| | | |
|---|---|---|
| 【零】 | • 先创造「/OrderBook」和「/OrderBookTest」<br>• 去「/OrderBook」里 建立 class 也就是「OrderBook.cpp」&「OrderBook.h」<br>• Create a git repository out of this directory. | 无法 Push 到 private repository 的解决<br>67<br>I had the same problem, with a private repo.<br>do the following:<br>remove the remote origin<br>`git remote rm origin`<br>re-add the origin but with your username and pwd with writing privileges on this pvt repo<br>`git remote add origin  https://USERNAME:PASSWORD@github.com/username/reponame.git` |
| 【一】<br>创建完文件就写一点代码嘛 | • 「OrderBook.h」里写点简单代码：<br>  ◦ 基本的structure: “bids”, “asks"<br>  ◦ 最简单的 “is_empty()" | |
| 【二】<br>写一行代码就要测试一下嘛 | • load “googletest” 到 「/OrderBookTest/libs/googletest」里，让这个 library 处在和我project一样的environment<br>  ◦ Remove 不必要的 google test version control<br>  ◦ Remove 不必要的 主文件夹的 CMake | |
| 【三】<br>建完了 fork完了主要文件，应该开始CMake建立关系了 | • 建立「/OrderBook」里的 CMake<br>  ◦ 希望可以单独 Build，所以 建立project<br>  ◦ OrderBook 是 library 所以 add_library<br>• 建立 Root CMake<br>  ◦ 回忆下 刚刚「/OrderBook」里有一个Library，首先，让这个 directory变得 available for “including” from。<br>  ◦ 然后，建立sub-directory的structure。<br>  ◦ 最后，主要的 target 「main.cpp」尽管没用，也Link一下。 | • 有「.h」就有 【add_library】<br>• “Executable” & “Library” 都是 Target<br>• 有【add_executable】/【add_library】就要【target_link_libraries】，除非没什么好 link 的。 |
| 【四】<br>Root CMake和Library CMake搞定，建立Test 的 CMake | • 建立「/OrderBookTest」里的 CMake<br>  ◦ 希望可以单独Build，所以建立 project<br>  ◦ add_subdirectory | |
| 【五】<br>建立Test 内部的 CMake | • 在「/OrderBookTest/tests」里创立「test_orderbook.cpp」和CMake<br>• 这个「.cpp」是executable哦<br>• CMake<br>  ◦ 需要include googletest 的 directory 【include_directory】<br>  ◦ add_executable<br>  ◦ Link 因为有 target了 | |
| 【六】<br>写第一个test | • 写一个 is_empty() 的测试 | |
| 【七】<br>增加 add_bid(), add_ask() 的function | • add_bid(), add_ask() 包裹private method: add() | |
| 【八】<br>写一个Out stream operator，输出整个 OrderBook | • 【friend std::ostream& operator << (std::ostream& os, const OrderBook& book)】 | |
| 【九】<br>Test with "Cling" | | |
| 【十】<br>Actual Unit Test | • 写一个 get_bid_ask() 来支持 测试<br>  ◦ 需要一个 BidAsk struct<br>  ▪ 需要 boost::optional<br>• 写 Unit Test | |
| 【十一】<br>增加 remove_bid(), remove_ask() 的function | | |
| 【十二】<br>在 BidAsk struct里 增加 spread() function | • 用 boost::optional<float>来return | • 用cling测试带有 boost library：<br>• 【#pragma cling add_include_path("/usr/local/include/“)】 |
| 【十三】<br>增加 BidAsk的 print function | • friend operator again | |
| | | |
| | | |