Exhibit A - Software Specification

I. Audio Handler

Overview

The Audio Handler Module's function is to load and store audio files streamed to it from a specified URL or local file path. It should provide the ability to loop the audio file once it has finished, and to start or stop a prepared playing audio file.

Functional Requirements

Constructors

No arguments are required for the class constructor.

Methods

registerAudio

Argument	Type	Description
url	String	URL for the source of the Audio File.
looping	Boolean	If true, the audio should loop back to the start after finishing.
id	String	ID assigned to the audio instance. If the requested ID is already in use, then the method should refuse the request.

startAudio

Argument	Type	Description
id	String	Start playback of the specified audio file.

stopAudio

Argument	Type	Description
id	String	Stop playback of the specified audio file.

deregisterAudio

Argument	Type	Description
id	String	Delete stored audio file.

CUBIXEL

II. Graphics Handler

Overview

The Graphics Handler Module's function is to draw JavaFX Graphics objects to individual JavaFX canvases and store those canvases with an ID so that each canvas can be individually added or removed as required. The Graphics Handler will take a specific set of inputs as detailed below and register and prepare a Line, Rectangle or Oval Object, draw or undraw a previously drawn object to the screen, or delete a stored graphic. The registerRectangle and registerOval methods should be overloaded to allow for either a single colour or a gradient fill.

Functional Requirements

Constructors

The Graphics Handler should provide the option to draw a line, rectangle and a circle via separate methods. The Graphics Handler should take as argument:

Argument	Type	Description
targetPane	StackPane (JavaFX)	This is the StackPane object onto which the Graphics Handlers methods should draw.

Methods

registerLine

Argument	Type	Description
xStart	float	X-coordinate for the starting point of the line, as a percentage of the width of the pane from the left edge.
xEnd	float	X-coordinate for the end point of the line, as a percentage of the width of the pane from the left edge.
yStart	float	Y-coordinate for the starting point of the line, as a percentage of the height of the pane from the top edge.
yEnd	float	Y-coordinate for the end point of the line, as a percentage of the height of the pane from the top edge.

lineColor	String	The colour of the line, in the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively.
id	String	The ID assigned to the drawn shape, used to later remove it. If the requested ID is already in use, then the method should refuse the request.

registerRectangle

Argument	Type	Description
xStart	float	X-coordinate for the top left point of the rectangle, as a percentage of the width of the pane from the left edge.
yStart	float	Y-coordinate for the top left point of the rectangle, as a percentage of the height of the pane from the top edge.
width	float	As a percentage of the width of the pane.
height	float	As a percentage of the height of the pane.
fillColour	String	The colour with which to fill the rectangle, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively. This is a block fill of the entire shape as one colour.
id	String	ID assigned to the drawn shape, used to later remove it. If the requested ID is already in use, then the method should refuse the request.

registerOval

Argument	Type	Description
xStart	float	X-coordinate for the top left point of the oval, as a percentage of the width of the pane from the left edge.
yStart	float	Y-coordinate for the top left point of the oval, as a percentage of the height of the pane from the top edge.
width	float	As a percentage of the width of the pane.
height	float	As a percentage of the height of the pane.
fillColour	String	The colour with which to fill the oval, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively. This is a block fill of the entire shape as one colour.
id	String	ID assigned to the drawn shape, used to later remove it. If the requested ID is already in use, then the method should refuse the request.

registerRectangle

Argument	Type	Description
xStart	float	X-coordinate for the top left point of the rectangle, as a percentage of the width of the pane from the left edge.
yStart	float	Y-coordinate for the top left point of the rectangle, as a percentage of the height of the pane from the top edge.
width	float	As a percentage of the width of the pane.

height	float	As a percentage of the height of the pane.
id	String	ID assigned to the drawn shape, used to later remove it. If the requested ID is already in use, then the method should refuse the request.
shading_x1	float	X-coordinate for the anchor point of colour 1 as a percentage of the width of targetPane.
shading_y1	float	Y-coordinate for the anchor point of colour 1 as a percentage of the width of targetPane.
shading_colour1	String	The colour anchored to the first specified point, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively.
shading_x2	float	X-coordinate for the anchor point of colour 2 as a percentage of the width of targetPane.
shading_y2	float	Y-coordinate for the anchor point of colour 2 as a percentage of the width of targetPane.
shading_colour2	String	The colour anchored to the second specified point, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively.
shading_cyclic	Boolean	True if the gradient pattern should cycle repeatedly between the two colors; False otherwise (see java.awt.GradientPaint).

registerOval

Argument	Type	Description
xStart	float	X-coordinate for the top left point of the oval, as a percentage of the width of the pane from the left edge.
yStart	float	Y-coordinate for the top left point of the oval, as a percentage of the height of the pane from the top edge.
width	float	As a percentage of the width of the pane.
height	float	As a percentage of the height of the pane.
id	String	ID assigned to the drawn shape, used to later remove it. If the requested ID is already in use, then the method should refuse the request.
shading_x1	float	X-coordinate for the anchor point of colour 1 as a percentage of the width of targetPane.
shading_y1	float	Y-coordinate for the anchor point of colour 1 as a percentage of the width of targetPane.
shading_colour1	String	The colour anchored to the first specified point, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively.
shading_x2	float	X-coordinate for the anchor point of colour 2 as a percentage of the width of targetPane.
shading_y2	float	Y-coordinate for the anchor point of colour 2 as a percentage of the width of targetPane.
shading_colour2	String	The colour anchored to the

		second specified point, with the form '#ffffff' where each pair of characters after the hash are the hexadecimal representation of red, green, blue respectively.
shading_cyclic	Boolean	True if the gradient pattern should cycle repeatedly between the two colors; False otherwise (see java.awt.GradientPaint).

drawGraphic

Argument	Type	Description
id	String	The id of the shape to be drawn.

undrawGraphic

Argument	Type	Description
id	String	The id of the shape to be undrawn.

deregisterGraphic

Argument	Type	Description
id	String	The id of the shape to be deleted.

III. Coding Standards

All code produced must meet the coding standards as laid out in the companies style guide, which can be found via the link below or else will be potentially rejected for revision.

Style Guide: https://google.github.io/styleguide/javaguide.html

Exhibit B - Milestone Schedule

Milestone	Delivery Deadline
Draft Contract Provided to Developer	26/02/2020
Proposal of Changes to Draft Contract	26/02/2020
Contract Agreed	27/02/2020
Client Demo	20/04/2020
Final Handover of all Deliverables	23/04/2020