

# ***Wise Installation System Version 5.0***

## ***Release Notes***

### **GENERAL INFORMATION**

The Wise Installation System creates installation programs for Windows 3.1x, Windows 95, and Windows NT applications and data files. It combines all of the files to be installed into a Windows Self-Installing Executable. This single file is then executed on the target computer to perform the installation. No additional software (i.e. dearchive program) is required. The installer does not need to execute the program under DOS first and then execute the extracted setup program. The installation executable is a true Windows program. The installation executable can be directly uploaded to a BBS or on-line service or sent via e-mail. Wise adds less than 68 K bytes overhead to the size of the compressed files.

Any complex program such as the Wise Installation System will have "Features" which are more commonly referred to as bugs. Even though this program has been tested thoroughly, please test any installation executables before distributing them. If you find a bug in Wise or have any comments or suggestions, please contact GLBS at:

Fax Line:	(313) 981-1819
GLBS BBS:	(313) 981-4684
Compuserve:	74777,3314 or GO WISEINSTALL
E-Mail:	tech@glbs.com
America On-Line:	GLBSInc
Voice Line:	(313) 981-4970

The following sections describe the new features added to the Wise Installation System version 5.0. They also describe how to access these new features.

### **SCRIPT FILE (.WSE)**

The script file is now ascii!! You can now have two copies of WISE running at the same time and cut/paste script items between them. You can also cut/paste to the clipboard, an e-mail message, or just about anywhere. We do not have documentation completely describing every command and the syntax yet, however, we do hope to eventually have something published.

We feel that the syntax we have chosen is self-documenting. Currently, just about everything is built on the premise of being an item. There are two types of items (i) Global, which describes the installation itself; (ii) script items, one for each script item in the installation. The syntax for item statement is "item: xxx". The block will always end with an "end" statement.

Each item statement may have properties and/or items of it's own. For example: a set variable script item will have only properties within the item block. A custom dialog script item will have properties and items such as pushbutton, listbox, static, etc.

The values of each item are defined to make it easy to create a script in a programming language or a text editor. The only values that may be hard to decipher are the flags. If you want to duplicate the flag values you may need to create a script item with your

options to see the flags that you want to set.

Once you have created your text script outside of the editor you can run WISE from the command line with the following syntax: "WISE (or WISE32) /C myapp.wse". This will create the executable as defined in the script.

## **INSTALLATION PROPERTIES**

### **Global:**

#### **Progress Bar Based on:**

Total Installation, Total Script, Files to be installed

### **Settings:**

#### **Installation EXE Name:**

By default the Installation Executable(s) will be created in the same directory as your .WSE file. If you would like to change where the installation executable is created fill in the directory and filename.

#### **Language INI Name:**

The Installation Expert can create multi-lingual scripts. This INI file needs to contain the translations of the messages used in the dialogs by the Expert.

(i.e. Welcome, Browse for Directory, Installation Complete, Etc.)

Setup Icon Pathname: This ICO file will be used as the icon included with the installation executable.

#### **Progress Bar DLL:**

You may create a DLL that will be used in place of the standard progress bar. You will find a complete sample DLL that may replace the progress bar in the directory DLL\PROGRESS below your main WISE Directory.

#### **Dialogs Directory:**

The Installation Expert uses 9 standard dialog boxes to create the Wizard installation. By default these can be found in the DIALOGS\TEMPLATES directory under your main WISE Directory. If you would like to customize these, we suggest you make a copy of them in a different directory and fill these fields in with the pathname.

#### **Temp Files Directory:**

By default WISE will create temporary files in the TEMP directory defined by Windows. You may specify a different directory in this field, possibly one that might have more disk space.

### **SMS/MIF:**

#### **MIF Filename:**

You can specify the name of the MIF file created for use with an SMS type distribution system. This file will be created in the Windows directory.

#### Manufacturer, Product, Version, Serial Number:

These fields will be inserted into the MIF file that is created at the end of the installation. You should consult your SMS manual for information about the values of these fields.

#### Compiler Variables:

Compiler variables are used exactly like an IFDEF in a C type programming language. These variables will look exactly like standard WISE variables except they need to be defined in this dialog. These can be used to create conditional compiles of an installation for deployment for testing. WISE uses a standard of an underscore ("\_") as the first and last character of a compiler variable to make it simpler to determine which type of variable it is. Once a variable has been defined you may set a default value to be used if it is not prompted for. You may also specify when you want to be prompted for the value. For example during testing (compiling from the IDE) you may not want to be prompted every time for each compiler variable. However, when you are creating a final installation (either from the IDE or from the Command Line) you may want to be prompted for the values. You can run WISE from the command line with the following syntax: "WISE (or WISE32) /C myapp.wse". This will create the executable as defined in the script.

## **DIALOG TEMPLATES**

#### Editing Dialog Templates:

The Installation Expert has 9 dialog templates as well as 11 other script items have dialog templates that you may edit. We suggest that you make a copy of the dialog templates before modifying them. You can change the default directory for the dialogs in the Installation Properties/Settings property page. You can modify any of the standard templates such as Browse for Directory, Prompt for Text, Select Components, Get Name Serial Number, Etc.

#### Events:

The Mainline "event" is actually the full installation script. WISE currently supports two other events:

#### Exit:

The Exit "event" is executed when the script runs to completion or executes an "Exit Installation" script item. You may add code that may be executed at this time such as Prompting to Execute the program just installed or any other common task that you do not want to duplicate code for within the MAINLINE.

#### Cancel:

The "Cancel" event is executed whenever the user chooses to cancel the installation. This can be on any dialog or during an actual file installation. When a user chooses to cancel an installation, WISE will bring up a dialog that asks if they are sure. If the user confirms to cancel the installation then the script items in this event will be executed. You may

check to see if a partial installation was done and notify the user that he may need to copy some of the BACKUP files into production. If you did not want the user to intervene the script could look for files in the backup directory and copy them into production.

## **SCRIPT ITEMS**

### **Compiler Variable IF:**

This script item works exactly the same as a standard IF Block, however, it will only accept compiler variables. You need to define the compiler variables in the Installation Properties/Compiler Variables property page. An example of using this statement would be if you created a single installation that installed both the 16bit version of your program and the 32bit version. You could use a compiler variable to determine what type of installation you would be creating and choose only to include the required files for that platform.

### **Compiler Variable Else, Compiler End:**

These are used in the exact same manner as "Else Statement" and "End Block". They may only be used in conjunction with a "Compiler Variable If" script item.

### **Copy Local File:**

This statement now supports wildcards in the Source field. If you do use a wildcard in the Source then your Local Path must be blank. WISE will determine what files will be copied at the time the installation is run, not when the installation is compiled.

Remember, since we do not know how many files, or what size they are, the progress bar will not move while this statement is being executed. The best way to have the progress bar look normal is to have it be Script Item Based.

### **Create Service:**

This script item gives you the ability to add a Service (background process) to a system. You should consult Microsoft's MSDN for more information about creating a service.

### **Edit INI File:**

This script item has been changed so that you only need one line for each INI file being edited. You only specify the file and the "INI File Contents" will look exactly like your INI file. You can cut and Paste from any standard INI file. If you include a section (i.e. [Section]) with no entries after it, then the Section will be deleted. If you include an entry in a section with no value (i.e. "LastFile=") then that line will be deleted. You may do multiple events in a single "Edit INI File". For example: you may delete a section(s), add a section, add an entry(s), remove an entry(s) all within a single script item.

### **Edit Registry:**

The new addition to this script item is that you may edit a Binary Value. When you are adding a new Binary Value you need to enter the Key Value as a Hex String. For example: "1E 2F 3C BB 23 65", any digit that is not a hex value will be converted to a "0" (zero).

**Execute Program:**

We have added the "Minimized" and "Normal" option for the Window Size when executing an external program.

**Find Files In Path:**

We have added an edit box that will allow you to specify what additional directories to search in, as well as the PATH defined by Windows.

**Get System Information:**

We have added three new system information:

**Volume Serial Number/Volume Label:**

This will return the serial number/volume label of the media defined in PATHNAME. Make sure that you use a variable reference and not a hardcoded pathname.

**Network Username:**

This information is obtained by calling the WNetGetUser API Call.

**Install DirectX:**

We have added the option to install Direct3D. You should consult the Microsoft MSDN for more information about installing DirectX.

**Read/Update Text File Settings:**

This script item creates a loop block, executing the loop once for each line of text in the file. You need to fill in the pathname with a file that exists on the user's system. Make sure that you use a variable reference and not a hardcoded pathname. The Variable that you use will hold a single line of text from the file each time the loop is executed. If you change the value of the variable inside of the loop the line will be updated in the file if you select to update the text file.

**Search for File:**

This script item now allows you to search for Files on Local Drives Only, Network Drives Only, or Everywhere. We suggest that if you are searching either on Network drives or everywhere that you limit the search depth. The search depth specifies how many levels (or directories) WISE will go down to on any single drive. A search depth of 0 will search to the bottom of a directory. We suggest that when you search a network drive you limit the level to 2 or 3 so the user is not waiting for a very long time.

**Self-Register OCXs/DLLs**

You can fill in the Description/Pathname with a pathname and check the radio button "Queue existing file for self-registration" to register a file that you did not install with the installation. This works very well for a Workstation installation where all the files are already installed on the network server. You could then use the same script item with a description to register all the queued OCXs/DLLs.

**Wizard Block:**

This script item controls the entire Wizard section of the installation. To better understand this, we suggest you use the Installation Expert to create a script so that you can see how this script item works. You need to end the Wizard Block with an "End Statement" script item.

#### Dialog Boxes:

This is a list of all the dialog boxes that are within the Wizard. We suggest that each Custom Dialog script item only contain one Dialog. Each dialog may have a "Skip Check" placed on it with a simple if statement. By default, the dialog will be displayed. If the statement returns true, then the dialog will be skipped. An example of when you would like to skip a dialog would be when prompting for a backup directory. One dialog would ask if you want to backup the files being overwritten and the next dialog would ask where you wanted the backup files placed. If you stored the question about backing up the files into the variable DOBACKUP, then the next dialog could reference that variable to see if it needed to be skipped. The Statement would be:

"If Variable" DOBACKUP "Equals" "The Value" "NO"

This would skip the dialog that would prompt the user for the directory to store the backup files.

#### Destination Variable:

This variable holds the value of what direction the Wizard is heading. All of the next buttons will set this variable equal to "N" (for next), and all of the back buttons will set this variable equal to "B" (for back).

#### Display Variable:

This variable will hold the name of the dialog that will be executed this time in the loop.

#### Wizard Bitmap Pathname:

You can display a single bitmap in all the wizard dialogs. This should be a standard size graphic that will fit within the wizard dialogs. A sample graphic (WIZARD.BMP) can be found in the DIALOGS\TEMPLATES directory below your main WISE Directory.

#### Wizard Bitmap X-Pos/Y-Pos:

This is a relative placement of the graphic within the dialog.

#### Wizard Bitmap Do not resize bitmap:

Some graphics do not look well when they are resized, you may want the graphic to remain the exact size no matter what resolution the end-user has.

#### Wizard Bitmap 3D Border:

This will add a 3D border to your graphic.

#### Wizard Bitmap Filler Color:

This color will be used around the edge of the graphic if the graphic is not sized at

different resolutions.

## **INSTALLATION EXPERT**

### **Installation Interface:**

#### Media:

You can select either a single file installation or a floppy based installation. The floppy based information has options for which type of floppy or how big you want the installation executable to be.

#### Application:

This dialog defines the software title and the default directory. The software title is used for the wizard dialog titles, as a background graphic, as the primary icon name, and as the title in Add/Remove Software dialog under Windows 95. The default directory will appear exactly how you type it under Windows 3.1 and Windows NT 3.51, however it will be changed to be under "Program Files" for Windows NT 4.0 and Windows 95.

#### Dialogs:

The dialogs that are selected will be used in your installation as wizard dialogs. Remember that some dialogs require additional information such as the ReadMe.

#### Graphics:

The graphics that you add here by default will be timed and the first one will be displayed right before the first file is installed. You may change the properties on a graphic by clicking on the details button after you have selected a graphic. The order that they appear on the screen is the order they will appear in the installation.

### **Application Files:**

#### Components:

You may create components of your installation that the user may choose to install or not install. Each component that you create in this dialog will be listed in the same fashion as a directory in the Files dialog.

#### Files:

The top half of this dialog is your machine (My computer) and the bottom half is the customer's machine (Destination computer).

#### Source Computer:

This is a modified version of the Windows 95 explorer and a file open dialog. On the left hand side is a list of all drives/folder on your machine. You may add the contents of a folder and all sub-folders (to the highlighted folder below) by clicking on the "Add Folder" button. You may move around the directory tree by clicking on an entry. The

right hand side contains a list of files in that folder, you may either double click on a file or use the "Add File" button to include a file.

#### Destination Computer:

The left hand side has a representation of how/where installation files will be installed. If you do not have any components installed, then the structure will have two folders off of the destination computer. The Application folder may contain files or more sub-folders. The Windows folder will contain files to be installed into the Windows directory and sub-folders for Fonts, System, and System32. You may only create folders under the Applications folder, not under the Windows folder. If you have components in your installation then the level just below will list your components and then have the other directories below that level. You may change the installation properties of a file by clicking on the details button.

#### Runtime Support:

##### Options:

You can click on as many different runtime support components as needed by your application. Currently we have support for Uninstall, Share, OLE2, MS-Access 16bit ODBC and MS-Access 32bit ODBC. This list will grow as we develop installation scripts for other various products.

##### VB4:

WISE supports both VB4-16 and VB4-32 installations. Some of the more common components are listed at the bottom of the dialog. You need to specify which directory your VB system is installed in so that WISE may know where to retrieve the runtime files.

##### BDE (Borland Database Engine):

WISE supports both BDE-16 and BDE-32 installations. You need to specify what platform, and for the 32bit, you need to specify whether you will be creating a full or partial installation. You may optionally specify an Alias for your database.

#### User Configuration:

##### Icons:

You may select any number of icons to be added during the installation. The default group name will be used as the Program Manager Group under Windows 3.1 and Windows NT 3.51, and as a sub-menu under Windows NT 4.0 and Windows 95. The icon name you use will be adapted for use under all operating systems.

##### Associations:

You may specify a file extension association for your executable. For example Microsoft would associate the .DOC extension with the WINWORD.EXE file.



#### INI Files:

You need to use the Add Button to create a new INI file within your installation or to edit an existing file. Once you specify the INI file, click on the details button to insert the text. You can Cut and Paste from any standard INI file. If you include a section (i.e. [Section]) with no entries after it, the Section will be deleted. If you include an entry in a section with no value (i.e. "LastFile=") then that line will be deleted. You may do multiple events in a single "Edit INI File". For example: you may delete a section(s), add a section, add an entry(s), remove an entry(s) all within one entry.

#### Registry:

This dialog functions very much like Regedit. The registry that you see is a representation of the changes made to the destination computer. You may create any number of sub-keys and values within the sub-keys. The default type of value created is String, you may change this by clicking on the details button.

#### **System Configuration:**

##### Devices:

You may add entries to the [386Enh] section of the system.ini file. You must first install the file before you can reference it in this dialog. By default the file should have a 386 extension.

##### Services:

This dialog allows you the ability to add a Service (background process) to a system. You should consult Microsoft's MSDN for more information about creating a service.

##### Autoexec.bat:

You can add a path to the autoexec by filling in the Edit Box. Make sure that the path that you are adding starts with a variable and is not a hardcoded path. You can also add commands to the autoexec.bat file by clicking the Add Button.

##### Config.sys:

You can add commands to the config.sys file by clicking the Add Button.