MPE4WP
MACRO PROGRAMMING ENVIRONMENT FOR WORDPERFECT

USER REFERENCE GUIDE

Copyright © 1991 by Michael H. Shacter
7825 Marion Lane
Bethesda, Maryland 20814

MPE4WP
USER REFERENCE GUIDE

MPE4WP is the Macro Programming Environment for WordPerfect. MPE4WP is a superior alternative to the WordPerfect Macro Editor. You can now create and edit WordPerfect macros on the full WordPerfect editing screen using all the advanced editing features offered by WordPerfect. Alternatively, you may use any text editor or word processor that saves text in ASCII format.

Until now, editing WordPerfect macros was not easy. The built-in WordPerfect Macro Editor lacks all but the most primitive editing functions. Even the simplest edits are difficult. To move a line of code in the Macro Editor, the line must be deleted, one character at a time, then re-typed at its new location, with the hope that no errors will be introduced in the process.

Some of these shortcomings are remedied by the ED program, available from WordPerfect Corporation for an additional fee. Unfortunately, you cannot access WordPerfect's menus, when using ED. Therefore, you must know in advance the keystrokes that are necessary for your macro. This can be difficult because WordPerfect has at least three sets of menus, each with its own keystrokes. One set is for routine editing. Another set comes into play when you **Search** for a code. And still a third set becomes active when **Block** is on.

The optimal macro editor should support all of WordPerfect's editing features and should allow instant access to and recording of all keystrokes required to use WordPerfect's menus. MPE4WP fulfills these requirements and does much more.

For example, MPE4WP includes a *Record Mode* that emulates WordPerfect's macro recorder, but with the advantage that it may be utilized while editing a macro. Thus you may switch at any time between recording and editing a macro source file.

These are a few of the additional features provided by MPE4WP:

■ Insert Macro Commands, complete with tildes, using no more than three keystrokes

■ Use numbers rather than control or ASCII characters to position characters on macro screens

■ Convert existing macros into ASCII text for further editing or printing

- Draw boxes using WordPerfect's line draw feature
- Block, copy and move Macro Commands and text
- Easily cut and paste routines from other macros
- Use search and replace tools to debug and edit macros quickly
- Draft, edit and spell check text of any length for inclusion in macros
- Create and edit macros that are too big for the Macro Editor
- Employ the entire WordPerfect character set in your macros, in a simple intuitive fashion
- Use DOS wild cards to process several files in one operation
- Extend the number of ALT key macros from 26 to more than 130

Because of MPE4WP's memory conservation techniques, you may still create your macros without ever leaving WordPerfect, by using the **Shell/Dos** feature. And MPE4WP's speed will considerably shorten the time required to create and edit macros.

MPE4WP has been designed with both advanced and novice macro programmers in mind. Beginning macro programmers will find all the tools needed to learn how to create macros for streamlining and customizing WordPerfect. If you have been deterred from experimenting with macros by the aggravation attendant upon using the WordPerfect Macro Editor, now is the time to start learning. Advanced macro programmers will have access to superior programming tools and will never be impeded with by safeguards for the beginners.

WHAT IS MPE4WP?

MPE4WP consists of three tools to overcome the inadequacies of the WordPerfect Macro Editor:

MC.COM, the Macro Compiler, compiles (converts) ASCII text into WordPerfect macros. MC.COM is blazingly fast and minimizes memory usage, so it can be used without ever leaving WordPerfect, by using the **Shell/Dos** feature. MC.COM includes advanced features not available through the WordPerfect Macro Editor to simplify writing WordPerfect macros.

M2T.COM, the Macro Decompiler, reverses the task accomplished by MC.COM, converting WordPerfect macros into simple ASCII text. You can use M2T.COM to examine, modify, or print macros created by yourself or others,

outside the confines of the Macro Editor.

The Macro Programming Environment, which includes the MPE4WP keyboard and a series of macros, facilitates the creation of macros at the WordPerfect editing screen. Macro Commands are no more than three keystrokes away and are inserted in your macros complete with tildes and the cursor correctly positioned to take the next argument. Key Commands can be added by pressing the key represented by the command. You may record keystrokes and access menus in the same manner as you would with the macro recorder, without leaving the editing screen. Help screens are accessible with a single keystroke. Incorrect Macro and Key Commands can be deleted by striking one key.

MPE4WP is fully compatible with all releases of WordPerfect 5.1. With three minor exceptions relating to Key Command syntax, MC.COM and M2T.COM are also compatible WordPerfect 5.0. Of course, if you are still using WordPerfect 5.0, you must restrict yourself to the Macro and Key Commands available in version 5.0. For a list of commands available in both versions 5.1 and 5.0, please see the file COMMAND.LST. Regrettably, the MPE4WP macros will not work in WordPerfect 5.0. Even without the MPE4WP macros, you will find yourself more productive at the normal WordPerfect editing screen or in your favorite editor than in the Macro Editor.

Because you are probably eager to learn how to use MPE4WP, I shall have only a few words to say about registration at the beginning of this document. MPE4WP is user supported software. It is not public domain. In brief, if you are using MPE4WP for personal use and find the program is worth $15, you should pay a $15 registration fee. If you feel the program is worth more than $15 (bless you for your perspicacity), you will not be penalized, you still only pay $15, but you get a much better bargain than those who believe it is worth less. MPE4WP may not be used in business without a license. More information concerning registration and licensing may be found at the end of this document in the section entitled "REGISTRATION/ LICENSE/ COPYRIGHT".

ROAD MAP

When I first began using computers, I was often bewildered by software documentation that assumed I knew more than I did. The fact that there were two ways of describing the same concept (and the author of the documentation used both to avoid repetitiousness) added to the confusion. I have attempted to avoid these pitfalls. In the interest of completeness, however, the documentation points out how the same objective may be achieved in more than one way. Sometimes, the alternative approach will not make sense, but the explanation is included for the insight it adds into

how the program is operating. Don't be confused. If one approach works for you, that is sufficient. Later, when you become more proficient, and if you are still curious, you will understand how the other method works.

Before proceeding further, a brief note on syntax may be helpful. From time to time you will find a statement, command, or instruction in quotation marks (""). The quotation marks are merely intended to set off the statement, command, or instruction from the rest of the text. You never type the quotes. All keystroke sequences refer to WordPerfect 5.1; the commands for 5.0 should be quite similar. Where the steps are very important, each one is described. In other cases, a shorthand approach is employed that begins with the statement "Keystrokes:" and lists the keystrokes, just as they would be listed in the Macro Editor.

In addition to this User Reference Guide, the file README.MPE contains Quick Start information for macro experts and, for novices, a glossary explaining some of the terms that are used in this User Reference Guide. The balance of this section describes the organization of the User Reference Guide.

As you have already observed, the User Reference Guide begins with a series of objective statements praising the features contained in MPE4WP and describing how they will improve productivity. Congratulations on your perseverance, you will arrive shortly at information about Installing MPE4WP.

The two sections after installation describe how to use MC.COM and M2T.COM. A great deal of attention has been devoted to the user interface of both programs. You could probably run either without a tutorial. Nevertheless, the documentation also contains information about usage, especially command line input, that would not ordinarily be apparent without reading these two sections.

The section about Formatting Text Files for MC.COM presents information about the special features offered by MC.COM, as well as a few precautions that are required when creating your source code. Among other things, you will learn any easy way to position messages in your macros. If you learn nothing else from this section, at least remember to save your source code in Generic WP format.

The next section contains information on the MPE4WP macros and keyboard. MPE4WP supplies you with a special keyboard and macros that eliminate the need to type out the names of Macro and Key Commands. Many commands may be entered with a single keystroke. No command requires more than 3 keystrokes. MPE4WP also allows you to record keystrokes and play them back, just like WordPerfect's macro recorder,

except that you can access menus while in the process of editing your macro. This section is worth reading carefully.

One of the more powerful features of WordPerfect's main editing screen for creating macros is its impressive line drawing tools. Read the next section for hints on drawing attractive boxes for your macros.

A common complaint about WordPerfect is the shortage of ALT key macros. Twenty-six doesn't seem to be enough. The section entitled "{One-Key} Macros and XALT Keys" explains how to simulate any number of additional ALT keys. The following section on Key Macros discusses the appropriate place for these utilities and offers a word of caution about using them in your macros.

MPE4WP supports all the Macro and Key Commands available in the present version of WordPerfect 5.1. But times change, and the next section on Anti-Obsolescence, explains how, when WordPerfect is ready with a new command, MPE4WP will be ready too.

In the course of preparing the MPE4WP macros, I learned a few things, which I try to share with you in the section on Macro Programming Tips.

I hope the MPE4WP programs and macros show a level of ingenuity and creativity by the author. They also rely in part on the work of others. Read Program Notes for information on sources.

What can I say about Standard Disclaimers except don't blame me if you don't read it.

The final section is one that is important to me and should be important to you. It deals with Registration and the related issues of License and Copyright. Please read it.

If you need help writing macros, you should not hesitate to call WordPerfect's toll free number. The number for macros is (800) 541-5129. Don't ask them questions about MPE4WP; those are my responsibility. If you need assistance with MPE4WP, you may write to me at the address listed in the section on Registration. This is the same address emblazoned on your screen when you run MC.COM or M2T.COM. You may also leave a message for me on CompuServe (ID No. 76170,1627). The WordPerfect Manual contains lucid explanations of many of the topics addressed in this User Reference Guide. Unfortunately, the WordPerfect Manual can be fairly opaque when explaining how to create and edit macros. You therefore may also wish to consider purchasing Gordon McComb's book *WordPerfect 5.1 Macros and Templates*, which appears to contain an excellent discussion on a

variety of macro-related subjects, for novice and expert alike, and was a *PC Magazine* Editor's Choice.

INSTALLING MPE4WP

The MPE4WP package contains the following files:

README.MPE

MPE4WP Quick Start and Glossary
MANUAL.MPE

MPE4WP User Reference Guide formatted as a WordPerfect document
MC.COM

Macro Compiler described in MANUAL.MPE and README.MPE
M2T.COM

Macro Decompiler described in MANUAL.MPE and README.MPE
MPE4WP-I.WPK

Integrated MPE4WP keyboard described in MANUAL.MPE and README.MPE
MPE4WP.WPK

Alternative MPE4WP keyboard described in MANUAL.MPE and README.MPE

CMDS_A-L.WPM
CMDS_M-Z.WPM
{CREATE}.WPM
{EXPAND}.WPM
{HELP}.WPM
{INSERT}.WPM
{KEY}.WPM
{RECORD}.WPM
{SPCIAL}.WPM

MPE4WP macro files for use with MPE4WP.WPK keyboard described in MANUAL.MPE and README.MPE
(Do not install these macro files if you elect to use the MPE4WPK-I.WPK keyboard)

COMMAND.LST

List of Commands supported by MPE4WP described in MANUAL.MPE and README.MPE

COMMAND.WPM

Test macro described in COMMAND.LST
REGISTER.FRM

Information on changes from Version 1.1

MPE4WP is furnished with two alternative keyboard files: MPE4WP.WPK and MPE4WP-I.WPK. The MPE4WP.WPK keyboard works in tandem with an external set of macro files. In contrast, MPE4WP-I.WPK is an integrated keyboard in which the macros have been incorporated into the keyboard file, eliminating the need for the external macro files. Each approach has its advantages and disadvantages.

In favor of the MPE4WP-I.WPK keyboard, fewer files are required, and the macros should execute more quickly on slower machines. The reduction in number of files comes at the expense of a larger keyboard file and a concomitant increase in the amount of memory required to load the keyboard file. When WordPerfect loads a keyboard file, all the Key Macros associated with the file are loaded into memory. If available memory becomes tight, WordPerfect will fail, on occasion, to execute Key Macros properly. This shortcoming can generally be remedied by going into **Setup** and re-selecting the keyboard file. Even though the process of re-selection seems redundant, it frequently causes WordPerfect to remember the special keyboard assignments. For most users memory economization will not be an issue, so MPE4WP-I.WPK will be a good choice.

The advantage of the MPE4WP.WPK keyboard is better memory conservation, because the external macros are loaded and released from memory one at a time, when the corresponding key is pressed. In addition, it is easier to edit conventional file macros than Key Macros. Finally, the confusion associated with macro file proliferation can be minimized by designating a separate directory for the MPE4WP macros. For information about locating the MPE4WP macros in a user-specified directory, see the section entitled "{One-Key} Macros and XALT Keys".

Copy either MPE4WP.WPK or MPE4WP-I.WPK to the same directory as your other macro and keyboard files. If you opt for MPE4WP.WPK, the MPE4WP macro files should also be copied to your macro directory or to a user-specified directory in accordance with the instructions contained in the section entitled "{One-Key} Macros and XALT Keys". To verify the name of your macro directory, check in **Setup** (Keystrokes: **{Setup}**L). Entry number 2 tells where your macro files are located:

Setup: Location of Files

1 - **B**ackup Files **C:\WP51\BAK**

-> **2** - **K**eyboard/Macro Files **C:\WP51\MACRO**

If you do not have a special directory for your macro and keyboard files, now might be a good time to create one. Otherwise, place the keyboard file (and the MPE4WP macro files, if you select the MPE4WP keyboard) in the same directory as WP.EXE. Ideally, MC.COM and M2T.COM will be placed in a directory on your path. If the directory containing WP.EXE is on your path, this would be a good place to locate MC.COM and M2T.COM. If you plan to run MC.COM from within WordPerfect, you may need to modify the *Create* macro supplied with MPE4WP. See the section entitled "Create" for further details.

The MPE4WP macros can also be renamed to operate with my other program, MALT (More Alt-Keys for the Perfects). MALT is a memory resident program (TSR) that adds 90 new Alt-like macro keys using CapsLock + Key for use with WordPerfect and other members of the Perfect family. MALT uses slightly more than 4K RAM, can be unloaded, and does not interfere with the normal operation of CapsLock.

MC.COM

MC.COM was created with the following design considerations: accuracy, speed, ease of use, and minimal memory usage, so it can be utilized by shelling out from WordPerfect. MC.COM provides extensive on screen prompts, so much of the following discussion is superfluous.

To start up MC.COM simply type MC [/M] [/N] at the DOS prompt. The brackets surrounding the two switches [/M] and [/N] indicate that they are optional. You do not type them. The /M[onochrome] switch forces MC.COM to run in monochrome, this is useful when a mono or LCD monitor is attached to a CGA video card as is the case with my NEC MultiSpeed. The /N[o error file] switch tells MC.COM not to print an error log when it encounters an error in your source code. See the sections entitled "Create" for details about starting MC.COM from within WordPerfect and "MC.ERR" for information about the error file. You can also use the command line to tell MC.COM which files to process. See the section on "Using the Command Line" for further details.

MC.COM will first ask for the name of your source file. Enter the name, using MC.COM's built-in editor. If the source file is not in your current directory, you may specify the full path name. The editor supports the following keys:

Home: Move cursor to beginning of file name
End: Move cursor to end of file name

ALT-C:	Clear input field
ALT-R:	Restore input field cleared by ALT-C
Insert:	Toggle insert and typeover
Enter:	Accept input
Down:	Move to next editing field (from Source)
Up:	Move to previous editing field (from Macro or Destination)
ESCape:	Move to previous editing field, if any, then quit.

If you understand the DOS directory abbreviations ".." and ".", you will find that MC.COM does too. The full path name including drive, directory and file name, cannot exceed 58 characters. If the full path name of your file is more than 58 characters, there are two things you can do. One is to move the file to a higher directory. The other is to rethink your directory organization strategy.

After you have entered the source file name press either ENTER or the DOWN arrow. MC.COM will verify that the file exists and will tell you its full path name. MC.COM will then prompt you for the name of your new macro. MC.COM assumes that you wish to place your new macro in the same directory as the source file, with the same root name and the extension WPM. If this assumption is incorrect, you may edit the prompt. Unlike some other input routines, the prompt is not obliterated when you type a letter in the first position. If the assumption is completely off base, press ALT-C to clear the input field. If you want your new macro to have the same root name as the source file, you need only specify the output directory; MC.COM will be responsible for filling in the macro name. If you omit the WPM extension from the name of your macro file, MC.COM will add it for you. If you do not want your macro to have an extension, put a period after the name of the macro.

You will always be warned if there is any possibility of overwriting an existing file. MC.COM differs from WordPerfect, however, in the following respect. You do not receive more than one warning, and you do not hit any special keys to proceed. You just hit ENTER. Since you probably want to overwrite an existing macro with the same root name as your source file, you ignore a warning by pressing ENTER and MC.COM continues processing. You press any key other than ENTER to revise the name of the macro file. In the simplest case, you type the name of the source file, press ENTER twice, and you are through. If you wish to revise the name of your source file, you may press ESCape or the UP arrow. If you do not like warnings and you do not want prompts, then you should see the section on "Using the Command Line".

If you make an error entering the source or macro file name, MC.COM will do its best to identify the error and give you an opportunity to correct it.

Use the UP and DOWN arrows to move between the source and macro input fields until you are satisfied.

When you have satisfactorily entered the source and macro file names, MC.COM will process your file. You should find this an extremely fast procedure.

MC.ERR

MC.COM searches for certain errors in your source code. Such errors as misspelled Macro and Key Commands, and unintentionally omitted open and close braces are identified. Regrettably, MC.COM does not identify errors in macro syntax, such as missing tildes (the bane of WordPerfect macro programmers). If MC.COM finds any apparent errors, it will incorporate the questionable command in your macro as text. It then looks for a file called MC.ERR in the same directory as MC.COM. If MC.ERR exists, MC.COM will append a list of the new errors to the end. Otherwise, MC.COM will create a new file MC.ERR.

MC.ERR identifies the file that is being processed and lists all misspelled Macro and Key Commands. If one brace is missing, then MC.ERR will print the preceding or following 16 characters, depending on whether the open or close brace is omitted. This 16-character string will be truncated, however, when it would include all or part of another command. In order to maintain a running log of errors and to avoid having more than one MC.ERR file in several directories, MC.ERR is always placed in the same directory as MC.COM. If you do not want an error file, use the /N command switch when you start MC.COM, i.e. MC /N.

Wild Cards

To process more than one file at a time, you may use DOS wild cards. When MC.COM sees that you have used a wild card in the source file name it will check and advise you how many files will be processed. You may press ESCape to revise your entry. Since MC.COM will faithfully follow your instructions, it is best to be as specific as possible. For example, specifying "*.*" as your source specification is inadvisable, because it may encompass more than source code files as well as more than one file with the same root name. The specification "*.TXT" would be suitably specific. In any event, if you get a message that MC.COM is ready to process 500 files and you thought there were only 6, you can press the ESCape key to revise the file specification. Regardless of what you use as your source specification, MC.COM (and DOS, too, for that matter) will not allow you to have two files

with the same name in the same directory.

In a wildcard operation, you may choose only the destination directory. You will be prompted with the name of the directory containing your source files. As with all MC.COM prompts, you may edit this one as you please. You should not enter a file name. The new macros will have the same root name as your source files and the extension "WPM". When you think about it, there really is not too much to specify in terms of output file names in a wild card operation, except for an alternative extension. Modifying the root name can be done, but requires more knowledge of wild card usage than most people care to possess. If you do enter a file name, MC.COM will think you mean a directory with the name of your file and you will receive a message saying something like, "I am sorry, I cannot find the directory \WP51\ TEST\ *.WPM\ on the C: drive." Leave the destination field blank to have your new macros placed in the default directory.

When you have selected the destination directory. You will be advised if any files located there match the root file specification of your source files combined with the extension "WPM". For example, if you enter the file specification "*.TXT", MC.COM will tell you if any files match the specification "*.WPM"; i.e., the root name "*" combined with the extension "WPM". This may not be too informative, because the specification for the root name is so general. For example, if you have specified "*.TXT" to process the three files: FIND.TXT, FORMAT.TXT, and PRINT.TXT and the macros MARGIN.WPM and HEADER.WPM are in the destination directory, you will get a warning message, even though no files will be overwritten, because none of the "TXT" file shares the same root name with any of the "WPM" files. It would have been possible to make MC.COM more precise in this respect, but would have required more memory, and MC.COM has been designed to minimize memory usage.

Command Line Processing

MC.COM also has the capacity to process files from the command line. Be warned, however, that MC.COM does not provide any warning before overwriting files, when you use the command line to enter file names. The format for command line processing is as follows:

MC [/M] [/N] [source file] [macro file].

If you use either or both of the command switches (/M and /N) you must leave a space before entering the source file name. Similarly, you must leave a space between the source file name and the macro file name, if you choose to specify one. Do not place slashes before the source and macro file

names. If you omit the macro file name from the command line, MC.COM will place your macro in the same directory as your source file, with the same root name as your source file and the extension "WPM". The rules for entering the source and macro file names at the command line are the same as those for entering file names on the MC.COM screen. If you make a mistake entering file names on the command line, MC.COM will advise you of the mistake and allow you to make corrections using its built in editing facilities.

M2T.COM

M2T.COM is the Macro Decompiler. (It cannot be named MD.COM because of a conflict with the DOS "Make Directory" command, so it is called M2T for "Macro to Text".) M2T.COM is similar in operation to MC.COM. To start it, simply type M2T [/M]. Since M2T.COM does not produce an error file, there is no /N switch.

Like MC.COM, M2T.COM begins by asking for the name of a source file, except that in the case of M2T.COM, the source file is a macro. If you omit an extension from the name of your macro, M2T.COM will add the extension "WPM", just as WordPerfect does. The default extension for the output file is "TXT". You may change the extension if you wish.

Before processing a file, M2T.COM will verify that it is a macro file. It is unlikely that M2T.COM will make a false assessment. Nevertheless, where possible, M2T.COM offers a choice of overriding its conclusion. You may also terminate processing, revise the file name when processing a single file, or skip to the next file in a wildcard operation. If you insist that M2T.COM process a file that is not really a macro file, M2T.COM will faithfully attempt to comply, but the output file will probably be filled with the notation **{?}**, the indicator of an unknown macro command.

M2T.COM automatically processes the output text file to conform to the format understood by MC.COM. This means that M2T.COM will add an extra open brace if you have used the open brace as text. M2T.COM does not add extra close braces, since they are usually unnecessary. M2T.COM will include the description from an existing macro (if it has one), it at the top of the text file. For additional information about formatting source files, see the section entitled "Formatting Text Files for MC.COM".

Since many text editors and word processors do not directly support the ASCII characters from 1 to 31 (the control codes) M2T.COM converts them to their numerical form enclosed in brackets. Thus ASCII character number 3 (♥) becomes **{3}**. For additional information about the **{n}**

notation, see the section entitled "Character Positioning Codes".

Whenever M2T.COM encounters a character in the WordPerfect character set that is not also an ASCII character, it interprets the character in the standard format understood by MC.COM; i.e. as two numbers in brackets separated by a comma. The first number is the character set, the second is the number of the character in the set. For additional information on the WordPerfect character set, see the section entitled "The WordPerfect Character Set".

M2T.COM does not write an error file corresponding to MC.ERR. In a fully operational macro, there should not be any errors to report. M2T.COM understands all Macro and Key Commands supported by the current release of WordPerfect. If new Macro or Key Commands are included in a future interim release of WordPerfect, M2T.COM will print the number of the new command in the format **{MACRO CMD n}** or **{KEY CMD n}**, where "n" is the command number. Of course, you can expect that a new release of MPE4WP will be released instantaneously to support the new release of WordPerfect. (For more information on the **{MACRO CMD n}** and **{KEY CMD n}** notation, see the section entitled "Anti-Obsolescence".) If M2T.COM should encounter a code that it is foreign to WordPerfect macros, it will insert the notation **{?}** into your text. This should only occur if you try to process a file that is not a macro.

There are other programs that will convert macros into ASCII text or WordPerfect format. The output of these programs may require editing for use by MC.COM. Some imitate the Macro Editor by using centered dots instead of spaces. Others use variants of Macro or Key Commands instead of the exact form utilized by WordPerfect. Minor variations, like the substitution of **{Left Search}** for **{Search Left}**, are not serious, if all you are planning to do is print or study the output, but they will not be understood by MC.COM. In addition, you will have to edit any information about version and description that may be added to the text.

FORMATTING TEXT FILES FOR MC.COM

The first step in creating or editing a source file for processing by MC.COM is simply to create an ASCII text analogue of the macro; i.e. the text should correspond to what would appear on the screen of the Macro Editor. Using the macro tools included with MPE4WP, you will find that you use fewer keystrokes to "type" a macro on the WordPerfect screen than you would use in the Macro Editor. For more information on the macro tools, see the section entitled "The MPE4WP Macros and Keyboard".

There are few rules that must be observed.

Spaces

Use ordinary spaces, not the centered dots (ASCII 250) utilized by the Macro Editor. For example, correctly formatted text will look like this:

{PROMPT}Press any key to continue~

not like this:

{PROMPT}Press·any·key·to·continue~

Spelling

Case is not important when entering Macro and Key Commands, but spelling is. Thus, you may enter the prompt Macro Command as **{PROMPT}**, {prompt}, or {PrompT}. Likewise, the spell Key Command could be **{Spell}**, {SPELL}, {SpEll}, or any other variation of upper- and lower-case letters. You will get an error message, however, if you misspell a command, as for example in {PROMT} or {Spill}. Spaces are an important part of spelling, too. **{ON CANCEL}** is spelled correctly. In contrast, {ONCANCEL} is spelled incorrectly and will yield an error message. If you use the MPE4WP macros, you will never have to worry about spelling errors. Otherwise, refer to the list of current Macro and Key Commands in the COMMAND.LST file.

Character Positioning Codes

MPE4WP eliminates the need for the awkward character positioning codes required by the Macro Editor. Instead of using cryptic Key Commands or ASCII characters, you may use the notation **{n}** (where "n" is any number between 0 and 254) to represent column and/or row numbers. Thus, if you want to position a prompt at column 3, row 65, you type **{^P}{3}{65}**, rather than **{^P}**♥A. The **{n}** notation may be used to insert any ASCII character into your macro and is recommended for control characters (ASCII 1 to 31). (For those who know BASIC, **{n}** is equivalent to CHR$(n), where "n" is any number between 0 and 254.) Incidentally, MC.COM correctly interprets **{0}** as the appropriate WordPerfect code (ASCII 254) to position the cursor at column or row 0. In addition, if you are creating your source code in WordPerfect, you may use automatic numbering within the braces to automatically increment the numbers. For more information on this

capability, see the section entitled "Making Boxes with MPE4WP".

If you are old fashioned, you may continue to use ASCII characters or Key Commands to position your messages, with the following exceptions:

You may not use ASCII characters to position a message on column 13, row 10 because the two ASCII characters (carriage return and line feed) together represent the standard symbol for a hard return. You may use either alone, or both in reverse order (column 10, row 13).

You may not use the tab character (ASCII 9) to position characters on row or column 9. ASCII 9 is used as a macro formatting character.

ASCII 26, which appears as a right pointing arrow, is understood by many text editors and word processors to be the end of file marker, and they will not read past the point in text where ASCII 26 occurs. MC.COM does understand ASCII 26 and will correctly interpret it wherever it occurs in your source code, either in your text or at the end. You are warned, however, that unpredictable results may arise from other quarters. For example, WordPerfect will allow you to enter the ASCII 26 character in your text, but will not save it in Generic WP format. In addition, WordPerfect will ignore ASCII 26 when it loads a DOS text file.

While WordPerfect will save all control characters except ASCII 26 in a Generic WP file, it may not recognize these characters when you try to retrieve the same file later.

The WordPerfect Character Set

The WordPerfect character sets contain 1500 characters, including all the ASCII characters. The characters are divided into 12 sets, numbered from 0 to 11. In addition a thirteenth set, set 12, is a user-defined character set. A character is identified by its set number followed by its character number within that set. At the editing screen, a WordPerfect character is inserted into a document by pressing CTRL-2 or CTRL-V then typing the number of the character set, a comma, the number of the character in the set, and pressing **{Enter}**. Unless the WordPerfect character is also an ASCII character, it will be converted into a block (ASCII 254) when you save your source code in Generic WP format.

Using MPE4WP you may instruct your macro to insert any character in the WordPerfect character set with the shorthand command {s,n}, where "s" is the number of the character set and "n" is the number of the character

within the set. For example, the copyright symbol--the "C" in a circle--is character number 23 in character set 4. To incorporate the copyright symbol in your macro, the format would be **{4,23}**. You can also use **{^V}** followed by (1) the number of the character set, (2) a comma, (3) the character number, and (4) **{Enter}**, just as you would in the Macro Editor. Using this format for the copyright symbol, the relevant part of your source code would read "**{^V}**4,23**{Enter}**".

Note: The character "ÿ" is ASCII character number 152 and is character number 75 in WordPerfect character set number 1. Because of a bug in WordPerfect, using the ALT key and the numeric keypad to enter ASCII number 152 inserts character number 139 in set 1, instead. Character 139 looks like a "ÿ" on-screen, but prints as the digraph "ij". Unlike WordPerfect, MC.COM correctly interprets a "ÿ" in your source code as character number 75 in set 1. To be sure the correct character is inserted in your macro, I suggest you use the notation {1,75} to insert the character ÿ and {1,139} to insert the digraph "ij", which is the approach followed by M2T.COM. You are in good company if you just read this note and shrugged, since that is the same response supplied by WordPerfect technical support.

Although the **{Compose}** Key Command (CTRL-2) is recognized by the MPE4WP programs, it is not recognized by WordPerfect. Thus the statement **{Compose}**4,23 **{Enter}** will not produce the intended result.

In general, MC.COM does not keep track of out range values for the WordPerfect character set. If you include values that are out of the appropriate range, MC.COM and WordPerfect will do their best to accommodate you, but the codes will be meaningless. Extreme out of range numbers, may yield unpredictable results. One exception to error checking is **{0,0}** which would cause the premature termination of your macro. Please do not confuse "**{0}{0}**", which, when used in conjunction with **{^P}**, means column 0, row 0, with the meaningless notation "**{0,0}**", which would mean character number 0 in character set 0.

Descriptions

A source file may contain an optional description for inclusion in a macro. The description should be preceded by the command **{DESCRIPTION}** (upper-case, lower-case or any combination of the two is ok) and terminated by either a tilde or a hard return. WordPerfect imposes a 39-character limitation on the length of a description, so MC.COM will truncate any description that is longer than 39 characters. The description may be on a line by itself or on a line with other commands. To minimize complication, place the description at the top of the macro source file. To

accommodate those programmers who prefer to begin a source file with a comment describing the operation of the macro and the programming strategy, the description may be placed anywhere within the first 4000 characters of the source file. If you have a reason for using the command **{DESCRIPTION}** as text, precede it with an extra open brace, like this: {{DESCRIPTION}. The following sample descriptions are all acceptable and equivalent:

{Description}Creates header with date & page number~

or

{DESCRIPTION}Creates header with date & page number

or

{:}chatter, chatter, chatter~
{description}Creates header with date & page number~

Saving and Retrieving Source Code

All files must be saved as ASCII text, i.e. Generic WP. If you are working in WordPerfect, you save your text by taking the following steps: (a) Press CTRL-F5 (**Text In/Out**); (b) select "**A**" or "**3**" for "Save As", (c) press "**1**" or "**G**" for "Generic". Alternatively, you may simply press CTRL-G on the MPE4WP keyboard to call the *Save Generic* macro. For additional information, see the section entitled "Save Generic".

Note: WordPerfect allows you to save your files in two formats of ASCII text. These are DOS Text and Generic Word Processing Text. Both DOS Text and Generic Word Processing conversions remove WordPerfect formatting codes. In the DOS Text conversion, tabs, indents, and center codes are converted to spaces and hard carriage returns are substituted for soft returns. In contrast, the important feature of Generic Word Processing Text, for our purposes, is that tabs are preserved. As discussed elsewhere, tabs are an important formatting tool in WordPerfect macros, which is lost if they are converted to spaces as occurs in the DOS Text conversion. In addition, the conversion of tabs to spaces would cause your macro to insert spurious spaces into your document when it is run. Therefore, you should always save your source code in Generic WP format, never as DOS Text.

WordPerfect will also permit you to retrieve source code saved in Generic WP format. For users of WordPerfect 5.1, the conversion is automatic. In WordPerfect 5.0, you must use **Text In/Out** (CTRL-F5), then

select **2 R**etrieve (CR/LF to [HRt]. Any control codes (ASCII 1 to 31) contained in your source code will either be converted to a space or rendered as a control character in the form of a caret (^) followed by a capital letter or punctuation mark. Thus, ASCII 10 is converted to a space, ASCII 19 becomes "^S" and ASCII 30 becomes "^^". The control characters may look funny, but no further action is required on your part, since these characters are correctly interpreted by WordPerfect and MC.COM, when you save them in Generic WP format. Please note that these control characters, although they look like two characters, are actually one character. To eliminate a potential source of confusion, please bear in mind that control characters such as ^P are not the same as the control macro characters such as **{^P}**. The former is a low order ASCII character, i.e. less than 32, and the latter is a macro formatting character. This is a good place to remind you that the simplest and most predictable way to insert a control code in a macro is to use the **{n}** notation.

You may save your source code in WordPerfect format. This may be desirable, if you are using formatting commands or you wish to avoid DOS Text conversion each time you make revisions. Just remember, the final step before compilation must be to save your source code in Generic WP format. If you wish to save your source code in WordPerfect format for posterity, give the generic format a different name. After your macro is compiled, you may delete the ASCII text.

Formatting Characters (Tabs and Hard Returns)

You may use tabs and hard carriage returns to format your macros, just as you would in the Macro Editor. If you are not creating your source code in WordPerfect, you should be sure that your editor saves text with hard tabs (ASCII 9) and does not convert tabs to spaces.

Since you will now be editing your macros at the normal editing screen, rather than in the Macro Editor, you may lose sight of the fact that you must use the Key Commands **{Tab}** and **{Enter}**, if you want your new macro to insert a tab or a hard return when it is run.

Braces

MC.COM expects that any text beginning with an open brace ({) is the beginning of a Macro Command, a Key Command, or one of the following special keys: **{ALT x}**, where "x" is any "legal" character, **{KEY MACRO n}**, where "n" is the number of a Key Macro, or **{VAR n}**, where "n" is a number from 0 to 9. (See the section entitled "XALT Keys" for a discussion of

"legal" characters.) An error message will be generated if the text between the braces is not a recognized command or if the close brace is missing. If you want to treat the open brace as text, or if you want a command to appear as text when the macro is run, simply add an extra open brace, so your source code would look like this: {{. Failure to add an open brace before ordinary text will result in a harmless error message; MC.COM will still treat the open brace as text. To treat a command as text (you usually won't), you must add an extra open brace.

MC.COM also recognizes when a close brace is not paired with an open brace. This is useful for detecting a missing open brace. For example, "**{ASSIGN}**key~ **{KTON} SYSTEM}**right~~~" would generate an error because the open brace of the **{SYSTEM}** command is missing. A lone close brace will be incorporated in your macro as text and MC.COM will report an error. A close brace that is paired with double open braces is not treated as a lone close brace. If you have a reason for incorporating a lone close brace in your macro as text, and you wish to avoid the error message, you should add an extra close brace. The only time I have legitimately encountered a lone close brace was in preparing the MPE4WP macro help screens. Some of these screens incorporate the Macro Commands as text and use special attribute commands to highlight letters of the command name, thereby separating the close brace from its open counterpart. In real life, you should never encounter this situation.

Silent Comments

In addition to the WordPerfect macro comment (**{;}**), which is incorporated into the final macro, MPE4WP also supports silent comments. Silent comments, as the name suggests, are ones that are not included in your macro. The silent comment may be helpful in reminding you why you adopted certain programming strategies, but would not be informative to the end user of your macro. Moreover, eliminating comments may speed up the operation of your macro. The beginning of a silent comment is marked by a colon in braces: "**{:}**" and, like the WordPerfect comment, terminated by a tilde. For example, **{:}**This is a silent comment~. When you have completed your source code, you may wish to use **Replace** (ALT-F2) to change one or more of your WordPerfect comments to silent comments by replacing the semi-colon with a colon.

Line Length

MPE4WP does not impose any artificial limitations on text line length. You may enter, edit, and spell check text of any length in your source code

for inclusion in your macro, subject only to memory limitations imposed by WordPerfect.

THE MPE4WP MACROS AND KEYBOARD

MPE4WP is supplied with a set of macros to simplify creating and editing macros at the WordPerfect editing screen. Each macro is assigned a key on one of the MPE4WP keyboards. The MPE4WP-I keyboard uses Key Macros exclusively. The MPE4WP keyboard uses a combination of Key Macros and {One-Key} Macros. The macros associated with the two keyboards are identical. You will need WordPerfect 5.1 to use these macros. If there is sufficient demand, I will try to prepare a set of corresponding macros for WordPerfect 5.0 to the extent that 5.0's more limited macro language permits.

Aside from the MPE4WP macro key assignments, the MPE4WP/MPE4WP-I keyboard is the original WordPerfect keyboard. If you are accustomed to using your own remapped keyboard, you may wish to make changes to the MPE4WP/MPE4WP-I keyboard. Please see the section entitled "Record Mode" for a discussion of how remapping may affect the operation of some of the MPE4WP macros.

To use the MPE4WP/MPE4WP-I keyboard, press **Setup** (Shift-F1). Now, select **5**, Keyboard **L**ayout. Position the cursor on MPE4WP or MPE4WP-I, as the case may be, and press **1**, **S**, or **Enter** to select the keyboard.

The following table lists each of the MPE4WP macros, identifies the key to which it is assigned, and summarizes the use of each key. The macro file associated with the key on the MPE4WP keyboard is also listed. (Note: The macro file name is irrelevant to the MPE4WP-I keyboard because all its macros are Key Macros.) You will notice that the root names of many of the MPE4WP macros are enclosed in braces. The purpose of the braces is to avoid name conflicts with existing macros. You may already have a macro named CREATE.WPM, but are unlikely to have one named {CREATE}.WPM. Each of the macros is described in detail in the pages following the table.

MPE4WP MACROS

Name

Key

File name

CTRL-H

Just what its name suggests
Record Mode

CTRL-R

{RECORD}.WPM

Record keystrokes in WordPerfect in an analogous manner to macro recording
Insert Mode

CTRL-I

{INSERT}.WPM

Insert Key Commands in an analogous manner to the Macro Editor's Command Insert Mode Key

CTRL-K

{KEY}.WPM

Insert a single Key Command in an analogous manner to pressing CTRL-V in the Macro Editor
Special Keys

CTRL-S

{SPCIAL}.WPM

Insert the commands **{ALT x}**, **{KEY MACRO n}** and **{VAR n}**
Macro Commands
(A-L)

CTRL-{

Help screen of Macro Commands from A to L; insert a command (with tildes) by typing two-key mnemonic
Macro Commands
(M-Z)

CTRL-}

Help screen of Macro Commands from M to Z; insert a command (with tildes) by typing two-key mnemonic
Expand

{EXPAND}.WPM

Type two-key mnemonic then press CTRL-X to expand to Macro Command with tildes; uses same mnemonics as CTRL-{ and CTRL-}, but without help screens
Delete

CTRL-D

Key Macro

Delete Macro or Key Command on which cursor is located; if cursor is not on a command, delete first command to left
Create

CTRL-C

{CREATE}.WPM

Run MC.COM from within WordPerfect
Save Generic

CTRL-G

Key Macro

Save file in Generic Word Processing format
Clear Prompt

CTRL-Print Screen

Key Macro

Clear status prompt from screen in the event of unanticipated errors
Playback

CTRL-P

Key Macro

Play back recording in case of unanticipated interruptions

Help

Help lists each of the MPE4WP macros with a reminder of the key used to call the macro.

Record Mode

One of the more exasperating aspects of using the Macro Editor or ED, is the inability to record keystrokes required to access WordPerfect's deep menus. WordPerfect is very good at recording keystrokes when defining macros, but if you decide later to add a new command that accesses a menu, you generally must leave the Macro Editor, take notes as to which keys access the menu, then return to the Editor to type them in. This is not what we have computers for! Although **Help** is available in the Macro Editor, it only gives the beginning keystrokes. Moreover, **Help** supplies key numbers, not mnemonics. Finally, **Help** does not supply the alternative keystrokes to search for a code, when that becomes necessary. The solution is MPE4WP's *Record Mode*.

Record Mode operates in a similar manner to macro recording. It remembers all the keys you press, including all Key Commands. You start recording by pressing CTRL-R. A flashing prompt appears on the screen as a reminder that you are recording. *Record Mode* will remember each key you type. When you are ready to incorporate the saved keystrokes in your source code, press CTRL-Q. If you are not at the main editing screen, you will be returned there. You will then be prompted to position the cursor and press **Enter** to play back the text of all the keys you typed, including all Key Commands.

Record Mode has been programmed to record the original key assignments of all Key Commands. If you wish to use *Record Mode* with a keyboard in which the key assignments have been remapped, you may modify the *Record Mode* macro. The macro includes a comment explaining how to make the change. You should be aware, however, that if you have assigned a Key Macro (i.e. more than one keystroke) to a key, *Record Mode* will not recognize the keys in the Key Macro, but will still record the original key assignment. This is not a failing of *Record Mode*. It occurs because of the way WordPerfect handles the keyboard. Regrettably, *Record Mode* cannot recognize ALT keys. To insert an ALT key, use the *Special Keys* macro.

The maximum length of a *Record Mode* recording is almost 1,000

characters, more than long enough for the deepest WordPerfect menu. The only reasonable way to reach the maximum is to forget you are recording. If you ever reach the limit, *Record Mode* will warn you that it is returning you to the main screen for playback. The next keystroke will return you to the main editing screen to play back the keystrokes you have recorded.

In addition to recording keystrokes, *Record Mode* also inserts any formatting codes created while recording. For example, recording the sequence **{Format}** lm1 **{Enter}**1 **{Enter} {Enter} {Enter}** will also result in the insertion of the code **[L/R Mar:1",1"]**, which you will see by turning on Reveal Codes. It may be prudent to remove these formatting codes, although most of them are ignored when you save your source code as ASCII text. According to the WordPerfect manual, the following formatting codes will result in the addition of spaces: Center, ->Indent, and Flush Right, which would appear as spurious spaces when your macro is run.

Record Mode utilizes the Macro Command **{STATUS PROMPT}** to remind you that you are recording. This creates an anomaly when you access other editing screens, like the Header, Footer, and Footnote editing screens. The normal prompt at the bottom of the these editing screens--which looks something like this: "**Header A: Press Exit when done**"--is hidden by the *Record Mode*'s prompt "**Recording CTRL-Q to quit**". It is possible to avoid this problem in the editing screens by repeatedly calling a conventional prompt, either with the **{PROMPT}** command or in connection with **{CHAR}**. Using one of these alternative prompts, however, causes the more serious problem of trapping the cursor. The cursor does not appear on the screen where you are typing, instead it is captured by the prompt. Incidentally, the problem with the Status Prompt cannot be solved by simply relocating the prompt with character positioning codes. Regardless of the location of the status prompt, it still overrides the WordPerfect prompt in the other editing screens.

There are two ways to tell you are at the appropriate editing screen when using *Record Mode*. One is faith: you pressed the right keys so you must be in the header, footer, or footnote editing screen. (The footnote screen is less problematic, since there is usually a footnote symbol in the upper left corner of the screen.) The more objective method is to notice the abbreviated status line. The status line on the main editing screen looks like this: "**Doc 1 Pg 1 Ln 4 Pos 71**". The status line in the header editing screen looks like this: "**Ln 1 Pos 10**". Notice the omission of information regarding Doc number and Pg. You will have to remember to press **Exit** when you are done at the editing screen.

When using *Record Mode* or writing macros in general, I recommend that you use mnemonics instead of numbers. In my view, although

admittedly without much research, mnemonics are more likely than numbers to remain consistent from version to version of WordPerfect. When a new feature is added to a menu, if it is added anywhere but at the end, all the numbers will be increased, and all your macros will have to be revised. It is likely, however, that the mnemonic for the new feature will be chosen to avoid conflicts with existing mnemonics. Even if one mnemonic changes, the rest presumably will remain the same. In addition, when editing a macro, the mnemonics give a hint as to what the keystrokes represent.

In the interest of completeness, the *Record Mode* macro, as well as the *Insert Mode* and *Key* macros, provide for all Key Commands. In my experience, however, some Key Commands, such as **{Compose}**, **{Keyboard}**, and **{Help-Help-Left}** cannot be entered at the keyboard.

Playback and Clear Prompt

Playback (CTRL-P) and *Clear Prompt* (CTRL-Print Screen) are precautionary measures that you may never require. Although unlikely, *Record Mode* may terminate prematurely. In this case, by pressing CTRL-P, you may play back the keystrokes you recorded prior to the failure. *Playback* is a failsafe mechanism only and will not work if *Record Mode* terminates normally.

If *Record Mode* or *Insert Mode* terminates prematurely, the Status Prompt may remain on the screen. Use *Clear Prompt* to clear the prompt from the screen. (By the way, the WordPerfect keyboard edit screen erroneously shows that *Clear Prompt* is assigned to "CTRL-Num *" (instead of CTRL-Print Screen).

Insert Mode

Insert Mode (CTRL-I) is similar in operation to the Macro Editor's Command Insert Mode, which is accessed in the Macro Editor by pressing CTRL-F10. While in *Insert Mode*, pressing a key to which a Key Command is assigned, will result in the insertion of the textual representation of the Key Command in your source code. You may also insert text by pressing any letter, number, or punctuation key. Press CTRL-Q to quit *Insert Mode*.

Like *Record Mode*, *Insert Mode* uses original key assignments, and may be modified by following the instructions in the comment in the macro. See the discussion under *Record Mode* for considerations relating to remapped keys. In contrast to *Record Mode*, *Insert Mode* does not cause the insertion of formatting codes into your source code.

Key

Key (CTRL-K) is for inserting a single Key Command and is similar in usage to CTRL-V in the Macro Editor. Press CTRL-K and you will receive a prompt "**Key Command =**". Now press the key associated with the Key Command to insert the text of the command. Do not press **Cancel** if you decide against inserting a Key Command--that would simply insert the command **{Cancel}** into your source code. Instead, press any key that is not associated with a Key Command, like the space bar or any alpha-numeric key.

Special Keys

Special Keys (CTRL-S) are those keys that are not sensibly assigned to one or more keys because they have so many permutations. The special keys are **{ALT x}**, where "x" is any legal character, **{KEY MACRO n}**, where "n" is the number of a Key Macro, and **{VAR n}**, where "n" is a number from 0 to 9. (For an explanation of legal characters, see the section entitled "{One-Key} Macros and XALT Keys".)

Pressing CTRL-S will bring up a menu of the special keys. Press a mnemonic or number for your selection. You will then be prompted for a legal character or number, depending on your selection. *Special Keys* will only allow you enter numbers for Key Macro and VAR. *Special Keys* does not verify that a legal character has been entered for ALT. Illegal characters are, however, identified by MC.COM.

Macro Commands

Macro Commands is divided into two parts: A to L (CTRL-{) and M to Z (CTRL-}). (The WordPerfect Keyboard Edit screen shows CTRL-{ and CTRL-} as CTRL-[and CTRL-], which is another valid way of looking at things.) *Macro Commands* displays help screens giving the syntax for each of the Macro Commands supported by the most recent version of WordPerfect. Each of the commands is also identified by a two-letter mnemonic. You press the mnemonic keys (upper- or lower-case is acceptable) and the Macro Command is inserted into your source code, complete with tildes and cursor correctly positioned for the first command argument, if one is required. In the case of **{FOR}**, **{IF}**, or **{WHILE}**, the **{END FOR}**, **{END IF}**, or **{END WHILE}** is also inserted.

Expand

After you have learned the mnemonics for the *Macro Commands* you use most often, you will wish to use *Expand* (CTRL-X). *Expand* is much faster than *Macro Commands* because it does not require WordPerfect to draw help screens. Simply type the two-letter mnemonic (upper- or lower-case is acceptable) then press CTRL-X and the command with tildes and correctly positioned cursor are inserted in your text. For example, to insert the **{ASSIGN}** command into your source code, type "as", then press CTRL-X. The "as" will disappear and in its place will appear **{ASSIGN}**~~. The cursor will be positioned on the first tilde. If you reflexively add a space every time you type a word, don't worry. *Expand* will compensate for this habit. If you accidentally type an erroneous mnemonic, you will find the two letters to the left of the cursor mysteriously capitalized after you press CTRL-X.

You may change the mnemonics used by *Macro Commands* and *Expand* by changing the two-letter labels associated with the command. Suppose for example that you wanted to change the mnemonic for **{LOOK}** from LO to LK. You would change the line of the macro that reads **{LABEL}**LO~ to read **{LABEL}**LK~. That's all there is to it. Be careful not to assign the same mnemonic to two different commands. If CH stands for **{CHAR}**, it cannot also represent **{CHAIN}**. If you are very adventurous, you might try changing the prompts on the *Macro Commands* help screens, but unless you are careful, you will distort the boxes. For additional information on drawing help screens in macros, see the section entitled "Making Boxes with MPE4WP".

Delete

MPE4WP makes it easy for you to insert Macro and Key Commands into your source code. It is even easier to delete commands using *Delete* (CTRL-D). Place the cursor anywhere on the offending command and press CTRL-D. The command will be erased. If the cursor is not on a command, *Delete* will delete the first command it finds to the left of the cursor.

Save Generic

Save Generic (CTRL-G) is so simple I am almost ashamed to include it, but properly saving your macro as ASCII text is important enough to merit its own macro. *Save Generic* is for saving your source code as Generic Word Processing Text. Press CTRL-G and WordPerfect will prompt you "**Document**

to be saved (Generic WP):" If you have previously saved your source code in Generic WP format during the current editing session, you will, as is customary, be prompted with the name of your file.

Create

Create (CTRL-C) first checks to see that you have not made any changes to your source code since it was last saved. If *Create* detects that there have been any changes to the text in the current window since it was last saved, it will ask if you want to save the file. If you answer "Yes", *Create* will pause while your text is saved in Generic WP format. After you have saved your text, press **Exit** to continue. *Create* will then call MC.COM to compile your macro, i.e. to convert your source code into an operational macro.

Create uses WordPerfect's **Shell/DOS Command** function to run MC.COM. You must tell *Create* where MC.COM is located in order for it to do its job. If MC.COM is located in a directory on your path, you do not need to modify *Create*, but depending on the speed of your hard disk and the size of your disk cache, *Create* may operate faster with a full path name for MC.COM.

Create may be modified either with the Macro Editor or by revising the source code and recompiling with MC.COM. The procedure for accessing the Macro Editor will depend on which keyboard you are using: MPE4WP or MPW4WP-I. If you are using MPE4WP-I: press **Setup** (Shift F1), select **5**, Keyboard **L**ayout, position the cursor on MPE4WP-I, and press **7 E**dit, to enter the **Keyboard: Edit** menu. Position the cursor over CTRL-C (Create) and press **1**, **A**, or **Enter** to enter the Macro Editor.

To use the Macro Editor in connection with the MPE4WP keyboard, press **Macro Define** (CTRL-F10). If you have previously selected the MPE4WP keyboard, simply hit CTRL-C (as you would to edit an ALT key), otherwise type "{CREATE}" at the prompt. (Use the full path name for {CREATE} if it is not in your macro directory.) WordPerfect will then prompt you **{CREATE}.WPM Already Exists**: **1 R**eplace; **2 E**dit; **3 D**escription. Select **Edit**. To modify *Create* by revising the source code, first use M2T.COM to generate a source code file from {CREATE}.WPM.

Create contains instructions for adding the full path name in the opening comment. The first command in *Create*, after the opening comment, reads: **{ASSIGN}**FullPN~MC~. This establishes a variable called "FullPN" which is later used to tell the **Shell/DOS Command** function to run MC. If you have placed MC.COM in the "C:\WP51\" directory, you would

modify this line to read: **{ASSIGN}**FullPN~ C:\WP51\MC~. (The instructions in the macro use the special hard tilde **{~}** command, because it is not possible to use a real tilde. Do not be confused. When you modify the macro, leave the real tildes in place.) If you do not add the full path name, *Create* will remind you of the advisability of doing so when you run it. When you run *Create*, if it cannot find MC.COM, you will be treated to a screen with the message "Bad command or file name" at the top, the prompt "Press any key to continue" at the bottom, and the reminder to add the full path name in the middle.

Since MC.COM requires a scant 77K of memory to perform all its functions, you should have no problem running it from within WordPerfect. If MC.COM detects that there is insufficient memory, it will issue a message to that effect. This is only likely to happen if you have a very large document in the other window. In that event, you should exit from the large document and try running MC.COM again.

Although *Create* does its best to verify that your source code has been saved before calling MC.COM, you should not depend on it for this purpose. Because of an idiosyncrasy of WordPerfect associated with timed back ups, *Create* may not realize that the document has not been saved. If WordPerfect has made a timed back up of your source code in between your last keystroke and the time you press CTRL-C, *Create* will think your document has already been saved and will immediately call MC.COM. In addition, you may have last saved your source code in WordPerfect format (using F7 or F10), in which case *Create* would not realize that your source code had not been saved in Generic WP format. Thus, it is always advisable to use *Save Generic* to save your text before calling *Create*.

Create also has the salutary effect of clearing your macro from WordPerfect's "cache". You may thus run your new macro immediately after compilation, even if you had just run a previous version with the same name.

A Note on Source Code

The complete source code for the MPE4WP macros may be generated easily with M2T.COM. The macros with help screens may look formidable with their double open braces and the distorted looking boxes. Do not be discouraged, you will rarely need the double braces. Drawing boxes with MPE4WP is much simpler than in the Macro Editor. If you ignore those aspects of the MPE4WP macros that are generic to the mission of macro creation, you may find some helpful routines.

The *Create*, *Special Keys*, and *Menu* macros use the **{ELSE}** command

in connection with **{CASE}**. If you are using an interim release of WordPerfect dated August 20, 1990, or later, you may substitute the **{OTHERWISE}** command for **{ELSE}**. In brief, **{OTHERWISE}** is required when a **{CASE}** statement combined with **{ELSE}** is nested in an **{IF}** statement. The three MPE4WP macros that use **{CASE}** are not subject to this problem, so the substitution is not necessary.

MAKING BOXES WITH MPE4WP

Creating boxed messages in the Macro Editor is an ordeal that many forgo. With MPE4WP, however, creating boxes is simplified with WordPerfect's line draw tools. By following these steps, you can create attractive boxed prompts for your WordPerfect macros:

If you are planning to use character positioning codes, you will find it useful to set up WordPerfect to give you column and row coordinate readings. First, set the WordPerfect status line measurements to units. (Keystrokes: **{Setup}** eusu **{Enter} {Enter} {Enter}**) Now set the top margin to zero and the left and right margins to 0 and 6, respectively. The status line "Ln" and "Pos" readings will now roughly correspond to the WordPerfect screen coordinates. The only difference is that the top line will read 1 instead of 0. Thus, you must remember to subtract 1 from the row position in your macro.

Note: Some printer definitions will not permit you to have zero margins. The Standard Printer works admirably for this purpose. Unfortunately, if you have installed a printer, you may find it difficult to find this printer definition. Here are the steps to take: **{Print}**sal. This will give you a list of the printer files on your disk. One of these should be STANDARD.PRS. If it is, select it. If you do not find STANDARD.PRS, it may be in another directory, or you may have deleted it in a burst of economy, when you installed your printer. If you cannot find it anywhere, the DOS Text Printer contained on your PRINTER disk works just as well. The only inconvenience is finding the printer disk and going through the install printer procedure. The DOS Text Printer is found in the WPDM3.ALL File.

I also set tab spacing at 3 units apart, corresponding to the spacing in the Macro Editor. Setting the margins and tab spacing at the WordPerfect editing screen will have no effect on final macro output. They are simply conveniences in drafting source code.

The next step is to enter the text that you intend to use with your screen. Allow enough white space to the left and top of the text to permit the use of line draw characters.

Use the WordPerfect Line Draw tools to draw a box around the text.

If you want to center your box use center justification (Keystrokes: **{Format}**ljc **{Enter} {Enter}**) to find the correct position on the screen. (It is simpler to use center justification than **Center**, because the entire box, rather than just one line, will be centered. Note: Center justification is not available in WordPerfect 5.0; use **Center** instead.) You may also wish to use tabs to position your box for the purpose of taking coordinates. Use hard returns to position your box on the appropriate row. When the box is located on the screen to your satisfaction, place the cursor on the upper left corner. Now refer to the status line to determine the correct column and row coordinates (corresponding to Pos and Ln, respectively). In the case of a fractional position number, I generally round up.

After you have determined the correct coordinates for your box, you should delete the center justification code and any other codes you may have inserted to determine the correct coordinates.

You may now add attribute characters and character positioning codes. For character positioning, you use **{^P}** followed by two numbers in brackets. The first number is the column number, the second the row. To position the first line of your box on column 29, row 9, your source code would read **{^P}{29}{9}**. The next line would begin **{^P}{29}{10}**. A sample box might look like this:

```
{^P}{29}{}╔═══════════════════════╗
{^P}{29}{}│ Hello from MPE4WP     │
{^P}{29}{}│ I hope you are        │
{^P}{29}{}│ having fun            │
{^P}{29}{}│ drawing boxes.        │
{^P}{29}{}╚═══════════════════════╝
```

It should be evident that the row numbers, which increment by one each line, offer a good opportunity for the advantageous use of WordPerfect's automatic numbering. First, a Paragraph Number Definition should be inserted immediately before the box. The Paragraph Number Definition should specify the number of the beginning row (10 in the example) and should define Level 1 as "1" without any preceding or trailing punctuation. The definition looks in part like this:

Paragraph Number Definition

1 - **S**tarting Paragraph Number **10**
(in legal style)

	Levels				
	1	2	3	4	5
2 - Paragraph		1.	a.	i.	(1) (a)
3 - Outline		I.	A.	1.	a. (1)
4 - Legal (1.1.1)		1	.1	.1	.1 .1
5 - Bullets		●		-	■ *
6 - User-defined					
Current Definition	**1**	A.	1.	a.	(1)
Attach Previous Level		No	No	No	No

Notice, in particular, the Starting Paragraph Number, and level 1 of the Current Definition. The rest is not relevant for our purposes.

Finally, instead of putting the number 10 in the column position for the first line, you put a code for Paragraph Number: Level 1. (Keystrokes: **{Date/Outline}** p1 **{Enter}**) Do not type the character positioning instruction, i.e. **{^P}{29}{[Par Num:1]}**, six times. Instead, type it once, then block the character positioning command and copy it in front of each successive line of your box. Using Reveal Codes, your box now looks like this:

[Par Num Def:][HRt]
{^P}{29}{[Par Num:1]} ╔═══════════════════════╗**[HRt]**
{^P}{29}{[Par Num:1]} ║ Hello from MPE4WP ║**[HRt]**
{^P}{29}{[Par Num:1]} ║ I hope you are ║**[HRt]**
{^P}{29}{[Par Num:1]} ║ having fun ║**[HRt]**
{^P}{29}{[Par Num:1]} ║ drawing boxes. ║**[HRt]**
{^P}{29}{[Par Num:1]} ╚═══════════════════════╝**[HRt]**

If you later decide to have your box begin at a different row number, you simply edit the Paragraph Number Definition to change the Starting Paragraph Number. In our example, to have your box begin on row 15, instead of row 10, you would change the Starting Paragraph number to 15. All the row numbers will be automatically adjusted.

You may now add attribute commands to embellish your screen. Do not be disturbed if the box in your source code looks distorted, so long as you use the correct character positioning codes and do not inadvertently add spaces, your screen will appear correctly when you run the macro. Do not forget to add a **{PROMPT}**, **{CHAR}**, or other prompt command at the beginning, and a tilde at the end.

When the source code is saved as ASCII text (Generic WP), the paragraph numbering codes will all be converted into numbers and the

definition code will be omitted.

If you are creating a wide box, you may wish to select a wider "paper size" to prevent your boxes and messages from wrapping around on themselves. (Keystrokes: **{Format}**ps) Do this only after you have determined the appropriate coordinates for your boxes and messages. You may find with the wider paper size that part of the box or message scrolls off to the right of the screen, but I find this preferable to having the boxes wrapped on the screen.

Some of the MPE4WP macros have extra spaces around the periphery of the boxes. The purpose of these extra spaces is to clear out an additional area of the screen.

If you use WordPerfect formatting codes in your source code, you will want to save the archival copy of your source code in WordPerfect format. Just remember to save a temporary copy as ASCII text (Generic WP) before compiling with MC.COM.

I find it simplest to prepare the boxes in **Doc 2**. I then test them by creating a temporary macro that only contains a **{PROMPT}** with the message and closing tilde. After I am satisfied the message works in a test macro, I copy it into the source code in **Doc 1**.

{ONE-KEY} MACROS AND XALT KEYS

Out of the box, WordPerfect only permits ALT keys to be assigned to the 26 letters of the alphabet. To remind you, in general terms, an ALT key macro is one that can be defined by pressing one key at the **Macro Define** (CTRL-F10) prompt and is later run by pressing the same key. By creating {One-Key} Macros, you may create simulated ALT-key macros up to the number of keys recognized by WordPerfect. In addition, MPE4WP is capable of increasing the number of ALT-key macros from 26 to more than 130. The additional ALT-key macros are called XALT keys.

A {One-Key} Macro consists of two components: a Key Macro and a standard file macro. The Key Macro component of a {One-Key} Macro takes the form: **{Macro}** [Your macro's name here] **{Enter}**. Suppose, for example, you have a macro called HEADER.WPM that you would like to assign to CTRL-H. Go to the **Keyboard: Edit** menu (Keystrokes: **{Setup}**k). Position the cursor over the keyboard file you wish to modify and press **{Enter}**. Now press **C** for Create. You will be prompted for a key; press CTRL-H. WordPerfect will place you in the tedious Macro Editor, but for a small macro like this, that is tolerable. Delete the default **{Home}**

command and type **{Macro}** Header **{Enter}**. (Remember, in the Macro Editor, you do not actually type Key Commands. You press CTRL-V, then press the key associated with the Key Command.) Press **Exit** when you are finished. The resulting {One-Key} Macro will be as functional as an ALT-macro. You may either run or define the macro by pressing CTRL-H. If you rename the file macro component of a {One-Key} Macro, you must also revise the Key Macro component to reflect the name change.

A {One-Key} Macro may also specify the directory in which the macro is located. Suppose, for example, that you did not want to place the MPE4WP macros in your regular macro directory, but preferred to have them in a special directory called "C:\WP51\PROG". Simply revise the MPE4WP Key Macros to include the full path names of the macros. For example, the {One-Key} Macro associated with CTRL-C would be changed from **{Macro}** {CREATE} **{Enter}** to **{Macro}** C:\WP51\PROG\{CREATE} **{Enter}**. The other MPE4WP {One-Key} Macros would be modified in a similar fashion. Macros in this form can still be defined and run by pressing CTRL-C.

MPE4WP also allows you to more than quintuple the number of ALT-key macros. Your macros will have strange names like ALT¥.WPM, and ALT╪.WPM. Of course, there are not even enough ALT keys to accommodate these macros, so you assign them to CTRL keys, too, or any other key recognized by WordPerfect that is currently idle or might be put to better use. If you regularly use more than one WordPerfect keyboard definition, another possibility would be to have the ALT-R key perform one function on keyboard 1 and another function on keyboard 2. To accomplish this the ALT-R key on keyboard 2 would really be assigned to something like **{ALT ¢}**.

Since an XALT key calls a DOS file with the same name as the XALT key, coupled with the extension "WPM", the rules for naming XALT keys are basically the same as the rules for naming DOS files. The first three letters, of course, are always ALT. The last letter; i.e. the "key name", must be a legal character for naming a DOS file. Those rules may be summarized as follows:

You may only use ASCII characters.

DOS automatically converts all lower-case letters in filenames to upper-case. Thus you cannot have both ALTE and ALTe macros, because DOS sees them both as ALTE.WPM. While that rule is simple enough when applied to English letters, it becomes more complicated in connection with accented foreign language characters.

With one exception, all lower-case accented letters are converted to their upper-case counterparts, where one exists. The exception is "é", which

DOS converts to "E", instead of "É". In the case of the following lower-case foreign characters, there is no upper-case counterpart: â, à, á, ê, ë, è, ï, î, ì, í, ô, ò, ó, û, ù, ú, ÿ. Each of these letters is converted to its unaccented upper-case counterpart. You do not have to worry about these rules if you are creating XALT keys with MPE4WP because MC.COM correctly performs all upper-case conversions, including the conversion of é to É. MC.COM also verifies that the ALT key is associated with a legal character.

The following characters are illegal:

+ = / [ ] " : ; ,[comma] ? * \ < > |[this is not a line drawing character, but ASCII 124, the character on top of the backslash key of most keyboards], .[period], and [space].

In addition all control characters (ASCII 0 to 31) are illegal.

Be cautious in using the character ⌂ (ASCII 127) because it cannot be entered at the DOS command line. Thus you could not easily use DOS to delete the file permanently, but you could delete the file using WordPerfect's List Files command.

That leaves you with numerals, some punctuation characters, and most of the upper ASCII set including all the line drawing characters and mathematical symbols. Appendix A to your WordPerfect manual contains a list of ASCII characters. As the instructions at the top of the Appendix indicate, you can enter an ASCII character by holding down the ALT key and typing the number of the character on the numeric key pad. The character appears when you release the ALT key. I am sorry, for the purpose of creating XALT keys, you cannot use the **{n}** (ASCII number in brackets) notation. In other words, {ALT {158}} will not work.

The following example sets out the steps to create the XALT key **{ALT √}** key and assign it to CTRL-H on your keyboard.

Create a source file with only one word: the command **{ALT √}**. In other words, the screen is completely blank except for **{ALT √}** in the upper left corner. (Hold down the ALT key and type 251 on the numeric keypad to enter the "√" symbol.

Save the file as Generic WP. Name it ALTKEY√.TXT. The file length will be 8, because WordPerfect adds an end of file marker (ASCII 26) to the end. Do not worry, MC.COM knows all about ASCII 26 characters.

Use MC.COM to compile your source code. Let MC.COM name your macro ALTKEY√.WPM.

Now go into Setup, press **K** to select "Keyboard Layout", move the cursor to the keyboard file you wish to modify, and press **E** to Edit the keyboard file. Press **R** to retrieve your new macro. WordPerfect will prompt you for the key assignment; press CTRL-H. WordPerfect will then prompt you for the name of your macro. Type ALTKEY√. Remember to add the full path name, if ALTKEY√.WPM is not in your regular macro directory.

Press **Exit** to accept your new XALT macro key. You may now delete your source code ALTKEY√.TXT and the macro ALTKEY√.WPM.

CTRL-H will now act just like any other ALT key, except that the macro associated with the key will be ALT√.WPM. You may either rename one of your present macros ALT√.WPM, or use CTRL-H for defining a new macro.

There are two other ways to assign a macro to a single key, of which you should be aware. You may also create a Key Macro at the **Keyboard: Edit** screen with one of the following commands: (i) **{NEST}** [Your macro's name here]~, or (ii) **{CHAIN}** [Your macro's name here]~. These two alternatives are slightly less versatile than {One-Key} Macros. They can be run by pressing one key, but the name of the macro (and the directory, if the macro is not in your default directory) must be typed to define the macro.

Note: You may be wondering if it is possible to create artificial **{VAR n}** commands with values of "n" greater than 9. The answer is yes and no. Yes, it is possible to create a command (but not with MC.COM) that is called {VAR 25}, but no, it will not act like a variable in your macro. Sorry, I do what I can.

KEY MACROS

Unlike the WordPerfect Macro Editor, MPE4WP allows you to insert Key Macros into your macros. (WordPerfect does permit the inclusion of Key Macros when a macro is initially defined.) Frankly, this is a mixed blessing, since Key Macros will frequently cause your macro to hang. Inserting the Key Macro command is not the same as nesting or chaining to a file macro. Thus, your macro is suddenly hit with a series of keystrokes for which it is unprepared.

I regret not having been methodical enough to formulate any general rules about the use of Key Macros, but I do have this advice. Because they reside permanently in memory, Key Macros should be restricted to relatively small macros that are used frequently. In addition, Key Macros are not desirable because you must go into the **Edit Keyboard** screen to edit them.

Finally, to use a Key Macro in a macro, you must keep track of its number, which might change when you revise your keyboard. Consequently, instead of using Key Macros for large or infrequently used macros, you should use {One-Key} Macros or XALT keys. With MPE4WP it is simpler to splice the source code for a routine contained in your Key Macro into a new macro, than it is to try and make your Key Macro work in your file macro.

ANTI-OBSOLESCENCE

MC.COM and M2T.COM support all Macro and Key Commands accessible in the present release of WordPerfect 5.1. From time to time WordPerfect Corporation adds new Macro and Key Commands. For example, the **{OTHERWISE}** command was added to the August 20, 1990 interim release of WordPerfect 5.1. MC.COM and M2T.COM will, of course, be revised immediately to support new commands, whenever they are made available. In the meantime, new commands may be accessed the instant they are released by WordPerfect Corporation by using the MPE4WP commands **{MACRO CMD n}** and **{KEY CMD n}**, where "n" is the number of a Macro or Key Command. (Of course, you will also need the updated version of WordPerfect. MPE4WP can teach WordPerfect to do some new tricks, but it can't teach an old version of WordPerfect to recognize a new command.)

Every Macro and Key Command has a number as well as a name. The numbers are listed in Appendix K to the WordPerfect 5.1 manual, under the instructions for **{STEP ON}**. Any new command will also have a number, which should be included with the information accompanying the interim release. (If the number is not included in the release materials, you may obtain it by calling WordPerfect's toll free number, or by taking the steps outlined below.) To access a new Macro Command with MPE4WP, simply type **{MACRO CMD n}**, where "n" is the number of the new Macro Command. Similarly, a new Key Command would be accessed by typing **{KEY CMD n}**, where "n" is the number of the new Key Command. (Note: You can also use the **{MACRO CMD n}** and **{KEY CMD n}** terminology to access existing commands, although you probably will not want to.)

Here are the steps to take to find the number of the new Macro or Key Command without reading the literature accompanying the interim release and without calling WordPerfect Corporation. Use the Macro Editor in the new release of WordPerfect to create a short macro. The first command should be **{STEP ON}**, the second one should be the new command. Press **Exit** to leave the Macro Editor, then run the short macro. A prompt will appear at the bottom of the screen telling you the Macro or Key Command number. Alternatively, you could process a macro containing a new Macro or Key Command with M2T.COM. The resulting text file will give the number of

the new command in the **{MACRO CMD n}** or **{KEY CMD n}** format.

MACRO PROGRAMMING TIPS

The process of programming WordPerfect macros is often one of trial and error. Rather than deleting text in the course of development, I sometimes prefer to put the deleted material in a WordPerfect comment. (In this regard, I am referring to a text comment created by blocking text and pressing CTRL-F5,C.) The comment is ignored when the trial macro is saved in Generic WP format and can be easily re-converted to text if necessary or desirable. To preserve comments for use in a subsequent editing session, you must save the source file as a WordPerfect document. You could also create a macro comment (**{;}** or **{:}**), but this frequently involves deleting or disguising intervening tildes. Incidentally, a simple method for disguising tildes is to block the fragment you wish to comment off, then use **Replace** to substitute a hard tilde (**{~}**) for the conventional macro tildes. Remember to place a comment command (**{;}** or **{:}**) before the fragment and a conventional macro tilde at the end.

MPE4WP does not support macro descriptions. In general, the only time you see a description is when you call the Macro Editor. If you are going to call the Macro Editor anyway, you can insert a description then. I prefer to put a comment at the beginning of the macro describing its usage, which is generally more informative and avoids the artificial 39-character limitation for descriptions. Incidentally, MC.COM inserts a single space in the description field. This not intended as a signature but helps MC.COM use advanced features to increase compilation speed.

When creating and debugging macros, I change the WordPerfect default directory to a special directory reserved for works-in-progress. I also reserve one XALT key, **{ALT @}**, for testing macros under development. During an editing session when I use *Save Generic* (CTRL-G), I tell WordPerfect to name the work-in-progress "ALT@". Now when I run MC.COM, using *Create* (CTRL-C), and tell MC.COM the source file is "ALT@", it will automatically supply the prompt that the macro should be called "ALT@.WPM", which I accept. When *Create* returns me to WordPerfect, I hit the key to which I have assigned **{ALT @}** and it immediately runs the revised macro. Since, *Create* also has the effect of clearing the WordPerfect cache, the new version of ALT@.WPM will run immediately. This procedure takes advantage of the fact that, so long as you don't have a macro called ALT@.WPM in your macro directory, WordPerfect will look next in your default directory for the macro when you hit the XALT key **{ALT @}**. In other words, this strategy will not work if you have an ALT@.WPM in your macro directory. For your convenience, the **{ALT @}** key has been assigned to CTRL-\ on the

MPE4WP/MPE4WP-I keyboard. It will take you longer to read this paragraph than it will to save your source code and run the next iteration of your macro. (Of course, I cheated by making this a very long paragraph.)

If you have a curious nature, you may (if you haven't already) try running M2T.COM on an existing macro, then re-compiling the result with MC.COM in order to compare the re-compiled version with the original. Do not be startled to find that the length of the re-compiled version may be slightly different from that of the original. First, remember there may be a difference in the length of the descriptions. The description length of a macro compiled by MC.COM will always be one. In addition, when WordPerfect retrieves a macro with very long lines into the Macro Editor, it will sometimes add a special code that is the equivalent of a soft return. M2T.COM and MC.COM ignore these macro "soft returns" since they add nothing to the operation of the macro and were not intended to be part of the macro by the macro's author. The omission of the "soft returns" will not prevent you from later retrieving the macro into the Macro Editor.

PROGRAM NOTES

MC.COM and M2T.COM are written in Microsoft Professional Development System (PDS) BASIC 7.10, using routines contained in Crescent Software's QuickPak Professional, and linked with Crescent's P.D.Q. Crescent is dedicated to the proposition that Basic is created equal. (Don't blame them for that joke; it's mine.) Crescent believes that programs written in the BASIC language can be equivalent to those written in assembly language and C. If MC.COM and M2T.COM do not support this premise, I hope they do nothing to detract from it. The Crescent documentation has been as instructive to me as their routines have been useful.

MC.COM and M2T.COM utilize information about WordPerfect macro file structure taken from Gordon McComb's book *WordPerfect 5.1 Macros and Templates*. I have not had an opportunity to more than scan the other sections of this book, but it appears to be an excellent introduction to writing WordPerfect macros. I have also noted Gordon's commendable generosity in distributing macros and information about macros on electronic bulletin board systems.

This documentation was prepared using XyWrite, then imported into WordPerfect for formatting. I should also add that these programs were created while running under DesqView, a superior alternative to Windows.

STANDARD DISCLAIMERS

By now all personal computer users should realize that there are no warranties attached to software, except for disk errors and performance in accordance with specifications. This software is no different from any other in that respect. Accordingly, there are no warranties of merchantability or fitness for a particular purpose, nor will the author be responsible for any damages (incidental, consequential, or otherwise) in excess of the purchase price. MPE4WP has been thoroughly tested by me and I believe it will work as described. If you encounter problems, please let me know.

REGISTRATION/LICENSE/COPYRIGHT

MPE4WP is a user supported program. It is not public domain. If you are using MPE4WP for personal use and find the program is worth $15, you should pay a $15 registration fee. If you feel the program is worth more than $15, you still only pay $15. MPE4WP has deliberately been priced low to encourage registration. MPE4WP may not be used in business without a license. For information about securing a license to use MPE4WP in business, please see the section entitled "Business Use".

The source code for and all of the MPE4WP macros are copyrighted by me in their entirety. Authorized users are, however, encouraged to incorporate in their macros any useful routines they find in the MPE4WP macros. In that case, I would enjoy seeing an appropriate acknowledgement, but it is not obligatory.

How to Register

To register MPE4WP for personal use, please send your $15 check or money order to:

Michael H. Shacter
7825 Marion Lane
Bethesda, Maryland 20814-1337

In exchange, you will receive an acknowledgement and at least $15 worth of satisfaction. Please note that you will not be sent a disk, only a registration number. A combined registration for MPE4WP and my other program, MALT, is available for $25.00. If you have already registered MALT, you may register MPE4WP for an additional $10. If you absolutely must have a copy of MPE4WP on disk, please send an additional $10. Please (a) specify 5¼ or 3½ disk size, (b) include 5% sales tax if you are a Maryland resident, and (c) be prepared to wait 2 to 3 weeks. For your convenience, a

registration form is included with this package. Please see the file REGISTER.FRM.

Business Use

A license is required to use MPE4WP in business, after a reasonable trial period. Business use includes use by governmental entities, non-profit organizations, and any other use that is not strictly personal. You may obtain a license by writing to me at the above address. The license fee will depend on the number of MPE4WP users in your business. Discounts are available for multiple copies. Please feel free to submit a reasonable proposal, including any special requirements for customization you may have. Remember, if you are using MPE4WP to make money, I am too, and in this respect, my expectation of a living is superior to yours.

Other restrictions

All authorized users are granted a limited license to make copies of MPE4WP without charge, subject to the restrictions contained elsewhere in this document as well these:

(a) MPE4WP must be distributed in absolutely unmodified form, including the programs, macro files, and documentation. You may modify the MPE4WP macros for your own use, but you may not redistribute them in their modified form.

(b) For-profit use of MPE4WP without a license is prohibited.

(c) MPE4WP may not be included or bundled with any other product for any reason.

Not-for-profit user's groups, such as the Capital PC User Group, are permitted to charge a small fee for materials, handling, postage, and general overhead. No other organization is permitted to charge for distribution of copies of MPE4WP.

Electronic bulletin board system operators are encouraged to post MPE4WP on their bulletin board systems for downloading by their users, if the above conditions are met, and if no special fee is necessary to access the MPE4WP files (a general fee to access the BBS is acceptable). Likewise, all authorized users are encouraged to upload to MPE4WP to BBS meeting these requirements.