

# Cliplt Version 1.0

---




# User Guide

## Copyright Information

---

© Copyright 2001, PaperClip Software, Inc. The PaperClip and CliplIt product name and PaperClip logo are registered trademarks of PaperClip Software, Inc. All brand and product names are registered trademarks of their respective corporations.



## Contents

---

Overview .....	4
Operation .....	4
Environment .....	4
Supported Application User Interfaces .....	4
Configuration.....	4
Integration with a Target Application .....	5
Overlay Templates .....	5
Database/Spreadsheet Overlays .....	5
Dialog Box Overlays .....	5
Dial-Up, Terminal Emulation, and Free-Form Text Overlays .....	6
Anchors .....	6
Creating Overlay Templates .....	7
Overlay Template Maintenance.....	25
Overlay Template Distribution .....	28
Keystroke Scripts .....	29
Select an Alternate Set of Keystrokes to Copy Data to the Clipboard.....	29
Automating Data Highlighting prior to Copying It to the Clipboard.....	30
Troubleshooting Overlay Creation.....	31
Database/Spreadsheet Overlays .....	31
Dialog Box Overlays.....	32
Dial-Up, Free-Form, or Terminal Emulation Overlays.....	32
DOS Boxes Running on Windows 2000/NT .....	32
Integrating Cliplt to Your Host Application .....	33
Scripting .....	33
Programming Environment .....	33
Programming Interface (API).....	35
Using Cliplt .....	37
Cliplt Hotkey (VCP Key) .....	37
User Preferences.....	37
Running Cliplt Upon System Startup .....	38
Appendices .....	39
System Diagram.....	39
Example Post Processing Script.....	40
Example Visual Basic Host Program .....	41
Command Line Switches .....	43

## Overview

---

ClipIt is an application and software toolkit that allows the seamless integration of two or more application software packages or systems without the need for programming. ClipIt is built on PaperClip Software's unique Visual Context Processor (VCP) technology used in the PaperClip32 document management system. VCP is a transparent technology that allows applications to share meaningful data to create "links" between themselves. The premise here is that it is the actual data that is important in establishing a link and not a programmed interface. Programs mature and change but the data remains (eg; A person's name, social security number, etc.). PaperClip's VCP technology provides application developers an independence from programmed links thus being able to provide integrated solutions with applications not otherwise possible.

## Operation

---

---

Users select a record as they normally would within their (target) application, whether that record is a row in a database, a record in form-view, or any other fixed format. In a single keystroke, users can invoke ClipIt, which accesses that record's associated record in the ClipIt enabled (host) application. ClipIt automatically recognizes different application screens, captures data entered or displayed on those screens then maps and inserts the data into the host application via script or program as defined by the user. Target application setup will require only a few keystrokes or mouse clicks to accomplish.

## Environment

---

ClipIt is application server and back-end database independent. ClipIt is a PC network client-based system. Workstations using ClipIt must run Microsoft Windows 9x/me or NT/2000.

## Supported Application User Interfaces

---

---

The user interfaces supported by ClipIt are as follows:

- Windows list view style containing database records or spreadsheet like data format.
- Windows dialog boxes containing text data controls.
- Free text windows containing text only such as terminal emulator (green screen) products.
- DOS windows running DOS mode applications under Windows 9x/me or NT/2000.

## Configuration

---

---

ClipIt can be setup to run on a single desktop or in multi-user LAN configuration. To run in a LAN configuration all ClipIt users must have shared access to the ClipIt Templates directory. In this way the system administrator can setup overlay templates for all users in one place. For LAN use you install the ClipIt program files on each client workstation with only the ClipIt Templates directory being shared. The installation program has an option to allow you to reference an already existing Templates folder. When ClipIt runs it reads all of the overlay templates into memory for optimal performance. Therefore, it is not advantageous to install the overlay templates on each client workstation.

If your configuration does not allow shared access to a common folder (eg; web clients, etc.) Cliplt then must be deployed as individual workstation installations. Each of these installations will then have their own overlay template files. In this case a system administrator can “publish” overlay templates to each user via Cliplt’s overlay template Import/Export feature.

## **Integration with a Target Application**

---

Cliplt enabled host applications can be setup to integrate to any DOS or Windows target application by using a simple point and click user interface. No programming is required for this setup. This setup would be typically done at the site where the target system is being used and performed by an administrator of the system.

### **Overlay Templates**

---

---

An Overlay Template must be defined in order to acquire data from a target application screen. The template enables Cliplt to extract the data from the application so that it can be mapped into the host application via VBScript or programmed interface.

Cliplt’s Overlay Template Manager allows you to identify the screens and data fields on the target application to be connected with. Any data visible on the screen can be acquired and mapped. Static data on the screen can be used to “anchor” or identify the screen. This identification is used to determine what overlay template to be used thereby allowing Cliplt to map the proper data fields. The overlay templates need to be defined only once per application screen and are shared for all network users of Cliplt. This allows centralized control of the Cliplt system. Non-authorized users have restricted use of the Overlay Template Manager as it is an admin function. Cliplt can be setup to transfer data between the target and host application in 20 minutes or less.

There are three different types of overlay templates each representing a specific application type.

#### **Database/Spreadsheet Overlays**

---

---

Most database applications use a form view for data entry and retrieval (i.e., you see an entire form or full screen of information). Often, they also offer a tabular view for examining multiple records in row-and-column format.

This overlay type is used to clip to most database applications (in either form or tabular view), as well as to spreadsheets which also contain columns and rows of data.

As the name implies, the Database/Spreadsheet Overlays apply to most common, commercially available database and/or spreadsheet Windows applications.

#### **Dialog Box Overlays**

---

---

This overlay type is used to clip a Windows dialog box or a form with a set of fixed location, editable fields. These applications typically display one record at a time in a dialog box in the application window. Some database applications in form view (for example, applications created with SQL Windows), off-the-shelf accounting applications (for example, Simply Accounting for Windows), as well as custom-built applications,

which are created with programming languages like Visual Basic, can be set up as Dialog Box overlays.

## **Dial-Up, Terminal Emulation, and Free-Form Text Overlays**

---

Free-form text applications (e.g., word processors), and applications accessed via Windows terminal emulation (green screen) are clipped in the same manner. In order to get data from one of these applications into PaperClip32, you must first highlight the appropriate data in that application. Such applications do not contain preset data fields or edit controls in a fixed location. Thus, you must tell PaperClip32 what data to extract.

Terminal Emulation	Windows-based terminal emulation programs are applications used to connect PCs to other PCs, minicomputers, mainframes, and information services which run text mode user interfaces (green screen)
Free-Form Text	Any application that manages free-form text, such as a word-processor or text editor applications are in the category of free-form text applications.

In order to create an overlay template for a terminal emulation or free-form text application, select the data in the other application by highlighting it and then use the Overlay Hot Key (Ctrl F2) or the VCP hot key (Ctrl F3). The keystrokes required to highlight the appropriate data can be automated with a ClipIt Keystroke Script.

### **Setting Up Overlay Templates**

Logically, because there are differing overlay templates types, the procedures for defining each varies slightly. Some applications, which are not terminal emulation or free-form text, might also not be easily distinguishable as either the Database/Spreadsheet or Dialog Box type. If this is the case with your application, as a general rule, try using the Database/Spreadsheet type first and then try the Dialog Box type. Try experimenting with a few sample documents first to see which type works best for you.

Before beginning overlay template setup for an application, you should:

- Verify that there are no conflicts between the ClipIt overlay hot key (Ctrl+F2) and/or the VCP hot key (Ctrl+F3) sequence and the selected application. If there is a conflict, change the hot key sequences.

Then select one of the following:

1. Setting Up Database/Spreadsheet Overlays
2. Creating Dialog Box Overlays
3. Creating Terminal Emulation, and Free-Form Text Overlays

## **Anchors**

---

**Note:** If you set up an initial template without anchors and later discover that additional overlay templates must be defined for these applications, you should delete the original template and re-create it with anchors.

---

For certain overlay types (i.e., Dialog Box, Dial-Up, Free-Form, and Terminal Emulation overlays), you should identify anchors on the target application window itself. Anchors are regions of text or specific controls that help Cliplt identify the window context. By contrast, Spreadsheet/Database overlays distinguish window context by means of the window's title bar. For such overlays, anchors are unnecessary.

### **Selecting Anchors**

When you define anchors, select ones that are unique to the application window and that will always be present when the window is visible regardless of the particular record in view. Select text, such as label fields, that do not change when moving between records. In addition, anchors must be different for each screen for which overlays are to be defined.

---

**Note:** Information on the screen such as the application title, controls (e.g., buttons and edit controls), or field names (e.g., LASTNAME) will probably not change position, nor is the text likely to change. Conversely, specific field values, such as SMITH, are not appropriate anchors, since the value changes from record to record.

---

## **Creating Overlay Templates**

---

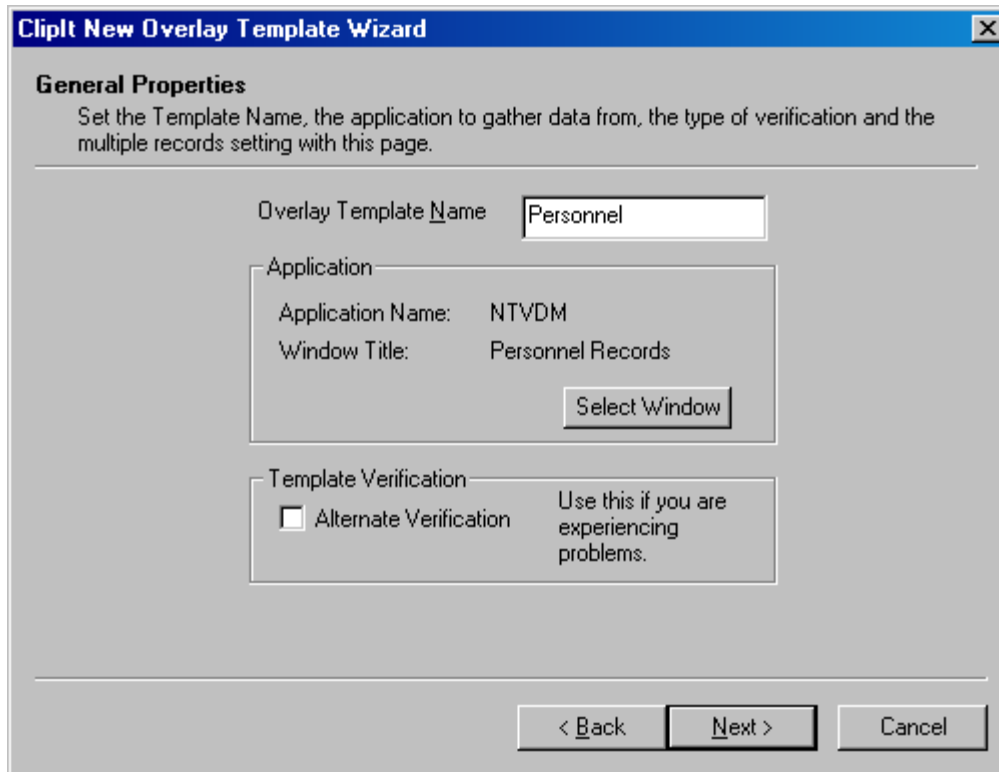
---

This section describes the procedures for creating overlay templates for each of the three target application types. These three application types represent all of the types of applications that can run in any version of Windows. Determine which one of the three application types your target application falls into and follow the appropriate procedure for creating an overlay template.

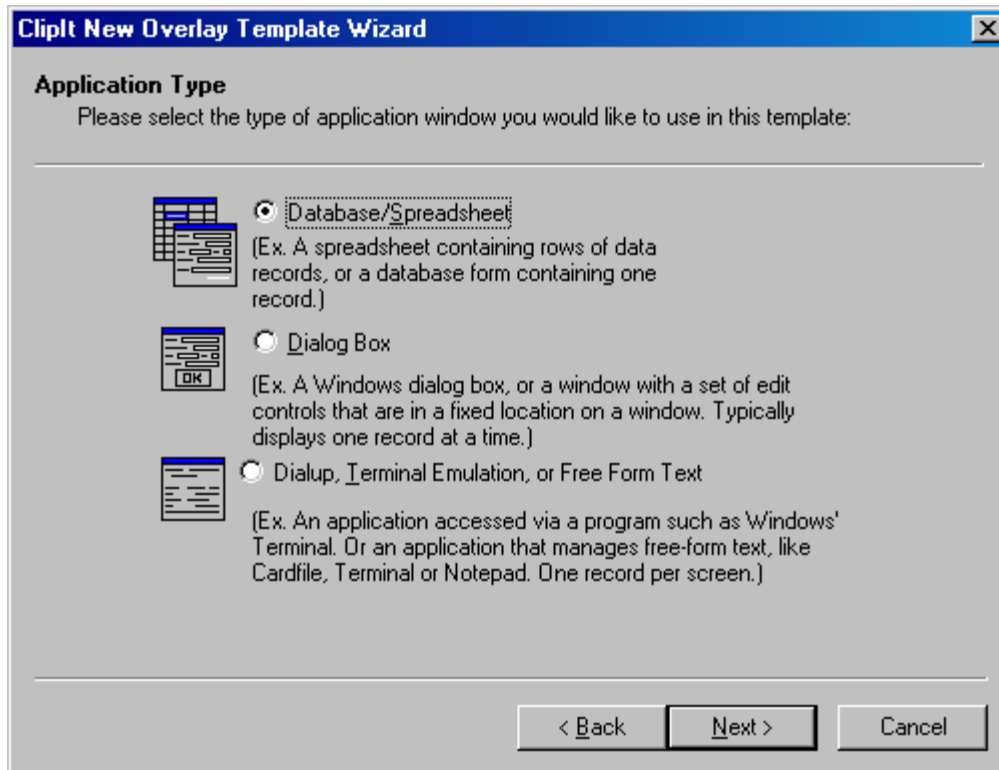


### **Set Up a Database/Spreadsheet Overlay Template**

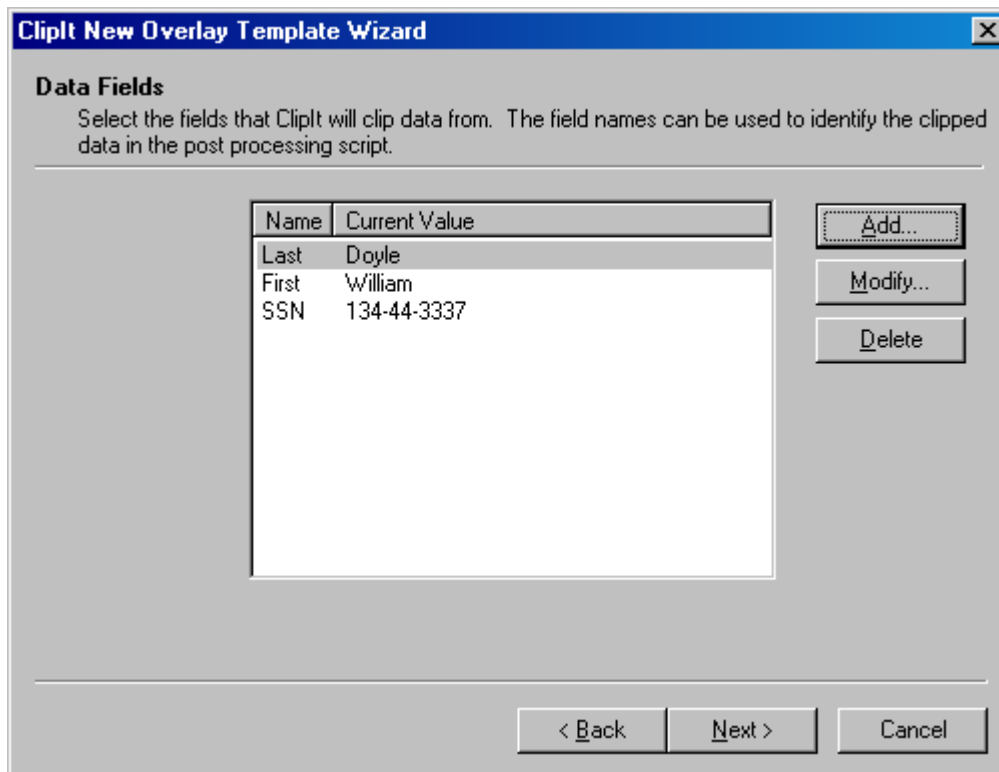
1. Make sure that Cliplt is loaded and running. See the Cliplt icon in the Windows task tray.
2. Launch your target application.
3. In that application, go to the screen displaying the data for which you wish to build a template. (If this screen displays multiple records, as in tabular view, highlight one record to serve as a sample upon which to build a template.)
4. Invoke the New Overlay Hot Key by pressing Ctrl+F2. The Cliplt Overlay Template Wizard appears. The first page is a welcome page. Press Next to continue and the General Properties page appears.



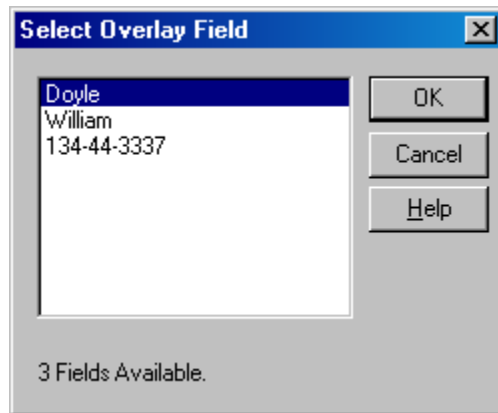
5. Press Select Window to select the application screen for this template. The Select Application Screen dialog box briefly disappears and your application window is active again.
6. Select the title bar of your application screen. The Select Application Screen dialog box reappears with the program name and window title filled in.
7. Enter the Overlay Template Name. Use a name that best describes the application screen that it is being defined for.
8. Press the Next button to continue.



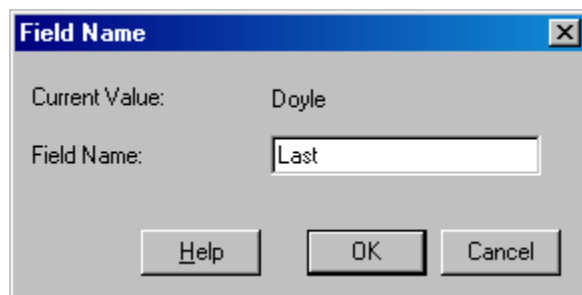
9. Select Database/Spreadsheet and choose Next button. The Data Fields dialog box appears. For Database/Spreadsheet type templates Anchors are not needed so the Anchor dialog box is omitted.



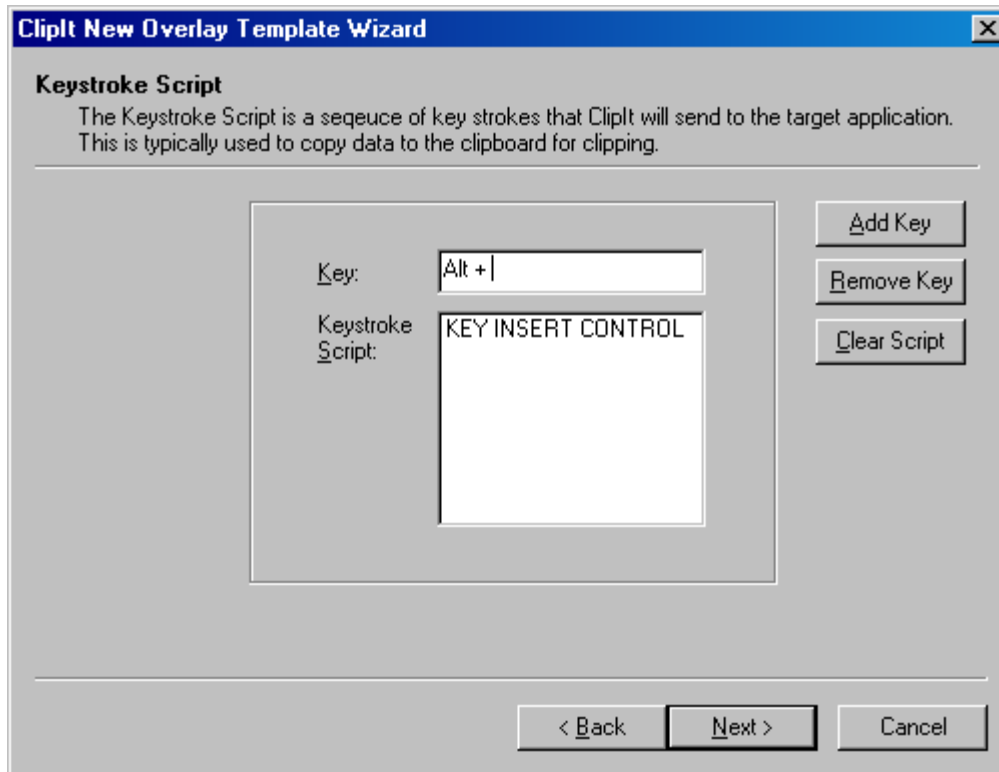
- Click the Add button to display the Select Overlay Field dialog, showing the record of data you highlighted in your application. Consider the following example.



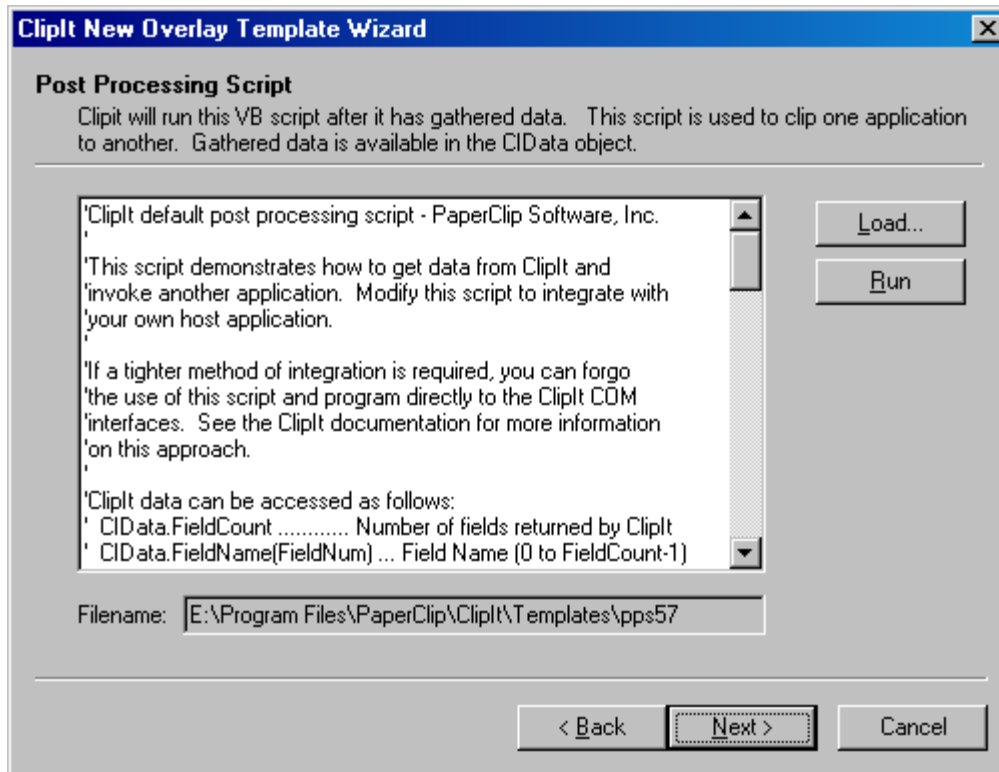
- Click one of the listed overlay fields and click OK. This displays the Field Name dialog, as shown in the following example.



- Enter a name for this field. This name can later be used to reference the data from a script or your host application.
- Click OK to add the field and close the dialog. This returns you to the Data Fields dialog, with your data item displayed in the field list. Repeat steps 10 – 13 until you have finished adding all of the data fields that you want to capture.
- Click Next from the data fields dialog box and the Keystroke Script dialog box is displayed.



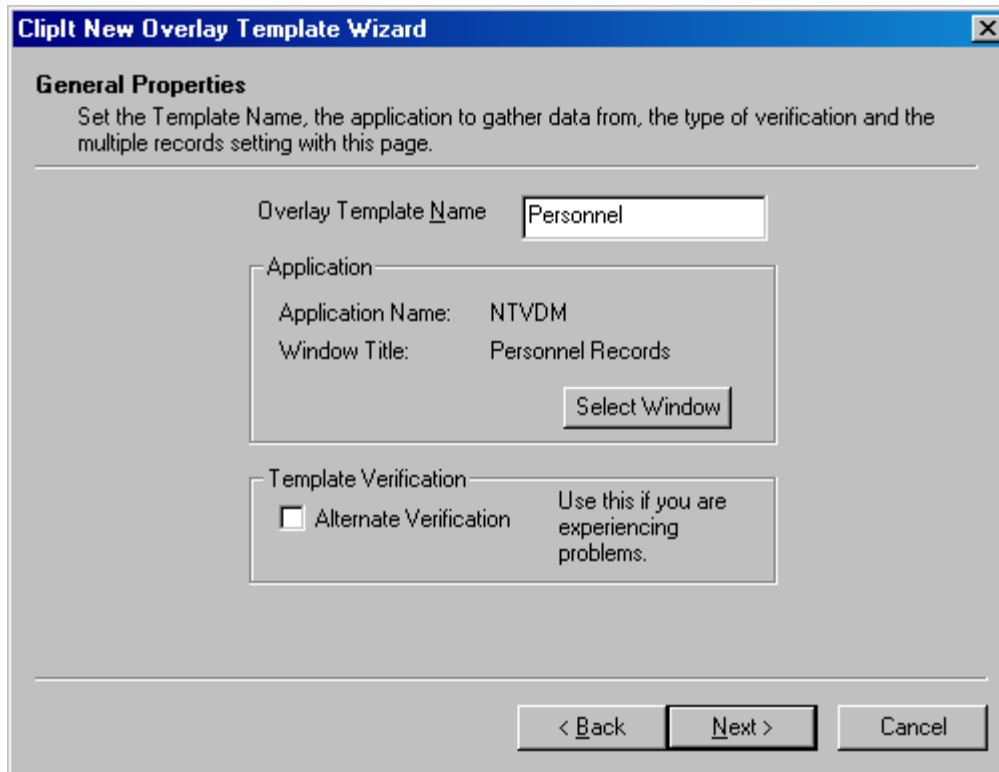
15. The default keystroke script sends a Ctrl + C to the target application. If your application requires different keystrokes for selecting and/or copying data (see your application's "Edit menu") then change the key sequence by hitting the key or key combination (eg; Alt + E) then press the Add Key button.
16. Press Next to continue to the Post Processing Script dialog.



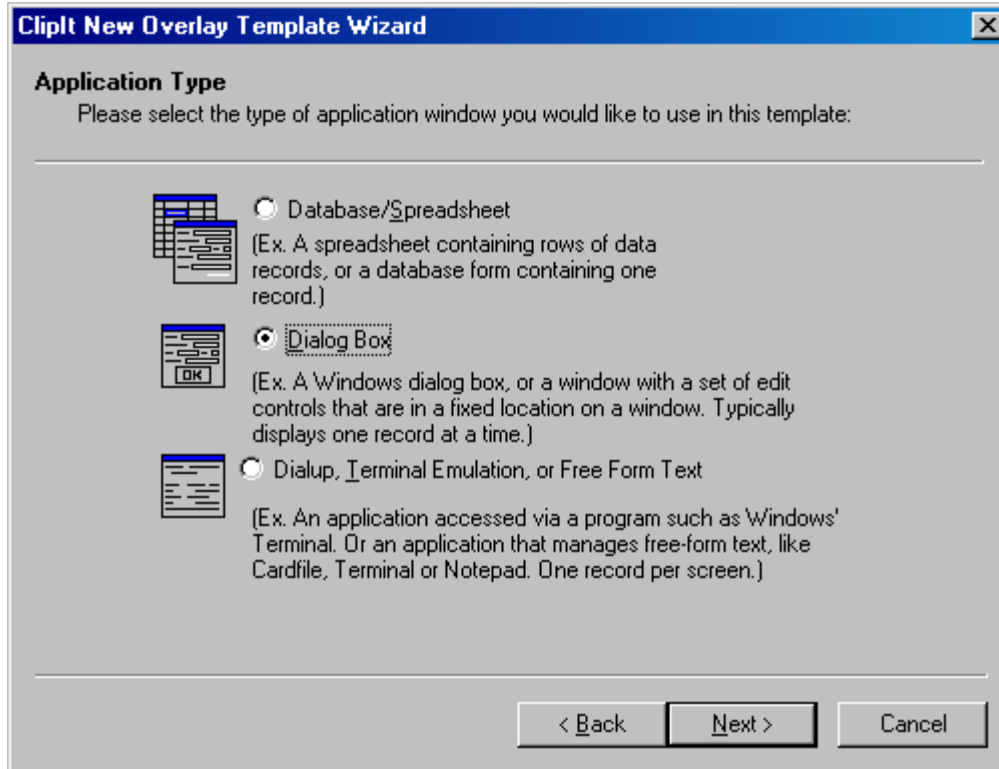
17. The default post processing script is automatically loaded. Use this script to test your overlay template. You will need to change this script to integrate with your own host application. If you are using a programmed approach, this script will not be used. See the section on Integrating with your host application for more details.

18. Press Next to continue.



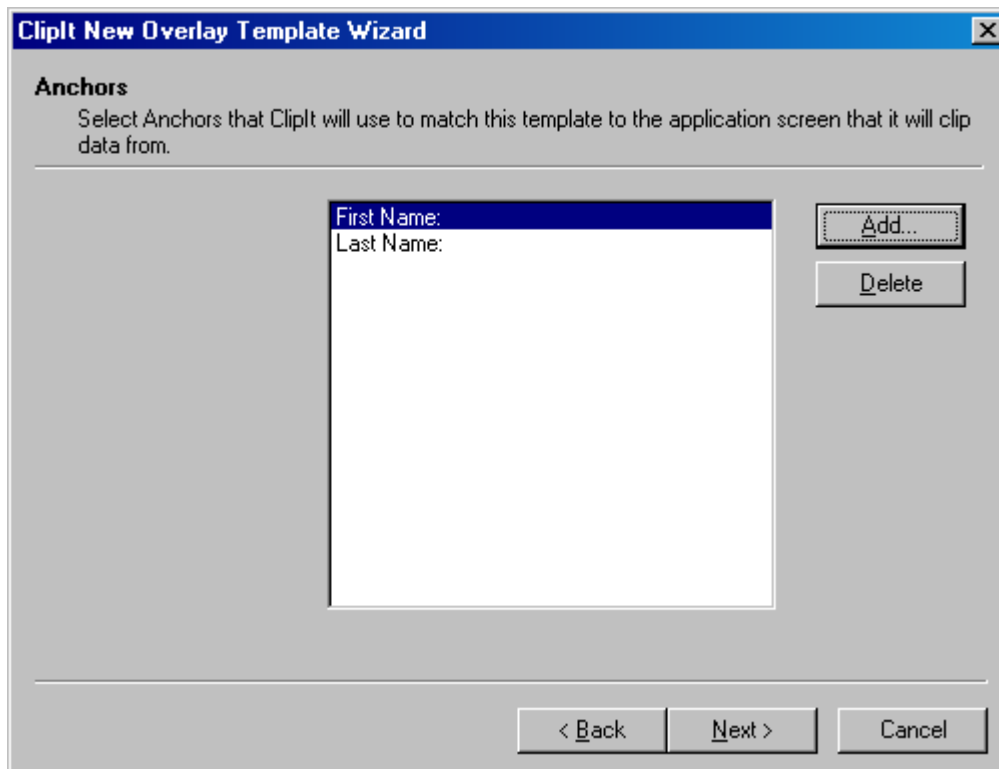


5. Press Select Window to select the application screen for this template. The Select Application Screen dialog box briefly disappears and your application window is active again.
6. Select the title bar of your application screen. The Select Application Screen dialog box reappears with the program name and window title filled in.
7. Enter the Overlay Template Name. Use a name that best describes the application screen that it is being defined for.
8. Click Next to proceed to the Application Type dialog box.

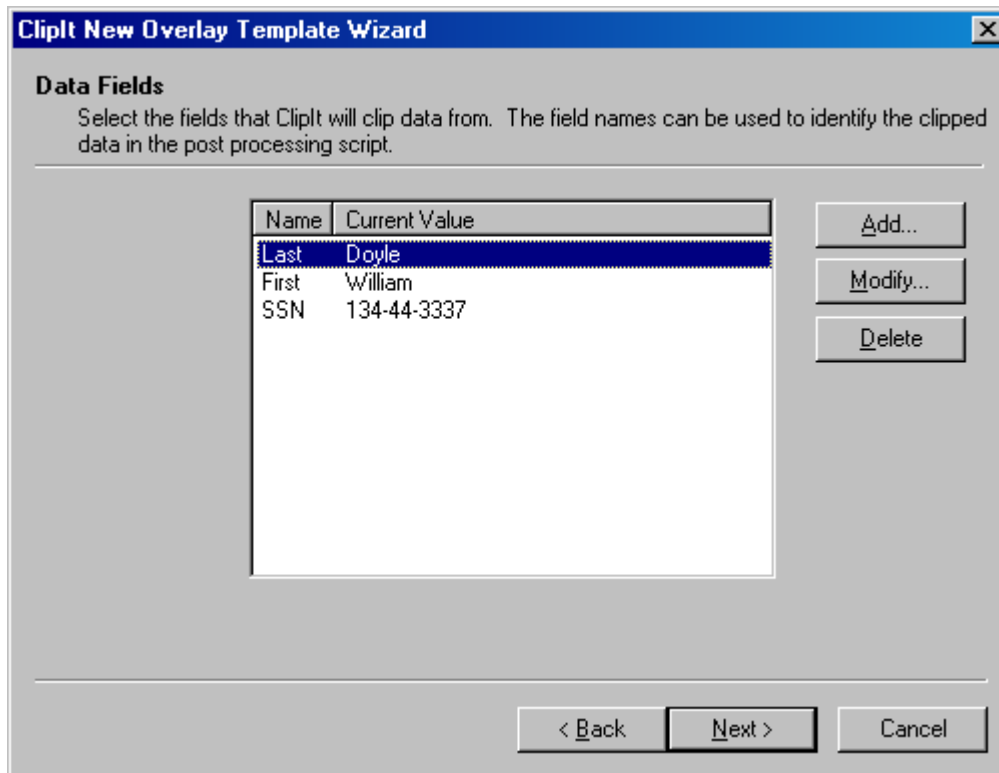


9. Select Dialog Box type.

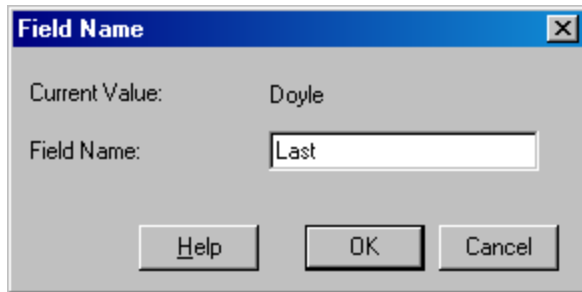
10. Click Next. The Anchors dialog box displays.



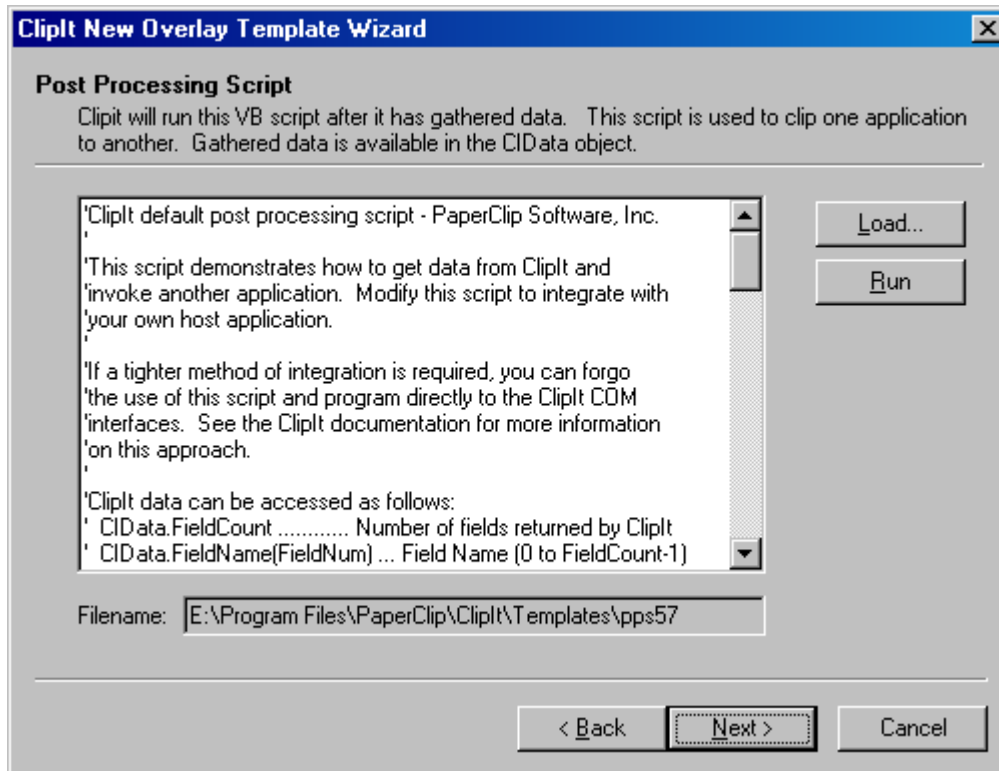
11. If you are planning to create only one overlay for this application, click Next to proceed without defining any anchors for this window. The system warns you that since you are not defining anchors, you will only be able to create one overlay template for this dialog box application.
  - Click Yes to proceed. Cliplt displays the Data Fields dialog box. Proceed to step 13.
  - If more than one overlay will be created for this application, click Add to define anchors for this dialog box. Your target application screen appears.
12. Select the appropriate text or screen area for this anchor from your application screen. Cliplt inserts that text into the anchors dialog box.
  - One or two anchors per overlay should be sufficient. Click Add to add additional anchors, or Next when you have finished adding anchor fields.
  - When you click Next, The Data Fields dialog box appears.



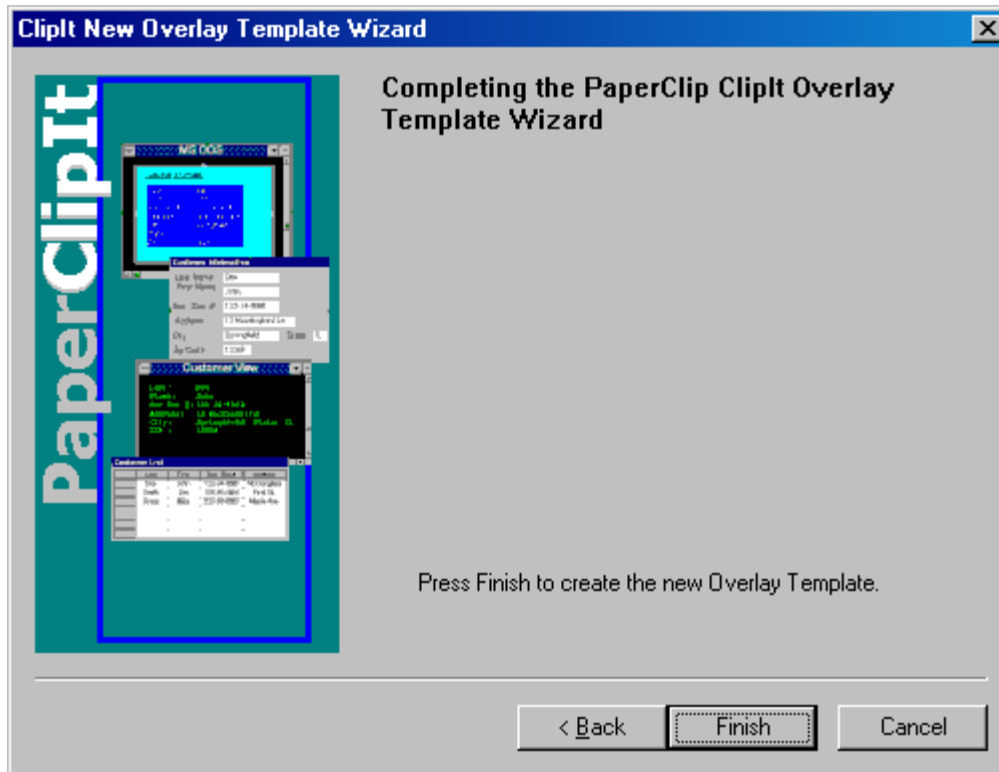
13. Click Add. Cliplt displays the application window itself; and the mouse pointer is changed to a cross hair. Select a data field in the application window and the Field Name dialog box appears.



14. Enter a name for this field. This name can later be used to reference the data from a script or your host application.
15. Click OK to add the field and close the dialog. This returns you to the Data Fields dialog, with your data item displayed in the field list. Repeat steps 13 - 15 until you have finished adding all of the data fields that you want to capture.
16. Click Next from the Data Fields dialog box and the Post Processing Script dialog box is displayed.



17. The default post processing script is automatically loaded. Use this script to test your overlay template. You will need to change this script in order to integrate with your own host application. If you are using a programmed approach, this script will not be used. See the section on Integrating with your host application for more details.
18. Press Next to continue.



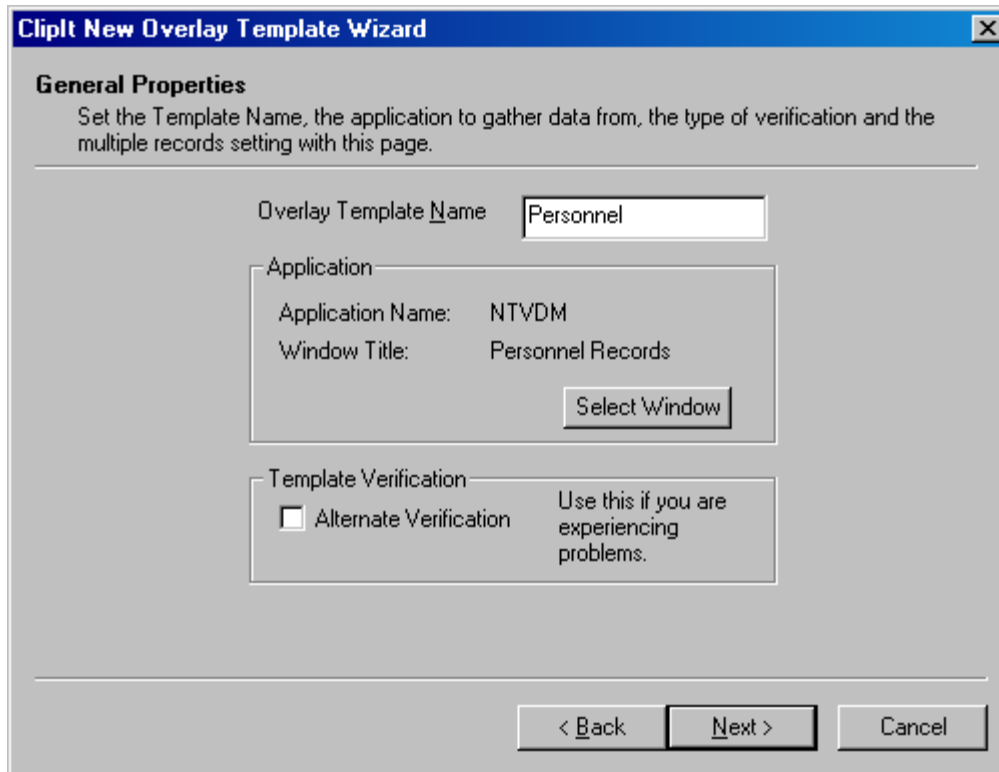
19. Press Finish to complete the overlay template creation.
20. You can now test your new template using the default post processing script. Go to a data record in your target application and press Ctrl+F3 the default Cliplt (VCP) hot key.



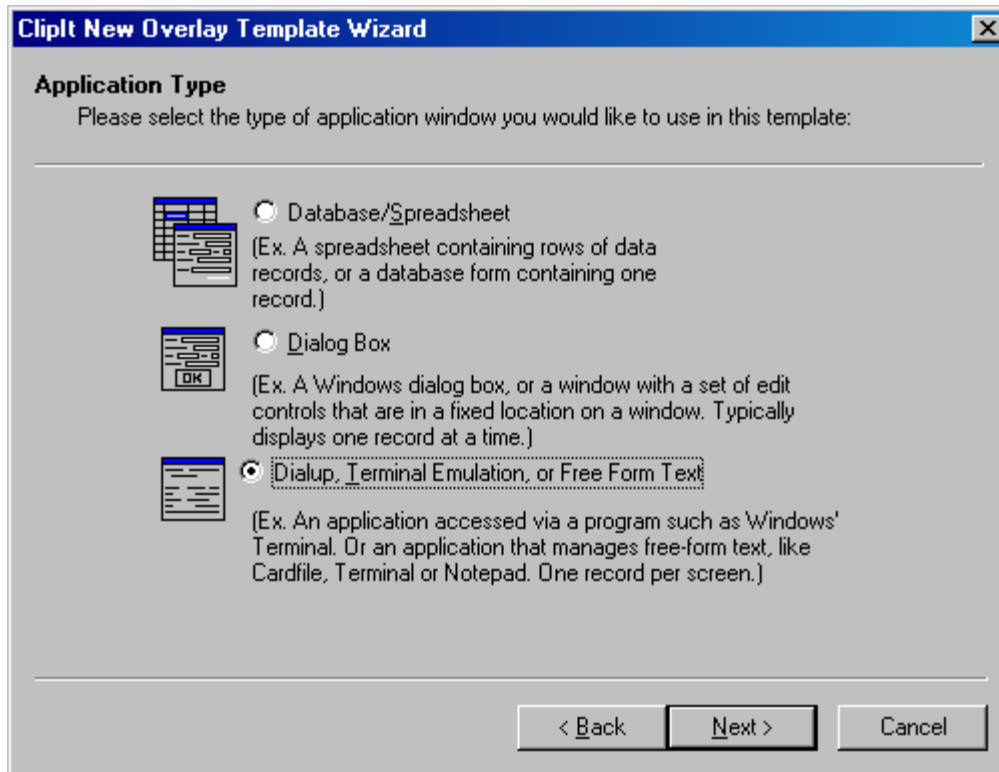
### **Set Up a Dial-Up, Terminal Emulation, or Free-Form Text Overlay Template**

Overlay templates for dial-up, terminal emulation, or free-form text applications instruct Cliplt to map to precise screen locations. This template extracts its index fields from the highlighted row-and-column screen coordinates.

1. Make sure that Cliplt is loaded and running. See the Cliplt icon in the Windows task tray.
2. Launch your target application.
3. In that application, go to the screen displaying the data for which you wish to build a template. Highlight the entire record, or the area of data for which you want to create the overlay.
4. Invoke the New Overlay Hot-Key by pressing Ctrl+F2. The Cliplt Overlay Template Wizard appears. The first page is a welcome page. Press Next to continue and the General Properties page appears.

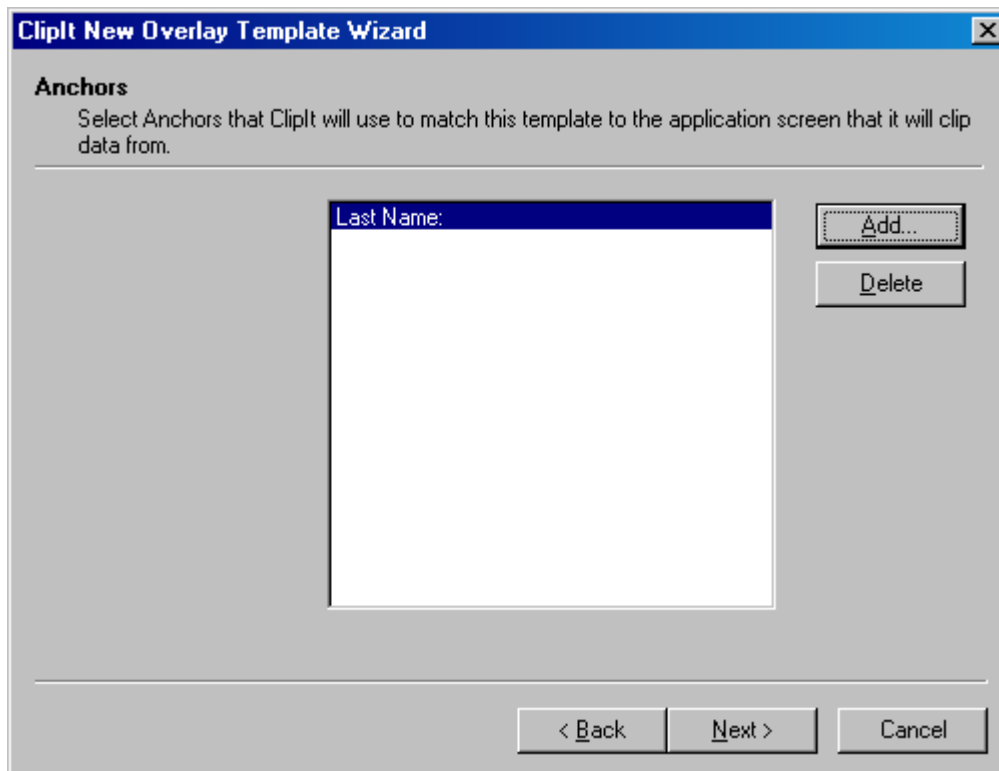


5. Press Select Window to select the application screen for this template. The Select Application Screen dialog box briefly disappears and your application window is active again.
6. Select the title bar of your application screen. The Select Application Screen dialog box reappears with the program name and window title filled in.
7. Enter the Overlay Template Name. Use a name that best describes the application screen that it is being defined for.
8. Click Next to proceed to the Application Type dialog box.

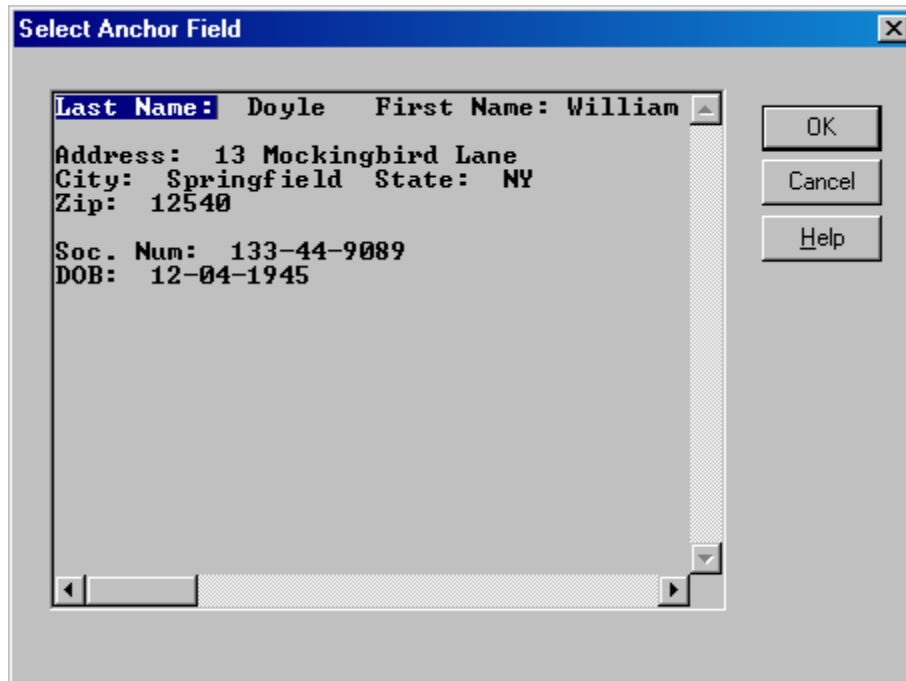


9. Select Dialup, Terminal Emulation, or Free-Form Text and click OK.

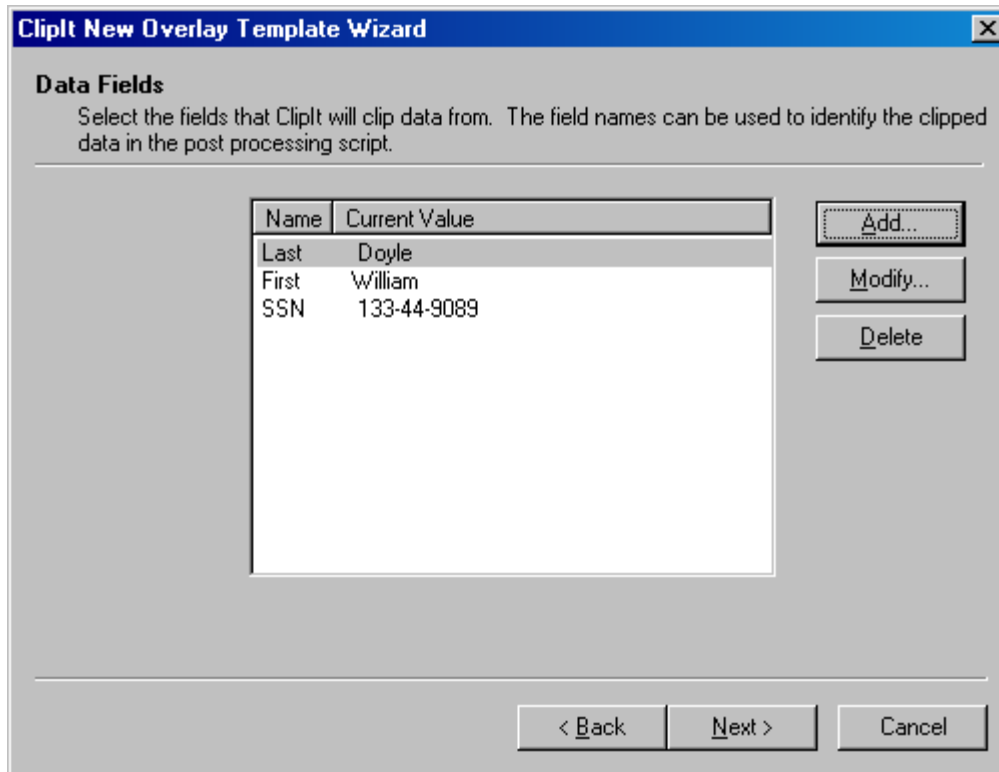
10. Click Next. The Anchors dialog box appears:



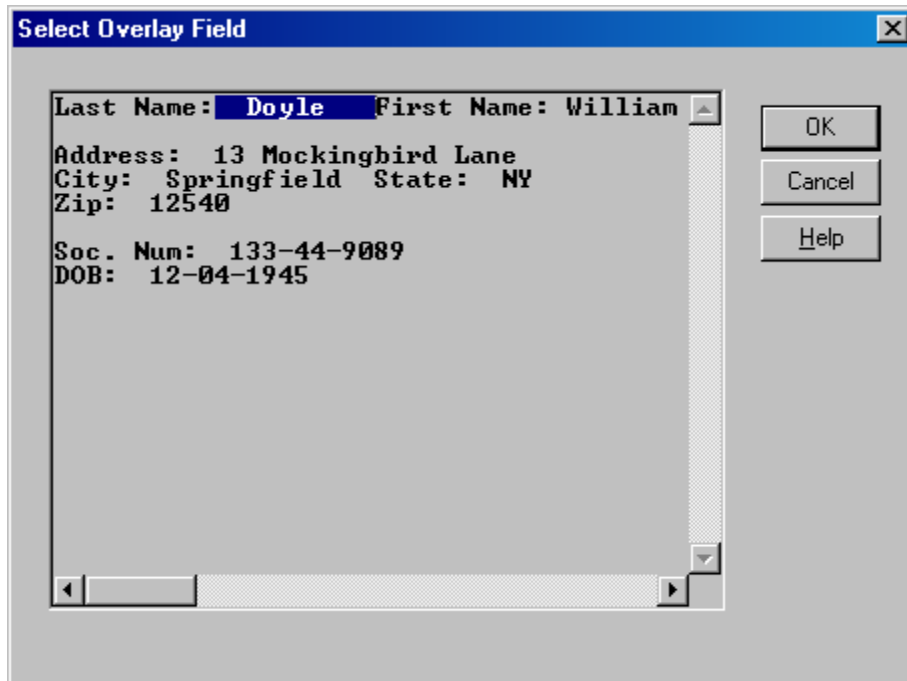
11. If you are planning to create only one overlay for this application, click Next to proceed without defining any anchors for this window. The system warns you that since you are not defining anchors, you will only be able to create one overlay template for this dialog box application.
  - Click Yes to proceed. Cliplt displays the Data Fields dialog box. Proceed to step 12.
  - If more than one overlay will be created for this application, click Add to define anchors for this screen. The Select Anchor Field dialog box appears with your application's screen data.



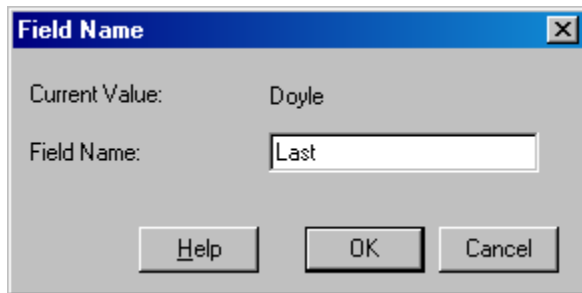
12. Highlight the text to be used as an anchor from your application screen then Click OK. Cliplt inserts the selected text into the anchors dialog box.
  - One or two anchors per overlay should be sufficient. Click Add to add additional anchors, or Next when you have finished adding anchor fields.
  - When you click Next, The Data Fields dialog box appears.



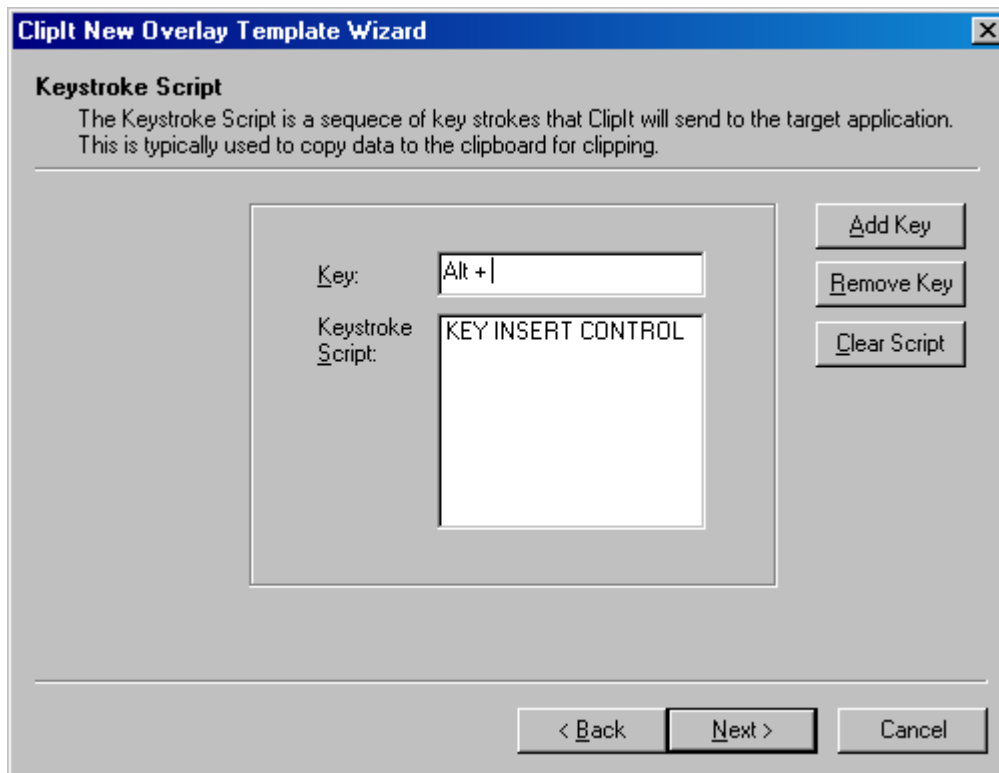
13. Click Add. The Select Overlay Field dialog box appears with your application's screen data.



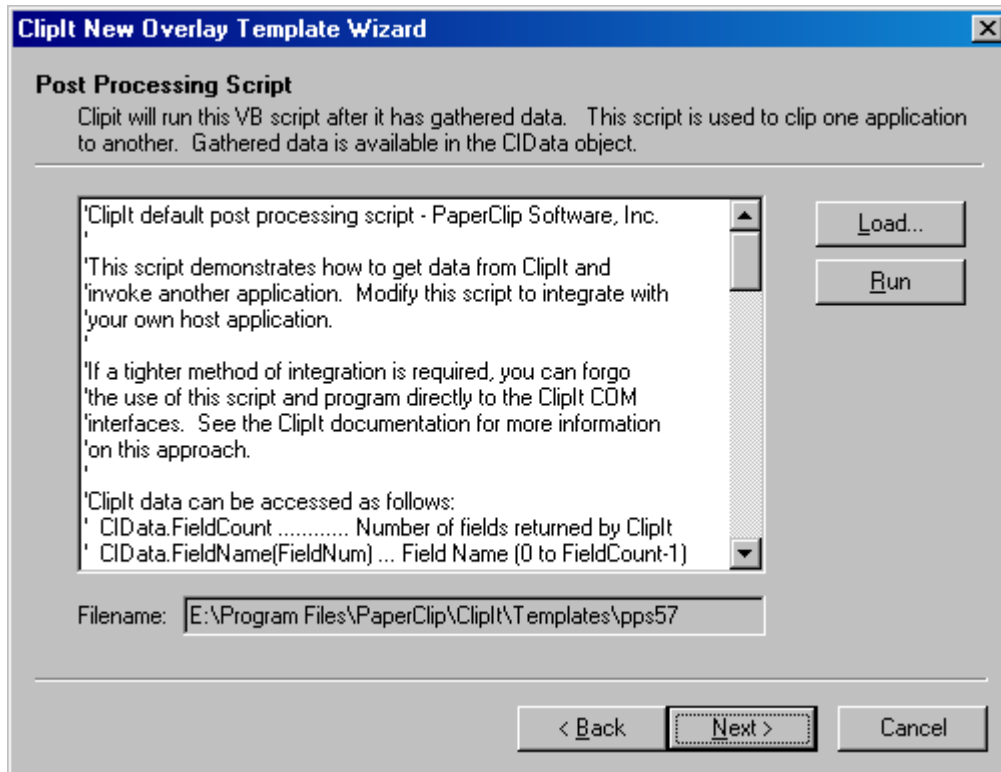
14. Highlight the data field including any blank space that may be before or after the data value so that Cliplt can capture the maximum length of the data field. Click OK and the Field Name dialog box appears.



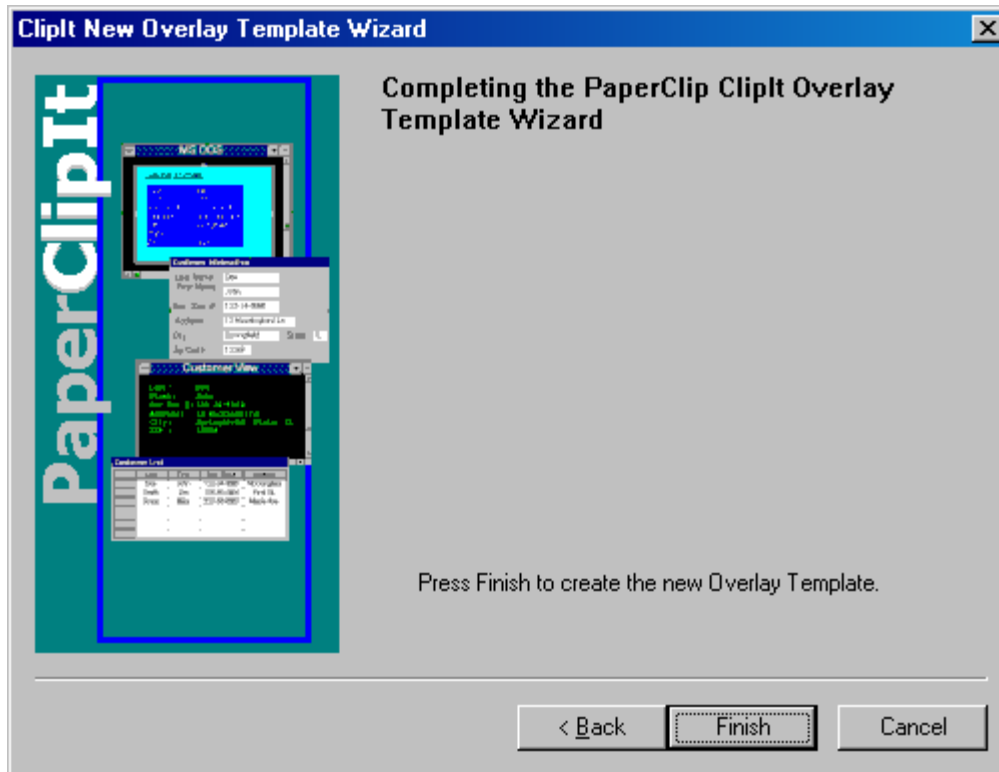
15. Enter a name for this field. This name can later be used to reference the data from a script or your host application.
16. Click OK to add the field and close the dialog. This returns you to the Data Fields dialog, with your data item displayed in the field list. Repeat steps 13 - 16 until you have finished adding all of the data fields that you want to capture.
17. Click Next from the Data Fields dialog box and the Keystroke Script dialog box is displayed.



18. The default keystroke script sends a Ctrl + C to the target application. If your application requires different keystrokes for selecting and/or copying data (see your applications "Edit menu") then change the key sequence by hitting the key or key combination (eg; Alt + E) then press the Add Key button.
19. Click Next from the Keystroke dialog box and the Post Processing Script dialog box is displayed.



20. The default post processing script is automatically loaded. Use this script to test your overlay template. You will need to change this script to integrate with you own host application. If you are using a programmed approach, this script will not be used. See the section on Integrating with your host application for more details.
21. Press Next to continue.



22. Press Finish to complete the overlay template creation.
23. You can now test your new template using the default post processing script. Go to a data record in your target application and press Ctrl+F3 the default Cliplt (VCP) hot key.

## Overlay Template Maintenance

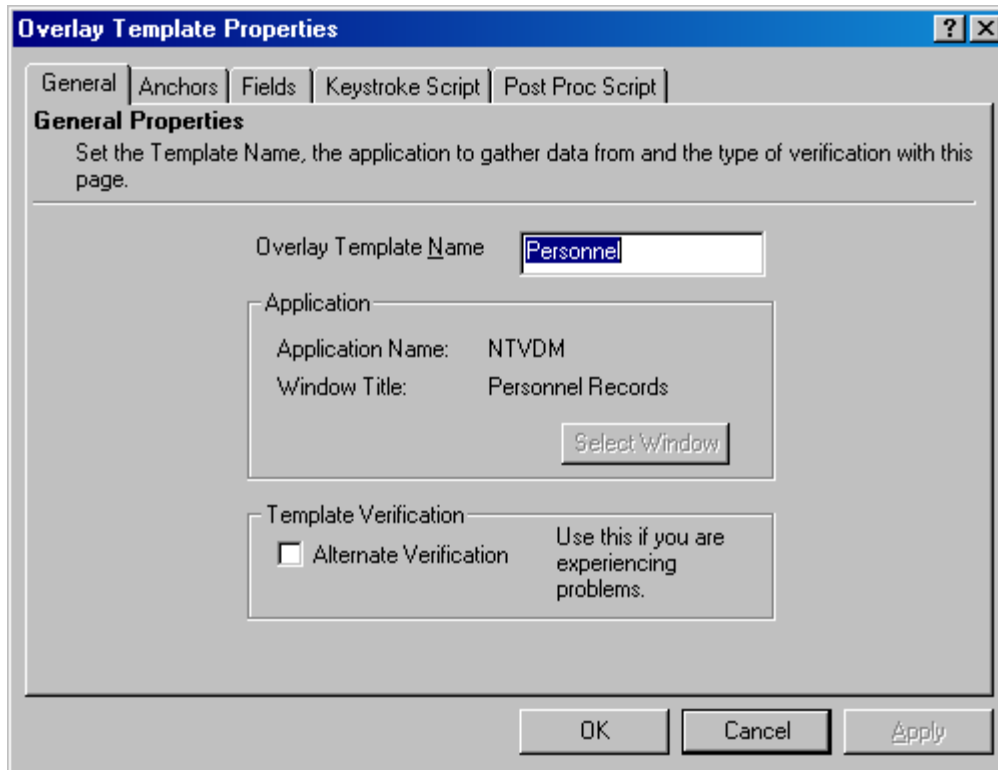
---

This section describes how to maintain overlay templates after they are created. The properties of an overlay template that pertain to the application screen information cannot be changed once an overlay is created. If you need to modify information such as anchors, field locations, application type, etc. you must re-create the overlay template.



### *Modify or View Overlay Template Properties*

1. Select Overlays... from the Cliplt menu. The Overlay Templates dialog box appears.
2. Select the desired overlay template and click Properties. The Overlay Template Properties dialog box appears.



3. The overlay template name, field names, keystroke script, post processing script can be modified. Other overlay template properties such as the position of the fields, selected application screen can only be viewed. In order to modify these other properties you should delete, then recreate your overlay template.



### *Delete an Overlay Template*

---

**Note:** Deleting an overlay template removes the ability to clip to the visual context for which the template was created.

---

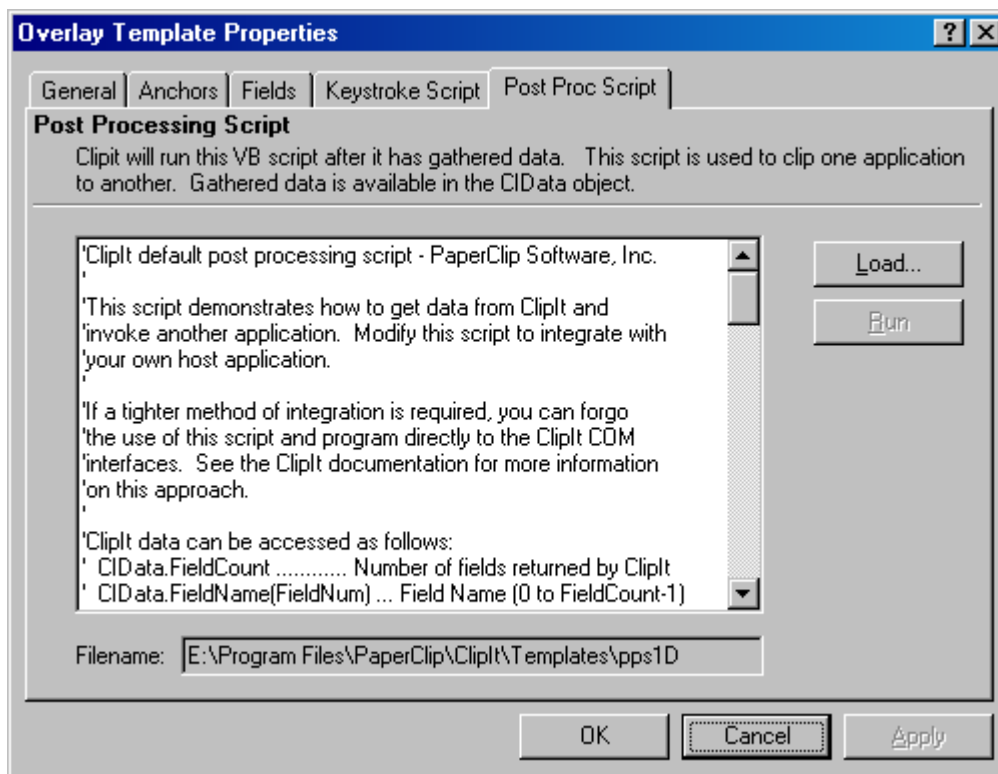
1. Select Overlays... from the Clipt menu. The Overlay Templates dialog box appears.
2. Select the desired overlay template and choose Remove. A prompt appears for you to confirm the deletion.
3. Click Yes to delete this template, or No to cancel.
4. Click Close to exit.



### *Re-creating an Existing Overlay Template*

If you need to modify any properties of an overlay template that pertain to the application screen such as the selected window, anchors, fields, field positions, etc. you must follow this procedure to re-create the overlay.

1. Select Overlays... from the Cliplt menu. The Overlay Templates dialog box appears.
2. Select the desired overlay template and click Properties. The Overlay Template Properties dialog box appears, then select the Post Proc Script tab.



3. Push the Save as... button and save the script file in a safe place. You may want to name it with the name of the overlay template so that you can remember what it is for.
4. Now that you have saved the post processing script, take special note of the field names and the order in which they were created in the original overlay template. You may need to reuse these names as defined previously so that they will work with the original post processing script or programmed interfaces.
5. Now you can delete the original overlay template.
6. Create a new overlay template for the target application screen.
7. Once you have tested the new overlay template with the default post processing script, you can replace the default script with the script file that you saved earlier. Use the Load... button on the Post Processing Script dialog to do this.

## Overlay Template Distribution

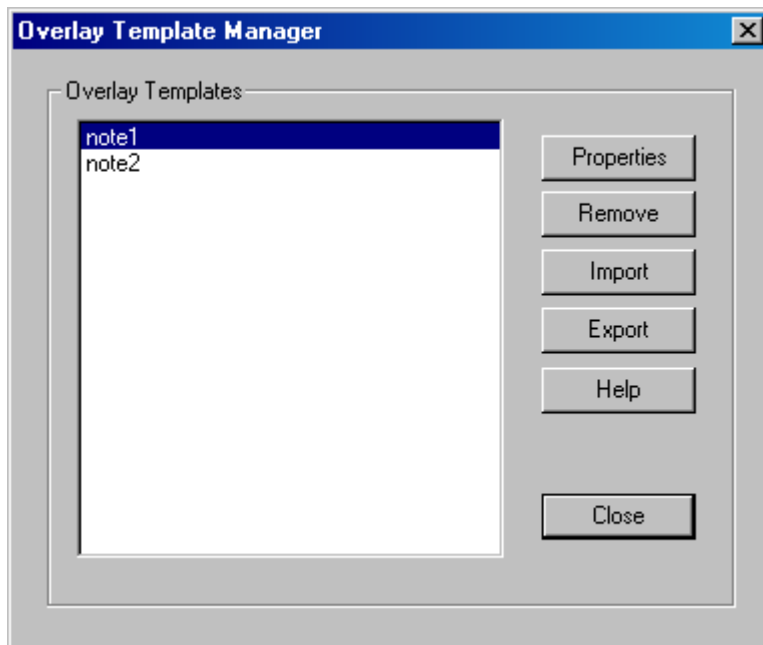
---

Clipt provides a method for system administrators to create overlay templates and distribute them to remote locations not connected to the local network. This is done by using the Clipt overlay template Export and Import functions.

### *Exporting an Overlay Template*

Once an overlay template is created and working, a system administrator can export that template including its scripts by performing one simple operation.

1. Select Overlays... from the Clipt menu. The Overlay Templates dialog box appears.
2. Select one or more overlay templates that you wish to export.

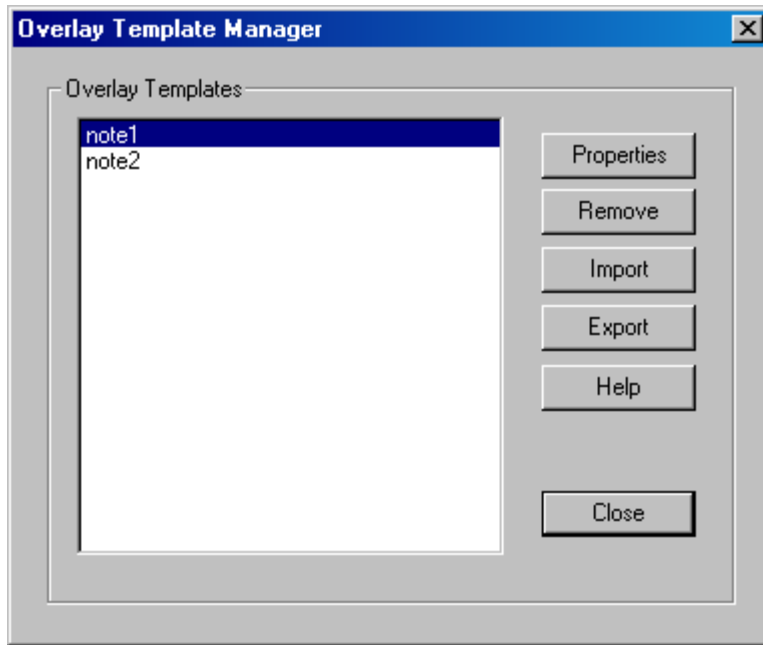


3. Push the Export... button, then name and save the template export file.

### *Importing an Overlay Template*

Overlay templates can be imported by following these simple steps.

1. Select Overlays... from the Clipt menu. The Overlay Templates dialog box appears.



2. Push the Import... button and browse to the template file for importing. If you are importing templates which names already exist in your system you will be prompted if you wish to overwrite them.

## Keystroke Scripts

---

When the Cliplt (VCP) Hot Key (Ctrl+F3) is invoked, Cliplt copies information from the selected application to the Windows clipboard, and then parses and makes the information available to your host application via the Post Processing Script or programmed interface (via COM object).

The Cliplt default Keystroke script assigns Ctrl+Ins to copy information to the clipboard. Most Windows applications conform to this standard sequence of keystrokes for copying data, but occasionally an application will require alternate keystrokes.

For terminal emulation and DOS box applications, the highlighting of data can be automated by changing the overlay script.

### **Select an Alternate Set of Keystrokes to Copy Data to the Clipboard**

If a Windows application does not copy data to the clipboard using the Ctrl/Ins key sequence, then the script for overlay templates associated with that application must be changed. Once the script for this application's first overlay has been changed, PaperClip32 will assign that script automatically to any subsequent overlays.

The Cliplt default keystroke script appears in the Keystroke Script dialog box as:

KEY C CONTROL

## Automating Data Highlighting prior to Copying It to the Clipboard

A script can be modified to automate the selection of data before it is copied to the clipboard.

Microsoft Access applications in form view are an example. To create or use an overlay, you must again choose Select Record from the Edit menu before you hot key.

You can automate this by adding the Select Record keystrokes (Alt+E+S) to the script before the Copy to Clipboard keystrokes (Ctrl+Ins).

KEY E ALT

KEY S

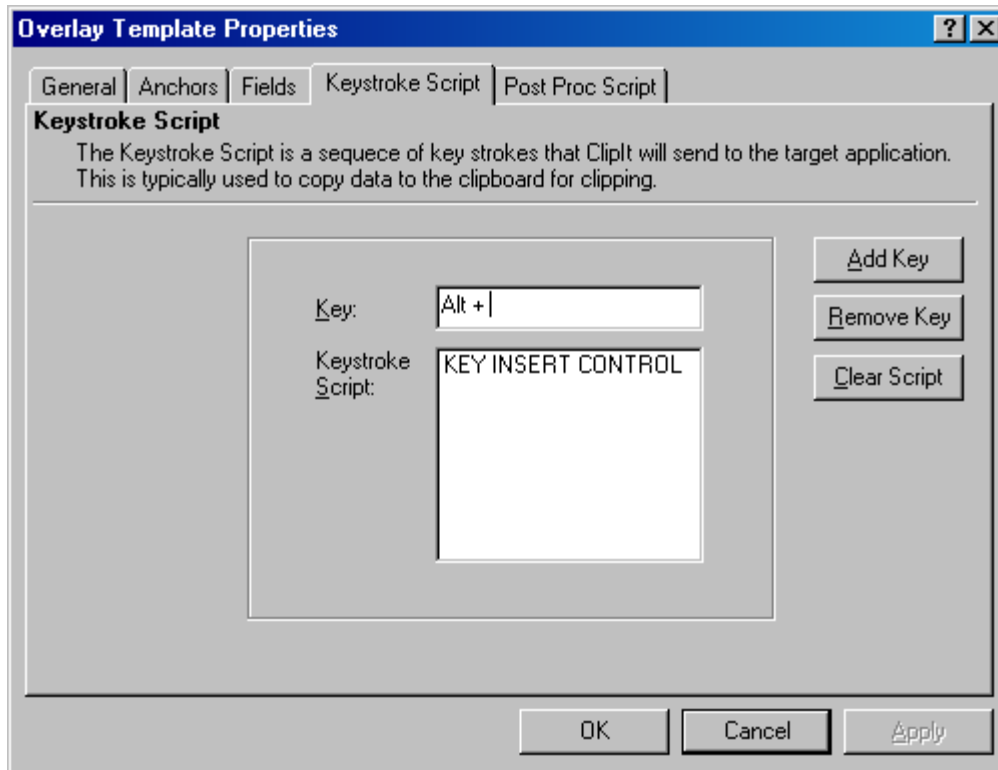
KEY INSERT CONTROL



### Modify a Keystroke Script

1. Select Overlays... from the Clipt menu. The Overlay Templates dialog box appears.
2. Select the overlay template to be edited, and click on the Properties button. The Overlay Template Properties dialog box appears with multiple tab controls.

3. Click on the Keystroke Script tab. The Script editing controls appear.



4. Enter script keys (by pressing the key or key combo) in the Key: edit box and click Add Key. The new key appears in the list box at below.
5. To select a set of keystrokes, click on the <Ctrl>, <Alt>, and/or <Shift> keys to assign the first key in the sequence and then click on the keyboard character key to be used with it.
6. To remove an existing line, select the line in the list box and then click Remove Key.
7. To replace an entire script, click Clear Script.
8. Repeat this process for each set of keystrokes to be added to the script.

## Troubleshooting Overlay Creation

---

Problems with overlay creation may occur if the selected application does not respond to the Cliplt default keystroke script which assumes that Ctrl+Insert will copy data to the clipboard. This problem does not occur in Dialog Box type Overlays. In most cases, by following the instructions below, the overlay can still be created. Once the overlay has been successfully created, the keystroke script can be modified so that data can be successfully copied to the clipboard.

### Database/Spreadsheet Overlays

---

- For records presented in tabular form, such as in a spreadsheet where each record is presented as a row, highlight the entire row before pressing Ctrl+F2. If Cliplt still cannot

see the data, try copying the record to the clipboard with an alternate set of keys (such as Ctrl+C) before pressing Ctrl+F2.

- For records presented in full-screen (form) view, ordinarily Ctrl+F2 selects the entire record and then copies it to the clipboard. If this does not occur try to select the record via the Edit menu (Select All or Select Record) before pressing Ctrl+F2. Consult the application's manual for instructions on copying data to the clipboard.

## **Dialog Box Overlays**

---

---

Make sure that the dialog box is the active window (click anywhere on it) before pressing Ctrl+F2. PaperClip32 does not use the clipboard for dialog-box type overlays, so no adjustments to the keystrokes for selecting data or copying it to the clipboard are necessary.

## **Dial-Up, Free-Form, or Terminal Emulation Overlays**

---

---

- Make sure all the appropriate text has been selected from the window (for terminal-emulation overlays, you should select an entire record) before pressing Ctrl+F2.
- If Cliplt still cannot see the data, try copying the record to the clipboard with an alternate set of keys (such as Ctrl+C) before pressing Ctrl+F2. Consult the application's manual for instructions on copying data to the clipboard.

## **DOS Boxes Running on Windows 2000/NT**

---

---

A special case exists for being able to clip to DOS boxes running on Windows 2000 or NT. In order for this to work you must invoke your DOS application from the PaperClip "shell" program (PCSHELL.EXE). Do this by creating a desktop shortcut with the following command line:

```
C:\pclip32\pcshell.exe yourapp.exe
```

Where: "yourapp.exe" is your application exe to run.

Make sure Cliplt is loaded and run the shortcut as defined above. Activate your DOS application window then hit the Ctrl+F2 Overlay Hotkey to create the overlay template. Define the template as you would for other character mode screens.

## Integrating Cliplt to Your Host Application

---

Minimal scripting is required for integrating with the host application. Once this integration is done, the host application is “Cliplt” enabled and ready to link to any target application without programming. The Cliplt toolkit provides two means to perform host integration. System integrators may want to use Cliplt’s scripting interface that requires minimal scripting. Application developers may want to use Cliplt’s API for tighter integration.

---

**Important Note:** When using the Cliplt API for programmed host application integration, the post processing scripts are not invoked. It is up to the host application to handle the data output from Cliplt directly. Only one host application can connect to Cliplt at a time.

---

## Scripting

---

---

Cliplt provides a default script file that allows you to test your created application screen overlay templates and give you a starting point on integrating with your host application. This script can be easily modified to suit your needs for your application. Each defined Cliplt screen overlay template can specify a unique script to run. This script is called the Post Processing Script and is written using VBScript. Cliplt only runs VBScript and is run via Microsoft’s ActiveX Scripting object that runs on all Windows platforms.

Cliplt’s post processing scripts are run after the data has been extracted from the target application screen.

### Programming Environment

---

---

Cliplt was developed to act as a scripting host application therefore, scripts than run from Cliplt will automatically have some Cliplt specific objects and methods available to the script writer.

### Language

All facilities as defined in the Microsoft VBScript language can be used. The documentation for VBScript can be accessed via the Internet by visiting the following site:

<http://www.msdn.microsoft.com/scripting>

### Cliplt Data Object (CIData)

The Cliplt default post processing script contains the best examples on how to use this object. The default post processing script is automatically loaded when you first define an overlay template.

The definition for the CIData object is as follows:

Field Name	Description
CIData.FieldCount	Number of fields returned by Cliplt
CIData.FieldName(FieldNum)	Field Name (0 to FieldCount-1)
CIData.Field(FieldNum)	Field Value (0 to FieldCount-1)
CIData.AppName	Application module name
CIData.TitleBarText	Application window title bar text
CIData.TemplateName	Template that has run this script
CIData.IsDos	Boolean (true/false) if DOS application

---

**Notes:**

1. The actual Overlay Template field name can also be used to access the field data  
eg; CIData.Field(ActualName)
  2. The field arrays are base 0. Therefore, the first element is indexed by a 0.
- 

***Cliplt Script Object (CIScript)***

The Cliplt Script Object provides a method for launching an application with a file.

**Method Name**

CIScript.ShellExecute

**Syntax**

CIScript.ShellExecute bsFile, bsParameters, bWaitForComplete, IShowCmd, bsDefaultDir, bsOpVerb

**Parameters**

Parameter	Values	Description
bsFile	Must specify a valid string.	A string that specifies the name of the file or program on which ShellExecute will perform the action specified by the bsOpVerb parameter. The system registry verbs that are supported by the ShellExecute function include "open" for executable files and document files and "print" for document files for which a print handler has been registered.  <b>Note:</b> If the path is not included with the name, the current directory is assumed.

bsParameters	(NULL)* or a valid string.	A string that contains the application parameters. The parameters must be separated by spaces.
bWaitForComplete	VbTrue or VbFalse *	Boolean (TRUE/FALSE) to tell ShellExecute to wait for the completion of the shelled program before returning to the script.
IShowCmd	ciSW_HIDE (0)	Hides the window and activates another window.
	ciSW_MAXIMIZE (3)	Maximizes the specified window.
	ciSW_SHOW (5) *	Activates the window and displays it in its current size and position.
	ciSW_MINIMIZE (6)	Minimizes the specified window and activates the next top-level window in the desktop window order.
	ciSW_RESTORE (9)	Activates and displays the window. If the window is minimized or maximized, Windows restores it to its original size and position.
bsDefaultDir	(NULL)* or a valid string.	A string that specifies the default directory for the program to run.
bsOpVerb	“open” * or any other valid application verb.	A string, referred to as a <i>verb</i> , that specifies the action to be performed. The set of available verbs depends on the particular file or folder. Generally, the actions available from an object's shortcut menu are available verbs. These may include open, print or edit.

\* Specifies the default value for this parameter.

Examples for using CScript.ShellExecute may be as follows:

1. CScript.ShellExecute “note.txt”,,ciSW\_MINIMIZE  
This will run notepad.exe with the file note.txt and will minimize the window.
2. CScript.ShellExecute “notepad.exe”,”note.txt”  
This will run notepad.exe with the file note.txt and show the window normally.

## **Programming Interface (API)**

---

The Cliplt VCP engine is developed as a set of COM objects that can be used in any of the visual development environments supporting the Microsoft COM standard including Visual BASIC and Visual C++. This level of integration would allow an application developer to seamlessly incorporate the Cliplt technology into their host application.

To use this interface, the application programmer must create an object that will start Cliplt. This object will fire an event when the user initiates clipping. The application programmer need only define a function to handle this event when the Cliplt VCP engine has data ready. The host application will then retrieve the data from Cliplt via its COM interface.

### **ClipltLIB Object**

This is the main Cliplt object that contains two useful object classes as described below.

<b>Class</b>	<b>Member/Property</b>	<b>Description</b>
ClipltTargetApp	AppName	Application module name
	Field(FieldNum)	Field Value (0 to FieldCount-1)
	FieldCount	Number of fields returned by Cliplt
	FieldName(FieldNum )	Field Name (0 to FieldCount-1)
	IsDos	Boolean (true/false) if DOS application
	TemplateName	Template that has run for this event
	TitleBarText	Application window title bar text
ClipltInterface	Clip	Event which fires when Cliplt has captured data

---

**Notes:**

3. The actual Overlay Template field name can also be used to access the field data  
eg; CLipltTargetApp.Field(ActualName)
  4. The field arrays are base 0. Therefore, the first element is indexed by a 0.
- 

An example using these object classes written in Visual Basic can be found on the product CD. The Visual Basic project file "ClipltInterfaceTester.vbp" can be found in the "API Example" subdirectory.

## Using Cliplt

---

Now that you have created an interface to your target application and have integrated Cliplt into host application you are ready to put Cliplt to use. Using Cliplt is extremely simple giving users a quick hot-key seamless interface into your application. To the user, it will seem as if the two applications are connected.

### Cliplt Hotkey (VCP Key)

---

Cliplt is invoked using the Cliplt Hotkey. The default key sequence for this key is Ctrl+F3. Another word for this key is the VCP (Visual Context Processor) key.

#### *Using the Cliplt Hotkey*

1. Make sure that Cliplt is loaded and running. See the Cliplt icon in the Windows task tray.
2. Launch your target application if not already done so.
3. In that application, go to the screen displaying the data record of interest.
4. Invoke the Cliplt Hotkey by pressing Ctrl+F3.
5. Using the data on the screen you will be automatically “linked” to another application in proper context.

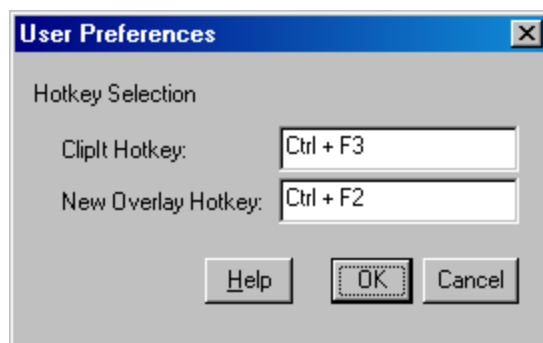
### User Preferences

---

You may need to change the Cliplt Hotkey if you run into a conflict with the same key being used by another application.

#### *Changing the Cliplt Hotkey*

1. Select Preferences... from the Cliplt menu. The User Preferences dialog box appears.



2. Select the Cliplt Hotkey and press the new key combination.

## Running Clipt Upon System Startup

---

Clipt can be configured to load upon system startup by simply adding a ShortCut to Clipt.exe to your Startup folder.



### *Adding Clipt to Startup*

Select Add ShortCut to Startup from the Clipt menu. Clipt will now load on startup.

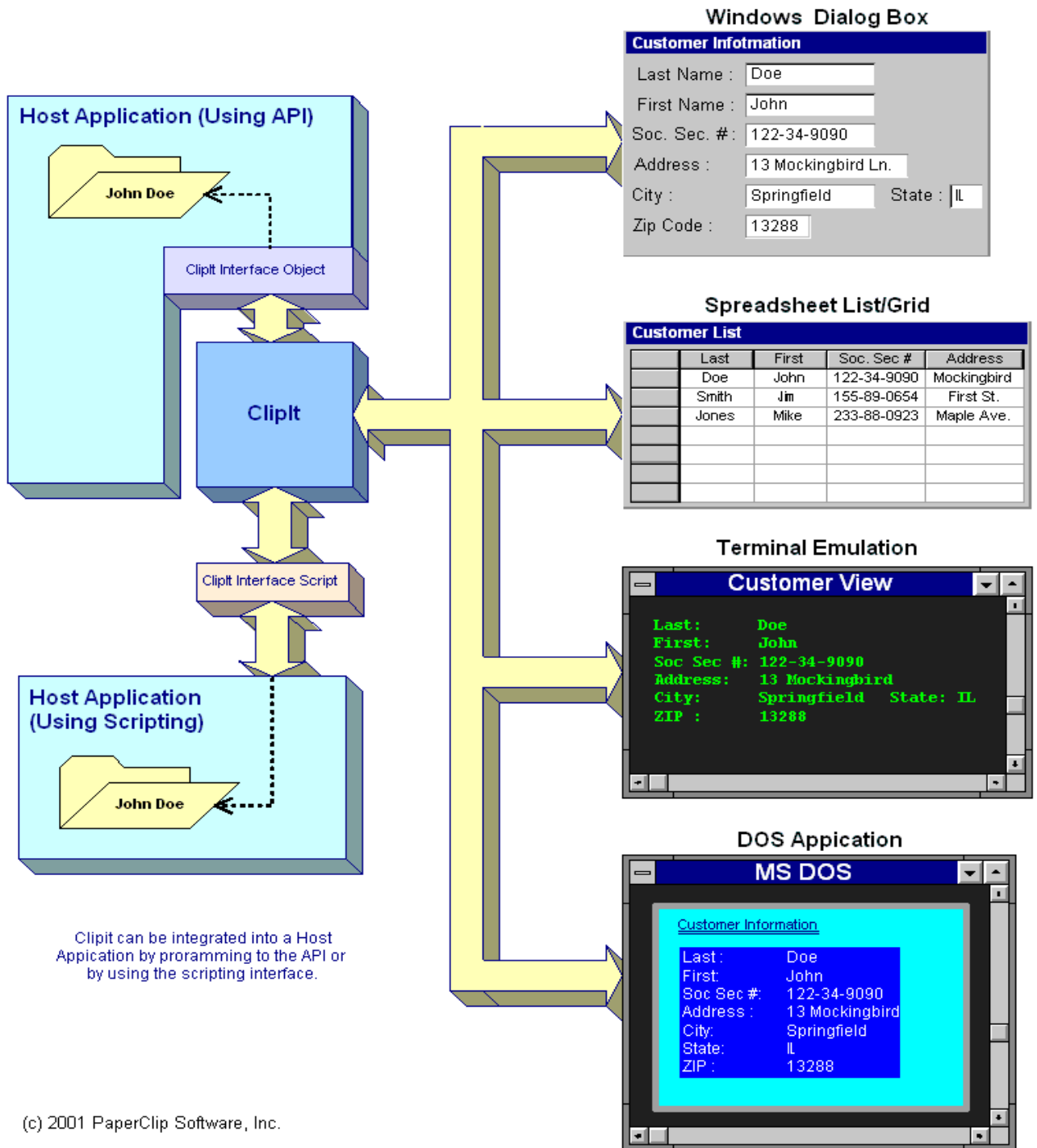


### *Removing Clipt from Startup*

Select Del ShortCut from Startup from the Clipt menu. Clipt will no longer load on startup. This does not remove Clipt from the system.

# Appendices

## System Diagram



## Example Post Processing Script

---

```
'This sample script writes the "clipped" data to a text file then
'invokes Notepad.exe by executing the created .TXT file via the
'ClipIt CIScript.ShellExecute method.

'Defines for CIScript.ShellExecute lShowCmd parameter
const    ciSW_HIDE = 0
const    ciSW_NORMAL = 1
const    ciSW_MAXIMIZE = 3
const    ciSW_SHOW = 5
const    ciSW_MINIMIZE = 6
const    ciSW_RESTORE = 9

Dim    fso, MyFile, TempFolder, FilePath
Const TemporaryFolder = 2

'Text file name to create for the output of this script
const textFile = "ClipIt.txt"

'Create a File System Object for performing file i/o
Set fso = CreateObject("Scripting.FileSystemObject")

'Create the output text file in the system temp folder
Set TempFolder = fso.GetSpecialFolder(TemporaryFolder)
Set MyFile = TempFolder.CreateTextFile(textFile, True)
FilePath = TempFolder & "\" & textFile

MyFile.WriteLine("<< ClipIt default post processing script output >>")
MyFile.WriteLine("")

'Write the basic target application information
MyFile.WriteLine("Target Application Name : " & CIData.AppName)
MyFile.WriteLine("Title Bar           : " & CIData.TitleBarText)
MyFile.WriteLine("Template Name        : " & CIData.TemplateName)
MyFile.WriteLine("DOS App              : " & CStr(CIData.IsDos))
MyFile.WriteLine("")

'Get the number of fields defined in the Overlay Template
fieldCount = CIData.FieldCount

MyFile.WriteLine("Clipped data fields:")

for fieldNum = 0 to fieldCount - 1
    'Write out each of the clipped data fields and its name
```

```

    MyFile.WriteLine(" Field " & CStr(fieldNum) & _
                    " : " & CIData.FieldName(fieldNum) & _
                    " = " & CIData.Field(fieldNum))
next

MyFile.Close

'Open the created output text file
CIScript.ShellExecute "Notepad.exe", FilePath,,ciSW_SHOW

'Debug message
'MsgBox FilePath & " Created"

```

## **Example Visual Basic Host Program**

---

```

' This Visual Basic example uses a form with a listbox to display
' the data from a ClipIt enabled target application.
' Code from this example would be integrated into your own application
' in order to get data from a target application via ClipIt .

' Declare a variable to handle events from ClipIt
Dim WithEvents EventHandler As ClipItInterface

' This subroutine is called when the test application's form
' is loaded.
'
Private Sub Form_Load()

' Create the object to handle the ClipIt events
Set EventHandler = New ClipItInterface

End Sub

' This subroutine is the event handler for the ClipIt "Clip" event
' When Clipit has data ready (ie; the hot-key is pressed and an overlay
' template is successfully processed, this routine is called.
'
' pTarget is defined as a pointer to the CLIPITLib.ClipItTargetApp class
' Properties of this class give access to the "clipped" data from the
' target application.
'
Private Sub EventHandler_Clip(ByVal pTarget As CLIPITLib.ClipItTargetApp)

' Use a list box object to display the results in the form

```

```

lbResults.Clear 'Clear the listbox

lbResults.AddItem ("Target Application Name: " & pTarget.AppName)
lbResults.AddItem ("Title Bar: " & pTarget.TitleBarText) 'Get the
application's title bar
lbResults.AddItem ("Template Name: " & pTarget.TemplateName) 'Get the
overlay template name
lbResults.AddItem ("DOS App: " & CStr(pTarget.IsDos)) 'Get "Is DOS" app
flag

lbResults.AddItem ("") 'Space for formating
lbResults.AddItem ("Clipped Data fields:")

FieldCount = pTarget.FieldCount 'Get the number of data fields
For fieldNum = 0 To FieldCount - 1 'Get the data for each field
    lbResults.AddItem (" Field" & CStr(fieldNum) & ": Name = " &
pTarget.FieldName(fieldNum) & ", Data = " & pTarget.Field(fieldNum))
Next

End Sub

```

## Command Line Switches

---

---

This section describes some Cliplt.exe command line switches that can be useful for system administrators.

Switch	Description
/st DirName	Set the template directory in the registry
/sa	Set Cliplt for administrator mode in the registry
/sw	Set Cliplt for workstation mode in the registry
/a	Force Cliplt to run in admin mode
/w	Force Cliplt to run in workstation mode