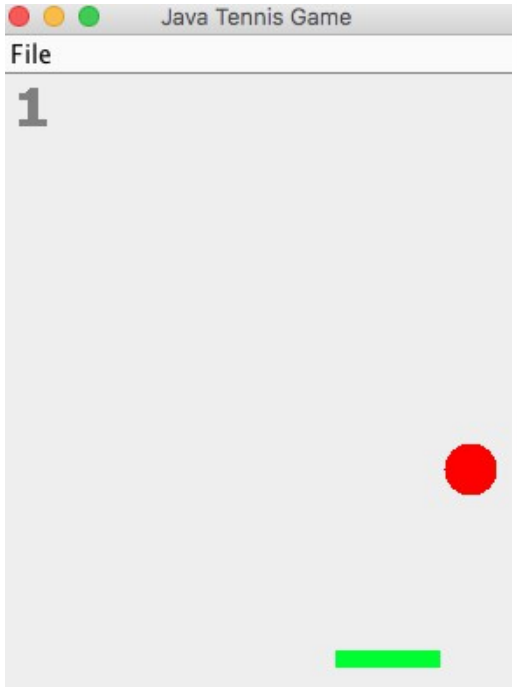
SchoolNova

IT101

Graphical User Interface



Foundation

- Swing is a platform-independent set of Java classes used for user Graphical User Interface (GUI) programming. Abstract Window Toolkit (AWT) is an older Java GUI technology, some of which is still useful.
- Create a new Java Application project in NetBeans;
- Create a main class named TennisGame;
- Import javax.swing.*, java.awt.* and java.awt.geom.* packages into the TennisGame class.
- Make your TennisGame class extend the JPanel class. JPanel is a painting canvas, by extending which you will be able to paint and display objects in TennisGame.
- A painting canvas must be contained in a window frame. JFrame is the class that serves as a frame for our TennisGame canvas. Create a new JFrame object inside the TennisGame's “main” method.
- Create an instance of the TennisGame class and add the TennisGame object reference to the frame.
- Set the frame's size using the frame's setSize method.
- Set the frame's visibility to "true".
- Shut down the Java thread that runs your program when the frame is closed.
- Build and run your program.
- You should see an empty 300px by 300px box.

```
    public static void main(String[] args) {
        JFrame frame = new JFrame("Java Tennis Game");
        TennisGame tg = new TennisGame();
        frame.add(tg);
        frame.setSize(300, 300);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```

Paint the Objects

- Start painting the tennis ball by overriding the JPanel's paint method: public void paint(Graphics g);
- The paint method receives by parameter a Graphics object. We will use a newer object: Graphics2D, which extends from Graphics. In order to do that cast Graphics g into Graphics2D.
- To draw something inside the canvas we should indicate in which position we are going to start painting. For this, each of the points in the canvas has an associated position (x,y) being (0,0) the point of the top-left corner.
- Set the color.
- Fill the tennis ball with color.
- Draw the ball.
- Build and run your program.
- You should see the red tennis ball at the top left of your canvas.

```
@Override
public void paint(Graphics g) {
    super.paint(g);
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.RED);
    g2d.fillOval(x, y, 30, 30);
}
```

Animate

- Every time we paint something we have to define its position (x,y). To make the ball move, modify the position (x,y) each time and repaint the circle in the new position.
- In our example, we keep the current position of our circle in two properties called "x" and "y". We also create a method called moveBall() which will increase in 1 both "x" and "y", each time we call it.
- At the end of the main method we start an infinite loop "while (true)" where we repeatedly call moveBall() to change the position of the circle and then call the JFrame's repaint() method.
- "Thread.sleep(10)" tells the processor that the Java thread which is being run must sleep for 10 milliseconds, which allows the processor to execute other threads and in particular the AWT thread which calls the paint method.
- Build and run your program. You should see the red tennis ball moving through the canvas.

```
private int x = 0;
private int y = 0;
/*
* Animation
*/
private void moveBall() {
        x = x + 1;
        y = y + 1;
}
```

```
public static void main(String[] args) throws InterruptedException {
    JFrame frame = new JFrame("Java Tennis Game");
    TennisGame tg = new TennisGame();
    frame.add(tg);
    frame.setSize(300, 300);
    frame.setVisible(true);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    while (true) {
        tg.moveBall();
        tg.repaint();
        Thread.sleep(10);
    }
}
```

Encapsulate the Ball

- Create a class called Ball which isolates everything that has to do with the ball.
- Make the ball go in different directions. Integers "xa" and "ya" represent the speed in which the ball is moving. If xa=1, the ball moves to the right, one pixel every round of the Game Loop, if xa=-1, the ball moves to the left. In the same way ya=1 moves the ball down and ya=-1 moves the ball up. This is done with the lines, "x = x + xa" and "y = y + ya" of the moveBall() method.
- Verify that the ball does not go out of the borders of the canvas. For example, when the ball gets to the right border, or when (x + xa > getWidth() - 30), what we`ll do, is change the direction of the movement on the "x": "xa = -1".

```
public class Ball {
        int x = 0;
        int y = 0;
        int xa = 1;
        int ya = 1;
        private TennisGame game;

        public Ball(TennisGame game) {
                this.game= game;
        }

        public void moveBall() {
            if (x + xa < 0){
                    xa = 1;
            }
            if (x + xa > game.getWidth() - 30) {
                    xa = -1;
            }
            if (y + ya < 0) {
                    ya = 1;
            }
            if (y + ya > game.getHeight() - 30) {
                    ya = -1;
            }
            x = x + xa;
            y = y + ya;
    }

        public void paint(Graphics2D g) {
                g.fillOval(x, y, 30, 30);
        }
}
```

Encapsulate the Racquet

- Create a class called Racquet which isolates everything that has to do with the racquet.
- Unlike "Ball", "Racquet" does not have any properties for the position "y" or for the speed "ya". This is because the racquet doesn´t change its vertical position; it will only move left or right, never up or down. In the paint method, the g.fillRect(x, 330, 60, 10) instruction defines a rectangle of 60 by 10 pixels in the position (x,y)=(x,330). As we can see "x" can change but "y" is fixed in 330 pixels from the top border of the canvas.
- When someone presses a key, the keyPressed method of "Racquet" will be called and this will set "xa" to 1, if the key pressed is the right direction (KeyEvent.VK_RIGHT), that will move the racquet to the right. In the same way if we press the key KeyEvent.VK_LEFT it will move to the left.
- When a key is released, the method keyReleased is called and "xa" changes its value to zero, which makes the racquet stop.

```
package tennisgame;
import java.awt.*;
import java.awt.event.*;

public class Racquet {
        int x = 0;
        int xa = 0;
        private TennisGame game;

        public Racquet(TennisGame game) {
            this.game= game;
        }

        public void move() {
            if (x + xa > 0 && x + xa < game.getWidth()-60) {
                    x = x + xa;
            }
        }

        public void paint(Graphics2D g) {
            g.fillRect(x, 330, 60, 10);
        }

        public void keyReleased(KeyEvent e) {
            xa = 0;
        }

        public void keyPressed(KeyEvent e) {
            if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                    xa = -1;
            }
            if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
                    xa = 1;
            }
        }
}
```

Revised TennisGame.java

- Instantiate Ball and Racquet objects in TennisGame;
- Add a TennisGame constructor.
- To read from the keyboard it is necessary to register an object which is in charge of "listening if a key is pressed". This object is known as "Listener" and it will have methods that will be called when someone presses a key. In our example the Listener is included in the TennisGame constructor.
- Add the “move” method to TennisGame. “move” calls Ball and Racquet “move” methods.
- Change the “while” loop in the “main” method to call the TennisGame's “move” method.

```
while (true) {
        game.move();
        game.repaint();
        Thread.sleep(10);
}
```

```
        Ball ball = new Ball(this);
        Racquet racquet = new Racquet(this);

        public TennisGame() {
            addKeyListener(new KeyListener() {
                @Override
                public void keyTyped(KeyEvent e) {
                }
                @Override
                public void keyReleased(KeyEvent e) {
                        racquet.keyReleased(e);
                }
                @Override
                public void keyPressed(KeyEvent e) {
                        racquet.keyPressed(e);
                }
            });
            setFocusable(true);
        }

        private void move() {
                ball.moveBall();
                racquet.move();
        }
```

Homework

- We will continue the Tennis Game project next week.
- You should have the TennisGame, Ball and Racquet classes ready (as shown above) for the next class.
- Build and run your project. Make sure your code compiles without errors, and that you see the Ball moving across the canvas, and that you can control the Racquet movement using the left and right arrow keyboard keys.