Chief's AutoCalc IDE

The Autocalc Integrated Development Environment ("the IDE") is the Chief's Installer Pro "wizard" for creating and managing installation projects. The IDE adopts a project-based approach, the aim of which is to simplify and automate the process of generating INF files and installation disks for your programs. It also provides a facility for **compiling Chief's Installer Pro INF and BATCH files into a binary format** (to stop those prying eyes); and also a facility for **converting Visual BASIC Setup Wizard project files (.VBZ files)** into Chief's Installer Pro projects.

Provided that some very simple rules are followed, the IDE is easy to operate, and will take the tedium out of the development process. This IDE is entirely optional, and users can build their installation projects manually - through the use of a text editor. The IDE is just provided as a productivity tool for those who care to use it.

The IDE is invoked by running **AUTOCALC.EXE**. This program requires the presence of **AUTODLL.EXE** in the current directory, or in a directory in the "PATH". When AUTOCALC.EXE is executed, the user is presented with a number of menu items, which are discussed below.

Chief Pro Project Manager

The Chief's Installer Pro Project Manager ("PM") is the place where all the installation project development takes place. The PM is centered around a main dialog box which contains most of the activities that can be performed on projects. It is based heavily on **DRAG and DROP from the File Manager**. When you drag files from the file manager and drop onto the PM dialog, the files are processed and split up into component records. You can then browse the records, amend, delete, save, etc., and add new records by dragging new files from the file manager.

Before you can do anything in the PM, you have to **open** an existing project or **create** a new one. Project files have the extension **.CPJ**. These files are comma delimited data files (ASCII format), so they can be manipulated by most database programs. Each project file also has a header file (**.HDR**) in which certain details about the project are kept. This header file is also in ASCII format and so can freely be examined and edited.

You need to treat Chief's Installer Pro projects as you would treat any database. When you open a project, you need to close it before opening another one, or creating a new one, etc. Changes made to any record on-screen are not kept until and unless you first **update** that record (either by clicking on the "Update" button, or my clicking on "Next" or "Previous", if the "Auto Update" check box is checked).

The PM dialog contains many buttons, edit fields, and check boxes, the functions of which are detailed below.

Add

This adds a record to the project. It is largely redundant and not very useful, because you are not expected to add records manually, but rather to **DRAG and DROP** files from the file manager.

Delete

This deletes the currently displayed record from the project, and empties it's contents both on screen and in memory. The project file itself is not changed until you save the changes. Note that deleting a record when the **Auto Update** box is checked will lead to an error message when you click on NEXT or PREVIOUS. So, if you will be deleting records, you have to first uncheck the **Auto Update** check box.

Update

When you **amend** a record on screen, the change is not carried out unless you click on this button (note that deletions are carried out immediately), and if you click on NEXT or PREVIOUS without having updated the amendments, they amendments **will be lost** and you will have to make them again. If you are about to make many amendments, make sure that the **Auto Update** box is checked. This will save you losing any changes you make. The Auto Update box is checked by default every time the Project Manager is started.

Sort

This sorts the project records in memory. There are many ways of sorting and you will be presented with a dialog box which allows you to specify the preferred one. The sorted project is not written to the project file until you click on **Save File**.

Previous

Move to the previous record. If the **Auto Update** box is checked, then any changes to the current record will be updated in memory - and if any of the necessary edit fields is blank, you will get an error message.

Next

Move to the next record. If the **Auto Update** box is checked, then any changes to the current record will be updated in memory - and if any of the necessary edit fields is blank, you will get an error message.

Auto Update

Changes made to records on screen are not committed to memory unless the **Update** button is clicked manually. If you move to the next or previous record without updating, such changes are lost. This check box allows updates to be made automatically every time you click on NEXT or PREVIOUS. It is checked by default every time the Project Manager is started.

New

This allows you to create a new project. A dialog box will be presented which allows you to supply some details for the new project. You first specify the project's working directory, then the name for the project (note that this should just be a name - maximum 8 characters, without any extension). A project file (.CPJ) and a header file (.HDR) will be created for the project. You should also specify the name of the project's **Template INF file**. This is the file that will be created when the **Make INF file** button is clicked.

When you clock OK, the details provided in the dialog will be written to the project header (.HDR) file, and a blank project (.CPF) file will be created. Both of these files are ASCII files which you can edit with a text editor.

After the "New Project" dialog is closed, you are returned to the Project Manager. There will be no records in the project, and you then need to open the file manager, and drag and drop your project's files from the file manager to the Project Manager dialog. As you do this, the file details are broken up as required, and the project's records increase. You can then manipulate the project, by editing, saving, adding new records (by drag and drop from file manager), etc.

Please **note that files should only be added to the project by drag and drop from the file manager**. Trying to add files manually by typing in the edit boxes is prone to error and is **NOT** supported at all.

Open

Open an existing project file. You will be presented with a list of Chief's Installer Pro project (.CPJ) files to select from. You need to close any open project before you can open a new one.

Close

Close an open project. If changes have been made, you will be prompted to save the changes.

Save to file

Any changes made to a project (either new, or old) will not be saved to the disk until you click on this button. It is a good idea to do this fairly regularly, in case something goes wrong.

File Manager

The Project Manager is designed to be operated in conjunction with the Windows file manager. Files should be added to a project **ONLY** by dragging from the file manager (or any   other shell that allows drag and drop) and dropping onto the Project Manager dialog. Clicking on this button tries to load the file manager (WINFILE.EXE).

File Name

When a file is dropped onto the Project Manager dialog, the full path name of the file is placed in the edit box titled **File Name**.

Target Directory

When a file is dropped onto the Project Manager dialog, the proposed target directory of the file is placed in the edit box titled **Target Directory**. Most times, this will be a Chief's Installer Pro **reserved word** (e.g., files dragged from the Windows directory will be marked as going to $WINDIR, and files dragged from the project's working directory will be marked as going to $DEST, etc).

User Option

When a file is dropped onto the Project Manager dialog, it is deemed that the file will be considered mandatory, so the it's **User Option** field is set to zero ("0"). If you want the file to fall under any of your user options, you need to change this field to the number of the relevant user option (and then make sure that the change is updated, either by clicking the Update button, or by making sure that the Auto Update box is checked).

Disk

When a file is dropped onto the Project Manager dialog, it is deemed that the file will be on disk #1 of the installation set, and the Disk # field is set accordingly. If this is not the case, you need to change this field to reflect the relevant disk number, and then update.

Icon Title

When a file is dropped onto the Project Manager dialog, it is deemed that no icon will be created for the file, so this field is empty. If you want an icon to be created for the file, then you need to put the title of the icon in this field, and then update.

Group file

By default, all files are marked as going to the group file pointed to by the **$GROUP** reserved word. If this is not the case, then you need to change this field to reflect the name of the group that you want the file's icon to go into, and then update.

User Options

Clicking on this button presents you with a dialog box containing the default titles for the project's user options (maximum, 10). You can then change the title of any or all of the user options.

Make INF file

This is the main goal of every project - to be able to automatically generate a template INF file for you. The things which are tedious to do are done for you in the template INF file (the project's name, with the extension .INF, or the file pointed to by the **$INF-FILE** entry in the project's **.HDR** file).

Clicking on this button generates the template INF file for the currently open project. This process might take a while, depending on the number of files in the project. When the template has been created, you are then expected to edit it for things like banner names, and other optional reserved words.

See also;
Build the Install Disks
Log the Build History
Compress the Files

Build the Install Disks

Checking this box will cause the Project Manager to try to build you installation disk set after creating the template INF file. You need to have your blanks disks ready if you are going to copy to floppy disk (you can supply a directory on the hard disk as the target directory, when prompted for the target drive/directory).

See also;
Make INF file
Log the Build History
Compress the Files

Log the Build History

The Project Manager can produce a log of every attempt to build the install disks. This log is saved in an ascii file (the project's name, with the extension **.LOG**). The date and time of each build, and the details of the attempts to copy each file in the project, (including whether the files were just copied or were compressed) are logged. If you want a build to be logged, then you need to check this check box. The log file can easily become very big, so it is a good idea to delete it occasionally.

See also;
Make INF file
Build the Install Disks
Compress the Files

Compress the Files

By default, when the install disks are being built, the files are simply copied to the target drive/directory. If you check this check box, then the files will be compressed with Microsoft's **COMPRESS.EXE**. Note that if COMPRESS.EXE is not found in the "PATH", then the attempt to compress the files will fail.

See also;
Make INF file
Build the Install Disks
Log the Build History

Compile File

Chief's Installer Pro obtains all its configuration information from a file called WINSTALL.INF. This file is an ASCII file, which can readily be browed and edited with simple text editors. While this makes it easy for you to edit and maintain, it also makes it easy for hackers to tamper with your install script. By the same token, BATCH files written to work with Chief's Installer Pro and in the same ASCII format, and subject to the same problems.

In order to combat these problems, Chief's Installer Pro comes with a command line compiler for INF and BATCH files, called **COMPILE.EXE**. The same functionality can be obtained by selecting the "Compile File ..." menu option from the AutoCalc main window. The compiler converts the input files into a binary format (different for INF and BATCH files) thereby protecting the files. Chief's Installer Pro can cope with either compiled files or ASCII files, so it does not matter whether or not your INF and BATCH files are compiled.

When you select the "Compile File ..." menu option, a dialog box will appear in which you are prompted for the names of the source and output files. The output file will contain the compiled version of the source file. If the source file is a Chief's Installer Pro batch file, you need to check the checkbox that reads "Source File is a Chief's Installer Pro Batch File" **BEFORE** compiling the file. Trying to compile an INF file as a batch file, or vice versa, will **certainly** lead to errors in running the installer.

Calculate Space Requirements

This facility is provided for backward compatibility with older versions of AutoCalc and Chief's Installer Pro. It does what simply running AutoCalc with no parameters did under version 1.x of Chief's Installer Pro. It's functionality has been replaced in version 2.0 with the functions in the Project Manager.

The purpose of this facility is to calculate the space requirements of your program, before you ship the disks. It takes its input (as AutoCalc did under Chief's Installer Pro, version 1.x) from **WINSTALL.INF** and **AUTOCALC.INI**. Note that it does not open or read any project file at all. It is therefore best used when the WINSTALL.INF file has been finalised, and the program's files have been put in places where AutoCalc can find them (see below for more details).

Specifically, the AutoCalc Space Calculator does the following;

[a] read the $DISK lines in your WINSTALL.INF file

[b] read the $WINDIR lines in your WINSTALL.INF file

[c] read the $SYSDIR lines in your WINSTALL.INF file

[d] read the $TEMPDIR lines in your WINSTALL.INF file

[e] read the $OPTIONAL lines in your WINSTALL.INF file

[f] check for all the files on those lines

[g] obtain the sizes of the files (if a file is compressed, it gets the size of the expanded image of the file)

[h] total the sizes of all the files in the installation set

[i] total the sizes of the files that make up each user option

[j] total the sizes of the files that go into the Windows directory

[k] total the sizes of the files that go into the Windows SYSTEM directory

[l] total the sizes of the files that go into the TEMP directory (for $SWAP-SPACE)

[m] add 0.5% to each of the figures, as a safety margin (because of disk cluster sizes)

[n] display the results on the screen

[o] write the result into WINSTALL.INF

Please try to ensure that the directory in which the files are only contains files that will go on your installation disks - otherwise, the calculations might be wrong. Also, try to be a little specific in your use of wildcards on $DISK lines (especially if your installation set consists of more than 1 disk) - e.g., do please **NOT** do something like;

```
$DISK1=*.*
$DISK2=*.*
$DISK3=*.??_
$DISK4=*.??_
```

This will only confuse the program, and the same files will be processed over and over again. One common cause of inaccurate space calculations by AUTOCALC is the indiscriminate use of wildcard characters. Please NOTE this point.

When AutoCalc has finished calculating the space requirements (the process might take some time) the results are written into WINSTALL.INF. Please note that you should do this BEFORE compiling the WINSTALL.INF file. **Note that this facility may be**

removed in future versions, in favour of the Project Manager.

Locations of files for space calculations;

In respect of matters described above, AutoCalc, when calculating space requirements, defaults to looking for all the files in the CURRENT directory (i.e., the directory being processed). If it does not find the files in that directory, then it will look for them in certain sub-directories **under the directory tree** of the current directory, or in directories pointed to in the AUTOCALC.INI file (the latter only applies to files going into $WINDIR, $SYSDIR, and $TEMPDIR - please see the help index of CHIEF.HLP for what these things mean).

The supported directories under the directory tree of the current working directory are:

1. Files going to the Windows directory: **$WINDIR**

   OR the directory pointed to by the **$WINDIR=** setting in **AUTOCALC.INI**

   e.g.,

   [a]. **C:\MYPROGRAM\$WINDIR**
   [b]. **$WINDIR=xxxx** in AUTOCALC.INI

2. Files going to the Windows SYSTEM directory: **$SYSDIR**

   OR the directory pointed to by the **$SYSDIR=** setting in **AUTOCALC.INI**

   e.g.,

   [a] **C:\MYPROGRAM\$SYSDIR**
   [b]. **$SYSDIR=xxxx** in AUTOCALC.INI

3. Files going to the TEMP directory: **$TEMPDIR**

   OR the directory pointed to by the **$TEMPDIR=** setting in **AUTOCALC.INI**

   e.g.,

   [a] **C:\MYPROGRAM\$TEMPDIR**
   [b]. **$TEMPDIR=xxxx** in AUTOCALC.INI

4. Files for each $DISK# line: **$DISK#**

   OR the directory pointed to by the **$DISK#=** setting in **AUTOCALC.INI**

   e.g.,

   [a] **C:\MYPROGRAM\$DISK1**
   [b]. **$DISK2=xxxx** in AUTOCALC.INI

NOTES:

1. AutoCalc will first look in the directory tree of the source directory for the relevant sub-directory. If the sub-directory does not exist there, then AutoCalc will look in the AUTOCALC.INI file for any relevant entry. If this is not found, or if it points to an invalid directory, then AutoCalc will look in the source directory.

2. AutoCalc sometimes gives you a negative value for $SPACE (or any of the others). In such cases, simply replace the negative value with a ZER0.

3. Sometimes the space calculations produced by AutoCalc are not accurate. There are many causes for wrong calculations by AUTOCALC. The primary cause is indiscriminate use of wildcards in your INF file. Other causes, which might be related to the one already mentioned are:

> [a] Mixing wildcards and full file names on $DISK, $SYSDIR, $WINDIR, $TEMPDIR, and $OPTIONAL lines - this might result in some files being processed more than once.
>
> [b] Having files in the directories being processed by AUTOCALC which files are not going to be on your distribution disks.

The bottom line is this - if you are going to use wildcards on your $DISK and/or $OPTIONAL lines, you need to think very carefully about what you are doing. It is up to you to arrange your lines so that no file is liable to be processed twice - this is because AUTOCALC processes your INF file **exactly** as it finds it.

In my view, it advisable to AVOID mixing wildcards and full file names on $OPTIONAL lines. Either use ONLY wildcards, or ONLY full file names. But if you are NOT relying on AutoCalc for your space calculations, then you can freely mix wildcards and full file names.

Convert VB Setup file

This presents a facility for Visual BASIC programmers who have been using the Setup Wizard to create their installation routines. The facility takes a Visual BASIC Setup Wizard .VBZ file as its input, and converts this into a Chief's Installer Pro project file. You can then manipulate the new project file with the Project Manager as normal.

By default, the new Chief's Installer Pro project is simply called "CONVERT". You need to rename it to something more meaningful before performing the conversion. Note that you should not give any file extension for the name of the converted project. Any project name supplied as the Chief's Installer Pro project will automatically have the extension **.CPJ** for the project, **.HDR** for the project's header file, and **.INF** for the project's template INF file.

Note that this is only a facility to simplify the process of moving from Setup Wizard. Most things (e.g., file lists) are converted, but some things may be missed out. You might still need to edit the new project afterwards.