“它山之石”｜Sugar：这门课你可以试试这么学

2021-10-04 Sugar

《手把手带你写一门编程语言》 课程介绍 >

讲述：正霖

时长 18:17 大小 16.75M

你好，我是这门课程的助教 Sugar，曾供职于百度，现就职于某大型互联网公司，是一名软件工程师。和你一样，我也是一名编译技术的爱好者。

我们的课程更新到今天，已经过半了，不知道你学习得怎么样呢？有没有卡在哪个知识点的实现上？不要担心，如果你有任何的问题，除了在留言区留言外，还可以添加我们的微信交流群，直接 @ 我或者是宫老师，或者是其他志同道合的同学们，我们都会来帮你解决。

回归正题，今天，刚好是国庆假期，我们第一部分的起步篇也更新完了。我们就先停下来休息休息，夯实基础。在这篇分享里，我想跟你聊聊我对编译技术的看法。我也会从我个人的角度，给你总结一下已更新完的起步篇都讲了什么，以及在日常工作中，我们又可以

从哪些方向把这门课学到的知识落到实践中来。最开始，我们聊聊为什么要学习编译技术这个话题。

我为什么推荐你学习编译技术？

我大概是在 2014 年入行互联网行业的，一晃就是 7 年。这 7 年间，很多公司的口号和观点都从“从 PC 业务向移动端转型”，变成了“国内移动端用户饱和，期待其他新兴领域带来业务增量”。科技行业发展之快，令人欣喜也叫人唏嘘。

不得不说，作为一名计算机专业科班出身的同学，我很惭愧。我其实是在走上工作岗位后，才对作为本科必修课的编译原理有了真正的了解，也真正感受到了这个技术领域独特的魅力。所以现在，我也推荐你关注这项技术。

为什么我会推荐你学习编译技术呢？如果你一定要问我理由，我想给你分享两点。

第一点，网红热点的“花开花落”，远不如底层技术的“静水流深”有力量。

在我还是一名“产品工程师”（Product Engineer）的时候，我从事过客户端、前端、服务端，以及一些面向业务应用的算法方面的工作。

近些年，每一个技术领域都涌现过各种新概念、新趋势，这可能会让你陷入“学习了这门新技术，就能分享这种技术形态的成长红利，提升自我价值”的错觉。我也曾追赶过这些“技术时尚”：做后端时一会儿学学这个语言，一会儿瞧瞧那个中间件；做前端时，一会儿用用这个框架，一会儿看看那个三方库。

但冷静下来，你就会发现，这些“网红型”技术热点，能够沉淀下来、为业务长期赋能的寥寥无几。真正值得长期学习和不断实践的，反倒是计算机专业的那些基础课，包括但不限于数据结构与算法、计算机网络、操作系统、编译原理、计算机体系结构，还有软件工程等等。这些课程你在学校里学习的时候，可能感觉枯燥乏味、过于抽象，甚至毫无成就感，然而在工作以后你才会发现，这可能是一名 coder 应对“行业内卷”最坚实有力的后盾。

第二点，技术上卡的“脖子”也正意味着更多的机会和可能。

其实，近两年来，我们国内 IT 行业很多龙头企业都遭遇了“技术卡脖子”的情况。一时间，芯片、操作系统以及其他基础软件国产化的呼声，都起来了。但是，在我看来，这些“low-level-stuff”需要的不止是时间与金钱的投入，更需要技术人才的“梯队化”。

国外的许多大公司里，你都能见到 50 多岁，甚至年龄更大的资深程序员，他们往往不是从事一些面向用户的业务逻辑开发，而是做一些被称为 Infrastructure 的基础架构工作，我们国内这样的趋势还不够明朗，毕竟我们起步太晚，做这些基础技术的工作，又需要公司雄厚且稳定的营收去支撑。这个差距是显而易见的，甚至短期内都没法快速弥补上。

我曾经就遇到过一个 Google V8 引擎方面的技术难题。我当时是在一个 c++ 的程序中集成了 V8，通过 V8 的 ObjectTemplate 和 HandleScope 等优化尽可能快速生成 jsObject，并传递到 V8-Isolate 内部的 jsContext 里。但我用尽了所有方法，依然达不到原生 JavaScript 中字面量 Object 的性能。我跟身边许多架构师同事进行了探讨，也通过 e-mail 和 v8 项目的一些参与者进行了交流。但是我发现，相比起国外的技术社区，国内工程师们能给出的一些建议确实非常有限。

当然，我不认为这是技术水平、或者是智力这些因素造成的，事实上，国内有大量非常聪明，而且比国外更加勤劳（内卷所致）的软件工程师。在我看来，造成这个现象的原因，主要是 V8 项目的历史实在太久远了，而且早期的核心开发人员又有许多来自于历史更久远的 Java 虚拟机。整个过程具有很强的“技术继承性”，国内的工程师很难有机会，真正深入地了解这些系统的技术内幕。

或许，这就是我们在很多核心技术领域被“卡脖子”现象的主要原因之一。不过，困境也就意味者突破和机会。在底层软硬件国产化的浪潮下，编译器和操作系统是两座绕不过、躲不开的大山，国产芯片也会创造出大量让编译技术大放异彩的机会。在可预见的未来 5～10 年内，国内的这个技术趋势都是存在的。所以，我看好编译技术、操作系统等这些技术领域的发展，也推荐你深入学习这些底层技术。

起步篇讲了什么？

前面聊完了“为什么”，现在我们就来解析一下“是什么”，聊聊我们这门课已更新完的起步篇里，都讲了什么。

不过，在这里我希望你能理解一点，当前这门课程是宫老师讲编译技术的第三季课程了，所以不可避免地存在着一定的知识继承性和延续性。对于编译器的前端部分，这次的课程

中讲得相对没有那么深入。如果你想深入了解前端的知识，我建议去看第一季《编译原理之美》和第二季《编译原理实战课》。第三季的重点是放在了编译器后端部分，和物理机、操作系统打交道。

如果你看到我梳理的概念中，有很多是你无法理解的，你可以带着疑问，试着把整个流程串起来。有了一个整体的“大局观”之后，再回头去第一季和第二季中找寻答案，当然也可以在我们的微信交流群里提问。

编译器是一个工业级的基础软件，因此从理论体系上我们就将编译器分成了前端、中端和后端三部分（有些文献上也把中段算为后端）。

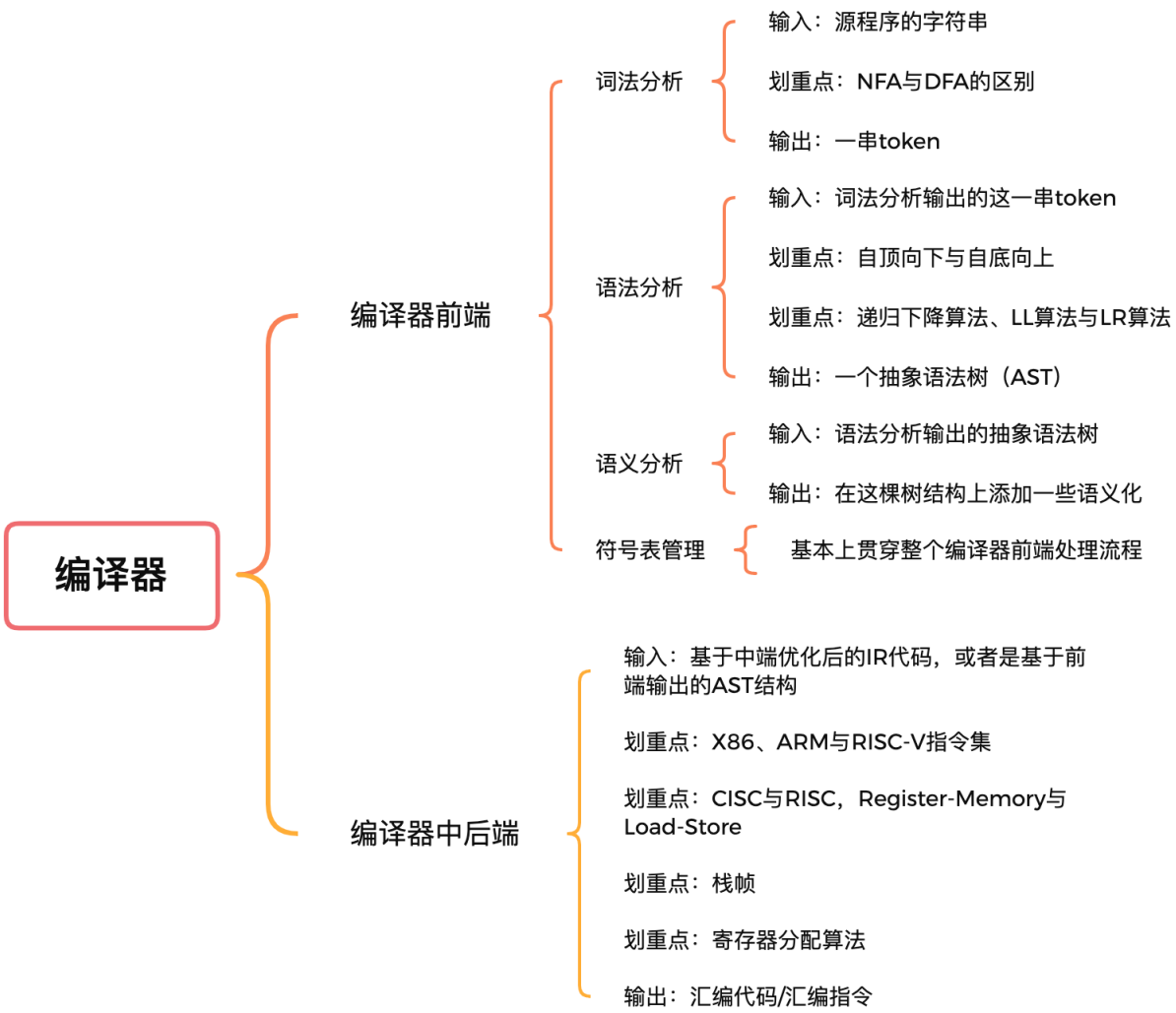
你在学习中也会发现，宫老师起步篇的安排，也是按照这个顺序：02 是讲词法分析，03 和 04 的前半部分是讲语法分析，后面的 04 到 06 的部分则是循序渐进地把编译器前端的语义分析和语法分析的功能，拆成一个有一个具体的 feature 一点一点放到我们的示范程序中来完成、实现。

07 到 11 节部分呢，是有关虚拟机的话题。严格意义上，其实虚拟机相关的技术并不算是传统的编译原理范畴。在编译技术的三大圣经（《龙书》、《虎书》和《鲸书》）里有关虚拟机、垃圾回收等方面的篇幅少之又少。不过这也是因为历史的局限性，毕竟 Java 这样的语言在 1995 年才诞生。不过我们的课程却是与时俱进的，在读到宫老师的这部分课程时候，让我眼前一亮。

接下来的 11-12 两节课呢，是一些基础知识的铺垫。这里涉及到编译器与操作系统、和计算机硬件之间“打交道”时的一些“责任边界”。后面的 14-18 节，则是对编译器后端技术的实践。由于我们课程的受众大部分是软件工程师，所以在 14 和 15 两节课，宫老师又花了不少篇幅为大家科普芯片指令集的一些基础知识。

在我看来，理解芯片和汇编语言有一个很好的方法，就是把芯片看成我们中学时期用过的“科学计算器”，甚至是更简单的“日常使用的普通计算器”。唯一的不同之处就是，芯片没有给人类手指去触摸的按钮，取而代之的是需要用程序通过一组组汇编代码去操纵这个“超级微型却功能强大的计算器”。希望我这样的描述，能减少你对芯片指令集的陌生感、缓解你对“超纲知识”的恐惧。

起步篇最后的 19-21 这三节课呢，是对一些难点知识的精讲。如果你的基础不牢固，我建议你优先学习前面的知识内容。除此之外，我还为你整理了一张脑图，帮你“高亮”出了一些学习这门课有必要弄懂的关键概念：



动手实践才是目的

理清了我们起步篇的内容，最后我们聊聊在日常工作中，我们可以从哪些方向把这门课学到的知识落到实践中来。我想从我从事过的前端、客户端、服务端和算法这四个软件工程师岗位，给你讲讲我是如何在日常工作中实践编译技术的，希望能对你有一些参考价值。

平心而论，我非常建议你，把课程作为自己学习的一个起点而不是终点。只有你真正实践过，你才能真正明白为什么大部分语言的前端都在依靠手写递归下降 + 算符优先级算法的组合去实现，而鲜有教科书上那样设计精巧的 LL 算法实现（因为 first 和 follow 集的维护成本太高了！）。

领域一：前端

你可能会问，前端领域真的有必要，学习编译原理这样的技术吗？我理解，毕竟很多前端的同学，每天的工作就是机械地进行设计稿（PS、Sketch 等生成的文件）到 HTML+ CSS 代码的转换。但你可能忽略了一些我们前端每天都在使用的构建工具，比如 webpack、rollup，或者是近两年涌现出的 esbuild、swc 等等，这些恰恰是我在从事前端工作期间，认为最有意思的一些 infra 类的工作。

而且，我们这门课也使用了 JavaScript/TypeScript 语言作为教学工具。如果你就从事前端，那我非常建议你把上面这些构建工具作为自己的研究目标，像电影《速度与激情》里的剧情一样，把自己每天开的“车子”拿过来拆开看看，动手改装“魔改”一番，这会是一件非常有乐趣和成就感的事。

另外，从 Typescript 到 Wasm 这些新工具、新技术的出现也能看出，前端是最有可能在近几年内，因为编译技术而出现新变革的技术领。如何设计出一种在开发阶段可以使用 JavaScript 技术栈、而在运行时又能提供尽可能像 C++ 一样高性能的编程语言工具链，将成为业界的一个关键课题。

领域二：客户端

准确地说，客户端开发这并不是一个单一的技术领域。但在今天这个时间点，它在语义上至少可以包含移动端上的 Android（Java-JVM 技术栈）和 iOS（OC/Swift 技术栈）两个研发岗位，涉及桌面 PC 端的话通常意味着 Windows 平台的开发（C#、C++ 技术栈）。

谈到这里，我想起了我在本科期间非常热衷研究的逆向工程技术（这是信息安全的一个细分领域），就是依靠对已经编译好、打包的客户端程序，通过反解、静态分析和动态调试等一系列技术手段，获悉到其源程序代码中的部分甚至全部逻辑，从而达到自己的目的。

客户端开发应该是逆向工程这门技术最主要的攻防战场，这个领域很能满足你对“黑客”这种角色的向往。但今天我回头再看，当年自己用过的像 IDA Pro 这样的神器，其底层实现原理就完全离不开编译技术，包括做动态调试的话，其实很多时候需要对底层汇编、IR 或者 Smali 这些贴近硬件的语言有足够的了解，这也是我们课程中所涉及到的知识。

另外，近几年，移动端对代码热更新、动态化方面的探索也涌现出了很多优秀的开源方案，这也离不开编译技术。你也完全可以自行设计一套自行编译、动态下发、解释执行甚至 AOT 的动态化技术方案，这是一件非常锻炼人能力的事情。我还能想到，“端智能”也是一个可以让编译技术大展身手的场景，不过这个话题我们放在下面的算法岗位部分再谈。

领域三：服务端

在我的印象里，后端工程师在一个研发团队中非常像足球场上的中场球员，他对技术全面性的要求最高，同时业务线上的服务端工程师也是最辛苦的一个岗位（有过 oncall 期间夜里 12 点刚完成代码的全量发布上线，又在凌晨三四点被服务器短信报警叫醒的经历之后，你会深刻体会到这一点）。

你可能也有体会，因为我们和业务数据离得近，总会有各种临时性的“需求”不仅会打断我们原本的工作计划，而且经常是无法计入研发排期的“脏活累活”，费力不讨好。我就曾经经历过这样一段时期，团队缺人，经常还有其他非技术岗同学排着队来我这里提这些临时性的统计数据需求。

工程师就该从技术的角度去想办法，于是我当时自己设计了一套简易的 DSL，用于描述特定的一些数据维度（比如用户 PV、比如文章分类 Tag），你可以理解为这是一个 SQL 语言的子集，不过不同之处是使用中文的“查询”、“关联”这些名词作为代码的关键词，方便非技术岗的同学。然后我专门开发了一个简单的编译工具，将这个 DSL 转译成集群中跑的 HQL（Hive SQL）。最后，我把这个工具，配合详细的文档，提供给了每天排队给我提需求的其他同事。自此我就完美地解决了这个问题，不知这个 case 是否对你有所启发。

当然了，服务端工程师其实有特别好的技术研究土壤，你可以根据自己团队所使用的语言运行机制，比如 Java、PHP、Golang 等，进行一些针对性的集群改造、甚至对语言编译期、运行时本身进行“魔改”。最初往往都是从研读语言本身编译器、解释器的源码开始，逐步输出一些对业务代码写法上的优化改造开始的，逐步才会过度到开发一些 C 扩展、甚至开始魔改解释器本身，以上都是很多后端 Infra 团队经历过的过程。这些都是实实在在能够产生业务价值的技术研究性课题，相信这也会给你带来非凡的成就感和价值感。

领域四：算法

算法岗位通常意味着，依靠 AI 算法来解决业务场景中的实际问题。在今天的互联网公司里，最常见的算法岗就是检索、推荐和广告这三个业务场景。

算法岗位在技术探索的角度上是我们软件工程师的“星辰大海”，上至数学理论、下至底层硬件技术，都可能为这个领域涉足，而编译技术在算法这一技术领域同样有它的广阔应用前景。在许多人看来，未来的自动驾驶、智能家居、智能穿戴等诸多新的业务场景，将带来移动端时代成百上千倍的数据规模。更大的数据规模当然意味着更大的训练样本，而基于现有的芯片硬件架构提供的算力，显然是吃不消的。

现阶段的深度学习往往是使用 GPU 进行大量矩阵运算，而各个公司也都在进行新一代 AI 专有芯片的研究探索，包括 NPU、XPU 等等。这也包括我们前文中提到的“端智能”场景，也就是出于性能和隐私安全等诸多原因，把一些 AI 相关的计算逻辑搬到手机本地的一块芯片上。

在未来，如何更好地将上游的数学计算问题转化为底层芯片上的逻辑运算，这将会是算法岗位偏工程团队关注的一大课题。如果你对此方向有兴趣，我建议你考虑加入你所在的公司的算法工程架构团队，比如现在很多企业都在用的 Tensorflow、PyTorch 等框架的服务端集群进行运维工作一般就是交由这样的团队来处理的，他们通常以 Python 和 C++ 作为主要的技术栈。

不过，算法领域其实是一个非常大、并且涉及到很多不同专业背景的工程师共同协作完成的一个技术领域，有很多其他计算机基础技术理论也都可以在这个岗位上进行实践，这里就不一一列举了。

最后，我想说，每当我们抱怨自己深陷繁重而又重复的业务开发工作中，无暇去实践很多有意思的技术问题的时候，其实机会往往就在我们每天工作的日常当中。我希望你能动手实践起来，把这些硬知识都结合到工作中来。

上一篇　期中测试｜快来检验你在起步篇的学习成果吧

下一篇　“屠龙之秘”｜实现计算机语言这样的技术能用在哪里？（一）

精选留言

写留言

> 由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。