //这个方法比上面有所改进，不用每次都进行生成对象，只是第一
次
    //使用时生成实例，提高了效率！
    if  (instance==null)
        instance＝new  Singleton();
return  instance;      }
}
```

其他形式:

定义一个类，它的构造函数为 private 的，所有方法为 static 的。

一般认为第一种形式要更加安全些

13、分类列举服务器和组件技术：

1）服务器端技术：Jsp、Servlet；

2）组件技术：JavaBean、EJB。

14、Http 与 Https：Https 即多了安全的 Http，s（Security Socket Layer）指加密套接字协议层（简写 SSL）。

15、OSI（Open System Interconnection）网络抽象模型：

1）由国际标准化组织（ISO）提出。

2）将互联网分为七层，从下至上分别为：物理层（physical）、数据链路层（data link）、网络层（network）、传送层（transport）、会话层（session）、表示层（presentation）、应用层 （application）。底层通过提供接口支持上层功能。各层详解：

物理层：LAN/ATM，为硬件层。

数据链路层：LAN/ATM

网络层：IP 协议，IOS

传输层：TCP/UDP 协议，支持 Java Socket。

会话层：

表示层：HTML、XML

应用层：HTTP 协议，使用 Java Servlet/JSP

<第八层（Web 服务层）：SOAP/UDDI>

16、J2EE 的容器与服务器：

容器负责 EJB 组件中生命周期的控制；

服务器包含在容器外，提供系统级操作底层服务，包括事务、事件、多线程……。

17、继承限制：

父类对象不可以赋给子类对象，因为子类可能具有更多的成员，反之可以。

18、逻辑操作：c=(a>b)?a:b;等同于下式

```
if(a>b) c=a; else c=b;
```

19、列举常见集合框架类型：

1）List、Set、Map。由这三个接口实现出 ArrayList、LinkedList、HashSet、TreeSet、HashMap、TreeMap 等常用集合框架。

2）Vector 属于重量级组件不推荐使用。

3)Map 类型维护键/值对,Hashtable 与 HashMap 相近但效率略低于 HashMap、高于 TreeMap，TreeMap 优点是可以排序。

4）Set 类型可装入唯一值，HashSet 效率高于 TreeSet 但 TreeSet 可以维护内部元素的排序状态。

5）List 类型可按某种特定顺序维护元素。ArrayList 允许快速随机访问，但如果添加或删除位于中间的元素时效率很低；LikedList 提供最 佳循序访问及快速的中间位置添加删除元素，并有 addFirst、addLast、getFirst、getLast、removeFirst、 removeLast 方法。

20、面向对象的特征：

1）继承：通过子类可以实现继承，子类继承父类的所有状态和行为，同时添加自身的状态和行为。

2）封装：将代码及处理数据绑定在一起的一种编程机制，该机制保证程序和数据不受外部干扰。

3）多态：包括重载和重写。重载为编译时多态，重写是运行时多态。重载必须是同类中名称相同参数不同（包括个数不同和类型不同），但返回类型不同不构成重载；重写发生于子类对父类的覆盖，子类继承父类方法名相同、参数列表相同、返回类型相同才构成重写。

21、Java 命名规范：必须以英文字母、下划线（’_’）或’$’开始，其余可以有数字但不允许 包含空格，且组合后的名称不能是 Java 关键字或保留字。

匈牙利命名法：以 m 开始为类成员变量，以 g 开始为全局变量，以 v 开始为本地局部变量，常量命名一般不以下划线、美元符开始。

驼峰命名：一般称由多个单词或缩写组成的变量名，并且该变量名每个单词首字母均为大写（一般类名全部首字母大写，方法或属性名第一个字母小写）的称为驼峰命名。

22、Java 语言共包含 47 个关键字。

23、设计模式：

一个设计模式描述了一个被证实可行的方案。这些方案非常普遍，是具有完整定义的最常用的模式。一般模式有 4 个基本要素：模式名称(pattern name)、问题（problem)、解决方案（solution)、效果（consequences)。

常见 23 种模式概述：

1） 抽象工厂模式（Abstract Factory)：提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。

2） 适配器模式（Adapter)：将一个类的接口转换成客户希望的另外一个接口。适配器模式使得原本由于接口不兼容而不能一起工作的类可以一起工作。

3） 桥梁模式（Bridge)：将抽象部分与它的实现部分分离，使它们都可以独立地变化。

4） 建造模式（Builder)：将一个复杂对象的构建与它的表示分离，使同样的构建过程可以创建不同的表示。

5） 责任链模式（Chain of Responsibility)：为解除请求的发送者和接收者之间耦合，而使多个对象都有机会处理这个请求。将这些对象连成一条链，并沿着这条链传递该请求，直到有一个对象处理它。

6） 命令模式（Command)：将一个请求封装为一个对象，从而可用不同的请求对客户进行参数化；对请求排队或记录请求日志，以及支持可取消的操作。

7） 合成模式（Composite)：将对象组合成树形结构以表示“部分－整体”的层次结构。它使得客户对单个对象和复合对象的使用具有一致性。

8） 装饰模式（Decorator)：动态地给一个对象添加一些额外的职责。就扩展功能而言，它能生成子类的方式更为灵活。

9） 门面模式（Facade)：为子系统中的一组接口提供一个一致的界面，门面模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。

10） 工厂方法（Factory Method)：定义一个用于创建对象的接口，让子类决定将哪一个类实例化。Factory Method 使一个类的实例化延迟到其子类。

11） 享元模式(Flyweight)：运用共享技术以有效地支持大量细粒度的对象。

12） 解释器模式(Interpreter)：给定一个语言，定义它的语法的一种表示，并定义一个解释器，该解释器使用该表示解释语言中的句子。

13） 迭代子模式（Iterator)：提供一种方法顺序访问一个聚合对象中的各

个元素，而又不需暴露该对象的内部表示。

14） 调停者模式（Mediator）：用一个中介对象来封装一系列的对象交互。中介者使各对象不需要显式的内部表示。

15） 备忘录模式（Memento）：在不破坏封装性的前提下，捕获一个对象的内部状态，并在该对象之外保存这个状态。这样以后就可将该对象恢复到保存的状态。

16） 观察者模式（Observer）：定义对象间的一种一对多的依赖关系，以便当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并自动刷新。

17） 原始模型模式（Prototype）：用原型实例指定创建对象的种类，并且通过拷贝这个原型创建新的对象。

18） 代理模式（Proxy）：为其他对象提供一个代理以控制对这个对象的访问。

19） 单例模式（Singleton）：保证一个类仅有一个实例，并提供一个访问它的全局访问点。

20） 状态模式（State）：允许一个对象在其内部状态改变时改变它的行为。对象看起来似乎修改了它所属的类。

21） 策略模式（Strategy）：定义一系列的算法，把它们一个个封装起来，并且使它们可相互替换。本模式使得算法的变化可独立于使用它的客户。

22） 模板模式（Template Method）：定义一个操作中的算法的骨架，而将一些步骤延迟到子类中。模板方法使得子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤。

23） 访问者模式（Visitor）：表示一个作用于某对象结构中的各元素的操作。该模式可以实现在不改变各元素的类的前提下定义作用于这些元素的新操作。

24、STRUTS 的应用(如 STRUTS 架构)

Struts 是采用 Java Servlet/JavaServer Pages 技术，开发 Web 应用程序的开放源码的 framework。 采用 Struts 能开发出基于 MVC(Model-View-Controller)设计模式的应用构架。 Struts 有如下的主要功能： 一.包含一个 controller servlet，能将用户的请求发送到相应的 Action 对象。 二.JSP 自由 tag 库，并且在 controller servlet 中提供关联支持，

帮助开发员创建交互式表单应用。 三.提供了一系列实用对象：XML 处理、通过 Java reflection APIs 自动处理 JavaBeans 属性、国际化的提示和消息。

25、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI,WSDL 解释。

Web ServiceWeb Service 是基于网络的、分布式的模块化组件，它执行特定的任务，遵守具体的技术规范，这些规范使得 Web Service 能与其他兼容的组件进行互操作。

JAXP(Java API for XML Parsing) 定义了在 Java 中使用 DOM, SAX, XSLT 的通用的接口。这样在你的程序中你只要使用这些通用的接口，当你需要改变具体的实现时候也不需要修改代码。

JAXM(Java API for XML Messaging) 是为 SOAP 通信提供访问方法和传输机制的 API。

WSDL 是一种 XML 格式，用于将网络服务描述为一组端点，这些端点对包含面向文档信息或面向过程信息的消息进行操作。这种格式首先对操作和消息进行抽象描述，然后将其绑定到具体的网络协议和消息格式上以定义端点。相关的具体端点即组合成为抽象端点（服务）。

SOAP 即简单对象访问协议(Simple Object Access Protocol)，它是用于交换 XML 编码信息的轻量级协议。

UDDI 的目的是为电子商务建立标准；UDDI 是一套基于 Web 的、分布式的、为 Web Service 提供的、信息注册中心的实现标准规范，同时也包含一组使企业能将自身提供的 Web Service 注册，以使别的企业能够发现的访问协议的实现标准。

26、存储过程和函数的区别

存储过程是用户定义的一系列 sql 语句的集合，涉及特定表或其它对象的任务，用户可以调用存储过程，而函数通常是数据库已定义的方法，它接收参数并返回某种类型的值并且不涉及特定用户表。

27、 你认为一个项目如何进行才正确？（比如：尽一切可能快的完成任务或完

全按照类似 CMM 来操作）

根据林锐博士的观点：企业的根本目标是合法地赚取尽可能多的利润，使企业利益最大化。企业所有的特定目标和行动都是围绕上述根本目标开展的，任何背离根本目标的行动都将对企业造成伤害，应当杜绝。

基于此任何人都不要强调我将严格遵守 XX 模式，带领团队开发出具有 XX 等级的产品，企业需要的是能够带领团队按时、合格的开发出产品的 Manager。

28、 你经常看或仔细研读过的书有哪些？

不用回答你看过的课本，枚举几个经典的当然前提是必须真的看过至少浏览过主题和目录。比如《Java 编程思想》、《Java 模式》、《人月神话》等，由于将来要做的是 team 中的替补 leader 或真正的 leader 所以你必须说出软工的东西。

29、 你认为你应聘我们公司的项目经理，你自身的优势在哪？

1）融洽，没有领导希望你带领团队每天打嘴仗然后他还要去开屁股（Sorry，这似乎不很文明），你必须说明你能在以往团队中与其他人和谐相处。

2）技术，千万不要谦虚，对方要的就是技术过硬、能力出众的人才，你只需要说明你成功解决过什么难题并且你对 J2EE、XX 中间件、XX 系统有多么的熟悉。

4、 如果给你一个 Team，公司决定让你的 Team 开发 A 产品，OK 这恰好是你的强项，你们很快开发出来了，但很沮丧的结果是它（你们的产品）没有销路；经过讨论公司又决定让你的 Team 开发 B 产品，OK 这恰好又是你的强项，你们很快开发出来了，但很沮丧的结果是它（你们的产品）又没有销路。请问你怎么面对这个问题，你是否觉得决策层很无能甚至要推翻他或者去一个更有前途的公司？

不要以为这个故事很单纯，这应该是人力在考核你。你一定要告诉她（人力多半是女的），我个人对于失败的项目肯定会难过（是的，我想了很久才想出“难过”这个词，它很中性），不过我认为这恰恰认证了公司在革新和新技术探索方面的魄力（自己想怎么说，如果天下人都说这句那才是悲剧呢），我肯定会以个人的名义向 领导层提出我自己的建议和看法当然它未必正确，我不会离开公司，因为有点小挫折未尝不是好事。（你应该在这个问题上好好想

想，尽量发挥到 10 分钟）

30、 你认为项目中最重要的是哪些过程？

分析、设计阶段（也可以加上测试，但千万别说编码或开发阶段），根据《人月神话》的观点：1/3 计划；1/6 编码；1/4 构件测试和早期系统测试；1/4 系统测试，所有的构件已完成

但根据国内目前的状况一般公司不会有很多的分析与设计时间（这取决于公司规模和时间成本），这样在一个工期很紧张的项目中我们应该尽量分配出进度优先级 来，首先拿出客户最希望看到的和最能证明成果的东西来，其他的留待 2 期甚至 3 期去作，你可以告诉客户需要进一步调试（专业人员的欺骗手段，实际上就是在进 行后续的开发）。

31、 如果给你一个 4－6 个人的 Team，那么你怎么分配他们、管理他们？

管理能力和经验的综合题，可能没有人有相同的观点，那你可以按照某些思路来侧面解答：我会挑选一个技术过硬的人作为我的替补和项目的轻骑兵，是的团队中必须有机动人员，否则你的项目十有八九会夭折。其他的人会被平均的分配任务。

我们会在每周进行全面的任务分配，每个人获取一周的大概工作，然后每天的工作由他自己完成并汇报。（很好，如果答出这些就差不多了，多说可能会出现漏洞）

32、 简述常用的软件开发文档。

1） 可行性研究报告（某些公司或模型没有）

2） 项目开发计划

3） 软件需求说明书（必有）

4） 数据要求说明书

5） 概要设计说明书（必有）

6） 详细设计说明书（必有）

7） 数据库设计说明书（必有）

8） 用户手册（一般会有）

9） 操作手册（必有）

10） 模块开发卷宗

11） 测试计划（必有）

12） 测试分析报告

13） 开发进度月报

14） 项目开发总结报告

33、 简述类的关系。

1） 当一个类是“一种”另一个类时：is-a 关系

2） 当两个类之间有关联时：

一个类“包含”另一个类：has-a 关系

一个类“使用”另一个类

还可以细分有聚合和组合（UML 宝典）或聚集和组成（包括国内某些知名学术团体都这么说）。

聚集（aggregation）表示整体与各部分之间的关系。例如汽车与轮胎，没有了汽车轮胎依然是一个整体。（用空心菱形表示）

组成是一种整体和部分所属更强的聚集关系，每个部分只能属于一个整体，没有整体部分也就没有存在的价值。比如桌子和桌腿，没有桌子也就没有桌腿的价值了。（用实心菱形表示）

> Java 开发 QQ 群：**518397333**，无广告、无买卖，这里只有程序猿（媛）。

学习网站推荐

1. 易极客：http://easygeek.com.cn/