Eric Wilson Project #6

PhotoViewerController.java

```java
/**
 * Created by EW043872 on 10/4/2015.
 */

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.*;

import javax.imageio.ImageIO;
import javax.swing.*;
import javax.swing.filechooser.FileNameExtensionFilter;

public class PhotoViewerController implements ActionListener {
    int currentImage = 0;

    static PhotoViewerLayout photoView = new PhotoViewerLayout();
    static PhotoCollection collection = new PhotoCollection();
    int maxImages = collection.getCollectionSize();
    private final JFileChooser fc = new JFileChooser();

    public PhotoViewerController(){
        readCollection();
        maxImages = collection.getCollectionSize();
        if(maxImages <= 0){
            photoView.disableNext();
            photoView.disablePrev();
        }
        photoView.updatePhotoCount(maxImages);
        photoView.updatePhotoNum(currentImage + 1);
        System.out.print(collection.getCollectionSize());
        maxImages = collection.getCollectionSize();
        photoView.disablePrev();
        if (currentImage > 0) {
            photoView.disableNext();
            photoView.disableDelete();
        }
        else if (collection.getCollectionSize() > 0)
photoView.updateDisplayPhoto(collection.getPhoto(0));


        photoView.deleteAddActionListener(new ActionListener() {
//delete action
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                collection.deletePhoto(currentImage);
                if (currentImage == maxImages - 1 && maxImages > 1) {
                    currentImage--;

photoView.updateDisplayPhoto(collection.getPhoto(currentImage));
                    maxImages = collection.getCollectionSize();
                    photoView.updatePhotoCount(maxImages);
                    photoView.updatePhotoNum(currentImage + 1);
                } else if (collection.getCollectionSize() == 0) { //if we're
deleting the last photo,
                    photoView.updateDisplayPhoto(new Photograph());
                    photoView.disableDelete();
                    maxImages = collection.getCollectionSize();
                    photoView.updatePhotoCount(maxImages);
                    photoView.disableNext();
                    photoView.disablePrev();
                    photoView.updatePhotoNum(currentImage + 1);
                } else {

photoView.updateDisplayPhoto(collection.getPhoto(currentImage));
                    maxImages = collection.getCollectionSize();
                    photoView.updatePhotoCount(maxImages);
                    photoView.updatePhotoNum(currentImage + 1);
                }
                if (currentImage == 0) {
                    photoView.disablePrev();
                    if (maxImages == 1)
                        photoView.disableNext();
                }

            }
        });

        photoView.addAddActionListener(new ActionListener() {           //add
action
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                JFileChooser chooser = new JFileChooser();
                FileNameExtensionFilter filter = new FileNameExtensionFilter(
                        "Images", "jpg", "gif", "bmp", "png");
                fc.setFileFilter(filter);
                int returnVal = fc.showOpenDialog(fc);
                BufferedImage img = null;
                if (returnVal == JFileChooser.APPROVE_OPTION) {
                    File file = fc.getSelectedFile();
                    try {
                        img = ImageIO.read(file);
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                    ImageIcon image = new ImageIcon(img);
                    collection.addPhoto(image);
                    maxImages = collection.getCollectionSize();
                    photoView.updatePhotoCount(maxImages);
                }
                photoView.enableDelete();
            }
        });

        photoView.saveAddActionListener(new ActionListener() {
//save action
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                writeCollection();
            }
        });

        photoView.nextAddActionListener(new ActionListener() {
//next button
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                collection.setDescription(currentImage,
photoView.getDescField());
                currentImage++;
                photoView.enablePrev();
                if (currentImage == maxImages - 1) photoView.disableNext();

photoView.updateDisplayPhoto(collection.getPhoto(currentImage));

photoView.setDescField(collection.getDescription(currentImage));
                photoView.updatePhotoCount(maxImages);
                photoView.updatePhotoNum(currentImage + 1);

                System.out.print(photoView.getDescField());
            }


        });

        photoView.prevAddActionListener(new ActionListener() {          //prev
button
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                //save desc&date
                collection.setDescription(currentImage,
photoView.getDescField());
                currentImage--;
                photoView.enableNext();
                if (currentImage == 0) photoView.disablePrev();

photoView.updateDisplayPhoto(collection.getPhoto(currentImage));

photoView.setDescField(collection.getDescription(currentImage));
                photoView.updatePhotoCount(maxImages);
                photoView.updatePhotoNum(currentImage + 1);
            }
        });

        photoView.browseAddActionListener(new ActionListener() {
//browse menu
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                photoView.disableMaint();
            }
        });

        photoView.maintAddActionListener(new ActionListener() {
//maint menu
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
photoView.enableMaint(); }
```

```
        });

    }

    public JFrame getFrame(){
        return photoView;
    }

    private void readCollection(){
        try {
            FileInputStream fileIn = new FileInputStream("collection.ser");
            ObjectInputStream ObjIn = new ObjectInputStream(fileIn);
            collection = (PhotoCollection)ObjIn.readObject();
        } catch (FileNotFoundException e) {
            System.out.print("FILE IN NOT FOUND!");
        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    private void writeCollection(){
        try {
            FileOutputStream fileOut = new
FileOutputStream("collection.ser");
            ObjectOutputStream objOut = new ObjectOutputStream(fileOut);
            objOut.writeObject(collection);
            objOut.close();
            System.out.print("Collection serialized to \"collection.ser\"");
        } catch (IOException i) {
            System.out.print("FILE OUT NOT FOUND\n");
        }
    }
    @Override
    public void actionPerformed(ActionEvent actionEvent) {

    }
    public static void main(String[] args) {
        JFrame frame = new PhotoViewerController().getFrame();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.pack();
        frame.setVisible(true);
    }
}
```

PhotoViewerLayout.java

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.InputStreamReader;

import javax.swing.*;

public class PhotoViewerLayout extends JFrame{

    JButton deleteButton = new JButton("Delete");
```

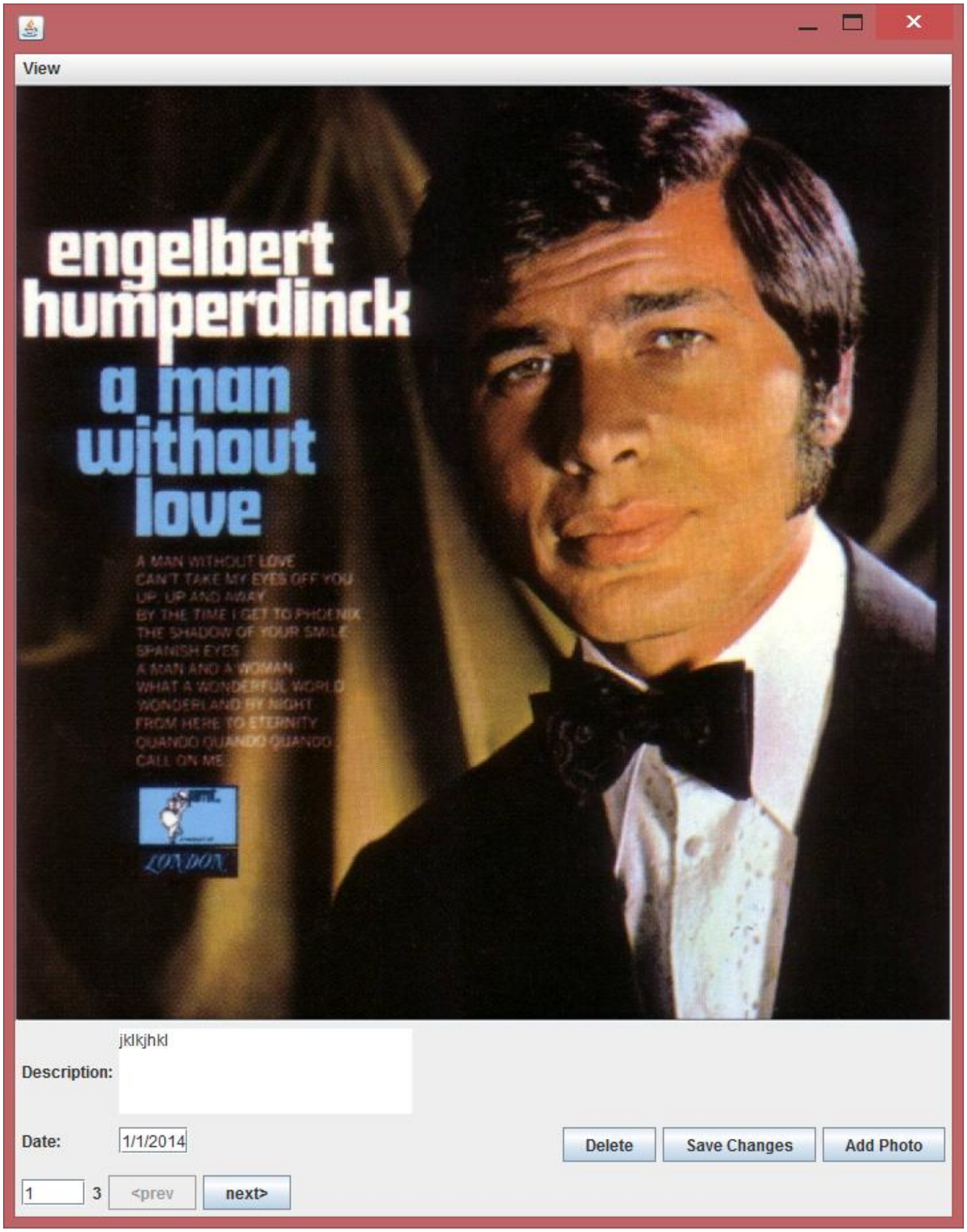
```java
    JButton saveButton = new JButton("Save Changes");
    JButton addButton = new JButton("Add Photo");
    JButton prevButton = new JButton("<prev");
    JButton nextButton = new JButton("next>");
    ImageIcon image = new ImageIcon();
    JLabel imageLabel = new JLabel("", SwingConstants.CENTER);
    JMenuItem browseMode = new JMenuItem("Browse Mode");
    JMenuItem maintMode = new JMenuItem("Maintenance Mode");
    JTextField pictureNumberTextField = new JTextField("3",4);
    JLabel pictureCountLabel = new JLabel(" of 3");
    JTextField dateTextField = new JTextField("1/1/2014");
    JTextArea descriptionTextArea = new JTextArea("jklkjhkl",4,20);


    public PhotoViewerLayout() {

        Container contentPane = getContentPane();
        descriptionTextArea.setEditable(true);
        JScrollPane scrollPane = new JScrollPane(imageLabel);

        JMenuBar menuBar = new JMenuBar();                                     //menu
        setJMenuBar(menuBar);
        JMenu viewMenu = new JMenu("View");
        menuBar.add(viewMenu);

        viewMenu.add(browseMode);

        viewMenu.add(maintMode);


        imageLabel.setIcon(image);

        contentPane.add(scrollPane, BorderLayout.CENTER);

        JPanel controlPane = new JPanel();
        controlPane.setLayout(new BoxLayout(controlPane, BoxLayout.PAGE_AXIS));

        JPanel descriptionPane = new JPanel();
        descriptionPane.setLayout(new FlowLayout(FlowLayout.LEFT));

        JLabel descriptionLabel = new JLabel("Description:");
        descriptionPane.add(descriptionLabel);
        descriptionPane.add(descriptionTextArea);

        JPanel datePane = new JPanel();

        JLabel dateLabel = new JLabel("Date:");
        dateLabel.setPreferredSize(new
Dimension(descriptionLabel.getPreferredSize().width,dateLabel.getPreferredSize().height));

        datePane.add(dateLabel);
        datePane.add(dateTextField);
        JPanel buttonPane = new JPanel();

        buttonPane.add(deleteButton);
        buttonPane.add(saveButton);
        buttonPane.add(addButton);

        JPanel leftRightPane = new JPanel();
        leftRightPane.setLayout(new BorderLayout());
        leftRightPane.add(datePane,BorderLayout.WEST);
        leftRightPane.add(buttonPane,BorderLayout.EAST);


        JPanel southButtonPanel = new JPanel();



        southButtonPanel.add(pictureNumberTextField);
        southButtonPanel.add(pictureCountLabel);
        southButtonPanel.add(prevButton);
        southButtonPanel.add(nextButton);
        FlowLayout flowLayout = (FlowLayout) southButtonPanel.getLayout();
        flowLayout.setAlignment(FlowLayout.LEFT);


        controlPane.add(descriptionPane);
        controlPane.add(leftRightPane);
        controlPane.add(southButtonPanel);

        contentPane.add(controlPane, BorderLayout.SOUTH); // Or PAGE_END
    }

    public void updateDisplayPhoto (Photograph photo){
        image = photo.getImage();
        imageLabel.setIcon(image);

    }
    public void updateDisplayPhotoTest (ImageIcon photo){
        image = photo;
        imageLabel.setIcon(image);

    }

    public void updateDescription(String desc){
        descriptionTextArea.setText(desc);
    }

    public void disablePrev(){prevButton.setEnabled(false);}

    public void enablePrev(){prevButton.setEnabled(true);}

    public void disableNext(){nextButton.setEnabled(false);}

    public void enableNext(){nextButton.setEnabled(true);}

    public void disableDelete() {deleteButton.setEnabled(false);}

    public void enableDelete() {deleteButton.setEnabled(true);}

    public void enableMaint(){
        deleteButton.setEnabled(true);
        addButton.setEnabled(true);
        saveButton.setEnabled(true);
    }

    public void disableMaint(){
        deleteButton.setEnabled(false);
        addButton.setEnabled(false);
        saveButton.setEnabled(false);
    }

    public void updatePhotoCount(int count) {
        pictureCountLabel.setText(Integer.toString(count));
    }

    public void updatePhotoNum(int count) {
        pictureNumberTextField.setText(Integer.toString(count));
    }

    public void deleteAddActionListener(ActionListener al){
        deleteButton.addActionListener(al);
    }

    public void addAddActionListener(ActionListener al){ addButton.addActionListener(al); }

    public void saveAddActionListener(ActionListener al){ saveButton.addActionListener(al); }

    public void prevAddActionListener(ActionListener al){ prevButton.addActionListener(al); }

    public void nextAddActionListener(ActionListener al){ nextButton.addActionListener(al); }

    public void browseAddActionListener(ActionListener al){ browseMode.addActionListener(al);}

    public void maintAddActionListener(ActionListener al){ maintMode.addActionListener(al);}

    public String getDescField(){
        return descriptionTextArea.getText();
    }

    public void setDescField(String string){
        descriptionTextArea.setText(string);
    }
}
```

PhotoCollection.java

```
import javax.swing.*;
import java.io.*;
import java.util.ArrayList;

/**
 * Created by EW043872 on 10/4/2015.
 */
public class PhotoCollection implements Serializable{
    public ArrayList<Photograph> photos = new ArrayList<>();

    public PhotoCollection(){

}



    public void addPhoto(ImageIcon image){
        photos.add(new Photograph(image));
    }

    public void deletePhoto(int index){
        photos.remove(index);
    }

    public Photograph getPhoto(int index){
        return photos.get(index);
    }

    public int getCollectionSize(){
        return photos.size();
    }

    public void setDescription(int index, String desc){
        photos.get(index).description = desc;
    }

    public String getDescription(int index){
        return photos.get(index).description;
    }
}
```

Photograph.java

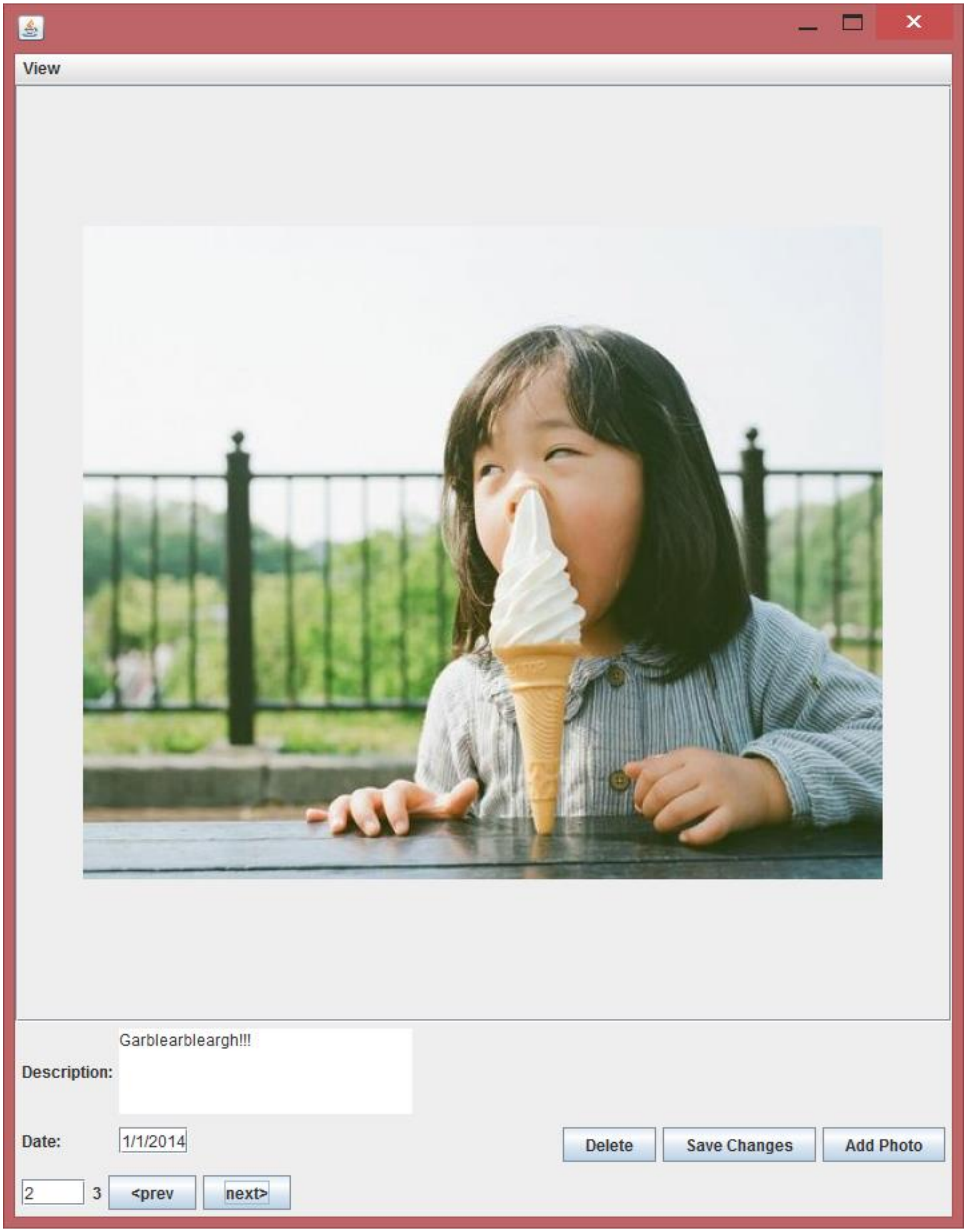
```
import javax.swing.*;
import java.io.Serializable;

/**
 * Created by EW043872 on 10/6/2015.
 */
public class Photograph implements Serializable {
    ImageIcon image = new ImageIcon();
    String date = new String("");
    public String description = new String("");

    Photograph(ImageIcon img){
        image = img;
    }
    Photograph(){}

    public void setImage(ImageIcon img){
        image = img;
    }

    public ImageIcon getImage(){
        return image;
    }

    public void setDate(String dat ){
        date = dat;
    }

    public String getDate(){
        return date;
    }

    public void setDescription(String desc){
        description = desc;
    }

    public String getDescription(){
        return description;
    }


}
```



Description: jklkjhkl

Date: 1/1/2014

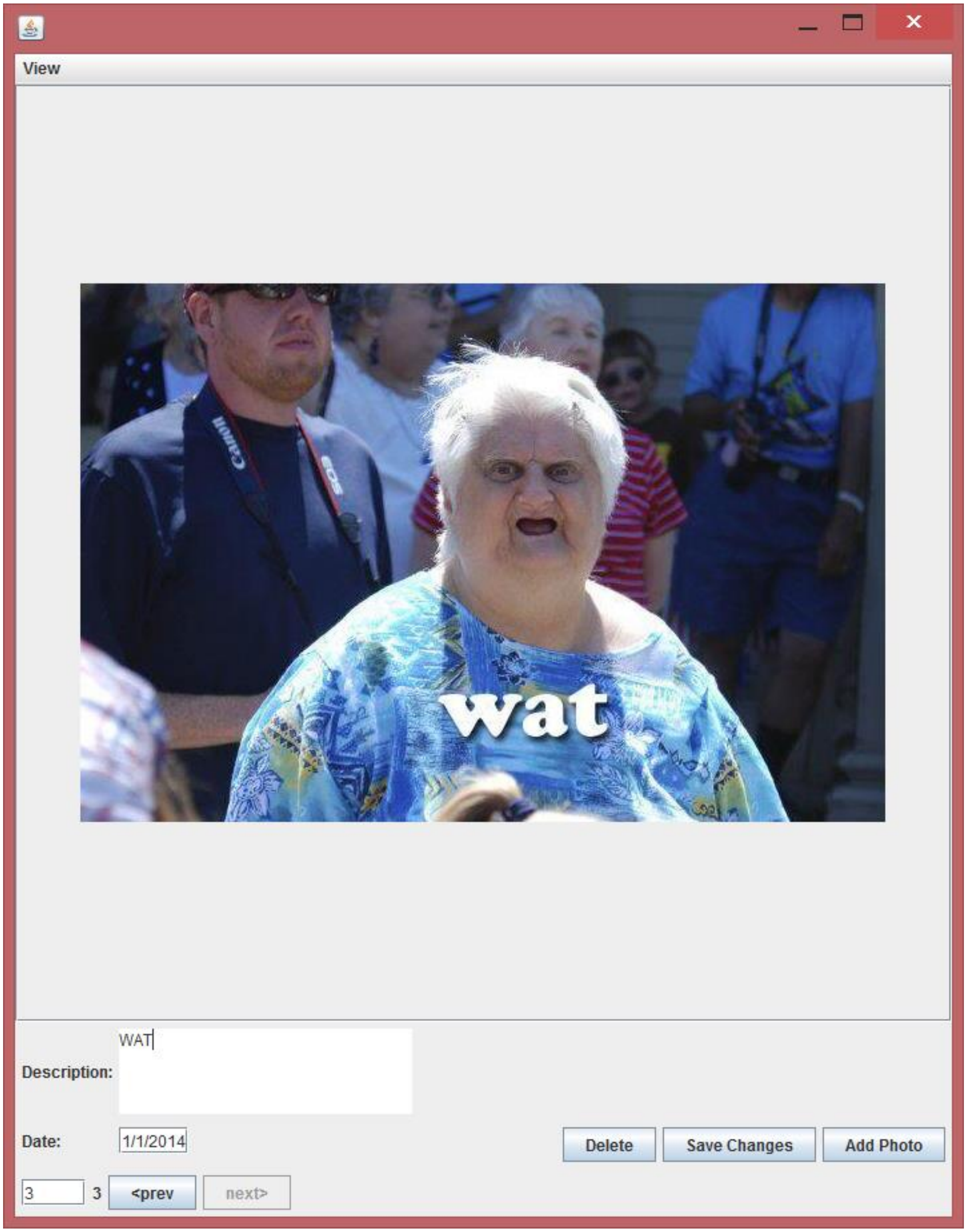
1 3



Description: Garblearbleargh!!!

Date: 1/1/2014

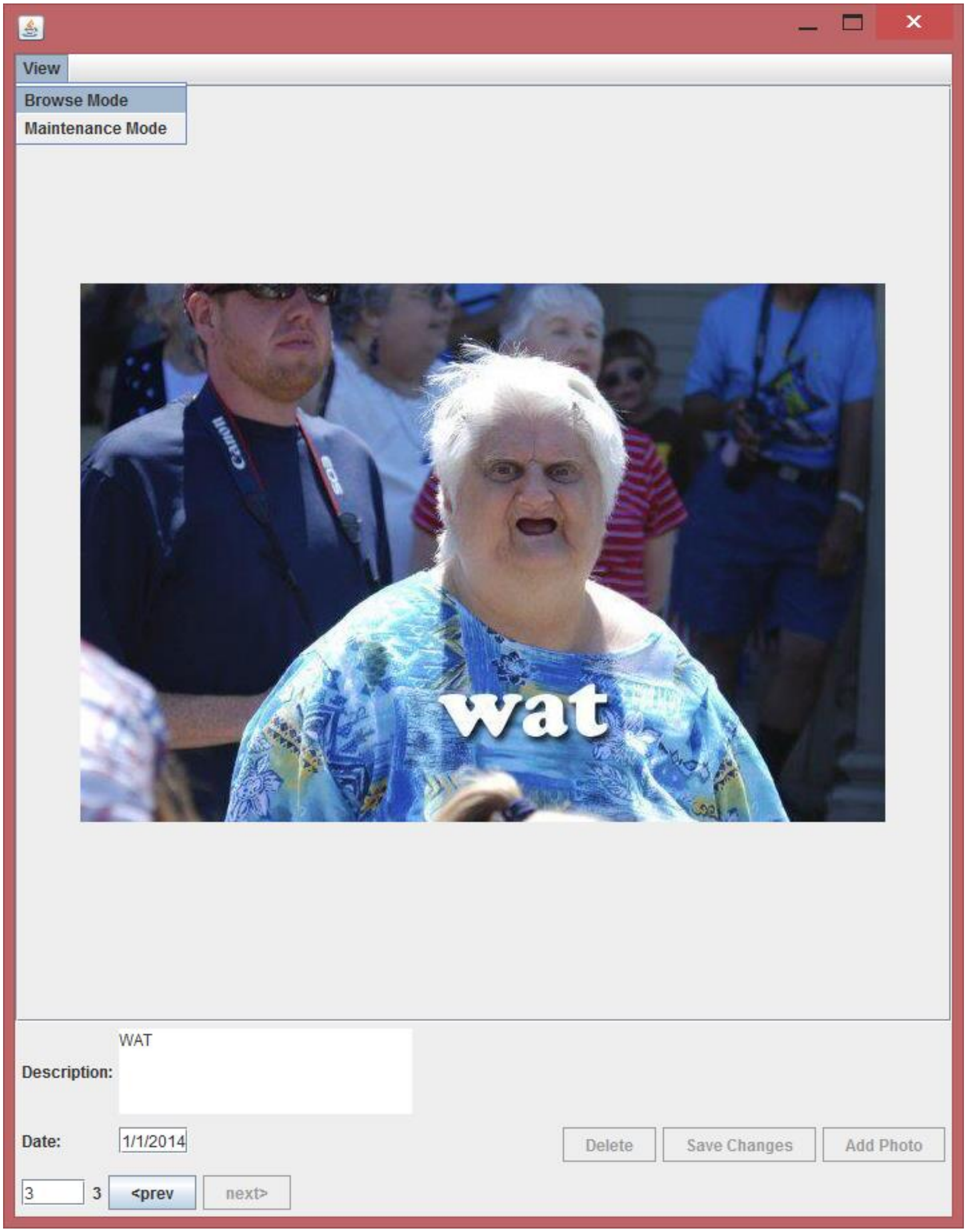
2



Description: WAT

Date: 1/1/2014

3 3



Description: WAT

Date: 1/1/2014

3 3