Easy Backup

User's Manual

Date: 23 Dec 2009
Author: Christoph Schudel, CTS, SIL PNG
Office: cts-prog2@sil.org.pg
Private: christoph.schudel@sil.org

Table of Contents

1. What is new?

1.1. Version 1.3 (Jan 2010)

1.1.1 General

- Changed failure handling.
  When a failure occurs EB continues with the next back definition and doesn't abort the processing altogether. In this way failure to backup one definition does not prevent EB from trying to backup other definitions.
- Option to display a window at the end that displays information what went wrong when during processing a failure occurred. This window gets displayed when pressing the appropriate button that becomes visible when a failure occurred.
- Server Mode: If EasyBackup doesn't find the backup folder on the stick it does not become visible. (in Windows EB quits silently)

1.1.2 Windows

- Fixed an issue when logging to screen (console) and python runs without console.

1.1.3 Linux

- Supports (runs under) Ubuntu 9.10

1.2. Version 1.2 (Nov 2009)

1.2.1 General

- Made buttons bigger on screens
- *EasyBackup* configuration: Main menu: moved items around
- *EasyBackup* Configuration Screen:
  - moved items around
  - Added "Advanced" button to hide / show some of the options
  - Changed logic and renamed tag "Ignore USB launch" to "Start backup automatically when inserting USB stick"
  - Changed checkbox "allow backup to any stick" to two radio buttons: "Backup only if folder 'basilBackup' is found on USB stick" and "Backup to any inserted USB stick"
- Some rewording of texts
- Indicates with a bar used space on backup media (all the space that is being used, not just the space that is being used by *EasyBackup*)
- Added functionality to run *EasyBack* in "Server mode". This means EB can be installed on a central server machine. Whenever a user inserts a USB stick it will copy the content of the EB folder from the stick to the server.
- Removed "ActivityLogger" so that only the "SystemLogger" remains.
- Display Version number on screen
- When clicking the button "Do not backup now" the UI just goes away.

1.2.2 Windows

- *EasyBackup*: When selecting a root folder to be backed up it defaults to "My Documents" (Windows XP) or "Documents" (Windows Vista)

1.2.3 Linux

- Changing configuration settings (i.e. changing the time the interrupt screen gets displayed) does not require logging off and on again. However, changes to log settings (log to screen, number of log files, etc) still require the user to log off and on.
- Removed any code related to XO sugar interface
- *EasyBackup*: When selecting a root folder to be backed up it defaults the user's home folder.
- Bug fix: When big sticks were inserted *EasyBackup* was trying to determine the mount point of the inserted stick too early while it still was being mounted. Now *EasyBackup* tries several times to determine the mount point.

1.3. Version 1.1 (Sept 2009)

1.3.1 General

- It's possible now to configure *EasyBackup* so that it does not do a backup to any USB stick that gets inserted but only on specified sticks.
  (This is **not** a solution for the use in sensitive countries. *EasyBackup* will just check if the backup folder "basilBackup" exists or not).
- More configuration options when configuring *EasyBackup*:
  - Logging options (log file size, number of rotating log files, logging to screen)
  - Set length of time the "interrupt" screen gets displayed
  - Flag to ignore launch when a USB stick gets inserted.
- Corrected some spelling mistakes.
- Native OLPC XO (Sugar) not supported any longer.
- Additional exit code when backup terminates.
- Changed some titles and labels.

1.3.2 Windows

- Backup doesn't get triggered anymore when inserting a CD or mapping a network drive.
- USB insert detection program is now also a python script, rather then a C# .net application. This means the .net runtime library is not needed any longer and therefore the download is smaller.
- Enhanced and much easier installation. Installation now uses NSIS (Nullsoft Scriptable Install System). Third party installers get also launched by NSIS.
- There are two downloads available, one that contains third party software that are needed and one without third party software in case they are already installed.
- Moved the log file location to the user's home folder as opposed to the folder where the python scripts are located.

1.3.3 Linux

- Creates a desktop shortcut for the *EasyBackup* configuration program.
- Corrected installation for Ubuntu. Should work now.
- Corrected on Ubuntu: Backup only gets triggered when USB stick gets inserted, not when removed.

2. Introduction

2.1. General

EasyBackup is a simple and easy to use user interface that allows creating one or more "backup definition". The actual backup is automatically done whenever the user inserts a USB stick into the computer. *EasyBackup* can be configured so that it won't work just on **any** USB stick, but only on

those that have been prepared. (It will basically look if the backup folder *basilBackup* exists on the drive or not.)
Each location defined with *EasyBackup* will get backed up. *EasyBackup* runs under Windows (XP, Vista, Windows 7) and Linux (Ubuntu and Fedora).
Support for the XO (OLPC) with "Sugar" interface has been dropped.

2.2. Backup Strategy

Backing up of the data is done by compressing them into a single "zip" file. The system keeps an adequate number of these "zip" files and automatically deletes older ones. This is done using a smart algorithm that ensures that the backup device is not suddenly full. Smart Deletion is responsible for keeping an adequate number of backup files without running out of space on the backup device. It will keep a **maximum** of 18 backup files per backup definition. If space becomes rare it will start deleting some of these 18 files until there is enough space for the new backup or there is only one backup left. It will never delete all the backups. If there is only one backup left with no space for a new the device is regarded as "full" and any further backup attempts will fail.

Each backup falls into one of these categories. Each category (apart from the "day" (D) category and the "year" (Y) category) holds one or two backup files if there is enough space on the backup device.

Category	**Descriptions**	**Number of files**
1D, 2D ,3D, 4D, 5D, 6D, 7D	The last 7 days	1 each → 7
1W, 2W	One week and two weeks	1 or 2 each → 2-4
1M, 3M, 6M	1 Month, 3 months, 6 months	1 or 2 each → 3-6
1Y	1 Year	1 each → 1
		Total: max 18

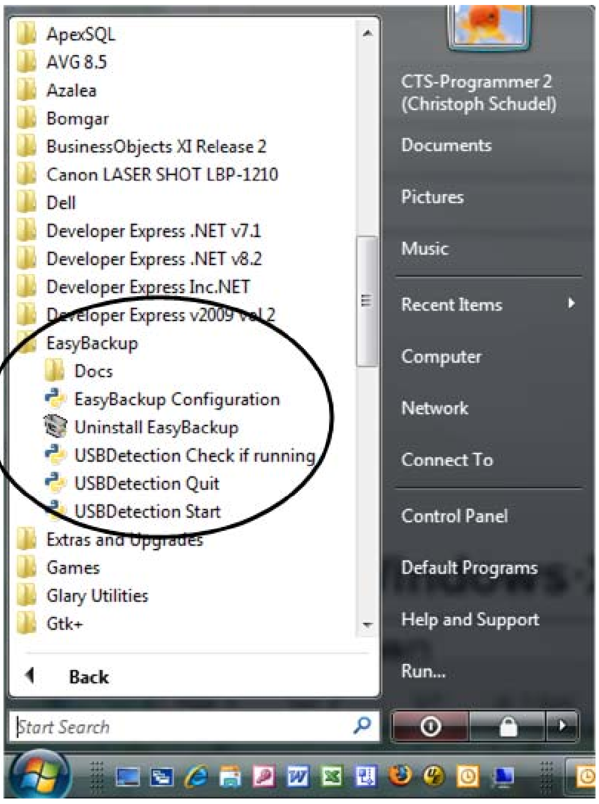
This means the system keeps a daily backup for the last 7 days (assuming a daily backup is done). It then keeps one or two backups (the youngest and the oldest) which are between 1 week and 2 weeks old, one or two backups that are between 2 weeks and 1 month old. It further keeps one or two backups that are between 1 and 3 months old, one or two that are between 3 and 6 months old, and one or two that are between 6 months and one year old. It finally keeps one backup that is older than one year.

The backup zip files are saved on the USB stick in the folder "basilBackup". Each backup definition has its own subfolder in which the zip files get copied. Each zip file contains all the data plus a log file "ziplog.txt" that contains information such as the roof folder of the data, excluded folders / files, included files as well actions taken to make space on the media.

3. Program Launch

3.1. Windows

Go to Start → All Programs → EasyBackup → EasyBackup Configuration



There are a number of other shortcuts available under the EasyBackup entry.
These shortcuts are described further below.

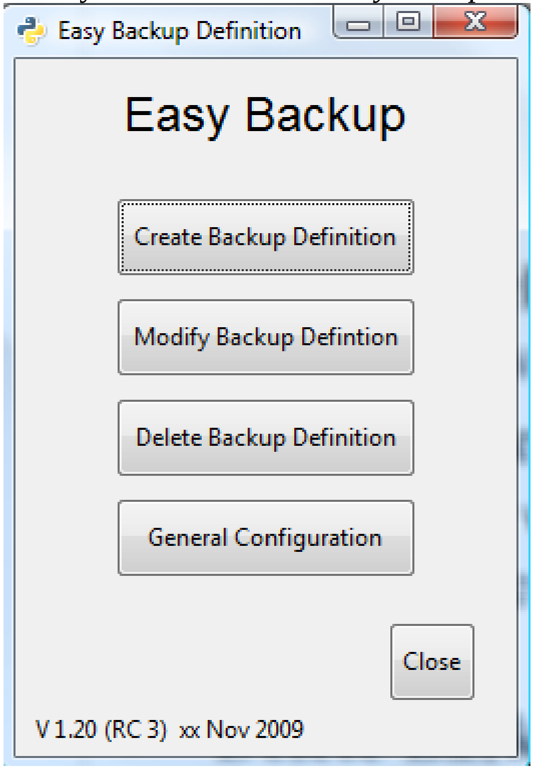
3.2. Linux

There should be a shortcut on the desktop called *EasyBackup Config*. Double click on it. If the shortcut does not exist, open a terminal window and then type:

```
python /usr/local/bin/bdec/basilEasyBackup.py
```

3.3. Main Menu

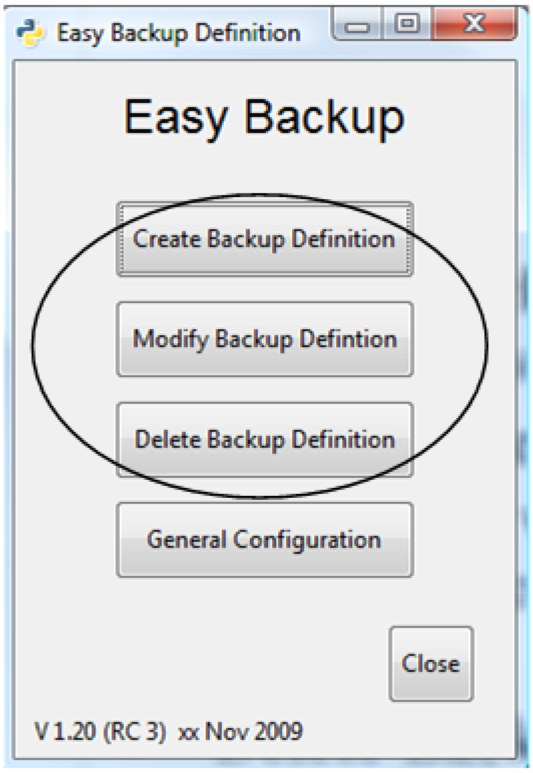
After you have launched *EasyBackup Configuration* you have these options:



- **Create Backup Definition**: Allows you to create a new backup definition
- **Modify Backup Definition**: Allows you to modify any existing backup definition. This included the root folder and optional settings.
- **Delete Backup Definition**: This will delete the specified definition.
- **General Configuration**: Allows you to configure *EasyBackup*.

4. Define a Backup

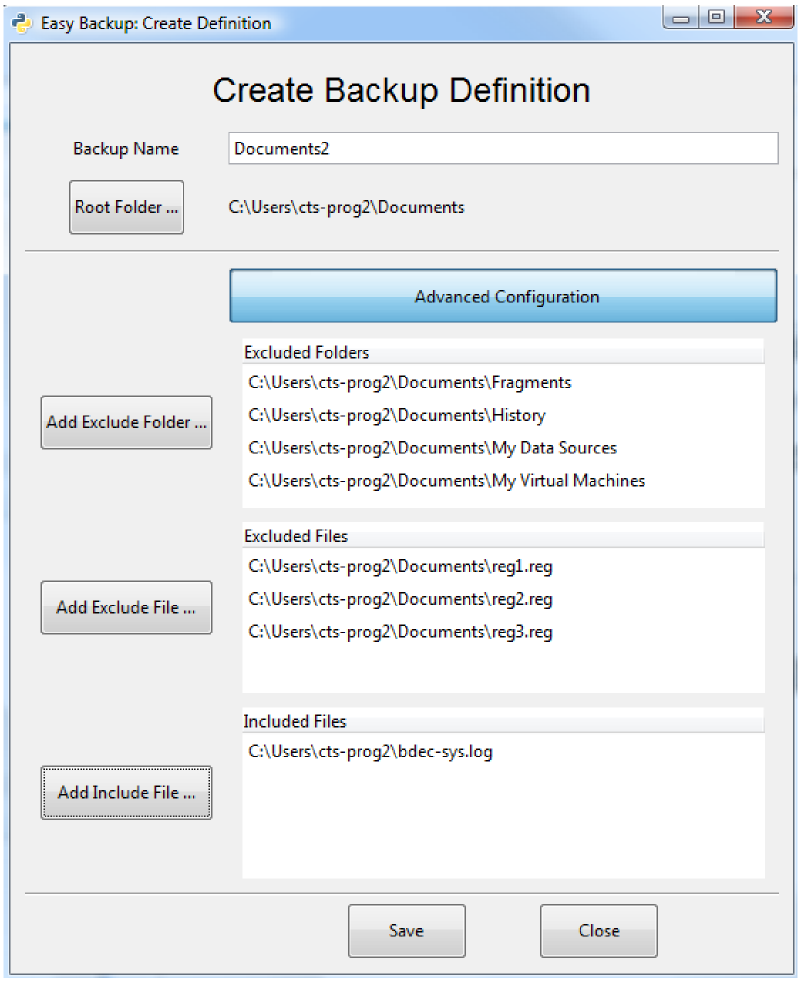
With *EasyBackup* you can create one or more “backup definition”. A “backup definition” consists of a name for the backup plus the name of the root folder that should get backed up. Normally the files in the root folder and all files in all its subfolders get backed up, unless otherwise specified with optional settings. Optional settings allow you to exclude entire folders (and its subfolders) or files within the root folder tree from being backed up. For example if you defined “my documents” as root folder you then can exclude “my pictures”, “my music” from being backed up. In addition optional settings allow you to include specific files from anywhere to be included in the backup.



On the main menu select on of the following options:

- **Create Backup Definition**: Allows you to create a new backup definition
- **Modify Backup Definition**: Allows you to modify any existing backup definition. This included the root folder and optional settings.
- **Delete Backup Definition**: This will delete the specified definition.

4.1. Create Backup Definition



Backup Name: Documents2

Root Folder: C:\Users\cts-prog2\Documents

Excluded Folders
- C:\Users\cts-prog2\Documents\Fragments
- C:\Users\cts-prog2\Documents\History
- C:\Users\cts-prog2\Documents\My Data Sources
- C:\Users\cts-prog2\Documents\My Virtual Machines

Excluded Files
- C:\Users\cts-prog2\Documents\reg1.reg
- C:\Users\cts-prog2\Documents\reg2.reg
- C:\Users\cts-prog2\Documents\reg3.reg

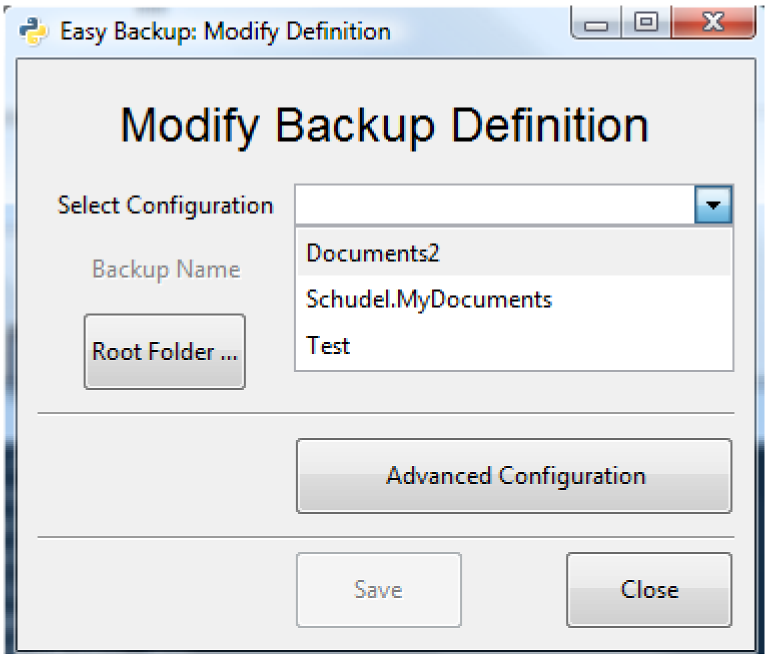
Included Files
- C:\Users\cts-prog2\bdec-sys.log

Only the *Backup Name* and the *Root Folder* are required.

Advanced Configuration contains additional (optional) settings to exclude folders / files from being backed up and additional files that are included in the backup.

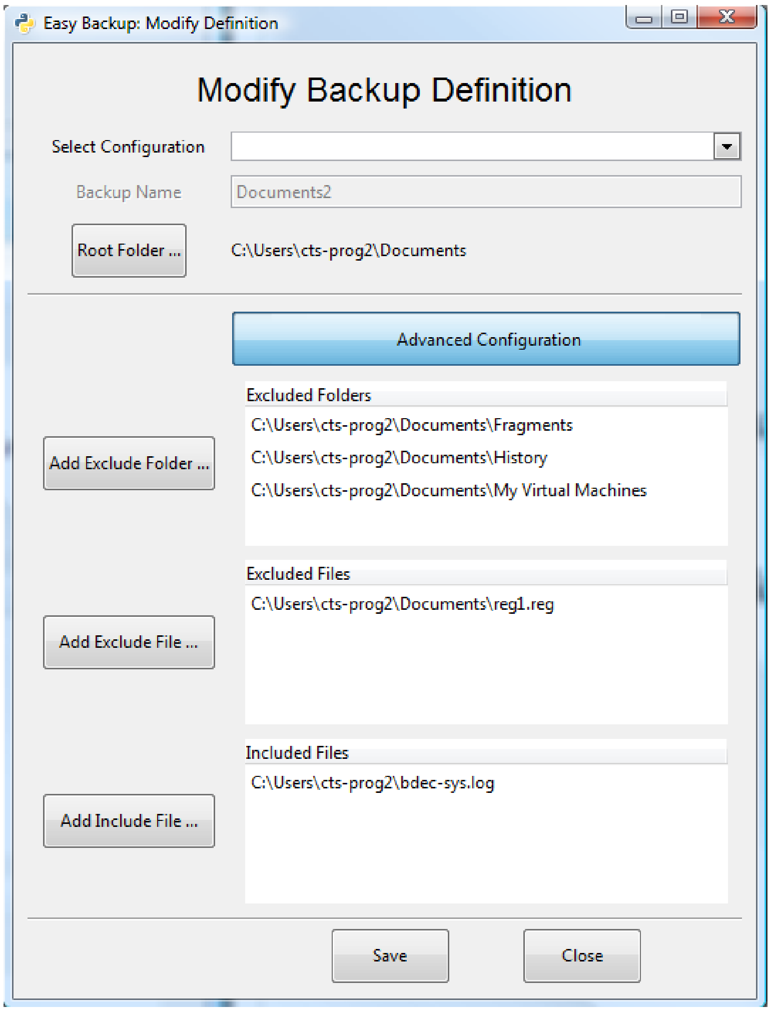
Please note that it is **not** possible to specify file patterns.

4.2. Modify Backup Definition



Select Configuration options:
- Documents2
- Schudel.MyDocuments
- Test

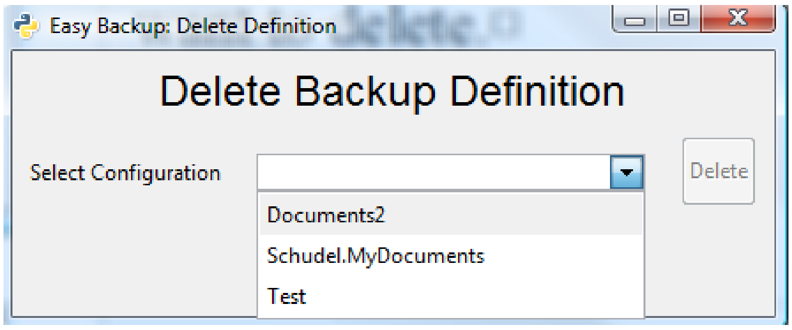
Select a configuration that you want to modify.


Easy Backup: Modify Definition
Modify Backup Definition
Select Configuration
Backup Name
Documents2
Root Folder ...
C:\Users\cts-prog2\Documents
Advanced Configuration
Add Exclude Folder ...
Excluded Folders
C:\Users\cts-prog2\Documents\Fragments
C:\Users\cts-prog2\Documents\History
C:\Users\cts-prog2\Documents\My Virtual Machines
Add Exclude File ...
Excluded Files
C:\Users\cts-prog2\Documents\reg1.reg
Add Include File ...
Included Files
C:\Users\cts-prog2\bdec-sys.log
Save
Close

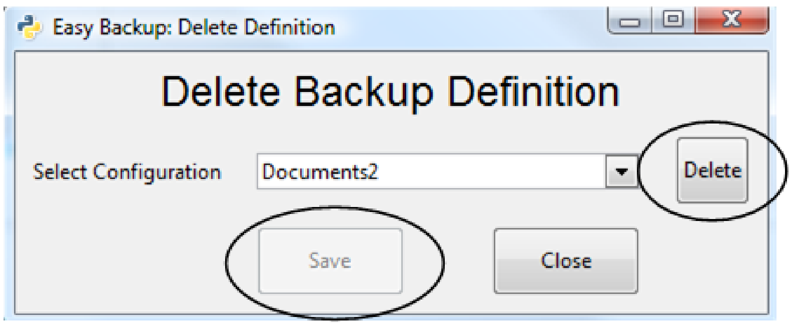
Make the appropriate changes and Save.

You need to double click on an entry to remove it.

4.3. Delete Backup Definition

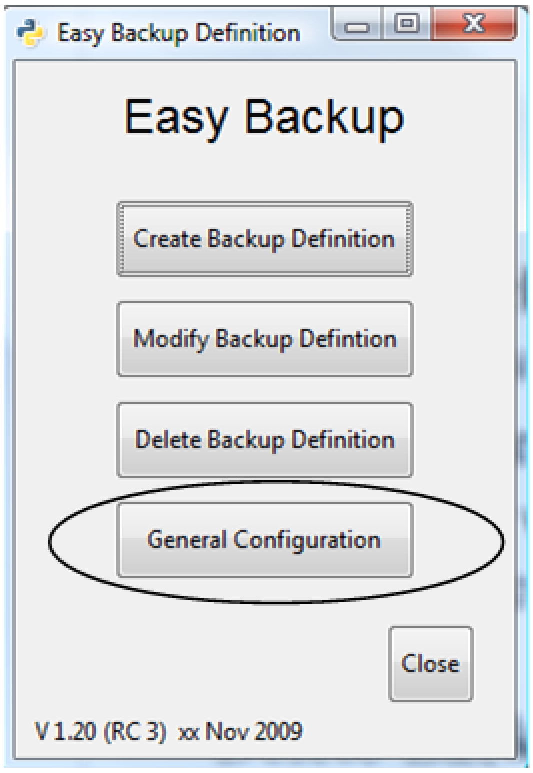


Select the configuration that you want to delete.

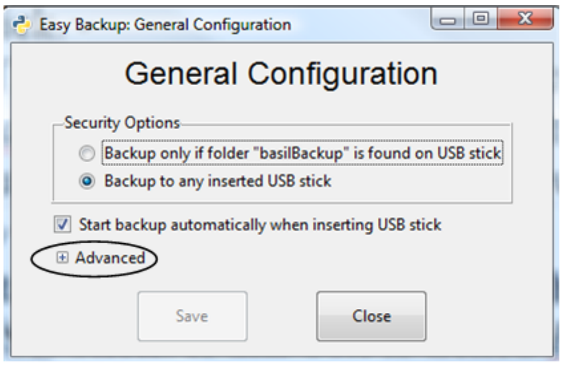


Then hit *Delete* and after that *Save*.

5. General Configuration



On the main menu click on the *General Configuration* button.



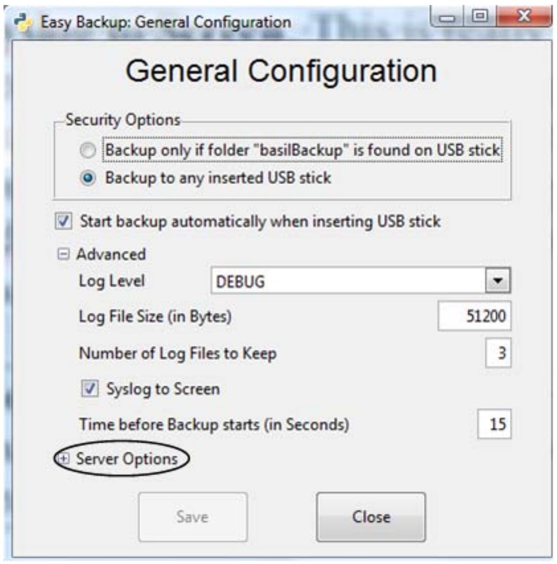
Security Options: Backup only if folder "basilBackup" is found on USB stick [● Backup to any inserted USB stick]

[✓] Start backup automatically when inserting USB stick

You have these options:

- **Security Options**: By selecting one of the two radio buttons you can specify if *EasyBackup* will do a backup to any stick that gets inserted (2nd radio button) or if it will only do a backup if it finds the folder “basilBackup” on the inserted stick.
  (The “basilBackup” folder is the folder where *EasyBackup* puts the backups. It gets automatically created when allowing backing up to any stick)
- **Start backup automatically when inserting USB stick**: If this box is ticked *EasyBackup* will start when a USB stick gets inserted and do a backup. If it is not ticked *EasyBackup* will ignore the insertion of an USB stick. With this you can temporarily “switch off” *EasyBackup*.

If you expand the *Advanced* “+” sign more options become available:



Easy Backup: General Configuration

General Configuration

Security Options
- ○ Backup only if folder "basilBackup" is found on USB stick
- ● Backup to any inserted USB stick

☑ Start backup automatically when inserting USB stick

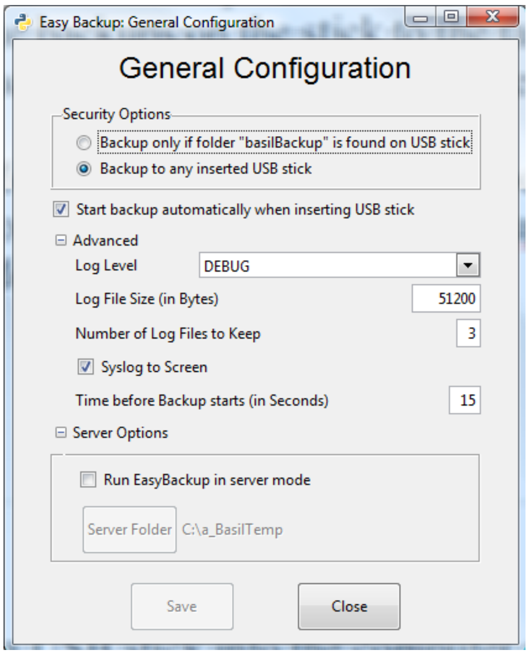
⊟ Advanced
- Log Level: DEBUG
- Log File Size (in Bytes): 51200
- Number of Log Files to Keep: 3
- ☑ Syslog to Screen
- Time before Backup starts (in Seconds): 15

⊞ Server Options

- **Log Level**: defines how much information gets logged to the syslog.
- **Log File Size (in Bytes)**: Defines how big the syslog file *bdec-sys.log* (under Windows) and the activity log file *bdec-activity.log* (Windows and Linux) can grow.
- **Number of Log Files to Keep**: Defines how many log files the system should keep before they get deleted.
- **Syslog to Screen**: This is really only needed when you run the program *bdecMain.py* (the program, that does the actual backup) from a command line for debug purposes. Each syslog message that bdecMain produces will then also be displayed on the command window.

Note: Under **Linux** the settings related to logging require the user to log off and log on again in order to take effect.

- **Time before Backup starts (in Seconds)**: Whenever you insert a USB stick and *EasyBackup* becomes active, you have a certain amount of time to stop *EasyBackup* from beginning doing the backup. (See description further below). This value sets this time.
- For **Server Options**: Expand the "+" sign.



Easy Backup: General Configuration

General Configuration

Security Options
- ○ Backup only if folder "basilBackup" is found on USB stick
- ● Backup to any inserted USB stick

☑ Start backup automatically when inserting USB stick

⊟ Advanced
- Log Level: DEBUG
- Log File Size (in Bytes): 51200
- Number of Log Files to Keep: 3
- ☑ Syslog to Screen
- Time before Backup starts (in Seconds): 15

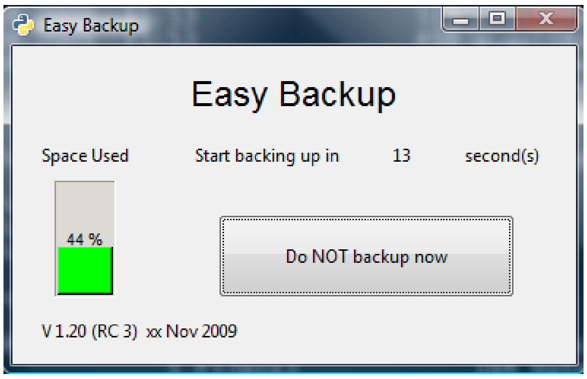
⊟ Server Options
- ☐ Run EasyBackup in server mode
- Server Folder: C:\a_BasilTemp

- **Server Options**: Tick the checkbox "Run EasyBackup in server mode" if you want to have *EasyBackup* act as server. This means when a stick gets inserted to the machine *EasyBackup* will not do a backup to the stick; instead it will copy the backups on the stick to the target location on the server as specified under "Server Folder".

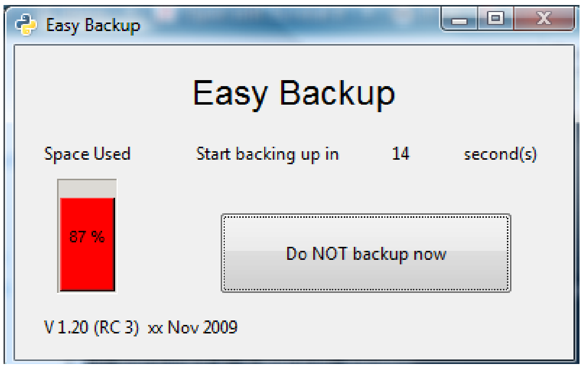
  More information about this mode follows further down in the section "EasyBackup in Server Mode"

6. Running a backup

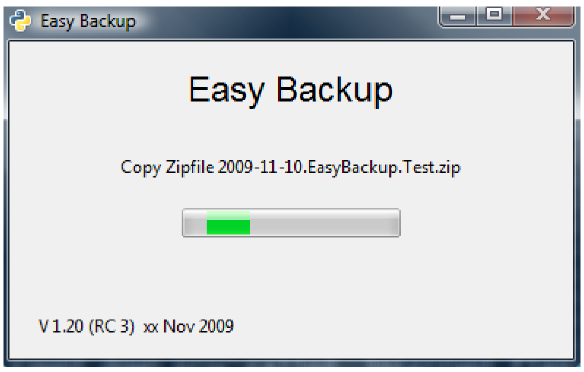
Running the actual backup is very easy. Just insert a USB stick into the computer and all your data that you specified in backup definitions get backed up:



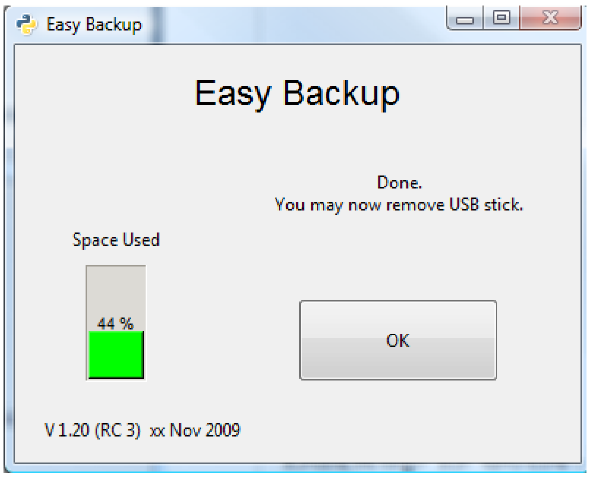
After a USB has been inserted into the computer this window comes up for a few seconds and if not interrupted then proceeds to do the backup. This value can be set under *General Configuration* as described further up.
On the left hand side a graphic bar indicates the space already used in the inserted disk.



If the inserted USB stick is more than 85% full the bar turns red.

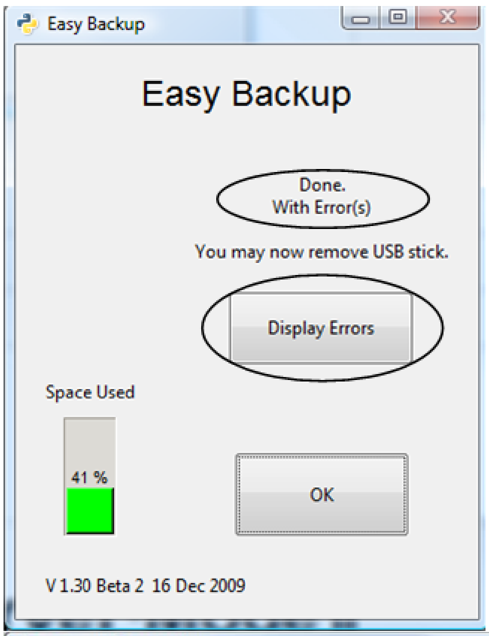

Easy Backup
Easy Backup
Copy Zipfile 2009-11-10.EasyBackup.Test.zip
V 1.20 (RC 3) xx Nov 2009

If on the first screen the button “Do NOT start processing” has not been clicked the backup starts and displays the status screen.



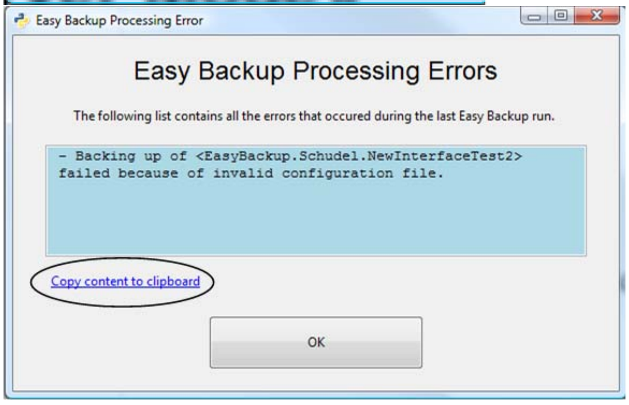
Backup is done without error. The USB stick can be removed.

If an error occurred sometime during the backup the next screen comes up.



An error occurred somewhere during the backup.

Click on the *Display Errors* button to see what went wrong.


Easy Backup Processing Error
Easy Backup Processing Errors
Copy content to clipboard
OK

The following list contains all the errors that occured during the last Easy Backup run.

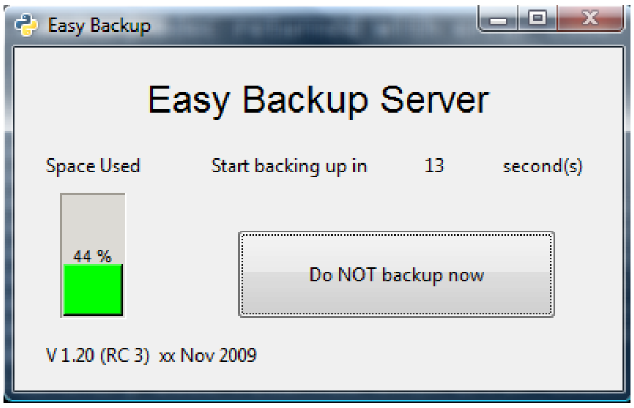
```
- Backing up of <EasyBackup.Schudel.NewInterfaceTest2>
failed because of invalid configuration file.
```

If desired click on the *Copy content to clipboard* link to copy the messages into the clipboard so you do not need to copy it manually.

Or click *OK* to close the window.

7. *EasyBackup* in Server Mode

As mentioned in the general configuration section *EasyBackup* can be put into “Server mode”. When *EasyBackup* runs in server mode it won’t backup the user’s files to the inserted stick. Instead it will copy the *EasyBackup* backup files found on the USB stick to the target location of the server machine as specified in the general configuration.



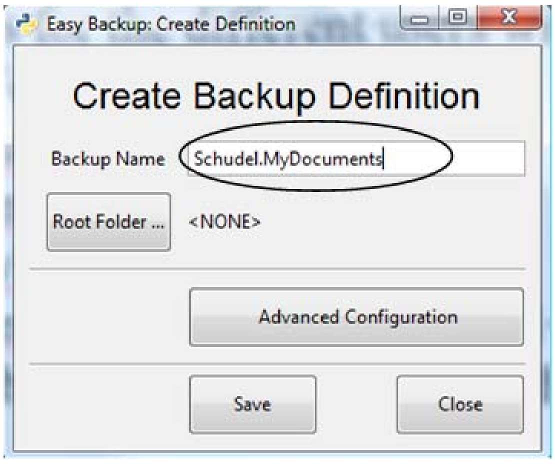
The title in the window reflects when *EasyBackup* is running in sever mode.

Note EasyBackup becomes only visible if it finds the EasyBackup folder (called “basilBackup”) on the inserted stick. Otherwise it silently quits.

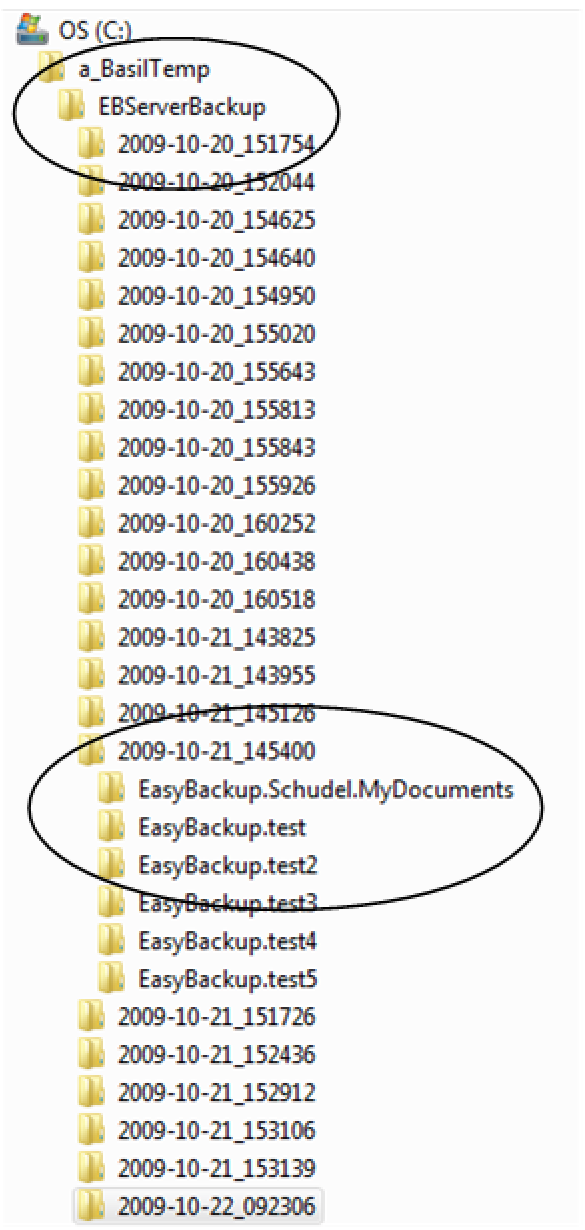
Running *EasyBackup* in server mode can be used in IT departments to centrally keep backups of the various users who use *EasyBackup* to backup their machines. Under the target location as specified in the *EasyBackup* configuration it creates a subfolder named “EBServerBackup”. Whenever *EasyBackup* runs in server mode and copies the backups from the inserted USB stick onto the server it creates a new subfolder in the “EBServerBackup” folder. The name of the new subfolder

has the format “yyyy-mm-dd_HHMMSS” (for example 2009-11-10_140518). In this subfolder *EasyBackup* then copies the backups found on the stick.
This means the server stores the backups in a folder that reflects the date and time when the stick was inserted. In this way the same stick could be copied more than once in a day without overwriting a previously written copy.
EasyBackup keeps a log of the backups it copied onto the server.

It might be helpful if you prefix the various backup definitions for the different users with their user name. For example: “Schudel.MyDocuments”, “Müller.MyDocuments”, “Baumgartner.MyDocuments”, “Holzer.MyDocuments” etc. Then it would be easy to find their backups on the server or the log file.



Backup Name Schudel.MyDocuments
<NONE>

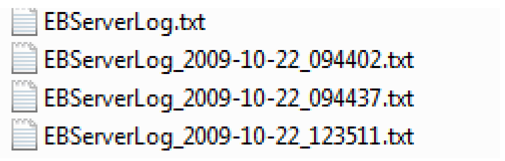


OS (C:)
a_BasilTemp
EBServerBackup
2009-10-20_151754
2009-10-20_152044
2009-10-20_154625
2009-10-20_154640
2009-10-20_154950
2009-10-20_155020
2009-10-20_155643
2009-10-20_155813
2009-10-20_155843
2009-10-20_155926
2009-10-20_160252
2009-10-20_160438
2009-10-20_160518
2009-10-21_143825
2009-10-21_143955
2009-10-21_145126
2009-10-21_145400
EasyBackup.Schudel.MyDocuments
EasyBackup.test
EasyBackup.test2
EasyBackup.test3
EasyBackup.test4
EasyBackup.test5
2009-10-21_151726
2009-10-21_152436
2009-10-21_152912
2009-10-21_153106
2009-10-21_153139
2009-10-22_092306

This picture shows the folder structure on the *EasyBackup* Server.
In the *EasyBackup* configuration the server folder is set to “a_BasilTemp”.
EasyBackup then created the folder “EBServerBackup” underneath. Then there are subfolders with various dates and times when *EasyBackup* copied user backups to the server.
Each of these folders contains data from only **one** user.

If the backup definitions reflect the user name (as suggested further up) it would be easy to find the backups for a particular user.

EasyBackup keeps a server log in the “EBServerBackup” folder where it records what was copied. If the logfile reaches a certain size (currently about 2 Mbytes) it makes a copy of the log with a name that reflects when it was created and then clears out the original log.



EBServerLog.txt
EBServerLog_2009-10-22_094402.txt
EBServerLog_2009-10-22_094437.txt
EBServerLog_2009-10-22_123511.txt

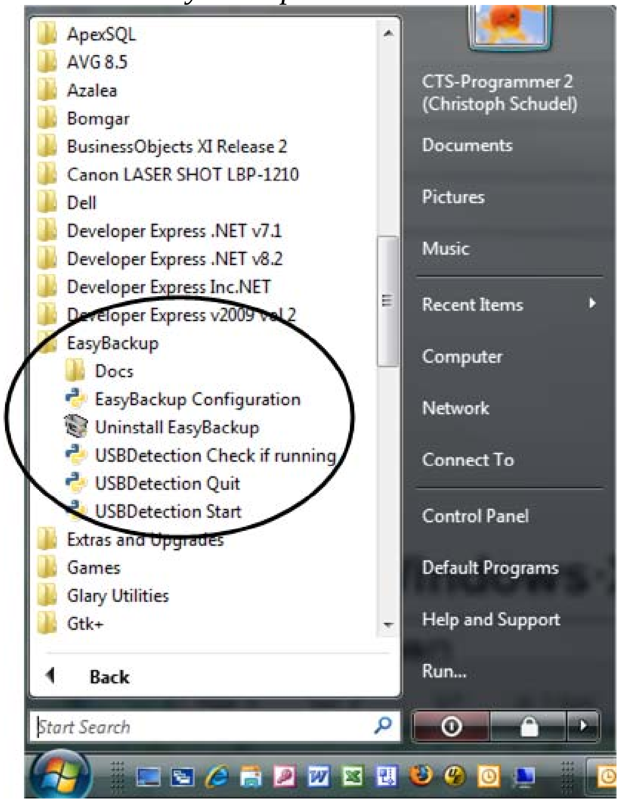
The server log reflects what has been copied:

```
2009-11-06 14:30:28  ***** Backing up USB Stick to EB Server **************
2009-11-06 14:30:28  Copy <EasyBackup.Schudel.MyDocuments> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:29  Copy <EasyBackup.test> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:29  Copy <EasyBackup.test2> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:29  Copy <EasyBackup.test3> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:30  Copy <EasyBackup.test4> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:30  Copy <EasyBackup.test5> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_143028>
2009-11-06 14:30:30  ***** Backing up done **************
2009-11-06 15:14:40  ***** Backing up USB Stick to EB Server **************
2009-11-06 15:14:40  Copy <EasyBackup.Schudel.MyDocuments> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:40  Copy <EasyBackup.test> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:41  Copy <EasyBackup.test2> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:41  Copy <EasyBackup.test3> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:41  Copy <EasyBackup.test4> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:42  Copy <EasyBackup.test5> to <C:\a_BasilTemp\EBServerBackup\2009-11-06_151440>
2009-11-06 15:14:42  ***** Backing up done **************
```

Again, it helps if the various backup definitions reflect the name of the user. It then would be very easy to find a particular backup for a particular user.

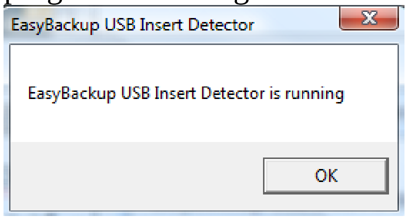
8. *EasyBackup* Windows Support Programs:

Under the *EasyBackup* Start Menu entries there are a number of other shortcuts available:



- **Docs**: This will open the document folder
- **EasyBackup Configuration**: This opens the EasyBackup configuration editor as described further up.
- **Uninstall EasyBackup**: runs the uninstaller (please note it will **only** uninstall EasyBackup. No third party program that was installed with EasyBackup gets uninstalled)

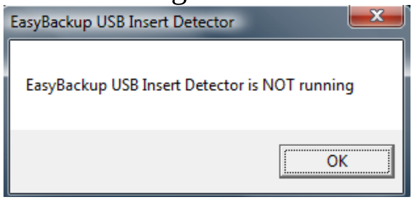
It will either display that the program is running …


EasyBackup USB Insert Detector

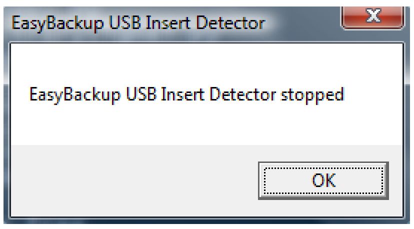
EasyBackup USB Insert Detector is running

- **USBDetection Check if running**: Launch this program to check, if the USBDetection program is running in the background or not.

or not running …


EasyBackup USB Insert Detector

EasyBackup USB Insert Detector is NOT running

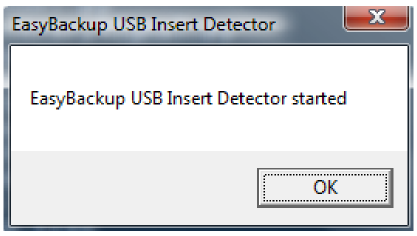

EasyBackup USB Insert Detector

EasyBackup USB Insert Detector stopped

- **USBDetection Quit**: Stops the USBDetection program.

- **USBDetection Start**: Starts the USBDetection program.

Note: If you run this program from the Startup folder it will be launched silently. This means the message box to the right will not be shown.


EasyBackup USB Insert Detector

EasyBackup USB Insert Detector started

9. Log files

The system keeps one log file which under **Windows** is called “EasyBackupSysLog.log” and stored in the user’s home folder. Under **Linux** the standard SYSLOG facility is being used.

10. Installation

Please see document *Installation.pdf*

11. Info for Programmes

11.1. General info

EasyBackup is simple program that allows you to create XML files that the “backup” program “bdec” (Basil Date Exchange Component) uses to do backup. *Bdec* was designed to do much more and the intention was to work in concert with other application. Normally those applications would create the appropriate XML files that bdec than uses. With *EasyBackup* it is possible to use *bdec* without other applications. For more information, particularly if you are a software developer and would like to incorporate *bdec* into your application, study the programmer’s documentation and related documents.

11.2. Localisation

Bdec supports localisation (translation of texts into various languages). In the configuration file “bdecConfig.xml” the language to be used can be set (tag: `<Locale Language="default"/>`)
Note that it should be set to **default**. I then sets the language according to “defaultLocale” as defined by the os or, if set, by the environment variable “LANG”. This because the translation for “Glade” **only** looks at these two system set variables! If the language to use is not “default” but for example “de_CH”) it will only affect text to translate in “normal” code. Text to be translated in “Glade” will not be affected.
If the program doesn’t want to work with the default Locale as set by the os but wants to look at the environment variable “LANG” this variable needs to be set **before** the program starts. It can’t be set by the program itself as the values of environment variables are read very early in the launch process and setting one by the program itself will be too late.
In Linux this variable can be set with: “LANG=de_CH”. Under Windows it is best to set an environment variable "LANG" in the control panel, system, environment variables.

There are four shell scripts (tested under Fedora and Ubuntu) that support generation of the necessary translated file. The scripts are located under the folder “locale” where all the python scripts are. They should be run in this order.
In order to localise you need to have the “intltool” installed. (In Ubuntu: `sudo apt-get install intltool`)
(Note: If you only localise for a new language you only need to run the **last two** scripts. The first two are used if the program changed and there are additional stings that need translation.)

Script Name	**What it does**
createGladeHeader.sh	Extracts the texts from the files bdecUI.glade and basilEasyBackup.glade and creates the files bdecUI.glade.h and basilEasyBackup.h. These header files now contains all the texts that need to be translated. Usage: `bash createGladeHeader.sh`
createTemplate.sh	• Creates the folder “Soucre” under the “locale” folder • Extracts all the texts that need translation from all the files with “.py” extensions from the parent folder and the header file created with the first script. • The template file is stored in “Source/bdec.pot” Usage: `bash createTemplate.sh`
createLocaleSource.sh	Run this script to create a locale file for that language that you want to translate. • Usage: `bash createLocaleSource.sh de_CH` Note: “de_CH” is an example. Use the actual language

Script Name	What it does
	that you want to translate into. This script will create a language file .po with the supplied language name and store it in folder “Source”. In the example above the file will be called de_CH.po • Now you need to open this file and translate all the strings. Usage: `bash createLocaleSource.sh nameOfLanuage`
createCompiledTranslations.sh	This will compile all .po files it finds in the “Source” folder and copies them into “locale/de_CH/LC_MESSAGES. They all have the name “bdec.mo”. (Please note that “de_CH” is just an example. It uses the names from the .po files.) Usage: `bash createCompiledTranslations.sh`

Folder hierarchy:

Location	Comment
somewhereOnTheDisk	all the python sources for bded and the glade header file.
somewhereOnTheDisk/locale/Source	Contains the four shell scripts Contains all the .po files as well as bdec.pot template file
somewhereOnTheDisk/locale/de_CH/LC_MESSAGES	Contains the translated language file for language de_CH. The name of the file is bdec.mo

Please note the “de_CH” is only an example (for German Swiss).