81 | 开源实战三（上）：借Google Guava学习发现和开发通用功能模块

2020-05-08 王争

讲述：冯永吉
时长 11:36 大小 10.64M

上几节课，我们拿 Unix 这个超级大型开源软件的开发作为引子，从代码设计编写和研发管理两个角度，讲了如何应对大型复杂项目的开发。接下来，我们再讲一下 Google 开源的 Java 开发库 Google Guava。

Google Guava 是一个非常成功、非常受欢迎的开源项目。它在 GitHub 上由近 3.7 万的 stars。在 Java 项目开发中应用很广泛。当然，我们并不会讲解其中的每个类、接口如何使用，而是重点讲解其背后蕴含的设计思想、使用的设计模式。内容比较多，我分三节课来讲解。

第一节课，我们对 Google Guava 做一个简单介绍，并借此讲一下如何开发一个通用的功能模块。

第二节课，我们讲 Google Guava 中用到的几种设计模式，会补充讲解之前没有讲到的 Immutable 模式。

第三节课，我们借 Google Guava 补充讲解三大编程范式中的最后一个：函数式编程。

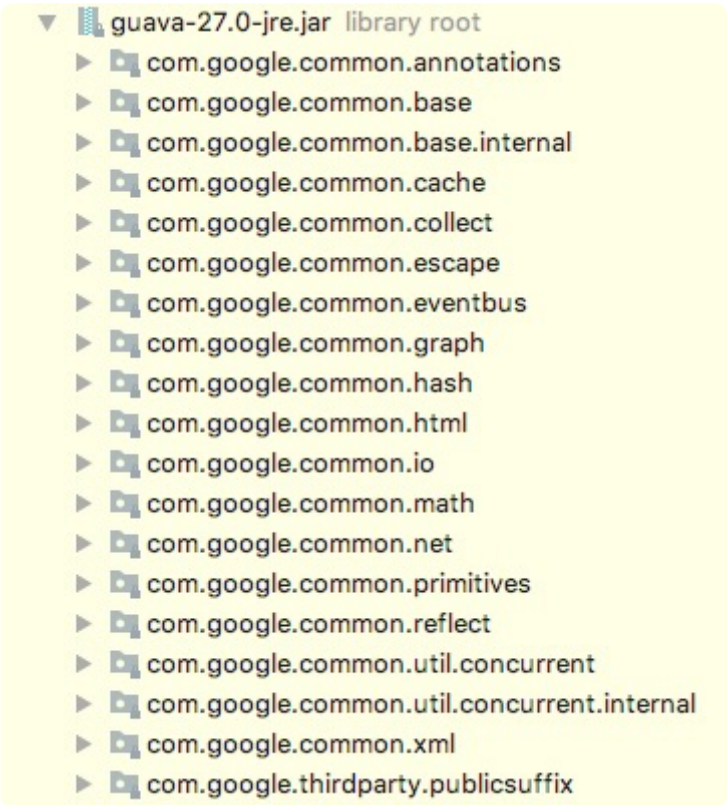
话不多说，让我们正式开始今天的学习吧！

Google Guava 介绍

考虑到你可能不熟悉 Google Guava，我先对它做下简单的介绍。

Google Guava 是 Google 公司内部 Java 开发工具库的开源版本。Google 内部的很多 Java 项目都在使用它。它提供了一些 JDK 没有提供的功能，以及对 JDK 已有功能的增强功能。其中就包括：集合（Collections）、缓存（Caching）、原生类型支持（Primitives Support）、并发库（Concurrency Libraries）、通用注解（Common Annotation）、字符串处理（Strings Processing）、数学计算（Math）、I/O、事件总线（EventBus）等等。

我截取了 Google Guava 的包结构图，贴到了这里，你看起来更加直观些。



我们知道，JDK 的全称是 Java Development Kit。它本身就是 Java 提供的工具类库。那现在请你思考一下，既然有了 JDK，为什么 Google 还要开发一套新的类库 Google Guava？是否是重复早轮子？两者的差异化在哪里？

带着这个问题，结合 Google Guava，我们来学习如何在业务开发中，发现通用的功能模块，以及如何将它开发成类库、框架或者功能组件。等到学习完之后，我希望你能自己回答这个问题。

如何发现通用的功能模块？

很多人觉得做业务开发没有挑战，实际上，做业务开发也会涉及很多非业务功能的开发，比如我们前面讲到的 ID 生成器、性能计数器、EventBus、DI 容器，以及后面会讲到的限流框架、幂等框架、灰度组件。关键在于，我们要有善于发现、善于抽象的能力，并且具有扎实的设计、开发能力，能够发现这些非业务的、可复用的功能点，并且从业务逻辑中将其解耦抽象出来，设计并开发成独立的功能模块。

在我看来，在业务开发中，跟业务无关的通用功能模块，常见的一般有三类：类库（library）、框架（framework）、功能组件（component）等。

其中，Google Guava 属于类库，提供一组 API 接口。EventBus、DI 容器属于框架，提供骨架代码，能让业务开发人员聚焦在业务开发部分，在预留的扩展点里填充业务代码。ID 生成器、性能计数器属于功能组件，提供一组具有某一特殊功能的 API 接口，有点类似类库，但更加聚焦和重量级，比如，ID 生成器有可能会依赖 Redis 等外部系统，不像类库那么简单。

前面提到的限流、幂等、灰度，到底是属于框架还是功能组件，我们要视具体情况而定。如果业务代码嵌套在它们里面开发，那就可以称它们为框架。如果它们只是开放 API 接口，供业务系统调用，那就可以称它们为组件。不过，叫什么没有太大关系，不必太深究概念。

那我们如何发现项目中的这些通用的功能模块呢？

实际上，不管是类库、框架还是功能组件，这些通用功能模块有两个最大的特点：复用和业务无关。Google Guava 就是一个典型的例子。

如果没有复用场景，那也就没有了抽离出来，设计成独立模块的必要了。如果与业务有关又可复用，大部分情况下会设计成独立的系统（比如微服务），而不是类库、框架或功能组件。所以，如果你负责开发的代码，与业务无关并且可能会被复用，那你就可以考虑将它独立出来，开发成类库、框架、功能组件等通用功能模块。

稍微补充一下，我们这里讲的是，在业务开发中，如何发现通用的功能模块。除了业务开发团队之外，很多公司还有一些基础架构团队、架构开发团队，他们除了开发类库、框架、功能组件之外，也会开发一些通用的系统、中间件，比如，Google MapReduce、Kafka 消息中间件、监控系统、分布式调用链追踪系统等。

如何开发通用的功能模块?

当我们发现了通用功能模块的开发需求之后，如何将它设计开发成一个优秀的类库、框架或功能组件呢？今天，我们不讲具体的开发技巧，具体的开发技巧在后面 Spring 开源实战那部分，我们会讲到一些，我今天打算先讲一些更普适的开发思想。我觉得先有了这些，你应该更容易理解后面的内容。

作为通用的类库、框架、功能组件，我们希望开发出来之后，不仅仅是自己项目使用，还能用在其他团队的项目中，甚至可以开源出来供更多人所用，这样才能发挥它更大的价值，构建自己的影响力。

所以，对于这些类库、框架、功能组件的开发，我们不能闭门造车，要把它们当作“产品”来开发。这个产品是一个“技术产品”，我们的目标用户是“程序员”，解决的是他们的“开发痛点”。我们要多换位思考，站在用户的角度上，来想他们到底想要什么样的功能。

对于一个技术产品来说，尽管 Bug 少、性能好等技术指标至关重要，但是否易用、易集成、易插拔、文档是否全面、是否容易上手等，这些产品素质也非常重要，甚至还能起到决定性作用。往往就是这些很容易忽视、不被重视的东西，会决定一个技术产品是否能在众多的同类中脱颖而出。

具体到 Google Guava，它是一个开发类库，目标用户是 Java 开发工程师，解决用户主要痛点是，相对于 JDK，提供更多的工具类，简化代码编写，比如，它提供了用来判断 null 值的 Preconditions 类；Splitter、Joiner、CharMatcher 字符串处理类；Multisets、Multimaps、Tables 等更丰富的 Collections 类等等。

它的优势有这样几点：第一，由 Google 管理、长期维护，经过充分的单元测试，代码质量有保证；第二，可靠、性能好、高度优化，比如 Google Guava 提供的 Immutable Collections 要比 JDK 的 unmodifiableCollection 性能好；第三，全面、完善的文档，容易上手，学习成本低，你可以去看下它的 Github Wiki。

刚刚讲的是“产品意识”，我们再来讲讲“服务意识”。我经常在团队中说，如果你开发的东西是提供给其他团队用的，你一定要有“服务意识”。对于程序员来说，这点可能比“产品意识”更加欠缺。

首先，从心态上，别的团队使用我们开发出来的技术产品，我们要学会感谢。这点很重要。心态不同了，做起事来就会有微妙的不同。其次，除了写代码，我们还要有抽出大量时间答疑、充当客服角色的心理准备。有了这个心理准备，别的团队的人在问你问题的时候，你也就不会很烦了。

相对于业务代码来说，开发这种被多处复用的通用代码，对代码质量的要求更高些，因为这些项目的影响面更大，一旦出现 bug，会牵连很多系统或其他项目。特别是如果你要把项目开源，影响就更大了。所以，这类项目的代码质量一般都很好，开发这类项目对代码能力的锻炼更有大。这也是我经常推荐别人通过阅读著名开源项目代码、参与开源项目来提高技术的原因。

具体到 Google Guava，它是 Google 员工开发的，单元测试很完善，注释写得很规范，代码写得也很好，可以说是学习 Google 开发经验的一手资料，建议你如果有时间的话，可以认真阅读一下它的代码。

尽管开发这些通用功能模块更加锻炼技术，但我们也不要重复造轮子，能复用的尽量复用。而且，在项目中，如果你想把所有的通用功能都开发为独立的类库、框架、功能组件，这就有点大动干戈了，有可能会得不到领导的支持。毕竟从项目中将这部分通用功能独立出来开发，比起作为项目的一部分来开发，会更加耗时。

所以，权衡一下的话，我建议初期先把这些通用的功能作为项目的一部分来开发。不过，在开发的时候，我们做好模块化工作，将它们尽量跟其他模块划清界限，通过接口、扩展点等松耦合的方式跟其他模式交互。等到时机成熟了，我们再将它从项目中剥离出来。因为之前模块化做的好，耦合程度低，剥离出来的成本也就不会很高。

重点回顾

好了，今天的内容到此就讲完了。我们一块来总结回顾一下，你需要重点掌握的内容。

做业务开发也会涉及很多非业务功能的开发。我们要有善于发现、善于抽象的能力，并且具有扎实的设计、开发能力，能够发现这些非业务的、可复用的功能点，并且从业务逻辑中将其解耦抽象出来，设计并开发成独立的功能模块，比如类库、框架、功能组件。

实际上，不管是类库、框架还是功能组件，这些通用功能模块最大的两个特点就是复用和业务无关。如果你开发的这块代码，业务无关并且可能会被复用，那就可以考虑将它独立出来，开发成类库、框架、功能组件等。

当我们发现了通用功能模块的开发需求之后，如何将它设计开发成一个优秀的类库、框架或功能组件呢？这里我们讲了一些更普适的开发思想，比如产品意识、服务意识、代码质量意识、不要重复造轮子等。

除此之外，我特别建议你去阅读一下 Google Guava 的开源代码。它的代码不复杂，很容易读懂，不会有太大阅读负担，但它是你获取 Google 公司开发经验的一手资料，特别是在单元测试、编码规范方面。

课堂讨论

针对你正在参与开发的项目，思考一下，有哪些通用的功能模块可以抽象出来，设计开发成独立的类库、框架、功能组件？它们都可能会包括哪些功能点呢？试着自己设计一下吧！

欢迎留言和我分享你的想法，如果有收获，也欢迎你把这篇文章分享给你的朋友。

上一篇 80 | 开源实战二（下）：从Unix开源开发学习应对大型复杂项目开发

下一篇 82 | 开源实战三（中）：剖析Google Guava中用到的几种设计模式

精选留言 (8)

写留言

Monday
2020-05-10

google guava， I'm coming!

展开

1

Jackey
2020-05-08

let’ s go guava

展开

1

makermade
2020-05-08

分布式id生成器，可以抽象成一个独立的服务

展开

1

呦呦鹿鸣
2020-05-08

沙发！开始翻guava源码

展开

1

Frank
2020-05-10

如何开发通用模块是不是可以对应到专栏之前讲过的“如何做需求分析与设计“中的非业务系统的开发那部分？

展开

javaadu
2020-05-08

我的项目中，有一个功能模块是做数据同步的：将db数据同步到运行时的内存或缓存中。我考虑，这个功能可以作为一个通用的框架来维护，主要功能包括：接入模块、数据同步模块、数据同步健康检查模块、数据主动同步模块

展开

Heaven
2020-05-08

现在正在编写一个项目的告警模块,支持用户的自定义规则,进行解析后,多次到达一定额度,进行触发报警事件,这就可以抽取出来,但是现在越开发越和现有的业务代码耦合度深,所以在考虑将最开始的基本版本进行完善,形成一个框架暴露出去
顺便一提,其实大家可以根据自己的项目,抽取出一个脚手架,做到尽可能的通用,作为一个框架暴露出去

展开

1

wakaka
2020-05-08

平时用的比较多，用起来很方便，期待后面的两章，再对比源码看看。