Java技能树

摘要：根据我的开发实践及理解，梳理了作为软件工程师解决现实问题应当具备的基础技能。

难度：中级。

编程开发本质是运用程序和计算机来解决现实中的各种问题。因此，编程开发人员的才智发挥载体是计算机或者说计算设备，直接武器是程序。程序员用来进行编程开发所需要的技能，归结起来主要如下：

理解计算模型和框架： 顺序计算、并发计算 （并行计算、分布式计算）； 理解应用所基于的计算模型和框架是非常重要的。传统应用通常基于顺序计算，而现代商业应用，比如云计算，通常基于分布式计算架构； 大规模的科学计算应用， 比如基因序列计算， 则基于并行计算。

高效的数据处理：熟悉常用数据结构和算法设计，能够针对计算目标设计高效的数据结构和算法并实现之， 满足性能需求。

大容量的数据管理与检索：至少精通一种主流数据库， 从数据处理中分离出的一种专门针对大量应用数据实时检索要求的技术。

系统底层机制： 进程、信号、 内存管理、API 等， 理解相关概念及实现，熟悉使用Linux&MacOS&Windows至少一种操作系统。

进程的网络通信： TCP/IP socket ， RPC ，Http ， 序列化与反序列化， 熟悉网络程序设计的底层原理。

编程语言： 面向过程、面向对象、脚本语言、函数型、浏览器语言等，最好每种类型语言掌握一种。

正则表达式：灵活地操作文本和字符串。

代码阅读与维护： 能够读懂代码，维护代码，重构或扩展代码。

错误处理技术： 异常机制、 返回值与全局变量、 直接退出或重续 等； 建造稳固可靠的应用的必要基础。

网络搜索： 寻求有益资源、远程协作伙伴及有效解答。

程序质量： 保证程序的时间性能、空间性能、安全性、可靠性、可用性、 可扩展性、可读性、可维护性、可移植性、用户体验。

设计模式： 解决实际的设计开发问题， 在现有项目的基础上扩展功能模块时应该会经常用到。

系统架构： 用于子系统的设计， 需要项目经验积累。

处理大数据量： 通常基于分布式计算架构，比如云计算基础设施。

选用可靠的工具及软件库、包： 首先必须确定具体需求，深入了解各种工具、技术与框架及其优劣，然后根据需求选择最适合的那种。

诊断和修复计算机或程序的故障： 上网搜索已有解决方案；根据原理进行推断和验证猜想； 排查缩小范围； 对比分析； 打印信息。

除了编程的其它使用计算机解决问题的技术和方法： 比如使用现有的软件。

针对每个问题，将自己掌握的技术和方法整理出来，并使用示例来说明，基本上就可以了解自己的技术水平了，同时亦可备日后用。积微知著。人们常说的那些“大技能”，其实可以分解为大量的微技能，而积累越多的微技能，在处理问题时便有了越多的选择和思考余地。具体而言：

语言： 至少熟悉六种编程语言，掌握两种编程语言。 “熟悉”是指大量学习和使用过，“掌握”是指能够使用该语言 构建大型可靠的软件系统。 对象式，函数式， 过程式，客户端语言，脚本语言，数据语言。 例如， Java / Scala / Python / C / Shell / JavaScript / SQL ；

工具： 编辑器 + IDE + 工程构建 + 版本控制 + 控制台 + 浏览器 + 笔记软件。 例如 Vim (Sublime) / IntellJ / Maven / Git / Shell / Chrome；

方法： 自顶向下，意图导引，自底向上； First Right, then Good ； 预先设计， 及早动手； 迭代式开发、发布、反馈和成长 ； 预估时间；

风格： K&R C ， Java 编程规范，排版整洁，代码简洁无冗余， 一致， 使用地道表达范式；

模型： 串行， 并发，多进程， 多线程， MapReduce ， B/S ， 异步， 回调，阻塞， 非阻塞，并行，分布式；

思想： 结构，控制，过程， 对象， 函数，声明， 原语， 集合， 组合， 抽象， 封装，复用，分层，模式，分治，缓存，中断，数学性质；

算法： 分治， 递归，动态规划， 时空权衡， 贪婪算法，优先级算法，迭代改进，深度遍历，广度遍历，问题转化；

排序： 插入排序， 选择排序，快速排序， 堆排序， 归并排序，基数排序，外部排序；

搜索： 顺序搜索， 二分搜索， 哈希搜索， 深度优先搜索，广度优先搜索，分段并行搜索；

数据结构： 位图，数组， 链表，栈，队列， 二叉树， 字符串， 哈希表，图；Json, XML, 记录，数据库； Key-Value 值对集合；非结构化存储。

文本处理： find , grep, awk, sed, sort, cut, uniq, tr, ls, cat, xargs, 正则表达式；

设计模式： 工厂， 单例，原型， 模板，策略， 状态， 观察者， 中介者， 责任链， 组合， 命令， 代理，外观，适配器，装饰器，迭代器；

应用模式： WrapperFacade, Component Configurator, Interceptor, Extension Interface, Reactor, Proactor, Asyn Completion Token, Acceptor-Connector, Strategized Locking, Thread-Safe Interface, Double-Checked Locking Optimization, Active Object, Monitor Object, Half-Sync/Half-Async, Leader/Followers, Thread-Specific Storage ；参见《面向模式的软件架构：并发与联网对象模式》

进程与网络： ps(pstree), top, kill, crontab, ifconfig, ping, telnet, netstat, tcpdump, ssh, scp, wget, curl, Socket , HTTP ； 任务管理器 ；

开发框架： Servlet, Spring, Struts, ibatis , Hibernate, Aop, jQuery , Extjs, Bootstrap, Flex ；

常用组件： 日志 , 实用工具 (字符串, 日期, JSON/XML, HTTP, Email) , 签名校验，集合容器，模板引擎，并发库，图表(Highcharts)等 ；

运行容器： apacheHttp , tomcat, jetty, nginx , Docker，JVM, 浏览器，集群；

中间件： Tair, Zookeeper, Dubbo, 数据库连接池，日志服务, 缓存, 定时任务，可靠消息服务，可靠心跳服务，工作流引擎, 负载均衡 ;

软件架构： MVCDD (Model - View - Controller - Domain - DB) ；流水线，管道与过滤器； 事件驱动； 领域建模；以数据为中心的架构；

测试技术： 质量风险分析，测试数据集， 测试用例，等价类划分， 分支覆盖， 域测试，正交表，决策表，状态转换图，边界测试， 空测试， 异常测试，漫游测试， 依赖Mock；单测；接口测试；集成测试；全链路压测；

代码组织： 头文件， 类文件，包机制，名字空间；

软件工程： 模块化， 封装，面向对象，用户故事，迭代开发，持续集成，单元测试，CodeReview，验收测试，持续交付；

测量监控： 性能测量，time ，系统/接口/业务指标，错误报警；

学习资源： 开源项目，公开课，论文，infoq, stackoverflow , blogs， google , baidu；

经典书籍： 参见《计算机专业及软件开发推荐书籍》