MINI PROJECT: Image Viewer using Linked List

SOURCE CODE:

```
package imageN;

import javax.imageio.ImageIO;
import javax.swing.*;
import java.awt.image.BufferedImage;
import java.io.File;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.LinkedList;

public class ImageViewer1 extends JFrame {
     private class LinkedListNode {
        private BufferedImage image;
        private String description;
        private LinkedListNode next;
        private LinkedListNode prev;

        public LinkedListNode(String imagePath) {
          next=null;
            loadImage(imagePath);
            prev=null;
        }

        public BufferedImage getImage() {
            return image;
        }

        public String getDescription() {
            return description;
        }

        public LinkedListNode getNext() {
            return next;
        }

        public void setNext(LinkedListNode next) {
            this.next = next;
        }

        public LinkedListNode getPrev() {
            return prev;
        }

        public void setPrev(LinkedListNode prev) {
            this.prev = prev;
```

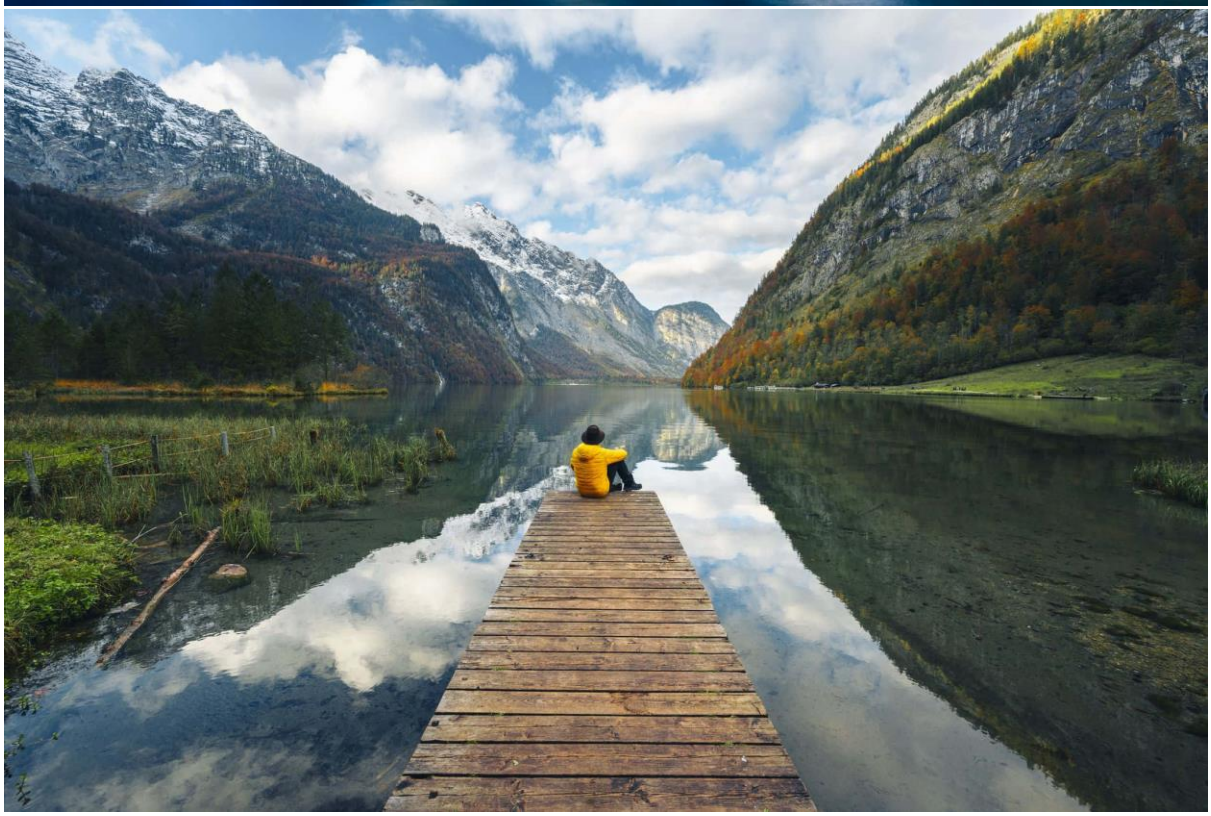
```
        }

        private void loadImage(String imagePath) {
            try {
                image = ImageIO.read(new File(imagePath));
                description = new File(imagePath).getName();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
    private LinkedList<String> imagePaths;
    private LinkedListNode currentImageNode;
    private JLabel imageLabel;
    private JButton prevButton;
    private JButton nextButton;

    public ImageViewer1() {
        setTitle("IMAGE VIEWER");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(800, 600);
        // Initialize the linked list of image paths
        imagePaths = new LinkedList<String>();
        File imageDirectory = new File("images");
        if (imageDirectory.isDirectory()) {
            for (File file : imageDirectory.listFiles()) {
                if (file.isFile()) {
                    imagePaths.add(file.getAbsolutePath());
                }
            }
        }
        // Initialize the linked list nodes
        for (String imagePath : imagePaths) {
          LinkedListNode temp=currentImageNode;
            if (currentImageNode == null) {
                currentImageNode = new
LinkedListNode(imagePath);
            } else {
               LinkedListNode newNode = new
LinkedListNode(imagePath);
               while(temp.next!=null) {
                temp=temp.next;
               }
               temp.setNext(newNode);
               newNode.setPrev(temp);
            }
        }

        //Create and add components to the frame
        imageLabel = new JLabel();
        getContentPane().add(imageLabel, BorderLayout.CENTER);
```

```
        prevButton = new JButton("Previous");
        prevButton.addActionListener(new ActionListener() {
          @Override
            public void actionPerformed(ActionEvent e) {
                showPreviousImage();
            }
        });

        nextButton = new JButton("Next");
        nextButton.addActionListener(new ActionListener() {
          @Override
            public void actionPerformed(ActionEvent e) {
                showNextImage();
            }
        });

        JPanel buttonPanel = new JPanel();
        buttonPanel.add(prevButton);
        buttonPanel.add(nextButton);
        getContentPane().add(buttonPanel, BorderLayout.SOUTH);

        // Display the first image
        showCurrentImage();
    }

    private void showCurrentImage() {
        if(currentImageNode != null) {
          LinkedListNode temp1=currentImageNode;
            BufferedImage image = temp1.getImage();
            imageLabel.setIcon(new ImageIcon(image));
            setTitle("Image Viewer - " +
temp1.getDescription());
        }
    }

    private void showNextImage() {
        if(currentImageNode.next != null) {
            currentImageNode = currentImageNode.getNext();
            showCurrentImage();
        }
    }

    private void showPreviousImage() {
        if(currentImageNode.prev != null) {
          currentImageNode = currentImageNode.getPrev();
            showCurrentImage();
        }
    }

    public static void main(String[] args) {
     JFrame j=new JFrame("IMAGE VIEWER");
```

```
      JPanel p=new JPanel();
      p.setLayout(new BoxLayout(p, BoxLayout.PAGE_AXIS));
      //p.setLayout(new FlowLayout());
      JLabel l=new JLabel("WELCOME TO IMAGE VIEWER");
      l.setFont(new Font("Comic Sans MS",Font.BOLD,24));
      l.setAlignmentX(Component.CENTER_ALIGNMENT);// Center the
label horizontally
      p.setBackground(Color.pink);
      JButton b=new JButton("START");
      b.setFont(new Font("Comic Sans MS",Font.BOLD,18));
      b.setAlignmentX(Component.CENTER_ALIGNMENT); // Center
the button horizontally
      p.add(Box.createVerticalGlue());
      p.add(l);
      p.add(b);
      p.add(Box.createVerticalGlue());
      b.addActionListener(new ActionListener() {
            @Override
              public void actionPerformed(ActionEvent e) {
                 SwingUtilities.invokeLater(() -> {
                       ImageViewer viewer = new ImageViewer();
                       viewer.setLocationRelativeTo(null);
                       viewer.setVisible(true);
                   });
              }
         });
      p.add(b);
      j.add(p);
      j.setSize(400,200);
      j.setLocationRelativeTo(null);
      j.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      j.setVisible(true);

    }
}
```

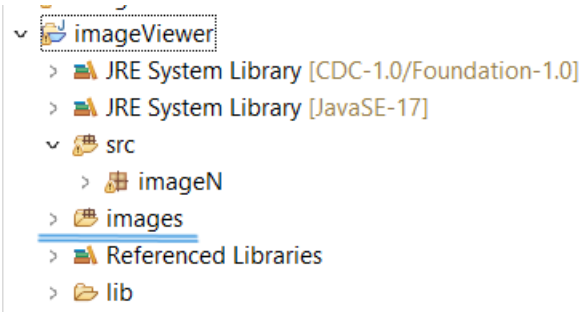
NOTE:

1.Images used in the above project:

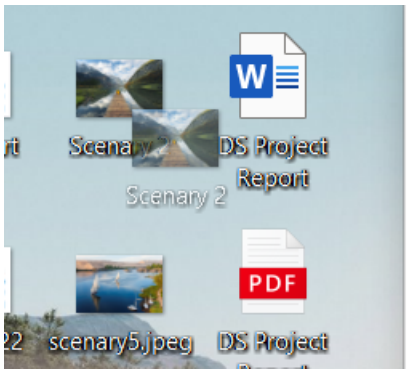


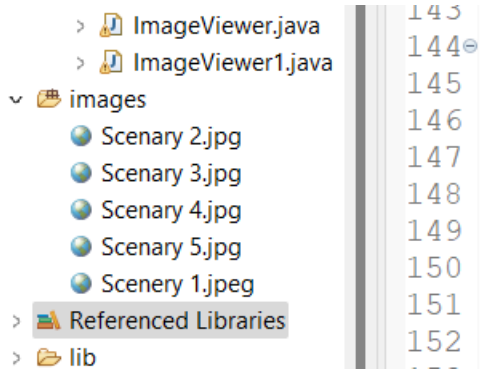
2.For above code to run perform following steps to first create a folder of images you want in your image viewer:

A] Create an empty folder in the java project in which the above code is saved. Name the folder as ‘images’(since code uses this file name).



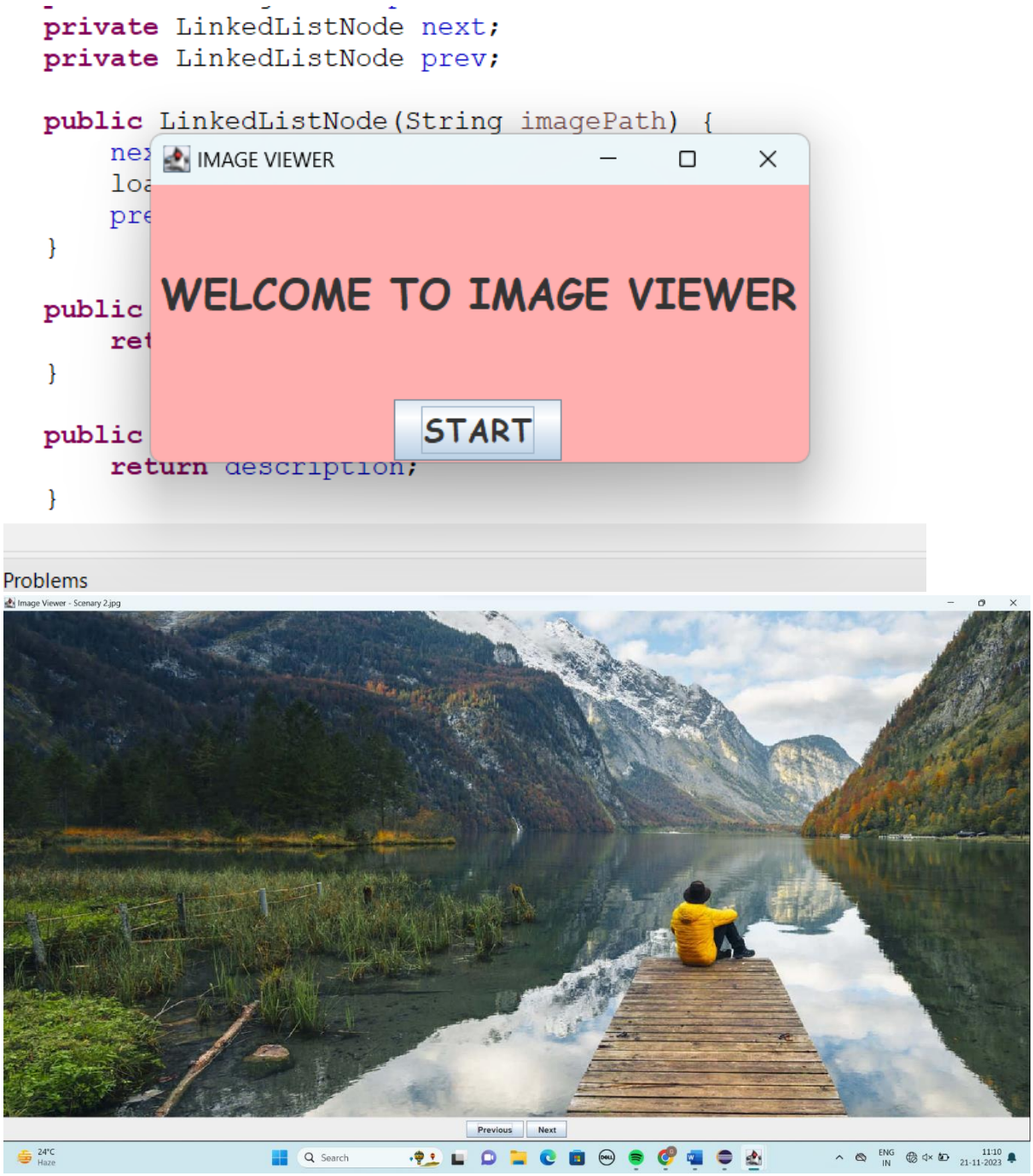
B] Drag the above images (or you can use images that you want) in this folder.





3.Make sure to use updated version of Eclipse IDE for Java Developers. Also load Swing jars since the program uses swing operations to run.

OUTPUT:



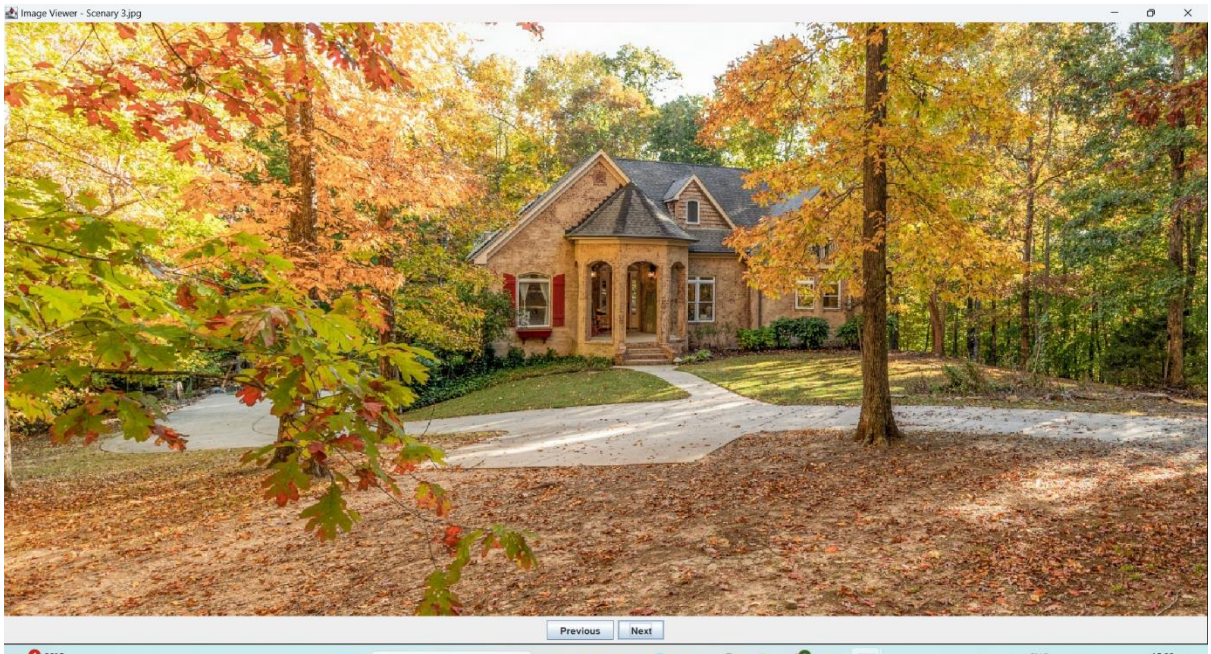


Image Viewer - Scenary 3.jpg
Previous
Next
30°C
Haze
Search
ENG
IN
15:03
21-11-2023


Image Viewer - Scenary 4.jpg
Previous
Next
30°C
Haze
Search
ENG
IN
15:04
21-11-2023

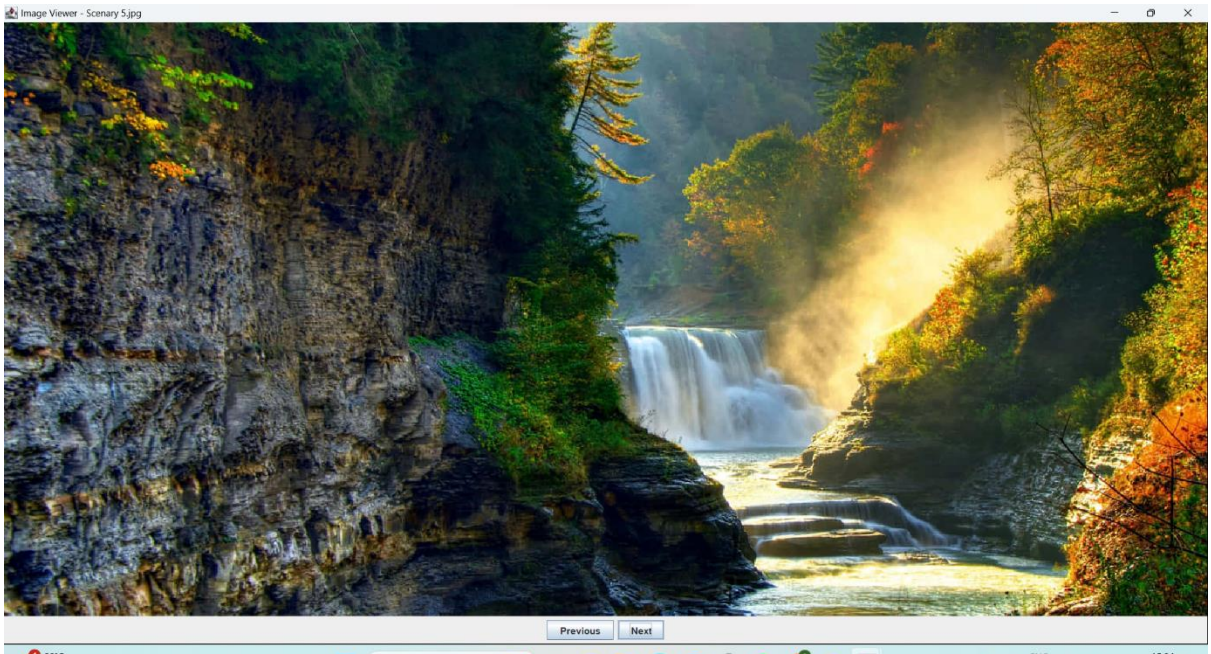


Image Viewer - Scenary 5.jpg
Previous
Next
30°C
Haze
Search
ENG
IN
15:04
21-11-2023

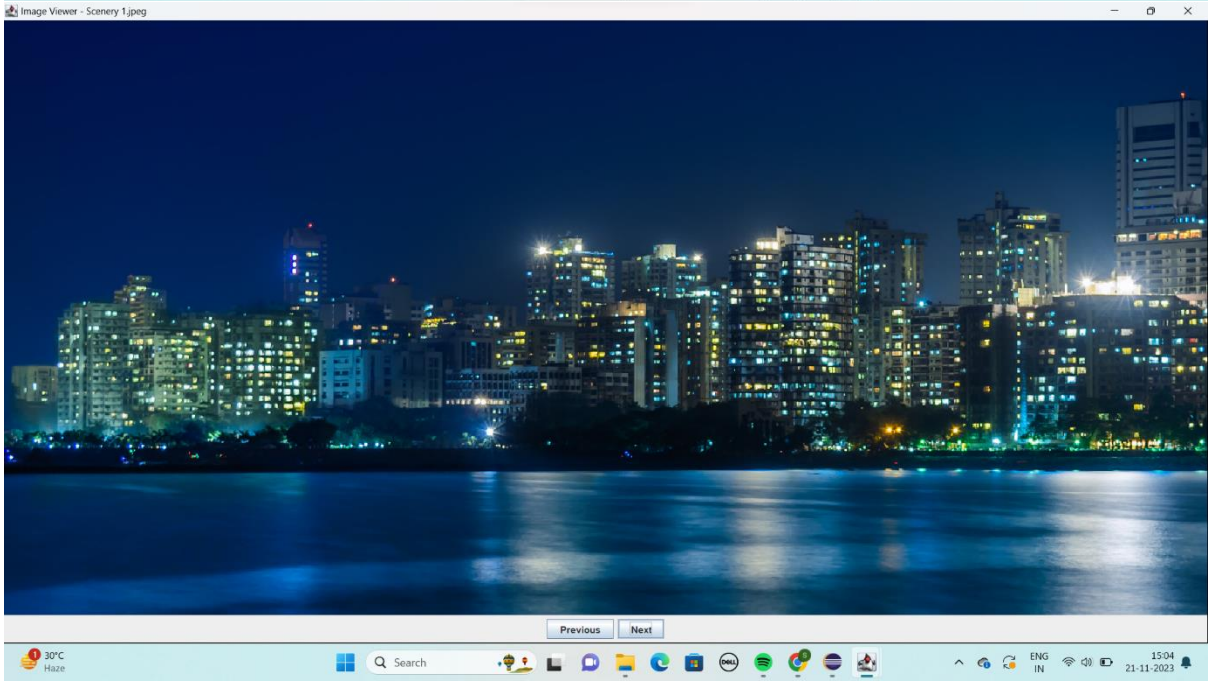


Image Viewer - Scenery 1.jpeg
Previous
Next
30°C
Haze
Search
ENG
IN
15:04
21-11-2023