COMP10062: Final Exam (June 2023)

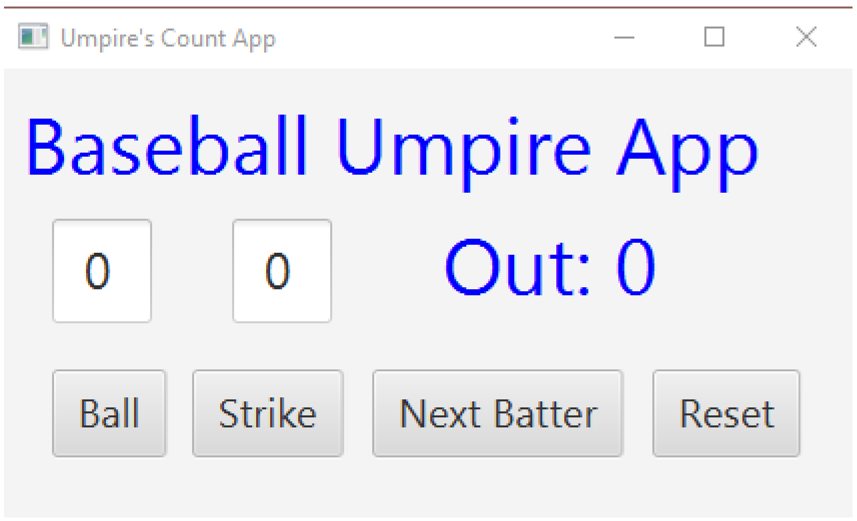
Mohawk College, Instructor: Dave Slemon & Nathan Rowbottom

2. In this question we are building a baseball umpire's counting app. See the product shot in figure 1. Create the app **EXACTLY** as shown in figure 1. Since this app maybe used in real games, follow the specifications carefully and when something is not specified, make smart assumptions.

A ***Counter*** class is provided in the starter kit and does not require any alterations. Below is a small example of how the Counter class works. You will need 3 separate counters in this app, one for the balls, one for the strikes and one for the outs.

```
Counter c = new Counter( );
c.increment( );             //adds one to the current count
int num = c.getValue( );    //returns the current value as an int, here 1 is returned
String str = c.toString();  //returns the current value as a String,
                            //          here "1" is returned
c.clear( );                 //resets the current count back to 0
```

Note: the ***toString()*** method above provides a *free string conversion* of the current count … this will be helpful in the GUI code.


Umpire's Count App

Baseball Umpire App

0 0 Out: 0

Figure 1. Product shot

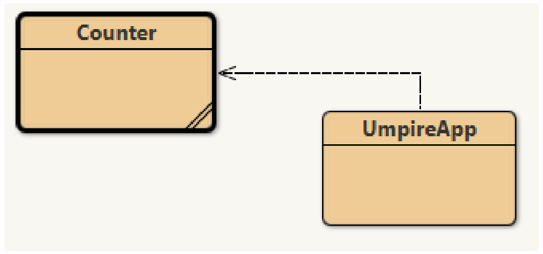


Figure 2. Class Layout

Button Specifications

Reset – sets the value in text fields *balls, strikes* and *outs* to 0.

Next Batter – sets the value in text fields *balls* and *strikes* to 0, leaving the value in *outs* unchanged.

Ball – increments the ball count by 1 until 4 is reached. The only values allowed to be displayed in the ball textfield are: 0, 1,2,3, and 4. When ball 4 is displayed, further presses of the Ball button does nothing and leaves the count at 4.

Strike – increments the strike count by 1 until 3 is reached. The only values allowed to be displayed in the strike textfield are 0, 1, 2, and 3. When strike 3 is reached, further presses of the Strike button will not do anything, 3 will remain displayed. Also, when strike 3 is reached, the number of *outs* will be incremented by 1.

Out Label – increments by 1 each time strike 3 is reached or the mouse is pressed against the canvas. The only counts allowed to be displayed in the *outs* label are: 0, 1, 2 and 3. If the left mouse button is clicked anywhere on the canvas, the number of outs is increased by 1. The *outs* displayed can not exceed 3.

Left Mouse Button – many times in baseball, an OUT will occur for several reasons, other than reaching strike 3. For example, an out might occur on strike 1 if the batter hits a fly ball which is caught. A left mouse click anywhere on the canvas will increase the number of *outs* by 1.

<u>Sample Session</u>

Recommended: Try out the sequence below to check your app's compliance.

```
Reset       0 0 0

Ball        1 0 0
Strike      1 1 0
Ball        2 1 0
Ball        3 1 0
Ball        4 1 0
Next Batter 0 0 0

Ball        1 0 0
Strike      1 1 0
Strike      1 2 0
Strike      1 3 1
Next Batter 0 0 1

Ball        1 0 1
Strike      1 1 1
Ball        2 1 1
Strike      2 2 1
Strike      2 3 2
Next Batter 0 0 2

Strike      0 1 2
Mouse Click 0 1 3

Reset       0 0 0
```

Helpful Things

```
balls.relocate(25,75);
balls.setPrefWidth(50);
balls.setPrefHeight(50);
balls.setFont(new Font("System",25));

ballButton.relocate(25,150);
ballButton.setPrefHeight(40);
ballButton.setFont(new Font("System",20));
```

```
title.relocate(10,10);
title.setFont(new Font("System",40));
title.setStyle("-fx-text-fill: blue;");
```

```
balls.setText(ballCount.toString());
```