Notes durring project development:

2016/12/31

Working on the MineSweeperRunner class, looking to get a system of drivers and stubs set up

Will probably use different batch files to run different groups of drivers and stubs, will keep all the drivers and stubs in the same package, so I can import them all at once and swap them out easily. Maybe I can have something in the compiler that changes them, as to make testing more automated. Also need to get my hands on that junit or something as to create testing classes.

Before all that though, need to get a basic setup made.

What the runner class is going to do, is drive the instance of minesweeper

Primary functions (taken from documentation)

* Prompts the user for info
  + This should be part of the gui file.
  + Uses MSData to contain it all
* Run the instance of MineSweeper
* Displays the finished screen using the GUI
  + A “current status” gameplay board will have to be taken from MineSweeper instance. Showing which bombs have been caught. This will be a map for the gui to deal with.
* A timer for the game will be run in MineSweeper – which will be returned and put in the end screen with the gui
* This time will be added to a file, with “highScores”
* The highScores will be a file in the same folder as MineSweeperRunner, which will be updated at the end of a game. The top 10(?) scores will be kept. The top 3 names will be presented (or maybe top 10 idk) when the user is prompted for data. This will be an array of strings that will be passed into the Gui method for user prompting.
* Repeats until the user says no. which will be part of the gui thing. Just a while loop I guess.

Noticed an issue, I don’t know how im going to be able to make the gui interchangable seemlessly. I think I could create something that navigates through a set of classes. But I wont bother with that for now. Needs stuff with classloader or reflections (neither of which I currently understand). Until then il make everything that isnt Runner generic, and just have Runner set the gui manually.

To do next

* Need to write MSData – will add stuff to it as the project continues.
* Need to write drivers for gui to have
  + promptUser
  + displayResult
* Need to write drivers for MineSweeperInstance to include
  + A constructor
  + Run

A way to make these “drivers” work, might be to to just have different batch files with different class paths to different driver source folders. These will basically be the same as the true source files, but will just be in different folders. This allows for automated testing of different files.

This will eventually be really useful for pre-determined games, and testing that their functionality remains consistent. Since the “gui” will just be a series of inputs, it will also just have pauses inbetween its inputs, which can surround information to be printed to the console. Or something. That’s for later.

Anyway, I need to finish the packages, and I need to create the drivers.

It tooks like the way to get the gui to “change” is to have a folder where the java files will drop into. To switch the gui you just switch the content in that folder. So all gui implementations will have identical file names, as well as an interface.

Signing off.

2017/01/01

Got all the file stuff lined up properly, turns out with class path you can actually just attach the “nozzle” to each file location and just compile it. I put all the class files into a class folder, because I heard that is good convention. I removed all the packages and imports, as they were useless. All the file structure is laid out, so I can move onto testing. The intial basic drivers seem functional. I will work on a better plan for the sequence of solving each module, including making test files and stuff in separate folders in their respecive places. The current skeletal driver is full functional and ready to be implemented with more specific drivers. Each update of a driver will take the OLD driver and put it into a subfolder, making the ORGINAL unchanged.

Next goals:

* Get a new plan created with outline of which modules to create
* Create test files to run said modules – do the first one initially
* Get the junit

Pushing to git.

Signing off.

2017/01/02

Looking to start with a plan of the modules to create -> maybe create a detailed plan of how I want to actually approach this whole project. Need to figure out where I can dissect stuff from my old code, and where I need to create new material.

TODO: Get MSMap made with a method that will read in a test map from a file. This will allow it to act as a driver early on to just have standard maps. We can worry about proper ones later.

Create 6 maps, easy medium hard and 3 custum.

Map selection has to be done through the console at compiling time, so have that as a method call to instance. Include 2 variables – testingMap boolean (if we are running a test map) set to true by doing a try catch with the call to index, if its an exception its set to false. Otherwise true. Then a variable called testMapName which is the paramater that is used.

Once that is done, we will have a relatively reliable driver for MSMap. After that, start work on MineSweeperInstance, getting the constructor set up properly and having run work.

Run calls the gui, so we will need to think of a way to make the gui have a set path of moves to make, which it will be its moveset. Maybe have it read in while its updating MSMap test, and then have that be put into instance, and have that be what it uses. Might need an incrementer to control flow, which can be used in a for or while loop in the driver for the gui.

This should be enough to have mineSweeperInstance fully running, once those are set up, we will run those against mineSweeperRunner. When those are done we can start substituting things in for proper methods.

End of the day. I got test batch files made for MSMap and MSData, and I reorganized the file structure in regards to testing. Java files will be kept in their respective files, with the testFile sitting right near them. However, the batch files and test class files and test text files will all be kept in their own subdirectory in a separate file structure. Those bat files are set up, and will allow for relatively easy testing and are reletively easy to set up.

Next set of things to do includes:

* Finishing testing MSMap
  + Need to make the 6 test files, decide on their lengths, and have them input. NOTE: the first line of the file is ignored, it is ment to be the data input for the file, read in during testing.
* Finish building MineSweeperInstance
  + That means il have to make the simulated gameplay in GUI, which will PROBABALY be a second line in each file… so that another line to be skipped NOTE: add the second skip line later.

Aim for those for now, think of where to go after that

Pushing to git.

Signing off.

2017/01/03

Started late today, cold as balls outside.

Things I gotta do today:

* Finish testing MSMap (as stated above)
* Finish building and testing MineSweeperInstance
* Need to change CellHold so that instead of a true/false boolean it is 1234 (0123?) where 1 refers to unvisited, 2 refers to visited, 3 refers to flagged as a bomb, and 4 refers to uncertain.
* Change EVERYTHING so it reflects the changes made in CellHold.

So I changed cellHolds varibale to cellStatus, and it doesn’t seem to be used anywhere, I spose itl pop up In further testing.

Redid the testing MSMap using @before and @after. Still need to create the testing files.

Branching into 2017/01/04

Ironed out the MSMap to work with the new implementation. Need to finish creating files

Important to NOT include filename in the test files, and make sure there is a space after the last number

Next thing to do – make LEGIT tests. Maybe cheat and use old code to generate random sets…

Remember – bombs are 9, empty spaces are 0 and 1-8 are the possible numbers.

Decide on the easy, medium, hard standards.

Easy – 9 9 10

Medium – 16 16 40

Hard – 25 25 90

Custum will probably just be for when its actually randomly made.

3 tests for both easy medium and hard all pass the test

Also included a maping of all the coordinates for bombs at the top of each file, for testing purposes with the gui.

Took a few days to get here. Should NOT have taken this long.

Need to:

* go and finish building and testing MineSweeperInstance
* Need to get the driver for gui set up
* maybe write the code for the actualy MSMap method. Its not like its hard….
* Keep stuff moving faster. Maybe I need to stay up late tonight. (didn’t happen).

Anyway this is 2017/01/05, signing off for now. Pushing to git. New page new notes.

2017/01/06

To do:

* Finish building MineSweeperInstance
* Finish testing MineSweeperInstance

So, kinda got caught up with school stuff, Tis 2017/01/07 so, gonna see how much I can do right now. Set up abit of testing stuff in MineSweeperInstance and MSMap (grabbing arrays and whatnot), looking to get started on the gui stuff.

Not a lot of focus right now, might come back to it later today. The struggle is real with keeping momentum.

2017/01/08

MineSweeperInstance looks decently set up, need to finish check point (should be trivial) and update map. Definetly want to have a tester set up for updateMap, as it is relatively complex.

Tester for updateMap should include:

* Testing clicking 1-8
* Testing clicking a bomb (9)
* Testing clicking white space (0)
  + Might be interesting to try having a complex white space setup made, so there is a large portion of the board made of whitespace, and each spot of that white space (that’s part of the entire group) has to be clicked individually and the results of each one has to match a baseline expectation. Should be good to stress test
  + Will need coordinates of all whitespace, which shouldn’t be to hard (write a program to go through a file and print all the coordinates of the 0s
* Test errors, if the spot clicked was WRONG what does it do
  + Will have to add an incorrect input in the click path that the game will create

The driver for the gui is done, it reads in a file that is the same as the target file, except it has pr.txt instead of .txt. I forgot what it stands for but it made sense at the time. Program route maybe?

Main steps to do next are to code the update map section (realistically, should be really easy, codes already done) and thoroughly test it. Then write run(), and thoroughly test it. Hence the drivers for gui. After that it would probably make sense to implement the actual MSMap class, which should also be quick. Once those are both done, moving on to gui would be the next sensible place to go, I only say to do the MSMap part because I don’t think it will be all that time consuming. I have a tester set up for gui and its funcioning with the driver. There will probably be massively extensive tests needed for the gui. Hopefully I can get them mostly ironed out.

Update map is fully tested and works fine.

**Really important**. Any pr.txt files MUST be in the order of place (1) max (how many clicks) then height length coordinates seperated by a space. MUST be like this or wont run.

Noticed that x and y coordinates appear reversed. Very bad. Probably just going to paint the x axis as the height and y axis as the length.

Next thing to work on is the run method, need to implement test files so that I can use the gui driver and make sure they work as intended. Most of the framework is already set up for that though, so there shouldn’t be too much left to do.

Gonna push this to git now. Will probably work on this more later today

Its 2017/01/13 (took awhile… was working on other project for a few days).

Signing off.

2017/01/13

So I decided to invest Friday into writing absurd amounts of tests.

Ive made a pr.txt file for every test file (all 9) and ive made tests that will use them all

I need to fill the pr.txt files with the moves to be made

Ive made it so that the results of the test will be printed out in a form of a new array being made. That way I can copy and paste the test results (after inspection of course, to be sure they did as expected). This saves immesurable amounts of time, writing it by hand. I know its technically cheating, but it will make this a million times easier.

To write the pr.txt files, I use this format:

1 4 0 0 1 1 1 1 2 2 1 3 2 1

It can be read as this:

1 4 001 111 221 321

The first one is the incrementer, starting at 1.

The second is the max count, which is equal to the number of truplets after it

The rest of the numbers are coordinates in which the user has clicked. The first two are the height and length coordinates, and the last one is the type of action taking place.

Easy1pr, Medium1pr, Hard1pr will just be all the bomb spots, followed by 1, so we can test if winning works. Could throw in a few White spaces aswell.

The rest will be turns based on difficulty of the game

Easy: 6 moves + lose

Medium: 10 moves + lose

Hard: 15 moves + lose

I make sure each of these is a loss, as to make it break out of the loop

The list of the entities will be on the next page. I think im gonna hold I here, the test writing stuff will probably take around 2 hours to write. These tests need to be extensive, since they are the groundwork for testing the other guis, whenever they are written

Pushing to git

Signing off

2017/01/14

Finished the tests for easy medium and hard 1. Which invovled bomb placements

gonna push to git, then add the rest of the tests later.

Signing off.

2017/01/14

Starting up the tests for MineSweeperInstance, looking to finish as many as possible in an hour.

Note – bomb numbers in the test files appear to be off…

Just realized all my tests are wrong. The recursive part is suppose to reveal ALL the squares connected to the blank space that arnt bombs, so that includes numbers……

I have to fix all the tests now… this could take awhile

I’ve fixed all the tests, I have the last 2 hard ones to complete. Those will take awhile, I need to do atleast like 25-30 moves for each… so that will take probably half an hour to an hour each.

Added 3 new tests, all custom sizes, took ages since I had to make each by hand.

Done testing run in MineSweeperInstance.

Going to quickly test

Got it working all together, everything passes all the tests.

Just one issue…….

There appears to be a lot of errors in the tests. The check runner, which happens at every single turn, is telling me that there are lots of illegal moves happening. This means that I need to go through every single test again and check for all the possible places of errors.

Well, shit.

Deal with it later. Its luckily reletively easy to fix, there are just so many tests to check.

Just comment out all the tests under the one your looking at, and check inside that one test. Also – change the command loop to NOT allow any error messages that arnt “”, if they are “” then update, otherwise don’t. This will tell me if the map is being incorrectly updated, and I can change the map accordingly (chances are there are plenty of tests that fail)

Anyway, im going to call it here. ALMOST done all the testing, just got a few bugs to wrinkle out.

Going to push to git and call it.

Signing off.

2017/01/16

Fixed the bugs, MineSweeper instance tester fully functioning

2017/01/19

Git has been redone so that it works as intended. All data is saved on git. The commit history is all ruined though.

The tester method for MSInstance is done, it is fully functional now. Next things to get done are

* MSMap, convert the code in it into a driver, and write real code to produce a 2D array
* GUI, work on gui, look into making the abstract class that all gui’s will be an extension of… not sure where to put it… maybe inside the gui folder?
* GUI, plan the tests and figure out what drivers need to be made
  + This includes dissecting the old code (for the useful bits) and improving the current design
  + Need to create drivers to simulate the input – this drivers will ask for which TEST file to use, and SHOULD display the information and the moves it makes. This might be more complex then the actual code, but it will be really useful for testing all guis… make sure it’s a driver that CAN be used for all the guis… or easily switch… so it can be reused… it will allow for the extensive tests done in MSI to be re-evaluated.
* Maybe prototype a version of the second gui, using a picture on the console – need to play around with how best to implement
* Prototype a REALLY basic javafx version of the gui, for proof of concept purposes.