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.