Lab 2: Intro to GUI Programming

Note that a version of this lab that includes a map of trees in Vancouver have been provided! Please read the starter code for more details. Tree inventory data from the city of Vancouver can be found at this URL: https://opendata.vancouver.ca/explore/dataset/street-trees/information/

1 Introduction

It's Graphical User Interface (GUI) time! In this lab we will revisit information about trees in Mississauga, but this time we're also going to be visualizing some of this information using JavaFX. JavaFX is a framework to support programming of GUIs in Java; it contains a variety of classes to generate layouts and widgets and other common GUI element.

To begin, log into MarkUs and click on the assignment called "Lab 2" and pull the files so that you can edit them on your local machine. You should find seven java files in the Lab 2 folder: *MunicipalTree.java, Location.java, TreeInfo.java, TreeReader.java, TreeViewer.java, TreeFilterEventHandler.java and TreeViewerTest.java*. You will also find a data file called *treelist-utm.csv* and an image called *map-utm.png*.

2 Programming Task

The code for this lab visualizes an inventory of trees assembled by the city of Mississauga (which is available at https://data.mississauga.ca/).

This week's code contains several classes. The class TreeReader exists to read input files containing tree information. The class MunicipalTree is used to generate instances of trees. Within each MunicipalTree is a Location, which denotes the tree's latitude and longitude position in the city. A TreeInfo object stores aggregated information about all of the trees in the city and can be used to query statistics about the city trees. Finally, a TreeViewer visually renders the trees on a map, and a TreeEventHandler processes GUI events (like button presses).

You will be completing *TreeInfo.java* to implement:

- The method *getTreeTypes*. This method accepts a list of trees and returns a set containing all of the NAMES of trees (e.g. Ash, Pine, Spruce, etc) in the list. A name of tree may appear many times in the input list, but it should appear only once in the output set.

- The method *getTreeCount*. This method accepts a list of trees and returns a count of all of trees in the list that are of a certain NAME.

When you have completed that, alter *TreeFilterEventHandler.java* to implement:

- The method *handle*. This method is triggered when the user selects a tree name within the TreeViewer and presses the button on the GUI. Rewrite the method to clear existing circles from the TreeViewer and redraw circles that represent trees of the selected name. Note that in order to make this routine work, you'll need to register any circles you draw on an "undo" stack. You can then use this stack to "undo" drawings of circles as is needed! If the user selects the "ALL TREES" option, make sure to render circles for all types of trees on the TreeView. You'll also want to adjust the text area that displays the count of these trees, accordingly.

3 What to Submit

1. TreeEventHandler.java
2. TreeInfo.java

HAVE FUN AND GOOD LUCK!