WinTask

The ultimate automation tool for Windows 9x, ME, Windows NT, 2000 and XP

Tutorial

INTRODUCTION 5

CHAPTER I : PRINCIPLES 7

CHAPTER II : INSTALLATION 9

CHAPTER III : FIRST SCRIPT, AUTOMATE WORDPAD 13

Creating the script with the Recording Mode 13

Running the script 15

Commands generated by the Recording Mode 17

CHAPTER IV : SYNCHRONIZE WINTASK WITH THE APPLICATION 19

CHAPTER V : SYNCHRONIZE WINTASK WITH A SELECTION IN A MENU 25

CHAPTER VI : AUTOMATE SEVERAL PIECES OF SOFTWARE 31

Commands generated by the Recording Mode 35

CHAPTER VII : HOW TO FIND WINDOW NAMES 37

More information on a Web page 40

CHAPTER VIII : SYNCHRONIZATION WITH A TEXT DISPLAYED ON THE SCREEN 43

CHAPTER IX : SYNCHRONIZATION WITH A BITMAP DISPLAYED ON THE SCREEN AND CLICK A BITMAP 48

CHAPTER X : SYNCHRONIZATION WITH A NEW WINDOW DISPLAYED ON THE SCREEN 54

CHAPTER XI : HOW TO RECORD NAVIGATION WITHIN WEB PAGES 58

How to measure response times : 59

How to detect Page not found : 60

CHAPTER XII : LAUNCH THE SAME SCRIPT EVERY WEEK 62

Include the script launch in the Startup group 67

CHAPTER XIII : AUTOMATE IN A WINDOWS DOS BOX 72

CHAPTER XIV : TIPS **76**

How can I stop a script ? **76**

© 1997-2002 Taskware May 2002 2.42
All rights reserved.

INTRODUCTION

WinTask is a task automation tool ; you can automate all the things you do in Windows 98, Windows Millenium Edition, Windows NT 4, Windows 2000 or Windows XP.

WinTask is a powerful, flexible platform for building automation solutions on a stand-alone or networked PC.

You can, for instance, launch an accountancy package, integrate data from other programs, calculate results with this new data and then print them.

You can also launch a terminal emulation package, type the userid and the password, navigate through the host screens, pick up data in different screens and paste them back on your PC.

You can create an installation script to automatically install software on thousands of PC's. WinTask can provide all the user information required to install software without any user interaction.

To summarize : all the tasks you manually do on your PC every day, week, or month can be automated using WinTask.

CHAPTER I : PRINCIPLES

To create an automation script, you have to :

1. Launch WinTask.

2. From the **Start** menu, select **Recording**, or click the

button. You can start an application before recording then the Edit module is minimized and all your actions are recorded to create the script.

3. Do all the actions you want to record, such as running software, keystrokes, selections in menus, etc…

4. To stop the recording, click the flashing icon.

5. Review the script which has been automatically generated.

Once your script has been generated, you can launch it immediately, or you can launch it later whenever you want to automate a task instead of doing it yourself. The Wintask Compiler will transform the source script into an executable script.

The commands generated in the WinTask Edit module are written in a Visual Basic-like language. In the script, you can modify, add or delete commands. You can improve the generated script by adding conditions, loops, etc…

Some notions are specific to an automation language.

For instance, during execution of a script, WinTask must know to which window the recorded actions must be sent ; WinTask uses for that a specific command : UseWindow. Those UseWindow commands are automatically generated when you record your actions but a WinTask specific module, **Spy**, can help you too to know the window names of each individual window on the desktop.

WinTask must simulate how a user interacts with his PC : the human user makes a new action only if the latest action has been finished and done correctly. WinTask includes synchronization tools in order to make the script wait until the latest action is completely done. A **Wizard** can help you to define such synchronizations.

The following chapters will help you in creating your first automated script.

CHAPTER II : INSTALLATION

The installation file « *SETUP.EXE* » is located on the first disk. Insert the first disk into the A drive.

From the **Start** menu, click **Run**. In the dialog box, type « *A:SETUP* » as shown below :

Click **OK**. InstallShield will display the Welcome screen; click **Next**.

The User Information window is displayed showing your name and your company name. (If the installation software does not find this information on your PC, it should be entered manually.) Click **Next**.

The window **Choose Destination Location** is displayed :

The default destination directory is « *C:\Program Files\WinTask* ». If this is correct, click **Next** ; if not, click **Browse** to modify the destination directory.

The window **Setup Type** is displayed ; the default is « *Typical* » :

Click **Next** if you agree with this default value ; otherwise select another Setup type before clicking **Next**.

The window **Select Program Folder** is displayed ; the default **Program Folder** is « *WinTask* » :

Click **Next** if you agree with this default value ; otherwise type the new program folder name before clicking **Next**.

Files are copied onto your hard drive ; you are then asked to put the second disk into your disk drive, the third disk and then the new program folder is created.

The next window **Setup Complete** is displayed ; click **Finish**.

WinTask is now installed and you can proceed with the exercises described in this manual.

CHAPTER III : First script, AUTOMATE WORDPAD

In this chapter, you will create your first automation script, using Windows WordPad word processor.

WordPad will be already opened, the script will type some lines inside WordPad and then close the software.

Creating the script with the Recording Mode

Close all other open applications and from Windows desktop, launch WordPad :

- Click **Start**.
- Select **Run**.
- In the dialog box **Run**, type *« wordpad »* as shown below :

And **OK**.

The WordPad window is opened without any text inside it.

Launch WinTask Editor by clicking **Start**, then select **Programs**, the group **WinTask** and then select the program **WinTask** (**Start/Programs/Wintask/Wintask)**. The Editor window with the title *« Untitled1 »* is displayed, as shown below :

To start the Recording Mode and record your keystrokes in the Editor window, select **Start** and then **Recording,** or just click the

button in the toolbar. If the dialog box below is displayed, click **No** as the application WordPad has been already launched.

The Editor window disappears and a small icon of a blinking video camera is displayed in the Windows taskbar on the righthand side. This icon reminds you that all your actions are being recorded.

- The focus is now on the WordPad window. Type, for instance :

« That's a demonstration of WinTask. »

- Select the whole text using the mouse and select in the WordPad menu **Format/Font/12**. Click **OK** ; the selected text is now displayed in larger characters.

- Quit WordPad by selecting **File,** then **Exit**. Answer **No** to the question *« Save changes to document? »* The WordPad window is now closed.

Stop the Recording Mode by clicking once on the blinking video camera icon in the taskbar. The Editor window is displayed with all your actions recorded in the script. You should have a script that looks like this:

```
UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
      SendKeys("That's a demonstration of WinTask")

UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
      ClickMouse(Left,Down,5,11)
      ClickMouse(Left,Up,5,11)

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
```

```
      ChooseMenu(Normal,"F&ormat|&Font...")

UseWindow("WORDPAD.EXE|#32770|Font",1)
      ChooseItem(Combo,"3","12",single,left)
      Click(Button,"OK")

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")

UseWindow("WORDPAD.EXE|#32770|WordPad",1)
      Click(Button,"&No")
```

Before explaining all these commands, let's first run this script.

Running the script

Before running the script, the conditions on your desktop must be the same as when you started the Recording Mode. So in this example, you must open WordPad with no text inside the Wordpad window.

Click **Start** and then select **Run.** Type *« WordPad »* and then click **OK**.

Come back to the Editor window where the script *« Untitled1 »* is :

In the **Start** menu, click **Run** to launch the script (or in the toolbar, click the

button).

The **Save as** window is displayed because the script does not have a name yet ; save it as *« Script1 »* (noting that it is not necessary to type the extension *« .src »*) as shown below :

Click **Save**. The source script is now compiled and saved.

The next window shows that there are no compilation errors :

Click **Run** to launch the script.

In the WordPad window, you will see the text being typed automatically. Then the text will be selected and then the font changed. When the script is finished, the WordPad window will be closed.

Maximize the Editor window where the script *« Script1 »* is displayed. We will now explain the commands inserted in that script.

Commands generated by the Recording Mode

The script starts with a *UseWindow* command :

```
UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
```

You will notice a lot of commands like this in a WinTask script. This command tells the program which window to send its actions. The parameters for this command are the window name, its class and the window title. The last number behind the comma is the instance number (this instance number would be different if WordPad had been launched more than once : several Wordpad instances would therefore be on the desktop).

The next command, *SendKeys*, simulates the user typing text inside the window specified by the previous *UseWindow* command :

```
      SendKeys("That's a demonstration of WinTask")
```

The text is put inside quotation marks. Keywords for special keys are surrounded by less than and greater than signs (as in *<Enter>* , for instance).

The next two commands automate mouse clicks :

```
      ClickMouse(Left,Down,5,11)
      ClickMouse(Left,Up,5,11)
```

The left mouse button (parameters *« Left Down »*) has been pressed in position 5,11. Then the left mouse button has been released (parameters *« Left Up »*) in position 5,11 (relative to the WordPad window).

The next two commands automate a selection in a menu :

```
UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"F&ormat|&Font...")
```

First, WinTask specifies from which window the selection in a menu must be made, using the *UseWindow* command. Then the command *ChooseMenu* automates the selection of the option *Format* and then the option *Font*. The ampersand (&) sign just in front of a character signifies that this character is the shortcut for this option.

The next three commands automate the selection of the font size. As usual, the *UseWindow* command specifies the window where the font size is selected.

```
UseWindow("WORDPAD.EXE|#32770|Font",1)
      ChooseItem(Combo,"3","12",single,left)
      Click(Button,"OK")
```

The *ChooseItem* command allows the selection of an item in a ComboBox ; in the above parameters, the third Combobox of the Font window is used and the number *« 12 »* is selected by a single left mouse click.

The *Click (Button, "OK")* command automates a mouse-button click. The button name is between the quotation marks .

In the last four lines of the script, the menu selection and button click commands are used again :

```
UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")

UseWindow("WORDPAD.EXE|#32770|WordPad",1)
      Click(Button,"&No")
```

Close all open applications before proceeding to the next chapter.

CHAPTER IV : SYNCHRONIZE WINTASK WITH THE APPLICATION

In order to behave like a human operator, WinTask must be able to wait until a certain result is obtained before going onto the next task.

For instance, in WordPad, a second sentence must be typed by WinTask only after someone presses the **Enter** key.

In the example below, we will make the automate wait for 10 seconds after the sentence *« That's a demonstration of WinTask »* has been typed in WordPad and after **Enter** has been pressed. For this we will use a tool included in WinTask : the Synchronization Wizard. It will help you insert a pause in your script when necessary.

Before starting this exercise, verify that no other application is open in the Windows desktop.
From Windows desktop, launch WorPad :

Click **Start**.

Select **Run**.

In the **Run** dialog box, type *« WordPad »* as shown below :

Click **OK**. The WordPad window is opened without any text in it.

Lauch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main WinTask window is displayed with the title *« Untitled1 »* as shown below :

You want the Synchronization Wizard to be displayed during recording every time **Enter** is pressed. To do this, select **Configure** in the top WinTask menu, then choose **Wizard** as shown below :

The **Configure Wizard** window is displayed :

Verify that the radiobutton **Beginner** is checked in the frame *« Select Wizard mode »* . Uncheck the checkbox **Inactivity** in the frame *« Beginner mode »* ; then check **Keys** and click **Key list** to display the following window :

Using the mouse, scroll down the list of available keys until you locate **Enter** and select it. Click **Add**. Quit this window by clicking **OK**. Click **OK** a second time to quit the **Configure Wizard** dialog box.

With this configuration, the Synchronization Wizard will appear each time **Enter** is pressed. The Synchronization Wizard will, for instance, create a pause until specified text appears in a certain window or until a specified action by the user (such as a mouse click or a menu selection).

In the next example, we will ask for a pause of 10 seconds after **Enter** has been pressed.

Start the Recording Mode of WinTask by selecting **Start/Recording** in the WinTask main menu, or click the

icon in the toolbar. If the dialog box below is displayed, click **No** as the application WordPad has been already launched.

The Editor window disappears and a small icon of the blinking video camera is displayed on the righthand side of the Windows taskbar. This

icon reminds you that all your actions are being recorded.

The focus is now on the WordPad window. Type :

« Repeat this sentence every 10 seconds. »

Press **Enter**.

The **Synchronization wizard** window appears :

In order to insert a 10 second pause in the script, select **a time delay** in the frame *« Or wait for… ? »* and click **Next.**

The dialog box **Time delay Synchronization** is displayed :

The default time delay value is 10 seconds and it is accepted by clicking **Resume recording**.

Stop the Recording Mode by clicking twice on the blinking icon. The WinTask main window is displayed :

```
UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
      SendKeys("Repeat this sentence every 10 seconds<Enter>")

      Pause 10 secs
```

The Synchronization Wizard has written the command «`Pause 10 secs`» which will make the script wait 10 seconds before going on.

In the Editor window, press **Enter** to move to the next line. Start the Recording Mode by selecting **Recording** in the **Start** menu. As usual, the blinking video icon is displayed on the righthand side of the taskbar.

Using the mouse, select (highlight) the sentence in the WordPad window you just typed *(« Repeat this sentence every 10 seconds. »*). In the **Edit** menu, select **Copy** ; move the cursor beneath the text. Then in the **Edit** menu again, select **Paste**.

Stop the Recording Mode by clicking the small blinking video icon. The WinTask main window is opened and the whole script is displayed.

In order to lauch the script with the same conditions we started with, close the WordPad application without saving the document.

Start WordPad again and in the WinTask main window, launch the script by selecting **Run** in the **Start** menu. The window **Save as** is displayed. Give the name *« Pause »* to the script and click **Run** to run the script.

You can see the automate type the first sentence, wait for 10 seconds and then do the **Copy/Paste** routine.

Quit WordPad and WinTask before proceeding to the next chapter.

CHAPTER V : SYNCHRONIZE WINTASK WITH A SELECTION IN A MENU

In this chapter, you will see how WinTask is able to wait for a specified selection in a menu.

WinTask will type a sentence in WordPad, wait until the user has selected the option **Undo** in the **Edit** menu of WordPad and then type a second sentence. The Synchronization Wizard will be used to create the different commands.

From Windows desktop, without any other application opened, launch WordPad :

Click **Start**.

Select **Run**.

In the dialog box **Run**, type *« wordpad »* as shown below :

Click **OK**. The WordPad window is opened with no text inside it.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The WinTask main window named *« Untitled1 »* is displayed as shown below :

While recording your actions, configure WinTask so that the Synchronization Wizard is displayed as soon a you press **Enter**.

Select the option **Configure/Wizard** as shown below in WinTask main window :

The **Configure Wizard** dialog box is displayed :

Check that your dialog box has the same parameters as those shown above and click **Key list** :

After you have completed the previous exercise, the **List** window should look like the screen above. Now click **OK**.

If no keys are listed in the **Selected keys** list, then use the mouse to scroll down the list until you find *<Enter>* :

Select it and then click **Add**.

Quit this window by clicking **OK.** Click **OK** a second time in order to quit the **Configure Wizard** dialog box.

With this configuration, the Synchronization Wizard will appear each time the **Enter** key is pressed. It will create a pause until specified text appears in a certain window or a specified action by the user (such as a mouse click or a menu selection).

After typing the first sentence, we will insert a pause in the script until the user has selected the option **Undo** in the **Edit** menu of the main WordPad window.

Start the Recording Mode by selecting **Start/Recording** in the WinTask main window or click the

button in the toolbar. If the dialog box below is displayed, click **No** as the application WordPad has been already launched.

The **Editor** window disappears and a small icon of a blinking video camera is displayed on the righthand side of the Windows taskbar. This

icon reminds you that all your actions are being recorded.

The focus is now on the WordPad window. Type :

« That's a demonstration of WinTask. »

And press **Enter**.

The **Synchronization wizard** window is displayed :

In order to insert a pause for a user action (in our example, a pause for a selection in a menu), click the radiobutton **a user action** in the frame *« Or wait for… ? »* and click **Next**.

The dialog box **Synchronization on a User action** is displayed, click the radiobutton **a menu selection** :

Click **Next**.

The small icon displayed on the righthand side of the Windows taskbar changes in order to tell you to select an option in a menu. Select **Edit/Undo**.

The definition of the Wait for menu is then displayed :

In the frame *« Item chosen »* , you can see the menu selection ; the shortcuts are preceeded by a & (for instance, Undo option is written &Undo).

Radio buttons in the frame *« In »* means in our example that the wait for menu selection Edit/Undo will be done only in the WordPad window.

Click **Resume recording** to go on.

Type for instance in WordPad :

« Second sentence after the wait for menu selection »

Stop the Recording mode by clicking once on the blinking video camera icon in the taskbar.

The editor window is displayed and you should have a script that looks like :

```
UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
      SendKeys("That's a demonstration of WinTask<Enter>")

      Pause until
            Menu("&Edit|&Undo Ctrl+Z")
            InWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      PauseFalse
            MsgBox("Wait for at line " + #ErrorLine$ + " has failed
!",16,"Runtime error")
            End
      EndPause
```

```
UseWindow("WORDPAD.EXE|RICHEDIT|Document - WordPad|1",1)
      SendKeys("Second sentence after the wait for menu selection")
```

We will now run the script.

In order to launch the script with the same conditions as when we started, close WordPad without saving the document and open it again (DO NOT open a new WordPad window without closing the first one!)

When WordPad is opened, put the focus on the WinTask main window and launch the script by selecting **Run** in the **Start** menu.

Because the script has not been saved, the **Save as** window is displayed :

Type the name *« Script2 »* and click **Save**.

There should be no compilation errors. Click **Run.**

The first sentence is typed in WordPad and the execution is stopped.

To go on with the execution, select **Edit/Undo.** You can now see the second sentence being typed.

As you can tell, WinTask waits for a selection by the user before going on with the execution.

NOTE: the selection **Undo** is ignored by Wordpad. WinTask uses the selection for its own purposes and **Undo** is invisible to WordPad.

CHAPTER VI : AUTOMATE SEVERAL PIECES OF SOFTWARE

In this chapter, we will see how WinTask can go from one application to another and how WinTask can create a macro ; not only for one specific piece of software but for different types of software.

WinTask will launch WordPad, type a sentence in WordPad, then launch Microsoft Word. It will type a sentence in Word, then do a **Copy/Paste** between WordPad and Word.

Make certain that you have started this chapter without any other application opened in Windows Desktop. Close any other opened application.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main **WinTask** window with the title *« Untitled 1 »* is displayed as shown below :

Modify the Synchronization Wizard configuration by selecting **Configure/Wizard**. In the **Configure Wizard** dialog box, uncheck the checkbox **Keys** as shown below :

Click **OK**. This will keep the Wizard from appearing again during recording.

Start recording by selecting **Start/Recording**.

A new dialog box asks you if you want to launch a program before starting your recording :

Click **Yes** as you want to launch Wordpad.

In the next dialog box, specify the program you want to launch, Wordpad.exe :

And click the **OK** button.

The Editor window is minimized and the blinking video icon is displayed on the righthand side of the taskbar.

The WordPad window is opened. Type the following text in it :

« Text entered in WordPad »

In order to launch Word by a specific statement in WinTask (instead of recording menu **Start/Programs/Word** which is less reliable than as shown here), stop Recording mode by clicking the WinTask icon in the Windows taskbar. WinTask main window is displayed ; select **Start/Recording after launching a program** :

The next dialog box lets you select the program you want to launch. You can use **Browse** button in order to find easily Word software.

Click OK button, Word is launched and you are again recording all your actions.

In the Word window, type the following text :

« Text entered in Word »

In the taskbar, click the application **WordPad**. Using the mouse, select (highlight) the text that you just typed in the window.

Copy that text by selecting **Copy** in the WordPad **Edit** menu.

In the taskbar, click the application **Word** to open its window. The cursor should be at the end of the text you have just entered. Press the **Enter** key to paste the text from WordPad two lines below. Copy the text by selecting **Edit/Paste**.

Quit Word by selecting **File/Exit** (DO NOT save this document !)

Quit WordPad by selecting **File/Exit** (DO NOT save this document !)

You are now in the same place and with the same conditions as when you started this chapter.

Click the **WinTask** icon in the Windows taskbar to stop the Recording Mode. The WinTask main window is displayed with all the commands inserted in the script :

```
Shell("wordpad.exe",1)

UseWindow("WORDPAD.EXE|RICHEDIT20A|Document - WordPad|1",1)
      SendKeys("Text entered in WordPad")

Shell(Chr$(34)+"C:\ProgramFiles\Microsoft Office\Office\WINWORD.EXE"+Chr$(34),1)

UseWindow("WINWORD.EXE|_WwG|Document1|1",1)
      SendKeys("Text entered in Word")

UseWindow("WORDPAD.EXE|RICHEDIT20A|Document - WordPad|1",1)
      ClickMouse(Left,Down,6,14)
      ClickMouse(Left,Up,7,14)

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&Edit|&Copy     Ctrl+C")

UseWindow("WINWORD.EXE|_WwG|Document1|1",1)
      SendKeys("<Enter>")
      SendKeys("<Enter>")

UseWindow("WINWORD.EXE|MsoCommandBar|Menu Bar",1)
      ClickMouse(Left,Down,76,13)

UseWindow("WINWORD.EXE|MsoCommandBarPopup|Edit",1)
      ClickMouse(Left,Down,47,100)
      ClickMouse(Left,Up,47,100)

UseWindow("WINWORD.EXE|MsoCommandBar|Menu Bar",1)
      ClickMouse(Left,Down,49,9)

UseWindow("WINWORD.EXE|MsoCommandBarPopup|File",1)
      ClickMouse(Left,Down,92,291)
      ClickMouse(Left,Up,92,291)

UseWindow("WINWORD.EXE|#32770|Microsoft Word",1)
      Click(Button,"&No")

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")

UseWindow("WORDPAD.EXE|#32770|WordPad",1)
      Click(Button,"&No")
```

Before explainig all these commands, we will first run the script.

In the **Editor** window, select **Start/Run**. A dialog box asks you to save this script. Type the name *« script3 »* and click **Save**.

The script is now compiled. There should be no compilation errors and you can click **Run** to launch the script.

You can see all your recorded actions being done automatically.

Commands generated by the Recording Mode

The script starts by a *Shell* command which runs the .EXE as specified.

```
Shell("wordpad.exe",1)
```

Then a UseWindow command is used :

```
UseWindow("WORDPAD.EXE|RICHEDIT20A|Document - WordPad|1",1)
      SendKeys("Text entered in WordPad")
```

You will notice a lot of commands like this in a WinTask script. This command tells the automate which window to send its actions. The parameters for this command are the window name, its class and the window title. The last number behind the comma is the instance number. Don't worry, WinTask automatically finds these window names.

In the last line, the *SendKeys* command simulates the user pressing a key on the keyboard. The simulated keys are typed in the window specified by the previous *UseWindow* command. The typed text is put between paranthesis and keywords are written by their labels between the two signs less than and greater than (for instance <Enter>, <Control>, etc…)

```
Shell(Chr$(34)+"C:\ProgramFiles\Microsoft Office\Office\WINWORD.EXE"+Chr$(34),1)

UseWindow("WINWORD.EXE|_WwG|Document1|1",1)
      SendKeys("Text entered in Word")
```

Again, a Shell command is used to run the program WINWORD.

A new *UseWindow* command shows that WinTask actions must now go to a Word window, a window with the title *« Document1 »* .

In the last line, the *SendKeys* command simulates text entered in the Word window.

```
UseWindow("WORDPAD.EXE|RICHEDIT20A|Document - WordPad|1",1)
      ClickMouse(Left,Down,6,14)
      ClickMouse(Left,Up,6,14)
```

```
UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&Edit|&Copy     Ctrl+C")
```

After having entered text in Word, WinTask must now select the text entered in WordPad. As usual, a *UseWindow* command shows that the actions must now take place in a WordPad window.

The next two lines simulate the text selection in the WordPad window. The first one simulates a mouse button being pressed, the second one the button being released.

The last two lines automate the selection **Edit/Copy**. Note that the first parameter of the *ChooseMenu* command is *« Normal »* because it is a selection from an application menu.

```
UseWindow("WINWORD.EXE|_WwG|Document1|1",1)
      SendKeys("<Enter>")
      SendKeys("<Enter>")

UseWindow("WINWORD.EXE|MsoCommandBar|Menu Bar",1)
      ClickMouse(Left,Down,76,13)

UseWindow("WINWORD.EXE|MsoCommandBarPopup|Edit",1)
      ClickMouse(Left,Down,47,100)
      ClickMouse(Left,Up,47,100)

UseWindow("WINWORD.EXE|MsoCommandBar|Menu Bar",1)
      ClickMouse(Left,Down,49,9)

UseWindow("WINWORD.EXE|MsoCommandBarPopup|File",1)
      ClickMouse(Left,Down,92,291)
      ClickMouse(Left,Up,92,291)

UseWindow("WINWORD.EXE|#32770|Microsoft Word",1)
      Click(Button,"&No")
```

A new *UseWindow* command makes the automate come back to Word and in the next line, the *SendKeys* command simulates the **Enter** key being pressed twice.

The other command used is *ClickMouse*. Because this version of WinTask (using version 7 of Word) isn't able to automate menu selections by name, the selections are *position dependent* instead.

```
UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")

UseWindow("WORDPAD.EXE|#32770|WordPad",1)
      Click(Button,"&No")
```

The last four lines close WordPad by simulating the selection of **File/Exit**. By clicking **No**, the WordPad document is not saved.

CHAPTER VII : HOW TO FIND WINDOW NAMES

In this chapter, we will study the Spy tool included in Wintask. The Spy tool allows you to find the name of a window and also the arguments for the *UseWindow* command. In WinTask version 2.42, if you use Spy tool on an Internet Explorer 5.0 or above window, it gives you also a description of the HTML elements displayed in the Web page.

The name of a window is usually found automatically when you use the Recording Mode. But even then, occasionally the error message *« Window not found »* is received. The Spy tool can help solve this problem and determine which window is needed.

Launch WinTask by clicking **Start/Programs/WinTask/WinTask**. The main **WinTask** window is displayed with the title *« Untitled 1 »* .

Launch another application, in this case WordPad, by clicking **Start/Run.** Type *« WordPad »* in the dialog box **Run** as shown below :

Click **OK**. The window titled *« Document – WordPad »* is opened.

Maximize the Editor window and select **Start/Spy** :

The **Spy** window is displayed as shown below :

Click **Find Window**.

The mouse pointer is transformed into a cross. When you move the cursor in the different areas of WordPad, you see different information in the **Spy** window.

The Spy tool provides the different names of the windows inside WordPad. For instance, if you put the mouse pointer on the title bar of WordPad, you get the following information :

If you put the mouse pointer on the Wordpad toolbar, you will see :

And so on ....

The field « *UseWindow* » contains the argument for the *UseWindow* command and you can directly paste this argument into your script. Click **Paste** and you will paste the following line into the Editor window :

```
"WORDPAD.EXE|Toolbarwindow32|Standard"
```

Using Windows NT (instead of Windows 98), the line is slightly different :

```
"WORDPAD.EXE|ToolbarWindow32|Standard|2"
```

You only have to add the word « *UseWindow* » before the paranthesis and to surround the paranthesis with two brackets and your *UseWindow* command is generated :

```
UseWindow("WORDPAD.EXE|ToolbarWindow32|Standard")
```

More information on a Web page

Spy tool gives more information in the frame «Web information» when it is used on an Internet Explorer window (IE version must be 5.0 or above).

Go to www.wintask.com using Internet Explorer. The IE window is displayed :

Launch **Spy** tool. The **Spy** window is displayed as shown below :

Click **Find Window**.

The mouse pointer is transformed into a cross. For instance, if you put the mouse pointer on the paragraph starting at Automate your tasks with WinTask, you get the following information :

The frame *« Information »* gives the IE window attributes and the frame *« Web Information »* tells on which HTML tag the mouse is pointed and what are the mouse coordinates.
Click **HTML Capture** button if you want to retrieve the text displayed in the Web page when you replay the WinTask script.

CHAPTER VIII : SYNCHRONIZATION WITH A TEXT DISPLAYED ON THE SCREEN

WinTask automatically waits until the next window is displayed before sending its actions to the new window. This automatic synchronization (through the *UseWindow* command) cannot be made when the text displayed in a window changes but the name of the window doesn't change. This typically occurs when a task is automated on a mainframe : the name of the emulator window is always the same. But each time **Enter** is pressed, the mainframe eventually sends a new screen with the same name but with different text in the window. This delay depends on how over- loaded the mainframe is. In this case, we will make WinTask wait until it detects new, specific text in the window.

We will show this synchronization using Windows Explorer, as everybody does not have a mainframe.

In this example, we will launch Explorer, look for the file *« WordPad.exe »* on the hard disk, wait until the message *« File found »* appears in the Explorer window and immediately launch WordPad.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main WinTask window is displayed with the title *« Untitled1 »* as shown below :

Modify the wizard configuration by selecting **Configure/Wizard**. In the dialog box **Wizard Configuration**, check the radiobutton **None** to get the screen below :

Click **OK**.

Start the Recording Mode by selecting **Start/Recording** in the WinTask main window or click the

button in the toolbar. If the dialog box below is displayed, click **No** as we want to start Recording directly.

The **Editor** window disappears and a small icon of a blinking video camera is displayed on the righthand side of the Windows taskbar. This

icon reminds you that all your actions are being recorded.

Click **Start/Find**, then **Files or Folders**. In the next dialog box, type *« wordpad.exe »* as shown below :

Click **Find Now**.

It takes some time to look for this file but when the file *« wordpad.exe »* is found, a message on the lefthand side of the window is displayed. We will make WinTask wait until this message appears.

Stop the Recording Mode by clicking the blinking video icon in the taskbar. The WinTask main window is opened.

In this window, select **Insert/Synchronization** and then **On text,** as shown below :

The next window is displayed :

Click **Capture**. The mouse pointer changes shape. Highlight the sentence *« 1 file(s) found »* . WinTask must wait for this text before going any further. It is now in the field *« Text »* of the **Text Synchronization** window, as shown below :

The *« Text »* field is now filled with the captured text that the script must wait for before preceeding.

Click **Paste into the script**. A couple of commands are automatically inserted in the script :

```
      Pause until
            Text("1 file(s) found")
            InWindow("EXPLORER.EXE|msctls_statusbar32|Find:      Files      named
wordpad.exe|1",1)
            InArea(0,0,18,79)
      PauseFalse
            MsgBox("Wait   for   at   line   "   +   #ErrorLine$   +   "   has   failed
!",16,"Runtime error")
            End
      EndPause
```

A *Pause* command has been inserted which means : wait until the text *« 1 file(s) found »* is displayed in the window specified by the *InWindow* command. If this text is not found, the command just after the *PauseFalse* keyword is executed. This command displays an error message indicating the line in the script where the error has occurred.

Move the cursor to the line following the *EndPause* keyword in the script by pressing the **Enter** key and start recording as usual.

In the **Find** window, click twice on **WordPad.exe** in order to launch it, then quit WordPad by selecting **File/Exit**.

Close the **Find** window by clicking the close button on the top right hand corner of the window.

Finally, stop the Recording Mode.

Run the script by selecting **Start/Run**.

A dialog box asks you to save the script. Type the name *« Script4 »* and then the script is compiled. There should be no errors and then click **Run** to launch the script.

CHAPTER IX : SYNCHRONIZATION WITH A BITMAP DISPLAYED ON THE SCREEN AND CLICK A BITMAP

WinTask automatically waits until the next window is displayed before sending its actions to the new window. This automatic synchronization (through the *UseWindow* command) cannot be made when the objects (text or bitmap) displayed in a window changes but the name of the window doesn't change. This typically occurs when a Web navigation is automated.

We will show this bitmap synchronization on a Web page. In version 2.3 and above, a new way of recording navigation within Web pages has been introduced ; see next chapter for this issue.

In this example, we will wait for the button Acrobat Reader in a page of our Web site www.wintask.com, then click that button. As this button is not a standard button (such as OK, Cancel buttons, …), we have to use the bitmap synchronization.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main WinTask window is displayed with the title « *Untitled1* » as shown below :

Modify the wizard configuration by selecting **Configure/Wizard**. In the dialog box **Wizard Configuration**, check the radiobutton **None** to get the screen below :

Click **OK**.

Lauch your Web explorer (in this manual, we use Internet Explorer but you can use Netscape explorer, the only difference will be the window names).

Go to www.wintask.com :

We want WinTask to click the big arrow (Click here …) which is not a text but an image.

Select **Insert/Statement** in WinTask main window.

The **Insert statement** window is displayed : it lists all the functions available in WinTask.

Select **ClickOnBitmap** statement and right click ; in the Context menu, select **Wizard**.

The **ClickOnBitmap()** window is displayed :

Click **Yes** to use the mouse to define the image.

The **Click on bitmap** wizard window is displayed :

Click **Capture**. The mouse pointer changes shape. Surround the arrow image in the Web page in order to capture this image. The next window **Preview** is displayed :

Close it to come back to **ClickOnBitmap** wizard. You must save this image in a bitmap file. Click **Browse**. A dialog box asks you to save that image. Type the name « arrow » and click **Save**.

You come back to **ClickOnBitmap** wizard. Click the radiobutton **Somewhere in the window** in the frame « The synchronization is done only if the captured area is :» as shown below :

Click **Paste into the script**. The editor window is displayed and you should have a script that loook like :

```
UseWindow( "IEXPLORE.EXE|Internet Explorer_Server|Automate your routine tasks using Wintask and forget batch, macro recorder and scheduler softwa - Microsoft Internet Explorer|1",1 )
ClickOnBitmap( "C:\Program Files\WinTask\Scripts\arrow.bmp", left, single )
```

Launch the script by selecting **Run** in the **Start** menu.

Because the script has not been saved, the **Save as** window is displayed. Type the name « bitmap » and click **Save**.

You can see that the script clicks automatically on the arrow.

We are now on another Web page. This Web page can be displayed within a couple of seconds or a couple of minutes. We will insert an image synchronization ; like that, the script will wait until this next Web page is present.

In the main WinTask window, select **Insert/Synchronization/On image** :

The **Image Synchronization** window is displayed :

Click **Capture**. The mouse pointer changes shape. Surround the text What's new which is in fact an image in the Web page. The next window **Preview** is displayed :

Close it to come back to **Image synchronization** window. You must save this image in a bitmap file. Click **Browse**. A dialog box asks you to save that image. Type the name *« whatsnew »* and click **Save**.

You come back to **Image Synchronization** window. Click the radiobutton **Somewhere in the window** in the frame « The synchronization is done only if the captured area is :» as shown below :

Click **Paste into the script**. A couple of commands are automatically inserted in the script :

```
      PAUSE until
      Bitmap("C:\Program Files\WinTask\Scripts\whatsnew.bmp")
      InWindow("IEXPLORE.EXE|Internet  Explorer_Server|What's  new  on  WinTask  -
Microsoft Internet Explorer|1",1)
PauseFalse
      MsgBox("'Wait  for'  at  line  "  +  #ErrorLine$  +  "  has  failed  !",0  ,"Runtime
error")
      End
EndPause
```

A *Pause* command has been inserted which means : wait until the image *« What's new ? »* is displayed in the window specified by the *InWindow* command. If this image is not found, the command just after the *PauseFalse* keyword is executed. This command displays an error message indicating the line in the script where the error has occurred.

Move the cursor to the line following the *EndPause* keyword in the script by pressing the **Enter** key and and you can now start recording as usual.

CHAPTER X : SYNCHRONIZATION WITH A NEW WINDOW DISPLAYED ON THE SCREEN

WinTask automatically waits until the next window is displayed before sending its actions to the new window. Sometimes this automatic synchronization (through the *UseWindow* command) is not enough as a long process is run before the window showing results is displayed. Window synchronization will force the script to wait until a specific window is displayed (or disappears).

We will show this synchronization on appearance of window Save As in Wordpad when menu File/Save is selected.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main WinTask window is displayed with the title *« Untitled1 »* as shown below :

Modify the wizard configuration by selecting **Configure/Wizard**. In the dialog box **Wizard Configuration**, check the radiobutton **None** to get the screen below :

Click **OK**.

Start recording by selecting **Start/Recording**.

A new dialog box asks you if you want to launch a program before starting your recording :

Click **Yes** as you want to launch Wordpad.

In the next dialog box, specify the program you want to launch, Wordpad.exe :

And click the **OK** button.

The Editor window is minimized and the blinking video icon is displayed on the righthand side of the taskbar.

The WordPad window is opened. Type the following text in it :

« Text entered in WordPad »

Select menu **File/Save**. The Save As dialog box is displayed ; WinTask will wait until this window appears.

For inserting this window synchronization, stop Recording mode by clicking the blinking video icon in the taskbar. The WinTask main window is opened.

In this window, select **Insert/Synchronization** and then **On Window,** as shown below :

The next window is displayed :

Click **Spy**. The mouse pointer changes shape. Click the window Save As and the dialog box Window synchronization is now filled with the proper window name you want WinTask waits for :

Click **Paste into the script**. A couple of commands are automatically inserted in the script :

```
Pause until
      WinStatus(Active,Exact)
      InWindow("WORDPAD.EXE|#32770|Save As",1)
PauseFalse
      MsgBox("Wait for at line " + #ErrorLine$ + " has failed !",16,"Runtime error")
      End
EndPause
```

A *Pause* command has been inserted which means : wait until the window Save As of Wordpad is displayed. If the specified window is not found after a Timeout, the command just after the *PauseFalse* keyword is executed. This command displays an error message indicating the line in the script where the error has occurred.

Move the cursor to the line following the *EndPause* keyword in the script by pressing the **Enter** key and start recording as usual.

Click **Cancel** button for not saving your document. Select menu **File/Exit** to exit wordpad and do not save your document.

Finally, stop the Recording Mode.

Run the script by selecting **Start/Run**.

A dialog box asks you to save the script. Type the name *« Script5 »* and then the script is compiled. There should be no errors and then click **Run** to launch the script.

Note for WinTask users (not available in WinTask lite) : this synchronization method combined with OnAction statement (see help on OnAction statement) allows you to create scripts doing actions as soon as a specific window (an error window for instance) is detected on the screen.

CHAPTER XI : HOW TO RECORD NAVIGATION WITHIN WEB PAGES

A new Recording method has been introduced with WinTask version 2.3 : instead of recording user actions, the result of user Web navigation is recorded. This chapter shows how it works.

Launch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The main WinTask window is displayed with the title *« Untitled1 »* as shown below :

Configure the Recording mode by selecting **Configure/Recording**. In the dialog box **Configure Recording**, check the radiobutton **Web mode Recording** to get the screen below :

Note that the toggle **Control+W** allows you to switch to and from the usual Recording mode at any time.

Start Recording mode by selecting **Start/Recording ;** Internet Explorer is automatically opened (Netscape is not supported in this version), type an URL address, for instance www.wintask.com**.** Stop the Recording mode by clicking the red flashing icon in the taskbar, the script is automatically generated as below :

The first statement starts IE, the second statement goes to the page you've asked for.

Close your browser, and run your script. You see IE starting and the Web page is displayed.
Note that each time you start Web mode recording, a new instance of Internet Explorer is opened, so it is better to record all your navigation in one shot.

For pages with frames, an other function is used : Attribute and if you fill forms, the PostData function is used. A full description of those functions is available in help.

How to measure response times :

The same Navigate function can be used to measure how long it takes for a page to download.

We will show how to do that for a couple of pages within the Web site www.wintask.com.

Create a new script by selecting **File/New**. Close all instances of Internet Explorer.

Start Recording mode (Web mode Recording), Internet Explorer is launched, Type the URL address www.wintask.com. In the menu, select **The language** and when the Language page is loaded, click Windows management functions. Stop Recording mode, the script should be as below:

Close your IE window before running the script.

Now we can add manually the functions for measuring response times. In our example, we will only add the return variable in front of all the Navigate functions and a MsgBox function to display the return code (which gives the response time). So the script should be :

Close IE window before running the script. At execution, dialog boxes display response times.

How to detect Page not found :

In WinTask 2.42 version, there is no easy way to detect that a page was not found. So you have to write a function to do it. In this function, we use CaptureIE$ function which allows you to capture text in an IE window (as text displayed in an IE window is not an ASCII text, you cannot use the usual Capture functions).

So, we need to capture the text displayed when a page is not found.

Start Recording mode (still Web mode Recording) and type an incorrect URL address, such as **www.wintas1.com**. The page with the title “The page cannot be displayed” is displayed. DON’T STOP WEB RECORDING MODE but maximize WinTask Editor window and click the icon Language, or press F4, to display the **Insert statement** window and in Web functions chapter, double click CaptureIE$ function. Define the area to capture using mouse. Surround the text “cannot be displayed” ; you can now stop Recording mode.

The generated script is as below :

From that script, we will use the area coordinates where the text “cannot be displayed” is, you find those coordinates on the right part of the CaptureIE$ functions, in our example : 138,23,32,188.

We can now write our error routine, note in the script below how we have shortened the window name for CaptureIE$ argument : using just the beginning of the window name, all the error pages starting with this name will be accepted.

To see what happens when a page is not found, run the above script after having modified one of the .htm page inside the script (in the script above, we changed language.com in langu1ge.com.

CHAPTER XII : LAUNCH THE SAME SCRIPT EVERY WEEK

In this chapter, we will learn how to create a script doing the same actions (in our example, just WordPad will be started) every week without any user intervention (PC's running under Windows 9x or Windows NT/2000/XP must be started first, of course).

If you are under NT/2000/XP, WinTask includes a scheduler, which is an NT/2000/XP service, so directly launch it and do not follow the example in this chapter.

If you are under Windows 98 or ME, you can use the Windows built-in Task Scheduler, see chapter Scheduler under Windows 98 or ME in WinTask help.

We will use the **Date/Hour** synchronization to create such a script.

Lauch the WinTask Editor by clicking **Start/Programs/WinTask/WinTask**. The following window is displayed :

Select **Insert/Synchronization** and then the option **On Date/Hour** :

We will configure this window so that every Monday at 10 AM, an action is performed by the script.

Check the radiobutton **Day** and in the list, select **Monday** :

In the frame *Hour*, scroll in order to display 10 and 0 as shown below :

Click **Paste into the script** in order to generate the correspondant commands in the script :

```
Pause until
      Date(Monday)
      Time(10:0)
EndPause
```

Before going on with the creation of the script, we will first study what can be configured in the window **Date/Hour Synchronization**. Make it display again by selecting **Insert/Synchronization** and then the option **On Date/Hour**.

In the frame *Hour*, scroll through the different hours until you find *« Any »* as shown above. In the field on the right of *« Any »* , scroll down until you find *« 10 »* . Because the field *Date* contains 13, 6 and 1999, this synchronization will cause some action to be launched every 10 minutes starting on the 13th of June, 1999.

If you want to launch an action every 10th of every month at noon, the **Date/Hour Synchronization** window must look like the example below :

You can see how the keyword *Any* is able to cover many different types of synchronization.

After you have pasted the synchronization commands in the script, you still have to record the actions you want the script to perform for you. We take as an example the action of launching WordPad. After maximizing the WinTask main window, you should have some lines at the beginning :

Verify that the cursor is on the line following the *EndPause* command. If not, move it into place. You can now use the Recording Mode to record all the actions you want the script to do for you, every Monday at 10 o'clock.

In our example, we will launch WordPad and then close it.

Start the Recording Mode as usual.

Click **Start/Run** and type in the dialog box **Run** *« WordPad »* , then click **OK**. Once WordPad opens, close it by selecting **Exit/File**. Stop the Recording Mode by clicking the blinking video camera icon on the right handside of the taskbar.

You should have a script like below :

```
      Pause until
            Date(Monday)
            Time(10:0)
      EndPause

UseWindow("EXPLORER.EXE|Shell_TrayWnd",1)
      Click(Button,"Start")
      ChooseMenu(Context,"&Run...")

UseWindow("EXPLORER.EXE|#32770|Run",1)
      WriteCombo("1","wordpad")
      Click(Button,"OK")

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")
```

We now have to insert a loop in order to make the script wait until Monday at 10AM :

For this we use the *GOTO* command :

At the beginning of the script, insert a line containing a label, for instance *« DEBUT »* (a label is a word followed by a colon). Then at the end of the script, insert a line which makes the execution go to the beginning of the script, *« GOTO DEBUT »* . Just before this *GOTO* command, include a

« Pause 61 » so that the script doesn't just continuously loop forever, monopolizing all the computer's resources.

The script should look like this :

```
DEBUT:

      Pause until
            Date(Monday)
            Time(10:0)
      EndPause

UseWindow("EXPLORER.EXE|Shell_TrayWnd",1)
      Click(Button,"Start")
      ChooseMenu(Context,"&Run...")

UseWindow("EXPLORER.EXE|#32770|Run",1)
      WriteCombo("1","wordpad")
      Click(Button,"OK")

UseWindow("WORDPAD.EXE|WordPadClass|Document - WordPad",1)
      ChooseMenu(Normal,"&File|E&xit")

pause 61
GOTO DEBUT
```

In order for this script to be launched at the a specific hour, it must be in memory as soon as Windows is started. To do this, we will put the script in the **Startup** group. Before doing this, compile the script by selecting **Start/Run**. Give the script the name *« Script6 »* and the following is displayed :

There should be no compilation errors but don't run the script for now. The compilation has created the file *« script6.rob »* which we will include in the Windows **Startup** group :

Include the script launch in the Startup group

Click the right mouse button in the taskbar. In the menu displayed, select **Properties**. You should see something like this :

Click the Tab **Start Menu Programs**. On next screen, click **Add** :

The next screen should be as follows :

Click **Browse** to look for the file *« Taskexec »* (this file is in the folder *« Bin »*, a sub-directory of WinTask, which in turn is a sub-directory of Program Files). When the file is located, click **Open**. The command line displayed in the screen above is automatically generated.

After this command line, add the script name you want to launch (in our example *« script6.rob »*), with its full path :

« C:\Program Files\WinTask\scripts\script6.rob »

The command line should look as follows :

Click **Next** :

And then click the folder **StartUp**.

The next screen displayed is :

Create a name for this shortcut and click **Finish**. On the next screen, click **OK**. The shortcut is added in the **Startup** group.

Restart the PC.

You can see in the taskbar the script script6 ready to be launched every Monday at 10.

CHAPTER XIII : AUTOMATE IN A WINDOWS DOS BOX

WinTask can automate tasks in a Dos box of Windows 9x, ME, Windows NT, 2000 or XP.

There is a limitation using a Dos box which makes creating this automation a bit more difficult than usual : the Recording Mode will not record the keystrokes you enter in a Dos box which means you have to create your script manually.

In the following example, we will create a script launching a Dos box, typing the *Dir* command and exiting the Dos box with the *Exit* command.

Lauch WinTask Editor by clicking **Start/Programs/WinTask/WinTask**.

The Editor window is displayed with the title « *Untitled1* » as shown below :

Start the Recording Mode by selecting **Start/Recording**.

A new dialog box asks you if you want to launch a program before starting your recording :

Click **Yes** as you want to launch command.com to open a Dos box (under Window NT, replace command.com by cmd.exe).

In the next dialog box, specify the program you want to launch, command.com :

And click the **OK** button.

The **Editor** window is minimized and the blinking video icon is displayed on the right handside of the taskbar. All your actions will now be recorded.

The Dos box is now opened. If this window is full screen (you don't see the taskbar), resize it by typing *« Alt+Enter »* . Type « *Dir* » and press **Enter**. The file list is displayed. Then type *« Exit »* and press **Enter**. You will then come back to the Windows environment.

Click the small blinking video icon to stop the Recording Mode. The main Editor window is displayed with the generated commands :

You can see that the generated commands stop at « Shell ». None of your actions in the Dos box were recorded.

We will manually add these commands in the script.

In the Editor main window, go to the first line immediately following all the generated commands by pressing the **Enter** key.

To use the Spy tool to find the name of the Dos box window, run the following script: Click **Start/Run**. Name the script *« dosbox »* and then run it. The Dos box is now opened.

We will now find the name of this window using the Spy tool.

Click the minimized WinTask window in the taskbar to put the focus on the Editor window and select **Start/Spy**.

Click **Find Window** in the **Spy** window. Move the cursor to the Dos window. Paste the name of this window in the script by clicking **Paste**. The following command is inserted in the script :

```
"WINOA386.MOD|ttyGrab|MS-DOS Prompt|1"
```

Under Windows NT, the Dos window name is a bit different :

```
"CMD.EXE|ConsoleWindowClass|C:\WINNT\System32\cmd.exe"
```

To make certain that WinTask sends all the actions to this window, you must include a *UseWindow* command in front of this window name.

The command should be :

```
UseWindow("WINOA386.MOD|ttyGrab|MS-DOS Prompt|1")
```

Two commands must be added to simulate the action of typing the *Dir* command and **Enter**, plus the *Exit* command and **Enter**.

```
SendKeys("Dir<Enter>")
SendKeys("Exit<Enter>")
```

The Dos box script should now look like :

```
Shell("command.com",1)

UseWindow("WINOA386.MOD|ttyGrab|MS-DOS Prompt|1")
      SendKeys("Dir<Enter>")
      SendKeys("Exit<Enter>")
```

Before launching this script, close the still-opened Dos box (type *« Exit »* in the Dos box and press **Enter**).

Run the script by clicking **Start/Run** in the **Editor** window. There should be no compilation errors and the script should be able to run.

CHAPTER XIV : TIPS

How can I stop a script ?

Click the right mouse button when the name of the script is highlighted in the taskbar and select **Close**.

or

Simultaneously press the keys **Shift**, **Ctrl** and **Pause** OR **Shift**, **Ctrl** and **ScrollLock** (Note: this stops ALL scripts).