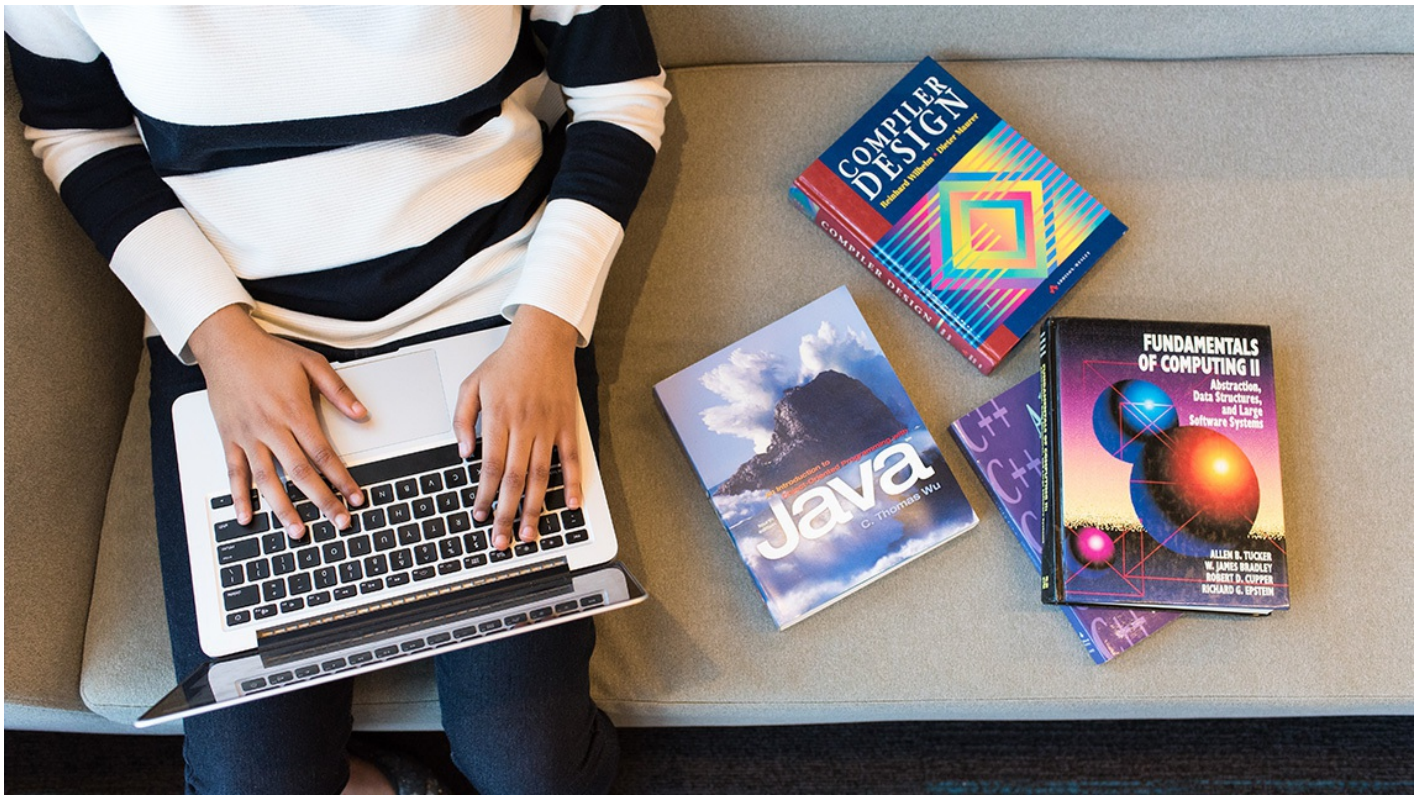
31讲程序员怎么学习运维知识



在上一讲中，我们讲到了开发过程的自动化，我们的关注点在于如何构建出一个有效的部署包，这个包最终是要上线部署的，那接下来，我们就来关心一下部署的相关工作。

零散的运维知识

在一些稍具规模的公司，为部署工作设置了一个专有职位，称之为运维。当然，这个岗位的职责远不止部署这一件事，还要维护线上系统的稳定。不过，如果你的团队规模不大，或是项目处于初始阶段，这些工作往往也要由程序员自行完成。

对于一个程序员来说，了解自己的程序怎么部署上线，是非常重要的。我们既要了解一个软件的逻辑，也要知道它的物理部署。只有这样，出了问题才知道怎么修复。

更重要的是，我们在设计时，才能尽量规避部署带来的问题。而部署，恰恰也是最适合发挥自动化本领的地方。

好，即便下定决心准备学习运维相关知识，你准备怎么学呢？我先来问你个问题，提到运维，你会想到什么？

如果你是一个刚刚步入这个行业的程序员，你或许会想到 Docker，想到 Kubernetes；如果再早一点入行，你或许还会想到 Chef、Puppet、Ansible；更早一些入行的话，你会想到 Shell 脚本。没错，这些东西都是与运维相关的。那我就这么一个一个地都学一遍吗？

就我个人的学习经验而言，如果所有的知识都是零散的，没有一个体系将它们贯穿起来，你原有的知识无法帮助你学习新知识，这种学习方式效率极低，过程也极其痛苦。

如果是有结构的知识，所谓的学习新知识不过是在学习增量，真正要理解的新东西并不多，学习效率自然会大幅度提高。所以，想学好运维知识，首先你要建立起一个有效的知识体系。

你可能会问，这些运维知识看上去就是一个一个独立的工具啊？我曾经也为此困惑了许久，虽然我对各个工具已经有了不少的了解，但依然缺乏一个有效的知识体系，将它们贯穿起来，直到我上了一堂课。

感谢 Odd-e 的柴锋，有一次，他给我上了一堂 DevOps 课，他对运维知识的讲解让我茅塞顿开，从此，我的运维知识有了体系。

准确地说，他的这堂课就是讲给程序员的运维课。今天，我就把这个体系按照我的理解，重新整理一遍分享给你，也算是完成一次知识输出。

好，我们开始！

Java 知识体系

正如我前面所说，学习一个新东西，最好的办法是学习增量，如果能够找到它与已有知识体系的联系，我们就可以把已有知识的理解方式借鉴过去。

作为程序员，我们其实已经有了一个完善的知识体系，这就是我们对于程序设计的理解，而理解运维的知识体系，刚好可以借鉴这个体系。怎么理解这句话呢?

以最常见的 Java 开发为例，如果要成为一个合格的 Java 程序员，我应该知道些什么呢?

首先肯定是 Java 语言，我需要了解 Java 语言的各种语法特性。不过，只了解语法是写不出什么像样程序的，我们还需要掌握核心库。

对于 Java 来说，就是 JDK 中的各种类，比如，最常见的 String、List、Map 等等。

理论上来说，掌握了基本的语法和核心库，你就可以开发任何程序了。但在实践中，为了避免重新发明“轮子”，减少不必要的工作量，我们还会用到大量的第三方类库，比如，Google Guava、SLF4J 等等。

除了功能实现，还有一些结构性的代码也会反复出现。比如说，在常见的 REST 服务中，我们要将数据库表和对象映射到一起，要将结果转换成 JSON，要将系统各个组件组装到一起。

为了减少结构上的代码重复，于是，开发框架出现了，在 Java 中最常见的开发框架就是 Spring。

至此，你就可以完成基本的代码编写，但这还不够。

在 Java 中，你不会从底层完成所有事情，比如，虽然你写 REST 服务，但你很少会接触到最底层的 HTTP 实现，因为这些工作由运行时环境承担了。

我们要做的只是把打好的包部署到这些运行时环境上，在 Java 的世界里，这是 Tomcat、Jetty 之类的容器承担的职责。

如果你刚刚加入这一行，上来就用 Spring Boot 之类的框架写代码，你可能并没有碰到这样的部署过程，因为这些框架已经把容器封装其中，简化了部署过程。

Tomcat、Jetty 往往还只是在一台机器上部署，在现实的场景中，一台机器通常是不够用的，我们可能需要的是一个集群。

你可能会想到用 Nginx 来做一个负载均衡，但如果用原生的 Java 解决方案，这时候就轮到企业级的应用服务器登场了，比如：IBM WebSphere、Oracle WebLogic Server、JBoss Enterprise Application Platform 等等。

至此，一套完整的 Java 应用解决方案已经部署起来了。但我们知道了这些，和我们运维知识有什么关系呢？我们可以用同样的体系去理解运维知识。

运维知识体系

首先，要理解运维体系的语言。运维的语言是什么呢？是 Shell，人们最熟悉的应该是 Bash。我们通过操作系统与计算机打交道，但我们无法直接只用操作系统内核，Shell 为我们提供了一个接口，让我们可以访问操作系统内核提供的服务。

你可能会以为我这里用的是比喻，将 Shell 比喻成语言，但还真不是，Shell 本身就是一门编程语言。绝大多数人都知道 Shell 可以编程，但几乎没有人把 Shell 当成一门编程语言来学习，基本上都是在需要的时候，搜一下，然后照猫画虎地将代码复制上去。

这样造成的结果就是，一旦写一个脚本，就要花费大量的时间与语法做斗争，只是为了它能够运行起来。

有了语言，再来就是核心库了。运维的核心库是什么？就是 Shell 提供的各种 Unix/Linux 的核心命令，比如：ls、cd、ps、grep、kill、cut、sort、uniq 等等，它们几乎与操作系统绑定在一起，随着操作系统一起发布。

了解了核心的部分，还需要了解一些第三方库，运维知识的第三方库就是那些不属于操作系统核心命令的命令，比如：rsync、curl 等等。

Java 有框架可用，运维也有框架吗？你可以想一下，Java 的框架提供的是一些通用的能力，在运维工作中，也是有一些通用能力的，比如：在安装某个包之前，要检查一下这个包是否已经安装了；在启动一个服务前，要检查这个服务是否启动了，等等。所以，能够帮我们把这些工作做好的工具，就是我们的运维框架。

到这里，你应该已经明白了，我在说的运维框架其实就是像 Chef、Puppet、Ansible 之类的配置管理工具。它们做的事就是把那些繁琐的工作按照我们的定义帮我们做好。

有了对软件环境的基本配置，接下来，就要找一个运行时的环境将软件跑起来了。这时候，我们要了解像虚拟机、Docker 之类的技术，它们帮我们解决的问题就是在单机上的部署。

一般来说，了解了这些内容，我们就可以构建出一个开发环境或测试环境。除非用户非常少，我们可以在生产环境考虑单机部署，否则，我们迄今为止讨论的各种技术还都是在开发环节的。

如果我们需要一个集群或是高可用环境，我们还需要进一步了解其他技术，这时候，就轮到一些更复杂的技术登场了，比如，云技术，Amazon AWS、OpenStack，包括国内的阿里云。如果你采用的是 Docker 这样的基础技术，就需要 Kubernetes、Docker Swarm 之类的技术。

至此，一个相对完整的运维知识体系已经建立起来了，现在你有了一张知识地图，走在运维大陆上，应该不会轻易地迷失了。希望你可以拿着它，继续不断地开疆拓土。

总结时刻

我们今天的关注点在于，将开发过程产生的构建产物部署起来。部署过程要依赖于运维知识，每个程序员都应该学习运维知识，保证我们对软件的运行有更清楚地认识，而且部署工作是非常适合自动化的。

但是，对运维工具的学习是非常困难的，因为我们遇到的很多工具是非常零散的，缺乏体系。

这里，我给你介绍了一个运维的知识体系，这个体系借鉴自 Java 的知识体系，包括了编程语言、核心库、第三方库、开发框架、单机部署和集群部署等诸多方面。我把今天提到的各种技术整理成一个表格列在下面，你可以参考它更好地理解运维知识。

类别	Java	运维
语言	Java 语言	Shell
核心库	JDK	Unix/Linux 核心命令
第三方库	第三方程序库，比如：Google Guava、SLF4J	第三方命令，比如：rsync、curl 等
开发框架	开发框架，比如：Spring	配置管理工具，比如：Chef、Puppet、Ansible 等
单机部署	应用服务器，比如：Tomcat、Jetty	部署环境，比如：虚拟机、Docker 等
集群部署	企业级应用服务器，比如：IBM WebSphere、Oracle WebLogic Server、JBoss Enterprise Application Platform	云服务，比如：Amazon AWS、OpenStack、阿里云等；Docker 集群，比如：Kubernates、Docker Swarm 等

如果今天的内容你只能记住一件事，那请记住：**有体系地学习运维知识。**

最后，我想请你分享一下，你还能想到哪些运维知识可以放到这张知识地图上呢？欢迎在留言区写下你的想法。

感谢阅读，如果你觉得这篇文章对你有帮助的话，也欢迎把它分享给你的朋友。

精选留言

hua168

现在运维流行DevOps，高级一点就是AI，
其中一篇文章《DevOps 详解》不错，链接如下：
https://infoq.cn/article/detail-analysis-of-devops
《DevOps知识体系与标准化的构建》也不错，下载地址：
https://yq.aliyun.com/download/778
运维知识体系：
https://www.unixhot.com/page/ops
Web缓存知识体系：
https://www.unixhot.com/page/cache

运维技能：
1.懂网络：
一般要求CCNA（最好CCNP）或同等水平

2.懂系统：
懂得主流的linux系统操作（Centos、ubuntu、debian等）
操作命令、维护、性能优化、故障排查

3.简单安全：
一些简单的安全知识

4.半个DBA：
一般中小公司前期没有DBA，需要运维做
最起码会SQL语句、主从

群集：redis、mysql、MongoDB等

5.会运维开发：
一般用于开发运维工具、运维系统（如CMDB、ELK日志系统等）
运维主要语言是shell、python/Go
python web框架：Django、tonado等
Go web框架：Beego、Gin、Iris等
有的还会用PHP及框架（TP、YII、Laravel做web前端）
中小公司运维一般都没有专职的前端，需要运维兼职所以要学前端知识

6.懂点开发：
般都懂一点本公司开发的语言，如公司用PHP需要学习、如公司用java web也需要学习一下，目标：
1）更好的维护网站，排错
2）运维自动化、DevOps，因DevOps是基于敏捷开发，极限编程的思想，所以得懂一点软件工程

7.主职：
1）各种环境的搭建：LAMP、LNMP、负载均衡(nginx、haproxy、VLS等)、web群集、数据库群集、主流的docker[必会]
2）排错[必会]
3）批量安装系统安装：Cobbler[少]
4）部署工具：Ansible/SaltStack[重要]
5）主流的部署方案：如云、docker、k8s等[必会]
6）监控系统：zabbix、Open-Falcon[至少掌握其中一种]
7）自动化：gitlab CI/CD、jenkins结合ansible/salt、docker[必会]
8）运维流程的制定
9）减少背锅的次数：运维是出名的“背锅侠”，制定明确的责任可以减少背锅
10）等等

8.会点构架
一般中小公司没有构架师，所以当业务增大出现瓶颈，运维得给出解决方案
和开发讨论如何扩展

总结：
在中小公司运维工作就是一件很杂的工作，什么都要求会一点
2019-03-22 12:05

作者回复

多谢补充！
2019-03-22 20:15

hua168
你们开发做兼职做运维了，那我们运维出路怎么办
运维又怎么学开发?
2019-03-22 10:57

作者回复

DevOps，马上就来！
2019-03-22 20:17

小白菜
如何系统学习呢？就是按照上面的表格类比，类比思维？
2019-03-22 10:23

作者回复

一层一层地了解，至少要了解基本的用法。还有一种极致的方式，联系柴锋老师去上他的课。

2019-03-22 20:21

enjoylearning

运维对网络和操作系统需要了解多一些，深一些

2019-03-23 19:11

Ankhetsin

vagrant，webpack，npm，jcenter，mavencenter，apt，brew，cocoapod，rip，pip，composer算运维吗?

2019-03-23 07:49

we

网络大类

2019-03-22 08:59

小伟

实际情况里，开发框架到单机部署还有很多步骤，如代码自动化编译、提交、环境版本管理、发布策略管理等。常用的工具依次是maven、github、jenkins。

2019-03-22 08:34

作者回复

下一讲就是持续交付、DevOps 登场了。

2019-03-22 20:22

Y024

DevOps 就是让 Eating your own dog food 来得再猛烈些，发布也得自己感受下。

2019-03-22 00:40

作者回复

做得越多，越全面!

2019-03-22 20:27

西西弗与卡夫卡

还有性能分析和调优，比如Java有JVM内存模型和运行参数调优，运维有单CPU多CPU的硬件模型和操作系统参数优化

2019-03-22 00:34

作者回复

这个类比很不错!

2019-03-22 09:13