12/20/2018

imageJoin

Report Submitted to

Prof. Amey Karkare
Computer Science and Engineering
Indian Institute of Technology Kanpur

by

Mr. K. Vamsee Krishna
B.Tech, 3rd Year
Computer Science and Engineering
Visvesvaraya National Institute of Technology Nagpur

Contents	Page No

Acknowledgement

I sincerely thank Professor Amey Karkare for giving me an opportunity to work under his guidance during the winter break (December, 2018), in IIT Kanpur. I really learned the art of making software for a nice and important problem given by Professor Amey Karkare. Thanks also due to for his wonderful discussions on code development to make the software. The task has been fulfilled by learning Java to develop the code, which we named it as imageJoin. I hereby dedicate this software imageJoin to my beloved Professor Amey Karkare.

K. Vamsee Krishna
B.Tech 3rd Year
Computer Science and Engineering
Visvesvaraya National Institute of Technology Nagpur

imageJoin Software

Key Task of the Project: Combining scanned images (for example, answer booklets[1]) through a semi-automated software.

The necessary Software to combine the images with the requirements below has been made using Java.

Requirements:

First and foremost, the framework of the software. The Software should be able to import the image as an input from a specified file-path. The file path can by typed-in from the user front or the user can access the file(s) by clicking the file chooser.

The software should be able preview the image in the specified image-panel and user should be able to give input to the respective fields (for example, the fields Class, Subject, Student ID) by seeing the image that is loaded in the image panel.

The user should be able to go through and preview the images back and forth by clicking the next and back buttons. To create a combined single file (for example a pdf) out of the selective images can be chosen by a add button. Why selective? There may be some corrupted or unwanted image files and these images can be skipped by not clicking the add button.

By mistake if the user adds an extra image (it may be corrupted, unwanted like the image does not belong to the image-cluster for which the single combined pdf file needs to be created), such image can be deleted by a delete button.

And therefore, as per the user's concern the software should be able to combine the images into a single file (pdf) and save the combined file in the path specified by the user.

As per the user requirement, the user can add or delete the images when and as required before combining the images into single pdf file.

After combining the images into a single file, the output filename can be created from the data in the fields. Therefore, the file name is read as **Class_Subject_StudentID.pdf** i.e., the software should be able to automatically save the filename as **Class_Subject_StudentID.pdf**.

Once the combined file has been created the entry in the fields should be made empty and ready for the next input.

[1] The images are ordered in sequence

During the task of creating the single pdf file out of the images, if the user closes the software window by mistake or intentionally or by any error (power shortage, etc,.) the software should be able to resume and preview the file next to the last image in the previously combined image cluster in the image preview panel.

Interface Design:
As per the requirements, the model consists of `

1) Text Fields for
   - File Input Path, File Output Path, Information (Image Name, Info.).
   - Attributes like Class, Subject, Student-ID.
2) Buttons like
   - Open
   - Next, Back
   - Add, Delete
   - Check
   - Combine
   - Save Progress
   - Resume Progress
   - Minimize, Maximize, and Exit.
   - Cancel, Exit-Without-Saving, Save & Exit.

Such an interface design has been implemented in Java with all the functionalities discussed above. A screen of the software, which we call as imageJoin is shown below.

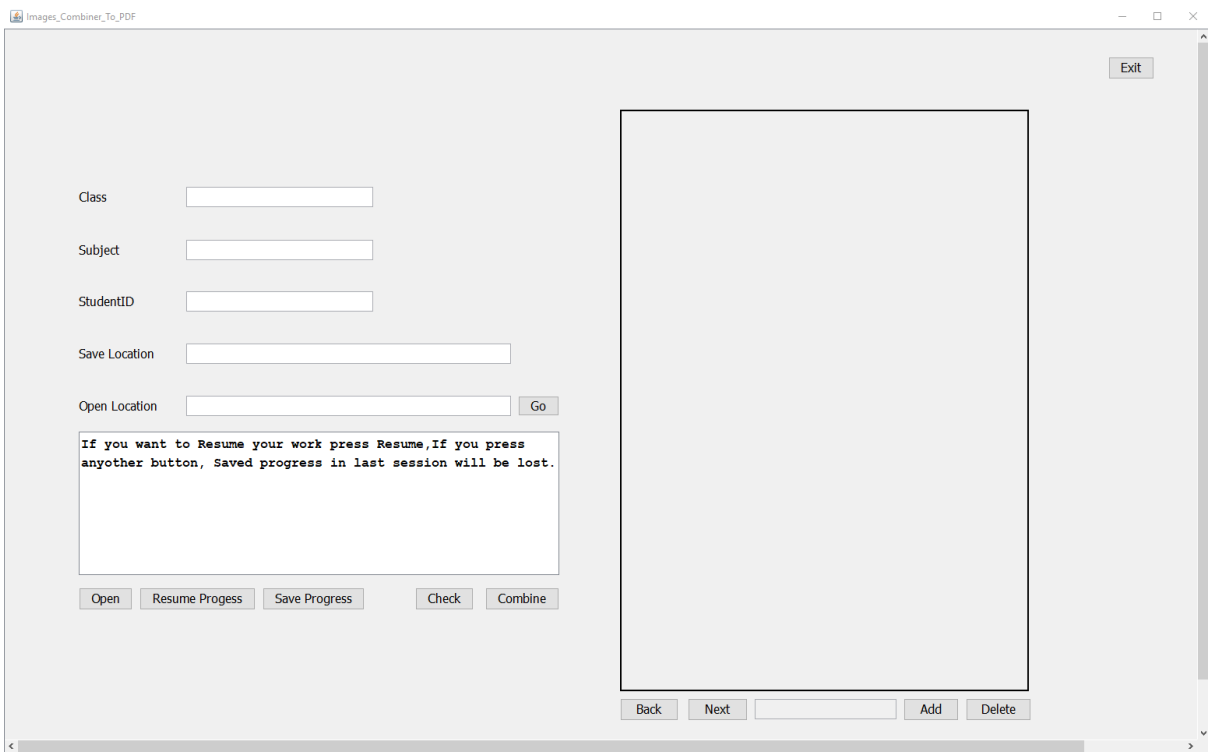

Figure 1. Screenshot of the imageJoin

Software Implementation through Java: In this implementation, it requires to design a suitable Java-frame (J-Frame) consisting of all the required fields, image preview panel and buttons as shown Figure 1. The Details of each component in J-Frame and its implementation is discussed below.

Designing the Java-Frame: The Java-Frame has been designed using NetBeans IDE (Integrated Development Environment.). The J-Frame as shown in Figure 1 is designed in a drag and drop manner.

Adding minimizing, maximizing, and Exit buttons: The minimizing, maximizing, and closing (inactive) buttons will be created by default with J-Frame, some lines of the code are added in the code block of the Exit button for the purpose of remembering the image next to last image in the cluster of previously combined images before the closing of the software. Closing of the software can only be done by Exit button.

Adding a file chooser: The file chooser for opening the file is added in this frame by linking the J-File-chooser with the Open button hence we can access the folder by simply clicking the Open button and browsing through the directories. The action of this activity is shown in the below as a screenshot.
The file chooser to select the Destination folder for the combined pdf to save is added in the same way as explained above.

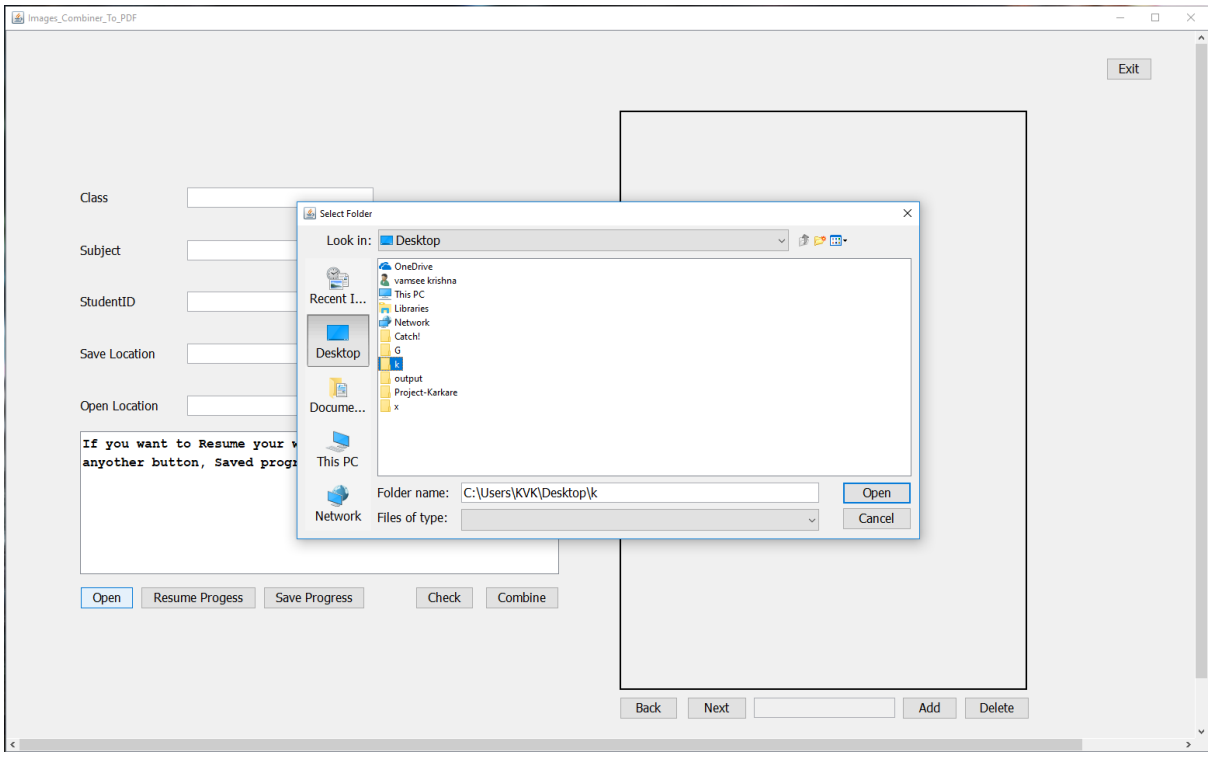

Figure 2. Screenshot showing the activity of the File-chooser.

Fields in the J-Frame: Name, Roll No etc: The Fields for the data such as the Open Location, Save Location, Class, Subject, Student-ID, Information field (below the Open Location Field in Figure 1) are added by using J-Label to name the fields visually on the panel and using the J-Text-Field to take the input from the respective fields and to display the respective information.

Preview Pane: The image preview pane is used to show the input image, which has been added by creating a J-Label dedicated only to display the image. This will display the first image of the selected folder for the first time.

Buttons: Open, Go, Back, Next, Add, Delete, Combine, Check, Save Progress, Resume Progress, Exit: These buttons are added by inserting the required number of J-Buttons and a layout similar to the one in Figure-1.

Back and Next Buttons: On clicking next and back buttons one would go to the next and previous elements (images), respectively. This was done by keeping track of the files in the given folder using Array of type String.

Check Button: On Clicking the Check button, the information of the images which are stored in the data structure (Array-List<>) by adding with an add button are displayed in the Information field.

Add and Delete: The add button stores the data (path) of the image which is displaying in the preview pane. By clicking the add button the path of the image file will be stored in some data-structure (Array-List<>). Likewise, if the image is unwanted, the delete button removes the data (path) of the image which is displayed in the preview pane. From the data-structure, if the data to be deleted is present in the data-structure, it will delete otherwise it will prompt a message.

File Saving to a location: On clicking the combine button, it combines all the files stored in the data-structure and generate a single combined pdf file. This action will also popups a window that will ask to select a folder to save the combined single pdf file (CSPF). After saving the combined single pdf file, the data in the data-structure will be deleted (by ArrayList.clear() method) and the data structure (Array-List<>) is ready to store the images for the next combined single pdf file.

The File Chooser to select the save destination occurs only, if the save location field is empty, and the other times the save destination folder will be the previously selected one or manually typed one, if user wants to change the save destination in between the session he/she can do it manually by editing the path in save location field or clear that and press combine button and therefore, the file chooser will appear again.

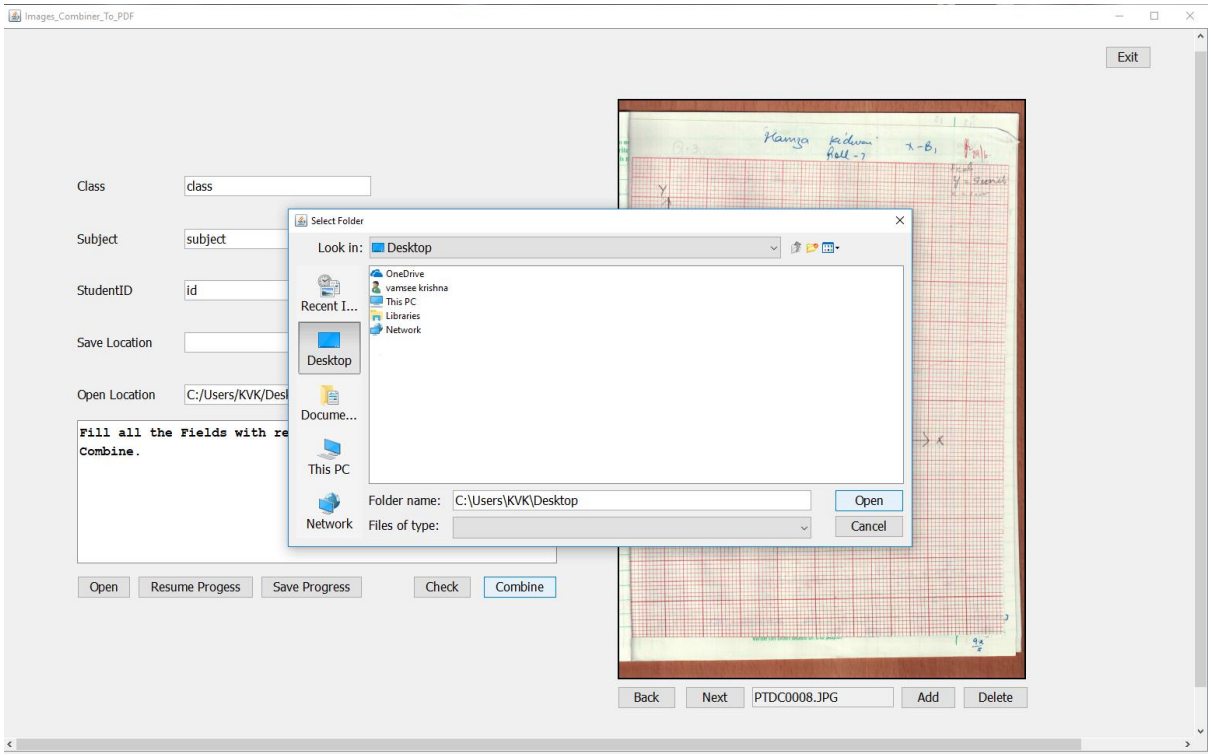

Figure 3. Screenshot showing the activity of the File-chooser for the save destination.

Task of Combining and Naming: On clicking the combine button and having the path of the output location and the attributes, the data-structure is accessed by another function which is invoked in sequence that actually does the combining process of the image files and saves at the specified output location with the file name as "Class_Subject_StudentID.pdf", as shown in the Figure 4, below.

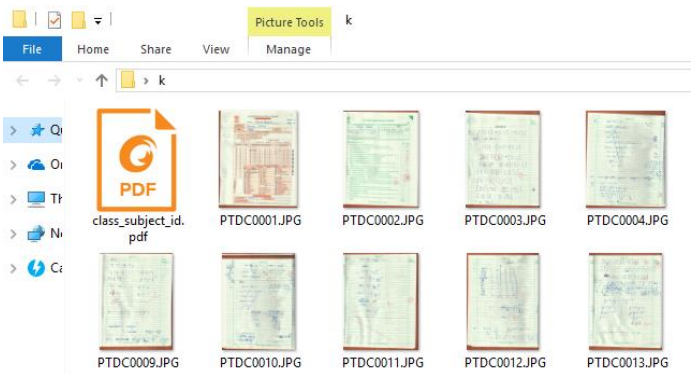

Figure 4. Screenshot showing the activity of saving the file in a desired location with the file-name as the combination of the attributes.

Erasing the Data in the fields after joining the images into a single file: On clicking the combine button, as usual a single pdf file will be generated and saved. Thereafter, the attributes in the fields like Class, Subject, Student-ID will be erased by invoking a method, VarName.setText("").

How to Save Progress*:

1) On Clicking the Save-Progress button in the J-Frame.
2) On Clicking Exit button, a dialog box will popup. On Clicking Save & Exit button in the Dialog box before closing the software.

On performing the above mentioned action(s), the imageJoin saves the data(path) of the last image in the previously combined cluster of images into an external file using File-Writer and Buffered-Writer.

* User has to save anyway as mentioned above to avoid loss of work progress before anything happens.

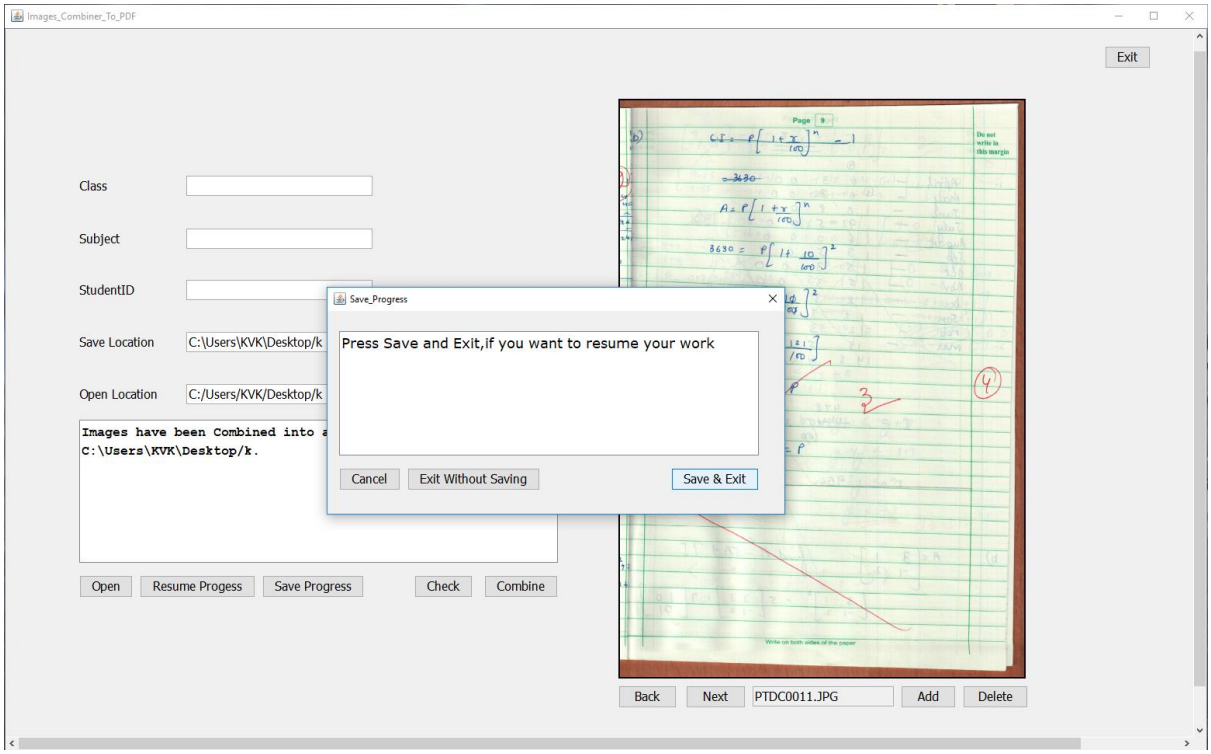

Figure 5. Screenshot showing the dialog box before closing the imageJoin.

Save Progress Dialog Box:

On Clicking the Exit Without Saving button, clears the previously saved progress (if any) and closes the application.

If the User Clicked the Exit button by mistake and wants to go back to work, he can do it by clicking the Cancel button, the software does not close or by clicking the traditional close button of the Dialog Box.

How to Resume: On clicking the Resume-Progress button it starts at the file location next to the last image in the previously combined file. It reads the path previously stored in the external file (by saving) using File-Reader and Buffered-Reader and shows the image.

Before Resuming, if the User opts to open a new directory by Clicking Open or by entering path manually in open location field, the previously saved progress (if any) will get cleared and user will not be able to resume the saved progress from the last session.

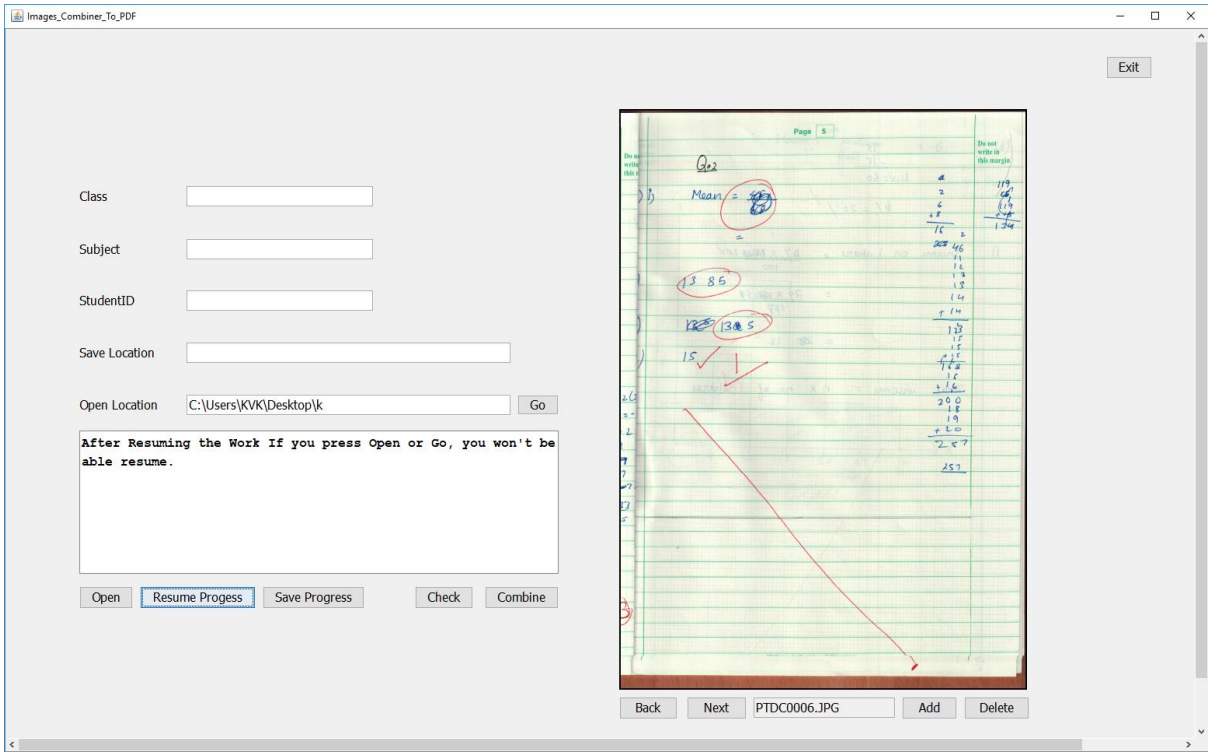

Figure 6. Screenshot showing the activity of Resuming the work progress.

System Requirements: This Software is compatible with system having

- **Java 8** (jdk 1.8_151 or jre 1.8_151) and above.
- **Screen resolution 1600 x 900 and above, scaled to 100%.**

----------End of the Report ----------