[java 1](#_Toc256000000)

[1. 01-前端知识 10](#_Toc256000002)

[1.1. html 10](#_Toc256000003)

[1.2. css上 10](#_Toc256000004)

[1.3. js 10](#_Toc256000005)

[1.4. vue 10](#_Toc256000006)

[1.5. react 10](#_Toc256000007)

[1.6. AngularJS 10](#_Toc256000008)

[1.7. bootstarp 10](#_Toc256000009)

[1.8. heigchar 10](#_Toc256000010)

[2. 02-java基础知识 10](#_Toc256000011)

[2.1. 算法和数据结构 10](#_Toc256000012)

[2.1.1. 数据结构 10](#_Toc256000013)

[2.1.2. 算法 10](#_Toc256000014)

[2.2. 设计模式，多态 11](#_Toc256000015)

[2.2.1. 23种设计模式 11](#_Toc256000016)

[2.2.2. 设计原则 11](#_Toc256000017)

[2.3. 集合结构 11](#_Toc256000018)

[2.3.1. Iterator 12](#_Toc256000019)

[2.4. java8新特性 13](#_Toc256000020)

[2.4.1. lambda 13](#_Toc256000021)

[2.4.2. map底层改变 13](#_Toc256000022)

[3. 03-jvm 13](#_Toc256000023)

[3.1. jvm加载过程 13](#_Toc256000024)

[3.2. 动态加载 13](#_Toc256000025)

[4. 04-基础框架 13](#_Toc256000026)

[4.1. springMvc 13](#_Toc256000027)

[4.2. spring 14](#_Toc256000028)

[4.3. struts2 14](#_Toc256000029)

[4.4. mybatis 14](#_Toc256000030)

[4.5. hibernate 14](#_Toc256000031)

[4.6. redis 14](#_Toc256000032)

[4.7. netty 14](#_Toc256000033)

[4.7.1. BIO-NIO-AIO的原理 14](#_Toc256000034)

[4.8. shiro 权限 14](#_Toc256000035)

[4.9. Quartz 14](#_Toc256000036)

[5. 05-微服务 14](#_Toc256000037)

[5.1. dubbo 14](#_Toc256000038)

[5.2. springboot 14](#_Toc256000039)

[5.2.1. 工具集 14](#_Toc256000040)

[5.2.2. Annotation讲解 14](#_Toc256000041)

[5.2.3. 集成Mybatis实现多数据源路由实战 15](#_Toc256000042)

[5.2.4. 集成Dubbo实战 15](#_Toc256000043)

[5.2.5. 集成Redis缓存实战 15](#_Toc256000044)

[5.2.6. 集成Swagger2 API管理及测试体系 15](#_Toc256000045)

[5.2.7. 实现多环境配置动态解析 15](#_Toc256000046)

[5.3. springcloud 15](#_Toc256000047)

[5.3.1. 1-注册中心 15](#_Toc256000048)

[5.3.2. 2-Fegin声明式服务调用 16](#_Toc256000049)

[5.3.3. 3-Ribbon 16](#_Toc256000050)

[5.3.4. 4-Hystrix服务熔断降级方式 17](#_Toc256000051)

[5.3.5. 5-Zuul实现微服务网关 17](#_Toc256000052)

[5.3.6. 6-Config分布式统一配置中心 17](#_Toc256000053)

[5.3.7. 7-Sleuth和Zipkin进行分布式链路跟踪 20](#_Toc256000054)

[5.3.8. 8-Bus消息总线 20](#_Toc256000055)

[5.3.9. 基于Hystrix实现接口降级实战 21](#_Toc256000056)

[5.3.10. SpringBoot集成Spring Cloud实现统一整合方案 21](#_Toc256000057)

[5.4. Docker虚拟化 21](#_Toc256000058)

[5.4.1. Docker的镜像，仓库，容器 21](#_Toc256000059)

[5.4.2. DockerFile构建Lnmp环境部署个人微博Wordpress 21](#_Toc256000060)

[5.4.3. Docker Compose构建LNMP环境部署个人博客wordpress 21](#_Toc256000061)

[5.4.4. Docker网络组成，路由互联，openvswitch 21](#_Toc256000062)

[5.4.5. 基于swarm构建Docker集群实战 21](#_Toc256000063)

[5.4.6. Kubernetes简介 22](#_Toc256000064)

[5.5. 浅谈微服务架构 22](#_Toc256000065)

[5.5.1. SOA架构和微服务架构之间的区别和联系 22](#_Toc256000066)

[5.5.2. SpringBoot流行因素及能够解决什么问题 22](#_Toc256000067)

[5.5.3. 寄去全局分析Spring Cloud各个组件所解决问题 22](#_Toc256000068)

[5.6. 微服务支撑 22](#_Toc256000069)

[5.6.1. 集成测试 22](#_Toc256000070)

[5.6.2. 云龙流水线 22](#_Toc256000071)

[5.6.3. 服务器支撑 22](#_Toc256000072)

[6. 06-编程工具 22](#_Toc256000073)

[6.1. git 22](#_Toc256000074)

[6.2. svn 22](#_Toc256000075)

[6.3. eclipse 22](#_Toc256000076)

[6.4. nginx 22](#_Toc256000077)

[6.5. maven 22](#_Toc256000078)

[6.6. ea软件 22](#_Toc256000079)

[6.6.1. 数据库建模 22](#_Toc256000080)

[6.6.2. 接口设计 22](#_Toc256000081)

[6.6.3. 流程图 22](#_Toc256000082)

[6.6.4. 时序图 23](#_Toc256000083)

[6.6.5. 类关系图 23](#_Toc256000084)

[7. 07-数据库 23](#_Toc256000085)

[7.1. mysql 23](#_Toc256000086)

[7.2. oracle 23](#_Toc256000087)

[7.2.1. oracle设计 23](#_Toc256000088)

[7.3. redis 23](#_Toc256000089)

[7.4. 数据库设计 23](#_Toc256000090)

[7.4.1. 1多对多 23](#_Toc256000091)

[8. 08-并发编程 23](#_Toc256000092)

[8.1. JAVA内存模型（JMM） 23](#_Toc256000093)

[8.1.1. 线程通信 23](#_Toc256000094)

[8.1.2. 消息传递 23](#_Toc256000095)

[8.2. 内存模型 24](#_Toc256000096)

[8.2.1. 重排序 24](#_Toc256000097)

[8.2.2. 顺序一致性 24](#_Toc256000098)

[8.2.3. happens-before 24](#_Toc256000099)

[8.2.4. as-if-serial 24](#_Toc256000100)

[8.3. synchronized 24](#_Toc256000101)

[8.3.1. 同步，重量级锁 24](#_Toc256000102)

[8.3.2. synchronized原理 24](#_Toc256000103)

[8.3.3. 锁优化 24](#_Toc256000104)

[8.4. volatile 24](#_Toc256000105)

[8.4.1. volatile实现机制 24](#_Toc256000106)

[8.4.2. 内存语义 24](#_Toc256000107)

[8.4.3. 内存模型 24](#_Toc256000108)

[8.5. DCL 24](#_Toc256000109)

[8.5.1. 单例模型 24](#_Toc256000110)

[8.5.2. DCL 25](#_Toc256000111)

[8.5.3. 解决方案 25](#_Toc256000112)

[8.6. 并发编程 25](#_Toc256000113)

[8.6.1. AQS 25](#_Toc256000114)

[8.6.2. CAS 25](#_Toc256000115)

[8.7. 锁 LOCK 25](#_Toc256000116)

[8.7.1. ReentrantLock 25](#_Toc256000117)

[8.7.2. ReentrantReadWriteLock 25](#_Toc256000118)

[8.7.3. Condition 25](#_Toc256000119)

[8.8. 并发工具类 25](#_Toc256000120)

[8.8.1. CyclicBarrier 25](#_Toc256000121)

[8.8.2. CountDownLatch 25](#_Toc256000122)

[8.8.3. Semphore 25](#_Toc256000123)

[8.9. 并发集合 26](#_Toc256000124)

[8.9.1. ConcurrentHashMap 26](#_Toc256000125)

[8.9.2. COncurrentLinkedQueue 26](#_Toc256000126)

[8.10. 原子操作 26](#_Toc256000127)

[8.10.1. 基本类型 26](#_Toc256000128)

[8.10.2. 数组 26](#_Toc256000129)

[8.10.3. 引用类型 26](#_Toc256000130)

[8.10.4. 原子更新字段类 26](#_Toc256000131)

[8.11. 线程池 26](#_Toc256000132)

[8.11.1. Executor 27](#_Toc256000133)

[8.11.2. ThreadPoolExecutor 27](#_Toc256000134)

[8.11.3. Callable和Future 28](#_Toc256000135)

[8.11.4. ScheduledExecutorService 28](#_Toc256000136)

[8.11.5. 合理配备线程池 28](#_Toc256000137)

[8.11.6. 线程池监控 28](#_Toc256000138)

[8.11.7. ExecutorCompletionService 28](#_Toc256000139)

[8.12. 其他 28](#_Toc256000140)

[8.12.1. ThreadLocal 28](#_Toc256000141)

[8.12.2. Fork/Join 28](#_Toc256000142)

[8.13. java如何实现原子操作 28](#_Toc256000143)

[8.13.1. 使用循环CAS实现原子操作 28](#_Toc256000144)

[8.13.2. 锁 28](#_Toc256000145)

[9. 09-性能调优 28](#_Toc256000146)

[9.1. 理解性能优化 28](#_Toc256000147)

[9.1.1. 性能基准 28](#_Toc256000148)

[9.1.2. 性能优化分析 28](#_Toc256000149)

[9.1.3. 衡量维度 28](#_Toc256000150)

[9.1.4. 知其然，知其所以然 29](#_Toc256000151)

[9.2. jvm调优 29](#_Toc256000152)

[9.2.1. 什么是JVM运行时数据区 29](#_Toc256000153)

[9.2.2. 什么是JVM内存模型JMM 29](#_Toc256000154)

[9.2.3. 各种垃圾回收使用场景（Throughput/cms） 29](#_Toc256000155)

[9.2.4. 理解GC日志，从日汉字看端倪 29](#_Toc256000156)

[9.2.5. 实战MAT分许dump文件 29](#_Toc256000157)

[9.3. nginx调优 29](#_Toc256000158)

[9.4. mysql调优 29](#_Toc256000159)

[9.4.1. 理解MySql底层B+ Tree机制 29](#_Toc256000160)

[9.4.2. SQL执行计划详解 29](#_Toc256000161)

[9.4.3. 索引优化详解 29](#_Toc256000162)

[9.4.4. SQL语句优化 29](#_Toc256000163)

[9.5. tomcat调优 29](#_Toc256000164)

[9.5.1. Tomcat线程模型 29](#_Toc256000165)

[9.5.2. Tomcat系统参数认识及调优 29](#_Toc256000166)

[9.5.3. 基准测试 29](#_Toc256000167)

[9.6. java代码优化 29](#_Toc256000168)

[9.6.1. 调用链过长 30](#_Toc256000169)

[9.6.2. 业务复杂多条sql connection链接查询 30](#_Toc256000170)

[9.6.3. 纵表存储inner jion关联过长 30](#_Toc256000171)

[9.6.4. 采用横表一对多，多对多存储 30](#_Toc256000172)

[9.6.5. 代码for循环调用，经常new线程池 30](#_Toc256000173)

[9.7. 利群搭建 30](#_Toc256000174)

[9.8. 影子表辅助表 30](#_Toc256000175)

[9.9. 分库分表 30](#_Toc256000176)

[9.10. 缓存，刷新缓存，同步缓存 30](#_Toc256000177)

[10. 10-分布式框架 30](#_Toc256000178)

[10.1. 分布式中间件 30](#_Toc256000179)

[10.2. 分布式id生成器 30](#_Toc256000180)

[10.3. 分布式通信 30](#_Toc256000181)

[10.4. 分布式事务 30](#_Toc256000182)

[10.5. RPC应用 30](#_Toc256000183)

[10.5.1. netty分布式通信 30](#_Toc256000184)

[10.6. 浅谈分布式框架 30](#_Toc256000185)

[10.6.1. 初始分布式架构及意义 30](#_Toc256000186)

[10.6.2. 单体应用-->分布式,分布式选型 31](#_Toc256000187)

[10.6.3. 大型分布式架构演进过程 31](#_Toc256000188)

[10.6.4. 构建分布式架构最重要的因素 31](#_Toc256000189)

[10.6.5. 分布式架构设计 31](#_Toc256000190)

[10.7. 分布式架构策略-分而治之 31](#_Toc256000191)

[10.7.1. 网络通信探究分布式原理 31](#_Toc256000192)

[10.7.2. 基于消息方式的系统间通信 31](#_Toc256000193)

[10.7.3. 通信协议传输过程中的序列化反序列化 32](#_Toc256000194)

[10.7.4. RPC通信技术 32](#_Toc256000195)

[10.7.5. 传统RPC技术在大型分布式架构下面的问题 32](#_Toc256000196)

[10.7.6. 分布式架构下的RPC解决方案 32](#_Toc256000197)

[10.7.7. 分布式系统的基石-Zookeeper 32](#_Toc256000198)

[10.7.8. Dubbo对单一应用服务化改造 32](#_Toc256000199)

[10.8. 分布式架构-中间件 32](#_Toc256000200)

[10.8.1. 分布式通信 32](#_Toc256000201)

[10.8.2. 分布式缓存 33](#_Toc256000202)

[10.8.3. 数据存储 34](#_Toc256000203)

[10.8.4. 后台服务 35](#_Toc256000204)

[10.8.5. 高性能NIO框架 35](#_Toc256000205)

[10.9. 分布式解决方案 36](#_Toc256000206)

[10.9.1. 分布式锁 36](#_Toc256000207)

[10.9.2. 分布式全局ID生成方案 36](#_Toc256000208)

[10.9.3. 分布式会话：session跨域共享及企业级单点登录解决方案实战 36](#_Toc256000209)

[10.9.4. 分布式事物解决方案实战 36](#_Toc256000210)

[10.9.5. 高并发下的服务降级，限流 36](#_Toc256000211)

[10.9.6. 分布式框架下分布式锁的解决方案 36](#_Toc256000212)

[10.9.7. 分布式框架下实现分布式定时调度 36](#_Toc256000213)

[10.9.8. 灰度发布 36](#_Toc256000214)

[11. 11-平台服务 36](#_Toc256000215)

[11.1. liunx 36](#_Toc256000216)

[11.2. docker 37](#_Toc256000217)

[11.3. redis集群 37](#_Toc256000218)

[11.4. nginx集群 37](#_Toc256000219)

[11.5. jenkins安装部署 37](#_Toc256000220)

[11.6. 工具集 37](#_Toc256000221)

[11.6.1. jdk安装 37](#_Toc256000222)

[11.6.2. xmind使用 37](#_Toc256000223)

[11.6.3. mybase 37](#_Toc256000224)

[11.7. 服务器 37](#_Toc256000225)

[11.7.1. tomcat 37](#_Toc256000226)

[11.7.2. jetty 37](#_Toc256000227)

[11.7.3. jboss 37](#_Toc256000228)

[11.7.4. aplach 37](#_Toc256000229)

[11.8. 平台服务 37](#_Toc256000230)

[11.8.1. iaas,paas,saas 37](#_Toc256000231)

[11.8.2. Infrastructure-as-a-Service（基础设施即服务） 37](#_Toc256000232)

[11.8.3. Platform-as-a-Service（平台即服务） 37](#_Toc256000233)

[11.8.4. Software-as-a-Service（软件即服务） 37](#_Toc256000234)

[12. 12-ORM框架 37](#_Toc256000235)

[12.1. mybatis 38](#_Toc256000236)

[12.1.1. 代码生成器：Generator 38](#_Toc256000237)

[12.1.2. MyBatis下1对多，多对多嵌套结果，嵌套查询 38](#_Toc256000238)

[12.1.3. 一级缓存，二级缓存使用场景及选择策略 38](#_Toc256000239)

[12.1.4. Mybatis与Spring集成Spring-mybatis.jar分析 38](#_Toc256000240)

[12.1.5. Sring集成下的SqlSession与Mapper 38](#_Toc256000241)

[12.1.6. Mybatis的事物 38](#_Toc256000242)

[12.1.7. 分析Mybatis的动态代理真正实现 38](#_Toc256000243)

[12.1.8. 手写MyBatis 2.0 38](#_Toc256000244)

[12.2. hibernate 38](#_Toc256000245)

[12.3. spring5 38](#_Toc256000246)

[12.3.1. Beans 38](#_Toc256000247)

[12.3.2. Context 38](#_Toc256000248)

[12.3.3. Transaction 38](#_Toc256000249)

[12.3.4. MVC 39](#_Toc256000250)

[12.3.5. Spring5新特性 39](#_Toc256000251)

[13. 13-网络编程 40](#_Toc256000252)

[13.1. http 40](#_Toc256000253)

[13.2. https 40](#_Toc256000254)

[13.3. socket 40](#_Toc256000255)

[13.4. webSocket 40](#_Toc256000256)

[14. 13-热门知识 40](#_Toc256000257)

[14.1. 区块链 40](#_Toc256000258)

[14.2. 人工智能 40](#_Toc256000259)

[14.3. 大数据 40](#_Toc256000260)

[15. 14-商业系统 40](#_Toc256000261)

[15.1. 三层架构 40](#_Toc256000262)

[15.2. cpq样机项目 40](#_Toc256000263)

[15.3. 电商项目 40](#_Toc256000264)

[16. 15-业务能力 40](#_Toc256000265)

[16.1. 业务推动技术 41](#_Toc256000266)

[16.2. 单元测试 41](#_Toc256000267)

[16.2.1. 功能测试 41](#_Toc256000268)

[16.2.2. 所有的测试代码方式 41](#_Toc256000269)

[16.2.3. 场景测试 41](#_Toc256000270)

[16.2.4. 压力测试 41](#_Toc256000271)

[16.2.5. 集成测试 41](#_Toc256000272)

[16.2.6. 异常测试 41](#_Toc256000273)

[16.2.7. bug测试 41](#_Toc256000274)

[16.3. PPT能力 41](#_Toc256000275)

[16.3.1. 痛点 41](#_Toc256000276)

[16.3.2. 解决了什么问题，为什么解决 41](#_Toc256000277)

[16.3.3. 行业对标怎么样 41](#_Toc256000278)

[16.3.4. 解决方式 41](#_Toc256000279)

[16.3.5. 整体架构的把控 41](#_Toc256000280)

[16.4. 演讲能力 42](#_Toc256000281)

[17. 16-项目经验 42](#_Toc256000282)

[17.1. 引用的逻辑 42](#_Toc256000283)

[17.2. 多对多版本管理 42](#_Toc256000284)

[17.3. Subtopic 3 42](#_Toc256000285)

[18. 17-面试经验 42](#_Toc256000286)

[18.1. 人事面试 42](#_Toc256000287)

[18.2. 技术面试 42](#_Toc256000288)

[19. 19-个人能力 42](#_Toc256000289)

[20. 18-领导能力 42](#_Toc256000290)