NAME

ExtUtils::MM_VMS - methods to override UN*X behaviour in ExtUtils::MakeMaker

SYNOPSIS

```
Do not use this directly.
Instead, use ExtUtils::MM and it will figure out which MM_*
class to use for you.
```

DESCRIPTION

See ExtUtils::MM_Unix for a documentation of the methods provided there. This package overrides the implementation of these methods, not the semantics.

Methods always loaded

wraplist

Converts a list into a string wrapped at approximately 80 columns.

Methods

Those methods which override default MM_Unix methods are marked "(override)", while methods unique to MM_VMS are marked "(specific)". For overridden methods, documentation is limited to an explanation of why this method overrides the MM_Unix method; see the ExtUtils::MM_Unix documentation for more details.

guess_name (override)

Try to determine name of extension being built. We begin with the name of the current directory. Since VMS filenames are case-insensitive, however, we look for a *.pm* file whose name matches that of the current directory (presumably the 'main' *.pm* file for this extension), and try to find a `package` statement from which to obtain the Mixed::Case package name.

find_perl (override)

Use VMS file specification syntax and CLI commands to find and invoke Perl images.

_fixin_replace_shebang (override)

Helper routine for MM->*fixin()*, overridden because there's no such thing as an actual shebang line that will be interpreted by the shell, so we just prepend `$Config{`startperl`}` and preserve the shebang line argument for any switches it may contain.

maybe_command (override)

Follows VMS naming conventions for executable files. If the name passed in doesn't exactly match an executable file, appends *.Exe* (or equivalent) to check for executable image, and *.Com* to check for DCL procedure. If this fails, checks directories in DCL$PATH and finally *SysSystem:* for an executable file having the name specified, with or without the *.Exe*-equivalent suffix.

pasthru (override)

VMS has $(MMSQUALIFIERS) which is a listing of all the original command line options. This is used in every invocation of make in the VMS Makefile so PASTHRU should not be necessary. Using PASTHRU tends to blow commands past the 256 character limit.

pm_to_blib (override)

VMS wants a dot in every file so we can't have one called 'pm_to_blib', it becomes 'pm_to_blib.' and MMS/K isn't smart enough to know that when you have a target called 'pm_to_blib' it should look for 'pm_to_blib.'.

So in VMS its pm_to_blib.ts.

perl_script (override)

If name passed in doesn't specify a readable file, appends *.com* or *.pl* and tries again, since it's customary to have file types on all files under VMS.

replace_manpage_separator
: Use as separator a character which is legal in a VMS-syntax file name.

init_DEST
: (override) Because of the difficulty concatenating VMS filepaths we must pre-expand the DEST* variables.

init_DIRFILESEP
: No separator between a directory path and a filename on VMS.

init_main (override)
init_tools (override)
: Provide VMS-specific forms of various utility commands.

  Sets DEV_NULL to nothing because I don't know how to do it on VMS.

  Changes EQUALIZE_TIMESTAMP to set revision date of target file to one second later than source file, since MMK interprets precisely equal revision dates for a source and target file as a sign that the target needs to be updated.

init_platform (override)
: Add PERL_VMS, MM_VMS_REVISION and MM_VMS_VERSION.

  MM_VMS_REVISION is for backwards compatibility before MM_VMS had a `$VERSION`.

platform_constants
init_VERSION (override)
: Override the *DEFINE_VERSION macros with VMS semantics. Translate the MAKEMAKER filepath to VMS style.

constants (override)
: Fixes up numerous file and directory macros to insure VMS syntax regardless of input syntax. Also makes lists of files comma-separated.

special_targets
: Clear the default .SUFFIXES and put in our own list.

cflags (override)
: Bypass shell script and produce qualifiers for CC directly (but warn user if a shell script for this extension exists). Fold multiple /Defines into one, since some C compilers pay attention to only one instance of this qualifier on the command line.

const_cccmd (override)
: Adds directives to point C preprocessor to the right place when handling #include <sys/foo.h> directives. Also constructs CC command line a bit differently than MM_Unix method.

tools_other (override)
: Throw in some dubious extra macros for Makefile args.

  Also keep around the old $(SAY) macro in case somebody's using it.

init_dist (override)
: VMSish defaults for some values.

```
macro description default

ZIPFLAGS flags to pass to ZIP -Vu

COMPRESS compression command to gzip
use for tarfiles
SUFFIX suffix to put on -gz
compressed files
```

```
SHAR shar command to use vms_share

DIST_DEFAULT default target to use to tardist
create a distribution

DISTVNAME Use VERSION_SYM instead of $(DISTNAME)-$(VERSION_SYM)
VERSION for the name
```

c_o (override)
: Use VMS syntax on command line. In particular, $(DEFINE) and $(PERL_INC) have been pulled into $(CCCMD). Also use MM[SK] macros.

xs_c (override)
: Use MM[SK] macros.

xs_o (override)
: Use MM[SK] macros, and VMS command line for C compiler.

dlsyms (override)
: Create VMS linker options files specifying universal symbols for this extension's shareable image, and listing other shareable images or libraries to which it should be linked.

dynamic_lib (override)
: Use VMS Link command.

static_lib (override)
: Use VMS commands to manipulate object library.

extra_clean_files
: Clean up some OS specific files. Plus the temp file used to shorten a lot of commands. And the name mangler database.

zipfile_target
tarfile_target
shdist_target
: Syntax for invoking shar, tar and zip differs from that for Unix.

install (override)
: Work around DCL's 255 character limit several times,and use VMS-style command line quoting in a few cases.

perldepend (override)
: Use VMS-style syntax for files; it's cheaper to just do it directly here than to have the MM_Unix method call `catfile` repeatedly. Also, if we have to rebuild Config.pm, use MM[SK] to do it.

makeaperl (override)
: Undertake to build a new set of Perl images using VMS commands. Since VMS does dynamic loading, it's not necessary to statically link each extension into the Perl image, so this isn't the normal build path. Consequently, it hasn't really been tested, and may well be incomplete.

maketext_filter (override)
: Insure that colons marking targets are preceded by space, in order to distinguish the target delimiter from a colon appearing as part of a filespec.

prefixify (override)
: prefixifying on VMS is simple. Each should simply be:

```
perl_root:[some.dir]
```

: which can just be converted to:

```
volume:[your.prefix.some.dir]
```

otherwise you get the default layout.

In effect, your search prefix is ignored and `$Config{vms_prefix}` is used instead.

cd
oneliner
echo
perl trips up on "<foo>" thinking it's an input redirect. So we use the native Write command instead. Besides, its faster.

quote_literal
escape_dollarsigns
Quote, don't escape.

escape_all_dollarsigns
Quote, don't escape.

escape_newlines
max_exec_len
256 characters.

init_linker
catdir (override)
catfile (override)
Eliminate the macros in the output to the MMS/MMK file.

(File::Spec::VMS used to do this for us, but it's being removed)

eliminate_macros
Expands MM[KS]/Make macros in a text string, using the contents of identically named elements of `%$self`, and returns the result as a file specification in Unix syntax.

NOTE: This is the canonical version of the method. The version in File::Spec::VMS is deprecated.

fixpath

```
my $path = $mm->fixpath($path);
my $path = $mm->fixpath($path, $is_dir);
```

Catchall routine to clean up problem MM[SK]/Make macros. Expands macros in any directory specification, in order to avoid juxtaposing two VMS-syntax directories when MM[SK] is run. Also expands expressions which are all macro, so that we can tell how long the expansion is, and avoid overrunning DCL's command buffer when MM[KS] is running.

fixpath() checks to see whether the result matches the name of a directory in the current default directory and returns a directory or file specification accordingly. `$is_dir` can be set to true to force *fixpath()* to consider the path to be a directory or false to force it to be a file.

NOTE: This is the canonical version of the method. The version in File::Spec::VMS is deprecated.

os_flavor
VMS is VMS.

AUTHOR

Original author Charles Bailey *bailey@newman.upenn.edu*

Maintained by Michael G Schwern *schwern@pobox.com*

See ExtUtils::MakeMaker for patching and contact information.