Филатов_Илья_БПМ_22_3_v231230_16

Тема: “Cmake интеграция с тестами”

Оглавление

Введение

CTest - это фреймворк для тестирования, поставляемый с CMake, который помогает управлять всеми модульными и функциональными тестами в одном месте. Кроме того, позволяет фильтровать прогоны тестов, передавать аргументы в исполняемые файлы тестов, параллельно выполнять тесты и многое другое. CMake упрощает тестирование вашего программного обеспечения с помощью специальных команд тестирования и CTest исполняемого файла.

Синтаксис

Чтобы добавить тестирование в проект на основе CMake, просто используйте **enable_testing()** и **add_test** команду. У **add_test** команды следующий простой синтаксис:

```
add_test(NAME TestName COMMAND ExecutableToRun arg1 arg2 ...)
```

Пример:
enable_testing()
add_executable(rational.test rational.test.cpp)
target_link_libraries(rational.test rational doctest::doctest)
add_test(NAME rational.test COMMAND rational.test)
Представленный код является частью сценария CMake, который настраивает тестовую среду для проекта на C++. Вот что делает каждая строка:

`enable_testing()`: Эта команда включает тестирование в текущем проекте. Без этой команды CMake не будет знать, что искать тесты в вашем проекте

3. `target_link_libraries(rational.test rational doctest::doctest)`: Эта команда связывает библиотеку `rational` и библиотеку `doctest` с исполняемым файлом `rational.test`. Это означает, что при запуске `rational.test` у него будет доступ к функциям и классам, определенным в обеих библиотеках `rational` и `doctest`

4. `add_test(NAME rational.test COMMAND rational.test)`: Эта команда говорит CTest запустить исполняемый файл `rational.test` как тест. Когда вы запускаете `ctest`, это один из тестов, которые будут выполнены

Ниже приведен пример выполнения тестов на уже готовых библиотеках

```
C:\spring_oop\filatov_i_a\build\prj.test>ctest -C Release
Test project C:/spring_oop/filatov_i_a/build/prj.test
    Start 1: rational.test
1/4 Test #1: rational.test .....   Passed    0.01 sec
    Start 2: arrayd.test
2/4 Test #2: arrayd.test .......   Passed    0.03 sec
    Start 3: matrixs.test
3/4 Test #3: matrixs.test ......   Passed    0.03 sec
    Start 4: arrayt.test
4/4 Test #4: arrayt.test .......   Passed    0.03 sec

100% tests passed, 0 tests failed out of 4

Total Test time (real) =   0.10 sec
```

Также условием для запуска тестов является указание конфигурации сборки (Release или Debug) иначе можно получить сообщение об ошибке, представленное ниже:

```
C:\spring_oop\filatov_i_a\build\prj.test>ctest

Test project C:/spring_oop/filatov_i_a/build/prj.test

    Start 1: rational.test

Test not available without configuration.  (Missing "-C <config>"?)

1/4 Test #1: rational.test .....***Not Run   0.00 sec
```

Опции CTest

Исполняемый файл CTest включает в себя несколько удобных опций командной строки, которые немного упрощают тестирование. Начнем с рассмотрения опций, которые обычно используются из командной строки.

```
-R <регулярное выражение> Запускайте тесты, соответствующие регулярному
выражению, в качестве регулярного выражения выходят название тестов
-E <регулярное выражение> Исключайте тесты, соответствующие регулярному
выражению
-L <регулярное выражение> Запускайте тесты с метками, соответствующими
регулярному выражению
-C <config> Выберите конфигурацию для тестирования
-V,--verbose Включите подробный вывод тестов.
```

Пример использования некоторых из этих опций

1. -R rational.test

```
c:\spring_oop\filatov_i_a\build\prj.test>ctest -C Release -R rational.test
Test project C:/spring_oop/filatov_i_a/build/prj.test
```

Start 1: rational.test
1/1 Test #1: rational.test ..... Passed 0.01 sec
100% tests passed, 0 tests failed out of 1

2. -E rational_test

c:\spring_oop\filatov_i_a\build\prj.test>ctest -C Release -E rational.test

Test project C:/spring_oop/filatov_i_a/build/prj.test

Start 1: arrayd.test

1/3 Test #1: arrayd.test ..... Passed 0.01 sec

Start 2: matrixs.test

2/3 Test #2: matrixs.test ..... Passed 0.01 sec

Start 3: arrayt.test

3/3 Test #3: arrayt.test ..... Passed 0.01 sec

100% tests passed, 0 tests failed out of 3

Дополнительные свойства CTest

Установка свойств для CTest выполняется с помощью функции `set_tests_properties()`.
Эта функция позволяет задать различные свойства для тестов, такие как время ожидания, ожидаемый вывод и другие параметры.

Вот пример использования этой функции:

```
set_property(TEST test_name
             PROPERTY prop1 value1 value2 ...)
```

Эта команда установит дополнительные свойства для указанных тестов. Примерами свойств являются:

ENVIRONMENT

Определяет переменные среды, которые должны быть определены для выполнения теста. Если задано значение списка переменных среды и значений формы MYVAR=value, эти переменные среды будут определены во время выполнения теста. После завершения тестирования среда восстанавливается до своего предыдущего состояния.

LABELS

Определяет список текстовых меток, связанных с тестом. Эти метки можно использовать для группировки тестов на основе того, что они тестируют. Например, вы могли бы добавить метку MPI ко всем тестам, использующим код MPI.

WILL_FAIL

Если для этого параметра установлено значение true, то тест пройдет успешно, если код возврата не равен 0, и завершится неудачей, если это так. Это отменяет третье условие требований к прохождению.

PASS_REGULAR_EXPRESSION

Если указана эта опция, то выходные данные теста проверяются на соответствие предоставленному регулярному выражению (также может быть передан список регулярных выражений). Если ни одно из регулярных выражений не совпадает, то тест завершится неудачей. Если хотя бы один из них совпадает, то тест будет пройден.

FAIL_REGULAR_EXPRESSION

Если указана эта опция, то выходные данные теста проверяются на соответствие предоставленному регулярному выражению (также может быть передан список регулярных выражений). Если ни одно из регулярных выражений не совпадает, то тест будет пройден. Если хотя бы один из них совпадает, то тест завершится неудачей.

Пример использования **PASS_REGULAR_EXPRESSION** и **FAIL_REGULAR_EXPRESSION**

```
add_test (NAME outputTest COMMAND outputTest)

set (passRegex "^Test passed" "^All ok")
set (failRegex "Error" "Fail")

set_property (TEST outputTest
              PROPERTY PASS_REGULAR_EXPRESSION "${passRegex}")
set_property (TEST outputTest
              PROPERTY FAIL_REGULAR_EXPRESSION "${failRegex}")
```

Используемая литература

1. Testing With CMake and CTest — Mastering CMake: https://cmake.org/cmake/help/book/mastering-cmake/chapter/Testing%20With%20CMake%20and%20CTest.html
2. ctest(1) — CMake 3.28.1 Documentation: https://cmake.org/cmake/help/latest/manual/ctest.1.html#description