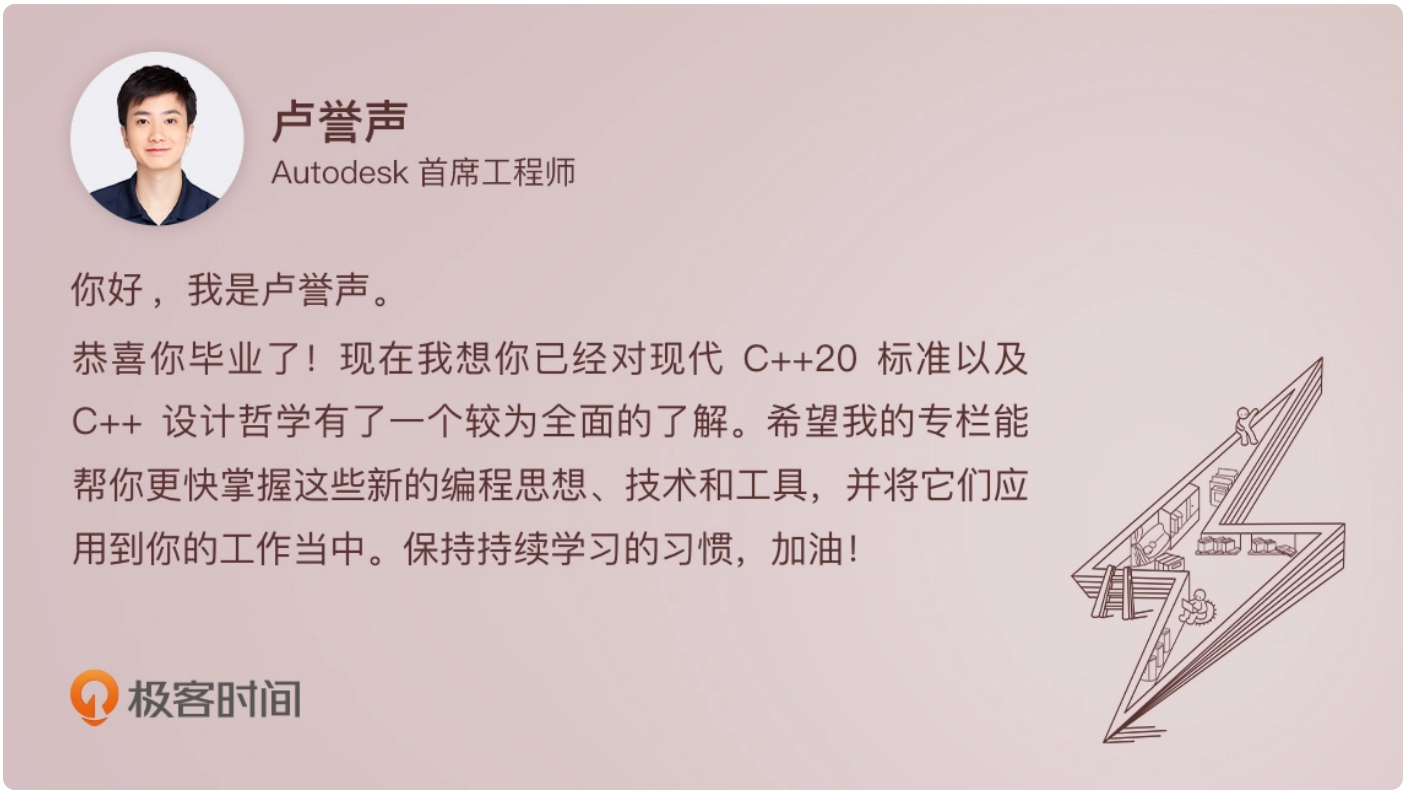
结束语丨这只长颈鹿在我家后院生活得不错

2023-03-17 卢誉声 来自北京

《现代C++20实战高手课》

课程介绍 >



讲述：卢誉声

时长 06:06　大小 5.57M

你好，我是卢誉声。

时间过得真快，专栏到这里就真的要结束了。回顾过去一段与你相聚的时光，我最大的感受就是，我们一同完成了一项不可能完成的任务，感谢你的一路相伴。

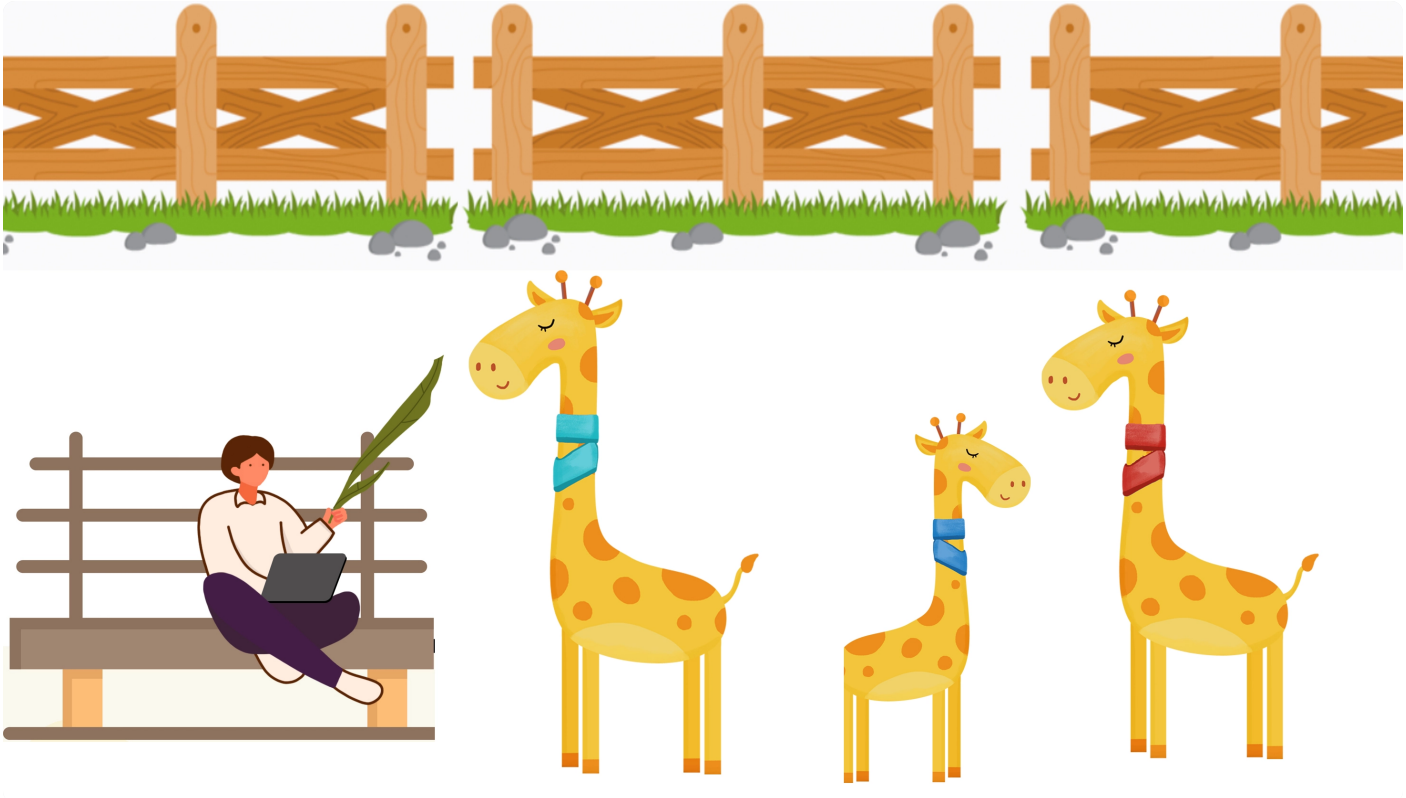
在我看来，C++ 是一门生命力旺盛、功能异常强大的编程语言，即便将评价的尺度放在诸多现代高级编程语言里，它的地位和流行程度也叹为观止。因此，归纳总结并有节奏地讲解它的新特性，是一项极为艰巨的任务。

但 C++ 这样的语言是一座潜力巨大的宝藏，它的新特性让我在编写代码时受益良多，也想把这种快乐分享给更多人，这也是我写这门课程的初心。

专栏中涵盖了自 C++11 开始支持的现代 C++ 特性，包括 C++14、C++17，同时将重点放在了全新的 C++20 核心特性变更上。甚至，我们还提前展望了未来的标准演进及其底层逻辑。

因此，你会觉得这趟学习之旅的内容体系比较庞大，而且有些“烧脑”。但坚持下来，相信你一定不虚此行。

在专栏的最后一课，你不妨切换一下大脑的上下文，跟随我一起探讨一下“如何喂养长颈鹿”的问题。



我曾在另一个专栏（《动态规划面试宝典》）中，就跟大家聊过有关长颈鹿的问题。不过你不用担心，你不一定需要去回顾原来那篇文章。今天我想借这个机会，跟你聊一聊我对个人学习和成长的感悟。

众所周知，IT 行业的一大特点就是发展迅速。这从一定程度上导致了一个问题，那就是有很多人会觉得背后有一双无形的手，在推着你不断学习新的编程语言、新技术、新方法，甚至是新的思维模式。在很多时候，这种所谓的学习驱动力是被动的。因为，你不得不跟随潮流或者趋势，否则，就会面临淘汰的风险。

但这件事真的无解，或者说就只能这样了么？

在这个信息爆炸的时代，我们获取信息的途径越来越多，信息量巨大，而且留给我们的大多都是碎片化时间，想要拥有一整块的时间专注于某件事，或者专心看完一整本书都是很困难的。所以，就会有很多人开始变得焦虑，在做很多事情的时候都希望能尽快得到“结果”。

但是，我自己反复实践以后却发现。越在乎结果，可能越“欲速则不达”。

越专注投入一件事儿，哪怕是一件小事，沉浸其中，享受过程，也许未来某个时刻就会碰撞出新的可能性。未曾期待结果，但惊喜却往往不期而至。

我曾经花费不少碎片时间，学习研究自己有兴趣的内容，比如“消息循环”这样一个朴素的概念和技术。

事实上，一句话基本就能概括这项技术是什么，以及它的工作原理。不过，我稍微多花了一些时间和精力，扒开几种不同运行时以及不同编程语言的具体实现后，我对它有了更深的理解，并在不久之后，“碰巧”将其应用到了正在研发的大规模复杂数据处理系统里——这是一个我未曾预想到的结果，就像 C++ 模板元编程被当作了一门“新语言”一样意外（当然了，我的小探索不能和它相提并论）。

任何一项流行技术或被业界广泛采纳的编程语言，它们都有存在的意义和价值。如果你发现一项技术历久弥新，那么必然有其独到之处。我之所以在专栏中多次提到“学习和掌握 C++ 标准演进的底层逻辑”，就是因为知识的积累和沉淀尤为重要。

对一项技术的底层逻辑的理解深度，以及编程思想精进的高度，往往决定了一个人的上限。比如说，当你对 C++ 设计哲学熟稔于心的时候，其实就不难理解为何 C++ 不会去默认提供垃圾回收（GC）机制。同时，在你了解 JVM 的垃圾回收机制时，也更能体会垃圾回收算法的设计思路和工作原理。

比如，当你能够透彻理解 STL 中不同算法和数据结构的原理，就能在解决实际问题时选用最合理的算法和容器。

又比如，当你对计算机体系结构（比如 CPU、内存和缓存机制）非常熟悉时，就能充分利用 C++ 与硬件直接映射的能力，合理组织内存结构，甚至嵌入汇编，使得你的程序保持最好的性能。

再比如，当你深入了解了 C++ 的编译与链接原理与具体实现后，能够开发出可维护性与可移植性更好的应用。

而这些深入的东西，可能其实源于你一点点的好奇心，从某个小的技术点作为起点，掌握并吃透它，然后再继续深挖，直到“底层”。**这一切，都是一脉相承的。保持持续学习的习惯，你一**

定能有实实在在、真真切切的收获。

就像我曾经心中的那只长颈鹿一样，我每天都会花一点时间专注在一件“持续”的事情上。而现在，经过这几年的专心致志，那个位置，已经有三只长颈鹿了。

在写这篇专栏结束语时，我的心情复杂，我觉得还有不少“干货”没来得及跟你分享。但我还是努力把我内心最想分享的内容提炼总结给你。我仍然十分期待着，能有机会与你交流更多。

如果你也有想对我说的话，不妨通过问卷告知，你们的每一条反馈我都会认真去看。

最后，我还是想借用我特别喜爱的电影《信条》中的台词，来结束我们的专栏。

> 我在整个专栏中为你穿针引线
> 对我来说，这是一段美好友谊的休止
> 但对你来说，这是一个新的开始
> 你我的这段经历将永存 You’ll love it, you’ll see

上一篇　23丨未来展望：透过未来标准演进看C++设计哲学

精选留言 (3)

写留言

peter
2023-03-17 来自北京

感谢老师的精彩课程！！
Q1：C++为什么不需要垃圾回收？
Q2：关于C++编译与连接，有好的文章推荐吗？
Q3：从源代码学习C++，有好的开源项目吗？

作者回复: 感谢你一路走过，我们一起完成了这项“壮举”——学习最新的C++标准和设计思想。

1. 垃圾回收本质上是一个运行时特性，首先这就违背了C++的设计哲学，即不为任何抽象付出不可接受的多余运行时性能损耗。垃圾回收从一定程度上的确能够简化运行时内存管理，但是他会带来极大的运行时性能损耗（相较于C++本身的性能来说）。C++已经通过智能指针提供了另一种解决内存管理问题的方案，他足够灵活，而且符合设计哲学。另外，C++是可以实现运行时垃圾回收的，你不妨搜索一些相关文章来学习了解，但是我还是想强调这仍然不是C++该主流支持或思考解决的问题。

2. 推荐你阅读《高级C/C++编译技术》这本书，我很欣赏原作者对内容尺度、深度的把控，以及讲解的方式，因此翻译了这本书。推荐给你。

3. 我曾开源了类Storm/Flink的C++版本的高性能分布式实时处理系统，另外还有一套高性能网络库，你可以到本专栏的代码链接中找到代码仓库。另外，推荐你去看一看 aws-cpp-sdk，他的实现足够模块化，而且博大精深，涉及的知识和领域很广，是一个不错的深入学习和掌握大规模工业级C++项目的好途径。

2

Geek_QiDian

2023-03-17 来自北京

太好了，老师的感觉和我的不谋而合

作者回复: 赞，十分开心能遇到志同道合的朋友

1

Geek_548f11

2023-03-28 来自广西

谢谢 老师，我想请问下c++20增加对管程内容吗？

作者回复: C++20并没有提供管程的实现，如果需要得自己基于C++提供的信号量等工具来实现。