WOPR 2.0 Library — Evaluation Copy

A Word for Windows 2.0 Library of WordBasic functions

To

The **WOPR** Library is a collection of several dozen WordBasic routines that will help you write your own macros. If you aren't interested in making your own macros, you probably don't need the Library.

On the other hand, if you're curious about WinWord macros, this is an excellent place to kick around and take a look: dozens of real-world examples, ranging form simple one-liners to very complex monsters. You'll also find several pieces of working macros that manipulate Windows directly, through the Windows Application Programming Interface. There's a whole section in there that does Dynamic Data Exchange with EXCEL. And you'll see how the **WOPR** Library adds completely new functionality — including global variables, many Windows calls, Time and Date functions, string and number manipulation, and much more — to WordBasic.

The **WOPR Library** is part of the Word for Windowsä Office POWER Packä — **WOPR** — the world's largest WinWord add-on. **WOPR** consists of the following files:

TBEDIT.EXE — **Toolbar Editor.** Now WinWord has the best icons in the business ... in color ... and you can *draw your own!*

ENVR.DOC — **Enveloper**. The fanciest, easiest envelopes ever! Logos, bar codes, custom envelope sizes, notes, multiple addresses, all fonts. Works on any printer.

2X4.DOC — **Two-by-Four**. Print duplex (front and back), squished side-by-side, squished duplex for *four pages on each sheet of paper*. Print booklets, even or odd pages only, forward, backward, multi-section documents, macro listings, and much more.

FILENEW.DOC — **FileNew**. Manage your documents; don't let them manage you! Keep track of templates, using full descriptions. Stick documents in the right directory, first time, every time.

WOPRLIB.DOC — **WOPR Library**. The largest collection of Word for Windows macro subroutines anywhere. Dozens of routines to make it easier to build your own, custom macros — and a nice cookie jar, chock full of ideas, if you're just starting at writing your own macros.

The **LittleWOPRs**. CHARVIEW.DOC, the **Character Viewer**, tells you what codes lie behind your inscrutable characters. CLOSEALL.DOC, to **Close All Files** with one click. FILEDEL.DOC, the most reliable **File Delete**. FILELIST.DOC, which **Lists Files** in a snap. INSERTIT.DOC **Inserts** "Page X of Y", file names, much more. SUPERSUB.DOC puts **Super and Subscripts** at your fingertips. CLOCK.DOC, our classic **WOPRClock** — the most-often-used WinWord macro anywhere — takes a lickin' and keeps on tickin'. COUNT.DOC **Counts Words and Characters.** BORDER.DOC draws **Full-Page Borders**. FIND.DOC brings the most sophisticated **Find** anywhere to WinWord. COMPOSE.DOC, **Character Compose**, uses simple two-letter abbreviations to create characters like § © ® ¤ £ ¢ ä — and many more.

All of these programs are to be distributed together, as one package, known as the

§

*If you are missing any of these files, you do not have the entire **WOPR2** package: contact the person or company that sent you the files, to ensure you receive any missing pieces.*

WOPR is Shareware, the "Try It Before You Buy It" kind of software that you can take through its paces *before* you write the check. You trust us to distribute the best Word for Windows add-ons we can

produce, in their entirety, with nothing held back. We trust you to buy **WOPR** if you use it.

Here's what you'll get when you register **WOPR**:

- **The Manual.** A fancy, bound, 144 page compendium of breathless prose, indexed, ready to help guide you through every WOPRnook and cranny.

- **The programs.** No nag screens. Latest versions. Ready to install over the top of the shareware versions. And we send you the whole enchilada — source code too (except FileNew and TBEdit) — so you can poke around and change anything you like.

- **30 minutes of free telephone support** (via toll call), valid for 30 days.

- **More free support**, via mail or CompuServe. And 900-number support if you need help, like, right away, after your 30 minutes/30 days expires.

- Your very own **Enveloper Cheat Sheet**. Help stamp out smudgies on laser-printed envelopes. A classic, suitable for framing.

- Since you'll be on our mailing list, you'll be the first one on your block to hear about improvements, new products, books from the Pinecliffe International PineNuts (including the Addison-Wesley hit, *Windows Programming for Mere Mortals*,available at a book store near you) and all sorts of other neat, innovative, time-saving stuff. You'll also get several great offers, including discounts on future **WOPR** upgrades and a free CompuServe sign-up.

- Most of all, you'll receive our sincere thanks for helping keep **WOPR** alive. Our registered users made **WOPR** 2 possible. Your registration will help us continue making innovative, useful products for WinWord in particular and Windows in general. We're counting on you; our families are counting on us!

WOPR is $49.95 plus $4.50 shipping and handling, $9.50 outside North America. Site licenses (more than ten users) are available at considerable savings.

You can register right now by calling 800-OK-WINWORD (800-659-4696), or 314-965-5630. We take Mastercard or Visa, and try hard to ship within 24 hours. To register by mail, send a check (in U.S. dollars, please) to:

Pinecliffe International
Advanced Support Group
11900 Grant Place
Des Peres, Missouri USA 63131

All Pinecliffe International products are backed by a 100% no-questions-asked lifetime money back guarantee. If **WOPR** ever fails to live up to your expectations, for any reason, let us know and we'll refund your money. Immediately. Period.

So much for the commercials. On to the main program.

Acknowledgments

While the **WOPR** Library has been a couple years in the making, one great breakthrough led us to experiment with WinWord and the "outside world" of Windows: Andrew Schulman's pioneering series of articles on WordBasic in *PC Magazine*, October 15 and October 29, 1991.

Andrew's astonishing insights — which essentially elevated WordBasic to a "real" Windows programming language — led to *Windows Programming for Mere Mortals*, by ... ahem ... Woody Leonhard, Addison-Wesley, 1992. The trickier parts of the **WOPR** Library derive from work done in that book.

If you're trying to make WinWord work with Windows or other Windows applications, *WinMortals* will save you hours (maybe days!) of frustration. Guaranteed.

Thanks, Andrew, for blazing the trail. And thanks to all those acknowledged in *WinMortals* for making the book possible.

Special appreciation to Guy Gallo, James Gleick, Robert Enns, and all the WinWord Gadflies. These are the people who first realized the power of WordBasic, and who keep fighting to improve the product.

Thanks also to JC Harris for several ideas that, unfortunately, are not completely implemented in this version of WOPRLib.

Dozens of Microsoft employees have helped make **WOPR** what it is: Michel Girard, Kate Edson, Mark Myers, Laurel Lammers, Scott Kreuger, Chase Franklin, Doug Timpe, and many others. We couldn't've done it without you, folks, and we appreciate it.

A conversation with Michel Girard got us kick-started on the Date and Time functions, after we'd beat our collective heads against a wall for the longest time. Michel, this one's for you!

Finally, a tip o' the Hacker's Hat to Vince Chen, for all of his help and insight. Many parts of the **WOPR** Lib benefitted from Vince's courageous explorations out there on the bleedin' edge.

Your assurance of quality

Pinecliffe International is proud to belong to the Association of Shareware Professionals. ASP protects you, the Shareware consumer, with one of the best guarantees in the business. Here are the details:

This program is produced by a member of the Association of Shareware Professionals (ASP). ASP wants to make sure that the shareware principle works for you. If you are unable to resolve a shareware-related problem with an ASP member by contacting the member directly, ASP may be able to help. The ASP Ombudsman can help you resolve a dispute or problem with an ASP member, but does not provide technical support for members' products. Please write to the ASP Ombudsman at 545 Grover Road, Muskegon MI 49442-9427 or send a CompuServe message via Easyplex (also known as "CompuServe Mail") to ASP Ombudsman 70007,3536

This ASP service is *in addition to* Pinecliffe International's solid-gold no-questions-asked lifetime money-back guarantee.

The ASP mark is your assurance of quality, backed by a group that cares about Shareware and the people who use it. Look for the ASP logo wherever you go searching for software.

Licensing

We've heard from many folks interested in licensing parts of the **WOPR** Library. If you're an in-house or commercial developer, we want to encourage you to use these routines.

The licensing rules are quite simple.

An individual who owns **WOPR** may use the routines on their own computer(s) any way they like.

A company or organization using the routines internally must have a **WOPR** site license. If you distribute a WordBasic program using a **WOPR** Library routine to, say, 100 internal users, you should have a **WOPR** site license for 100 users. (The site license covers all of the rest of **WOPR**, too, of course, so those 100 people can use Enveloper, TBEdit, 2 X 4, WOPRClock, and all the others!) Site licenses are available by calling 800-OK-WINWORD (800-659-4696) or 314-965-5630, or writing to:

Pinecliffe International
Advanced Support Group
11900 Grant Place
Des Peres, Missouri USA 63131

If you want to distribute any of the **WOPR** Library routines as part of a commercial package, there's a small licensing fee. You must also retain the Pinecliffe International copyright notice on the **WOPR** code, and the printed documentation must include a note to that effect. For details, contact:

Woody Leonhard
Pinecliffe International
Post Office Drawer 7337
Coal Creek Canyon
Golden, Colorado USA 80403-0100

Disclaimer

Obviously, we can't guarantee that these routines are going to work in every situation. You'll have to try them and use your judgment to see if they'll work for you. The code contains all sorts of suggestions, tips, warnings and the like, and you should take a close look at what's being said.

We *can* promise two things: if you have a problem, we'll work hard to get it solved; and if you're ever dissatisfied with the **WOPR** Library, for any reason, our 100% no-questions-asked lifetime moneyback guarantee applies. Period.

Overview: Why a Library?

WordBasic — the macro language behind Word for Windows — has long suffered an undeserved inferiority complex. Since it's "just" a macro language, **real programmers** tend to think of WordBasic as a toy, suitable for, uh, automating simple, repetitive tasks. Conversely, since it's a macro "language", the **real users** who care more about putting words on paper than programming their word processor think it's too tough to be worth learning ... except for, uh, automating simple, repetitive tasks.

Caught between a conceptual rock and a hard place, WordBasic languished for a long time.

Suddenly, though, people are starting to catch on. Spurred by VisualBasic and other enormously powerful "macro" languages, the real programmers started looking, and the real users started understanding.

Buried within Word for Windows is a fully functional programming language. Yes, it's called a macro language, but that skirts the issue: WordBasic has a tremendous breadth and depth of function, it's easy to learn, easy to use, easy to mold to your needs. If you've ever put together a computer program, or written a DOS .BAT file, a WordPerfect or 1-2-3 macro, *you* can learn to program in WordBasic.

And any programming language as powerful as WordBasic deserves a Library.

The **WOPR** Library is a collection of routines that would be at home in any programming language. It includes subroutines and functions that you can call — you can use — from your own macros. It also contains hundreds and hundreds of lines of working code that you can pick apart while you're learning WordBasic: if you learn best by looking at examples, the **WOPR** Library will give you hours of ... shall we say ... learning pleasure.

WOPR Library 2.0

What Duzzit Do?

The **WOPR** Library installs itself as a global macro. You can then call any **WOPR** Library routine from one of your own macros by prefixing the call with "WOPR". For example, if you want to Shell to DOS from inside a WordBasic macro, all it takes is one line:

WOPR.Shell2DOS

If you're just curious, you can look at all of the **WOPR** Library by clicking on Tools, then Macro, then clicking once on WOPR, and clicking Edit. Kick up your feet and stay a while. Much of what you'll see is undocumented. Some of it is only documented in *WinMortals*. Lots of it adds new functionality to WordBasic.

In case you hadn't guessed, we're a bit proud of the **WOPR** Library. Here are the highlights:

Common Characters

WOPR Library lists most of the common characters you'll encounter in WordBasic: the paragraph mark, quotes, tabs, and much more. They're set up so you can copy them directly into your WordBasic programs.

Search Characters

WOPR Library lists all of the things you can search for in an EditSearch or EditFind: white space, footer codes, field markers, all sorts of things. These characters are all valid both manually (that is, you can use them when you click on Edit, then Search), and in WordBasic programs.

This listing will be useful for any WinWord power user, whether they write macros or not. Many of the characters listed do not appear in any official Microsoft documentation.

Windows Routines

WOPR Library contains an extensive collection of subroutines that call Windows directly. You can call these functions and subroutines from inside your programs, or you can copy them into your own WordBasic routines. Much of the code is derived from Chapter 4 of *Windows Programming for Mere Mortals*; if you use any of these Windows routines, you'll want to get a copy of the book — Windows has many quirks that can't be covered in a few hundred lines of code!

The Windows routines:

- Change the title of the Word for Windows window
- Return the directory from which Windows was loaded
- List the width and height of the screen in pixels
- Determine what kind of processor is running
- Show you how many Windows tasks are currently active
- List all Window titles and handles
- Give you the Windows time, for stopwatch applications
- Show Free System Resources (Windows 3.1 only)
- Return the DOS Environment string, tab delimited
- Reliably shells to DOS, regardless of what command interpreter is being used
- Turn screen updating on and off ("Echo Off")
- Allow the user to input one key at a time ("InKey$")

Echo Off may not work in some circumstances. We've had outstanding success when the amount of change going on behind the scenes is minimal — and there are no dialog boxes being tossed up — but then again we've had reports where Echo Off didn't work at all in complex situations. You'll have to

experiment to see what happens in your case — and you'll need to exercise some caution because when screen updates go south, sometimes they can't readily be brought back again. Details are in the Library.

InKey$ has its shortcomings, as well. It cannot detect the difference between upper and lower case. And it will not respond to the ancillary keys like "Ins" and "Del" and "Pause". Adding to the problem: it's slow as a snail on a Redmond winter's eve, and it can't tell a space from an Enter.

On the positive side, it recognizes all the letters and numbers, and most of the oddball keys like ` - = [ ] ; ' , . / \ . Also on the positive side, we've tried and tried to break it — type so fast that it drops a key, say — and it hasn't fallen apart yet. If you hit a 150 word per minute typist, you might have problems. But it ain't bad. Especially compared to the alternative.

Global Variables

One common complaint about WordBasic is its lack of global variables — variables that will retain their value between executions of a macro. Many **WOPR** routines use global variables, so we decided to make those routines available to you.

Here's **WOPR**'s Global repertoire:

- Write Global Variable — String
- Get Global Variable — String — optionally delete the variable
- Write Global Variable — Integer
- Get Global Variable — Integer — optionally delete the variable

If you're conversant with Windows terminology, these routines use WOPR.INI to store Private Profile strings, under the [GLOBAL] heading. That's not the most elegant way to implement global variables, but it has its advantages: among other things, the variable values persist even when restarting WinWord.

You can use this Global Variable capability to pass arrays between WordBasic routines. It isn't terribly efficient, but it works.

(By the way, **WOPR** doesn't use WIN.INI — and you shouldn't either!)

The routines are easily modified to work with your own .INI files, if you're in the mood.

Time and Date Calculations

The WinWord capabilities for Time and Date are pretty meager. And when you realize that the built-in Date$() and Time$() functions may return completely different text strings, depending on the "International" setting in the Windows Control Panel, well, it's a mess! EXCEL, on the other hand, has wonderful date and time functions — almost anything you can imagine.

Heh heh heh.

We built a little bridge to EXCEL. You can now get at *all* those fancy EXCEL date and time functions from inside Word for Windows.

In order to implement these calls, you must have EXCEL installed on your system; you must have EXCEL in your DOS Path (or you'll have to edit the macros to point them to EXCEL); and you have to have enough memory to run WinWord and EXCEL simultaneously. That's no small trick, but it's the price you'll have to pay to get at those gorgeous time and date functions.

No, it isn't fast. It's Dynamic Data Exchange — by definition, not fast. But if you need to, say, subtract one date from another, it's a whole lot better than writing a custom macro.

If you need to make many Time and Date calls, you can rewrite the **WOPR** macros to group things together, establish just one DDE link with EXCEL and thus minimize the EXCEL load times. We've even built some hooks into the macros to make bunched-up calls easier.

EXCEL works with "serial" dates and times. **Serial dates** are integers, starting at one for January 1, 1900. **Serial times** are decimal values, fractions of twenty four hours, to eight digit's precision. For example, January 1, 1987, at 12:30:00 pm is represented by 31778.52083 (because January 1, 1987 is 31,777 days after January 1, 1900). On January 1, 1987, at 12:40:00 pm, the serial date and time would be 31778.52778.

You can add and subtract serial dates and times, strip off the time (using Int()), add dates (simple addition), feed the routines compound dates and times like "10/20/51 2:15:30 PM", and much more. The EXCEL time and date routines may be the most complete anywhere. Take a look.

To calculate how many days have elapsed since October 20, 1951, for example, all it takes is one line:

```
DaysElapsed = Int(WOPR.wNow) - Int(WOPR.wDateValue("10/20/51"))
```

Have a loan maturing in 180 days? Here's how you calculate the date:

```
i = WOPR.wToday + 180
Print "Due on: "; WOPR.sMonth(i); WOPR.sDay(i); WOPR.sYear(i)
```

Ah, but what if the lender is closed Saturday and Sunday? No problem:

```
i=WOPR.wToday + 180
'Advance weekend dates to following Monday
If WOPR.sWeekDay(i) = 1 Then i = i + 1    'Sunday is WeekDay 1
If WOPR.sWeekDay(i) = 7 Then i = i + 2    'Saturday is WeekDay 7
Print "Due on: "; WOPR.sMonth(i); WOPR.sDay(i); WOPR.sYear(i)
```

Want to know the time in Perth, which is 13 hours ahead of local time?

```
i=Wopr.wNow + (13/24)  'advance the clock 13/24th of a day
Print  "Perth Time: "; WOPR.sHour(i); WOPR.sMinute(i)
```

You get the idea.

The EXCEL time routines are amazingly flexible. For example, wDay("8/11/91") gives "11", as does wDay("11-Jan"). Dates "flop over", so wDay(91,1,32) — "January 32, 1991" — is the same as wDay(91,2,1). Omitted years default to the current year. And on and on. Duplicating that kind of flexibility would take a talented WordBasic developer many months. It's all there for the pickin' in EXCEL.

Here's the EXCEL shtick, as adapted by **WOPR**:

- wNow — returns the current serial date and time
- wToday — returns today's serial date
- wDateValue(DateText$) — returns the serial date of the specified date
- wTimeValue(TimeText$) — returns the serial time of the specified time
- wDate(Year, Month, Day) — returns the serial date of the specified date
- wTime(Hour, Minute, Second) — returns the serial time of the specified time
- tYear(DateText$) — returns the year (1900-2078) of the specified date
- tMonth(DateText$) — returns the month (1-12) of the specified date
- tDay(DateText$) — returns the day (1-31) of the specified date
- tWeekDay(DateText$) — returns the day of the week (1-7) of the date

- tHour(DateText$) — returns the hour (0-23) of the specified time
- tMinute(DateText$) — returns the minute (0-59) of the specified date
- tSecond(DateText$) — returns the second (0-59) of the specified time
- tDays360(StartDateText$, EndDateText$) — number of days between the dates, based on a 360-day calendar; can go negative
- sYear(SerialNumber) — returns the year (1900-2078 of the specified serial date)
- sMonth(SerialNumber) — returns the month (1-12 of the specified serial date)
- sDay(SerialNumber) — returns the day (1-31) of the specified serial date
- sWeekDay(SerialNumber) — returns the day of the week (1-7 of the serial date)
- sHour(SerialNumber) — returns the hour (0-23) of the specified serial time
- sMinute(SerialNumber) — returns the minute (0-59) of the specified serial time
- sSecond(SerialNumber) — returns the second (0-59) of the specified serial time
- sDays360(StartSerialNumber, EndSerialNumber) — number of days between the dates, based on a 360-day calendar; can go negative

We had to rename the EXCEL functions (apologies for the strange usage of those "w", "t" and "s" prefixes; think of 'em as Hungarian Hacker's Postfix Notation) because of what appear to be obscure bugs in WordBasic that clobber references to Minutes, Time, and who knows what else. Jeeez. Where's the *Hacker's Guide* when ya need it?

Typically, the first call to one of these routines takes about 3.5 seconds on a quick 486. (Loading EXCEL and initiating a DDE conversation can take a coon's age!) But if you load EXCEL *before* executing any of these commands and let it run in the background, the time drops down to under a second per call. That's not bad at all: a custom WordBasic macro would probably take almost as much time.

Loading EXCEL is as easy as running this one-liner, perhaps in an AUTOEXEC or AUTOOPEN macro:

```
Shell "c:\excel\excel", 0
```

where "c:\excel\excel" should be replaced by the path to your copy of EXCEL. Once you have one copy of EXCEL running, with the "SHEET1.XLS" default worksheet available (that's the worksheet **WOPR** uses as a scratch pad), the **WOPR** routines are smart enough to set and break DDE connections without stopping EXCEL.

For a detailed description and pages of examples of Time and Date calls, take a look in the EXCEL Function Reference manual. **WOPR** is calling EXCEL directly; any oddities you experience in the EXCEL date and time functions will appear in these guys, too.

String Conversion

WordBasic lacks a few common string manipulation functions.

Here's what the **WOPR** collection will do:

- Swap one substring for another
- Strip non-alphanumeric characters out of a string
- Trim spaces from the left or right of a string, or both left and right
- Return the *last* location of a substring in a string
- Mirror: turn a string front-to-back

The "last location" function is similar to WordBasic's InStr function, but it starts at the end of the source string.

String Identification

WOPR can tell you a few things about your strings:

- IsAlpha lets you know if the string is all alphas
- IsNumeric does the same for numerics
- IsAlphaNumeric will tell you both
- IsUpper checks for all upper case
- IsLower checks for all lower case

All of those are pretty standard fare, missing in WordBasic.

Numeric Routines

WordBasic wasn't really designed to do a lot of arithmetic. The **WOPR** Library compensates for that, at least a bit:

- IsEven tells you if a number is even
- IsOdd does too
- Ordinal changes a number — say, 40 — to an ordinal like "40th"
- Round will round off most numbers to the specified number of decimal places
- RoundMod rounds up numbers to multiples of a base
- FormatDollar takes a number — say, 40.1 — and changes it into "$ 40.10"
- Max returns the larger of two numbers
- Min does the opposite

There are a few restrictions in these routines; check in the Library for details. Round is a very complex routine, primarily because of limitations on the size of integers. It's described fully in *WinMortals*, and if you plan on using it extensively you should consult the book for important details.

File Routines

Finally, the WOPR Library has a few functions that will help you manipulate files:

- IsDirectory tells you if you have a valid, existing directory
- IsFile does the same for files
- StripFinalBackslash gets at the redundant backslash on some directory names
- AddFinalBackslash does exactly the opposite
- MakeFileName returns a concatenated file name, with the backslash in place

IsDirectory uses an old DOS trick: it appends "\nul" to the end of the path and checks to see if a file exists with that name. Just as we went to press, we discovered that IsDirectory may not work on *some* (but not all!) LanManager 2 systems, when accessing remote drives. It's not yet clear if the problem is with WinWord, LanMan, Windows, or some combination of the three.

The StripFinalBackslash routine is necessary in any Microsoft Basic; it compensates for the odd way that MS Basics deliver Directory names.

Some Other Stuff

As we went to press, there were a few last-minute ideas that made it into the fray:

- Encrypt encrypts all macros attached to a document or template
- ShellSort is a very quick way to sort arrays

We had trouble using the Macro Encrypt macros that are available, so we made one to use internally on the **WOPR** team; figured that if we could use it, somebody else probably could too. Encrypt is a dangerous, but important, capability. Use it with care — once a macro is encrypted, you'll never get it back again.

ShellSort is the classic, adapted from Knuth, ready to use for WordBasic arrays (which start at zero; translating Knuth to start at ground zero was lots of fun — kinda like being an undergrad all over again). It's about as quick as you can get for sorting medium-sized arrays. Since WordBasic won't let you pass arrays to subroutines, you'll have to copy this routine into your macros and change the names of the variables to suit your application.

That's all she wrote. As we were finishing the WOPR beta, the WOPR Library grew so large that WinWord croaked on it: Word for Windows has strict limits on the size of a single "macro", even if that macro is a library, like this one. Rather than split WOPR Library into two libraries — and trying to explain which library does what — we cut back on some of the in-line comments, took out a couple of things that seemed to be marginally useful, wrapped it up and sent it on.

Use it in good health.

WOPR 2.0 Library ends here.

Post Office Drawer 7337
Coal Creek Canyon
Golden, Colorado USA 80403-0100

Have fun!