Objectives

To develop a java application that

1. Will download like audio, video, pdf, ppt and any type of files at a time.
2. Will be able to use in all operating systems.

Introduction

A download manager is a software tool that manages and schedule the downloading of files from the internet, which may be built into a web browser. It can use full bandwidth. It has recovery and resume capabilities to restore the interrupted downloads due to lost connection, network issues and power outages. It supports a wide range of proxy servers such as firewall, MP3 audio, MP4 or MPEG video processing and so on. It efficiently collaborates with web browser like Google Chrome, Mozilla Firefox, Internet Explorer, Opera and many other popular browsers to manage the download.

The programming language that I used for this application is "JAVA". It is a high-level, class-based, object-oriented programming language that is designed to have as implementation dependencies as possible. It is a general-purpose programming language intended to application developers write once, run anywhere, meaning that compiled java code can run on all platforms that support Java without the need for recompilation. That means it is a Platform Independent Language. For this reason, this application will be able to run in all operating systems.

An IDE(Integrated Development Environment) is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of at least a source code editor, build automation tools and a debugger. Some IDEs, such as NetBeans, Codeblocks and Eclipse contain necessary compiler, interpreter or both while some other IDEs such as SharpDevelop do not contain these things. There are many IDEs for doing such java applications. Among them, I choose "Eclipse" IDE, which is a very popular and efficient ide, where my jdk version is 16.

Implementation

This is a prototype of a fully functional download manager which accepts an URL from the user and starts downloading content from that URL. The major features of this application are-

1. The GUI maintains a list of downloads that are currently being managed.
2. GUI contains Progress Bar to show the percentage of downloading completed.
3. GUI has buttons allowing user to add, pause, resume, cancel and to clear a download.

The GUI has been made in this application is by using Java Swing.

The Application is divided into a few classes for natural separation of functional components. These are Download, DownloadsTableModel, ProgressRenderer and DownloadManager Classes.

The DownloadManager class is responsible for GUI interface and makes use of DownloadsTableModel and ProgressRenderer for displaying current list of downloads.

The Download class is responsible for actual downloading of a file and represents “managed” download.

The Download Class

The Download class is the workhorse of the Download Manager. Its primary purpose is to download a file and save the file’s contents to disk. Each time a new download is added to Download Manager, a new Download object is instantiated to handle the download. The Download Manager has the ability to download multiple files at once. To achieve this, it’s necessary for each of the simultaneous downloads to run independently. It’s also necessary for each individual download to manage its own state so that it can be reflected in GUI(Graphical User Interface). This is accomplished with the download class. The entire code for Download is shown here. Notice that is extends Observeble and implements Runnable.

The ProgressRenderer Class

The ProgressRenderer class is a small utility class that is used to render the current progress of a download listed in the GUI’s “Downloads” JTable instance. Normally, a JTable instance renders each cell’s data as text. However often it’s particularly useful to render a cell’s data as something other than text. In the Download Manager’s case, I want to render each of the table’s Progress column cells as progress bars. The ProgressRenderer class shown here makes that possible. Notice that it extends JProgressbar and implements TableCellRenderer.

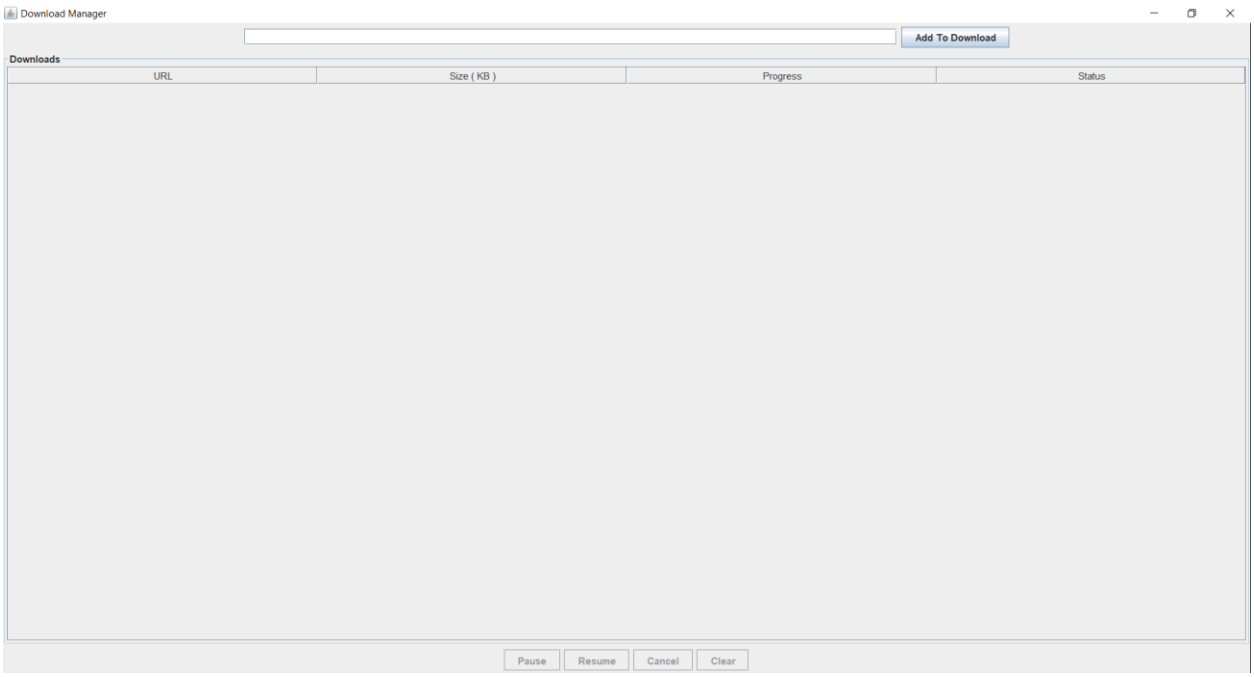
The DownloadsTableModel Class

The DownloadsTableModel class houses the Download Manager’s list of downloads and is the backing data source for the GUI’s “Downloads” JTable instance. It extends AbstractTableModel and implements the observer interface.

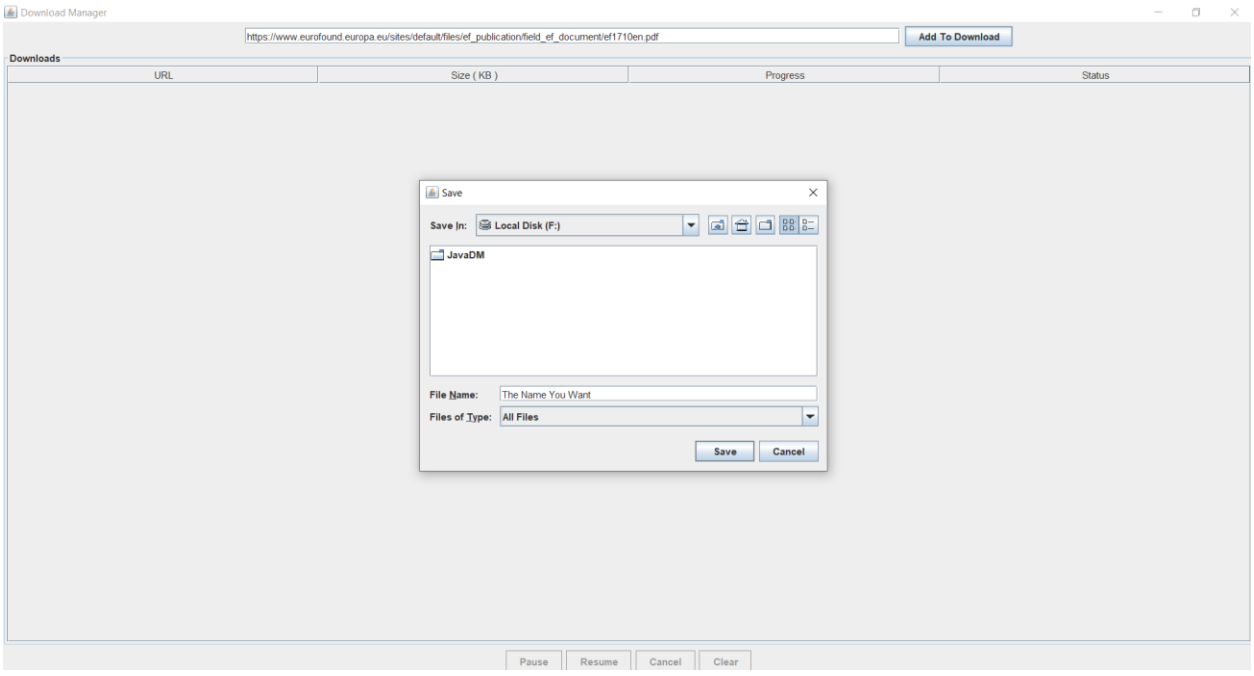
The DownloadManager Class

The DownloadManager class is responsible for creating and running the Download Manager’s GUI. This class has a main() method declared. So on execution it will be invoked first. The main() method instantiates a new DownloadManager class instance and then calls its show() method, which causes it to be displayed. It extends JFrame and implements Observer.

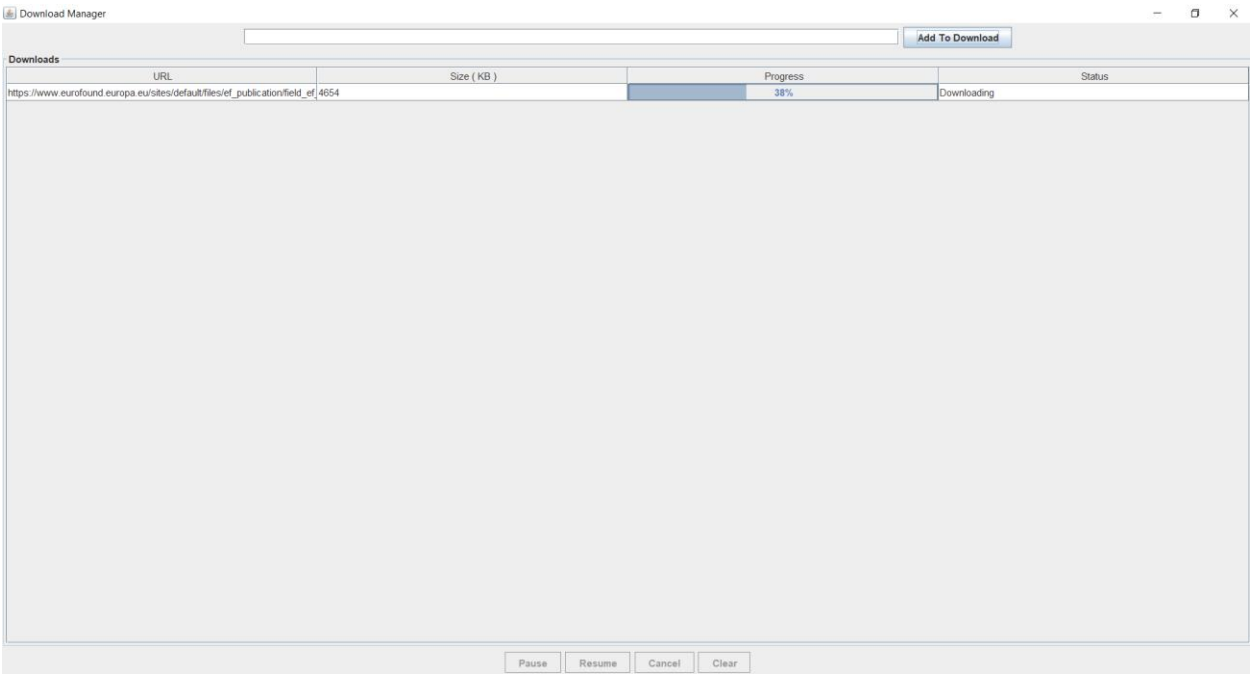
Now, when a user runs the application, he will be able to see the following figure



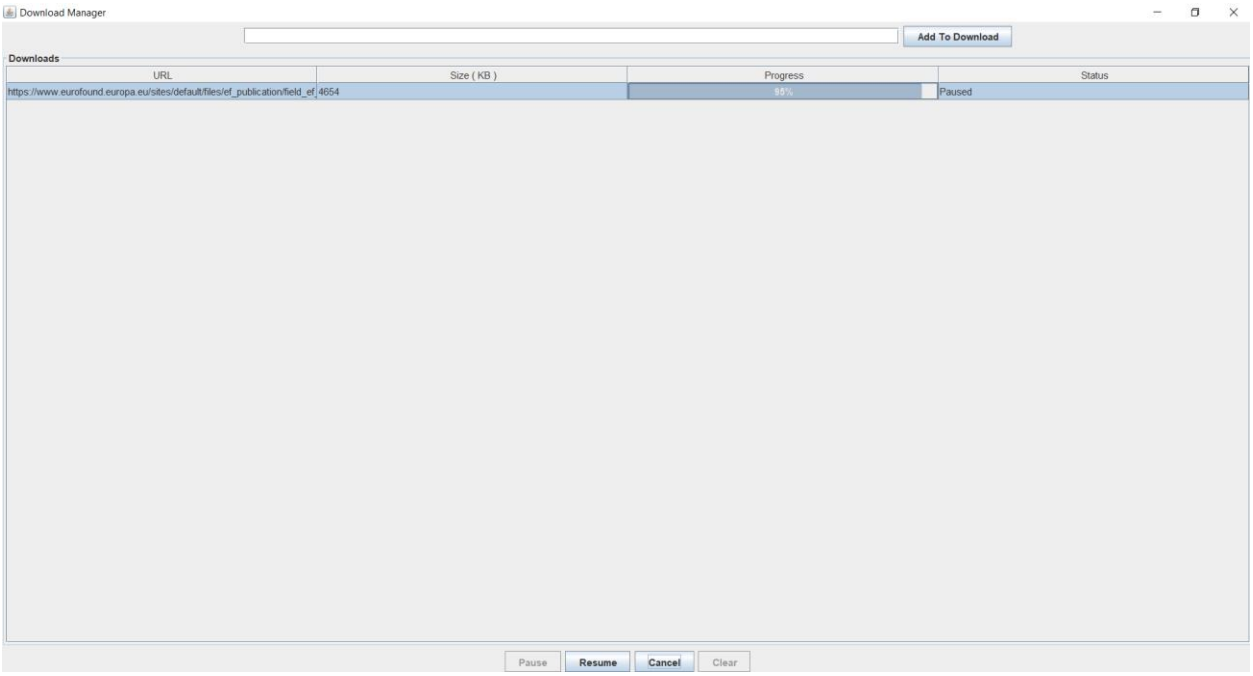
Now, after this table arises, he has to put the URL of the Download File to the textField which is on the upwards. After putting the URL into the textField, he has to press the “Add To Download” Button. After pressing it the following figure will arise



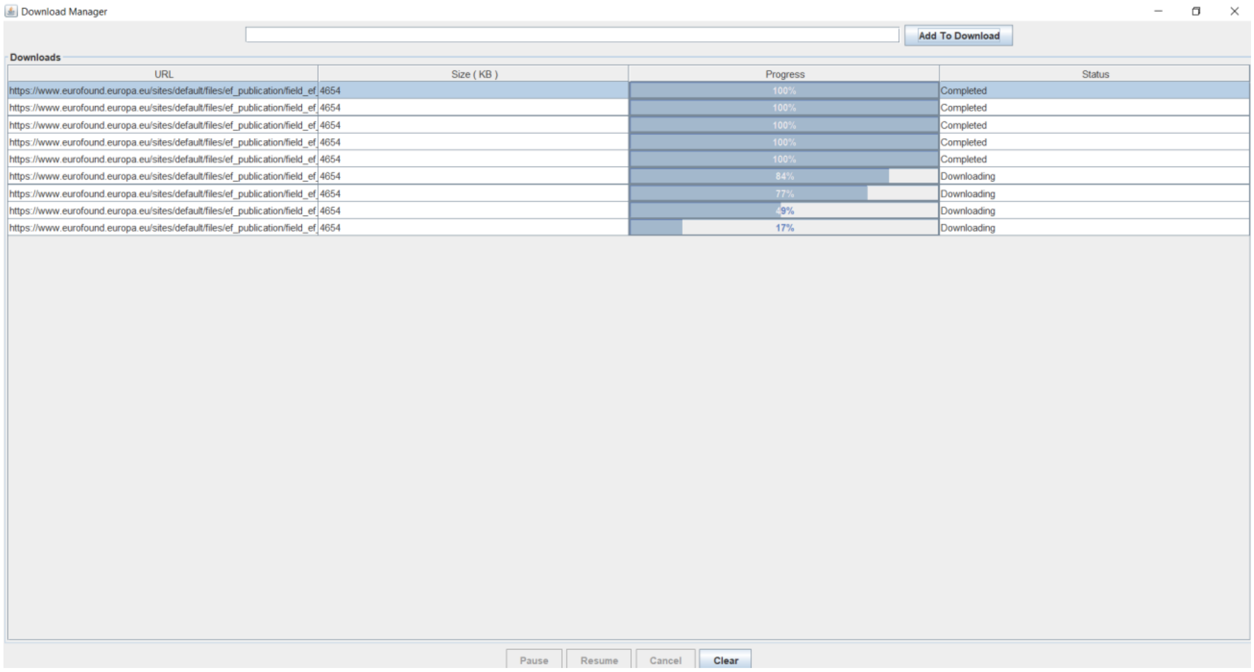
This tells to select the location of your hard disk where he wants to keep the file to be downloaded and also write down the file name in the “file name” textbar the name he likes to keep. The file type will be generated automatically. After doing these two things, he has to push the button “Save” to start downloading. After pushing the “Save” button, the following figure will arise



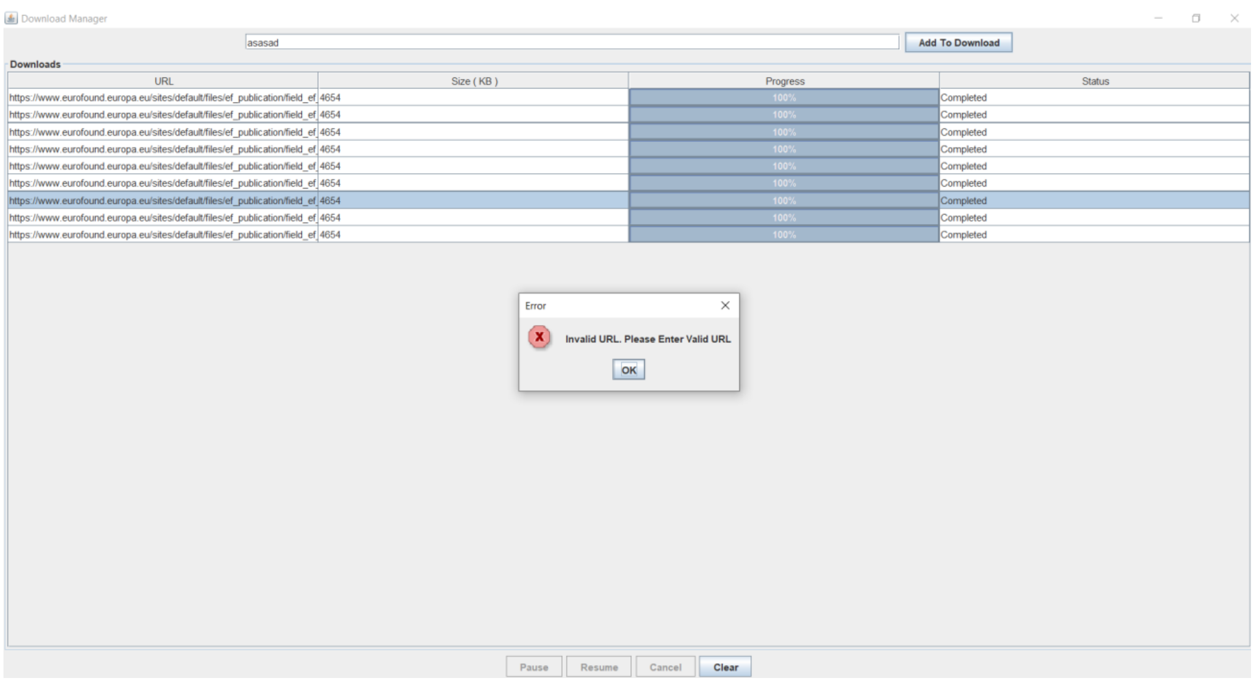
Now, the file is downloading and showing the percentage of size downloaded till now in the progress bar. If he wants to pause this, just select the row of the downloading file and press the pause button. The figure will look like the following



Now he can also resume it or cancel it if he wants. He can do multiples files download at a time just doing the same process again and again just like the following



He can clear a file from the row when the download is completed. He can also see the size of the downloaded file from the size column. If the URL is invalid, then a messege will show like the following that the URL is invalid.



Like this way , a user can download any files.

So, If the steps is shown in a flowchart, it will be seen like this

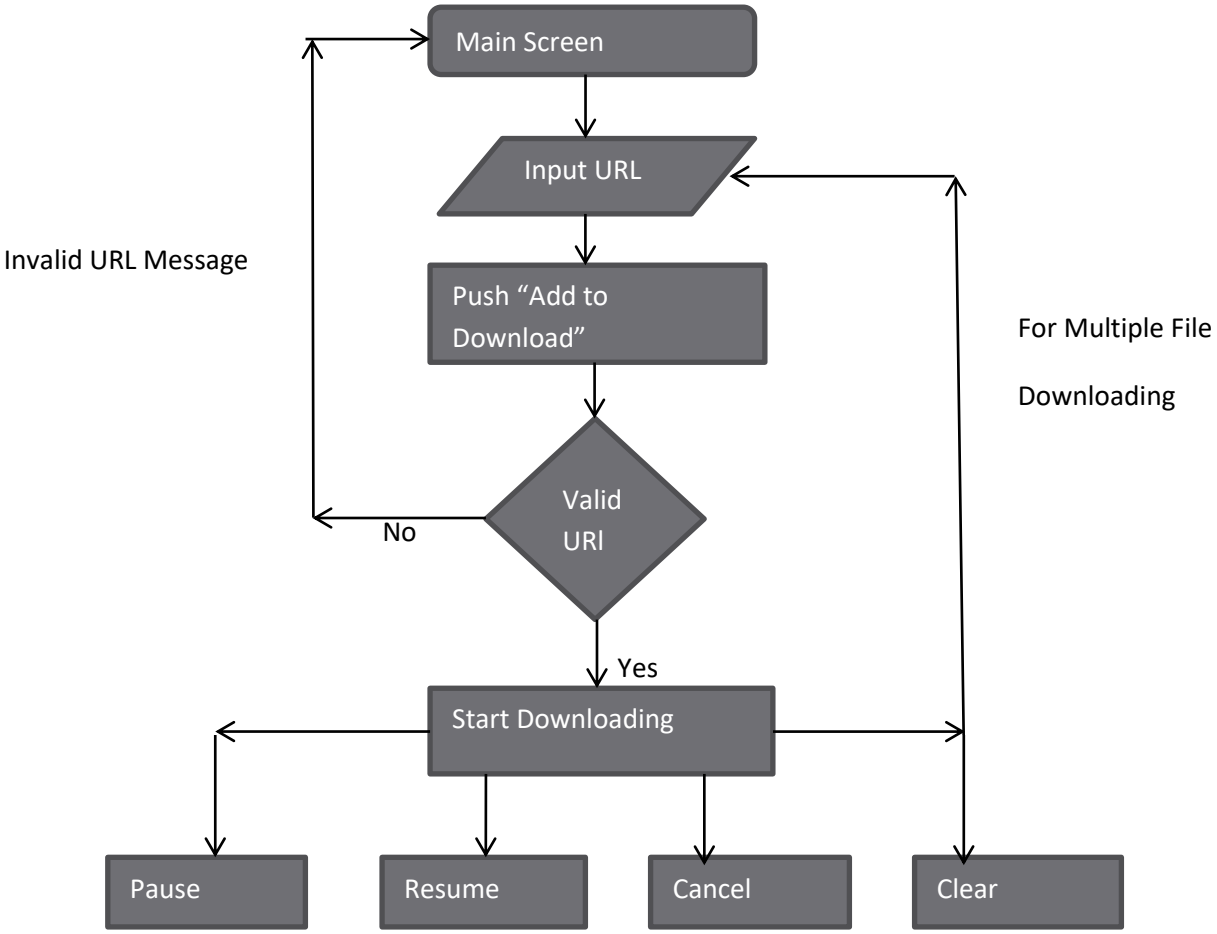


Figure: Flowchart of the process of downloading

Target vs. Actual Accomplishments

My target was to download multiple files at a time, add pause, resume, cancel, clear button, user location browsing system and connect with the web browsers to download the file from the browsers directly. My goal is fully achieved except the last one. That means I couldn't connect the URL with my application from the browser directly. The other targets I have gained successfully.

Risks and Issues

When a file downloading, it must have to have a internet connection in the device that is using to run the application. At the time when internet connection is lost, the downloading will stop. So there is a risk user might have to download the file again.

Discussion & Conclusion

The application is running perfectly. Each classes doing their methods without any error. I have used Java Swing for GUI(Graphical User Interface). The Download Manager Table is working fine. The ProgressBar is showing the progress of a downloading file. It is working fine with the help of ProgressRenderer Class. The location browsing is working fine and it also sets the extension of the file automaticly. There are no issue in multiple file downloading at a time. That means It is also working correctly. The URL checker is also working fine. The application runs at any operating system. Finally, I can say that the application is working with no issue. I have learned a lot of things while doing this project.

References

1. https://stackoverflow.com/
2. https://stackoverflow.com/questions/10486931/jprogressbar-in-jtable-not-updating
3. https://www.youtube.com/watch?v=UfzvNKk89wQ