

期中周 | 扩展现有协程框架，实现高级任务调度

2023-02-13 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 02:38 大小 2.41M



你好，我是卢誉声。

时间过得真快，从 1 月 16 日上线到现在，我们的学习之旅已经走过了将近 1 个月，不知道你的收获如何呢？

之前我说过，如何在实际工程项目中通过 C++20 的新特性解决问题，将是我们学习的重点。从我的经验看，想要从初步了解到熟练应用这些新特性，也确实需要多花些功夫，而且不光要多看，更要多练，这样才能形成“手感”。

所以，我特意安排了这次期中周。从 2 月 13 日开始到 2 月 19 日结束，这期间我们会暂停更新正文内容，你可以好好利用这一周的时间，回顾一下前面学过的知识，查漏补缺。

在实际工作里，新的挑战通常是我们学习锻炼的良机。所以，期中周我还给你准备了一道测试题，一起来挑战一下吧！

C++ Coroutines 是 C++20 标准带来的最令人兴奋和激动的全新核心语言特性之一。课程第七讲至第十讲，用了不少篇幅详细讲解 C++ Coroutines。不过，正如我们所见的，C++20 中提供的协程仅提供了语言层面的支持，缺乏标准库的支持。因此，在标准得到进一步演化之前，我们不得不实现绝大多数接口约定。

这让我们使用 C++ Coroutines 的成本提高了。我在课程中已经给出了基于接口约定的实战案例。但是，这个案例还有不少可以提升的空间。请你在现有代码的基础上，尝试实现后面的功能。

现有代码地址： [🔗 https://github.com/samblg/cpp20-plus-indepth/tree/main/09_and_10/Asynccpp](https://github.com/samblg/cpp20-plus-indepth/tree/main/09_and_10/Asynccpp)

首先，在 Windows 上，通过 CMake 编译出课程里提供的 asyncpp 代码。


在目前的设计中，只支持简单的、耗时较短的异步 I/O 任务。那么，当遇到耗时的异步计算时，我们还是需要一种更智能的任务分发机制来处理耗时任务。

所以要实现的功能就是：请你使用专有线程、线程优先级来处理这种特殊情况。具体包括两个方面。

首先，通过参数，判断一个异步函数是否需要通过高优先级的线程上执行。

另外，请你使用 `std::thread::native_handle`，实现在 Windows 系统上的优先级线程调度。

实时设置线程优先级的示例代码是后面这样。

 复制代码

```
1 #include <iostream>
2 #include <thread>
3 #include <chrono>
4 #include <cstdint>
5 #include <string>
6 #include <Windows.h>
7
8 void simpleSleep() {
9     using namespace std::literals::chrono_literals;
10
11     std::cout << "[SIMPLE] Before simple sleep" << std::endl;
12     std::this_thread::sleep_for(2000ms);
```

```

13     std::cout << "[SIMPLE] After simple sleep" << std::endl;
14 }
15
16 int main() {
17     std::cout << "[MAIN] Before create simple thread" << std::endl;
18     std::thread simpleWorker(simpleSleep);
19
20     // 获取native handle
21     auto nativeWorkerHandle = simpleWorker.native_handle();
22
23     // 设置线程优先级
24     std::cout << "THREAD_PRIORITY_HIGHEST: " << THREAD_PRIORITY_HIGHEST << std::
25     ::SetThreadPriority(nativeWorkerHandle, THREAD_PRIORITY_HIGHEST);
26     std::cout << "Thread Priority: " << ::GetThreadPriority(nativeWorkerHandle)
27
28     simpleWorker.join();
29
30     std::cout << "[MAIN] Main function exited" << std::endl;
31
32     return 0;
33 }


```

你可以考虑使用以下开发环境和工具。

- Windows 10 或更高
- Visual Studio 2022: <https://visualstudio.microsoft.com/zh-hans/vs/>
- CMake 3.16 或更高: <https://cmake.org/download/>

期待你的作业！

分享给需要的人，Ta购买本课程，你将得 18 元

 生成海报并分享

 赞 1  提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

精选留言 (1)

[写留言](#)

peter

2023-02-13 来自北京

准备搭建环境试试编译代码：

Q1：老师给出的链接中，VS22有三个版本：community、Professional、Enterprise。Community估计是最低版本了，能满足需要吗？

Q2：VS22自身没有CMake吗？如果有，为什么老师又提供了一个CMake的链接？（需要下载此CMake，然后安装到VS22吗？）

Q3：记得有VS2015，盗版的，不记得放在哪里了，也许可以找到。VS2015或VS2018可以吗？

作者回复: Q1：满足需求。

Q2：安装时可勾选安装 CMake。

Q3：需要 Visual Studio 2022 及其包含的 Visual C++以支持最新的C++标准。

