Test Plan

# Use Case Testing

1.

Title: Line Placement

Actors: User

Prerequisites: None

Main Scenario:

1. User selects line tool.

2. User clicks on an empty grid point to place a line.

3. User clicks on multiple empty grid points to continue adding new points to the line.

4. User clicks on the first point of the line to close the loop and finish line placement.

Alternatives:

2a. User clicks on an already occupied point.

2a1. Line is not placed, line tool remains selected.

3a. User clicks on an empty point that is not in a straight line from the previous point.

3a1. New point is not placed and the line tool remains selected.

3b. User clicks on a point already occupied by another line.

3b1. New point is not placed, line tool remains selected.

4a. User double clicks when placing a new point.

4a1. Line is finished but not closed.

Test Situations:

1. User places a closed loop line.

2. User places an open loop line.

3. User attempts to place a point that is not in a straight line.

4. User attempts to place a point that collides with an existing line.

5. User attempts to begin line placement on a point that collides with an existing line.

Test Coverage:

Number of main and alternatives: 5

Number of test situations: 5

100% Coverage.

2.

Title: Grid Panning

Actors: User  
Prerequisites: At least one line is drawn on the canvas with at least one orb on that line

Main Scenario:

1. User right clicks on canvas and drags to pan the grid.

2. The grid points all move in sync in the direction of the mouse's movement.

2. All orbs and lines are panned properly in sync with the grid.

Alternatives:

None.

Test Situations:

1. User pans the grid.

Test Coverage:

Number of main and alternatives: 1

Number of test situations: 1

100% coverage.

3.

Title: Line Editing

Actors: User

Prerequisites: At least one line is drawn on the canvas.

Main Scenario:

1. User selects the edit tool.

2. User clicks on a line.

3. The sidebar opens on the side of the screen farthest from the clicked point.

4. The user clicks and drags a slider on the line.

5. The user selects another tool and the sidebar disappears.

Alternatives:

2a. User clicks on an empty space that is not on a line

2a1. No response from the system.

5a. The user clicks on another line.

5a1. The sidebar re-appears showing the parameters of the newly selected line.

Test Situations:

1. User selects a line, edits a parameter, and deselects the edit tool.

2. User selects a line, edits a parameter, and selects another line to edit.

3. User does not select a line to edit.

Test Coverage:

Number of main and alternatives: 3

Number of test situations: 3

100% coverage.

4.

Title: Orb Placement

Actors: User

Prerequisites: At least one line is drawn on the canvas.

Main Scenario:

1. User selects the orb tool.

2. User clicks on an orbless point on a line.

3. An orb is placed on the line where the user clicked.

Alternatives:

2a. The user clicks on a point not occupied by a line.

2a1. Nor orb is placed.

2b. The user clicks on a point on a line already occupied by an orb.

2b1. No orb is placed.

Test Situations:

1. The user places an orb on a line.

2. The user attempts to place an orb on empty space.

3. The user attempts to place an orb on a line that already has an orb at that location.

Test Coverage:

Number of main and alternatives: 3

Number of test situations: 3

100% Coverage.

5.

Title: Orb Editing

Actors: User

Prerequisites: At least one line with at least one orb is on the board

Main Scenario:

1. User selects edit tool.

2. User clicks on an orb.

3. The sidebar opens with the orb's parameters.

4. The user adjusts a parameter.

5. The user clicks on another tool.

6. The sidebar closes and the orb is deselected.

Alternatives:

2a. User clicks on a line instead of an orb.

2a1. The line edit panel opens in the sidebar.

2b. User clicks on an empty space.

2b1. No response from the system.

5a. The user clicks on another orb.

5a1. The sidebar re-opens with that orb's parameters.

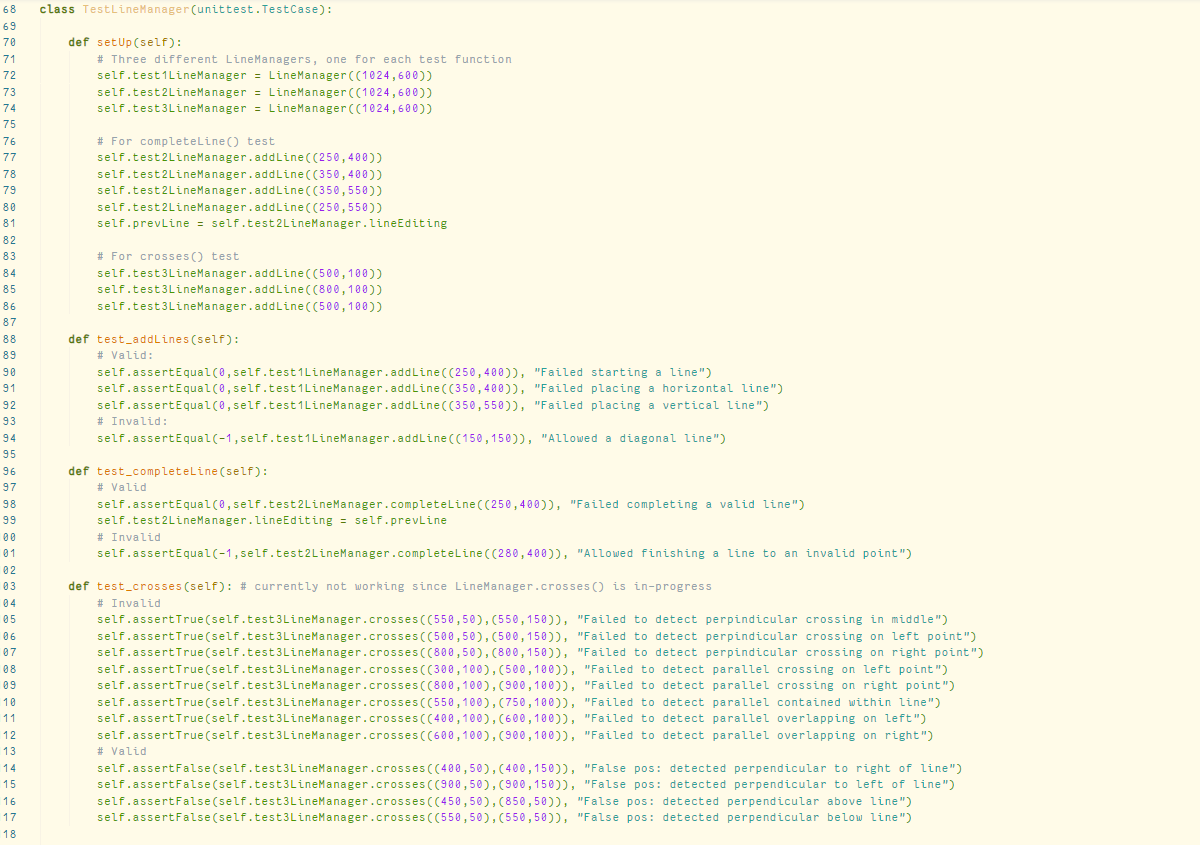
# Unit Testing

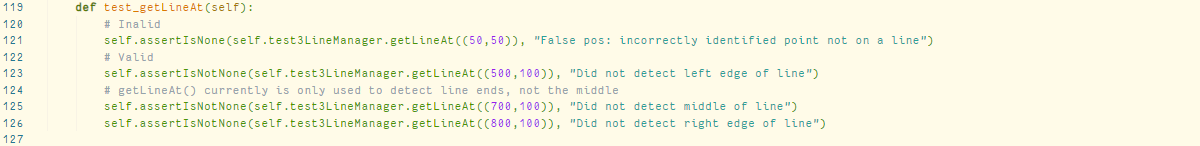
13 unit tests were implemented.

Unit tests for the Line class:

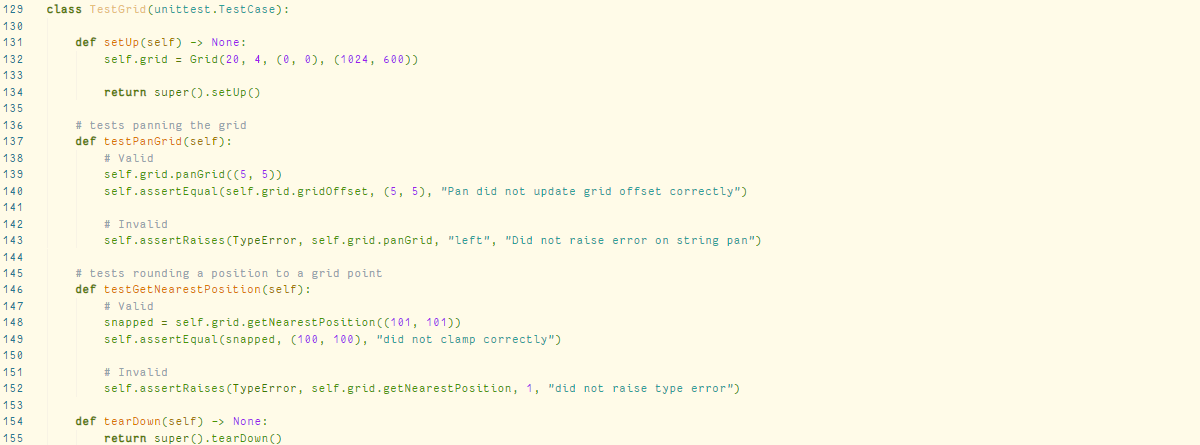


Unit tests for the LineManager class:





Unit tests for the Grid class:



Unit tests for the Tool class:



Below is the output of running the unit tests at the time of the creation of this document. From this, it can be seen that the crosses() and getLineAt() functions of the LineManager class are bugged and need fixing. Crosses() is in-progress, and getLineAt() currently is not used to detect points in the middle of lines, since the line only contains its endpoints.



# Acceptance Testing

At this time there has been no acceptance testing done for Acies. However, there is some acceptance testing planned. It is planned that an acceptance test for Acies would involve giving the application to a child and gauging their interest in the finished product by the amount of time they willingly spend interacting with it and the amount of unique song loops they create. For the best results, it would be preferable to run this acceptance test on a wide variety of children from a diverse selection of backgrounds, such as homeschooled, preschool age, and grade school age.