NAME

mup - music publisher

SYNOPSIS

mup [**-c***N*] [-C] [**-d***N*] [**-D** *MACRO[=macro-def]*] [**-e** *errfile*] [-E] [**-f** *outfile*] [**-F**] [**-m** *midifile*] [**-M**] [**-o** *pagelist*] [**-p***N*] [**-v**] [**-x** *N*[,*M*] [*file...*]

DESCRIPTION

Mup is a program for producing printed music. There is an optional companion program called Mupmate that provides a more graphical user interface on top of Mup, but this manual page describes the command line interface. The Mup User's Guide should be consulted for details of the format of the input file. Options include:

-c *N* Combine consecutive measures of all rests or spaces into multirests (multiple measures of rest printed as a single measure, with the number of measures of rest printed above the staff). Any time there are *N* or more measures in a row that consist entirely of rests or spaces, they will be replaced by a multirest. The combining of measures stops when there is a visible staff that contains notes, lyrics, or other musical symbols, when there are parameter changes on a visible staff or in score context, or when there is a bar line other than an ordinary or invisible bar. This option is most likely to be useful when printing a subset of staffs, where the particular staff(s) you are printing have long periods of rests. (See the -s option.)

-C This option is only used in connection with the -E option. It specifies that comments are to be passed through rather than deleted.

-d *N* Print debugging information. *N* is a bitmap.

1	parse phase information
2	high level parse phase tracing
4	low level parse phase tracing
8	reserved
16	high level placement phase tracing
32	low level placement phase tracing
64	reserved
128	contents of the main internal list
256	high level print phase tracing
512	low level print phase tracing

N can be specified in decimal, octal (by using a leading zero), or hex (by using a leading 0x). This information is intended for debugging of **Mup** itself and thus is not likely to be of use to the average user.

-D *MACRO[=macro-def]*

Define the macro *MACRO*. The macro name must consist of upper case letters, digits, and underscores, beginning with an upper case letter. The *macro_def* is optional, and gives the text of the macro. If it contains any white space or other special characters, it must be quoted (if quoting is supported by your operating system or shell).

-e *errfile*

Place error messages into *errfile* instead of writing them to the standard error output stream.

-E Rather than produce PostScript or MIDI output, just expand macros and includes, and write the result to the standard output stream. Comments in the input are deleted, unless the -C option is also specified.

-f *outfile*

Place the output into *outfile* instead of writing it to the standard output.

-F This is like the **-f** option, except the name of the output file is derived from the name of the Mup input file. If the name of the Mup input file ends with a ".mup" suffix, the generated PostScript output file will end with a ".ps" suffix instead. If the name of the Mup input file ends with a ".MUP" suffix, the PostScript file will end with a ".PS" suffix. Otherwise, a ".ps" suffix will be appended to the end of the Mup input file name. If multiple input files are listed, the last is used. If none are specified (input is read from standard input), the name "stdin.ps" will be used for the output file.

-m *midifile*

Instead of generating PostScript output, generate standard MIDI (Musical Instrument Digital Interface) output, and put it in *midifile*. This option also causes the macro "MIDI" to become defined.

-M This is like the **-m** option, except the name of the MIDI file is derived from the name of the Mup input file. If the name of the Mup input file ends with a ".mup" suffix, the generated MIDI file will end with a ".mid" suffix instead. If the name of the Mup input file ends with a ".MUP" suffix, the MIDI file will end with a ".MID" suffix. Otherwise, a ".mid" suffix will be appended to the end of the Mup input file name. If multiple input files are listed, the last is used. If none are specified (input is read from standard input), the name "stdin.mid" will be used for the MIDI file.

-o *pagelist*

Print only the pages given in *pagelist*. The *pagelist* can be a comma-separated list of numbers or ranges, where a range is two numbers separated by a dash. For example, -o1,7-9,12-14 would print pages 1, 7, 8, 9, 12, 13, and 14. Pages will be printed in the order given. They need not be in order, and a page number may be included more than once. Alternately, the *pagelist* can be the special keyword "odd" or "even" which will cause all odd or even numbered pages to be printed. This may be useful if you have a printer that only makes single-sided copies, but you wish to print Mup output double-sided. You could print odd-numbered pages, then turn the paper over and feed the pages through again for the even-numbered pages.

-p*N* Start numbering pages at *N* instead of at 1. If **-o** and **-p** are used together, the page numbers given in the **-o***pagelist* must be the printed page numbers. For example, if you use -p10 and want to print just the second page, you would need to specify -o11.

-r Print a copy of the Mup shareware registration form to the standard output.

-s*stafflist*

Only print the staffs that are included in *stafflist*. This can be a comma-separated list of staff numbers or ranges, such as "1,5" or "1-3,7-8" To further restrict to a single voice on a staff, add **v***N* where *N* is the voice number (1, 2, or 3), after the staff, as in "2v1,5v2" You can't specify a list or range for voices; if you only want to make two out of three voices visible, you have to specify them separately, like "1v2,1v3". No spaces are allowed in the list.

-v Print the Mup version number and exit. This manual page is for version 5.3.

-x*M,N* Extract measures *M* through *N* of the song. This allows you to print or play a part of a song. The comma and second value are optional; if not specified, the default is to go to the end of the piece. Positive values specify the number of measures from the beginning of the piece, while negative values are relative to the end, with -1 referring to the last measure of the song. So -x1,-1 means the entire song, if the song doesn't have a pickup measure. If the song has a pickup measure, that is specified by 0. So for a song with a pickup, -x0,-1 would mean the entire song, and -x0,0 would mean just the pickup measure. As other examples, -x-1,-1 means just the final measure of the song, -x2 means starting after the first full measure, -x3,4 means only measures 3 and 4, and -x6,6 means just measure 6. The starting measure is not allowed to be inside an ending. A common use for this option might be to generate a MIDI file for just a few measures. For example, if you were trying to tweak tempo values for a ritard in the last 2 measures of a song, you could use -x-2 to listen to just those measures.

The options, if any, can be followed by one or more *files* in Mup format. If no *files* are specified, standard input is read. If several *files* are listed, they are effectively concatenated together and treated as one big file. Since there are some things (such as header and footer) that are only allowed to occur once, if you have several independent pieces, mup should be called on each individually rather than trying to print them all with one command. If a specified file does not exist, and its name does not already end with .mup or .MUP, then Mup will append .mup to the specified name and attempt to open that.

On most systems, the environment variable MUPPATH can be set to a list of paths in which to look for 'include' files. The components are separated by a colon on Unix or Linux systems, and by a semicolon on systems with DOS-like file naming conventions.

For more debugging, in addition to the -d option, if the environment variable MUP_BB is set to "bcfghnsu" or any subset of those letters, the generated output will include "bounding boxes" for the things Mup internally calls bars (b), chords (c), feeds (f), grpsyls (g), header/footer and top/bottom (h), notes (n), staffs (s), and stuff (u). While this is intended for use in debugging Mup itself, it may also help you understand why Mup places things the way it does, since in general, Mup only allows bounding boxes to overlap according to specific rules. If viewed with a color PostScript viewer (not mupdisp), these boxes will be in color.

SEE ALSO

gs(1), mkmupfnt(1), mupdisp(1), mupmate(1), mupprnt(1).
Mup — Music Publisher User's Guide