There will be more animation in Chapter 16
Learn to interact with animation using
- **Mouse**
- **Key**
- **Slider**

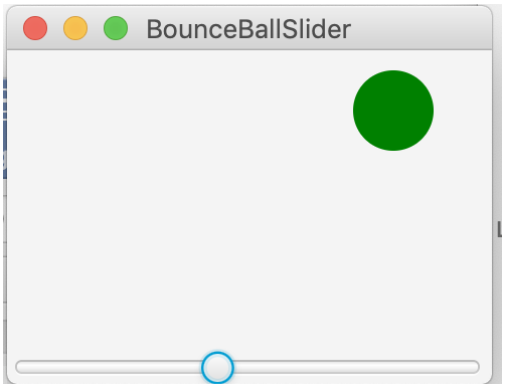
This is another example of animation

§15.12 Animation using

- **mouse to control start, stop**
- **key up and down to increase decrease the rate of animation**
  - recall requestFocus() method for keys does not work as expected
- **slider to control frame rate of animation**

Create the animation
discussion and description

Example: Demo, Source code



Bouncing balls
Ball is a circle filled with green.
The ball starts moving with increments in x and y by 1.
As long as the ball does not hit the boundaries, it keeps incrementing x, and y with dx=+1, dy=+1.
If it hits the **left** boundary then change the direction by incrementing dx=+1.
If it hits the **right** boundary then change the direction by decrementing dx= -1
If it hits the **top** boundary then change the direction by incrementing dy=+1.
If it hits the **bottom** boundary then change the direction by decrementing dy= -1.

Timeline is subclass of Animation abstract class.
BallPane creates Animation object
timeline
It has
duration of time between two frames.
rate property,
rate is the time difference between frames in microseconds
Methods
play() – start the animation,
stop() – stop
pause() – temporarily stop the animation

Timeline manages the animation **Frames**.

KeyFrame is a class that specifies the duration and creation of frame for animation object to be animated
Duration is the time difference between two frames
EventHandler<ActionEvent> create the frames

Example

```
Timeline    timeline = new Timeline(
      new KeyFrame(Duration.millis(50), e -> moveBall());
     timeline.setCycleCount(Timeline.INDEFINITE);
      timeline.play(); // Start animation
  }
```

```
package code;
// import statements needed
import javafx.animation.Timeline;
import javafx.animation.KeyFrame;
import javafx.util.Duration;

import javafx.beans.property.DoubleProperty;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;

public class BallPane extends Pane {
  public final double radius = 20;
  private double x = radius, y = radius;
  private double dx = 1, dy = 1;
  private Circle circle = new Circle(x, y, radius);
  private Timeline timeline;

  public BallPane() {
    circle.setFill(Color.GREEN); // Set ball color
    getChildren().add(circle); // Place a ball into this pane
```

```
    // Create an animation for moving the ball
    timeline = new Timeline(
      new KeyFrame(Duration.millis(50), e ->
moveBall())
);
    timeline.setCycleCount(Timeline.INDEFINITE);
    timeline.play(); // Start animation
  }

  public void play() {
    timeline.play();
  }

  public void pause() {
    timeline.pause();
  }

  public void increaseSpeed() {
    timeline.setRate(timeline.getRate() + 0.1);
  }

  public void decreaseSpeed() {
    timeline.setRate(
      timeline.getRate() >= 0.1 ? timeline.getRate() - 0.1 :
0);
  }
```

```
  public DoubleProperty rateProperty() {
    return timeline.rateProperty();
  }

  protected void moveBall() {
    // Check boundaries
    if (x < radius || x > getWidth() - radius) {
      dx *= -1; // Change ball move direction
    }
    if (y < radius || y > getHeight() - radius) {
      dy *= -1; // Change ball move direction
    }

    // Adjust ball position
    x += dx;
    y += dy;
    circle.setCenterX(x);
    circle.setCenterY(y);
  }
}
```

Driver with slider

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Slider;
import javafx.scene.layout.BorderPane;
```

```
public class BounceBallSlider extends Application {
  @Override // Override the start method in the
Application class
  public void start(Stage primaryStage) {
    BallPane ballPane = new BallPane();
    Slider slider = new Slider();
    slider.setMax(20);

ballPane.rateProperty().bind(slider.valueProperty());//
slider connection with the ball

    BorderPane pane = new BorderPane();
    pane.setCenter(ballPane);
    pane.setBottom(slider);

    // Create a scene and place it in the stage
    Scene scene = new Scene(pane, 250, 250);
    primaryStage.setTitle("BounceBallSlider"); // Set the
stage title
    primaryStage.setScene(scene); // Place the scene in the
stage
    primaryStage.show(); // Display the stage
  }

  /**
   * The main method is only needed for the IDE with
limited
   * JavaFX support. Not needed for running from the
```

```
command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

Keys/Mouse interface Demo

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.input.KeyCode;

public class BounceBallControl extends Application {
   @Override // Override the start method in the
Application class
   public void start(Stage primaryStage) {
      BallPane ballPane = new BallPane(); // Create a ball
pane

      // Pause and resume animation
      ballPane.setOnMousePressed(e -> ballPane.pause());
      ballPane.setOnMouseReleased(e -> ballPane.play());

      // Increase and decrease animation
      ballPane.setOnKeyPressed(e -> {
        if (e.getCode() == KeyCode.UP) {
          ballPane.increaseSpeed();
```

```
            }
            else if (e.getCode() == KeyCode.DOWN) {
                ballPane.decreaseSpeed();
            }
        });

        // Create a scene and place it in the stage
        Scene scene = new Scene(ballPane, 250, 150);
        primaryStage.setTitle("BounceBallControl"); // Set
the stage title
        primaryStage.setScene(scene); // Place the scene in
the stage
        primaryStage.show(); // Display the stage

        // Must request focus after the primary stage is
displayed
        ballPane.requestFocus();
    }

    /**
     * The main method is only needed for the IDE with
limited
     * JavaFX support. Not needed for running from the
command line.
     */
    public static void main(String[] args) {
        launch(args);
    }
```

```
}
```