100 | 如何将设计思想、原则、模式等理论知识应用到项目中?

王争 · 设计模式之美

上一节课，我们对整个专栏的理论知识点做了串讲复习，不知道你掌握得如何？对于上节课总结的重点内容，我希望你能多花点时间搞透彻，对于一些不那么重要的内容，我建议你把专栏当作工具资料，用到的时候随手查阅，再深入学习研究。

实际上，很多小伙伴反应，虽然理论掌握得差不多，专栏也很贴近实战，每个知识点的讲解都有结合实际的代码案例，并且最后还有集中的项目实战，但落实到自己写代码的时候，还是无法将学到的理论知识很好地应用到其中。今天，我们就再聊一聊，如何将设计思想、原则、模式等理论知识应用到实际的项目开发中。

话不多说，让我们正式开始今天的学习吧！

吃透理论、先把书读厚再把书读薄

把理论知识灵活地应用到实践的前提是，对理论有透彻、无盲点的理解。如果我们对理论知识掌握得似懂非懂，在实际软件开发中，遇到跟专栏中讲过的问题类似，我们可以照葫芦画瓢去

解决，但是，如果问题背景稍有改变，我们就会比较难联想到对应的理论知识，更难灵活地应用理论去解决。

要想透彻理解专栏中的每个知识点，一个是要多看几遍，二是要有死磕精神。虽然这两个方法可能已经是老生常谈了，听起来也没有什么高大上的，但从我自身的学习经验来讲，它们确实很有用。

书读百遍其义自见。有的时候，对某个知识点，如果看一遍看不懂，你就硬着头皮多看几遍，或者隔几天再回过头来看一遍，你会发现原来很多看不懂的地方，自然而然就懂了。

慢就是快，快就是慢。专栏涉及的内容很多，但我们花一年把所有的知识点学透彻，实际上是件一劳永逸的事情。这个过程虽然看似漫长，但收益却很多。对比而言，如果你只是为了追求结课速度，花一两个月、甚至一两个礼拜，把课程学完。这看似很快，但实际上收获会很少。

先把书读厚，再把书读薄。反复地学、持续地看。先把书读厚，等到你把所有的知识点都理解透彻，并且在脑子里建立起清晰的知识体系之后，你会发现，实际上专栏的内容也就那么点东西，并不难记忆。前提是你先要花时间把书读厚，然后才能做到把书读薄。

虽然这里我们讲到书读百遍其义自见，也讲到死磕精神，但是，我必须强调一下，有的时候，对于某个知识点，我们看了很多遍、死磕了很长时间，如果还是没法透彻理解，我们也不要过于钻牛角尖，非得“现在立刻马上就要”把它拿下。我们可以先把这个知识点放一放，先看看后面的内容，隔一段时间，让知识沉淀、消化一下，再回过头来看也是可以的。

在实战中反复学习、模仿和借鉴

很多人说，理论的知识学了就忘，忘了是不是就等于白学了呢？实际上不是的，起码对于我们这个专栏的内容来说，并非如此。在专栏中，几乎每个知识点，我都结合具体的案例和代码来讲解，目的就是为了让你在实战中学习。所以，你学习的重点不是理论知识，而是跟随我的思维逻辑，学习如何分析代码问题，解决代码问题。通过专栏，经过上百个代码案例的剖析学习，即便理论知识你有可能会忘记，但这种潜移默化的能力锻炼，是不会丢掉的。

对于新手来说，最好的学习方法之一就是“模仿”。我之前在《数据结构和算法之美》专栏中，也曾经讲过，如果你是一名算法或者编程初学者，自己编写代码实现各种数据结构和算法，可

能会比较困难。在这种情况下，你就可以先从“照抄”开始，把所有的代码都抄一遍或者抄几遍，然后再慢慢地过渡到自己去默写。

对于我们这个专栏来说，如果你项目经验不多，要想把理论一下子就灵活地应用到项目中，实际上这个要求也有点过高了。同样，你也可以先从模仿开始，对于项目中遇到的跟专栏中相似的开发场景，你可以借鉴专栏中的设计思路、代码实现。实际上，除了专栏中的案例之外，我们还有很多借鉴的来源，比如我们前面剖析过的经典开源项目（Spring、MyBatis），还有项目中大牛同事写的代码等等。

有人为了刷 LeetCode、刷算法题，会积累一些算法模板，对于相似的问题，套用算法模板来快速解决。同样的，我们也可以积累设计模板、代码模板，对于相似的功能需求，我们可以套路设计模板、代码模板来解决。比如，在前面讲到的限流框架、灰度组件中，加载配置文件这样一个常用功能的设计和实现，我们就可以抽象成模板。对于其他项目中类似的功能需求，直接套用就可以，不用从零开始设计和实现了。

刻意思考、刻意训练、追求极致

要想把理论知识应用到项目中，并且做到润物细无声、融会贯通、无招胜有招，我们需要经过漫长的刻意思考和刻意训练。

拿到一个功能需求的时候，我们先去思考一下如何设计，而不是上来就写代码。写代码时，我们也要时刻思考代码是否遵循了经典的设计思想、设计原则，比如是否足够可扩展、是否满足 SOLID 原则、可读性如何等等。

写完代码之后，我们再思考一下，代码是否有进一步优化的空间。做 Code Review 的时候，看到别人的优秀的代码，我们就去思考一下，有哪些值得借鉴的地方。

总之，在平时的开发中，我们要刻意的去做这种跟代码质量、代码设计相关的思考训练。时间长了，这种思考就能成为习惯和本能反应，慢慢地，你的代码能力也就不自觉地提高了。

刻意训练的过程在前期会比较痛苦。为了尽可能写出高质量的代码，为了刻意训练在代码中应用理论知识，原本半天就能写好的代码，可能需要好几天才能完成。在最开始的时候，我建议

你把专栏中讲到的经典的设计思想、原则、模式，打印出来贴在电脑旁，每次写代码的时候，对照着每个知识点，一个一个去审视代码。

跟前面讲到的花很多时间把理论知识搞透彻的道理一样，刻意训练虽然在前期需要投入更多的时间和精力，但也是一件一劳永逸的事情。等到训练到一定程度之后，你就会发现，在不依赖这个知识点列表的情况下，你开始不自主地考虑代码质量问题、设计问题，不经意写出的代码，就完全符合高质量代码的要求了，而且，写出好的代码并不会花费更多的时间了。相反，如果不愿意为刻意训练付出时间和精力，每次写代码都马马虎虎，代码质量永远都提高不了，也永远都达不到灵活应用理论知识到项目中。

我经常说，多花点心思和时间把一段代码写好、优化到极致，比写十段凑活能用的代码，对提高代码能力更有效。实际上，这就好比刷 LeetCode 算法题，对于一些经典算法的经典题目，我们一定要刻意地多花点时间搞清楚，死磕一下。虽然死磕的过程很痛苦，可能会花掉你很多时间，但一旦搞明白之后，其他类似的题目都可以很快解决。相反，如果看到不会的问题，连思考都不思考，就去看答案，那做十道题，也还是没有太多长进，看到题目不看答案还是写不出来。

课堂讨论

关于如何将设计思想、原则、模式等理论知识应用到项目中，除了我分享的这些经验之外，你还有哪些经验可以分享给大家呢？另外，经过这么长的学习和训练，在自己参与的项目中，你是否开始关注代码质量问题，代码能力是否有提升了呢？

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。

全部留言 (36)

最新　精选

业余草
2020-07-07

总结一套脑图：https://static001.geekbang.org/horde/e8/e812cf142371fed874cd5faaa797cc5d.png

作者回复: 👍

共 2 条评论>　33

W
2020-07-09

求出系统设计课

作者回复: 😂

6

岁月神偷
2020-06-22

结课撒花，真的非常感谢争哥的知识分享。在这半年的学习时间里，我的代码质量和编码水准确实有质的提升，同时将所学的知识转化为实践，为公司写了一套用于记录日志的通用底层框架，目前已被广泛使用。有一点学习感悟想要分享，一家之言，仅供参考。
在整个学习过程中，我非常重视那些在日常工作中能即刻运用起来的知识点，例如常用的设计模式，编程规范，持续重构的技巧和单元测试。这些容易被运用到日常工作的知识点，是我实践的切入点，快速将这些理论知识运用到实战中，加快了我对理论知识的掌握速度。
有了一定的积累后，我开始尝试将所学的知识赋能给团队，给团队内的小伙伴提供编程的指导，提升他们的代码质量。这个过程是更加困难的，从学生到老师，需要更加扎实的知识基础，从懂了到完全懂了是有非常长的一段路要走的，想要给别人讲清楚，自己必须最清楚，讲着讲着发现什么讲不清楚了，就知道自己是哪里没搞懂了。在沟通交流的过程中，能更好的发现自己还有什么不足，以便更有针对性的查漏补缺。

共 1 条评论>　70

Jxin

2020-06-22

1.看来还得做下专项的刻意训练。虽然大部分东西，问答我应该都没问题。但对部分场景的敏感识别，这个应该还是比较薄弱的，需要刻意训练。

2.这是一门知识型+技能型的课程。

3.作为一门知识型课程，我觉得很棒。栏主的见解清晰独道，既好理解也能抓到重心。

4.作为一门技能型的课程，很遗憾，有所欠缺。但这主要也是受限于授课方式。技能型的课程，需要演示，实践，反馈三个环节。课程里面有实战的部分，但仅限于演示。而技能型课程缺少实践和反馈是很难被掌握的。所以才会有开篇栏主说的，部分同学理论感觉都ok，但却用不到项目中去的现象。

5.拉个刻意训练的群应该不错。读书百遍其义自见。针对性的写设计模式，阅读他人的应用，参与讨论，应该会是个很好的补充。

共 3 条评论>

16

小乙哥

2020-11-01

花了四个月时间，走完了专栏。相信多年之后，回头再看，这个专栏还是对自己有影响的！其实，一开始不理理解争哥花那么大的篇幅功夫讲解面向对象和设计原则的，可是，走到设计模式，尤其快看完设计模式的时候，我就发现前半部分的“心法”显然要重要设计模式的“招式”（站在架构设计层面，其实设计模式也是心法）。我还发现自己两个转变：一是在写代码的时候不拿着设计模式到处钉钉子了；二是看spring、mybatis这些源码的时候，不是无头绪的到处debug了，也不是上来就抱着大段代码乱啃一起。而是尝试从设计、扩展性、解决什么问题、应用在什么场景、框架易用性和低侵入性是怎么做的？怎么把设计模式巧妙地糅合进问题场景中的

先mark一下这一程的感受，为未来的自己和当下自己再次见面，留下点交流的内容

共 1 条评论>

12

Monday

2020-06-22

我把 极客时间 当作工具使用

共 1 条评论>

12

mamba

2020-06-23

今天是2020年6月23日。在专栏更新完最后一节课的后一天，我要开始学习专栏了。不知道需要过多久，我能从第一篇再来到这里。期待与自己的下一次相遇。那一定是收获满满的一段旅程。

共 1 条评论>

6

skull

2020-06-29

我花了5个月学，我自认为掌握的挺扎实，项目中也能合理适度应用设计模式了

4

Heaven

2020-06-22

反复重读,将书读薄,知识都是厚积薄发.后来路越来越宽,这已经不是我第一次看设计模式相关的知识了,但是每次都能有新的理解

3

return

2021-02-09

争哥牛逼， 读了很多设计模式的资料 看的云里雾里， 争哥一讲 明明白白， 如同算法专栏，神。

期待争哥带来其他方面的学习

2