1 mfc

mfc — a tool for simple creation of makefiles

Synopsis

mfc [*PARAMETERS*]

DESCRIPTION

mfc is a small tool for the simple creation of simple makefiles. The mainadvantage compared to other tools is, that you don't have to mention, which files are existing, for your code files are found and identified by their endings automaticly.

mfc is designed to use intelligent defaults. However, not everything you might want to do, is intelligent for everyone and different people might want to use different compilers. Therefore it is of course possible to set severall things via commandline and config-files. These will be parsed in the following order:

1. /etc/mfc/mfc.conf
2. ~/.config/mfc/mfc.conf
3. ./mfc.conf
4. commandline

Later options will override older ones, if they are obviously conflicting (e.g. setting a different compiler will overide the former setting but setting a new compiler option will keep the old ones too).

If not noted otherwise all ways are sharing most of the commands, except for the short options: You can only use them as commandlineparameters, not in the config-files.

OPTIONS

-s, --source-dir
Sets the root-directory, that contains all your codefiles (those may be in subdirectories though). The defualt is the current directory.

-b, --build-dir
Set the build-directory, where all object-files are stored. The defualt is the current directory.

-t, --target
Set the target-file. This has to be done anywhere, because there is no default.

-o, --output
Set the output of mfc. The default is “makefile”. This must not be utside the current directory.

-v, --verbose
Set the level of verbosity. The default is 0, which means nearly no output

-d, --debug
Set the debug level.

-c, --compiler
set the compiler.

-l, --link
Set what the target libraries should be linked against.

-L, --link-dir
Add a directory to the programms list, where it should look for libraries.

-I, --include-dir
Add a directory to the compilers include-list.

-w, --ignore
Don't create an objectfile from the given sourcefile. This is for example usefull if you wrote a class, that uses templates and therefore include the implementation parts in the header.

import
This option only works in config-files. It imports another config-file.

-f, --config-file
interpret the given files as config-file. This option doesn't work in config-files.

-u, --use
Use the given config-set.

-O, --compiler-ops
add a compile-option.