Interface

- In Java, an interface is used to specify required or common operations for classes using that interface.

Ex. method drive() Potential classes that use drive() (Car, Motorcycle, Boat, Tank, Bus)

Differences between Interface and Class

1. All methods in interfaces are abstract.
2. All methods in interfaces are public.
3. Interfaces do not have instance variables.

This means that any class using an interface must implement all of its abstract methods.

Why use Interfaces?

- Inheritance provides code reuse.
- Interface forces overriding methods.
- You can only have one super class, but you can have multiple Interfaces.

class A extends B class A implements B (B is Interface)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
// import java.awt.Rectangle;

public class MovingRectangle extends JComponent {
    private Rectangle rect;

    public MovingRectangle(){
        rect = new Rectangle(0, 100, 100, 100);
    }

    public void paintComponent(Graphics g){
        Graphics2D g2 = (Graphics2D) g;
        g2.draw(rect);
    }

    public void moveBy(int x, int y){
        rect.translate(x,y);
        repaint(); // calls the paintComponent
    }
```

```
    public static void main (String [] args){
        JFrame Frame = new JFrame();
        Frame.setSize(1000, 1000);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        MovingRectangle rect = new MovingRectangle();
        frame.add(rect);
        frame.setVisible(true);

        class TimerListener implements ActionListener {
            public void actionPerformed(ActionEvent event){
                rect.moveBy(1,1);
            }
        }

        ActionListener listener = new TimerListener();
        Timer t = new Timer(300, listener); // delay is 300 milliseconds
        t.start();
    }

}
```