EECS 4313 Assignment 2
- Black-box Testing with JUnit
Due: Noon, Monday, February 29, 2016

1. Assignment goals

The purpose of this assignment is to give you experience in applying the functional testing techniques we discussed in class, as well as creating automated test code with JUnit. Your task will be to create a test suite in JUnit for a system of your choice, produce bug reports (if any), and submit a written report describing your testing.

2. Getting started

1. Select an open source system to test (see Appendix 1 for the detailed requirement for such system). Download the code and install it in into Eclipse. While it is not necessary for this assignment to be able to run the application code through Eclipse, it is recommended that you do.
2. Select 3 methods from the system you are testing. The methods must be chosen so that you can demonstrate the application of the three testing techniques discussed in class: boundary value testing, equivalence class testing, and decision table testing. It is okay to use a combination of testing techniques, but each technique must be used at least once.
3. Define a detailed specification for every selected method based on your understanding of its purpose. Following is an example of a specification for a Java class from a sample system:

   Sample Specification

   public generateGraph (int numberOfNode, int maxDegree,boolean isConnected, String outputFileName)

   This method generates a random graph and outputs it in the designated file in XML format.
   - The first argument is the number of nodes that the generated graph should have.
   - The second argument is the maximum degree for any node.
   - If the boolean argument is true, then the produced graph must be connected in the non-directed sense. If the boolean argument is false, then the graph may or may not be connected.
   - Finally, the String argument is the name of the file that will contain the produced output.

   If the number of nodes or the maximum degree is less than 1, then a NegativeInputException is thrown.

4. Reporting bugs or proposing enhancements

For this assignment, you should create a written report (a2.pdf). The report must include the following information:

- Specification of the selected Java methods.
- Justification of the testing technique chosen, i.e., why is it appropriate for this method.
- Description of your application of the three testing strategies. Be clear about which test cases you implemented.

- Evaluation of the test cases derived by the testing technique. If you had to complement the derived test cases with special value testing, describe that as well. The marker will not read your code in order to see what you tested. You have to describe it.
- Attaching bug reports if bugs are discovered using your testing methods. You should use the same bug report format as in Assignment 1. **Do not** file these bug reports to the project's bug report system.

Presenting your thought processes in English is an important skill for a software engineer. If you have trouble in writing English, ask someone to proof-read and correct your writing. You can also consult the English as a Second Language Open Learning Centre (http://www.yorku.ca/eslolc) at the York University.

5. Submission

You can work with at most **two** partners (a team thus has a maximum of 3 members). Submit a PDF of your report (a2.pdf) and the test code package (i.e., eecs4313a2) electronically. Your report must include the names and the student numbers of all the team members. **Only 1 submission per group, please!** You should also bring a hardcopy of the report before the class.

Navigate to the directory where it contains the code package and the report. Use the following commands to submit:

- submit 4313 a2 eecs4313a2
- submit 4313 a2 a2.pdf

Appendix A – Requirements for the Studied Open Source Systems

You may pick another system to test in this assignment. However, this system needs to satisfy the following criteria:

1. The system must be an open source system written in Java;
2. The source code of the system should be publically available to download;
3. The system must be under active development. In other words, the system needs to contain code commits or software releases in the last 12 months;
4. The system is not a toy system and must be reasonably complex. One way to judge is to count the lines of code (LOC) for this system. The LOC should be in similar range or bigger than the suggested system. SLOCCount (http://www.dwheeler.com/sloccount/) is a tool which can automatically output the lines of code for the selected system;
5. The system should contain (some sort of) documented specifications. The version of the specification can be slightly older than the version to be tested on.

Appendix B – Importing and Building a Maven Project

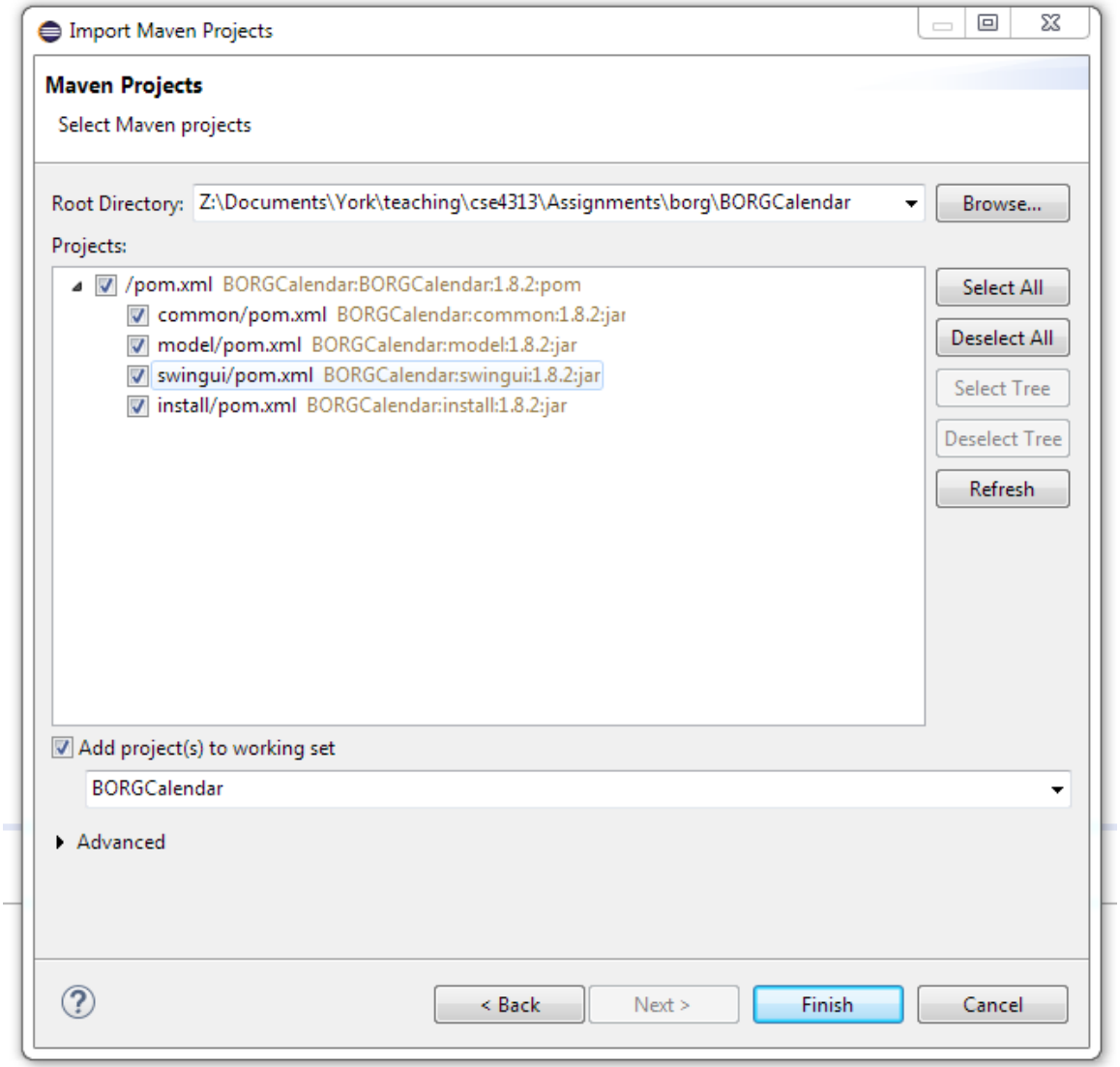
Here we describe the process of importing and building the Borg calendar into Eclipse. You may also find this link useful (https://github.com/mikeberger/borg_calendar/wiki/Building). The described process below should be similar for other projects which use Maven as their build tool.

1. Install the Maven plugin

You need to install the Maven plugin in Eclipse if you have not done so. Open the Eclipse IDE, click the Help menu -> Install new software -> All Available site -> Enter the following URL http://download.eclipse.org/releases/mars, and enter "m2e" as the filtered text -> pick the items to install. Once done, close and re-open Eclipse.

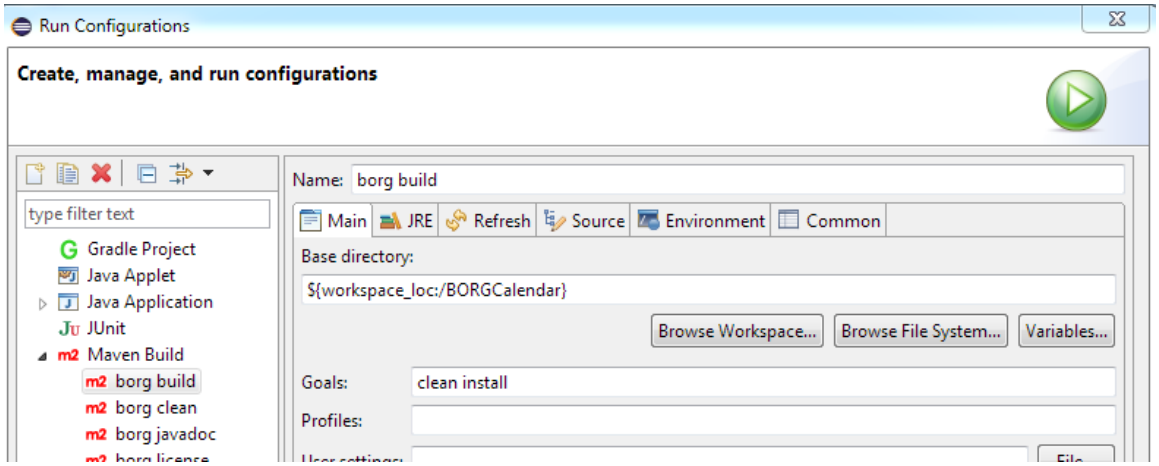
2. Import the project source code

Once you have installed the Maven plugin, the next step is to import the Borg calendar source code into the Eclipse IDE. Click the File menu -> Import -> Maven -> Existing Maven Projects -> Pick the source code directory which contains the Borg source code -> Finish.



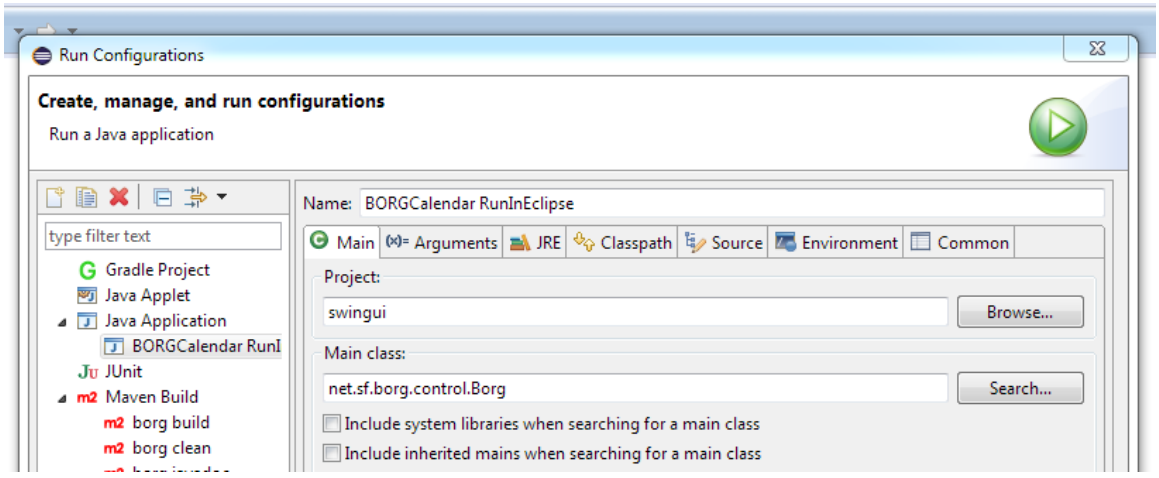
3. Build the Borg Calendar Application

After you have successfully imported the source code of the Borg application into Eclipse, you need to build the project. Click the Run menu -> Run Configurations -> Double click Maven build, and you will see a dialogue box, fill in the information as shown below:



4. Launch Borg inside Eclipse

Click the Run menu -> Run configuration -> double click Java Application -> name it as "BORGCalendar RunInEclipse -> select the swingui project -> under Main class say "search" -> pick "Borg - net.sf.borg.control" and say OK.



Once that is done, then click the Run button. If it asks whether you want to proceed with error, just click the Proceed button. You should see the Borg application launched.