CAS 741, CES 741 (Development of Scientific Computing Software)

Fall 2017

25 Discussion

Dr. Spencer Smith

Faculty of Engineering, McMaster University

December 6, 2017

Discussions

- Administrative details
- Coding style
- Test report
- Questions?
- Discussion

Administrative Details

- Course evaluation
  - Nov 23 to Dec 7
  - https://evals.mcmaster.ca
- GitHub issues for colleagues
  - Assigned 1 colleague (see `Repos.xlsx` in repo)
  - Provide at least 5 issues on their MIS
  - Grading as before
  - Due by Tuesday, Dec 5, 11:59 pm
- GitHub issues for implementation
  - Not required as part of course
  - Will assign names anyway
- Source in src folder
- Added an INSTALL.txt file to BlankProjectTemplate

Administrative Details: Deadlines

Final Documentation	Dec 18, 11:59 pm

Coding Style

- Having a coding standard is more important than which standard you use
- Examples
  - Google guides
    - Python
    - C++
    - Java
  - Mozilla Developer Network
  - NASA C Style Guide
- Your decisions on style may evolve over the project
- Important to be consistent

Installability and Learnability

- You can test this
- Ask a colleague to install your software
- Run it on a virtual machine, like VirtualBox
- Use a “light weight” VM like docker
- Include installation instructions (INSTALL.txt)
- Include instructions so that someone else can run your tests cases

Final Documentation: Test Report

- Completing what you proposed in your test plan
- You do not need to repeat material from your test plan - the emphasis is not on the rational for test case selection, but on the results.
- If your test plan does not match what you are now testing, edit your test plan to “fake” a rational design process.

Test Report Continued

- Provide specific test cases
- Summarize your test results
  - Test case name
  - Initial state
  - Input
  - Expected results
  - Whether actual output matched expected
- Summarize and explain usability tests - quantify the results
- Performance tests - quantify the results
- Stress tests
- Robustness tests
- After quantification of nonfunctional tests, explain significance of results

Test Report Continued

- In cases where there are many similar tests
  - Summarize the results
  - If the expected result is obvious, you might not need to state it
  - Give an example test case, and explain how similar tests were constructed
  - If the tests were random, describe how they were selected, and how many, but not all of the details
  - Use graphs and tables
  - You need enough information that
    - Someone could reproduce your tests
    - Your test results are convincing
    - Evidence that you have used testing to improve the quality of your project

Test Report Continued

- Summarize changes made in response to test results
- Explain your automated testing set-up (if require more detail than from the test plan)
- Provide traceability to requirements (if not in test plan)
- Provide traceability to modules (if not in test plan)
- Make sure you show test results for “bad/abnormal” input

Sample Test Report Documents

- Screenholders
- 2D Physics Based Game (Uses doxygen)
- Follow given template
- Examples are not perfect
- Examples are intended to give you ideas, not to be strictly followed
- You can modify/extend the test report template as appropriate

Questions?

- Questions Final documentation?

Discussion

- Thoughts on documentation
  - SRS
  - VnV Plan
  - MG
  - MIS
- Thoughts on technology
  - Git
  - GitHub
  - LaTeX
  - Make
  - Your programming language

Discussion: Course Content

- What ideas from the course will you continue to use?
- Thoughts on
  - Drasil
  - Assurance cases
- Other thoughts?

Discussion: Course Structure

- What can be done to improve the course on its next iteration?
- Increase number of reviewers for GitHub issue creation?
- How to get more discussion in class?