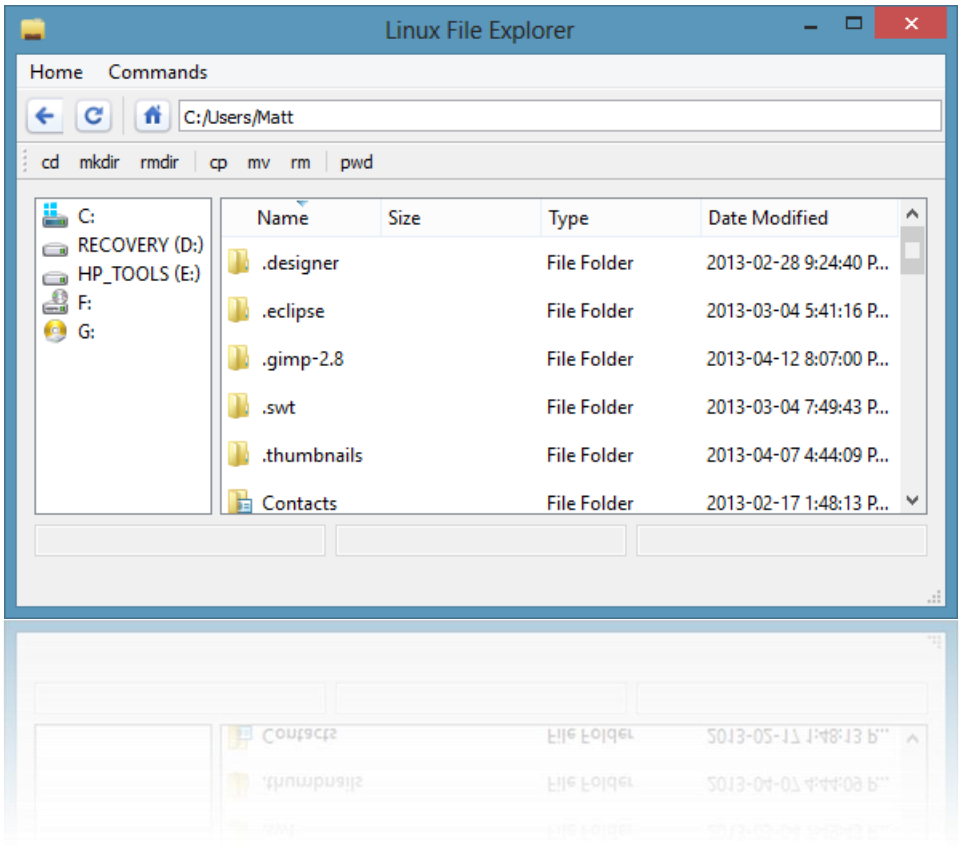
Linux File Explorer for Windows



Team members:

Justin Mancini
Kevin Mason
Matthew Tang

Table of Contents

Title: Linux File Explorer for Windows

Date: April 17, 2013

Authors:

Justin Mancini, 100823164, justinmancini@cmail.carleton.ca
Kevin Mason, 100819904, kevinmason@cmail.carleton.ca
Matthew Tang, 100818849, matttang@cmail.carleton.ca

Note

The work presented in this paper is new and was prepared only for the course COMP 3000 Operating Systems, Winter 2013, School of Computer Science, Carleton University.

1. Introduction

1.1 Context/Background

Linux File Explorer is a file browser for Windows which combines the features of the modern file browser with Linux's shell terminal. It is targeted towards users who are familiar with the basic Linux commands and would like a visual representation of their directories and files. File browsers are similar on most operating systems, so the goal of Linux File Explorer is to provide a comfortable file browser for beginners in Linux.

1.2 Definition of the problem

This project intends to support users who have been using Linux or Windows operating system and would like to switch to a more comfortable interface that support features of both environments. It will help new Linux users learn simple, everyday commands with visualization, whereas the Linux terminal is just text-based.

1.3 Summary of the result

There are numerous shell commands available for manipulating files such as moving to another directory, adding and/or removing files and folders, and more. The Linux File Explorer provides fast and easy use with input devices like a keyboard and mouse. It supports basic commands such as moving, adding, removing directories and files. It also supports changing directory with a mouse click and opening any file on your system (with the appropriate software).

1.4 Outline of the report

Section 2 provides more details on the background of this software product. Section 3 provides information on the description of the result. Section 4 presents the evaluation of the result. Section 5 concludes the documentation. Section 6 provides the contributions of each team member and Section 7 features the assets of the Linux File Explorer application.

2. Background and Previous Work

File browsers are an integral part of any computing experience. Without a browser you cannot view and edit files that you have already created. Although command line file browsing is always an option it is usually neither the fastest nor friendliest way for beginners because most main stream computer users are used to using a mouse for everything. The file browser is a real time system and must be responsive to the user.

Although we have no experience designing and building a file browser or any other important GUI elements we have decided to give the traditional Windows file browser a new feature. We stumbled upon this idea when we were in third year university and we had just completed the COMP 2401 and COMP 2402. Having to use the lambda servers daily for programming in C and C++ we became bedevilled with the Linux commands and command line entries. We never had to use a mouse. Then when we went back to our original Windows consoles the inefficiencies hit us immediately, and so we decided for this group project we would implement Linux commands to a Windows File Explorer.

3. Description of the Result

Our immediate result is an easy to use simple file explorer. Our file browser is intended to be used by newcomers to technology or by users who are just learning how to navigate with Linux commands. The more prominent subjects being the elderly and young children. We also feel that Linux users who would like a visual representation of their directory structure would find this software very useful. This software would be ideal for an elementary school where children are being taught the basics of computing.

Only essential commands are present in our Linux File Explorer, such as viewing a directory, the present working directory, and commands to do simple tasks such as create a directory, copy, move, and change to another existing directory (see **Error! Reference source not found.** We were also able to integrate file meta data into our browser so that a user may have more information available to them. They will be able to associate the box with whatever command they have used, depending on whether the command has one or two parameters. Hotkeys allow users to be more efficient and work quicker while using the Linux File Explorer. We feel that hotkeys are always a great addition to any software that strives for efficiency.

The integration of speed and efficiency is prevalent in the Linux File Explorer. Every command will focus your text cursor to the parameter field to start typing instantly (see Appendix 3). After the parameter fields are filled, the user may press enter to use the command. There will be a notice on the bottom of the application stating whether the action was successful or not (see Appendix 5 and 6). This allows the user to know whether or not their command worked.

The file system view allows the users to open any file or folder with a double click for users who prefer a mouse. Another resemblance to Windows is that it is able to open a file such as an

image or an executable and more through their system defaults. The commands can either be activated from clicking on corresponding command on the command bar or by using their dedicated hotkeys (see Appendix 2). The address bar resembles Windows such that there is a field that shows up while typing which displays the existing folders with its corresponding path (see Appendix 7). The home button allows the user to quickly return to the Home (default) path which is selected by the user. The back button allows the user to go one directory up rather than the generic back. This implementation was decided for the sole purpose that it would be fairly similar to the File Browser that comes with the Windows OS but it would also add a little touch of Linux.

With the addition of all these features and functionality to the standard application, the Linux File Explorer's interface is fully customizable to provide users with the freedom of using their own preference. A good example of such is the command bar, which is able to move from the top, to the left or right of the window simply by being dragged from the left side of the bar (see Appendix 9). The Linux File Explorer also allows the user to hide the command bar and address bar by right clicking and unchecking the bar's name (see Appendix 10).

Using Qt as the foundation allowed us to focus on the aesthetic appeal of the software and to establish an environment where it was not too difficult to complete all the goals we set as a team in the beginning. Although the Linux File Explorer runs as a standalone executable, integrating it into any GUI as the default file browser would be very possible.

4. Evaluation of the Result

In order to implement Linux commands in a Windows environment file browser, Qt was used as the application framework. The features added to the original file browser include: the addition of *cd, mkdir, rmdir, cp, mv, rm, pwd* command buttons, the possibility to open a file by double clicking a file and having it open with the correct program, and the option of using hotkeys for example *Alt+1*, to access commands.

4.1 Highlights

The Linux File Explorer consists of all the basic features needed in a Linux File Explorer with the addition of clickable Linux command buttons and simple file browsing. It was specifically made for Windows Operating Systems, because the implementation of these foreign commands will help Linux users to easily alternate from Linux to Windows. The ability to use hotkeys to access commands adds a quick simplistic touch the Linux File Explorer because it allows for faster access of commands instead of always having to use the mouse. These command hotkeys are made visible beside their command under the commands tab at the top which is extremely important and different from most existing file browsers. Most file browsers today most likely have the possibility to use hotkeys however many users are not aware they exist and therefore have never used them. For those not familiar with the parameters of each Linux command and their capabilities, Linux File Explorer allows users to hover over the

commands with their mouse and then provide them with a demo calling of the command (see Appendix 8)**.**

4.2 Methodologies

The Linux File Explorer was developed with Qt, following the model-view-controller architecture. Although we had no prior experience with Qt, we decided it was our best choice for programming the file browser because it had built-in UI (user interface) components. Everything pertaining to the GUI was coded in a separate file which was in charge of initializing the file system and the Linux commands. The following templates were used to enhance our application: QFileSystemModel, QLineEdit, QStyledItemDelegate, QDir, QDesktopServices, QUrl, and QCompleter.

The main template used to create our application was the QFileSystemModel. This template allowed us to do many things such as get the current root path and return all the items in that path. We were able to manipulate its functions such as *rootDirectory()* to return a QDir which holds the current directory. For example, to change directory, we manipulated the resulting QDir object to call a function *cd(String)* which changes the directory to the passed in parameter. Every change we made to the file system, we would have to update by setting the root path to the latest directory and updating the file system view. The file system model also updates the file browser after any change to the system from outside the file browser. This is an essential part of any file browser.

The default directory on the file system view is a directory path which changes on different systems. With the QDir template provided by Qt, we were able to call the function *homePath()* to successfully set the default path to the user's home directory. The QDesktopServices and QUrl templates allowed the application to open any file from our Linux File Explorer. On the interface side, there were built-in components we used which were not needed to be declared in the main header file. We used components like QSplitter, QLineEdit, QListView, QTableView, and QStatusBar. These components helped define our interface design and simplicity to our application.

The QCompleter template helped our application resemble Windows OS (operating system) due to its auto-completion feature. It is placed in the address bar but not in the command line because we wanted to separate the Windows and Linux features to itself by having an alternative.

We focused our easy-to-use and simplicity design on our parameter fields where the command line is held. The command field is always unchangeable but the parameter fields are only changeable depending on the command used, whether it uses one parameter or two. Once a command has been selected, the text cursor is instantaneously focused on the first parameter to allow typing and the enter key to submit the command.

Linux File Explorer is made to be clean, easy to use and very efficient. Its purpose is to help bring together Linux and Windows users in the same environment and to leave users with an effortless experience.

4.3 Testing

With that being said there are many strengths attached to the Linux File Browser and like all software some weaknesses. The table below displays three different scores (from testers) each of which come from different OS backgrounds. The rating scale is from 1 to 5, 1 being poor and 5 being excellent.

The first test case was done by a user who came from a Mac OS. She thought the user interface was brutal. It was nothing like a Mac and like all Mac users she was very narrow minded toward anything else. She gave it a 5 for efficiency, ideology and performance because she deemed it well made and she was able to browse through folders on the test case computer faster than she was on her own. Since it was so different from a Mac she had some questions and problems managing the Linux commands and their purposes. Overall a score of 19 out of 25 was satisfactory for her and if she had to make any improvements they would be in the user interface.

The second case was done by a user who came from Linux, someone who was entirely bathed in the text-based terminal to run commands. He thought that the application would most definitely appeal to those learning Linux and would be a great visual representation. He gave the ideology, simplicity, and efficiency a 5 because of the educational purpose and easy-to-use interface. The idea was there, everything was working, and it was a seemingly perfect mix of features from Linux and Windows. The user interface was attractive but given a score of 4 because the user was not very familiar with the Windows UI. Performance was given a score of 4 because he found that sometimes the loading of the file view would be quite slow. The next traverse would be instantaneous, probably because of the caching of the file system. The overall score given was 23, he said it was a job well done.

The third test case was done by a Windows user and they absolutely loved the idea. They thought the hotkeys were well labeled and this new interface allowed navigation and customizations to be done a lot easier than the previous versions where they always had to right click, make new folder, delete new folder and then rename it to the specified name. This user happened to be familiar with Linux commands and command line entries so they were able to truly value the idea. They gave it 25 out of 25, a perfect score.

Tester	User Interface	Efficiency	Simplicity	Ideology	Performance	Total
Mac OS	1	5	3	5	5	19
Linux OS	4	5	5	5	4	23
Windows OS	5	5	5	5	5	25

5. Conclusion

In conclusion the Linux File Explorer was created to fix the problem most Linux users faced when they made the transition from Linux to Windows, which is compatibility. The two operating systems are very different and Linux users view some of the ways Windows users do things very inefficient, in particular the typical Windows File Explorer. Users coming from the command line background don't like to take their fingers off the keyboard to reach for the mouse to browse, make saves or changes to files because it's just not practical. On the other hand Windows users believe in the mouse for everything except typing. The Linux File Explorer brings the best of both worlds together in one easy to operate and convenient File Explorer. For users who don't like to take their hands off the keyboard there are hotkeys and for users who like to use a mouse there are clickable command buttons. All in all Linux File Explorer was created to bring together users from two of the main OS platforms in the market today and in the opinion of the software's testers, it has succeeded in doing just that.

6. Contributions of Team Members

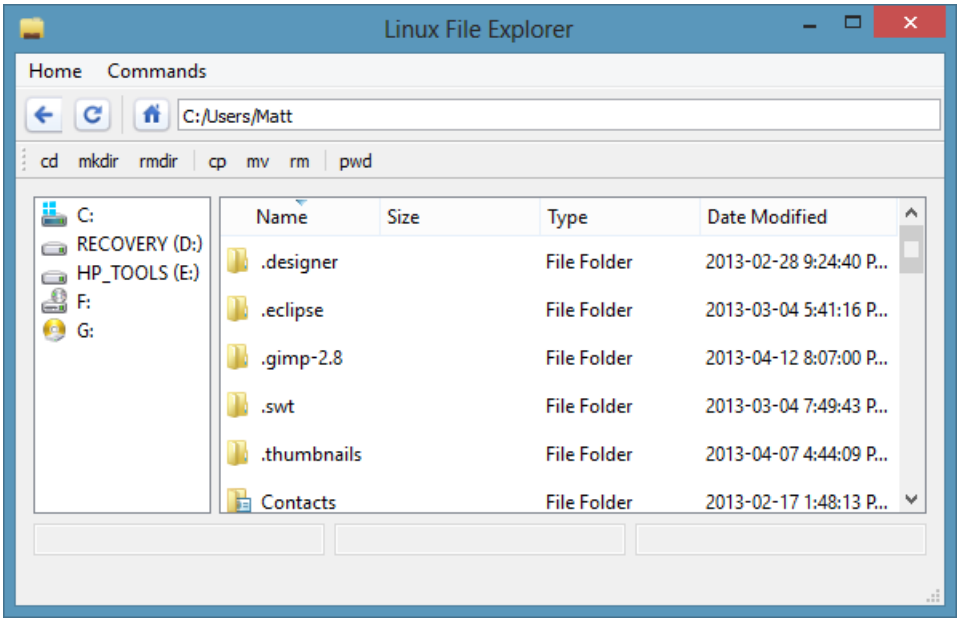
Justin Mancini has developed the command bar and its functionality. He was also responsible for testing the commands on his system to ensure that each command did their job successfully. He also developed the idea of disabling the parameter fields and enabling them when needed.

Kevin Mason has developed the file system view and the functionality to open any file or folder with the click of a button (an alternative to the Linux command: cd). He has allowed the table and list view to update itself according to the user's actions.

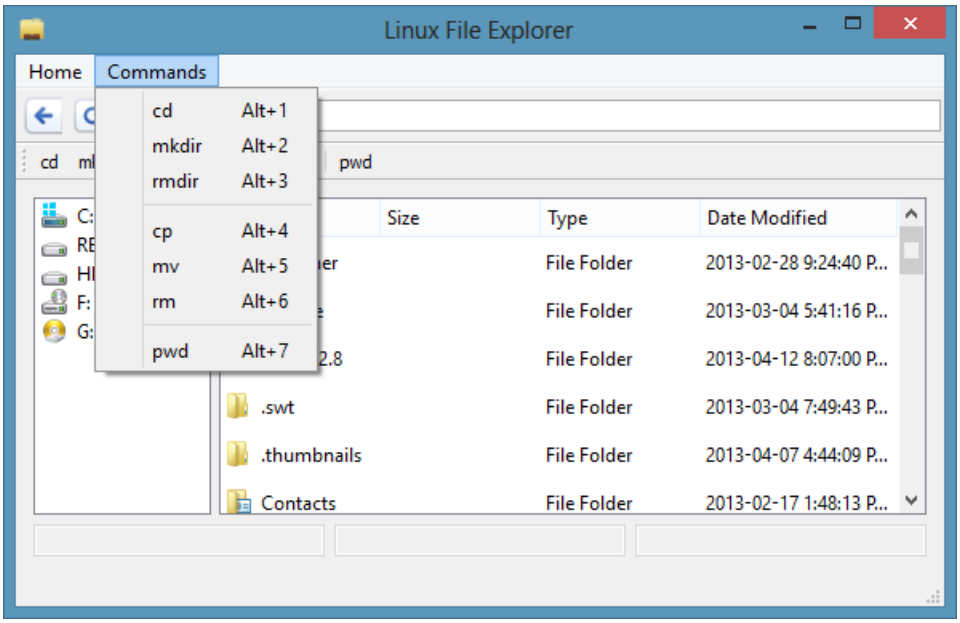
Matthew Tang has developed the user interface including the back button, home button, address bar and parameter fields. He has also added the QCompleter class to the address bar to allow similar features to Windows. QCompleter class pops up a field of valid folders and files according to what the user typed in.

7. Appendices

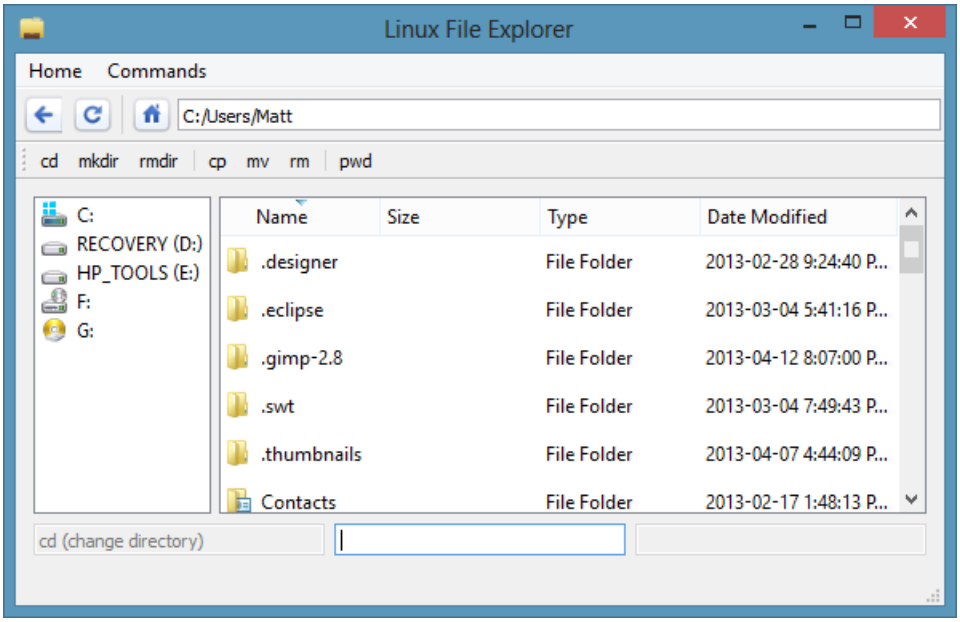
Appendix 1 – Linux File Explorer's graphical user interface.



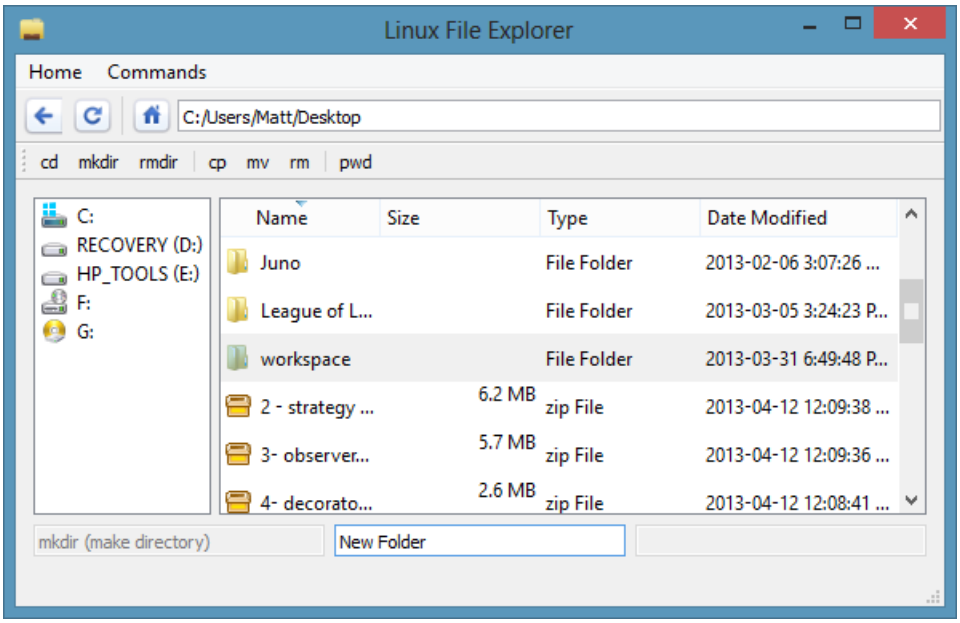
Appendix 2 – Commands menu with shortcut keys shown.



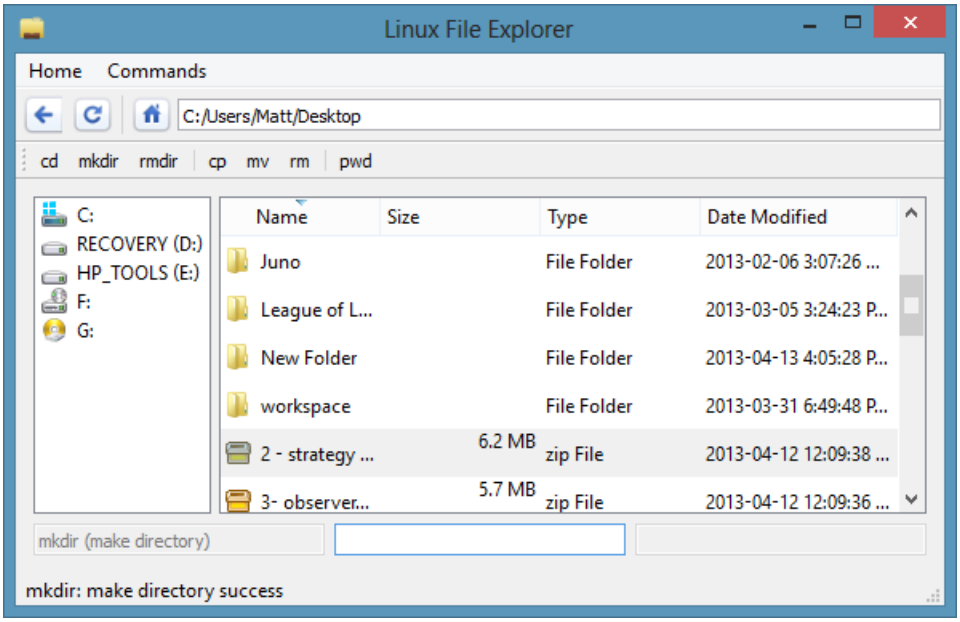
Appendix 3 – An example with *cd* command selected.



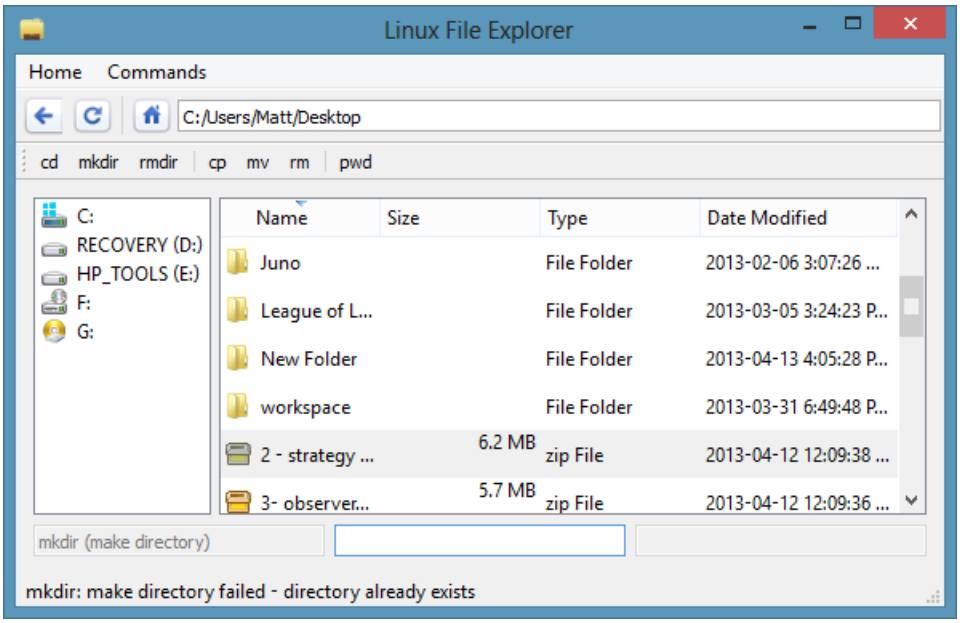
Appendix 4 – An example of making a directory called *New Folder.*



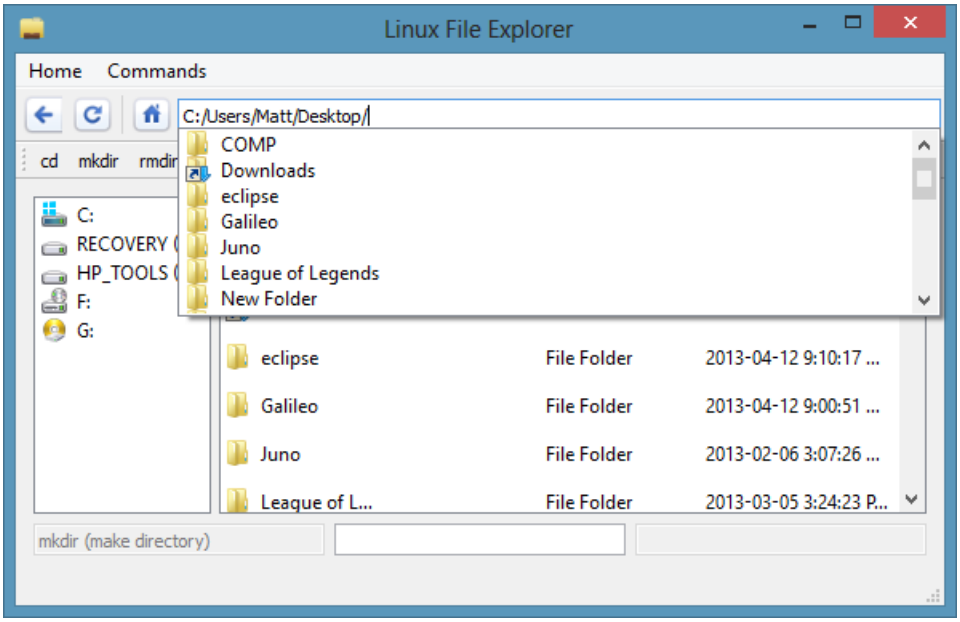
Appendix 5 – Result of *Appendix 4* (making a new directory): successful status.



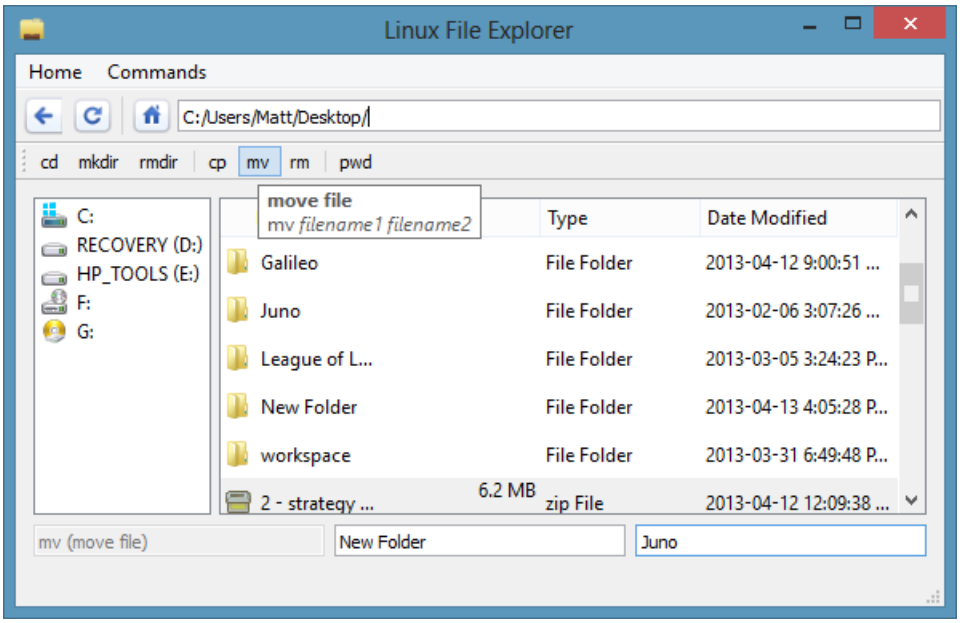
Appendix 6 – Result of creating an already existing directory.



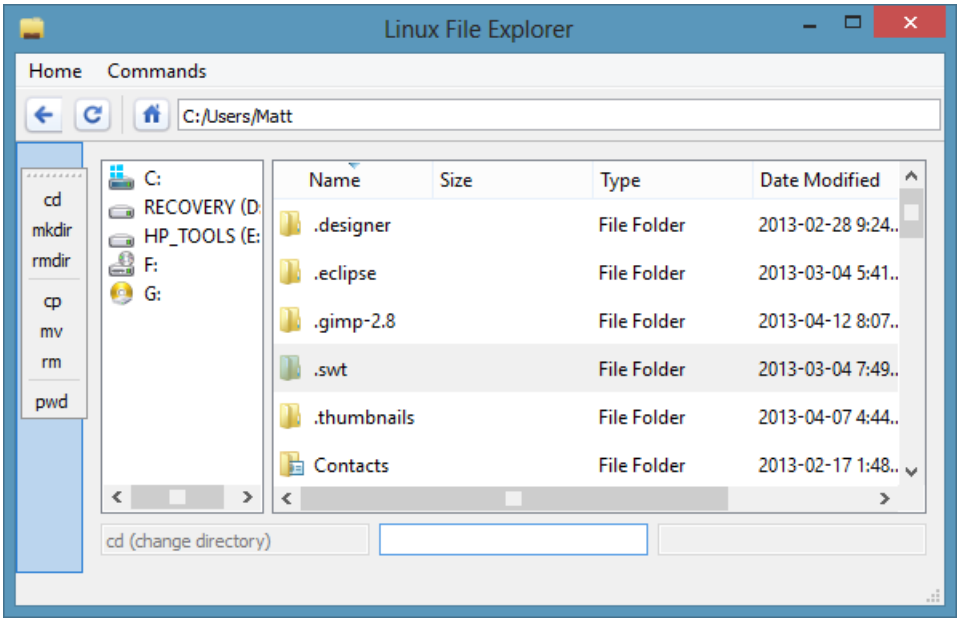
Appendix 7 – Changing directory using the file path completer.



Appendix 8 – Description of a command by mouse hover.



Appendix 9 – An example of moving the commands bar to the left side.



Appendix 10 – Right-clicking will display pop-up menu to toggle visibility of bars.