Caio Novelini #100985497

Mir Abbas #100956642

Team number: 52

PROJECT REPORT

Cron job scheduler

Submitted to
Professor Michel Barbeau
for the course
Operating Systems (COMP3000)

Carleton University
9 December 2016

1. Introduction

1.1 Context

Cron is a tool that schedules jobs in a Unix/Linux and mac operating system. Such jobs can be a commands or a script that will run periodically at specified time and date or interval. Those are very useful to automate tasks that needs to be repeated periodically.

Qt creator is a cross-platform (C/C++, JavaScript, QML) that allows developers to create and integrate applications with tools for designing and building graphical user interfaces.

1.2 Problem Statement

The Linux OS is used as web hosting for many websites. Cron jobs are an essential tool for servers that schedule many type of jobs; such as maintenance tasks, database backups and overall backups.

The way that Linux OS offers the user to manage the crontabs is technical and demands a certain experience to be correctly done. There are no filters in the user's input and it shows no errors when the crontab is not properly set. The goal with this software is to create an easy way to schedule cron jobs and at the same time, prevent the most common user input errors by presenting only the valid possible options of periodicity entries. Together with a friendly interface, it will provide a better user experience.

These are few of the problems that could occur while scheduling a cron job by using the crontab -e command:

1. Not easy to use for different tasks.
2. It's not a user friendly way to do job scheduling.
3. Days of the week and months are represented in numbers that can concatenate together to generate periods. (e.g. Saturday is represented as 5).
4. Do not provide any help to find the right frequency.

1.3 Result

The project resulted in a software that is able to manage cron jobs with a much better user experience. The user is then presented with all the jobs already in the system and with a few clicks he is able to add or delete any job in a much simpler way compared to what the Linux OS offers. By creating a user interface using QT, we were able to instruct the user in a very simple way and by limiting the user's input using list views instead of a text field, we managed to offer only the valid inputs. That way, we are helping to prevent possible errors.

1.4 Outline

The rest of this report is structured as follows:

Section 2 - Present the background information needed for the project.

Section 3 - Describes in detail the result obtained.

Section 4 - Evaluation of the result.

Section 5 - Conclusion.

2. Background Information

The tool Cron is a time-based job scheduler in machines that operate a Linux/Unix OS. It is used to setup and maintain networking and software environments, to schedule jobs using commands or shell scripts which can be customized to run periodically at fixed times, dates, or intervals. It typically automates system maintenance or administration, make backups, download files, among other operations.

The way Linux OS present to manage the jobs is by accessing the terminal and typing the command "crontab -e". After selecting the text editor, you will be presented with crontab file.

The crontab are files where the lists of jobs and other instructions to the cron are kept. All the commands in the crontab file are periodically checked by the cron daemon and executed in the system background. Users can have individual crontab files and there is also a system-wide crontab file that only administrators can edit. The structure of a crontab file is that every line represents a job and contains the cron expression and the command to be executed. For example: "** * * * 6 sh example.sh"*. The *"* * * * 6"* represents the cron expression that dictates when this command will be executed and *"sh example.sh"* represents the command to be executed. [1]

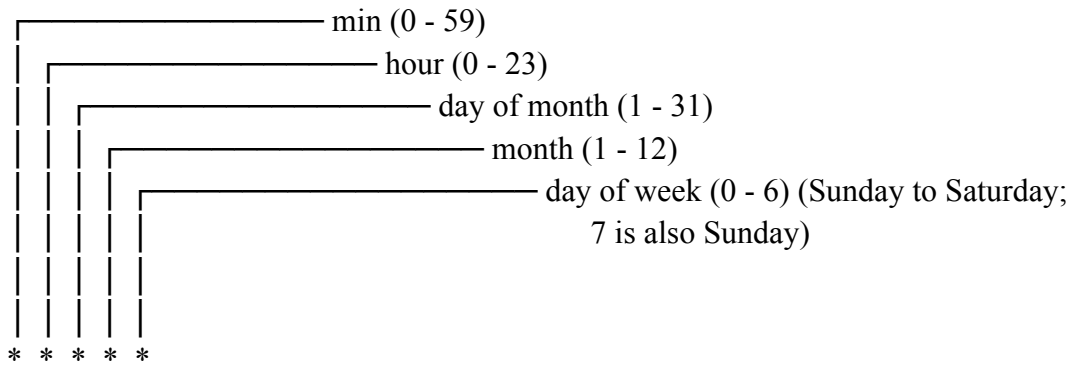
Cron permissions

Two files are used to manage cron permissions:

- **/etc/cron.allow** - If it exists, only users listed are allowed to use the cron and the cron.deny is ignored.
- **/etc/cron.deny** - If the cron.allow file does not exist, users listed in cron.deny are allowed to use the cron. [2]

Cron expressions

Cron expressions are represented according to the following scheme:



Concatenated in a string and separated by a white space, it determines when the command will be executed. “ * ” represent all. [3]

Special string may appear in the cron expressions [3]:

@yearly	Execute once a year (midnight of 1 January)
@monthly	Execute once every month
@weakly	Execute once every week
@daily	Execute once every day
@hourly	Execute once every hour
@reboot	Execute at system startup

Some combinations

0 0 1 1 * [command]	Execute once a year (midnight of 1 January)
0 0 1 * * [command]	Execute once every month
0 0 * * 0 [command]	Execute once every week
0 0 * * * [command]	Execute once every day
0 * * * * [command]	Execute once every hour

QT Creator

QT Creator is an integrated development environment cross-platform C++, Javascript, QML that allow users to integrate code with GUI in an efficient way. It uses the C++ compiler from the GNU compiler collection. Many plugins are available to extend the use of the software although none was used during this project.

The QT Design, which is part of the QT Creator package, make easier the creation of the GUI by providing QT Widgets that can be used. It also allows the creation of animations by using the programming language QML [4].

Algorithm used in our software to manage the cron jobs:

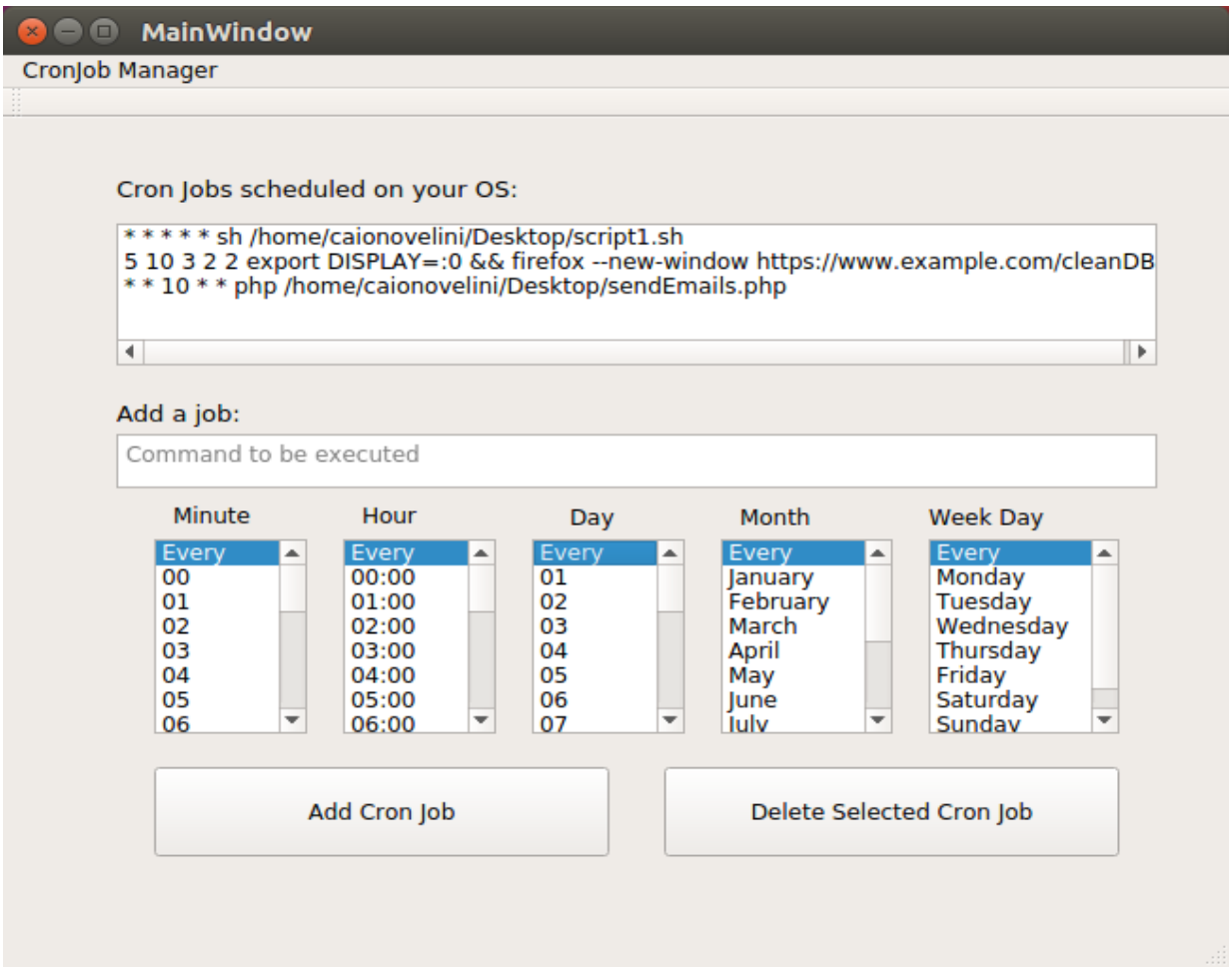
The algorithm used by the application to manage the cron jobs is as follows:

1. Create and execute a QTprocess (system call) that access the crontab file from the system.
2. Create a temporary cron.txt with all the crontabs retrieved from the system on step 1.
3. Once the process gets previous information from the crontab file, the UI is updated.
4. Now the application is waiting for user to schedule any task by specifying year, month, week, day, hour and a custom command to execute any program as needed.
5. Once the user has scheduled the cron job, the program writes the scheduled jobs on a temporary buffer (cron.txt).
6. The program creates a new process, access the crontab file from the system and writes the cron.txt to be the new crontab.
7. Whenever user adds or delete any cron job repeat step 1 to 6

3. Result

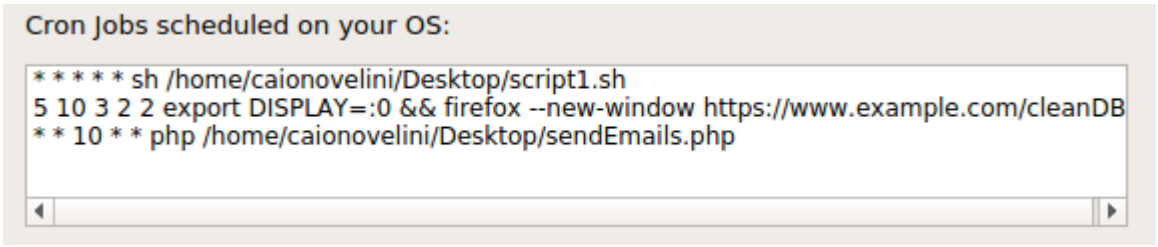
The application easily allows the user to schedule a cron job by having separate fields for command to be executed and for every sub-item of the frequency. The frequency inputs (Minute, Hour, Day, Month and Weekday) are represented as it is and are internally converted to the numbers that represent this frequency in the crontab file. A separate input is designated to hold the command to be executed in the particular job as shown in the image below (Image 1).

Image 1: Screenshot of the main screen.



Once the application is open, the user will immediately see all the cron jobs scheduled on his OS as shown in (Image 2). From the main screen the user can manage all the jobs in the system.

Image 2: Application showing the current jobs scheduled in the OS



If the user wants to delete a job he only need to click in the crontabs list and select the desired job. After the selection by clicking in the button “Delete Selected Cron Job” the job is

deleted (Image 3). If desired, the deletion can be confirmed by accessing the terminal application and typing the command “crontab -l” that lists all the current jobs in the OS (Image 4).

Image 3: User deleting a job

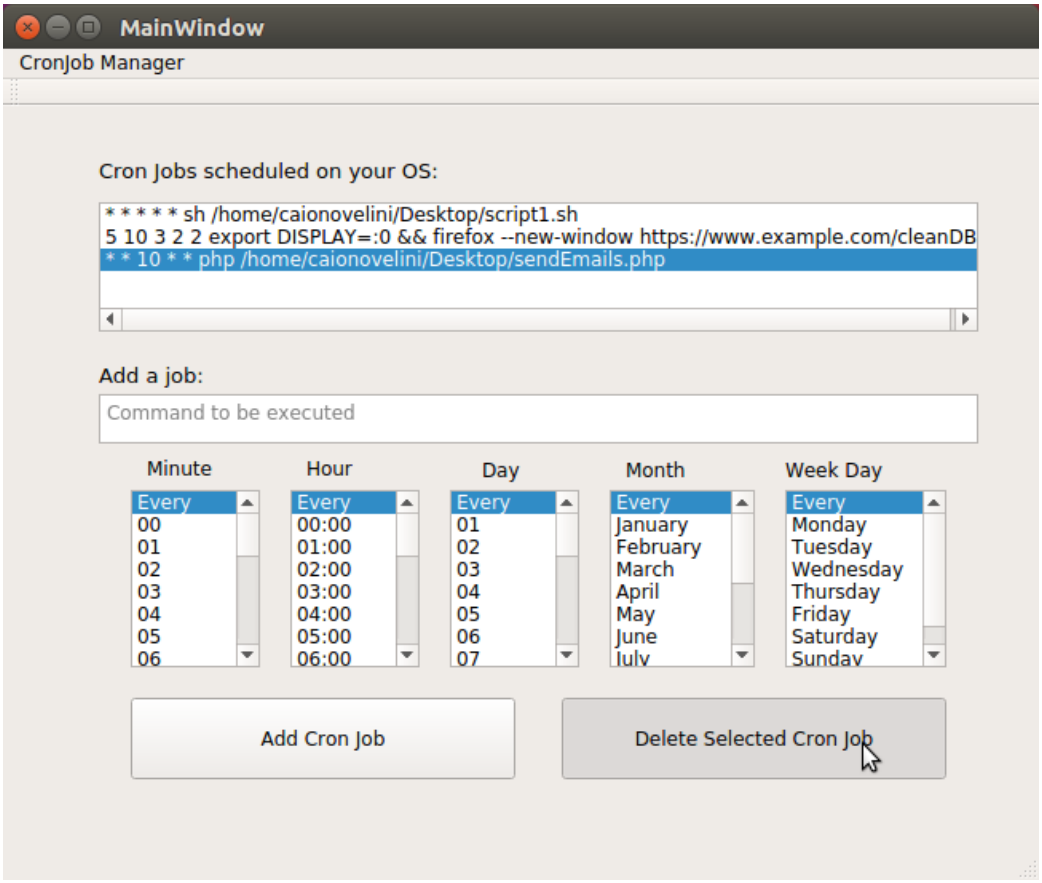
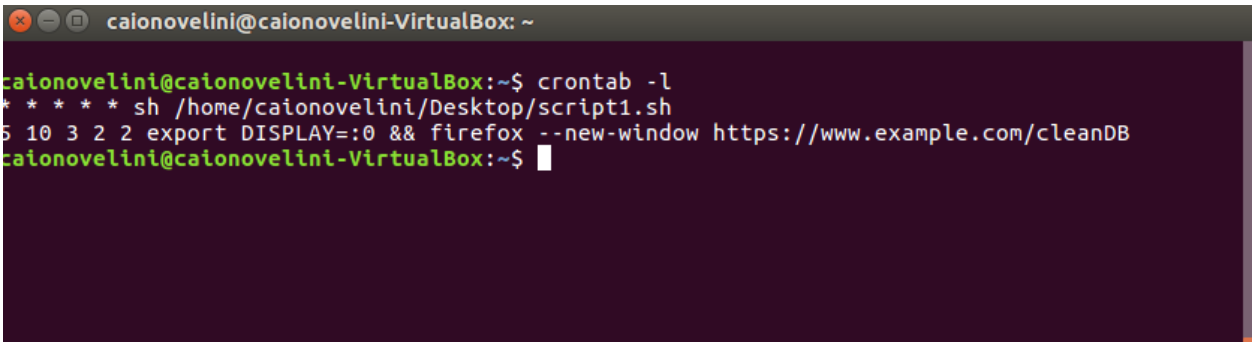


Image 4: Terminal listing the current jobs in the OS



As shown in the image above the cron job was successfully removed from the jobs list.

In the other hand, if the user wants to create a new job, it can easily do it by following the steps:

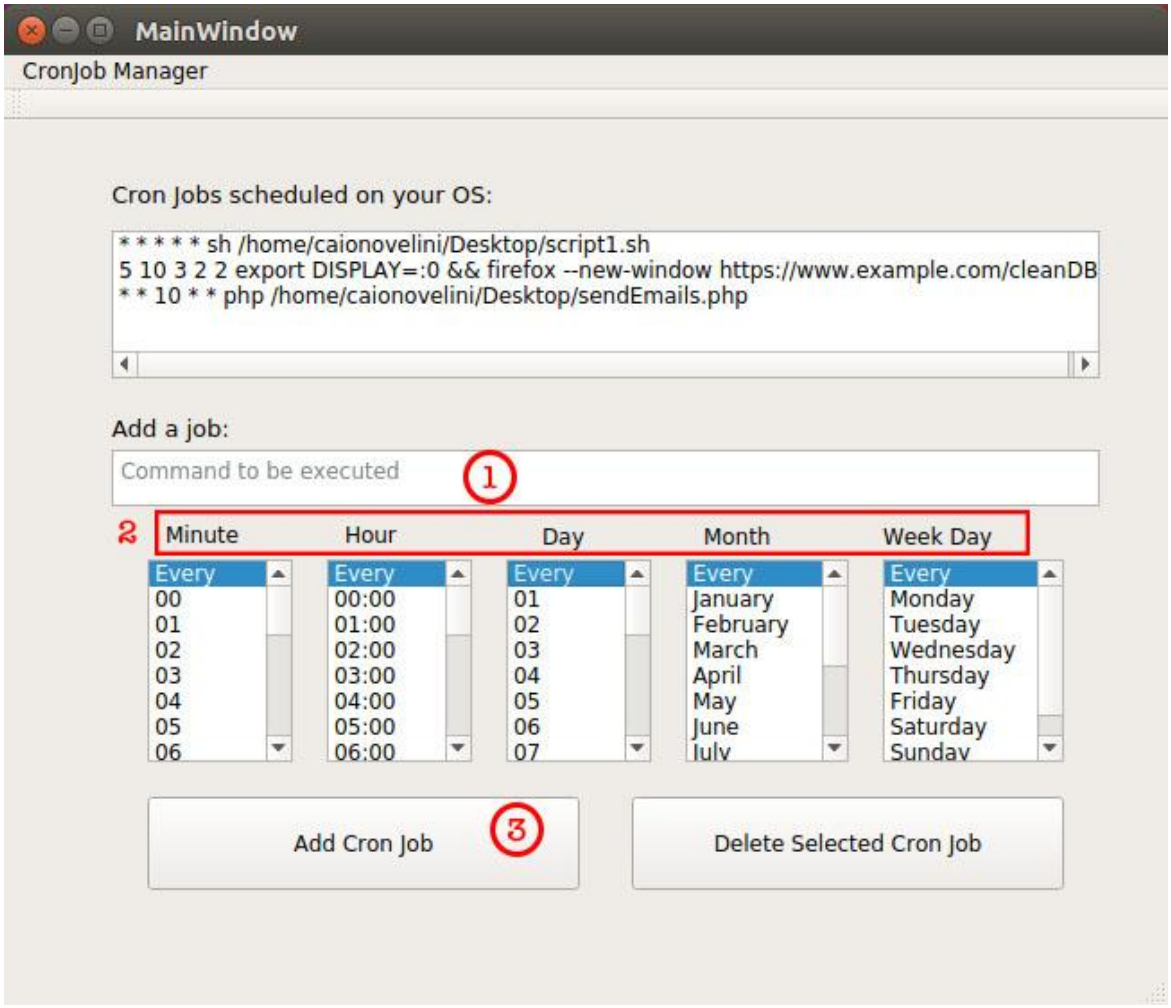
1 - Insert the command to be executed in the area (1).

2 - Select the frequency in (Minutes, Hours, Days, Month and Weekdays) in the area (2) that this job will be execute.

3 - Press the button (3) “Add Cron Job”.

As represented in the image below.

Image 4: Print screen with steps to add a job

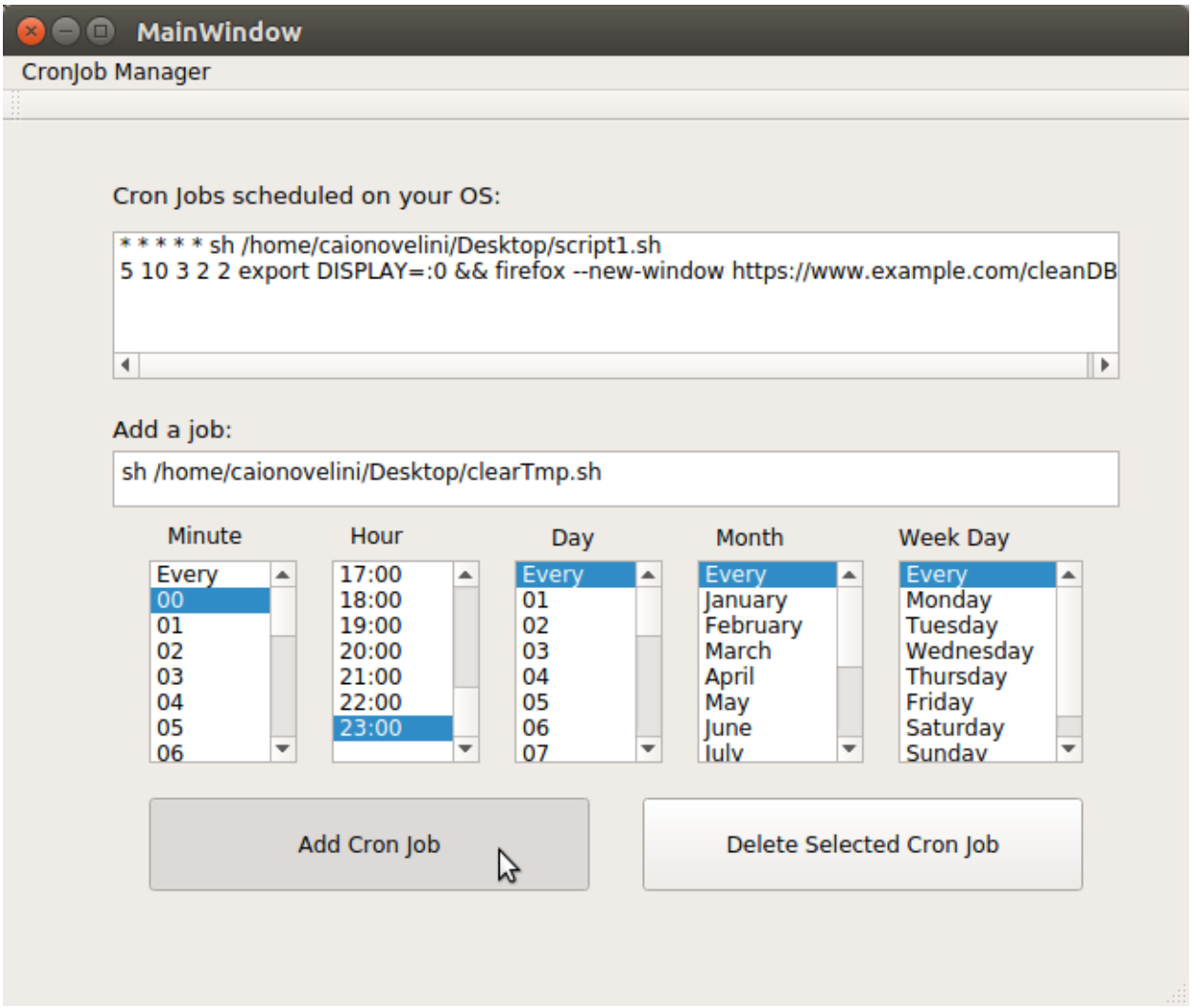


By providing these tools jobs can be added and removed in a very efficient way as to be shown in the usability study in section 4. In the next paragraphs we will go through some real life applications of the software.

Example 1: Run every day at 11:00 p.m. a shell script that removes all files from the tmp folder.

First step is to create a shell script that clears the tmp folder and saves it as a clearTmp.sh file in the desired directory, in this case /home/caionovelini/Desktop/. With the shell script file created, we can now launch the application and fill the entry fields as shown in the image 5.

Image 5: Creating the job that executes clearTmp.sh file every day at 11:00 pm

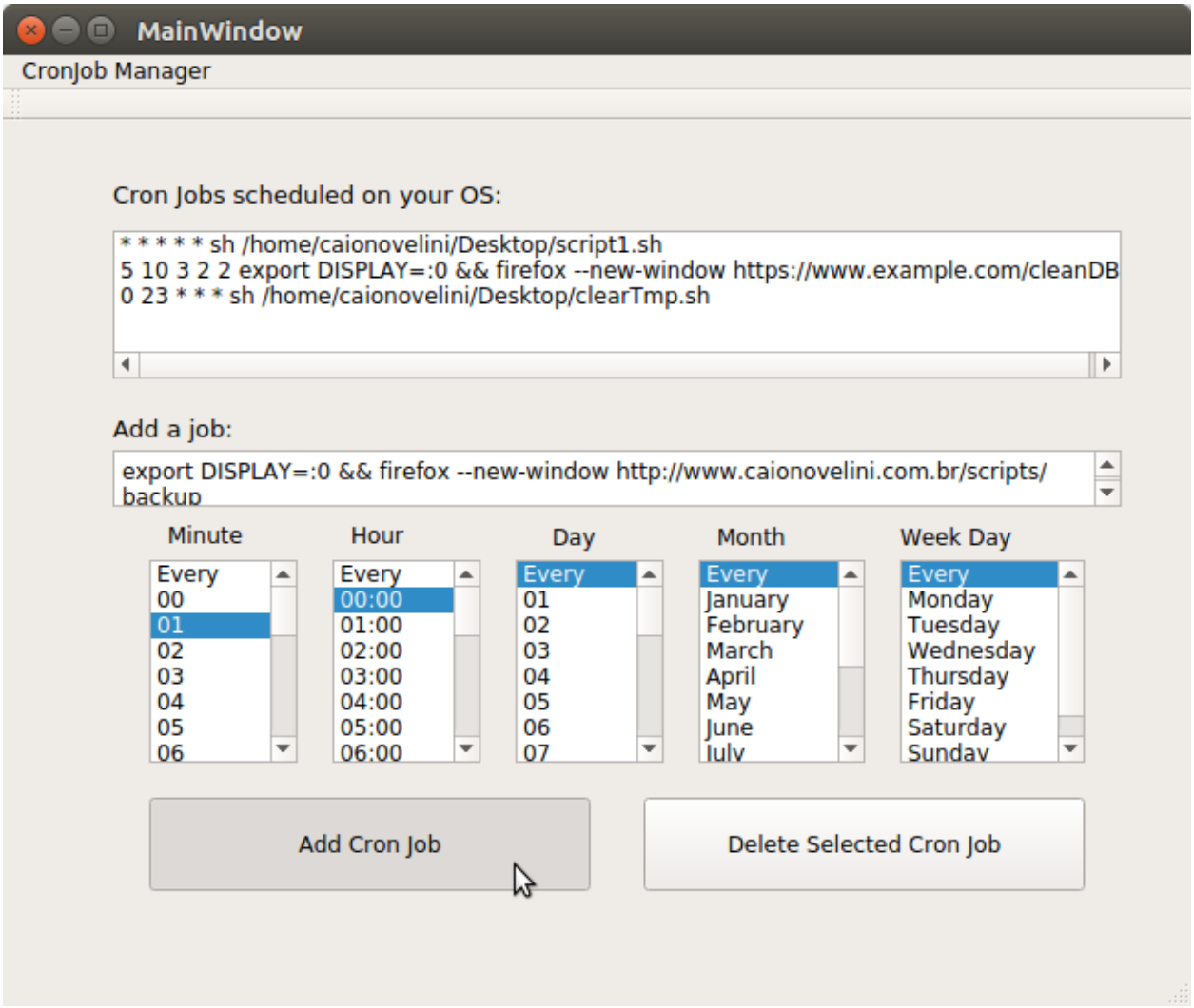


In the image above, the command sh *path* is added to indicate that a shell script will be running. The time parameters were set so that every day at 11:00 pm this file is executed.

Example 2: Open the Firefox web browser in a certain web page every day (in this example we will use the webpage http://www.caionovelini.com.br/scripts/backup that runs a PHP script which performs a backup of the database in the server).

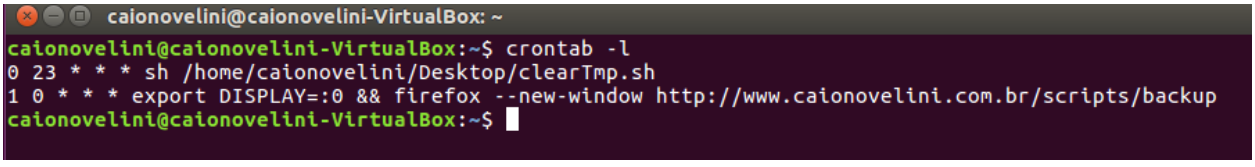
We can accomplish this task by filling the fields in the software in the following way (Image 6).

Image 6: Screenshot of the software scheduling to open a website every day.



After adding the job, we can verify that it was added by opening the terminal application and typing crontab –l (Image 7).

Image 7: Terminal showing the jobs added



Those are just some real life examples in where this software can help the user to schedule jobs in a much easier way.

4. Evaluation

4.1. Usability study

The goal of building this software was to provide a better way to manage cron jobs when comparing to the way Linux OS traditionally offers through terminal. In order to evaluate if the software is actually easier to use we created a usability study.

In the usability study we presented the software to 14 volunteers that are recurring users of cron job in the Linux OS together with a list of questions that was presented after the use of the application. Ideally we wanted more users to test the software, however in our scope of known people it was relatively hard to find regular users of cron job with availability to test the software.

The usability study was conducted as follows :

On a scale from 1 - 5 being: (1) Much worse, (2) Worse, (3) The same, (4) Better, (5) Much better. Compare your experience while using the software with your previous experiences using the native crontab -e command in the OS. To do so, perform and evaluate the following tasks:

1 - Setting the frequency that the job will execute.

2 - Creating a job.

3 - Visualizing the jobs that are scheduled on your machine.

4 - Deleting a job.

5 - Your overall experience.

The table below shows the raw answers acquired in the usability test:

User	**Question 1**	**Question 2**	**Question 3**	**Question 4**	**Question 5**
User 1	5	4	3	4	4
User 2	4	4	3	4	4
User 3	5	4	3	3	3

User 4	5	4	4	4	4
User 5	5	5	3	5	5
User 6	4	3	4	5	4
User 7	5	4	3	4	4
User 8	4	5	3	5	4
User 9	4	5	4	4	4
User 10	5	4	3	4	4
User 11	5	4	4	5	5
User 12	5	4	4	5	5
User 13	4	4	4	4	4
User 14	5	5	5	5	5

The table below shows the results of the usability test:

	1	**2**	**3**	**4**	**5**
Question 1	0.00%	0.00%	0.00%	35.72%	64.28%
Question 2	0.00%	0.00%	7.14%	64.28%	28.58%
Question 3	0.00%	0.00%	50.00%	42.86%	7.14%
Question 4	0.00%	0.00%	0.00%	57.15%	42.85%
Question 5	0.00%	0.00%	0.00%	71.43%	28.57%

By analyzing the resulting table, we can see that all the users had an overall better experience scheduling a cron job using the application when comparing to the use of the command crontab -e offered by the Linux OS.

5. Conclusion

5.1 Summary

The application allow users to manage the cron jobs scheduled in the Linux operating system in a clean and easy way. Tasks such as adding and deleting jobs can be done within few steps and all cron jobs are visible in the screen. The user needs no previous knowledge of the CRON expression to set the frequency that the job will be executed. The friendly interface provides a multi-select option for each field (Minute, Hour, Day, Month and Weekday) where the user can simply select in order to define the frequency of execution.

5.2 Relevance

The application make use of system calls to access and set information to the cron system. By using the information retrieved in the system calls we were able to create a connection between the system (kernel) and the user-interface. As a result, we have a software that perform changes in the OS in an easier way for the end user.

5.3 Future Work

With the use of the software we expect to gather the feedback and improve the application by fixing potential bugs and creating new features that can improve the overall quality of the product.

Some features are still to be implemented in the software, such as tools that allow user to find shell commands in an easy way. Another feature to yet be implemented is a select button that allow the user to select the file to be executed. That way we eliminate the need to know the complete path of the file.

6. Contributions of the team members

Caio Novelini:

Regarding the Software: created the functions that display the cron jobs, created the functions that add the cron jobs, created the user interface, connected the code to the user interface to have the final piece of software.

Regarding the report: Wrote the problem stated, algorithm, results, elaborated and analyzed the usability study and did the conclusion. Was also responsible for the final editing.

Mir Abbas:

Regarding the Software: created the functions related to the deletion of jobs, and many helper functions that transform types of data.

Regarding the report: Wrote the context, background information, implemented the usability study and helped writing the result.

7. References

[1] CronHowTo. Ubuntu Documentation. https://help.ubuntu.com/community/CronHowto .Accessed: November 28, 1016.

[2] Centos.org. https://www.centos.org/docs/5/html/5.1/Deployment_Guide/s2-autotasks-cron-access.html. Accessed: December 02, 2016

[3] Freebsd. https://www.freebsd.org/cgi/man.cgi?crontab%285%29 . Accessed: December 02,2016.

[4] Manual. Qt Documentation. http://doc.qt.io/qtcreator. Accessed: November 30, 1016.