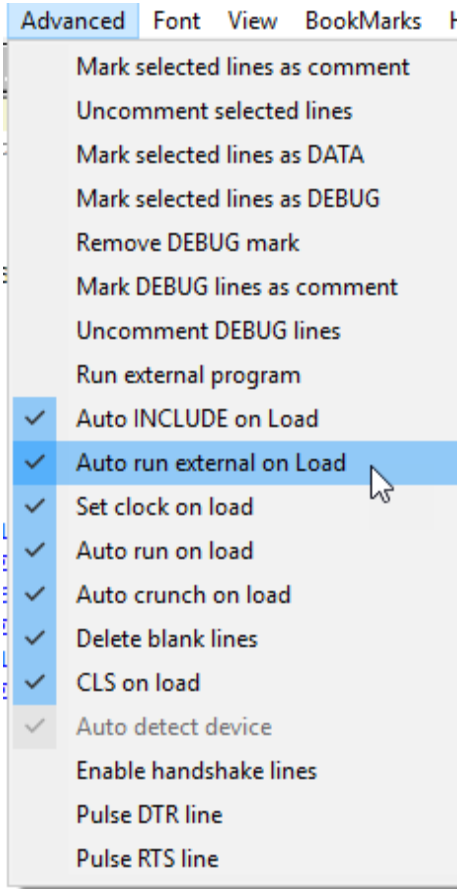
Additional Directives for MMEdit

Version 1.03 - 16 Apr 2018

MMReplace.exe is a utility written in PureBasic which supports the addition of several new Directives which can be used while developing code in MMEdit. It is integrated into MMEdit by calling it from within MMEdit menu options which allow running an external program:

a) ***Advanced/Run External Program*** – called before code is loaded using the Load and Run option and is able to manipulate the code before it is loaded by MMEdit.
The Advanced/Auto Run External Program on Load option should be selected so the program is called before code is loaded by MMEdit.



b) ***File/Run in DOS*** – In the special case of the DOS_v4.5 and Pi-cromite_V5.4.16.dta syntax files being selected in MMedit will run the program here when Load and Run is selected.

In both the cases above the selected program is called with the location of file temp.bas as a parameter. temp.bas is a copy of the current code in MMEdit and the selected program can then manipulate the contents of temp.bas before control is handed back to MMEdit which will then load

the code. The contents of temp.bas is loaded by MMEdit and temp.bas is then deleted, the current code in the MMEdit editor is not changed when Load and Run is called from the icon on the menu ribbon.

Note***: Running the menu option Advanced/Autorun External Program will place the contents of the modified temp.bas into the MMEdit window. You can use the menu option Edit/Undo to restore the original code. This does not happen when the Load and Run icon on the toolbar is used.***

The location of the programs called by the MMEdit menus is initially selected by holding the ***Shift*** key while selecting the menu options. A file selection window opens and the program to be used can be selected. This is then stored in the configuration and will remain for future use.

MMReplace.exe is called from the Advanced/Run External Program menu.

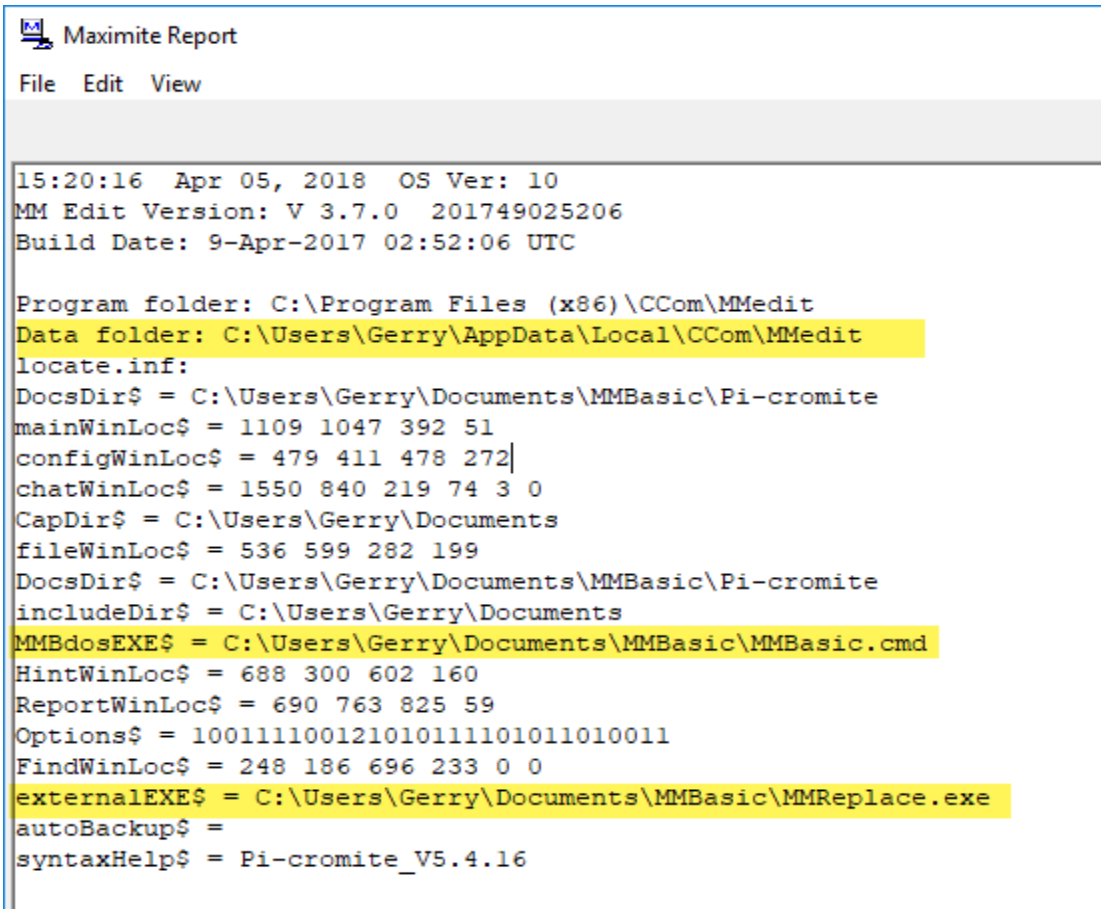
MMBasic.CMD is called from the File/Run in DOS menu.

The files can be located anywhere. The MMBasic.cmd file contains one line as below:

@start C:\Users\Gerry\Documents\MMBasic\MMREPLACE.EXE %1 "C:\Program Files (x86)\MMBASIC\mmbasic.exe"

The paths to MMReplace.exe and MMBasic.exe should be adjusted to match their particular installation locations.

Menu option ***Help/Configuration Report*** can be used to verify the programs assigned.



The Data folder: is where the various syntax files are located. This is where the Pi-cromite_V5.4.17.dta file should be placed if using MMedit for the Pi-cromite.

The MMReplace utility implements a number of additional directives that can be applied to code developed using the MMEdit code editing utility. Each directive is added into the code and all begin with a # and must be at the start of the line.

Replacing variable names at load time

#REPLACE *longname shortname*

At load time all occurrences of *longname* are replaced with *shortname*. This can be used to save space and/or increase speed but still allow the use of descriptive variable names in the code. e.g.

#REPLACE millivoltstep1 mvst1
#REPLACE millivoltstep2 mvst2
#REPLACE millivoltsperdivision mvpd
#REPLACE indexcalibrationx1probe ic1p

Managing Library Code Micromite & MicromitePlus

#LIBRARYLOAD #LIBRARYSTART #LIBRARYEND

These directives were added to simplify the management of a project by allowing the library code to be kept with the main code. The #LIBRARYSTART and #LIBRARYEND directives are used to mark the beginning and end of the code to be kept in the library code.

When the #LIBRARYLOAD directive is present only the library code is placed in temp.bas and loaded. If the directive is commented then the library is not loaded with the normal code but can reside in the one source file. e.g.

```
'---- Library Stuff ----------------------------
'Uncomment the #LIBRARYLOAD directive below to load only the library code.
'which is bounded by #LIBRARYSTART #LIBRARYEND directives
'#LIBRARYLOAD   'Must start in column 1. Uncomment to enable.
'---some handy library commands ---
'LIBRARY SAVE
'LIBRARY LIST
'LIBRARY DELETE
```

Placing code that is stable into the library during development has the advantage of reducing the amount of code reloaded each time during the development cycle. This can be quite a saving if the program is large. The code transferred to the library is still visible in the MMEdit code window if you need to inspect it.

Managing Library Code Colour Maximite (CMM)

Library code can be kept within the main code so it can be easily referenced . The library is bounded by **#LIBRARYSTART & #LIBRARYEND** directives and this will prevent it from loading with the normal code.

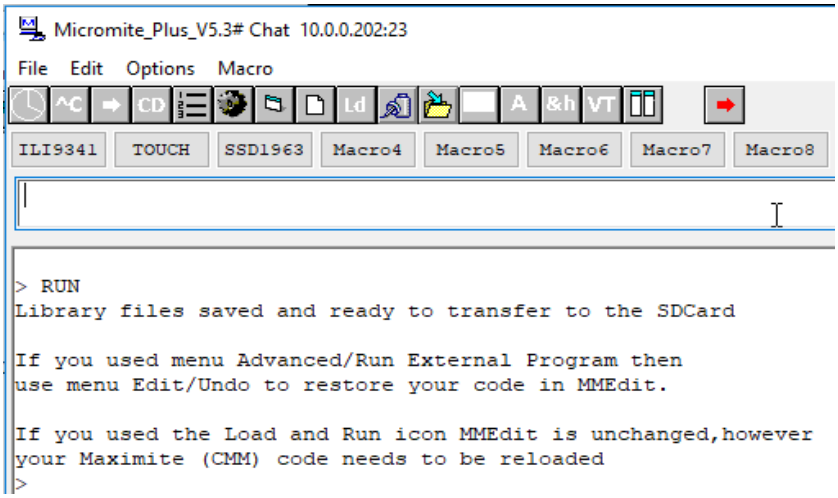
The **#LIBRARYLOAD** directive is expected to have a filename as a parameter, it being the name of the library. The existence of the filename parameter signifies that it is a CMM library and that a .lib file with that filename should be created in the same directory as the current .bas file. This would then be transferred to the CMM SD-Card using MMEdit. The .lib extension is added if not supplied. e.g

#LIBRARYLOAD *maths.lib*

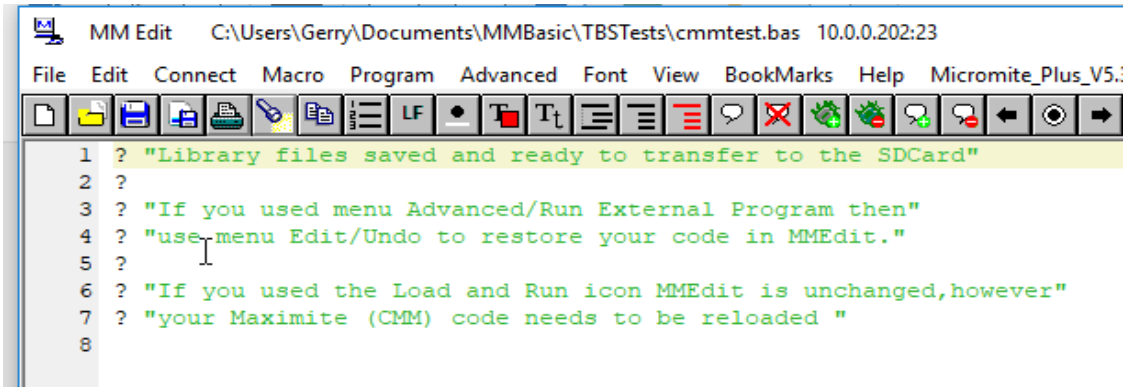
Each #LIBRARYLOAD *filename* **directive** will only generate a .lib file for one #LIBRARYSTART / #LIBRARYEND, the first that is found after the directive. Multiple libraries may be defined, each having its own **#LIBRARYLOAD** directive**.** Commenting the #LIBRARYLOAD directive will stop the library file being generated. E.g. **'#LIBRARYLOAD**

The library file(s) can be generated in two ways once the #LOADLIBRARY directive is enabled.

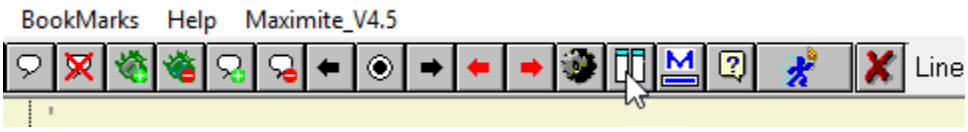
- Using the **Load and Run** icon on the menu ribbon.
  This will generate the library files and will also load a small program into the CMM that prints this message saying that the CMM code needs to be reloaded.



- Using the **Advance/Run External Program**
  This will generate the libraries ready to transfer to the SD-CARD and leave any code in the CMM unchanged, however the MMEdit edit window will contain the contents shown below. You need to use menu Edit/Undo to restore the previous code.



The library files need to be transferred from the PC to the SD-CARD using MMEdit before they can be used. This item on the menu ribbon.



#ENDCODE
This directive also helps support a single source file for the project. This directive is placed at the end of the valid MMBasic code and marks the end of code to be loaded. This allows other relevant code/notes to be stored after the directive but in the same source file.
e.g. associated ESPBasic code. Arduino libraries to reference during development, other notes etc. these do not affect the load time.

/*

These C style comments can be used in the code. There are made into comments at load time. The MMEdit feature to comment and uncomment selected blocks is just as effective. Originally these where not going to load as comments but this interferes with MMEdit's ability to map the program lines to the loaded code when it is responsible for removing comments.

*/

MMEdit and Pi-cromite

The following directives allow MMEdit to be used as the editor for code on the Pi-cromite running on a remote connected Pi.

#PI_CLIPBOARD

#PI_AUTOSAVE ***filename.bas***

#PI_AUTORUN

Code is developed in MMEdit with its various tools available (syntax highlighting, variable report etc) and the resultant code is also automatically save in a file on the Pi-cromite.

Code in the clipboard can be pasted into the Pi-cromite when using PUTTY as the remote terminal by using ***Shift+INS***

When a Pi-cromite specific syntax file is selected MMedit will not attempt to load the code when Load and Run is selected. Instead it will try to run the program under menu ***File/Run in DOS*** with the selected code in temp.bas. This would normally be MMBasic.exe but we have replaced it with MMBasic.CMD so that MMReplace.exe is called first. The #PI_CLIPBOARD directive causes code to be placed in the clipboard and not passed to MMBasic.exe. If the #PI_CLIPBOARD directive does not exist then the code is passed to MMBASIC.EXE and the code run under the DOS version of MMBasic.

The #AUTOSAVE filename.bas directive adds an AUTOSAVE command to the end of the clipboard so the pasted code is automatically save to the nominated file on the Pi.

The #PI_AUTORUN directive adds a RUN command to the end of the clipboard contents so the pasted code is run automatically.

An AUTOLOAD command is pre-appended to the clipboard so the command is automatically applied when pasting code into the Pi-cromite.

MMEdit and ESPBasic

The following directives allow MMEdit to be used as the editor for code ESPBasic running on an ESP8266 wireless device.

#ESP_CLIPBOARD

#ESP_KEEPCOMMENTS

Code is developed in MMEdit with its various tools available (syntax highlighting, variable report etc.) and the resultant code is pasted into the ESPBasic edit window.

When the ESPBasic syntax file is selected and the #ESP_CLIPBOARD directive is present code is placed in the clipboard at Load and Run. The clipboard can be pasted into the ESPBasic edit window and saved.

By default all comments are stripped out when populating the clipboard. The #ESP_KEEPCOMMENTS directive causes any comments beyond that point to be kept in the code.

If the #ESP_KEEPCOMMENTS directive if before the #ESP_CLIPBOARD directive the comments are kept for the entire file.

Loading Arrays with simpleArrayFuncGen

The simpleArrayFuncGen.exe written by @Nathan from The Backshed Forum (TBS) allows the loading of static arrays into flash to be stored as either CFunctions or CSubs. The details and exe file are available in the following forum post.

http://www.thebackshed.com/forum/forum_posts.asp?TID=10253&PN=7

simpleArrayFuncGen

The following directives allow the MMEdit code window to include the definitions for the simpleArrayFuncGen config files and the array data and for the CFunction/CSub to be generated at load time by calling simpleArrayGenerator at that time.

#ARRAYSTART [string|integer|float] [cfunction|csub] *functionorsubname*

Indicates the start of the array definition and the parameters detail, the type of array, where a cfunction of a csub should be reduced and the *name* of the function or sub. The *name* is used as the base *name* for the *name*.cfg and *name*.bas file names passed to simpleArrayFuncGen.

#ARRAYEND

Indicates the end of the array definition.

#ARRAYINCLUDE *functionorsubname*

Place in source where the generated cfunction/csub is to be included during Load and Run

Placing between #LIBRARYSTART and #LIBRARYEND directives ensures it only included when loading the library code.

#ARRAYGENERATE

This directive causes the array cfunction/csub to be generated during Load and Run. If not present the simpleArrayFuncGen .exe is not called so only any previously generated array cfunction/csub functions would be loaded. If the code is to be included in the library then using both the #LOADLIBRARY and #ARRAYGENERATE would achieve this. They could then be both commented during development of the main code.

The array data is entered between the #ARRAYSTART and #ARRAYEND directives. The format of this is determined by simpleArrayFuncGen as it is passed straight through, but essentially each line containing data begins with a: followed by a space.

‘Example of string array
#ARRAYSTART
: this is string1
: this is string2
#ARRAYEND
‘Example of int array
#ARRAYSTART
: 1 2 3 4 5
: 10 20 30 40 50
#ARRAYEND

Installation

The installation consists of a number of files below provided in a zip file mmreplace.zip

./
MMReplace.exe - the main executable
MMBasic.CMD - command file that calls MMReplace.exe but passes an extra parameter which is the location of MMBASIC.EXE the DOS version of MMBasic.
MMReplace.pdf – This help file.
DebugXXX.txt - empty file to be rename to ***debug.txt*** to pause MMReplace.exe closure so console output can be viewed before it closes if required.

./syntaxfiles
Pi-cromite_V5.4.17.dta – the Pi-cromite syntax file for MMEdit.
ESPBasic.dta – the ESPBasic syntax file for MMEdit.
Micromite_Plus_V5.3#.dta – the standard MMEdit syntax file but with the directives added so they are highlighted in the code.

./source
MMReplace.pb - The PureBasic source file for MMReplace.exe

./examples
This sub directory contains various examples of .bas files to be opened in MMEdit

- Place the files in the root directory in any suitable location, in the directory above where you store your MMBasic programs would be suitable.

- Start MMEdit and assign the ***MMReplace.exe*** command to menu ***Advanced/Run External Program*** by holding the Shift when selecting the menu item.

  Note***: The above step will run MMReplace and place the modified code back in the MMEdit window. You can use the Edit/Undo to restore the original code. This only happens here, in normal operation the modified code is loaded and NOT placed back in the MMEdit window.***

- Move the files in the ./syntaxfiles folder to the MMEdit data folder. The Menu option ***Help/Configuration Report*** will generate a report to show this location and verify the assigned programs

- Ensure option ***Advanced/Auto Run External Program on Load*** is checked.

- Place the example files in a suitable directory with your other MMEdit .bas files.

The next step is only necessary if you want to use the Pi-cromite or ESP8266 directives.

- Edit the ***MMBasic.cmd*** file to ensure it has the correct paths to the ***MMbasic.exe*** and ***MMReplace.exe*** (If the DOS MMBasic is not installed then you can leave its path as the default value.)
- Start MMEdit and assign the ***MMBasic.cmd*** command to menu ***File/Run in DOS***
  .

The next step is only necessary if you want to use the #ARRAYGENERATE directives.

- Read the TBS forum post at http://www.thebackshed.com/forum/forum_posts.asp?TID=10253&PN=7
- Download the ***simpleArrayFuncGen.exe*** file. This should be placed in the same directory as ***MMReplace.exe***.

Summary of Directives added by MMReplace.exe	
Directive	*Usage*
#REPLACE *longname shortname*	Replaces all occurrences of *longname* with *shortname* when code is loaded.
#LIBRARYSTART	Marks beginning of code to be stored in the library
#LIBRARYEND	Marks end of code to be stored in library
#LIBRARYLOAD	Indicates only the library should be loaded
#LIBRARYLOAD *libname.lib*	Indicated library file should be generated for use with the Colour Maximite (CMM)
#ENDCODE	All code beyond this directive is not loaded
#PI_CLIPBOARD	Code should be loaded into clipboard for pasting into Pi-cromite using Putty. The Pi-cromite sysntax file should be selected.
#PI_AUTOSAVE *filename.bas*	Appends an AUTOSAVE command to the clipboard contents
#PI_AUTORUN	Appends a RUN command to the clipboard contents
/*	Start comment
*/	End comment
#ESP_CLIPBOARD	When the ESPBasic syntax file is selected and the #ESP_CLIPBOARD directive is present code is placed in the clipboard at Load and Run. The clipboard can be pasted into the ESPBasic edit window and saved.
#ESP_KEEPCOMMENTS	
#ARRAYSTART [string|integer|float] [cfunction|csub] *functionorsubname*	Indicates the start of the array definition and the parameters detail, the type of array, where a cfunction of a csub should be reduced and the *name* of the function or sub. The *name* is used as the base *name* for the *name*.cfg and *name*.bas file names passed to simpleArrayGenerator.
#ARRAYEND	Indicates the end of the array definition.
#ARRAYINCLUDE *functionorsubname*	Place in source where the generated cfunction/csub is to be included during Load and Run
#ARRAYGENERATE	This directive causes the array cfunction/csub to be generated during Load and Run. If not present the simpleArrayFuncGen.exe is not called so only any previously generated array cfunction/csub functions would be loaded.