AutoKey

User's Manual

Table Of Contents

Introducing AutoKey

Now you can LAUGH at power failures, snicker at UAE's. Or take the buffoon who kicked your power cord to coffee, without "accidentally" dumping a cup of steamin' jo in the klutz's lap.

And even when USER ERROR rears its ugly head, potentially destroying four hours of your otherwise brilliant work, you can breathe a sigh of relief...

Because you had the foresight to run AutoKey.

Windows applications are becoming increasingly powerful, and let even neophytes do wonderful things with information. But the more complex a system, the more opportunities for things to go wrong, and Windows is no exception. Even with recent improvements in reliability, some unfortunate Windows users will rarely see the day when their system doesn't hang up. And of course no system hangs until immediately **before** you were going to save your data, or **after** you've completed an excruciatingly painstaking addition to your work.

AutoKey protects the data in every Windows application you're working with, by periodically instructing the other applications to save their data. Even if you're frequently switching between applications, AutoKey will protect the data in ALL your Windows applications.

To use AutoKey, you just tell it the applications you wish to protect, minimize it, and forget it. At the pre-defined intervals, AutoKey examines the Windows desktop, and if any of the protected applications are active, sends the keystrokes you request to all the applications, just as if you typed the keys yourself. ("Okay fine," you say, "I'll just type the keys myself, and so I don't need AutoKey.") AutoKey even protects the data in multiple running copies of a program. (Ha! Try doing **that** simultaneously! And besides, who wants to bother themselves every 10 minutes to go through every program they've touched and save what they've been doing?)

You can even tell AutoKey to send non-saving keystrokes, causing an application to perform special functions at the intervals you set up. For example, if you're a CIA analyst, tell AutoKey to tell your news reader to download the latest news from CryptoNet every hour, then automatically search for such terms as "Japanese Ministry of Trade" or "terrorist attack on New York City." Or as a stockbroker you might want to get quotes for Ice Cubes, Ltd. automatically throughout the day. And if you're a weather caster in Tornado Alley, you can have AutoKey instruct your analysis program to download satellite radar images of your area every hour.

Significant features of AutoKey include:

*   Keystroke control by way of menu commands or through keyboard accelerators[1].

*   Execution of nested menu commands, up to five levels of nesting.

*   Time, keystroke or mouse click intervals.

*   The ability to "screen out" applications if they contain untitled works.

*   A handy notification of pending autokeys[2].

* A defer keystroke, which allows you to intercept and skip a pending autokey.

This section of the AutoKey User's Manual contains some introductory information, a description of the conventions used in this document and AutoKey itself, an inventory of the AutoKey package, a list of what you need to run AutoKey and installation instructions.

Package Contents

The distribution archive of AutoKey contains the following files:

READ.ME	The generic info and late-breaking news file.
AUTOKEY.EXE	The AutoKey executable file.
AUTOLIB.DLL	The run-time support Dynamic Link Library.
AUTOKEY.INI	The initialization file, already set up to protect numerous popular Windows applications.
AUTOKEY.HLP	The help file.
MANUAL.WRI	This manual.
PACKING.LST	A list of the files in the archive.
SHR-WARE.DOC	Information about Shareware and the ASP.
VENDOR.DOC	Information and restrictions for disk vendors, distributors, user groups, and more.
SYSOP.DOC	Information for Bulletin Board System distribution.
DESCRIBE.DOC	Sample descriptions for catalogs, BBSs, etc.
WARRANTY.DOC	Important warranty information.
LICENSE.DOC	Important license and usage information.
SITELICE.DOC	Site license information and agreement.
REGISTER.DOC	The registration form.

The registered version of AutoKey contains the following additional files:

MANUAL.WRI	Windows Write version of the manual, with pictures and appendices.
MANUAL.WPW	WordPerfect for Windows version of the manual, formatted for clean printing on your own printer.
MANUAL.W4F	Word for Windows 2.0 version of the manual, also formatted for clean printing.
SETUP.EXE	The setup program.
METER.DLL	The meter control for the setup program.
SETUP.INF	The setup information file.

Of course, you also receive the printed and bound copy of this manual and free updates for a year when you register.

What You Need

AutoKey requires an IBM PC or compatible with an 80286, 80386 or 80486 microprocessor; at least 1 MB of RAM (2 MB or more suggested); and Microsoft Windows 3.0 or newer. A mouse is recommended.

AutoKey doesn't support Windows on an 8086 machine, or Windows' real mode on any other machine. Real mode doesn't allow multitasking, so AutoKey offers nothing to real mode users. (If you don't know what "real mode" is, you probably don't need to worry about it.)

Besides the above physical items, you should have a working knowledge of Microsoft Windows. (Not knowing what "real mode" is is okay.) AutoKey looks and feels like any standard Windows application, and what works in most other applications works in AutoKey. This manual doesn't go into detail about standard Windows functions like opening an application, minimizing an application, or clicking a button or check box. (Check your Microsoft Windows User's Guide for these techniques.)

How to Register

To register your copy of AutoKey, simply print the file REGISTER.DOC using your favorite word processor or the DOS `print` command, fill it out, and send it and $20.00 (US dollars) to:

Simple Software
PO Box 1465
Salt Lake City, UT 84110-1465

Registered users receive:

* The latest version of AutoKey (without the registration reminders);
* A nice printed user manual (with lasered pictures and an elegant, inexpensive cover);
* Free updates for a year after registration (at least one, usually two);
* Technical support for as long as you use AutoKey (that's *forever*).

Technical support for non-registered users is available for installation and evaluation purposes.

Technical Support

If you are a registered user, you can get free answers or educated guesses to your questions (particularly if they apply to AutoKey!), as well as timely bug fixes by contacting:

Lynn Wallace
Simple Software
PO Box 1465
Salt Lake City, UT 84110-1465
CompuServe: 70242,101   Phone: (Country code 1) 801-322-4040

For best results using the telephone, call between 5 PM and 11 PM Mountain Standard Time. An answering machine is available during other hours, Mountain Standard Time.

As stated above, technical support is available to registered users for as long as you're using AutoKey.

Manual Conventions

Phrases that appear in AutoKey windows are italicized when they're referenced in this document. Text that you type using the keyboard, or text displayed by DOS or other programs, is generally printed in the `Courier` font.

Most controls in AutoKey have keyboard mnemonics, which allow you to "click" each control with a keystroke sequence. Pressing `Alt` and the character underlined on the control's title will click the control. These mnemonics are generally not described in this manual.

Installation Instructions

If you are installing AutoKey from a floppy disk:

Insert the floppy in the drive.
Choose **File/Run** from Program Manager.
Enter `<drive>:SETUP`.

Follow the instructions as they are displayed. SETUP will automatically copy AutoKey to the directory you specify, then add it to the *Windows Applications* group if you're running Windows 3.0, or the *Startup* group if you're running Windows 3.1 or higher.

Now proceed to "To run AutoKey," below.

If you are installing AutoKey from an archive file:

Copy the archive file to the desired disk and directory.

For a .EXE file:

Simply execute the self-extracting archive:

From DOS `C:\AUTOKEY>AUTOKnn.EXE`
From Program Manager **File/Run** `C:\AUTOKEY\AUTOKnn.EXE`

(Where *nn* is the version number, if any.)

For a .ZIP or .LZH file, use the extraction process as described in the PKUNZIP or LHA documentation.

Add AutoKey to a Program Manager group (Windows 3.1 users should put AutoKey in the Startup group.) by opening up the desired group window, then:

From Program Manager **File/New** `[Program Item] C:\AUTOKEY\AUTOKEY.EXE`

(Or the actual drive and directory where you placed AutoKey.)

To run AutoKey:

At this point using both methods, you are ready to run AutoKey. Either use **File/Run** or double-click on AutoKey's icon.

AutoKey normally runs as an icon, with only the registration reminder window displayed for a short period (unless of course you're a registered user). However, if it can't find the AUTOKEY.INI file, AutoKey will open its Main Window when running, and the application list will be empty.

To make AutoKey run whenever you run Windows:

If you're running Windows 3.0, you will probably want to edit your WIN.INI file (in the Windows directory) to cause Windows to load AutoKey whenever Windows starts. For Windows 3.1, you should have a "Startup" application group. (AutoKey's disk-based SETUP program automatically puts AutoKey in that group.)

If you're still running Windows 3.0, you'll have to modify WIN.INI. Here's how. Near the top of WIN.INI you should find something similar to:

```
[windows]
load= <other programs>
run= <other programs>
```

Modify the "run=" line as follows:

```
run= <other programs> C:\AUTOKEY\AUTOKEY.EXE
```

(Be sure to specify the correct disk and directory if you installed AutoKey in a different location on your disk.)

After you've performed these setup actions, AutoKey should be doing its thing quite unobtrusively every time you use Windows

How To Delete AutoKey

If for some inconceivable reason you decide to delete AutoKey from your hard disk, here's how:

AutoKey's been written to run out of one (count 'em!) directory. You don't have to scour WIN.INI, or \WINDOWS, or \WINDOWS\SYSTEM, or... Just delete everything in the \AUTOKEY directory, or whichever directory you put AutoKey in when you installed it. Then change to the parent directory and delete the AutoKey directory with `rmdir AUTOKEY`.

If, however, you've manually moved some of AutoKey's files, then you're on your own of course.

Using AutoKey

Run AutoKey using your preferred Windows method, and double-click on its icon. Access on-line help by clicking the Help button.

The Main Window

The main window includes a system menu box, a minimize box, a window frame, the *Help* button, the *About* button, the *Quit* button, the application list, the *Add...* button, the *Configure...* button and the *Remove* button.

The system menu box and minimize box both behave as you expect. But the window frame is for display purposes only, to make the AutoKey window stand out from other items on the desktop. You can't use it to re-size the window. (Well, you **can**, but AutoKey will snap itself right back to its original shape.) The remaining items in the main window are discussed below.

The *Help* Button

As you might expect, clicking this button provides you with help. Pressing F1 produces the same effect.

The *About* Button

Clicking this button creates a dialog that contains background information and instructions for registering your copy of AutoKey. (You can also find these instructions in Help.)

The *Quit* Button

Clicking this button exits AutoKey. Alt-F4 produces the same effect, as does selecting *Close* from the system menu.

The Application List

The application list contains all the applications AutoKey will affect. For each application, it shows the intervals between autokeys in time, keystrokes and mouse clicks. If the list contains more applications than can fit in its display, a vertical scroll bar appears.

One of the applications in the list is highlighted[3]. The *Configure...* and *Remove* buttons affect this application. You can select a different application by clicking in the list or by moving the highlight bar with the arrow keys.

The *Add*... Button

Click this button to add a new application to AutoKey's control. You can add more than one copy of an application to the list, complete with its own parameters. (But you must give it a unique name.) Clicking this button creates the Add dialog box, described later under **Adding an Application**. The new application is added to the application list and highlighted when you're done.

AutoKey can control up to 64 applications!

The *Configure*... Button

This button allows you to change the autokey, intervals and special conditions for an application. Clicking this button creates the Configure dialog box, described under **Setting Up an Application**.

The *Remove* Button

This button removes the current selected application from the list. You can't retrieve it; to restore it, you must *Add...* and *Configure...* it again.

Adding an Application

Clicking the *Add...* button on AutoKey's main window causes the Add dialog to appear.

This dialog box allows you to enter the application name as it will appear in the application list, and either the application's window title or its EXE file name. In addition, there are pushbuttons to OK or Cancel the entries.

There's nothing special about the *Application Name* box, just fill it as you desire, up to thirty characters ("Word for Windows," "Windows Write" or "WordPerfect" for example). But the *Window title* and *EXE file name* both need a little explanation.

Since many applications change the title of their window depending on their state (they usually put the name of the file they're processing in the title), you can't specify every possible case. So the *Window title* box allows you to use the * (asterisk) character as a wildcard. The * can only be used at the end of a title string. For example, AutoKey comes set up for Microsoft Word for Windows, and Word's window title has been entered as "Microsoft Word*". This means that any window containing "Microsoft Word" followed by any other characters (or no other characters) will fall under AutoKey's control.

You may decide to enter the application's EXE file name instead of its window title. This is the full name, including the disk and directory path, of the application's executable file as it is stored on your system. This field appears as all capital letters. AutoKey searches for the file when you click *OK* and notifies you if it can't find the file. You can then re-check the file's name and location and re-enter it.

It takes AutoKey slightly less time to find an application by its window title than its EXE file name, so you might wish to enter the application's window title instead of its file name. Though AutoKey shouldn't create any noticeable processing overhead, entering a window title will speed it up a bit.

Clicking *OK* causes AutoKey to collect the information you've entered and perform any checks on it. The Add dialog box disappears and the new application is entered into the application list with the following default parameters:

Menu command:	the File/Save menu command.
Accelerator:	none.
Time interval:	10 minutes.
Keystroke interval :	2000 keystrokes.
Mouse click interval:	500 clicks.
5-second notification:	off.
Defer key:	none.
Window title screening:	none.
Background autokeys:	on.
Iconized autokeys:	off.

Of course, only the interval values appear in the application list. To see the other default values, click the *Configure...* button on the main window.

Clicking the *Cancel* button closes the Add dialog box and causes AutoKey to forget any data you've entered.

Removing an Application

To remove the current application from the list, click the *Remove* button on the main window. To restore the application to the list, you must add it again. This is a lot easier if you have a backup copy of AUTOKEY.INI.

Setting Up an Application

When you click the *Configure...* button on the main window, the Configure dialog box appears, allowing you to view or change the current application's parameters.

There's a lot going on here, so we'll split it up.

Setting Up a Menu Command

Clicking the *Menu Command* button creates a new dialog box.

The *Menu* box lists the name of the menu containing the command you wish AutoKey to invoke for the application, for example File, Edit, Options, etc. The *Submenu/command* boxes contain the names of menus within the previous menu, or the command within a menu.

You can use the * character as a wildcard at the end of the menu items if you wish. Don't worry about including underlines in the menu items you enter here. Also, if the menu item has the name of a menu accelerator to its right, you can ignore it too. So for "Save   Shift-F3", just enter "Save".

AutoKey comes set up with most of its target applications using the File/Save menu command, so you can get an idea of how to use this dialog by looking at the menu commands for those applications. For a more complex example, let's say you have an application with the following menu structure:

```
+-------+-------+-------------+------------------------+
|File   |Edit   |Screen       |                        |
+-------+-------+-------------+------------------------+
|               |Calculations |                        |
|               |Text         +-------------+--------+ |
|               |Graphics     |Update       |Curves  | |
|               +-------------+Scale        |Edges   | |
|                             |Change Colors|Rotate  | |
|                             +-------------+Faces   | |
|                                           |Sprites | |
|                                           +--------+ |
+------------------------------------------------------+
```

and you want AutoKey to execute the Screen/Graphics/Update/Sprites command at every interval. You would set up the Menu Command dialog box like this:

Menu:	Screen
Submenu/command:	Graphics
Submenu/command:	Update
Submenu/command:	Sprites
Submenu/command:	(empty)

The *OK* and *Cancel* buttons cause AutoKey to accept or ignore your changes, respectively. The *Test* button is useful for making sure your menu command works the way you expect. If the target application is active, clicking this button sends the menu command to the application. You can observe the results in the application window, provided it's visible, and confirm that AutoKey does what you intend.

Setting Up an Accelerator

Clicking on the *Accelerator* button in the Configure dialog creates a new dialog box.

This dialog allows you to specify a menu accelerator, a control mnemonic, or any other keystroke, for an application.

The *Accelerator Keystroke* box works differently from standard Windows edit boxes.

Only a single keystroke can fill the box at one time, and the box contains the name of the key. All keys available on your keyboard are usable in this box except Shift, Ctrl, Alt, Tab, Esc and Enter. Tab, Esc and Enter perform their standard Windows dialog functions: Tab moves the input focus to the next control (the *Clear* button in this case); Esc cancels the dialog; and Enter OK's the dialog. (You can tell AutoKey to use these keys by entering them directly into the AUTOKEY.INI file. A discussion of how to do this comes later.) The *Accelerator Keystroke* box ignores the Shift, Ctrl and Alt keys, but you can combine them with the keystroke using the check boxes in the dialog. Only one of the key's shifted states is displayed in the box; check the *Shift* box if you want AutoKey to send the key's other character.

If AutoKey can't determine the name of a key on your keyboard, "Unmapped" appears in the *Accelerator Keystroke* box when you enter the key in the box. The key should still work as expected. (Please let us know if this happens, and inform us of the model of your computer and keyboard, and which key AutoKey can't identify.)

The *Clear* button allows you to clear the accelerator. This button is here because the Delete and Backspace keys are intercepted by the *Accelerator Keystroke* box, and thus can't be used to clear it. Clicking on *Clear* clears the *Modifiers* check boxes as well as the *Accelerator Keystroke* box.

The *Modifiers* check boxes allow you to enter the Alt, Shift and Ctrl keys as modifiers to the accelerator keystroke. If you normally press one (or more) of these three keys at the same time you press the accelerator key, click the appropriate box(es).[4]

The *OK* and *Cancel* buttons cause AutoKey to accept or ignore your changes, respectively. The *Test* button works just like the Menu Command *Test* button. If the target application is active, clicking this button sends the autokey to the application. You can observe the results in the target application's window, provided it's visible, and confirm that the keystroke does what you intended.

Entering the Interval

The *Invoke Autokey* section of the Configure dialog box allows you to enter the intervals at which AutoKey will send the autokey(s) to the target application.

You specify intervals in minutes (from 1-999 minutes), keystrokes (100-9999) and mouse clicks (50-9999). You can cancel an interval check by clearing the appropriate check box. Entering 0 for an interval also cancels the interval.

Special Conditions

The *Special* section of the Configure dialog allows you to streamline AutoKey so it runs with a minimum of intrusion.

Deferring Autokeys:

There are cases where you don't **want** your file saved. If you're in the middle of a quick, concise modification to your work, or have just saved your file yourself, or don't want to be interrupted by a looong saving process, you'll want AutoKey to skip an interval for you. But just one! So isn't it nice that AutoKey provides a means of skipping a single autokey, and that it's really easy to use?

In order to defer an autokey, you've got to know when it's coming. Checking the

Beep before sending the autokey box makes AutoKey sound a mellow beep on the PC's speaker five seconds before it sends the autokey to the application. When you check this box, AutoKey instantly sounds the beep to let you know what to expect. The notification beep happens only before the autokey for the active application.

Once you know it's coming, you can cancel the autokey. Checking the *Allow me to defer it by typing   within 5 seconds* box enables the edit box containing the defer keystroke. You enter the key you want to use to abort the pending autokey in this box, which functions just like the *Accelerator Keystroke* box described earlier: a single keystroke is displayed in the box, and all keystrokes except Esc, Tab, Enter, Shift, Ctrl and Alt are valid. (And you can still use those keys to defer an autokey by entering them into the AUTOKEY.INI file by hand.)

The *Modifiers* check boxes let you combine the Shift, Ctrl and Alt keys with the defer keystroke. You can mix them as desired. For example, if you want to be able to abort an application's autokey with Shift-Ctrl-/ (three keys close together on the keyboard, and unlikely to be tied to a function in any application), you'd press the / key in the keystroke box, and check the *Shift* and *Ctrl* check boxes next to *Modifiers*. The defer key section of the dialog would look like this:

```
                               +---------+
Allow me to defer it by typing |/        |
within 5 seconds.              +---------+

Modifiers:      Alt     X Shift X Control
```

Then, when you hear the beep indicating that AutoKey is about to send a key to the application you're working in, you can press Shift-Ctrl-/, and AutoKey will ignore the application for that interval.

The defer key is only useful when combined with the notify beep, otherwise you won't know when an autokey is on the way. So if you turn off the notify beep, AutoKey clears the defer key section of the Configure dialog.

You can only defer the autokey for the application you're working in. You can't defer autokeys to applications that are open but sitting idle on the desktop.

<u>Title Screening:</u>

If you're just writing a quick note to a colleague, or doodling with your sketch program, you probably don't want AutoKey to send a File/Save to your program. Since most applications pop up their **Save As** dialog when you save an unnamed work, you have to reach **way** over to the corner of your keyboard and press Esc to cancel the dialog before continuing. Fortunately, most applications have a standard "name" they give to untitled works. In fact, in some cases, it really is "untitled".

The *Don't send autokey if window title contains*   box allows you to enter part of a window title that AutoKey looks for before it sends a key to the application. If the text appears anywhere in the application's window title, AutoKey won't send the key. Don't use the wildcard (*) here, just enter the text you want AutoKey to look for in the application's title.

If you have had problems with inadvertently killing untitled works that you later wanted to use by mistakenly exiting a program without saving the work, you may wish to ignore this feature. Thus, when AutoKey sends a save command to an application, the application will probably prompt you for the file name, which may disrupt your work but help keep you "honest".

As examples of this feature, Microsoft Word uses "Document1", "Document2", etc., as its generic title of documents that haven't been named. So AutoKey comes pre-set for Microsoft Word with "Document" as the text to check for in Word's title. When an autokey is due for Word, and "Document" appears in Word's window title, AutoKey will skip Word for the current interval. Nearly every pre-set target application uses this feature.

<u>Other Streamlining</u>

The remaining check boxes allow you to tell AutoKey when to send keys depending on the protected application's state.

Don't send autokey if window is in background applies to applications that are open but inactive. If you've started an application, but are currently working in another, and you've checked this box for the first application, AutoKey will skip that application for the current interval.

Send autokey even if iconized applies to applications that you've opened, but that you've minimized. If you've done work in an application but have minimized it to free up space on the Windows desktop, AutoKey won't send the key unless you've checked this box for the application.

The OK Button

Clicking *OK* on the Configure dialog box causes AutoKey to save all the information you entered. Any changes in the intervals between autokeys will show up in the application list.

The Cancel Button

Clicking the *Cancel* button on the Configure dialog box causes AutoKey to forget all your changes. The setup for the application you were modifying will be unchanged.

Turn It Loose!

When you finish setting up all the applications you want AutoKey to affect, minimize it. Then continue with your other work. (AutoKey won't send any keystrokes if its window is open.) If AutoKey has been installed in the Startup group, or is mentioned in WIN.INI in the proper place, it'll be running every time you start Windows.

Using Help

Help is available by clicking the *Help* button. (We considered placing Help somewhere other than the *Help* button, but then couldn't figure out what to use the *Help* button for...) Most of the information in this manual appears in Help. The first thing you see is an introduction, and you can then click *Index* or *Contents* for the Help Index. You can jump to associated topics by clicking on the highlighted text, and you can view the help from start to finish using the *Browse* or « and » buttons. Dotted-underlined words in Help have definitions associated with them. Clicking the word or phrase displays the definition.

Usage Notes

DOS Applications

Does AutoKey work with DOS programs running under Windows? No, but it does work around them. DOS applications are a special case in Windows. Windows creates a "virtual 8086" for DOS programs, letting them pretend they're running on a dedicated PC. In doing so, it turns off much of its normal processing, including the standard keyboard and mouse interface. So AutoKey can't talk to DOS applications. (It might barely be possible to solve this problem, but by the time this feature gets implemented, DOS will be extinct. If we're lucky.)

This also means that Windows can't let AutoKey know about keystrokes and mouse clicks that you're making inside a DOS program, so keystroke and mouse click intervals that you've set up for target applications don't take into account work you're doing in DOS programs.

If you're running a DOS program in a window, AutoKey runs normally as far as other Windows applications are concerned, except for the problem with keystrokes and mouse clicks mentioned above. Timed intervals work normally.

If you're running a DOS program full-screen, Windows' timers slow down or even stop, depending on the multitasking conditions you've set up for the DOS program. This has the following effect on AutoKey. If the DOS program isn't running "Exclusive" (meaning it isn't hogging the entire microprocessor), AutoKey runs slowly, depending on system load. If the DOS program is running "Exclusive" (meaning Windows shuts all of itself down except for its task-switching keys), Windows, including AutoKey, goes to sleep until you switch out of the DOS program using Alt-Tab, Alt-Esc, Ctrl-Esc, Alt-Space, or Alt-Enter.

Screen Blankers

So what if you leave your PC on overnight, with AutoKey running? Will it still send its autokeys, doing a lot of harmless but useless work? It will, unless you have a good screen-blanker running at the same time.

Some screen blankers notify Windows when they are shutting down the display, and AutoKey watches for this. When such a program takes over the display, AutoKey deactivates its timers, and reactivates them when the system wakes up.

Another concern: When AutoKey "presses" a key or two, does that mean your screen blanker resets its timer and never blanks the screen? Nope. Most screen blankers for Windows look for keystrokes at different points and from different sources than AutoKey uses, so they are blissfully ignorant of AutoKey's activities.

What if I'm in the middle of something and AutoKey tries to send a key?

You don't want AutoKey to step on you by typing keys for you while you're in the middle of typing a file name in a dialog, for example. So before it sends a key to an application, AutoKey checks to see if the application is ready for it, and if it isn't, waits for five seconds before checking again. When you finish your operation, AutoKey sends the autokey to the application.

But my program already has an auto-save!

Ah, but does it protect you from yourself? Most programs with an auto-save feature save their work in a temporary file(s), and delete the temporary file when you exit the program. AutoKey works differently from such programs. Your file is saved in the original file, which you created. There's hardly ever a real difference between the schemes, and the first approach even allows you to tinker with a work while preserving a reasonably complete version on disk.

So let's say you've been working on a document all morning, using WordPerfect for Windows. A co-worker pops into your office and invites you to lunch with the office group. You say, "Okay, just a sec." They're walking out the door, obviously tarrying for your benefit, and you rush to join them, "finishing" your work before you go. Because you're an experienced WordPerfect user, you enter "ExitNoSaveYesI'mSure" without thinking. The nerve impulses race down your arm to your fingers, followed immediately by frantic **STOP!!!** signals from your brain... (This has actually happened to someone near and dear to the author.)

So, AutoKey **is** worth running against all such programs.

Where does AutoKey store its data?

When you change the application list, AutoKey updates the AUTOKEY.INI file in the AutoKey directory (usually C:\AUTOKEY). This is an ASCII file, so you can modify it using a text editor or word processor. The format of the AUTOKEY.INI file is described later.

If something goes wrong, and you believe AutoKey's application data is bad, simply restore your backup AUTOKEY.INI file (You **do** make backups, don't you?), or pull out your hardcopy and re-enter the data by hand.

Once you get familiar with AutoKey, you might find it easier to make changes directly to the AUTOKEY.INI file. (It's easy to set up uniform parameters for a number of target applications this way.) You can also edit window titles or EXE file names in AUTOKEY.INI; you can't change them using AutoKey's dialog boxes.

If you do edit AUTOKEY.INI directly, make sure AutoKey isn't running at the time, or it might over-write your changes, or more likely, cause a file sharing violation.

Problems and Limitations

If you specify a defer keystroke that creates a printed character, that character will show up in your application when you defer an autokey. For example, using the Z key to defer an autokey for Microsoft Word isn't a good idea - every time you press Z after hearing AutoKey's beep you'll have to delete the "z" from your text.

Some applications don't follow standard procedures when processing accelerator keystrokes. For these applications (Word for Windows 2.0 is one), you'll have to specify a menu command.

Word for Windows has been known to run out of memory after many File/Save commands. This is likely to be a problem only if you run the program for days at a time without exiting.

If you specify an accelerator keystroke in the AUTOKEY.INI file and it doesn't seem to work, don't give up. Try using AutoKey's user interface (the *Accelerator Keystroke* dialog) to enter the key. This will avoid any machine-dependent translation errors, since the dialog box copies the key codes directly from your keypress. If this doesn't work, you'll have to use a menu command instead.

Ventura Publisher takes a non-standard approach to everything, including its internal window, menu and keyboard processing. Don't expect to run AutoKey successfully with Publisher, but with enough tinkering you might get some good results.

Modification History

Version 1.0. March 1992. Initial release on CompuServe and a few other places.
Version 2.0. September 1992. Fancied up the Main Window, added a .INI file capability, improved help file access (don't have to put AutoKey in \AUTOKEY directory to get help), removed the redundant Turn On/Off button. Released lots of places.

Things to look for in the future:

* Much cleaner user interface. No more typing; AutoKey will learn what you want it to do by recording your actions.

* **More** glitz.

* Complete application configuration through the user interface. You should be able to edit the application name, window title or EXE file name without disturbing other parameters. This is a really stupid limitation, and it'll go pretty soon. (You can get around this now by directly editing the AUTOKEY.INI file.)

* Shortcut keystroke macros. "Schiz" magically becomes "Schizophrenia", etc. The

advantage of having AutoKey do this is it would happen in every application you use, without requiring a macro for each application. Psychiatrists, theoretical physicists and General Accounting Office employees should appreciate this one.

* Enhanced interval control. Instead of every *x* minutes, you could dispatch autokeys once a day, every hour, once a year, after certain keys are typed, when intruders appear at the front door, etc.

* Auto-pilot mode. AutoKey will sit in the background and watch what you're doing. When it sees you telling an application (**any** application) to "File/Save", it'll remember that program, and automagically repeat that File/Save at some master interval.

* Expanded application control. Instead of simple keystrokes or menu commands, AutoKey could become a full-fledged macro recorder, and it could also execute other applications or terminate them at user-specified times.

* (Read with an Austrian accent:) "AutoKey becomes self-aware at 4:23:34 PM on August 29, 1996. In the ensuing panic, they try to pull the plug..."

Let us know if any of these enhancements trigger your salivary glands. And if you have other suggestions, we're all ears!

Appendix - Glossary

Auto-save program. An auto-save program helps you preserve your work by automatically saving your data to disk at certain intervals. It's not the same as a backup program, which periodically backs up existing files on your disk to other media. Backup programs don't protect data in memory, just that on disk.

autokey (vs. AutoKey). "autokey" refers to the menu command or accelerator that is sent to target applications. "AutoKey" is the name of the program.

Default application configuration.   This is the set of parameters given to a new target application that you've just added to AutoKey. This setup should work well for common Windows applications. This is the default target application configuration:

Menu Command:	File/Save
Accelerator:	none
Time interval:	10 minutes
Keystroke interval:	2000 keystrokes
Mouse click interval:	500 clicks
5-second notification:	off
Defer key:	none
Window title screening:	none
Background autokeys:	on
Iconized autokeys:	off

Defer keystroke. This is the keystroke you use to skip an impending autokey. AutoKey can beep five seconds before it sends the autokey to a target application, allowing you to defer it by pressing the key(s) in this sequence. This keystroke can include Shift, Ctrl and/or Alt. You define this keystroke in the Configure dialog box.

EXE file name. This is the name of a target application's executable file. AutoKey uses this to determine if a target application is active, and to send the autokey to the application. See also Window title.

Menu command. This is a command in the target application's menu which AutoKey invokes at the proper interval(s).

Menu accelerator. This is a shortcut keystroke in the target application's menu which AutoKey invokes at the proper interval(s). Menu accelerators usually appear to the right of a menu command in the target application's menu.

Mnemonic. In Windows, this word refers to accelerator keystrokes that apply to window controls instead of menu items. They function identically.

Nested menus. Some Windows applications have complex menus. To organize their menus, they may "nest" menus by associating another menu with a menu selection. Selecting such a menu item will create a new menu instead of invoking a command, and the new menu may contain more menus as well as commands. AutoKey can invoke commands in nested menus.

Target application. This term refers to another Windows application which AutoKey can control. In general a target application can be any Windows application other than AutoKey. AutoKey target applications appear in the application list.

Window title. This is the title of a target application's window. AutoKey uses this to determine if a target application is active, and to send the autokey to the application. Because many Windows applications change their window titles, you can use the * character as a wildcard at the end of the window title. See also EXE file name.

Association of Shareware Professionals

The Association of Shareware Professionals (ASP) is an association for shareware authors with the general goals of educating shareware authors and distributors and the public, setting standards, sharing resources and information among members.

AutoKey's author is a member of the Association of Shareware Professionals. 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 or send a CompuServe message via CompuServe Mail to ASP Ombudsman 70007,3536.

FOOTNOTES

[1] A keyboard accelerator can affect a menu item or a dialog box control. Accelerators that invoke menu commands are called "menu accelerators" and acclerators that click or select a control are called "control mnemonics" (pronounced "neMONic"). Typically, a menu accelerator is listed to the right of the menu item which it invokes, and a control mnemonic is entered by pressing Alt and the character underlined on the control's label.
[2] The term "autokey" with all lower-case letters is what we use to denote either a menu command, a menu accelerator or a control mnemonic.
[3] We call the highlighted application the "current selected application".
[4] A note about modified keystrokes: To send a Shift, Alt or Ctrl keystroke to another application, AutoKey must "fool" Windows into acting as if the Shift, Alt and/or Ctrl key is pressed. This means that other applications "see" these keys being "pressed" also. So if you're in the middle of typing a line of text in a text editor, and AutoKey sends a menu

accelerator to another application, the editor may be fooled into acting as if you've pressed Shift, Alt or Ctrl, and your text may be affected. You should set up an application that uses a modified menu accelerator to only receive its autokey when it's active. You do this by checking the *Don't send autokey if window is in background* box in the Configuration dialog. Unmodified menu accelerators (keystrokes that don't involve the Shift, Alt or Ctrl keys) should cause no problems.