The KBackup Handbook

Martin Koller

Contents

Abstract

KBackup is an application which lets you back up your data in a simple, user friendly way.

Chapter 1

Introduction

KBackup is a program that lets you back up any folders or files, whereby it uses an easy to use folder tree to select the things to back up. It lets you save your settings in so-called 'profile' files, where a profile is a simple text file containing definitions for folders and files to be included or excluded from the backup process. Also, it lets you define where the backup shall be saved to. The target can be either a local folder (e.g. a locally mounted device like a ZIP drive, USB stick, etc.) but it can also be any remote URL (e.g. smb://remote/some_path) to back up your data to some central server, etc.

The program can also run an automated backup without using a graphical user interface. One can simply create a profile and use these settings to tell KBackup what to do when running in non-interactive mode, e.g. by starting it from a cron job.

The program was designed to be very simple in its use so that it can be used by non-computer experts.

The storage format is the well known TAR format, whereby the data can still be stored in compressed format (xz, bzip2 or gzip).

The current implementation features only the backup step. To restore data back into your system, you currently have to use, e.g., Dolphin to open the TAR backup files and drag/drop the files back to your file system. This is also an advantage of the usage of the well known and well supported TAR file format.

If the files are compressed, you can uncompress all files from the current folder recursively down with the following command:

```
find . -name \*bz2 -print0 | xargs -0 bunzip2
```

Alternatively you can use Ark to extract a full backup or just a few files from a backup.

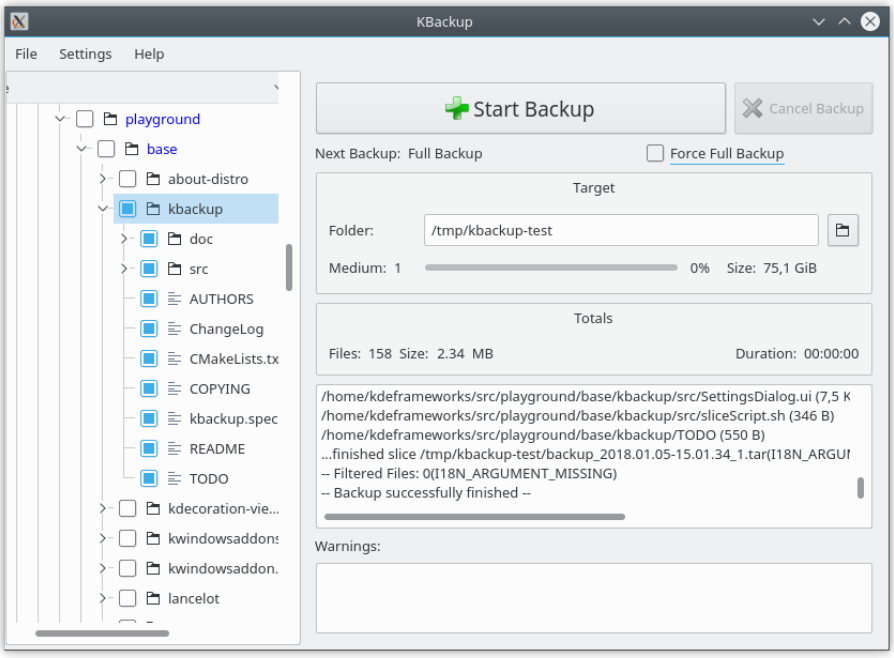
Chapter 2

Using KBackup

All you need to do is to select which folders you want to store. This is done by selecting all the folders in the tree view on the left side of the main window.

If you select a folder, KBackup will automatically store all files and subfolders below it. If you want to exclude parts of a selected folder, simply deselect that files/folders inside the still selected folder.

In general, this means: A selected folder will store everything in it and below it, except the deselected parts in it. This also means, whenever you reuse a profile (see below) later on and new files have been added to a folder already selected for backup, all the new files will also be stored.



2.1 Using profiles

To keep a selection for later use, simply save it into a KBackup profile file. Use the **File** menu and select **Save Profile**.

To reload a selection into KBackup, use the **File** → **Load Profile...** menu item.

KBackup saves in a profile the selections for all included folders/files, excluded folders/files, the target folder/URL, defined archive prefix, the defined maximum slice file size, etc.

If you want to ease the usage of backing up every day the same set of files, simply store your settings into a KBackup profile (a `.kbp` file) and pass that file on the command line.

e.g.:

```
kbackup myData.kbp
```

Hint: you can also create a shortcut on your desktop to a `.kbp` file as the file type is registered to start KBackup on double click.

2.2 Archive slices

As a medium has normally limited capacity (e.g. 100MB ZIP disc), KBackup will create several archive slices.

Each archive slice will get its own name, which looks like this:

backup_2006.08.26-13.04.44_1.tar

The name contains the creation date and time (which will be the same for all slices of one backup) and a trailing slice sequence number (_1 in this example).

In the menu **File** → **Profile Settings...**, you can define a different archive prefix than the default 'backup'.

In the **Profile Settings** dialog, you can also define a maximum archive slice size, which limits the slice size even if there would be more space left on the target device. This helps to create archive slices which can then be later burned on a CD/DVD, etc. If you explicitly limit the size of an archive slice, the available size will be marked with (*) in the main window.

But even if you define a slice to be of 'unlimited' size, there are other constraints which limit the size of a slice:

- limited by the target folder (when stored directly into a local folder)
- limited by the `/tmp` folder when we create a tmp file for later upload to a remote URL

In the **Profile Settings**, you can also define a maximum number of backups being kept in the target folder, and therefore automatically deleting all older backups there. e.g. if you set it to 3, KBackup will keep the last 3 backups and delete all older ones.

2.3 Incremental Backup

With an incremental backup not all files will be saved every time the backup runs, but only the files which have changed since the last backup. This has the great advantage that the incremental backup will usually include much fewer files than a full backup and therefore will be finished in a much shorter time.

This works as follows: In the profile, you define an interval (in days) for the full backup. e.g. when you define 5 days, then KBackup will do a full backup of all files every 5 days. Whenever you start KBackup before the interval expires with this profile - regardless how often you run a backup - only the files which have changed since the last backup will be saved. KBackup stores the time stamp of the last backup into the profile and knows therefore what to do when running the next time.

The archive slice files created during an incremental backup will contain the text '_inc', e.g.:

backup_2010.06.14-18.50.26_1_inc.tar

Full backup slice files will not include '_inc' in the name, e.g.:

backup_2010.06.13-20.58.14_1.tar

When one wants to restore files from an incremental backup, it's important to look for the most recent version of a file to be restored in all '_inc' files and finally also the last full backup slice file. This exactly is also the disadvantage of the incremental backup (but no advantage without disadvantage ;-))

If you want to do a full backup earlier than the defined incremental cycle time defined in a profile, you can do so by checking the **Force Full Backup** option in the user interface. When KBackup is started via the command line, this can be achieved by using the option `--forceFull`

A forced full backup will restart the backup cycle, i.e. KBackup counts the days to the next full backup from the time of the last full backup.

2.4 Archive Compression

KBackup will compress the files stored if you activate this in the profile settings. Depending on the availability on your system it chooses **xz**, **bzip2** or **gzip** compression. KBackup will compress every single file and store all files with an added file extension (`.xz`, `.bzip2` or `.gz`) into the then not-compressed `.tar` archive.

When you have selected to create the backup on some local filesystem (e.g. your extra disc, ZIP drive, etc.) - which means you did not enter a remote target URL - KBackup might split the whole backup into several archive slices due to media capacity limitations.

e.g.:

backup_2006.08.26-13.04.44_1.tar

backup_2006.08.26-13.04.44_2.tar

2.5 Automating Backup

If you want to automate the process of the backup, KBackup offers different command line options to help with this:

- `--auto`
  When you run KBackup with this option and a given `.kbp` profile, it will start, load the given profile, run the backup and terminate when done. All this is done with a visible KBackup user interface.
- `--autobg`
  When you run KBackup with this option and a given `.kbp` profile, it will run the same process as with `--auto` but *without* showing any graphical user interface. Therefore the suffix 'bg' which stands for 'background' - everything is done in the background so this is the right option to be used when you do your backups, e.g., started by a cron job.

  When using `--autobg` the output from KBackup - showing the progress of the backup - is written to stderr. By default, the output includes just a few important messages and a summary at the end. If you pass `--verbose` in addition, then you will also see each file name currently being backed up.

Chapter 3

Command Reference

3.1 The main KBackup window

3.1.1 The File Menu

File → Open Recent
Shows a submenu with the recently used profiles for easy selection.

File → New Profile
Clears the selection and the target input field, to be able to define a new profile.

File → Load Profile...
Loads a profile.

File → Save Profile
Saves all settings into the currently loaded profile.

File → Save Profile As...
Saves all settings into a profile with a new name.

File → Profile Settings...

In the settings, you can define whether the archive-slice files start with the default name 'backup' or with an alternative name. Also you can limit the archive slice size. See chapter Archive Slices. These settings are also stored into the profile.

File → Quit (Ctrl+Q)
Quits KBackup.

3.1.2 The Settings Menu

Settings → Dock in System Tray

When this option is activated, an icon is shown in the system tray, which reflects the current status of a backup operation. An animation is shown when a backup is in progress, else you see a static icon. If this option is selected, the closing of the main window will not terminate KBackup. The application must be explicitly terminated by selecting the **Quit** action. Via the context menu of the KBackup system tray icon you can start and cancel a backup operation - which is the same as you can do in the main window. The tooltip on this icon also gives progress information (Number of saved files, size of the backup and the last saved file).

Settings → Enable All Messages

Activating this entry clears all internally stored **Do not ask again** flags for the dialogs shown in KBackup.

Settings → Show Hidden Files

Enable or disable the display of hidden files (preceded by a dot character) in the tree view. Use this option if you want to exclude some hidden files from the backup. If you want to exclude all hidden files, use a filename filter in the profile settings.

3.1.3 The Help Menu

KBackup has the common KDE **Help** menu item, for more information read the section about the Help Menu of the KDE Fundamentals.

Chapter 4

Developer's Guide to KBackup

KBackup can be extended by using a shell script (or any other executable) which will be started at three different points during the backup process. The idea behind it is to allow to mount, unmount, eject media in a system specific way, or do other things with the produced archive files.

The script to execute must be given with the `--script` command line option.

Here is a sample script:

Example 4.1 sliceScript.sh

```
#!/bin/sh

mode=$1
archive=$2
target=$3
mountPoint=$4

case "$mode" in
 "slice_init" )
    if [ "$mountPoint" != "" ]
    then
      mount /media/zip
      rm -f /media/zip/backup_2*.tar*
    fi
 ;;

 "slice_closed" )
 ;;

 "slice_finished" )
    if [ "$mountPoint" != "" ]
    then
      umount /media/zip
      eject /media/zip
    fi
 ;;
esac
```

The script is always invoked with four command line arguments:

- invocation mode
- archive (slice) file name
- target directory/URL
- mountpoint of the target directory if it's a local directory, else an empty string

There are three possible invocation modes:

- slice_init
  called before a new archive slice is being created on disc
- slice_closed
  called after an archive slice has been created, but before it has been put into the target directory
  This can be used if you want to copy the archive slice to some additional place, e.g. the archive is sent to the main server (via a target URL), but you want to keep the last backup also onto the local disc.
- slice_finished
  called after an archive slice has been successfully transferred into the target directory

Chapter 5

Credits and License

KBackup