Running Tests

Overview

Tests for the project can be executed using the `run_tests.py` script provided in the `unit_tests` directory. This script facilitates the testing process by incorporating additional unit tests, compiling the project with the necessary flags, running the tests, and generating a detailed code coverage report using lcov.

Running the Tests

To run the tests, execute the following command in the terminal:

```
python3 run_tests.py
```

This command initiates a sequence of actions to ensure comprehensive testing of the project.

Test Execution Process

1. **Adding Additional Unit Tests:**
    - The script copies two additional unit test files, namely `MyUnitTests1.cpp` and `MyUnitTests2.cpp`, into the `MrSnowman_tests` directory.
2. **Compiling MrSnowman:**
    - Changes are applied to the `MrSnowman.pro` file to include necessary flags for code coverage (`-fprofile-arcs -ftest-coverage`).
    - The project is cleaned, reconfigured using qmake, and then rebuilt using make.
3. **Running MrSnowman_tests:**
    - Similar modifications are made to the `MrSnowman_tests.pro` file to enable code coverage (`--coverage`).
    - The project is cleaned, reconfigured, and rebuilt.
    - The executable `MrSnowman_tests` is then executed.
4. **Generating Code Coverage Report:**
    - The script uses the `geninfo` tool to generate coverage information (`coverage.info`) and the `genhtml` tool to produce an lcov report.
    - The resulting report is stored in the `coverage-html` directory and is accessible via the `index.html` file.
5. **Cleaning Up:**
    - Once the tests are run and the coverage report has been generated, the script removes the additional unit test files.

Viewing Code Coverage Report

After running the tests, you can view the code coverage report by opening the `index.html` file located in the `coverage-html` directory in your web browser.