MY BACKUP README

version 2.2.1

contact: ammdias@gmail.com

website: https://ammdias.duckdns.org/downloads

Changes history:

- 2.2.1: Corrected bug that prevented an 'ok' to be shown after correctly parsing a composite profile.
- 2.2: Added option to edit the configuration file
- 2.1: Added option to list backup profiles Added option to display configuration file Composite profiles can now only be composed of simple profiles, thus preventing 'infinite loop' profiles
- 2.0: Complete rewrite in Python. Configuration with possible multiple profiles in a separate file.
- 1.0: Original bash script.

ABOUT THE PROGRAM

This program makes easy the backup of a group of directories to another location or remote directory, with the help of an external utility like `rsync` or `cp`. It depends on Python 3.

The following instructions are for Linux systems. If you intend to use the program on another system, please refer to the included MANUAL.

USAGE

Initial test of the program

1. Uncompress the program zip file to a convenient place and edit the configuration file, "mybkp_profiles", with your favorite text editor.

   The configuration file is divided in "profiles", each starting with the profile name between brackets. Each profile must have four values:

   - `command`: the command to perform the backup (e.g. `rsync` or `cp`), including all the commands' options
   - `base`: the parent directory where the directories to backup reside (e.g. your home directory)
   - `backup`: the directory where the backup will reside (e.g. an external disk drive or network storage)
   - `directories`: a comma separated list of directories, below the `base` directory, to backup

   The provided sample file comes with a single profile, named "default":

   ```
   [default]
   command: echo cp --archive --update --verbose --strip-trailing-slashes
   base: /home/USER
   backup: /home/USER/tmp
   directories: Documents, Desktop
   ```

   This profile, when used, will backup the "Documents" and "Desktop" directories under the user's home directory ("/home/USER"), using the command

   ```
   echo cp --archive --update --verbose --strip-trailing-slashes
   ```

   (Actually, it will do nothing but print the commands to the terminal, as that is all the `echo` program does :)

2. To try it out, open a terminal and change to the directory where you uncompressed the program. Then, make sure the program is executable and execute it, passing the configuration file and profile as argument:

   ```
   $ chmod +x mybkp.py
   $ ./mybkp.py --config mybkp_profiles default
   ```

   It should output something like this:

   ```
   Parsing profile: default... ok.
   ```

```
Processing profile: default
Backing up directory Documents ...
cp --archive --update --verbose --strip-trailing-slashes \
   home/USER/Documents/ /home/USER/tmp

Backing up directory Desktop ...
cp --archive --update --verbose --strip-trailing-slashes \
   /home/USER/Desktop/ /home/USER/tmp
```

3. If you want to really test it, remove the `echo` from the `command` value, replace USER with your username and execute the program again. Now, it should actually copy the "Documents" and "Desktop" directories into the "tmp" directory.

   (Of course, you can – probably should – change the "Documents", "Desktop" and "tmp" directories to any others of your choosing).

   If you wish to check the commands that will be issued by the program, instead of actually execute them, use the `--no_act` option:

   ```
   $ ./mybkp.py default --config mybkp_profiles --no_act
   ```

Installation and actual basic usage

The following instructions describe the installation process for basic usage by a single user in a Linux environment. For more advanced instructions, including MS Windows usage, please refer to the user manual in the file "MANUAL" or "MANUAL.pdf".

1. Open a terminal and change to the directory where you uncompressed the program. Execute the local installation script:

   ```
   $ bash local_install.sh
   ```

   This will copy the program to the current user's "~/.local/lib" hidden directory, create a symbolic link to the program in "~/.local/bin", which should be in the user's PATH, and copy the configuration file to "~/.config/mybkp_profiles".

2. To check that the program is working, open *another* terminal and type:

   ```
   $ mybkp --version
   ```

   This should print the program's name and version. If not, check if "~/.local/bin" is in the PATH:

```
$ echo $PATH
```

Also, check that the program was copied to the locations mentioned above and that the symbolic link was created.

3. Now edit the configuration file in "~/.config/mybkp_profiles" with a text editor, changing it to your particular needs.

   Start with something really simple, as making the backup of two or three directories to an external drive. Make sure of including the full paths to the base and backup directories.

4. Run the program in "dry run" mode and check that the copy commands are all correct:

   ```
   $ mybkp default --no_act
   ```

   (notice that when the configuration file is in the correct place, you don't need to include it as an option)

5. Finally, run the program and check that the backup was correctly made:

   ```
   $ mybkp default
   ```

6. To restore the backup (copy the files from the backup location to their original place), use the restore option:

   ```
   $ mybkp default --restore
   ```

7. Read the included manual to learn how to create configuration files with multiple profiles and different backup strategies. For simple command line help you may type:

   ```
   $ mybkp --help
   ```