Test Document of Venn Diagram

EECS2311 W20

GRB #11

SUBMITTED BY: Tianpei Liao
Arda Temel
Ruth
Jeyaprashanth Sivasubramaniam

YORK UNIVERSITY

Table of Contents

Introduction

Venn Diagram application provides user friendly interface with venn diagram that allow users to create and update entries and customize within the venn diagram. The system provides a wide range of features such as adding/removing data entries, updating existing entries' attributes (color, shape, text), drag and drop and categorize the entries within the diagram. We tested to make sure the software did not crash under simple user mistakes and run time. The total testing coverage from the JUnit test cases is around 87.5**%** but there have been several manual tests to ensure the functionality of the user interface is flexible across different OS platforms such as Windows and Mac and also user friendly.

Implemented Classes

Major Classes	Description
Main	Start application, initialize contents, Load Main Scene, Add New Entry, Entry templates, Configure buttons and images
VennController	Configure venn diagram contents and initialize circle and Control mouse events, key events and import multiple new entries
EditController	Edit existing entries and its attributes
GetDataController	Create new entries, changes Sample Preview label when creating new entry
SetCircle	Configure the circle attributes and methods
DraggableText	Configure the draggable text, text's color, setting tooltip, Configure the mouse events

Testing Implementation

The following test cases were derived through examining various attributes of the application.

1. **TC: Main GUI visual presentation and all UI contents in fixed layout**
   Users successfully launch the application without any error and menu item, buttons, venn diagram and its colour contents are properly visible.

   When a user double-clicks the app/executable jar file the system launches the main window within recent amount time. In MainTest class a method called testWidgets () tests the contents in the main window.

2. **TC: Successfully add new entry**
   The user clicks the "plus" sign icon button in the main window then a second window called "Add New Entry "will popup through that user input text, color, and shape of the entry then click the create button. When a user input a new text, color for new entry the samplePreview label also changes accordingly. Created entries will be visible in the main window.

   Using GUI testFx under MainTest class an implemented testGetData () method automates this process and creates a single new entry and is added to the main window. Automated actions are done by FxRobot javafx library.

3. **TC: User does not input text for new entry**
   Users are responsible for input text for new entries. When a user does not input the text and try to create the new entry using the create button or **ENTER** key pressing the system won't allow the creation.

   Using GUI testFx we tested by without writing anything in the text field and clicking the create button and ENTER key press.

4. **TC: Successfully edit the existing entry**
   The user double clicks the existing entry in the main window then a second window called "Edit New Entry" will popup through that user edit the text, color, and shape of the entry then click the edit button. When a user edits the text, color for new entry the samplePreview label also changes accordingly. Edited entries will be updated in the main window.

Using GUI testFx under MainTest class an implemented testEditData () method automates this process and edit the existing selected single new entry and is updated in the main window. Automated actions are done by FxRobot javafx library.

5. **TC: Successfully delete the existing entry**
   User drag and drop the existing entry on top of the delete button icon then the released entry will be deleted from Draggable text array list and removed from main window.

   Using GUI testFx under MainTest class an implemented testDeleteEntry () method automates this process. Create an entry and drag and drop on top of the delete icon button and deleted entry removed from the main window.

6. **TC: User drag existing entry across the windows, over the venn diagram area**
   User drag and drop the existing entry across the screen window and over the venn diagram circle area. When dragged object over the circle area the receiving circle are should lit up and corresponding area's #elements count will be increased. And when dragged object remove from the circle area that area should lit down and corresponding area #elements count will be decreased.

   Using GUI testFx under MainTest class an implemented testDraggedEntry () method automates this process.

7. **TC: Prevent entries overlapping**
   Because users can have more than one entry they possibly drag and drop the existing entry on top of another entry. When overlapping happened the overlapped object auto arranged next to the previous entry.

   Using GUI testFx under MainTest class an implemented testDraggedEntry () method automates this process. When overlapping happened from Venn Controller event handler method triggered and prevented from two objects overlapping. And get the previous entry scene location and arrange the dragged object next to another entry.

8. **TC: Add multiple entries from file import**
   Users import more than one entry by file import from .txt format files. Select import option from file menu and open selected file from file opener. System will read the imported file and input the entries to the main window next to each other with default color and shape.

9. **TC: User dragged the created entry out of window**
   When a user drags and drop the entry outside the screen size the system restricts that entry from dragging outside the window.

   Using GUI testFx under MainTest class a implemented boudaryTest () method automates this process. This test creates four new entry labels and drag each entry using a mouse event outside of the window and check this by all sides such as right, left, up and down.

   We tested this feature completely by manual and using GUI testFx under MainTest class an implemented testImportedData () method automates this process and test up to open file explorer for import.

   ****Note:** Test cases for methods such as captureData () are import data to text file and exportData () is export entries to text file are not implemented but both methods and functions completely tested manually.

Coverage

Main responsible for initializing and loading widgets and components of the main screen window and triggered the related controller classes and methods to launch the main GUI "App.fxml". In the main class we have two methods showAddStage () and showEditStage () both are responsible for launching the "AddNewEntry" and Edit Data" UI. Testing only missed exception handling branches. Hence our overall coverage is about

And in GetDataController class there are methods such as create Text () for creating new entries and changePrev () method for changing the SamplePreview. **91.4%** for the main class.

VennController handles the initialization and loads the controller events and action listeners when the main window launches. Whenever a user interacts with main windows components events handlers are triggered and responded. It has the main method Initializations () which will set the venn diagram components such as circle size, name, color, border thickness etc. and also sets SetCircle class properties.

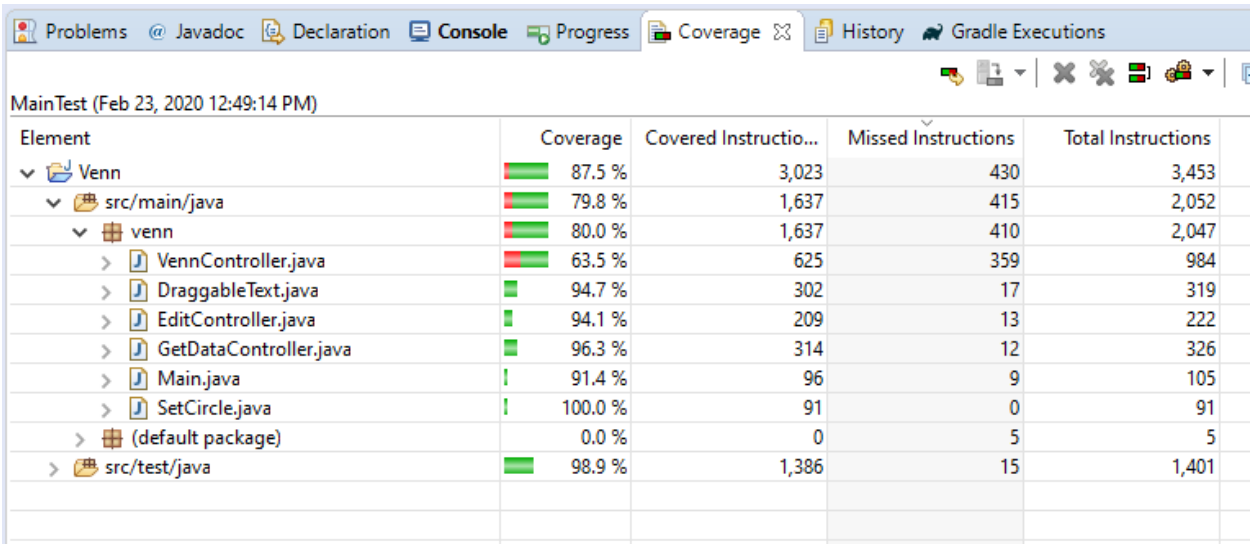
Some of the if statements branch and CaptureData () method which import multiple entries from file import and exportData () which is export existing entries into text file are not covered by our implemented test cases. Still we have obtained **63.5%**

DraggableText is responsible for all draggable text setters and getters methods. Handle all mouse and key events. Tooltip handling. Also handle the properties of draggable text objects such as background color, border radius, text. Prevent entries drag and drop Collison. We have **94.7%** for this class.

SetCircle is responsible for implementing the circle property and initializing the SetCircle components when the main GUI launches. This class is tested when we test the main class using GUI test. It has only circle attributes which are used to set circle properties. So, we have **100%** coverage for this class.

GetDataController is responsible for creating a new entry. When a user clicks the create a button createText () method will be triggered and created text implement in the main window. And also, it has a changePrev () method to change SamplePreview label according to user input. Here also we missed some if branches only. We have **96.3%** coverage for this class.

EditController is responsible for editing existing entries. When a user clicks the edit button an edit () method will be triggered and updated entry implemented in the main window. And also, it has a changePrev () method to change SamplePreview label according to user input. Here also we missed some if branches only. We have **94.1%** coverage for this class.



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Venn	87.5 %	3,023	430	3,453
src/main/java	79.8 %	1,637	415	2,052
venn	80.0 %	1,637	410	2,047
VennController.java	63.5 %	625	359	984
DraggableText.java	94.7 %	302	17	319
EditController.java	94.1 %	209	13	222
GetDataController.java	96.3 %	314	12	326
Main.java	91.4 %	96	9	105
SetCircle.java	100.0 %	91	0	91
(default package)	0.0 %	0	5	5
src/test/java	98.9 %	1,386	15	1,401

Summary

Our current system is completely tested manually and above GUI and function tests are sufficiently enough that proves current systems working in acceptable conditions. And also, we have an overall **87.5% Test Coverage** rate. The amount of missing coverage percentage is only because of some unreachable branches and two major methods which we did not implement the test cases yet such as captureData () and exportData (). As the program currently stands it is impossible for to unexpectedly terminate, behave unexpectedly or provide different output for above given functions and features except the above mentioned two methods.