How to Learn and Practice C++ Learning modern C++ with Linux

邓钰川

四川大学计算机学院

2023年2月27日





- 1 引言
- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- 5 后记





1 引言

引言

- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- 5 后记



引言

C++ 编程风格

- 数据抽象
- 过程化
- 面向对象
- 泛型编程

C++ 知识层级

• 层级一: 语法/语意

• 层级二: 专家经验

• 层级三: 底层机制

• 层级四:设计观念复用

网友的评价

- 精通 C++ 是一个艰巨的任务。为什么 C++ 比别的语言难学这么多? 是因为 C++ 之父 Bjarne Stroustrup 说过的一句话"我特别讨厌语言的设计者把自己的喜好强加给用户" C++ 能够自由的让你放弃某些部分,而别的语言会阻止你放弃某些部分。
- 谷歌工程师师对 C++ 的掌握有两个级别: 拥有 C++ 的 readability (可读性) 认证; 顾问级 C++ 程序猿
- Never trust a programmer who says he knows C++

- 1 引言
- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- 5 后记



(ロ) (部) (注) (注) (注) り(○)

如何学习 C++

引言

- ① 熟悉基础语法: 学习 C++ 的第一步是掌握其基础语法。您可以阅读 C++ 的教程和文档,以及编写一些简单的程序来加深对语法的理解。
- ② 学习面向对象编程: C++ 是一种支持面向对象编程的语言, 学习面向对 象编程是深入学习 C++ 的必要步骤。您可以学习 C++ 中的类、对象、 继承、多态等概念,以及如何使用它们来编写高效的代码。
- ③ 掌握模板和泛型编程:模板和泛型编程是 C++ 的特色之一,掌握这些概念可以帮助您编写更加通用和高效的代码。
- ④ 学习标准库: C++ 标准库是一组提供了许多有用功能和数据结构的库, 包括容器、算法、迭代器、文件操作等。深入学习 C++ 需要掌握标准库的使用方法,以便在编写代码时使用它们。
- 5 探索高级主题: C++ 还有许多高级主题,如多线程编程、网络编程、元编程等。了解这些主题可以让您成为一名更加高级的 C++ 开发人员。

总之, C++ 是一种相对于 C 语言更加高级和复杂的编程语言, 具有更多的特性和功能。C++ 的面向对象编程、函数重载、新的运算符和关键字、以及强大的标准库是其与 C 语言的主要区别。

◆ロト ◆部 ト ◆ 恵 ト ◆ 恵 ・ 釣 へ ⊙

如何学习现代 C++

- ① 了解 C++11、C++14、C++17 和 C++20 的新功能: 现代 C++ 引入 了许多新功能,如 auto 关键字、范围 for 循环、lambda 表达式、智能指 针、并发编程等。了解这些新功能可以让您更加高效地编写 C++ 代码。
- ② 学习 RAII 和智能指针:现代 C++ 强调资源获取即初始化 (RAII) 的编程风格,可以帮助您编写更加安全和健壮的代码。智能指针是现代 C++中常用的 RAII 技术之一,可以自动管理动态内存分配。
- ③ 掌握模板和泛型编程:模板和泛型编程是现代 C++ 的重要特性之一,可以让您编写更加通用和高效的代码。了解模板元编程、可变参数模板、模板特化等高级技术可以让您更好地使用模板和泛型编程。
- ◆ 学习并发编程:现代 C++ 提供了一组强大的并发编程库和工具,如 std::thread、std::async、std::atomic、std::mutex 等。学习并发编程可以 让您编写更加高效和可靠的多线程程序。
- ⑤ 阅读现代 C++ 书籍和文章: 有许多现代 C++ 的书籍和文章可以帮助 您深入学习现代 C++.





- 1 31=
- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- 5 后记



(ロ) (部) (注) (注) (注) の(○)

Stanford CS106L 的建议

When developing C++, developers rarely start by copying a pasting a blank Qt Creator project. Although Qt is an excellent integrated development environment (IDE) for learning C++, it's important for you to learn how to build and run your C++ files yourself.¹

在开发 C++ 时,开发人员很少从复制粘贴空白的 Qt Creator 项目开始。尽管 Qt 是学习 C++ 的优秀集成开发环境 (IDE),但了解如何自行构建和运行 C++ 文件对您来说非常重要。



¹https://web.stanford.edu/class/cs106l/assignment-setup → ⟨ ≥ ⟩ ⟨ ≥ ⟩ ⟨ ○ ⟩

为什么选择在 Linux 下学习 C++

- ① 开源环境: Linux 是开源的操作系统,许多 C++ 的开源工具和库都是在 Linux 环境下开发的,这些工具和库可以让您更好地理解和学习 C++ 语言。
- ② 强大的命令行工具: Linux 系统提供了许多强大的命令行工具,这些工具可以让您更好地掌握 C++ 编译和调试的过程。例如,您可以使用 gcc和 g++ 编译器来编译 C++ 代码,并使用 gdb 来调试代码。但是在其他操作系统上,您可能需要安装不同的编译器和调试工具,可能会遇到一些困难。
- ③ 许多开发工具和库: Linux 环境下有许多开发工具和库可供选择,如GNU 工具集、Qt等,这些工具和库可以帮助您更好地学习和应用 C++编程语言。但是在其他操作系统上,您可能需要寻找不同的工具和库,并花费更多的时间来了解它们。
- ④ 跨平台性: C++ 是一种跨平台的编程语言,可以在不同的操作系统上编译和运行。选择在 Linux 环境下学习 C++ 可以帮助您更好地理解如何在不同的操作系统上运行 C++ 代码。但是,如果您没有在 Linux 环境下学习 C++,您可能需要更多的时间来了解如何在不同的操作系统上编译和运行 C++ 代码。

←ロト→面ト→直ト→直ト 直 りへの

- 11 引言
- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- 5 后记





算法竞赛与实际工程开发的区别

- 動据结构和算法的重要性:算法竞赛强调设计高效的算法和数据结构,对算法的时间和空间复杂度要求很高,而在实际工程开发中,对算法的复杂度有一定要求,但更注重实现的可读性、可维护性、可扩展性和稳定性。
- 标准库的使用:在算法竞赛中,为了减少代码量和提高效率,通常使用 STL(标准模板库)和其他高级库。而在实际工程开发中,使用标准库可以提高开发效率和代码质量,但需要注意库的选择和使用,以免引入不必要的复杂性和性能问题。
- 3 错误处理和调试:在算法竞赛中,通常只需要输出错误信息并停止程序, 而在实际工程开发中,需要对各种异常情况进行处理,并记录错误信息 以便排查和修复。此外,在实际开发中需要使用调试工具来快速定位问 题和调试程序。
- 《代码风格和规范:在算法竞赛中,由于时间紧迫,代码风格和规范可能不是首要考虑因素,而在实际工程开发中,良好的代码风格和规范可以提高代码的可读性、可维护性和可扩展性。
- **⑤** 代码优化和性能:在算法竞赛中,对算法和代码的优化非常重要,因为效率是比赛成败的关键,而在实际工程开发中,需要考虑优化和性能,但通常不需要过度优化,因为代码的可读性和可维护性也很重要。



为什么养成良好的代码风格

良好的代码风格是指编写代码时遵循一定的规范和标准,以保证代码的可读 性、可维护性和可扩展性。养成良好的代码风格有以下几个好处:

- 提高代码可读性:良好的代码风格可以让代码更易于理解、减少读者理 解代码的困难度,从而提高代码的可读性。
- 2 提高代码的可维护性:良好的代码风格可以让代码更易干维护、例如命 名规范、注释规范等,可以让开发者更快地了解代码的结构和功能,从 而更容易进行修改和维护。
- 3 提高代码的可扩展性:良好的代码风格可以让代码更易于扩展和改进, 因为遵循一定的规范和标准可以使代码更加一致和清晰。
- 提高开发效率: 养成良好的代码风格可以减少编写代码时的困惑和错误、 从而提高开发效率。同时,良好的代码风格也有助于更好地组织代码和 提高代码的质量。

强烈地推荐您使用谷歌的 C++ 风格。2



- 1 引言
- 2 学习 C++ 的方法
- 3 在 Linux 下学习 C++
- 4 养成良好的代码风格
- **5** 后记





推荐 C++ 书籍

- 层级一: 语法/语意 (C++)
 - ① C++ Primer (中文版, 侯俊杰译) by Stanley B. Lippman
- 层级二: 专家经验 (C++/OOP)
 - ① (More)Effective C++(中文版, 侯俊杰译), by Scott Meyers.
 - ❷ (More)Exceptional C++ (中文版, 侯俊杰译), by Herb Sutter
 - **3** Effective Modern C++, by Scott Meyers
- 层级三:底层机制 (C++ Object Model)
 - Inside the C++ Object Model (深度探索 C++ 物件模型, 侯 俊杰译),by Stanley Lippman.
- 层级四: 设计观念的复用 (C++/Patterns)
 - 1 Design Patterns: Elements of Reusable Object Oriented Software, by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
 - Modern C++ Design: Generic Programming and Design Patterns Applied by Andrei Alexandrescu.

3更多参考书籍可以查看 https://stackoverflow.com/questions/388242/thedefinitive-c-book-guide-and-list

邓钰川

四川大学计算机



◆ロト ◆部ト ◆恵ト ◆恵ト 恵 めなぐ