INTRODUCTION

This macros pops up a dialog from which you can pick a variety of bullets, special characters, and fractions to insert into the text. Simply load the macro, and hit a key. That's all there is to it.

There are two files that come with this package:

BULLETS.DOC   This documentation file.
BULLETS.WPM   The macro file.

Copy the files to your WP macro directory. Then, press **Alt+F10** and type "Bullets". Better yet, rename the macro to an Alt key that doesn't conflict with WordPerfect's pull-down menus. **Alt+B** would be a good choice. Then you will be able to quickly insert a bullet at any time. All it would take is two keystrokes, as opposed to flogging through various nested dialogues.

A couple of things to note:

o This macro doesn't check to see if you are somewhere other than the normal editing screen. That allows you to insert bullets and special characters into headers, footers, etc.

o The macro **does** force insert mode on, to prevent the accidental overwriting of text. It will restore the typeover state when it is finished running.

NOTES FOR MACRO PROGRAMMERS

This macro demonstrates some tricks. Note that there is only one dialog, and that it contains a total of 35 CtrlOption! controls. All 35 are active, rather than the somewhat limiting 9 or less that are usually present in WordPerfect's dialog boxes. While this is an atypical approach, it is useful in those circumstances when you don't want to hit multiple keystrokes to get the job done.

Creating more than 9 active CtrlOption controls is easy. Just use **StyNoNumber!** as the control style to remove the number on a numbered control. Then, incorporate an alpha or numeric character, preceded by a tilde (**~**) in the control's title. Thus, the following:

        DLGCONTROL(CtrlOption!;;"~B. Back & Forth";StyNoNumer!;;;;)

would appear as: **B**. Back & Forth

Then, when you hit "**B**", the dialog is exited, and the control's number is passed to the DLGCREATE variable. With this approach, you can have as many as 36 active controls available (A-Z and 0-9). The trick with numbered controls (such as CtrlOption!) is to keep track of the control's number, so that you can test the DLGCREATE variable when the dialog is exited.

When writing a macro, you can check what a control's number is by placing the following right after DLGEND: **PROMPT(result)WAIT(20).** Then, the control's number is briefly displayed when it has been selected.

The macro also makes use of arrays, which are a wonderful feature of
WordPerfect's new macro language. Note that this macro has two types of
arrays.

o One, called **chars**, has the actual bullets and special characters as
its elements. The macro inserts one of these elements into the current
text when an option is chosen from the dialog.

o The other array, called **chartext**, holds the text descriptions of those
bullets and characters. Its elements are used by the macro in the dialog
box control's titles.

A powerful feature of arrays is that each item can be checked for and
used in FORNEXT statements. If you are new to macro programming, I have
added a segment of text from the macro to demonstrate one way that
arrays can be used. A portion of the macro is listed below, followed by
an explanation.

```
// ================= Begin example
ctrlOP=CtrlOption!
sty=StyNoNumber!

DLGCREATE(result;"Bullets & Fractions";DlgNoOK!+DlgNoShadow!;;;80;)
   hpos=6 vpos=2 boxwide=40 boxtall=21
   DLGCONTROL(CtrlLabel!;;"Bullets";;hpos-3;vpos-1;boxwide;boxtall)
     x=97
     FORNEXT(i;1;19;1)
        y="~"+TOUPPER(NTOC(x))+". "
        DLGCONTROL(CtrlOption!;;y+chartext[i];sty;hpos;vpos)
        vpos=vpos+1
        x=x+1
     ENDFOR
   .    .    .
   .    .    .
   .    .    .
   .    .    .
   .    .    .
   .    .    .
DLGEND

IF(result=-1)
   GO(End)
ENDIF

TYPE(chars[result])
GO(End)
// ================= End example
```

The code that's listed above creates controls for a dialog box (options
A-S), displays them, checks for a result of action taken, and inserts
the corresponding bullet into the document. Here is how it works:

o Rather than inserting actual numbers for position and size into the DLGCONTROL commands, I prefer to use variables such as **hpos** for horizontal position, **vpos** for vertical position, etc. That way, if you have to move a group of controls, you can just edit the variables, rather than having to edit each DLGCONTROL command.

o x=97 stores WordPerfect's numeric equivalent for the "**a**" key to a variable named **x**. **NTOC(x)** converts that number to the character "**a**". Then, the **TOUPPER** command capitalizes it to "**A**". When you tack on the ". " character string, you get: "**A. **". This is assigned to a variable named **y**.

o The preceding is acted upon in a FORNEXT statement 19 times, and **x** is incremented each time, so that x = 97,98.... and y = "A. ", "B. "....

o The **DLGCONTROL(CtrlOption!;;y+chartext[i];sty;hpos;vpos)** command that is in the FORNEXT loop then uses the **y** variable 19 times as well, as part of its title.

o The **chartext[i]** is also part of the control's title. The variable **i** belongs to FORNEXT. Initially, **i=1,** and for each loop the value is incremented by 1. Now, **chartext[i]** refers to one of the items that make up the **chartext** array. On the first loop, it is **chartext[1],** which is the character string "**Small round black circle   4,3**". Therefore, on the first FORNEXT loop DLGCONTROL'S title is **y+chartext[i],** translated: "**A. Small round black circle**".

o The **TYPE(chars[result])** statement at the end of the sample also demonstrates the use of arrays. The **result** variable belongs to the DLGCREATE command, and stores the number of the control that was picked to exit the dialog. Assume that the 5th choice was picked. Then, **result**=5, and **chars[result]** is the same as **chars[5],** the 5th element of the **chars** array. Therefore, **TYPE(chars[5])** means TYPE the 5th element of the array called **chars.** This happens to be the character for the Medium round hollow circle (4,1).

To sum it up, if you have one array for text descriptions of the bullets (**chartext**), and another array for the actual bullets themselves (**chars**), *and the text elements correspond to the bullet elements*, then it is fairly easy to use them in both the creation of dialog boxes, via the DLGCONTROL command, and in action items such as TYPE.

Hope you find this brief explanation helpful.

All comments, suggestions, etc. are most welcome. Enjoy!

Onik Arian, CompuServe #70671,507