Coded File DB - Usage Instructions

A program for storing files in a way that preserves confidentiality.

Introduction

At least Java 1.8 is required to run this program.

To create helper scripts and this manual, run the Jar file. All helper scripts show a message once they finish asking you to press enter before it closes the window.

Setting Up A Database

1. Choose or create a folder to interact with the database in.
2. [**OPTIONAL**] Choose or create a folder where the data will be stored (relative paths are allowed and are relative to the folder from step one, Relative Paths Help Page)
3. Copy the “init database” script into the folder from step 1.
4. Run the copied “init database” script
    a. In the prompt for “enter storage folder path” either:
        i. Type “nevermind” to cancel,
        ii. [**RECOMMENDED**] Leave it empty or type “default” to use the default storage path,
        iii. The path to the folder to store protected information from step 2.

b. In the prompt for “password”, enter the database password and then confirm it. [notably, the passwords will not show up as you type, this is a security feature]. THE PASSWORD CANNOT BE RECOVERED IF LOST.

c. The script has finished, so press enter to close it.

Once init is run, additional helper scripts will be created that allow easy re-opening of the database. Do not move or rename the storage folder, it will break the retrieval connection. If the default storage path was used, the folder containing everything may be moved.

What Does Coded File DB Do?

The following are encrypted:

- folder / group names
- list of all folder group names
- the database password

The following may be put in groups:

- images
- text documents
- zip or winrar archives
- any other files
- NO FOLDERS

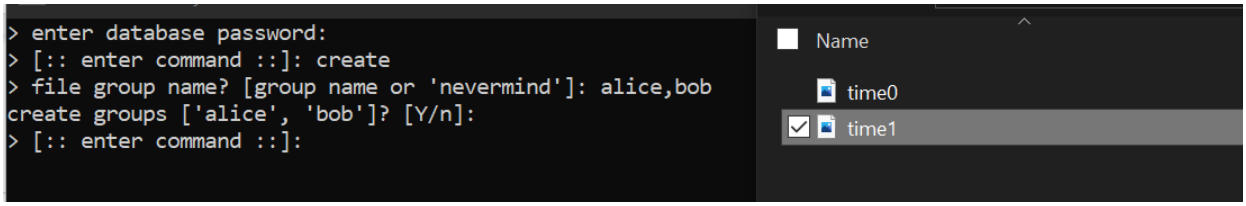
Using A Database

To open a database, run the “open” file created in a folder you ran init in and then, when prompted, enter the database password. To pull up the help screen and a list of commands, type “help” and then enter, or for just a list of commands, press enter with the command line empty.

When a group is created or viewed, its data will be temporarily shown in the working directory, the contents of these subfolders will be stored when “save” or “close” is run. **It will not autosave if the window is closed; if this happens, run “open” again, and then save and enter the names of the groups that got left behind (“*” will not work here because the program forgets about views between executions)**.

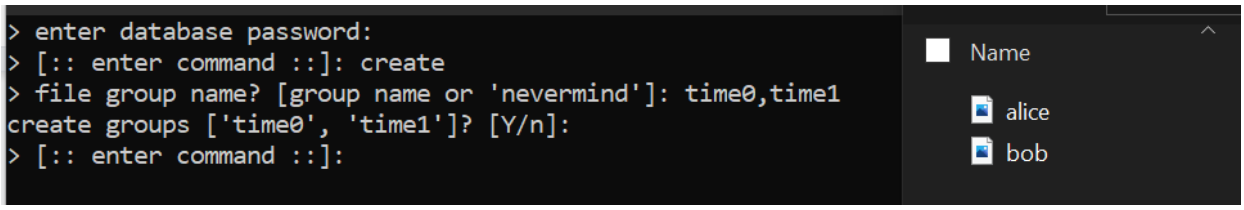
When a group is created, its name will get coded, but files inside will not be encrypted.

SAFE :



Only the names “time0” and “time1” will be visible to an unintended observer. The names “alice” and “bob” will be safely encrypted.

NOT SAFE :



Only the names “alice” and “bob” will be visible, “time0” and “time1” will be encrypted away. This does not protect patient information.

Step by step, here is how to add and protect a patient’s data, decrypt stored data, add more data.

Add new patient’s data:

1. Open and log into the database.
2. Enter command: “create”
3. Enter the patients’ names as the group name, separating different patients’ names with commas (ids may be given instead of names; the “group name” is used to get the data back. Typos in the name will cause issues when attempting to retrieve it, so check spelling).
4. Put the data for the patient in the folder that appeared with the group name that was entered.

5. Enter command: “save”
6. Either enter the group name that was just created or “*”, which saves all.
7. Enter command “close” to close.

Decrypt/edit stored data:

1. Open and log into the database.
2. Enter command: “view”
3. Enter the group names given to the patients, separated by commas, OR, type “*” to view all the stored data.
4. The patients’ data has now appeared in the working directory as folders named with their names/ids. These folders’ contents may now be modified or added to.
5. Enter command: “save”
6. Enter the group names of the patients being viewed or “*”, which saves all.
7. Enter command “close” to close

Command details can be seen by typing “help” on an opened database

---

Relative Paths Help Page

Here is an example absolute path:

- Windows: [`C:\Users\<username>\Desktop\folder\on\desktop`]
- Unix: [`~/Desktop/folder/on/desktop`]

Here are those same folders relative to [`Desktop/folder/`]

- Windows: [`./on/desktop`]
- Unix: [`./on/desktop`]

The “`./`” at the start tells the path that it is relative.

[example relative to `Desktop/folder/`]
Furthermore, relative paths like “`./../another/dir`” will resolve to something like “`C:\Users\<username>\Desktop\another\dir`”. This is because “`../`” tells the path to

go up one directory (in the example, going from "`~/Desktop/folder`" to "`~/Desktop`"). "`../`" may be repeated to go up multiple directories: "`../../../../some/folder`"