Moving Category-Matching Files from Directory1 to Directory2


SWFileFilterScreen.java


ResourceFilePaths.java


ReadFileToArrayList.java


GetUserDrive.java


CreateOrigDirListing.java


CategoriesArrayList.java


TextFileNamesWriter.java

SWFileFilterScreen.java:

```
import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.awt.event.ActionEvent;

public class SWFileFilterScreen {

	private JFrame frame;

	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					SWFileFilterScreen window = new SWFileFilterScreen();
					window.frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}

	/**
	 * Create the application.
	 */
	public SWFileFilterScreen() {
		initialize();
	}

	/**
	 * Initialize the contents of the frame.
	 */
	private void initialize() {
		frame = new JFrame();
		frame.setBounds(100, 100, 450, 490);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.getContentPane().setLayout(null);

		JButton btnInitProcess = new JButton("Initiate File Filtering");
		btnInitProcess.addActionListener(new ActionListener() {
```

```
                        public void actionPerformed(ActionEvent e) {
                                // button clicked
                                ResourceFilePaths pathsResource = new ResourceFilePaths();

                                // put into the final filter test class where the filesList is read item-by-
item for each category (TO DO):
                                CategoriesArrayList categoriesArr = new CategoriesArrayList();
                                ArrayList<String>categoriesList =
categoriesArr.loadCategoryArrayList();
                                CreateOrigDirListing createListing = new CreateOrigDirListing();

        createListing.writeOrigDirectoryFiles(pathsResource.getOriginalDirFileWriterPath());

                        }
                });
                btnInitProcess.setBounds(144, 131, 127, 23);
                frame.getContentPane().add(btnInitProcess);

                JButton btnTestZone = new JButton("Run Test");
                btnTestZone.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                // btnTestZone clicked - run test
                                GetUserDrive driveLetter = new GetUserDrive();
                                System.out.println("The Drive Letter = " +
driveLetter.getDriveLetter());
                        }
                });
                btnTestZone.setBounds(170, 200, 89, 23);
                frame.getContentPane().add(btnTestZone);
        }
}

>
```

ResourceFilePaths.java

```
public class ResourceFilePaths {
        // path to Categories file:
        public String getCategoriesFilePath() {
                String filePath = getUserDriveLetter() + "Categories/Categories.txt";
                return filePath;
        }

        // path for file writer to place all the original file name from pdfOriginal/
        public String getOriginalDirFileWriterPath() {
                String filePath = getUserDriveLetter() + "OriginalFilesRpt/Report.txt";
```

```
                return filePath;
        }

        // pdfOriginal/ directory
        public String getOriginalDirectory() {
                String directory = getUserDriveLetter() + "pdfOriginal/";
                return directory;
        }

        // pdfMoveTo/ directory
        public String getMoveToDirectory() {
                String directory = getUserDriveLetter() + "pdfMoveTo/";
                return directory;
        }

        public String getUserDriveLetter() {
                String drive;
                GetUserDrive driveLetter = new GetUserDrive();
                drive = driveLetter.getDriveLetter();
                return drive;
        }

}

>
```

ReadFileToArrayList.java

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;

public class ReadFileToArrayList {

        public ArrayList<String> readBufferedFile(String pTestFile)  {

            // Declaring a string and initializing it with
            // path of file present on the system
                ArrayList<String> arrayList = new ArrayList<String>();
            String path = pTestFile;

            // Try block to check for exceptions
            try (BufferedReader br
                = new BufferedReader(new FileReader(path))) {
```

```
                // Declaring a new string
                String str;

                // It holds true till there is content in file
                while ((str = br.readLine()) != null) {
                    //put into a list:
                    arrayList.add(str);
                    // Printing the file data
                    //System.out.println(str);
                }
            }

            // Catch block to handle the exceptions
            catch (IOException e) {

                // Display pop up message if exceptionn occurs
                System.out.println(
                    "Error while reading a file.");
            }
            return arrayList;
        }
    }

>
```

TextFileNamesWriter.java

```
import java.io.FileWriter;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.time.LocalDate;

public class TextFileNamesWriter {
    //Open new file and add the headings
    public String filePath;

    public void setFilePath(String pFilePath) {
        this.filePath =  pFilePath;
    }

    private String getFilePath() {
        return this.filePath;
    }

    public void writePreambleWithTitles( ) {
        LocalDate myDTS = LocalDate.now();
```

```
		try (var fw = new FileWriter(getFilePath(), StandardCharsets.UTF_8,false)) {
			fw.write("Sheet Music PDF File copy/move process run on " + myDTS + "\n");
			fw.close();
		} catch (IOException e) {
			e.printStackTrace();
		}
	}

	public void createNewFile( ) {
		LocalDate myDTS = LocalDate.now();
		try (var fw = new FileWriter(getFilePath(), StandardCharsets.UTF_8,false)) {
			fw.close();
		} catch (IOException e) {
			e.printStackTrace();
		}
	}

	public void writeRow(String textLine) {
		try (var fw = new FileWriter(getFilePath(), StandardCharsets.UTF_8,true)) {
			fw.write(textLine + "\n");
			fw.close();
		}
		catch (IOException e) {
			e.printStackTrace();
		}
	}

	public void writeClosing() {
		try (var fw = new FileWriter(getFilePath(), StandardCharsets.UTF_8,true)) {
			// fw.write(".....End File\n");
			// fw.close();
			ProcessBuilder pb = new ProcessBuilder("Notepad.exe", getFilePath());
			pb.start();
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
<
```

GetUserDrive.java

```
import java.io.File;

public class GetUserDrive {

   public String getDriveLetter() {
                        String userDir = "";
                        File file = new File(".").getAbsoluteFile();
                        File root = file.getParentFile();
                        while (root.getParentFile() != null) {
                           root = root.getParentFile();
                        }
                        userDir = root.getPath();
                        String s = userDir.substring(0, userDir.length() - 1);

                        System.out.println("Drive is: "+s + "/");
                        return s + "/";
          }
}

>
```

CreateOrigDirListing.java

```
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;

public class CreateOrigDirListing {
          public void writeOrigDirectoryFiles(String pOriginalFilesRptPath) {
                        TextFileNamesWriter dirFileNameRpt = new TextFileNamesWriter();

                        ResourceFilePaths pathResource = new ResourceFilePaths();
                        dirFileNameRpt.setFilePath(pathResource.getOriginalDirFileWriterPath());
                        //dirFileNameRpt.writeRow("first row");
                        //dirFileNameRpt.writeRow("second row");

               File directoryPath = new File(pathResource.getOriginalDirectory());
               //List of all files and directories in directoryOfFiles:
               File filesList[] = directoryPath.listFiles();
               System.out.println("List of files and directories in the specified directory:");
```

```
            //create new append file:
            dirFileNameRpt.createNewFile();
            int iFileCounter = 0;
            for(File file : filesList) {
                    iFileCounter++;
                    dirFileNameRpt.writeRow(file.getName());
                    System.out.println(iFileCounter + "." + file.getName());
                    }
            System.out.println("Beginning cycling.....");
            cycleOriginalDirectoryFiles();
        }


        private void cycleOriginalDirectoryFiles() {
                ResourceFilePaths pathResource = new ResourceFilePaths();
                File directoryPath = new File(pathResource.getOriginalDirectory());
                File filesList[] = directoryPath.listFiles();

            int iFileCounter = 0;
            for(File file : filesList) {
                    iFileCounter++;
                    System.out.println("IN cycleOriginalDirectoryFiles - " + iFileCounter + " " +
file.getAbsolutePath().toLowerCase().toString());
                    cycleCategoriesFiltration(file.getName().toLowerCase().toString());
            }
        }

        public void cycleCategoriesFiltration(String pFileNameToTest) {
                CategoriesArrayList categoryStreamer = new CategoriesArrayList();
                ArrayList<String> catArrayList = new ArrayList<String>();
                catArrayList = categoryStreamer.loadCategoryArrayList();

                int iCategoryCount = 0;
                //String category;
                for(String category : catArrayList) {
                        iCategoryCount++;
                        if(pFileNameToTest.contains(category.toLowerCase())) {
                                moveFilteredFile(pFileNameToTest);
                                System.out.println(">>>>>> Matched and Called
moveFilteredFile(file " + pFileNameToTest + ") matched category " + category);
                                break;
                        }
                }

        }


        private void moveFilteredFile(String fileName) {
```

```
                System.out.println("IN moveFilteredFile - " + fileName);
                ResourceFilePaths resourceFilePaths = new ResourceFilePaths();
                Path source = Paths.get(resourceFilePaths.getOriginalDirectory() + fileName);

                Path target = Paths.get(resourceFilePaths.getMoveToDirectory() + fileName);

                try{
                   Files.move(source, target);
                 } catch (IOException e) {
                   e.printStackTrace();
                 }
        }
}

>
```

CategoriesArrayList.java

```
import java.util.ArrayList;

public class CategoriesArrayList {
        public ArrayList<String> loadCategoryArrayList() {
                ArrayList<String> categoryArrList = new ArrayList<String>();
                ReadFileToArrayList streamCategories = new ReadFileToArrayList();
                ResourceFilePaths resourcePaths = new ResourceFilePaths();
                categoryArrList =
streamCategories.readBufferedFile(resourcePaths.getCategoriesFilePath());

                //System.out.println("Number of categories = " + categoryArrList.size() );

                return categoryArrList;
        }

}

</> End
```