Dial-Up Scripting Command Language
For Dialer Scripting Support

Using This Document

This document is written primarily for the Internet service provider (ISP) to create scripts that end users can edit for their own connections.

How to Use Scripting in Internet Explorer

With IEscript.exe, an end user can use a script file (.scp) to make a connection. IESCRIPT adds settings to the connection (.con) file as shown below. Use IESCRIPT to edit the connection file—do not edit the connection file manually.

Section Name: **Script**

The following keywords are supported:

ScriptEnabled
Yes/No Specifies whether to activate the scripting utility.

ScriptFileName
A fully qualified string that specifies the location of the script file associated with this connection file.

ScriptTerminal
Yes/No Specifies whether the scripting terminal window should be loaded during script playback.

ScriptRecord
Yes/No Specifies whether the script executable should go into record mode on activation.

Sample Connection File Entries for Scripting
Example 1:

```
[Script]
ScriptEnabled=Yes
ScriptFileName=C:\Shiva\Myscript.Scp
ScriptTerminal=Yes
ScriptRecord=No
```

Example 2:

```
[Script]
ScriptEnabled=Yes
ScriptFileName=C:\Shiva\NewScrip.Scp
ScriptTerminal=Yes
```

ScriptRecord=Yes

Note

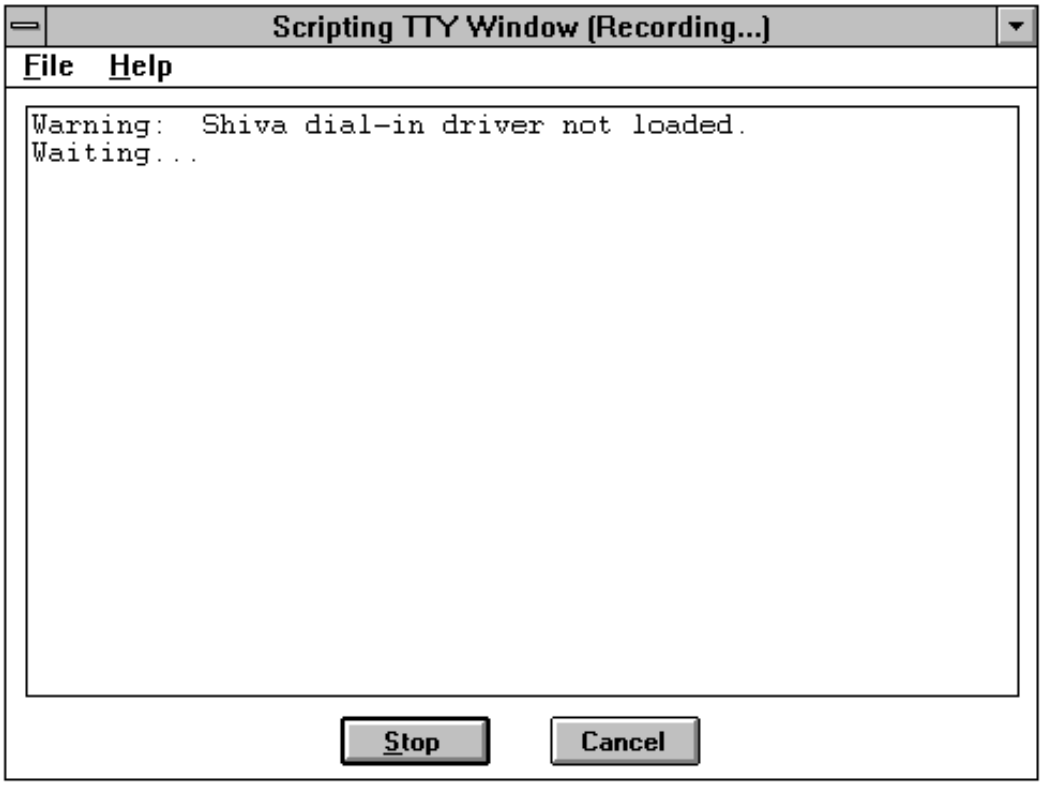
- Example 2 shows what appears while a script is recording.

Creating a Script in Internet Explorer

To create a script file for end users, you must first record a script file with Scripter.exe. After you've created this file, the end user can edit it and insert his or her own name and password.

Record and play back the script with the TTY window

1. Run Scripter.exe. It is usually located in the same directory as the Internet Explorer program.
2. On the File menu, click Record.
3. Locate a script file, and then select a script (.scp) file from the file list. Or type in a new file name. When you start recording, the following window appears.



4. Open your Internet Explorer program group, and then click New Connection. Follow the instructions on your screen to create a new connection. (Or choose a connection icon you have already made.)
5. Select the connection icon. From the File menu, click Properties. Make sure the check box labeled Bring Up Terminal Window After Dialing is selected.
6. Allow a connection to the internet server to be completed.
   The ShivaRemote Scripter window displays a "Started!!!" prompt.
7. Click in the record window (currently acting as the terminal window), and then enter the information needed to login to the ISP server.

8 After login is established, click End Record in the Scripter window.
9 In the ShivaPPP Scripter window, click OK.
10 To end communications, click Cancel in the connection dialog box .

Note
When using this script file, the user will want to make sure the check box labeled Bring Up Terminal Window After Dialing is not selected.

When you're finished entering the required information in the script, click Stop. The window closes, and the normal PPP negotiation begins.

The script created by the program is as follows:

```
; C:\SCRIPTS\IE\LOGIN.SCP
; Created:  7-1-1996 at 17:07:11
;
;
proc main
string szPassword
transmit "^M"
waitfor "Host Name:  ",matchcase until 15
transmit "spry01^M"
waitfor "UIC: ",matchcase until 6
transmit "spry053514^M"
waitfor "Password: ",matchcase until 3
if $PASSWORD then
 transmit $PASSWORD
 transmit "^M"
 goto doneTxPassword
endif
getinput "Password: " szPassword
transmit szPassword
transmit "^M"
doneTxPassword:
endproc
```

Edit the script file based on the recorded file, or create a script file from scratch

Scripter.exe records a script file that contains user-specific information, including the user name and password. If you are an ISP, you may want to provide a generic script file that can be edited and used by any end user.

Use the following example script as a guideline for creating a generic script that does not contain specific user information.

```
;-----------------------------------------------------
; File Name:    LOGIN01.SCP
; Date Created: 10-18-1996
; Time Created: 12:00:00
;-----------------------------------------------------

proc main
```

```
; Some systems need a keypress to initiate login procedure
;-------------------------------------------------------------
----

transmit "^M"


; Send host name, hard-coded by the service provider
;-----------------------------------------------

waitfor "Host Name:"
transmit "hostname"
transmit "^M"


; Send user name
; Taken from User Name box of associated connection file
;-------------------------------------------------------

waitfor "Username:"
transmit $USERID
transmit "^M"


; Send password
; Taken from Password box of associated connection file
;-------------------------------------------------------

waitfor "Password:"
transmit $PASSWORD
transmit "^M"

endproc
```

Advanced Scripting Information

1.0 Overview

Many Internet service providers and online services require the user to manually enter information, such as the user name and password, to establish a connection. With Dial-Up Scripting support, the user can write a script to automate this process.

A script is a text file that contains a series of commands, parameters, and expressions required by the Internet service provider or online service to establish the connection and use the service. You can use any text editor, such as Microsoft Notepad, to create a script file. After you've created your script file, you can then assign it to a specific Dial-Up Scripting connection by running the Dial-Up Scripting Tool.

2.0 Basic structure of a script

A command is the basic instruction that a script file contains. Some commands require parameters that further define what the command should do. An expression is a combination of

operators and arguments that create a result. Expressions can be used as values in any command. Examples of expressions include arithmetic, relational comparisons, and string concatenations.

The basic form of a dial-up script follows:

```
;
; A comment begins with a semicolon and extends to
; the end of the line.
;

proc main
       ;  A script can have any number of variables
       ;  and commands

       variable declarations

       command block

endproc
```

A script must have a main procedure, specified by the **proc** keyword, and a matching **endproc** keyword indicating the end of the procedure.

You must declare variables before adding commands. The first command in the main procedure is run and then any subsequent commands are run in the order they appear in the script. The script ends when the end of the main procedure is reached.

3.0 Variables

Scripts may contain variables. Variable names must begin with a letter or an underscore (_), and can contain any sequence of upper- or lowercase letters, numbers, and underscores. You cannot use a reserved word as a variable name. For more information, see the list of reserved words at the end of this document.

You must declare variables before you use them. When you declare a variable, you must also define its type. A variable of a certain type may only contain values of that same type. The following three types of variables are supported:

Type	Integer
Description	A negative or positive number, such as 7, -12, or 5698.

Type	String
Description	A series of characters enclosed in double quotation marks—for example, "Hello world!" or "Enter password:".

Type	Boolean
Description	A logical boolean value of TRUE or FALSE.

Variables are assigned values using the following assignment statement:

variable **=** *expression*

The variable gets the evaluated expression.

Examples:

```
integer count = 5
integer timeout = (4 * 3)
integer i

boolean bDone = FALSE

string szIP = (getip 2)

set ipaddr szIP
```

The above example uses script commands that are not supported by Shiva.

3.1 System variables

System variables are set by scripting commands or are determined by the information you enter when you set up a connection. System variables are read-only, which means they cannot be changed within the script. The system variables are:

Name	$USERID
Type	String
Description	The user identification for the current connection. This variable is the value of the user name specified in the Connection Connect To dialog box.

Name	$PASSWORD_
Type	String
Description	The password for the current connection. This variable is the value of the user name specified in the Connection program's Connect To dialog box.

Name	$SUCCESS
Type	Boolean
Description	This variable is set by certain commands to indicate whether or not the command succeeded. A script can make decisions based upon the value of this variable.

Name	$FAILURE
Type	Boolean
Description	This variable is set by certain commands to indicate whether or not the command failed. A script can make decisions based upon the value of this variable.

These variables may be used wherever an expression of a similar type is used. For example,

```
transmit $USERID
```

is a valid command because $USERID is a variable of type string.

4.0 String literals

Scripting for Dial-Up Scripting supports escape sequences and caret translations, as described below.

String literal	Description
String literal	**^***char*
Description	Caret translation
	If *char* is a value between '@' and '_', the character sequence is

translated into a single-byte value between 0 and 31. For example, ^M is converted to a carriage return.

If *char* is a value between '@' and '_', the character sequence is translated into a single-byte value between 0 and 31. For example, ^M is converted to a carriage return.

If *char* is a value between 'a' and 'z', the character sequence is translated into a single-byte value between 1 and 26.

If *char* is any other value, the character sequence is not specially treated.

String literal **<cr>**
Description Carriage return

String literal **<lf>**
Description Linefeed

String literal **\"**
Description Double quotation mark

String literal **\^**
Description Single caret

String literal **\<**
Description Single 'less-than' sign (<)

String literal ****
Description Backslash

Examples:

```
transmit "^M"
transmit "Joe^M"
transmit "<cr><lf>"
waitfor "<cr><lf>"
```

5.0 Expressions

An expression is a combination of operators and arguments that evaluates to a result. Expressions can be used as values in any command.

An expression can combine any variable, or integer, string, or boolean values with any of the unary and binary operators in the following tables. All unary operators take the highest precedence. The precedence of binary operators is indicated by their position in the table.

The unary operators are:

Operator -
Type of operation Unary minus

Operator !
Type of operation One's complement

The binary operators are listed in the following table in their order of precedence. Operators with higher precedence are listed first:

Operators	* /
Type of operation	Multiplicative
Type restrictions	Integers

Operators	+ -
Type of operation	Additiveintegers
Type restrictions	Strings (+ only)

Operators	< > <= >=
Type of operation	Relational
Type restrictions	Integers

Operators	== !=
Type of operation	Equality
Type restrictions	Integers, strings, booleans

Operators	and
Type of operation	Logical AND
Type restrictions	Booleans

Operators	or
Type of operation	Logical OR
Type restrictions	Booleans

Examples:

```
count = 3 + 5 * 40
transmit "Hello" + " there"
delay 24 / (7 - 1)
```

6.0 Comments

All text on a line following a semicolon is ignored.

Examples:

```
; this is a comment

transmit "hello"          ; transmit the string "hello"
```

7.0 Keywords

Keywords specify the structure of the script. Unlike commands, they do not perform an action. The keywords are listed below.

proc *name*

Indicates the beginning of a procedure. All scripts must have a main procedure (**proc** main). Script execution starts at the main procedure and stops at the end of the main procedure.

endproc

Indicates the end of a procedure. When the script is run to the **endproc** statement for the main procedure, Dial-Up Scripting will start PPP or SLIP.

integer *name* [**=** *value*]

Declares a variable of type integer. You can use any numerical expression or variable to initialize the variable.

string *name* [**=** *value*]

Declares a variable of type string. You can use any string literal or variable to initialize the variable.

boolean *name* [**=** *value*]

Declares a variable of type boolean. You can use any Boolean expression or variable to initialize the variable.

8.0 Commands

The initial implementation of the scripting language for Internet Explorer Dialer will be a subset of the Dial-Up Scripting Command Language as specified by Microsoft Windows(R) 95.

All commands are reserved words, which means you cannot declare variables that have the same names as the commands. The commands are as follows:

delay *nSeconds*

Pauses for the number of seconds specified by *nSeconds* before running the next command in the script.

Examples:

```
delay 2       ; pauses for 2 seconds
delay x * 3   ; pauses for x * 3 seconds
```

goto *label*

Jumps to the location in the script specified by *label* and continues running the commands following it.

Example:

```
 waitfor "Prompt>" until 10
 if !$SUCCESS then
       goto BailOut  ; jumps to BailOut and executes commands
                     ; following it
 endif

 transmit "bbs^M"
 goto End

BailOut:
 transmit "^M"
```

halt

Stops the script. This command does not remove the terminal dialog window. You must click Continue to establish the connection. You cannot restart the script.

if *condition* **then**
commands
endif

Executes the series of *commands* if *condition* is TRUE.

Example:

```
if $USERID == "John" then
       transmit "Johnny^M"
endif
```

label **:**

Specifies the place in the script to jump to. A label must be a unique name and follow the naming conventions of variables.

transmit *string* [**, raw**]

Sends the characters specified by *string* to the remote computer.

The remote computer will recognize escape sequences and caret translations unless you include the **raw** parameter with the command. The **raw** parameter is useful when transmitting $USERID and $PASSWORD system variables when the user name or password contains character sequences that, without the **raw** parameter, would be interpreted as caret or escape sequences.

Examples:

```
transmit "slip" + "^M"
transmit $USERID, raw
```

waitfor *string* [**, matchcase**] [**then** *label*
{ **,** *string* [**, matchcase**] **then** *label* }]
[**until** *time*]

Waits until your computer receives one or more of the specified strings from the remote computer. The *string* parameter is not case-sensitive unless you include the **matchcase** parameter.

If a matching string is received and the **then** *label* parameter is used, this command will jump to the place in the script file designated by *label*.

The optional **until** *time* parameter defines the maximum number of seconds that your computer will wait to receive the string before it runs the next command. Without this parameter, your computer will wait forever.

If your computer receives one of the specified strings, the system variable $SUCCESS is set to TRUE. Otherwise, it is set to FALSE if the number of seconds specified by *time* elapses before the string is received.

Examples:

```
waitfor "Login:"
```

```
waitfor "Password?", matchcase

waitfor "prompt>" until 10

waitfor
       "Login:"     then DoLogin,
       "Password:"  then DoPassword,
       "BBS:"       then DoBBS,
       "Other:"     then DoOther
 until 10
```

The following command is a Shiva extension to the Microsoft-designed scripting command set.

getinput "*Display String* " sz*ToReceiveInput*

Prompts the user to enter information during script execution.

Example:

```
Getinput "System Password" szSystemPassword
```

9.0 Reserved Words

The following words are reserved and may not be used as variable names:

and	boolean	databits	delay	
do	endif	endproc		endwhile
even	FALSE	getip	goto	
halt	if	integer	ipaddr	
keyboard	mark	matchcase	none	
odd	off	on	or	
parity	port	proc	raw	
screen	set	space	stopbits	
string	then	transmit	TRUE	
until	waitfor	while		

10.0 Shiva command words not supported by Scripter

getip
port databits
port parity
port stopbits
set screen keyboard
ipaddr
while/endwhile