

Федеральное государственное автономное  
образовательное учреждение высшего  
образования  
«Национальный исследовательский университет  
ИТМО»

Факультет Информационных технологий и программирования

Работа: Лабораторная работа №6

Выполнили:

Ерёмин Владимир Ильич

Кириленко Илья Андреевич

Проверил: Ельников Сергей Сергеевич

Санкт-Петербург

2022 г.

## Лабораторная работа 6: CI/CD.

### 1. Вступление.

**CI/CD** — это комбинация непрерывной интеграции (*continuous integration*) и непрерывного развертывания (*continuous delivery* или *continuous deployment*) программного обеспечения в процессе разработки.

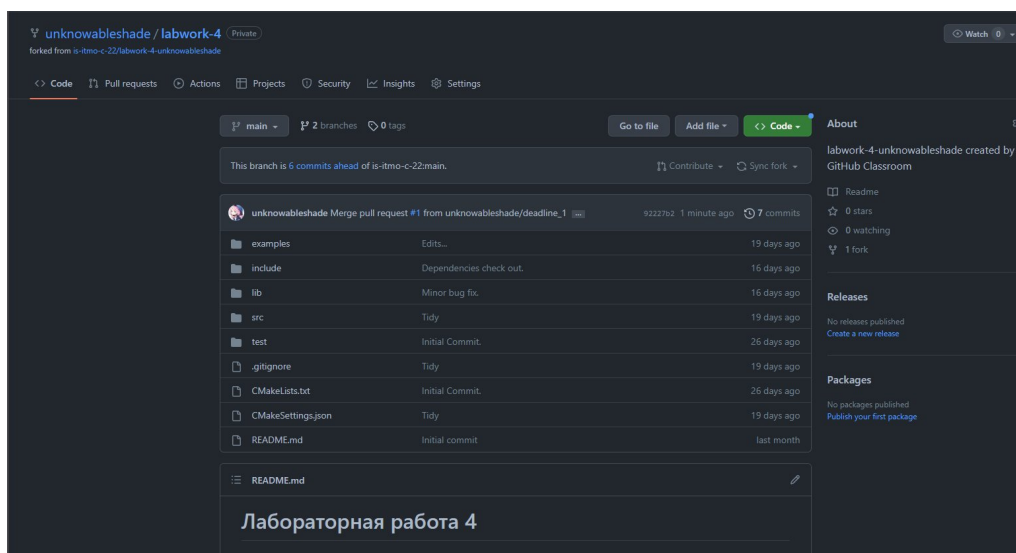
**CI/CD** объединяет разработку, тестирование и развертывание приложений.

**GitHub Actions** - это, CI/CD платформа реализующая автоматическую сборку, тестирование и поставку прямо внутри Github репозитория. Может использоваться как для личных, так и для коммерческих проектов. Благодаря такому подходу, можно безопасно делать вклад в проект, а Github actions в случае неудачи на любом этапе, может прервать дальнейшую сборку.

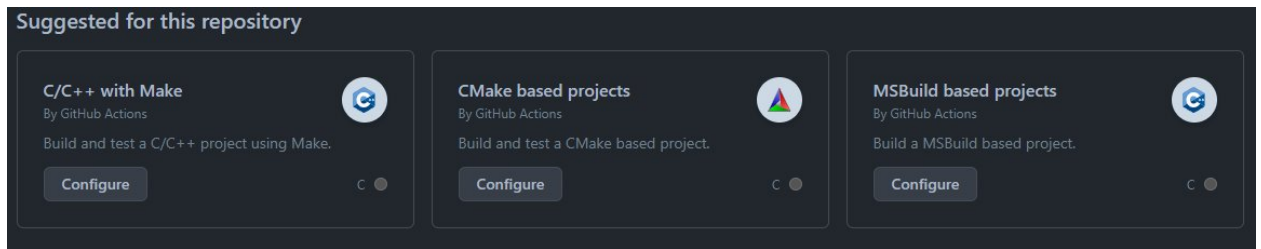
### 2. Проект и обзор Github Actions.

*Имеем проект с основ программирования написанный на C++*

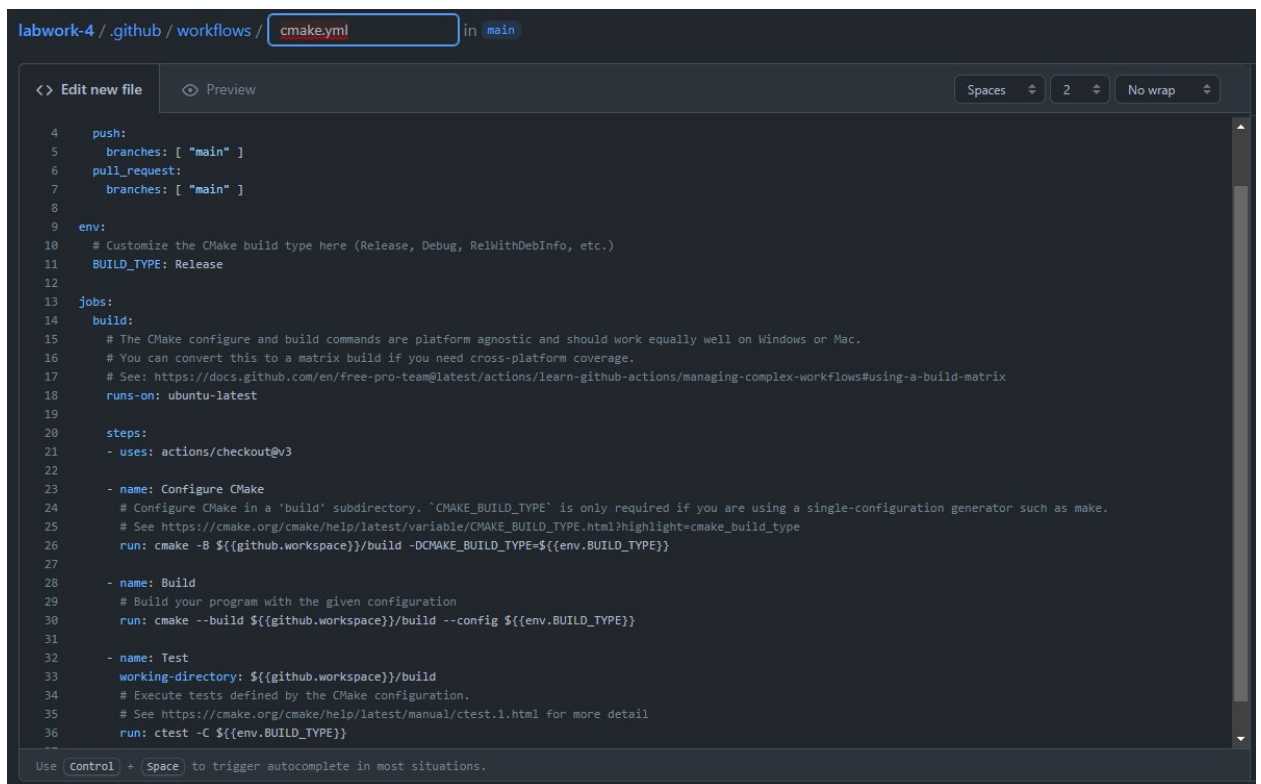
Цель: добавить в репозиторий Github Actions для автоматической сборки, тестирования и генерации исполняемого файла.



Во вкладке “Actions” Github предлагает нам уже готовые шаблоны. Более того, основываясь на том, что содержится в рассматриваемом репозитории он может предложить шаблоны под стать нашим целям.



*Я использовал CMake для работы, поэтому будем использовать его и здесь тоже.*



Actions построены на YAML файлах, которые описывают окружение, шаги... Чем-то схоже с DockerFile.

Шаблон для CMake говорит делать сборку и тестирование при каждом коммите/pull-request в ветку Main. Для генерации артефактов (в нашем случае исполняемый файл) воспользуемся тем, что предлагает нам сообщество. К уже готовому в репозитории Actions можно добавлять и другие созданные сообществом. ([softprops/action-gh-release](https://github.com/softprops/action-gh-release)) будет отвечать за выгрузку артефактов в Release.

Можно отфильтровать запуск Actions по тегам, если добавить данные строки:

```

on:
  push:
    branches: [ "main" ]
    tags:
      - "v*.*.*"

  pull_request:
    branches: [ "main" ]
    tags:
      - "v*.*.*"

```

Добавим Release в YAML файл:

```

- name: Release
  uses: softprops/action-gh-release@v1
  with:
    files: ${github.workspace}}/build/src/hamard

```

После добавляем этот файл, по указаниям самого Github, и делаем коммит с тэгом.

Автоматически запускается процесс, описанный в файле:

The screenshot shows the GitHub Actions interface for a workflow named 'CMake #3'. The 'build' job is selected, and the 'Test' step is expanded, showing the following output:

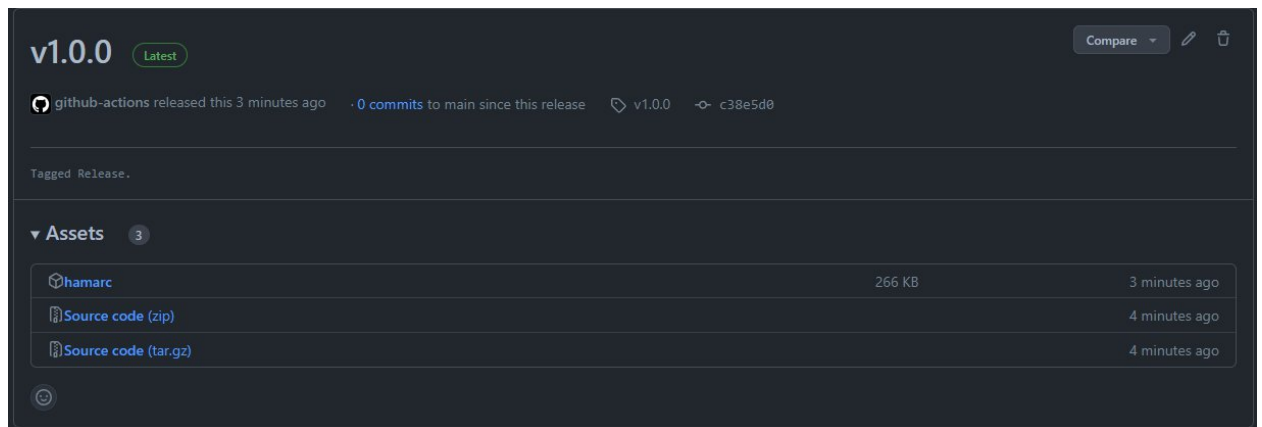
```

1 ▶ Run ctest -C Release
6 Test project /home/runner/work/labwork-4/labwork-4/build
7   Start 1: EncoderTestSuite.EncodingTest
8 1/4 Test #1: EncoderTestSuite.EncodingTest .... Passed    0.00 sec
9   Start 2: EncoderTestSuite.DecodingTest
10 2/4 Test #2: EncoderTestSuite.DecodingTest .... Passed    0.00 sec
11  Start 3: EncoderTestSuite.HealingTest
12 3/4 Test #3: EncoderTestSuite.HealingTest ..... Passed    0.00 sec
13  Start 4: EncoderTestSuite.IOTEST
14 4/4 Test #4: EncoderTestSuite.IOTEST ..... Passed    0.01 sec
15
16 100% tests passed, 0 tests failed out of 4
17
18 Total Test time (real) =  0.05 sec

```

Below the 'Test' step, the 'Release' step is also visible, indicating it has been completed successfully.

Ошибок не случилось - наблюдаем наш исполняемый файл во вкладке Release:



Дополнительно заметим, что в данном случае выполнялась сборка только под Linux. Однако с возможностями CMake и Actions можно спокойно собирать под разные ОС и тестировать там ваш проект.