加餐（三）| Kaito：我希望成为在压力中成长的人

2020-09-25 Kaito

Redis核心技术与实战 进入课程 >

讲述：蔡正兵
时长 11:22 大小 10.43M

你好，我是 Kaito。

上一次，我分享了我总结的 Redis 学习路径，在留言区的交流和互动中，我有了很多新的收获。今天，我想再分享一下我对学习这件事儿的认识以及我的学习方法，包括领先一步的心理建设、事半功倍的学习方法以及提升效率的小技巧。

领先一步：保持好奇 + 不设限

我认为，任何领域的学习，在研究具体的方法之前，我们都需要先在心理上领先别人一步。什么意思呢？其实就是要建立并保持好奇心，并且不给自己设限。

我发现，很多人是缺乏好奇心的，突出表现在只知其然，不知其所以然，不善于思考和挖掘问题。

给你举个小例子。刚开始接触 Redis 时，你肯定听说过一句话，**Redis 是单线程，高性能**。很多人听完也就过去了，但是有好奇心的人，会进一步思考：“单线程如何处理多个客户端的网络请求呢？采用单线程的话，只能用到一个 CPU 核心，怎么达到高性能呢？”

顺着这个思路去学习的话，你就会发现，Redis 虽然采用了单线程，但是它使用了多路复用技术，可以处理多个客户端的网络请求。而且，它的数据都存储在内存中，再加上高效的数据结构，所以处理每个请求的速度极快。

你看，带着好奇心去看问题，最终我们得到的远远超出想象。所以，我们要**永远保持好奇心和深入探究的精神**，它是我们不断进步的核心驱动力。

我要说的第二点，就是**不要给自己设限**。

不要没有做任何尝试，就先去说“我做不到”。如果你这样做，就相当于提前放弃了自己的成长机会。我特别喜欢的一个心态是：“**我现在虽然不会，但是只要给我时间，我就能学会它。**”

说到这儿，我想给你分享一个我的小故事。

之前我在业务部门做开发时，大部分时间都在写业务代码，对 Redis 也只停留在“会用”的层面，并不了解它的原理，更别说分析和定位性能问题了。

后来一个偶然的机会，我可以去公司的基础架构部门做数据库中间件相关的工作。我当时非常犹豫：一方面，我知道，这个工作要求熟练掌握 Redis 的方方面面，难度非常高，我觉得我可能无法胜任；但另一方面，我也非常想踏出舒适区，突破一下自己。最终，我还是选择了接受挑战。

刚开始时，我确实遭遇了难以想象的困难，比如说不熟悉 Redis 的运行原理、看 Redis 源码时一头雾水、在系统发生问题时不知所措等等。还好，面对压力，我的斗志被激发了，于是就疯狂地恶补数据结构、网络协议、操作系统等知识，一行行地去啃源码……

真正走出舒适区之后，我看到了自己的飞速成长和进步，不仅很快就胜任了新工作，而且，我越来越自信了。之后，每次遇到新问题的时候，我再也不会害怕了，因为我知道，只要花时间去研究，就可以搞定一切。

所以，我真的想和你说，面对任何可以让自己成长的机会，都不要轻易错过，一定不要给自己设限。你要相信，你的潜能会随着你面临的压力而被激发出来，而且它的威力巨大！

事半功倍：行之有效的学习方法

有了强烈的学习意愿还不够，我们还要快速地找到科学有效的学习方法，这样才能事半功倍。接下来，我就聊聊我的学习方法。

首先，我们要学会快速地搜集自己需要的资料。在搜索的时候，我们要尽量简化检索的内容，避免无用的关键词，例如，如果想要搜索“Redis 哨兵集群在选举时是如何达成共识的”这个问题，我一般会搜索“Redis sentinel raft”，这样只搜索重点词汇，得到的结果会更多，也更符合我们想要的结果。

如果在查资料时，遇到了细节问题，找不到答案，不要犹豫，**一定要去看源码**。源码是客观的，是最细节的表现，**不要只会从别人那里获取东西，要学着自己动手觅食**，而源码，往往能够给我们提供清晰易懂的答案。

比如说，Redis 的 String 数据类型底层是由 SDS 实现的，当要修改的 value 长度发生变更时，如果原来的内存空间不足以存储新内容，SDS 就需要重新申请内存空间，进行扩容，那么，每次扩容时，会申请多大的内存呢？

如果你看到了`sds.c`中的 sdsMakeRoomFor 函数，就会知道，当需要申请的内存空间小于 1MB 时，SDS 会申请 1 倍的内存空间，这样就可以避免后面频繁申请内存而导致的性能开销；当需要申请的内存空间大于 1MB 时，为了避免内存浪费，每次扩容时就只申请 1MB 的内存空间。类似于这样的问题，我们都能很快地从源码中找到答案。

很多人都觉得看源码很难，不愿意走出这一步，刚开始我也是这样的，但是后来有一天，我突然想到了“二八定律”。我所理解的“二八定律”，就是 80% 的人甘于平庸，遇到稍微难一点的问题就会停下脚步；而另外 20% 的人，一直不愿意停留在舒适区，只要确定了目标，就会一直向前走。我当然希望自己是那 20% 的人。所以，每次我觉得有压力、有难

度的时候，我就会告诉自己，得坚持下去，这样才能超越 80% 的人。不得不说，这招儿还挺有用的。

另外，我还想说，掌握新知识最好的方式，就是把它讲给别人听，或者是写成文章。

尤其是在写文章的时候，我们需要确定文章的结构，梳理知识点的脉络，还要组织语言，在这个过程中，我们会把一些零碎的内容转化为体系化、结构化的知识。那些散乱的点，会形成一棵“知识树”，这不仅方便我们记忆，而且，在复习的时候，只需要找到“树干”，就能延伸到“枝叶”，举一反三。

而且，在梳理的过程中，我们往往还能发现自己的知识漏洞，或者是对某些内容有了新的认识和见解。

例如，我在写《Redis 如何持久化数据》这篇文章的时候，就已经知道了 RDB+AOF 两种方式，但在写的过程中，我发现自己并不清楚具体的细节，比如，为什么生成的 RDB 文件这么小，这是如何做到的？RDB 和 AOF 分别适合用在什么场景中呢？

翻阅源码之后，我才发现，RDB 文件之所以很小，一方面是因为它存储的是二进制数据，另一方面，Redis 针对不同的数据类型做了进一步的压缩处理（例如数字采用 int 编码存储），这就进一步节省了存储空间。所以，RDB 更适合做定时的快照备份和主从全量数据同步，而 AOF 是实时记录的每个变更操作，更适合用在对数据完整性和安全性要求更高的业务场景中。

这种用输出反哺输入的方式，也是强化收获的一种有效手段，我真心建议你也试一试。

持续精进：做好精力管理

拥有了好奇心，也找到了合适的方法，也并不是万事大吉了。我们可能还会面临一个问题：“我非常想把某个技术学好，但是我总是被一些事情打断，而且有的时候总想犯懒，这该怎么办呢？”

其实这是一个效率问题。人天生是有惰性的，所以我们需要借助一些东西去督促我们前进。想一下，工作时，什么时候效率最高？是不是接近 deadline 的时候呢？

这就说明，当我们有目标、有压力的时候，才会有动力、有效率地去执行。所以，我常用的一个方法是，在学习某个领域的知识时，我会先按照从易到难的顺序，把它拆解成一个个大的模块，确定大框架的学习目标；接着，我会继续细化每个模块，细化到一看到这个任务就知道立马应该做什么的程度。同时，我还会给每项任务制定一个 deadline。

简单举个例子。我在学习 Redis 的基础数据类型时，首先确定了 String、List、Hash、Set、Sorted Set 这五大模块。接着，我又对每个模块继续进行拆分，例如，Hash 类型底层实现可以拆分成“压缩列表 + 哈希表”这两种数据结构实现，接下来，我就继续细化这两个模块的内容，最终确定了一个个小目标。

怎么完成这些小目标呢？我采用的方式是用**番茄工作法**。

我会把这些细化的目标加入到番茄任务中，并且排列好优先级。随后，我会在工作日晚上或者周末，抽出一整块的时间去完成这些小目标。在开启番茄钟时，我会迅速集中精力去完成这些任务。同时，我会把手机静音，放在自己够不到的地方。等一个番茄钟（25 分钟）结束后，休息 5 分钟，调整下状态，然后再投入到一个番茄任务中。

在实施的过程中，我们可能会遇到一些阻碍，比如说某个任务比想象中的难。这个时候，我会尝试多用几个番茄钟去攻克它，或者是把它的优先级向后放，先完成其他的番茄任务，最后再花时间去解决比较难的问题。

长时间使用这种方法，我发现，我的效率非常高。而且，把番茄任务一个个划掉之后，也会有一些小小的成就感，这种成就感会激励我持续学习。

最后，我还想再说一点，就是要**投入足够多的时间**。不要总是抱怨想要的得不到，在抱怨之前，你要先想一想，有没有远超出他人的投入和付出。想要走在别人的前面，就要准备好投入足够多的时间。

有时候，可能你会觉得，学习某一个领域的技术感觉很枯燥，因为细节很多、很繁琐，但这都是很正常的。现在我们所看到的每一项技术，都是开发者多年的总结和提炼的成果，它本身就是严肃的，你必须花足够多的时间去分析、研究、思考，没有捷径。

千万不要指望着借助碎片化学习，搞懂某个领域的知识。我所说的碎片化有两层含义：一是指内容碎片化，二是指时间碎片化。

不知道你没有遇到这种情况，当你看完一篇技术文章时，可能以为自己已经掌握了这些知识点，但是，如果别人稍微问一下相关的知识点，你可能就答不上来了。这就是因为，你学到的东西是碎片化的，并没有形成知识体系。

但是，只有系统化学习，你才能看到这项技术的全貌，会更清晰边界，知道它适合做什么，不适合做什么，为什么会这样去设计。

另一方面，不要幻想着只在地铁上学一会儿，就能把它学会，这样就有点太高估自己了。因为在很短的时间内，我们没有办法深入地去思考，去深入了解这个知识点的前因后果。你必须在晚上或者周末抽出一整块时间，去理清每个知识点之间的关系和边界，必要时还需要动手实践。

因此，如果真正想去要握某项技术，就必须需要付出整块的时间去学习，而且，必须是系统化的学习。

总结

今天，我跟你分享了我的一些学习总结，包括领先别人一步的心理建设，事半功倍的学习方法，以及持续精进的精力管理方法。

这些道理其实很简单，也很容易理解，但是能真正做到的，也只有 20% 的人，甚至是更少。所以，希望我们都能真正地行动起来，进步的路很长，我们一定要让自己在路上。

最后，希望这些内容对你有所帮助，我也很期待你在留言区聊一聊你的学习方法或习惯，我们一起交流和进步。

上一篇 加餐（二）| Kaito：我是如何学习Redis的?

下一篇 期中测试题 | 一套习题，测出你的掌握程度

精选留言 (17)

写留言

Kaito
2020-09-25

这篇文章主要和大家分享了我的学习方法论，道理其实都很简单，但最重要的一点就是：行动！

我们每个人读懂一篇文章的成本都很低，但吸收的东西却不一样，而采取行动的人更少之又少。但也正因为如此，能和别人拉开差距的，就是这些采取行动的少数人。如果还能...

展开

5 44

Geek_9a0c9f
2020-09-29

大佬，很多东西看完，很容易遗忘，这块你是怎么处理的。

1 1

Mr.蜜
2020-09-28

这是典型的费曼学习法。在以前读书的时候，总感觉学霸们越学越厉害，怎么都追不上，其实就是因为，他们懂了以后，会用自己的语言组织，再交给其他同学。我也这么做过，在教授的过程中，等于自己也过了一遍知识点。学一遍、整理一遍、教授一遍，这样至少把知识点过了三遍。人就成长起来了。在学习redis的时候，我也在一边学习的时候，一边和同事一起探讨，一起学习和提升。

展开

1

zhou
2020-09-27

有共鸣，收藏起来，懒惰的时候看看这篇文章😃

展开

1

慎独明强
2020-09-25

自己目前的状态就是自己目前要学的东西很多很多，但是又不知道从何开始下手；去年自己花时间在rocketmq，因为基础不够，学到的有限，不过自己感觉比没有系统去看过、实践过的了解的要多。后面系统学习了jvm、mysql，对于这三块，在面试当中基本没什么压力； redis没有系统学过，在今年找工作遇到很多阻碍；下半年计划是系统性学习redis这块，netty，学完了自己还想去重看 rocketmq；还有设计模式这块，自己都是碎片化学...

展开

2 1

FuriousEric
2020-09-29

2-8理论新的解释很赞啊,感谢kaito, 有所收获

展开

金鑫
2020-09-27

这篇好

展开

张先生丶
2020-09-25

很棒！

展开

咸鱼
2020-09-25

集中的时间，全身心的投入，知难而进的斗志，确实可以很高效的学习新的知识，只是种状态需要克服的东西比较多，所以也难

展开

hello
2020-09-25

挺优秀的

展开

dasheyuan
2020-09-25

又是加倍的收获，血赚～

展开

小杰
2020-09-25

优秀

展开

五河士稻
2020-09-25

能把自己的学习经验和学习方法分享出来就很伟大，作为一个憨憨学习者，太需要大佬这些经验了，一方面可以借阅学习方法进行学习，另一方面可以认识到和大佬的差距催促自己学习。

williamcai
2020-09-25

像优秀的人学习

展开

东

2020-09-25

赞番茄工作法

展开

jinjunzhu

2020-09-25

收获很大，加油

展开

jacky

2020-09-25

讲得很好，鞭策我继续努力!

展开