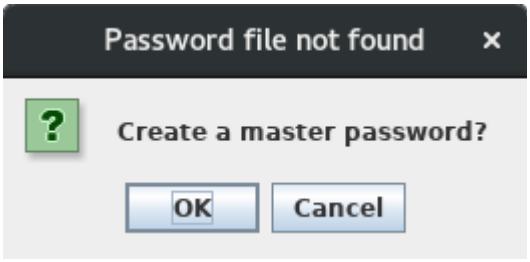
PasswordManager
By Joshua Stone

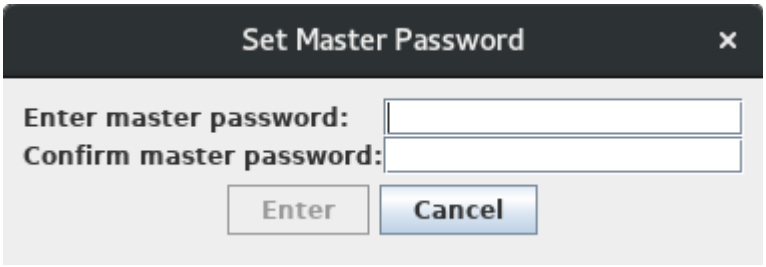
About:
This document is for demonstrating the functionality of the PasswordManager program, a graphical utility that acts as a username and password storage tool and protects data with a master password.

HOWTO:

Step 1: Starting the application
On the first run, the program will check for the existence of a password files used for storing all user data. If no such file exists, then it should ask to create a new master password.

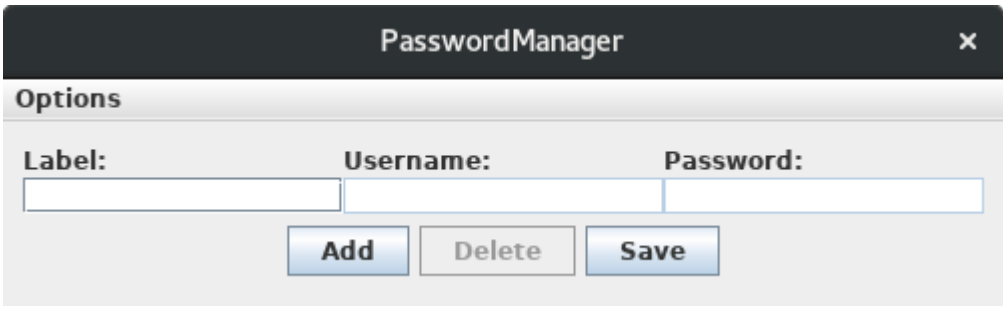


Step 2: Creating the master password
After clicking **Ok**, there should be a new window with two password input fields. The **Enter** button is disabled until both fields have a matching passwords for input.

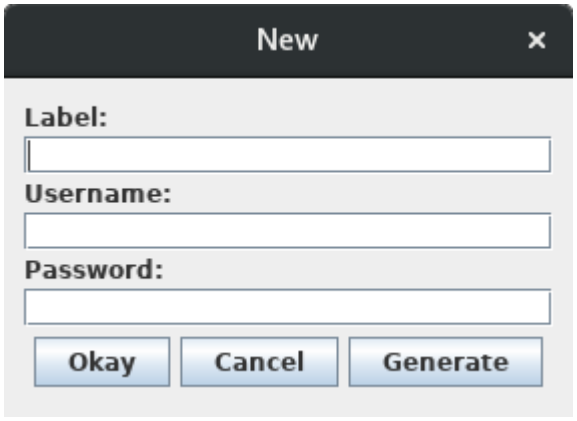


The **Enter** button is disabled until both fields have a matching passwords for input.

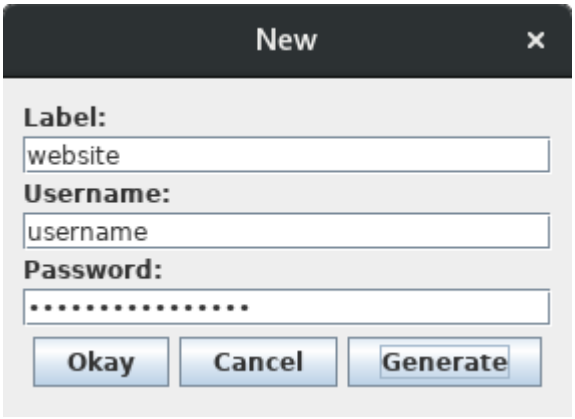
Step 3: Entering the main session
After a master password has been created, a new password store has been initiated and the main session window will open. All fields should be empty and the **Delete** button should be disabled.



Step 4: Adding a new username and password
Pressing **Add** will create a new dialog that'll leave the main window disabled until the new window is closed.

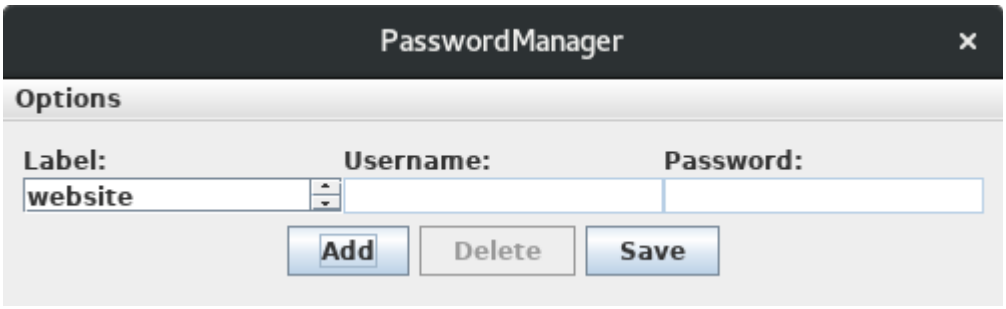


A *label* is for identifying which username and password is used for what. It can the name of a website, email address, etc. Optionally, pressing **Generate** will generate a random combination of letters and numbers in the password field.

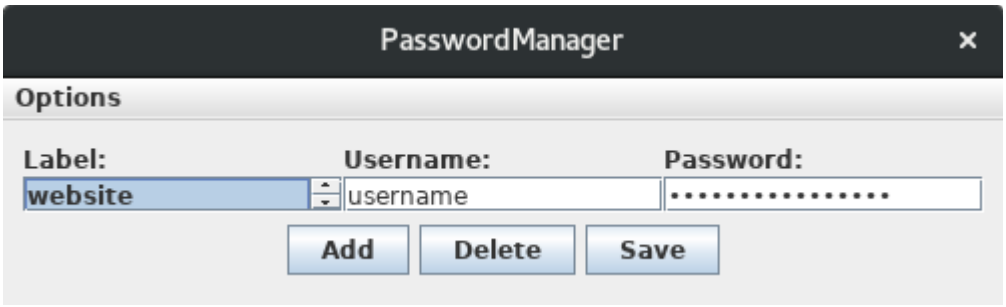


Attempting to press **Okay** with one or more of the fields empty will result in an error message, so be sure to fill all three fields.

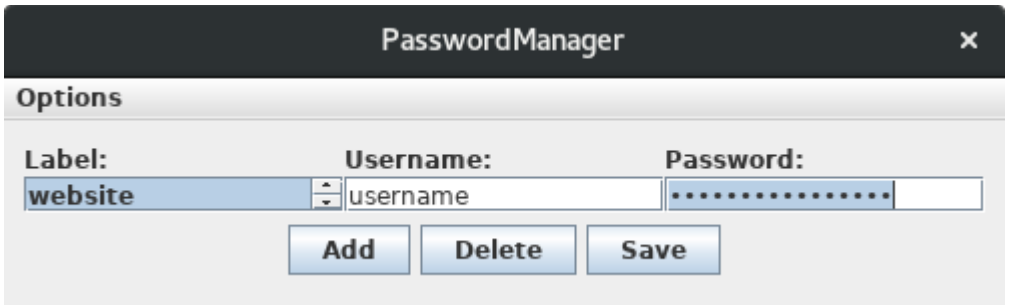
Step 5: Viewing user credentials
After pressing **Okay**, the the newly-entered label should be visible in the main window



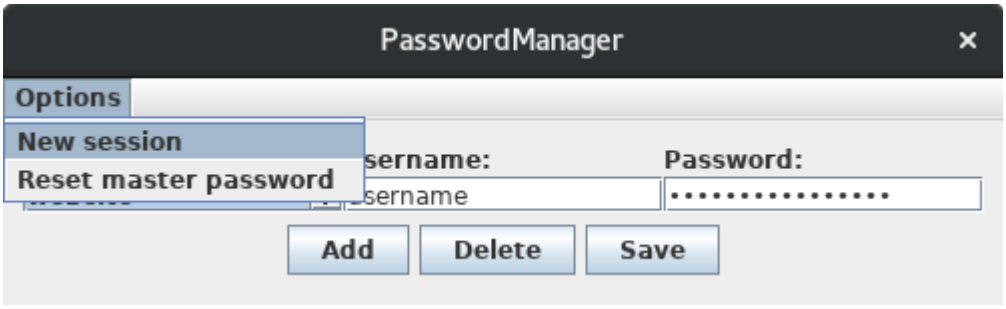
Selecting the label will show the username and password associated with it, as well as making both fields editable. The **Delete** button will also be enabled if one wants to remove credentials.



Step 6: Retrieving a password
Right-click is disabled in the window, instead **Ctrl-C** will copy passwords to the clipboard.

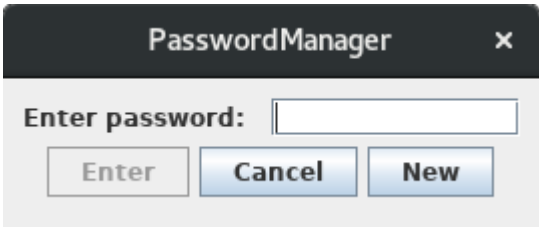


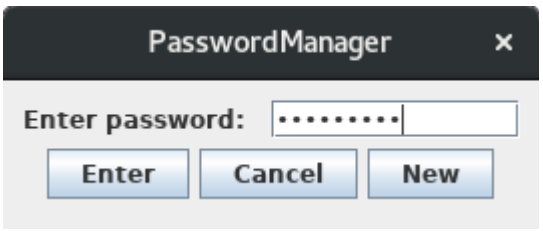
Step 7: Resetting the session or master password
If there's ever a need to reset a session, then just click on the **Options** menu.



Clicking **New session** will remove username and password data while keeping the master password. Clicking **Reset master password** will open a window that looks like to the one in **Step 2.**

Step 8: Saving a session
Once all data has been entered, click **Save** and click the **Close** button (Note: all fields must be filled or else an error message will appear). Restarting the application should open a different window prompting for a password.



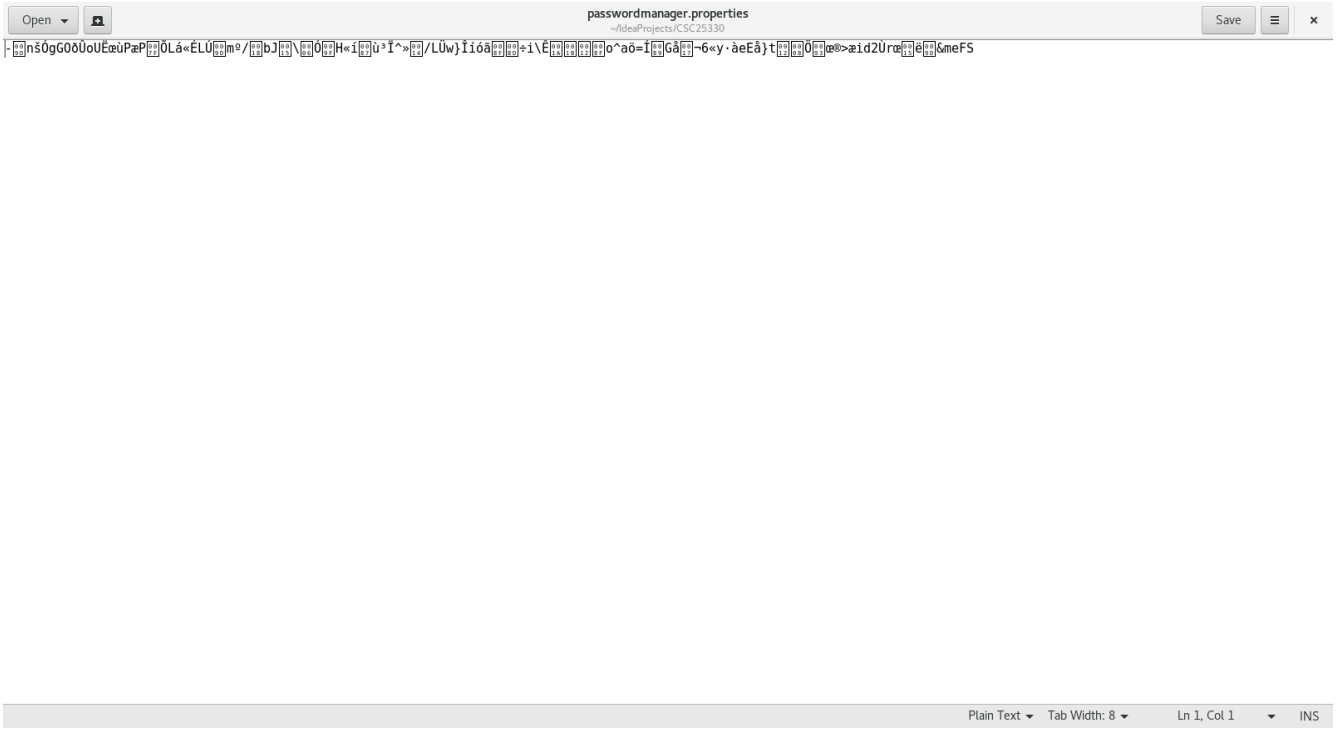


Pressing the **New** button will start a brand new session, although it won't overwrite the previous password file until new data has been entered and **Save** is pressed.

Step 9: Verifying that the passwords are secure
The PasswordManager program is built around standard encryption algorithms for file encryption, where a master password acts as a *key* to unlock the password file and read its contents.

The use of a randomly generated *salt* acts as an additional input of random data, and an *initialization vector* (iv) is used for ensuring randomization when using the same key. This has the property of producing a completely random sequence of bytes every time PasswordManager writes to the disk, even if none of the data changed and the same password is being used.

Data is stored in *passwordmanager.properties* found in the same directory where the program is being run, and attempting to open the file will show completely random data.



Repeatedly pressing **Save** in the application will keep writing a random byte sequence, but every one will contain the exact same data.