Forums Questionnaires Resources

Respondent: **Hannes Karppila** Submitted on: Thursday, 30 November 2017, 3:46 PM

Mid-term review of C++ project

Name of the project group evaluated

sim-city-1

C1.1: The implementation corresponds to the selected topic and scope. The extent of project is large enough to accommodate work for everyone (2 p)

Scope seems to be large enough that everyone should have work. However, not all parts have been implemented yet, as expected in the midterm review.

C1.2: The class structure, information hiding and modularization is appropriate, and it is explained and justified in documentation. The file structure corresponds to the class structure (2 p)

Class structure seems clear, and the file structure closely matches the class structure. Information hiding is also good, and no unnecessary properties were exposed.

Documentation doesn't exist yet, not even in the README.md file. At the very least instructions to compile and run the code would be nice.

We could not access the project plan document, so we cannot tell how well the current implementation follows the project plan.

C1.3: Use of at least one external library (in addition to C++ standard library). Comment the appropriateness of libraries and their use. (2 p)

SFML is used, and in most places correctly. However, in the main function when loading textures, if texture loading fails, the program does not exit, and just prints a warning, leaving the texture empty. This is probably not the desired functionality.

C2.1: Git is used appropriately (e.g., commits are logical and frequent enough, commit logs are descriptive) (2 p)

Cache files are stored in Git, which should probably not be done. Commit message style is somewhat inconsistent (capitalization, tense). Some commits are lacking a proper commit message (e.g. "fix"). Also there are some commits modifying multiple unrelated things.

Commit frequency seems good, and with a few exceptions commits are logical and clear.

C2.2: Make or Cmake (recommended) is used appropriately. The software should build easily using these tools without additional tricks. Nevertheless, instructions for building the project should be provided (1 p)

Make is used. Makefile is quite simple, and follows best practices. As mentioned before, instructions to build the project are missing.

I ran make in the project root, and the C++ compiler reported multiple errors and warnings. Most errors were about "implicit instantiation of undefined template". (System: Mac OS X 10.11.6, Apple LLVM version 8.0.0 (clang-800.0.42.1))

C2.3: Work is distributed and organised well, everyone has a relevant role that matches his/her skills and contributes project (the distribution of roles needs to be described) (1 p)

Based on the commit history everyone seems to have work. However, we do not have information about everyones skills, but based on the meeting, everything seems to be going well.

C2.4: Issue tracker is used appropriately to assign new features and bug fixes (1 p)

Issue tracker is used, and some issues are mentioned in the commit messages.

Completed issues were also assigned appropriately.

C2.5: Testing and quality assurance is appropriately done and documented. There should be a systematic method to ensure functionality (unit tests, valgrind for memory safety, separate test software and/or something else.) (1 p)

There was no documentation about testing. Testing is not yet implemented, but there is an open issue for that.

C3.1: C++ containers are used appropriately (including appropriate use of iterators), and justified (e.g., why certain type of container over another) (2 p)

Container usage seems good, and proper containers are used e.g. for the map.

C3.2: Smart pointers are used in memory management, describe how (1 p)

Vertexes use shared pointers, and based on the comments in the code, transition to smart pointers is planned.

C3.3: C++ exception handling is used appropriately, describe how (1 p)

Exceptions are not used.

C3.4: Rule of three / rule of five is followed, describe how (1 p)

There is no code specific to the rules of three/five in the project.

C3.5: Dynamic binding and virtual classes/functions are used, describe how (1 p)

Virtual functions are used e.g. for Vehicle::getType. However, this seems somewhat forced, as duck typing should not be used in C++.

[illegible]

Other comments and feedback to the evaluated project group.

Keep up the good work. Code looks otherwise good, but some indentations were a byte or two off.

If you did this review together with (some of) your group members, list the names of the group members here. Everyone needs to turn in a review, either separately or as a group.

We all reviewed this together. Names: Iiro, Aleksanteri, Hannes, Petteri.

English (en)
Suomi (fi)
Svenska (sv)