Chapter 4

Controlling Macro Flow

Links: The Chapter Name, above, moves to the next chapter. All page header links go to the contents menu. Within the chapter, Topic Titles returns here. Other Red Links are to other locations in this paper. Blue Links are to web sources.

Top	Contents	Glossary	Index	Release Notes
Chapter 1 Additional Resources	Chapter 2 Understanding Macros	Chapter 3 First Things	Chapter 4 Controlling Macro Flow	Chapter 5 Using the Dialog Editor
Chapter 6 Math Routines	Chapter 7 Date Routines	Chapter 8 DialogShow/Callbacks	Chapter 9 Macro Examples	Chapter 10 Glossary

Main Topics In This Chapter

Quit	Go	Call	Return
Label	OnError	OnNotFound	OnCancel

This short chapter describes the main commands/statements a common macro user needs to know to control the "flow" a macro takes during its operation. As noted in Chapter 2, macros generally run "lineally", top to bottom, and they do so unless you enter commands which instruct differently.

! QUIT The Quit command totally stops any/all macros from running whenever it is encountered. Usage: Type the word Quit in the macro.

! GO () The Go command tells your macro to "go to" a different label (location) within the same macro. The parameter is the name of the referenced label. Usage: Go(NextLabel).

! CALL () Call "sucks up" (sorry ... executes) the commands contained in and after the label name – the parameter – identified in the Call command. At some point after the called label, a Return command "returns" execution to the point in the macro immediately following the Call(NextLabel) command. Usage: Call(NextLabel) – however, a "Call" command is implied when merely used with the label name, so, NextLabel is the same as Call (NextLabel).

! RETURN Within a macro, the Return command concludes a Called statement/ callback/routine. When encountered, macro flow "returns to" and resumes at the exact point in a macro following the Called statement/callback/routine. The Return command need not be included within the called label (i.e., one or more labels may follow the called label if there is a need/reason to do so), but, unless Return is encountered somewhere, macro flow will never return to the point in the macro which contained the Call command. In the context of the Run (Macroname) command, in which one macro "runs" another, Return concludes the macro being run and returns macro flow to the macro immediately following the Run (Macroname) command. Return can be accompanied with optional parameter, the enumerations of which are CancelCondition!, ErrorCondition!, and NotFoundCondition!, not covered in this manual.

! LABEL () Label isn't a command but is a "place", an address, in a macro. The "name" (the parameter) of the address is contained within parentheses, as follows: Label(NextLabel). It is ordinarily used as the destination address in statements such as Go(NextLabel) or Call(NextLabel). A label name must begin with a letter but can contain letters, numbers, or "_" after the initial letter. Case is not important when naming a label or when referencing it, so, Go(NextLabel) is the same as Go(nextLabel).

! ONERROR () If a macro operation "error" is encountered when a macro is running, it ordinarily stops the macro and some sort of error message dialog will be present on-screen. OnError can be used to prevent that, particularly when you can anticipate that a user might make an error in data entry when a macro is running. Use an OnError command to direct macro flow to the name of the label identified in the OnError statement, e.g.,

OnError (NextLabel). You can have different OnError statements at different places in a macro, but if you don't reset the OnError statement at the end of a subroutine the secondary OnError statement remains effective. An OnError statement without a label-name parameter turns off all OnError statements that may be effective before that time. Usage: OnError (NextLabel). An alternative: OnError Call ().

! ONNOTFOUND () If a macro contains search routines, and if a search routine fails to produce a match, then an on-not-found condition occurs, the macro will stop, and a message dialog will appear on-screen. To prevent that, an OnNotFound statement tells the macro to go to the label identified in the OnNotFound parameter, e.g., OnNotFound (NextLabel). An alternative: OnNotFound Call().

! ONCANCEL () A macro will stop running when a cancel condition occurs unless it is preceded by an OnCancel statement. A cancel condition is not the same thing as a Quit command but includes such events as a Quit button or a Close button being clicked in a dialog. OnCancel(LabelName) tells the macro where to go if a cancel condition occurs and macro statements in that label will be executed. An alternative: OnCancel Call(), e.g., OnCancel Call (NextLabel) - but remember to use a Return command at the end of the called label.

Here's a simple example that uses most of the above commands/statements – no colored items are links but are colored/bolded for emphasis – Labels and Commands – if you want to copy this macro, give it a name, e.g., Test.wcm - color/bold/other formatting codes don't matter either way, yes or no:

```
Application (WordPerfect; "WordPerfect"; Default!; "EN")
   Label(vBegin)
OnCancel (vSure)
MessageBox(x;"Hi - this is the starting point of this macro";"This macro don't REALLY do nuttin!";OkCancel!)
If(x=2) Assert (CancelCondition!) EndIf
   // in the above, if x=2, e.g., the Cancel or Close button, a Cancel Condition is emulated
   Label(vNum)
GetString(x;"Type a real number, not a letter";"Please enter a number to test")
OnError (vNotNum) x=StrNum(x)
MessageBox(;"Great! You typed a number!";"You typed "+x)
MessageBox(y;"Do you want to search for "+x+" in the current document?";
"If you want to see if "+x+" is present in the current document, click 'Yes'";YesNo!)
If(y=6) Call(vSearch) EndIf
Call (vAgain) If(x=6) Go (vBegin) Else Quit EndIf
   Label(vAgain)
MessageBox(y;"Do you want to do the macro again?";
"If you want to start at the beginning, click Yes."; YesNo!)
If(y=6) Go(vBegin) Else Go(vEnd) EndIf Return
   Label(vSearch)
OnNotFound (vSorry) If(?DocBlank) Assert(NotFoundCondition!) EndIf
PosDocTop SearchString(x) MatchPositionAfter SearchNext
MessageBox(;"Paydirt!";"Yes, "+x+" is in the open document") Return
```

```
   Label(vSorry)
Messagebox(;"Sorry!";x+" is not in the open document") Return
   Label(vNotNum)
Messagebox(x;"Sorry, you did not enter a number";x+" is not a number. Do you want to try entering a number again?";YesNo!) If(x=6) Go(vNum) Else Go (vEnd) EndIf
   Label(vSure)
OnCancel
Messagebox (x;"Are You Sure You Want To Cancel This Macro?";
"You clicked the Cancel button in a prior message box. Are you sure you want to quit?";YesNo!)
If(x=6) Quit EndIf Go (vBegin)
   Label(vEnd)
Messagebox(;"We're done with this macro!";"It's been swell!") Quit
```

Top of Chapter 4 Chapter 5

NOTES