## 基本功

* 面向对象的特征

->抽象，封装，继承，多态

参考<https://www.cnblogs.com/gudulijia/p/5841607.html>

* final, finally, finalize 的区别

final用来修饰类,方法,变量 final修饰的对象都是线程安全的

finally主要是方法最肯定终执行

finalize主要是通知jvm的垃圾回收器进行gc操作,具体还是依赖gc调度

* int 和 Integer 有什么区别

int是基础类型不初始化默认为0 而Integer默认为null

int和Integer进行比较的时候会被自动拆箱

对于Integer对象比较时需要注意 -128~127范围对象的比较 Integer.valueOf()会获取缓存的对象，而不会新创建对象

* 重载和重写的区别

重载是对方法的拓展而重写则是存在于继承或者接口实现的类，主要是覆盖父类的方法逻辑

* 抽象类和接口有什么区别

抽象类是包含了抽象方法的class而接口定义使用interface

一个类只能extends一个抽象类 但是可以implements多个接口

接口中的方法,变量都是public权限的

* 说说反射的用途及实现

在运行期可以获得一个对象的类,从而构造新的对象,可以获取这个类的成员变量和方法.

最常用动态代理.最大的好处可以进行系统内部依赖的解耦(spring aop).

* 说说自定义注解的场景及实现

注解是 Java 1.5 引入的，目前已被广泛应用于各种 Java 框架，如 Hibernate，[Jersey](http://www.journaldev.com/498/rest-using-jersey-complete-tutorial-with-jaxb-exception-handling-and-client-program" \t "_blank)，Spring。注解相当于是一种嵌入在程序中的元数据，可以使用注解解析工具或编译器对其进行解析，也可以指定注解在编译期或运行期有效

* HTTP 请求的 GET 与 POST 方式的区别

区别:

Get可以被缓存 Post不能被缓存

Get有数据长度限制(url最大长度2048个字符)而Post没有长度限制

Get只允许ASCII字符 Post没有限制

Get传输数据是url显示显现 Post不显示在url中

Get幂等 Post不幂等

* session 与 cookie 区别

session 在服务器端，cookie 在客户端（浏览器）

session 的运行依赖 sessionid，而 sessionid 是存在cookie中的，也就是说，如果浏览器禁用了 cookie ，同时 session也会失效

session 可以放在 文件、数据库、或内存中都可以

* session 分布式处理

利用数据库 -> 添加数据库和表来记录用户session 优点 简单 缺点对数据库压力比较大

利用cookie ->将用户session返回给浏览器,下次主动获取cookie信息来保证session同步. 优点服务端实现比较简单 缺点客户端依赖比较强,安全性很难保证.

利用redis -> 存放在内存中,性能比较高,redis支持集群. 缺点:依赖redis集群,如果集群出现问题,session分布式方案就不能保证.

利用nginx -> 例如tomcat本身支持集群方式同步,但是效率比较低. 利用nginx负载转发进行ip哈希,可以保证单一用户请求始终落地到后端的一台服务器,但是假如这台机器宕机,落地到本机器的用户就会出现问题

* JDBC 流程

注册驱动->链接数据库->创建statement->发送执行语句->接收数据库返回的结果集->关闭链接

* MVC 设计思想

Mvc 包含了model层 view层 controller层是现在很多技术领域常用的设计思想

Model层主要包含了业务领域模型

View层主要是展示模型层相关数据

Controller层主要是处理view层的请求获取model层数据进行流程控制

好处:分层处理,降低了模型层和视图层之间的耦合,回想以前jsp中写业务逻辑代码.从而提高了代码的可维护性.

缺点:代码的解耦提高了代码的复杂程度.对架构设计的要求也会比以前高

* equals 与 == 的区别

equals比较的是具体的对象的值,自定义对象要具体看重写的equals方法

==比较的是内存引用地址

## 集合

* List 和 Set 区别
* List 和 Map 区别
* Arraylist 与 LinkedList 区别
* ArrayList 与 Vector 区别
* HashMap 和 Hashtable 的区别
* HashSet 和 HashMap 区别
* HashMap 和 ConcurrentHashMap 的区别
* HashMap 的工作原理及代码实现
* ConcurrentHashMap 的工作原理及代码实现

## 多线程

* 创建线程的方式及实现
* sleep() 、join（）、yield（）有什么区别
* 说说 CountDownLatch 原理
* 说说 CyclicBarrier 原理
* 说说 Semaphore 原理
* 说说 Exchanger 原理
* 说说 CountDownLatch 与 CyclicBarrier 区别
* ThreadLocal 原理分析
* 讲讲线程池的实现原理
* 线程池的几种方式
* 线程的生命周期
* 说说线程安全问题
* volatile 实现原理
* synchronize 实现原理
* synchronized 与 lock 的区别
* CAS 乐观锁
* ABA 问题
* 乐观锁的业务场景及实现方式

## 数据存储

* MySQL 索引使用的注意事项
* 说说反模式设计
* 说说分库与分表设计
* 分库与分表带来的分布式困境与应对之策
* 说说 SQL 优化之道
* MySQL 遇到的死锁问题
* 存储引擎的 InnoDB 与 MyISAM
* 数据库索引的原理
* 为什么要用 B-tree
* 聚集索引与非聚集索引的区别
* limit 20000 加载很慢怎么解决
* 选择合适的分布式主键方案
* 选择合适的数据存储方案
* ObjectId 规则
* 聊聊 MongoDB 使用场景
* 倒排索引
* 聊聊 ElasticSearch 使用场景

## 缓存使用

* Redis 有哪些类型
* Redis 内部结构
* 聊聊 Redis 使用场景
* Redis 持久化机制
* Redis 如何实现持久化
* Redis 集群方案与实现
* Redis 为什么是单线程的
* 缓存奔溃
* 缓存降级
* 使用缓存的合理性问题

## 消息队列

* 消息队列的使用场景
* 消息的重发补偿解决思路
* 消息的幂等性解决思路
* 消息的堆积解决思路
* 自己如何实现消息队列
* 如何保证消息的有序性

## Spring

* BeanFactory 和 ApplicationContext 有什么区别
* Spring Bean 的生命周期
* Spring IOC 如何实现
* 说说 Spring AOP
* Spring AOP 实现原理
* 动态代理（cglib 与 JDK）
* Spring 事务实现方式
* Spring 事务底层原理
* 如何自定义注解实现功能
* Spring MVC 运行流程
* Spring MVC 启动流程
* Spring 的单例实现原理
* Spring 框架中用到了哪些设计模式
* Spring 其他产品（Srping Boot、Spring Cloud、Spring Secuirity、Spring Data、Spring AMQP 等）

## Netty

* 为什么选择 Netty
* 说说业务中，Netty 的使用场景
* 原生的 NIO 在 JDK 1.7 版本存在 epoll bug
* 什么是 TCP 粘包 / 拆包
* TCP 粘包 / 拆包的解决办法
* Netty 线程模型
* 说说 Netty 的零拷贝
* Netty 内部执行流程
* Netty 重连实现

## 微服务

* 前后端分离是如何做的
* 微服务哪些框架
* 你怎么理解 RPC 框架
* 说说 RPC 的实现原理
* 说说 Dubbo 的实现原理
* 你怎么理解 RESTful
* 说说如何设计一个良好的 API
* 如何理解 RESTful API 的幂等性
* 如何保证接口的幂等性
* 说说 CAP 定理、 BASE 理论
* 怎么考虑数据一致性问题
* 说说最终一致性的实现方案
* 你怎么看待微服务
* 微服务与 SOA 的区别
* 如何拆分服务
* 微服务如何进行数据库管理
* 如何应对微服务的链式调用异常
* 对于快速追踪与定位问题
* 微服务的安全

## 分布式

* 谈谈业务中使用分布式的场景
* Session 分布式方案
* 分布式锁的场景
* 分布是锁的实现方案
* 分布式事务
* 集群与负载均衡的算法与实现
* 说说分库与分表设计
* 分库与分表带来的分布式困境与应对之策

## 安全问题

* 安全要素与 STRIDE 威胁
* 防范常见的 Web 攻击
* 服务端通信安全攻防
* HTTPS 原理剖析
* HTTPS 降级攻击
* 授权与认证
* 基于角色的访问控制
* 基于数据的访问控制

## 性能优化

* 性能指标有哪些
* 如何发现性能瓶颈
* 性能调优的常见手段
* 说说你在项目中如何进行性能调优

## 需求分析

* 你如何对需求原型进行理解和拆分
* 说说你对功能性需求的理解
* 说说你对非功能性需求的理解
* 你针对产品提出哪些交互和改进意见
* 你如何理解用户痛点

## 设计能力

* 说说你在项目中使用过的 UML 图
* 你如何考虑组件化
* 你如何考虑服务化
* 你如何进行领域建模
* 你如何划分领域边界
* 说说你项目中的领域建模
* 说说概要设计

## 设计模式

* 你项目中有使用哪些设计模式
* 说说常用开源框架中设计模式使用分析
* 说说你对设计原则的理解
* 23 种设计模式的设计理念
* 设计模式之间的异同，例如策略模式与状态模式的区别
* 设计模式之间的结合，例如策略模式 + 简单工厂模式的实践
* 设计模式的性能，例如单例模式哪种性能更好。

## 业务工程

* 你系统中的前后端分离是如何做的
* 说说你的开发流程
* 你和团队是如何沟通的
* 你如何进行代码评审
* 说说你对技术与业务的理解
* 说说你在项目中经常遇到的 Exception
* 说说你在项目中遇到感觉最难 Bug，怎么解决的
* 说说你在项目中遇到印象最深困难，怎么解决的
* 你觉得你们项目还有哪些不足的地方
* 你是否遇到过 CPU 100% ，如何排查与解决
* 你是否遇到过 内存 OOM ，如何排查与解决
* 说说你对敏捷开发的实践
* 说说你对开发运维的实践
* 介绍下工作中的一个对自己最有价值的项目，以及在这个过程中的角色

## 软实力

* 说说你的亮点
* 说说你最近在看什么书
* 说说你觉得最有意义的技术书籍
* 工作之余做什么事情
* 说说个人发展方向方面的思考
* 说说你认为的服务端开发工程师应该具备哪些能力
* 说说你认为的架构师是什么样的，架构师主要做什么
* 说说你所理解的技术专家

以上摘自<http://blog.720ui.com/> 梁桂钊博客

## 数据结构

* 数组