Computer
Science

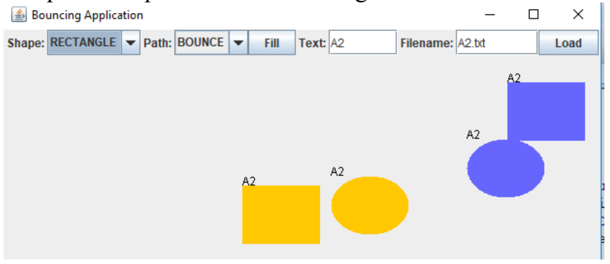
COMPSCI 230
Assignment TWO

Due Date

Due: **11:59pm 25th September 2022**
Worth: **4% of the final mark**

Introduction - The Bouncing Program

This assignment project aims to develop and reinforce skills in object-oriented programming. The context of this project is extended from A1. The assignment project contains three parts, and each part has several tasks. You will generally need to complete one part before continuing to the next. **This document only provides a summary of the requirement of A2**.



Assessment Criteria

Complete CodeRunner questions and submit the entire program via the **assignment dropbox (https://adb.auckland.ac.nz/)** at any time from the first submission date up until the final due date. You will receive an electronic receipt. Submit a single zip file containing all source files – remember to include your name, UPI and a comment at the beginning of each file.

You may make more than one submission, but note that every submission that you make replaces your previous submission. Submit ALL your files in every submission. Only your very latest submission will be marked. Please double check that you have included all the files required to run your program in the zip file before you submit it. Your program must compile and run to gain any marks.

Description

Task 1: Drawing Shapes [4 marks]

Your program should support the following:

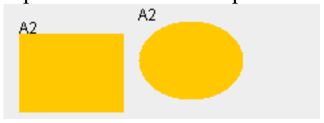
- Users would be able to create new shapes by clicking anywhere within the panel area of the program. The properties of the newly created shape are:
  - (`x`, `y`): x and y coordinates of the mouse point.
  - `width/height`: `currentwidth/curerntheight` from the `AnimationViewer` class.
  - `panelWidth/panelHeight`: `currentPanelWidth/currentPanelHeigh` from the `AnimationViewer` class.
  - `color`: `currentColor` from the `AnimationViewer` class.
  - shape type: `currentShapeType` from the `AnimationViewer` class.
  - path type: `currentPathType` from the `AnimationViewer` class.
- Once created, the shape will start moving.
- Users would be able to select a shape or multiple shapes by clicking anywhere on the shape.
  - A selected shape shows all its handles.
  - Users would be able to change the fill colour for all selected shapes by changing the current values with the help of the tools provided at the top of the application interface.
  - Note: The shape and path types cannot be modified once a shape has been created.
  - New shapes are created based on the updated values.
- Users would be able to deselect shape(s) by clicking anywhere on the shape(s).

Instructions
Download the template file from Canvas. Complete the entire program**:**

- Complete the `createNewShape()` in the `AnimationViewer` class to create a new rectangle or a new oval depending on the selected shape type in the shape type combo box. The method should also add the newly created shape to the array list.
- Complete the `paintComponent()` in the `AnimationViewer` class to draw a list of shapes.
  - The method should loop through each shape in the array list. The method should
    - move each shape by invoking the `move()` method,
    - draw each shape by invoking the `draw()` method, and
    - draw the handles by invoking the `drawHandles()` method.
- Complete the `draw()` method in the `RectangleShape` class to fill a rectangle.
  - The mehod should set the colour of the pen by invoking the `setColor()` method. The colour of the pen is the `color` defined in the `Shape` class.
  - The method should fill a rectangle by invoking the `fillRect()` method.
- Complete the `draw()` method in the `OvalShape` class to fill an oval.
  - The method should set the colour of the pen by invoking the `setColor()` method.
  - The method should fill an oval by invoking the `fillOval()` method.
- Complete the `setCurrentColor()` method in the `AnimationViewer` class to update the current fill colour and the fill colour of all the **selected** shapes.
- Complete the `FillActionListener` inner class in the `A2` class to allow users to change the current fill colour and the fill colour of all the **selected** shapes by invoking the `panel.setCurrentColor()` method.

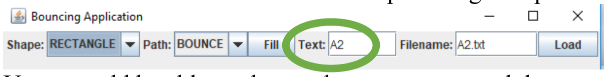
Task 2: The Text property [5 marks]

You must modify the **`Shape`** class hierarchy to allow text to be displayed when a shape is painted. Text should be shown in the top-left corner of a shape.



Your program should support the following:

- Users would be able to create new shapes using the specified text at the top of the program.



- Users would be able to change the current text and the text of all the selected shapes by changing the text in the text field at the top of the program.

Continuing on your solution from Task1, complete the CodeRunner questions (3 marks) and the entire program (2 marks). CodeRunner checks if the text attribute is correctly defined.

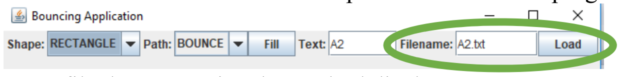
Instructions:

- Modify the `Shape` class. Follow the instructions given in CodeRunner.
- Modify the `RectangleShape` class and the `OvalShape` class. Follow the instructions given in CodeRunner.
- Modify the `AnimationViewer` class. Follow the instructions given in CodeRunner.
- Modify the `A2` class
  - Complete the `TextActionListener` inner class in the A2 class to allow users to change the current text message and the text message of all the **selected** shapes.

Task 3: The Load Button [5 marks]

Your program should support the following:

- Users would be able to add new shapes by clicking the `Load` button. The method clears the filename in the `filenameTextField` text field if shapes are added to the program successfully.



- If the text file does not exist, the method displays "**`Invalid filename.`**" in the `filenameTextField` text field.

The text file contains the shape type, the path type, the colour, two integers, and the text message per line. The first integer represents the x coordinate, and the second integer represents the y coordinate. For example, A2.txt contains the following:

```
OVAL,BOUNCE,red,10,20,Good Job
RECTANGLE,BOUNCE,orange,50,100,Testing
RECTANGLE,FALL,blue,80,120,A2Rectangle
```

Your program should create three shapes and add them to the shapes array list. The properties of the 3 shapes are:

- The top left corner is (10, 20), the dimension is 80 x 60, it is a red oval, the path is bouncing, and the text message is "Good Job".
- The top left corner is (50, 100), the dimension is 80 x 60, it is an orange rectangle, the path is bouncing, and the text message is "Testing".
- The top left corner is (80, 120), the dimension is 80 x 60, it is a blue rectangle, the path is falling, and the text message is "A2Rectangle"

Note: **`width`**, **`height`**, **`panelWidth`**, **`panelHeight`** are based on the current values stored in the **`AnimationViewer`** class.

Instructions:

- Modify the `AnimationViewer` class. Follow the instructions given in CodeRunner.
- Modify the `A2` class:
  - Complete the `LoadActionListener` inner class in the A2 class to allow users to load a list of shapes from a text file into the array list.

MARKING SCHEME

The total marks of this assignment is 16.

1 mark	Comments at the top of each class (containing your name, upi, and a brief description of the class)
1 mark	The code is self-documenting (good and meaningful variable names, etc).
Task1	
1 mark	Users would be able to create new shapes with correct properties.
1 mark	Users would be able to change the fill colour of all selected shapes.
1 mark	New shape are created based on the updated values
1 mark	Shapes are moving around the bouncing area correctly.
Task2	
3 marks	CodeRunner questions
1 mark	Text is drawn at the toft-left corner of each shape
0.5 marks	User would be able to change the text property of all selected shapes.
0.5 marks	User would be able to create a new shape using the given text property
Task3	
2 marks	CodeRunner questions
1 mark	Users would be able to load shapes from a text file.
1 mark	If shapes are added successfully, the method clears the filename in the filename `JTextField`
1 mark	If the file does not exist, the method sets an error message in the filename `JTextField`

ACADEMIC INTEGRITY

The purpose of this assignment is to help you develop a working understanding of some of the concepts you are taught in the lectures. We expect that you will want to use this opportunity to be able to answer the corresponding questions in the tests and exam. We expect that the work done on this assignment will be your own work. We expect that you will think carefully about any problems you come across, and try to solve them yourself before you ask anyone for help. The following sources of help are not acceptable:

- Getting another student, or other third party to instruct you on how to design classes or have them write code for you.
- Taking or obtaining an electronic copy of someone else's work, or part thereof.
- Give a copy of your work, or part thereof, to someone else.
- Using code from past sample solutions or from online sources dedicated to this assignment.

The Computer Science department uses copy detection tools on all submissions. Submissions found to share code with those of other people will be detected and disciplinary action will be taken. To ensure that you are not unfairly accused of cheating:

- Always do individual assignments by yourself.
- Never give any other person your code or sample solutions in your possession.
- Never put your code in a public place (e.g., Piazza, forum, your web site).
- Never leave your computer unattended. You are responsible for the security of your account.
- Ensure you always remove your USB flash drive from the computer before you log off, and keep it safe.