mseedrename(1)

Name

mseedrename - systematically rename miniSEED files using a template

Synopsis

> mseedrename [--dry-run] [-v|--verbose]
>
> [--template=*FILENAME*] [--output-dir=*DIRECTORY*]
>
> [--force-overwrite] [--transfer-mode=*MODE*]
>
> [--include-pattern=*PATTERN*]...
>
> *file* | *directory*...

> mseedrename [-h|--help] [--version] [--sysinfo]

Description

Mseedrename systematically renames the given (miniSEED) files using a user provided *NAME* template. If directories are given at the command line, **mseedrename** searches recursively for files inside those directories applying the template to each file found.

When processing files, **mseedrename** will read the first miniSEED record contained in the source file. From this record information about recording start time, station id, etc. is obtained. This information is used when interpreting the *NAME* template, replacing each variable found by its corresponding value. The new filename then is constructed from the interpreted *NAME* template and the *DIRECTORY* given by the **--output-dir** option (if available). Finally, the source file is copied or moved to its new destination depending on the **--transfer-mode** option.

Obviously, the capability to replace variables in the *NAME* template depends on the information that can be obtained from the source file in the first place. If the source is not a miniSEED file and therefore cannot provide the necessary information replacing most template keywords must fail! So, in addition to using the first record of a miniSEED file as source of information, the program also can obtain at least some (but not all) variable values from the filename of the source file. This fallback mechanism is used when the source is not a miniSEED file but its filename follows the pattern "e*NNNNYYMMDDHHMMSS*.*"". Here, the letter e followed by four digits *NNNN* is used as station id, *YYMMDD* represents the date and *HHMMSS* the time. (*Note:* Almost all files generated by GIPP EDLs follow this convention.)

Options

The program pretty much follows expected Unix command line syntax. Some of the command line options have two variants, one long and an additional short one (for convenience). These are

shown below, separated by commas. However, most options only have a long variant. The '=' for options that take a parameter is required and can not be replaced by a whitespace.

-h, --help

Print a brief summary of all available command line options and exit.

--version

Print the **mseedrename** release information and exit.

--sysinfo

Provide some basic system information and exit.

--dry-run

Perform a trial run with no changes and modifications made whatsoever to the file system while at the same time producing (almost) the same user feedback as a real run. It is most commonly used in combination with the **--verbose** options to see what an **mseedrename** command is about to do before one actually runs it for real.

-v, --verbose

This option increases the amount of information given to the user during the program execution. By default (i.e. without this option) **mseedrename** only reports warnings and errors. (See the diagnostics section below.)

--template=*FILENAME*

The given *FILENAME* template controls the automatic generation of the new filename. Here, every character of *FILENAME* is used for the new filename with the exception of the following variables. They are interpreted and replaced with their corresponding value (taken from the currently processed input file). The following keys are possible:

%%

a literal % character

%f

original filename (including suffix)

%b

basename (i.e. filename without suffix)

%e

extension (filename suffix without the dot)

%Y

year (including century, i.e. 4 digits)

%y

year (without century, i.e. 2 digits)

%m
: month (2 digits)

%d
: day (of month, 2 digits)

%j
: day of year (3 digits)

%H
: hour (2 digits)

%M
: minute (2 digits)

%s
: second (2 digits)

%T
: complete time (equivalent to %H:%M:%S)

%F
: complete date (equivalent to %Y-%m-%d)

%G
: GEOFON style (equivalent to %Y/%S/%j/%f)

%S
: miniSEED station id (up to 5 characters)

%L
: miniSEED location id (up to 2 characters)

%C
: miniSEED channel id (up to 3 characters)

%N
: miniSEED network id (up to 2 characters)

%r
: miniSEED sample rate (rounded to the closest integer number)

Please note that usage of / characters is explicitly allowed in the filename template indicating additional subdirectory levels. The respective directories will automatically be created as needed. Variables containing time information are read from the first miniSEED record in the file and usually represent the start time of the recording in that file.

If no filename template is given at the command line, the filename keyword ("%f") is used as

default. This will collapse any complex directory tree used as input into a single flat output directory.)

You will run into trouble when you try to rename a non-miniSEED file using a keyword only available for miniSEED files (e.g. the network id "%N"). Since a non-miniSEED source file simply cannot provide the network id required to replace the "%N" keyword in the template string the rename operation must fail!

--output-dir=***DIRECTORY***

Use *DIRECTORY* as a starting point for all paths and filenames that were generated from the filename template (see option **--template** described above). The directory must exist and be writable! If this option is not used, the current working directory of the user will be used as starting point instead.

Already existing files will not be overwritten unless the option **--force-overwrite** is used as well.

The rule that resulting filenames are always relative to the output *DIRECTORY* or the users current working directory is meant to protect you from accidentally overwriting system files. Please note however that there is no guarantee! The mechanism could be easily circumvented by e.g. fooling around with symbolic links or using dot-dot components ("..") in the filename template.

--force-overwrite

If this option is used, already existing files will be overwritten without mercy! The default behavior however is **not** to overwrite existing files and to simply skip processing the corresponding input file.

--transfer-mode=***MODE***

Select the file transfer mode. Here *MODE* can be one of the following:

COPY

Copy files and do not modify the source. This is the default mode.

MOVE

Move files, deleting already transferred source files in the process. Only successfully transferred files will be deleted. The source directory tree (even if the remaining directories are empty) is not touched. This mode is mostly intended for situations where you are desperately short on disk space.

Use the *MOVE* mode only if you are very sure that there will be no problems (e.g. by first testing the command using the **--dry-run** option).

--include-pattern=***PATTERN***

Only process files whose filename matches the given *PATTERN*. Files with a name not matching the search *PATTERN* will be ignored. This option is quite useful to speed up recursive searches for input files through large subdirectory trees and can be used more than once in the same

command line.

You can use the two wild card characters (*, ?) when specifying a PATTERN (e.g. *.pri?). Or alternatively, you can also use a predefined filter called GIPP that can be used exclude all files not following the usual GIPP naming convention for miniSEED files recorded by Earth Data loggers (e.g. message logging or status files).

The search *PATTERN* is only applied to the filename part and not to the full pathname of a file.

Environment

The following environment variables can optionally be used to influence the behavior of the various GIPPtool utilities during startup.

GIPPTOOLS_HOME

This environment variable is used to find the location of the GIPPtools installation directory. In particular, the Java class files that make up the GIPPtools are expected to be in the java subdirectory of **GIPPTOOLS_HOME**.

GIPPTOOLS_JAVA

The utilities of the GIPPtools are written in the programming language Java and consequently need a Java Runtime Environment (JRE) to execute. Use this variable to specify the location of the JRE which should be used.

GIPPTOOLS_OPTS

You can use this environment variable for additional fine-tuning of the Java runtime environment. This is typically used to set the Java heap size available to GIPPtool programs.

It is usually not necessary to define any of those variables as suitable values should be selected automatically. However, if the automatic detection build into the start script fails or you need to choose between different GIPPtool or Java runtime releases installed on your computer, these environment variables might become quite helpful to troubleshoot the situation.

Diagnostics

Mseedrename occasional will produce user feedback. In general, user messages are classified as *INFO*, *WARNING* or *ERROR*. The *INFO* messages are only displayed when the **--verbose** command line option is used. They usually report about the progress of the program run, give statistical information or write a final summary.

More important are *WARNING* messages. In general, they warn about (possible) problems that may influence the outcome. Although the program will continue with execution, you certainly should check the results carefully. You might not have gotten what you (thought you) asked for. Finally, *ERROR* messages inform about problems that can not be resolved automatically. Program execution usually stops and the user must fix the problem first.

A good method to see what will happen is to use the **--dry-run** and the **--verbose** command line option at the same time. If the user feedback indicates that **mseedrename** works as expected it can be started again, this time without the **--dry-run** option.

Exit codes

Use the following program exit codes when calling **mseedrename** from scripts or other programs to see if **mseedrename** finished successfully. Any non-zero code indicates an ERROR.

0

Success.

64

Command line syntax or usage error.

66

An input file did not exist or was not readable.

74

I/O error.

99

Other, unspecified errors.

Examples

1. You have a hundreds of miniSEED files in a single unsorted directory and would like to sort them by date into separate subdirectories for easier processing:

   ```
   mseedrename --verbose --template=%F/%f -output-dir=./sorted ./unsorted
   ```

2. Due to some mishap in the field, all filenames are wrong and you need to rename them. Unfortunately, you are also running low on disk space. Try e.g. the following command and carefully study the diagnostic messages

   ```
   mseedrename --dry-run --verbose --transfer-mode=MOVE --template=%S%y%m%d%H%M%s.%C
   --output-dir=./sorted ./unsorted
   ```

   After checking for any warnings or severe errors you can re-run the command, this time without the **--dry-run** option.

Files

$GIPPTOOLS_HOME/bin/mseedrename

The **mseedrename** "program". Usually just a symbolic link pointing to the standard GIPPtools start script.

$GIPPTOOLS_HOME/bin/gipptools

The GIPPtools start script. Almost all utilities of the GIPPtools package are started from this shell script.

See also

gipptools(1), **cube2ascii**(1), **cube2mseed**(1), **cube2segy**(1), **cubeevent**(1), **cubeinfo**(1), **mseed2ascii**(1), **mseed2mseed**(1), **mseed2pdas**(1), **mseed2segy**(1), **mseedcut**(1), **mseedinfo**(1), **mseedrecover**(1)

Bugs and caveats

- The **mseedrename** utility assumes that each miniSEED input file contains only one continuous time series respectively! It will only look at the first miniSEED record of each file when replacing variables in the template string. Unfortunately, this approach fails (without warning) when **mseedrename** is confronted with a multiplexed miniSEED file.
- Java 1.5 does not know about symbolic links and treats them as plain files. Consequently **mseedrename** will move/copy the content of the file to which the encountered symbolic link points as if it were a normal file. This might or might not be what you expected...
- **Mseedrename** attempts to preserve the "last modified" time associated with the renamed file. However, not all file systems are equal and the attempt might not be entirely successful.