QPlexView T27 Overview

QPlexView T27 Overview

QPlexView T27 is a companion product of QPlex Client T27, the Web-based connectivity software from KMSYS Worldwide, Inc. QPlexView T27 is an ActiveX component that provides a mechanism to Web-enable legacy applications running on Unisys mainframes.

QPlexView is designed for use in Microsoft's Active Server Page (ASP) environment. ASP is a server-side scripting environment for developing dynamic, interactive Web applications. ASP Scripts are written using a combination of HTML, scripting languages and COM components. ASP can use several scripting languages, but Microsoft Visual Basic Scripting Edition (VBScript) and Microsoft JScript are both included with ASP.

For more information on ASP see "Introducing Active Server Pages" from Microsoft. On the Web go to msdn.microsoft.com and use the MSDN search engine to look for "Introducing Active Server Pages".

QPlexView ActiveX components are created within an ASP using VBScript or JScript whenever access to a legacy mainframe application is needed.

QPlexView provides a terminal-level interface to the legacy application's normal screens, thus no change to the legacy application is required.

Unlike QPlex Client, which provides terminal emulation via a Web page, QPlexView runs on the Web Server. Only the HTML generated by the ASP is sent to the client PC (Web Browser).

QPlexView can also be run on the client side of an application. It is kind of like QPlex Client with a read-only screen. All terminal interaction has to be done programmatically in whatever language is hosting the QPlexView OCX.

In addition to running in a client side application, QPlexView also supports all the printing capabilities of QPlex Client. The QPlex Client configuration program can be

used to generate printer settings for use in QPlexView. See the ClientMode and Printer properties below. Also, see the AddPCXlate, RefreshScreen and Send methods below.

How QPlexView Works

The best way to explain how QPlexView works is by example. The example shown here is an ASP where a legacy, host application screen is accessed to retrieve a customer's name when an account number is entered by the end user.

The end user only sees the generated Web page in his browser. All the underlying ASP script and host access is invisible because it runs completely on the Web sever — not the client machine.

The initial Web page seen by the end user appears as follows. Notice that this browser page is the only thing seen by and accessible to the end user.

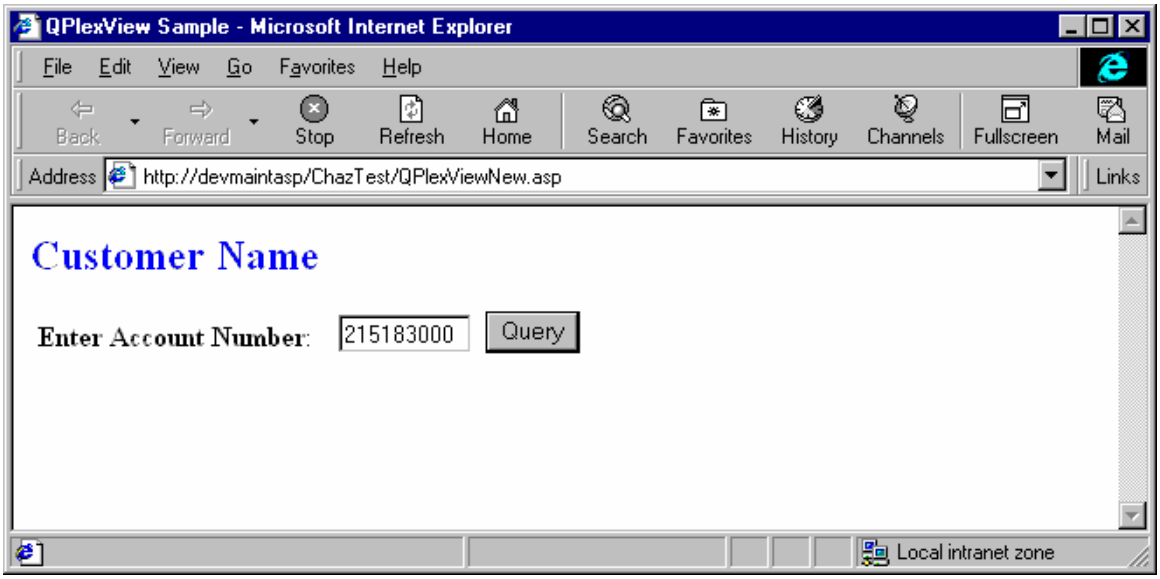

Figure 1: Initial (Blank) Page with User-Entered Account Number

Upon successful completion of a Query, the Web page may appear as follows:

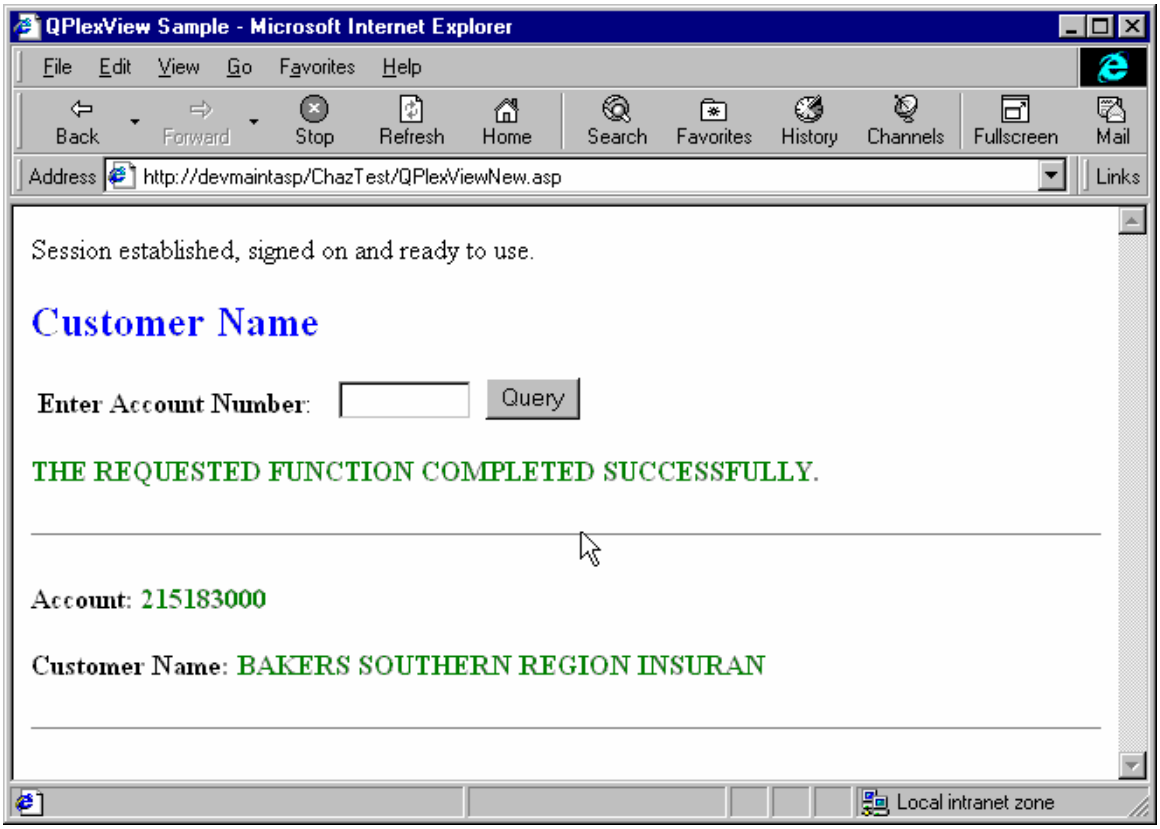

Figure 2: Web Page after Successful Query

The QPlexView ActiveX component actually sees the legacy application as if the user was running it from a dumb terminal. Properties and methods of the QPlexView component provide the mechanism that allows the ASP to interact with the legacy application screen.

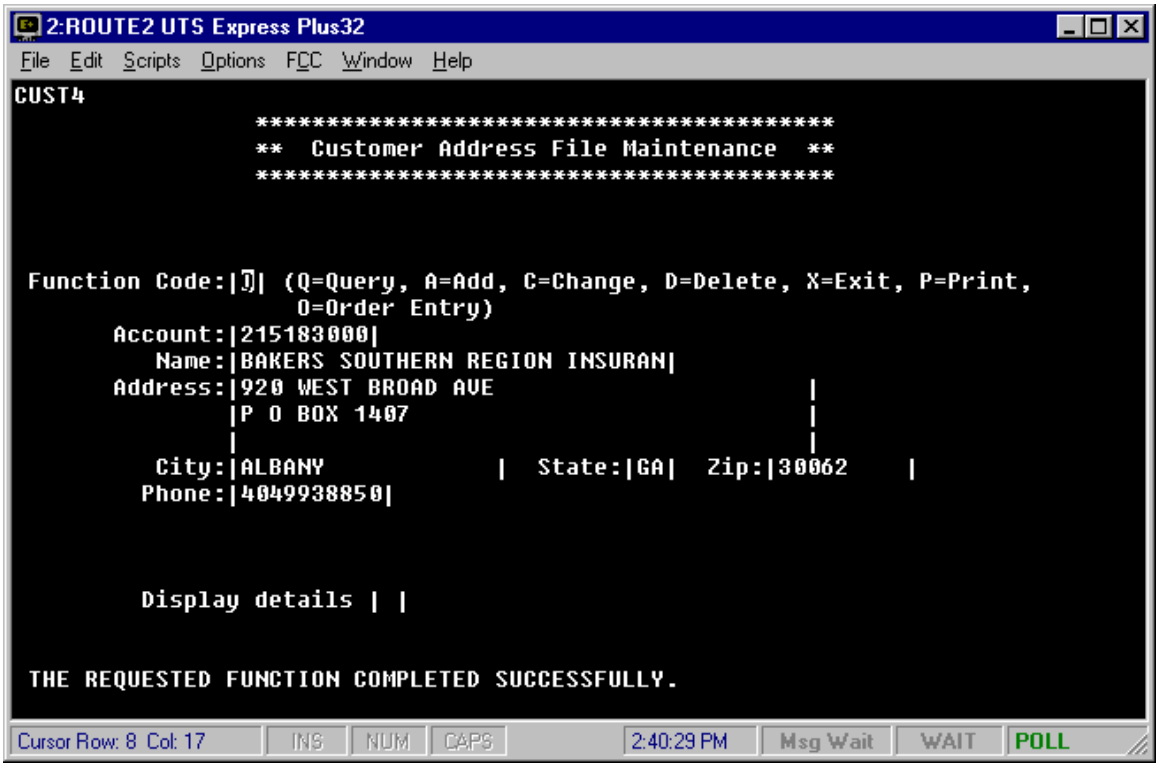

Figure 3: Legacy Transaction Screen

In this particular screen, there is much more information available, but only the customer name is being retrieved to keep the example as simple as possible.

The ASP Script

As mentioned earlier, the ASP script combines both HTML and VBScript or JScript. In this case, VBScript was used. All VBScript portions of the ASP are enclosed between the <% and %> HTML sequences. The amount of VBScript enclosed can be anything from a simple VB variable name to several lines of VBScript code. The following ASP was developed using Microsoft Front Page:

```
<html>

<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>QPlexView Sample</title>
</head>
<%
'
Response.Buffer = True

if Request("Btn_Query") = "Query" then
'
' Look for and or construct QPlexView Object and establish session
'
'
If IsObject(Session.Contents("SQPlexView")) then
'     The line above determines if a QPlexView control has been created.
'     Session.Contents is a container of all variables and objects stored at the
'     session level. IsObject is a method that determines if its argument is an
'     object(instantiated control) or not. Since session variables are local to a
'     single session, if SQPlexView is an object we should re-use it, because it is
'     a copy of the QPlexView control that we have already instantiated..
' Response.Write "Existing QPlexView object found<BR>"
'     The line above, if not commented out, would display a line in the resulting
'     output page, that would indicate we found an existing control. This is the
'     equivalent to using a COBOL DISPLAY statement for debugging.
 If Session("SQPlexView").SessionOpen = 1 Then
'     Since we are re-using a control, we should see if we already have a session
'     open to the MCP. (In actual usage, we would most likely have the session open,
'     but this is just more code to make the application more reliable.
'  Response.Write "Existing session open<BR>"
'     Once again a debug display.
  OpenStatus = 1
'     Indicate that we have a valid session to the MCP.
 Else
  OpenStatus = Session("SQPlexView").OpenSession("KMSMCP", Request.ServerVariables("SERVER_NAME"), 23)
'     We did not have a session so the line above is used to open a session. The
'     code 'Request.ServerVariables("SERVER_NAME")' is used to retrieve the name of
'     the web server that is running the script on this page. This makes the ASP
'     code portable, if the Host Gateway Server that we are connecting through is on the
'     same machine.
  Session("SignedOn") = "N"
 End If
Else
' Response.Write "Creating QPlexView object<BR>"
'     Once again a debug display.
 Set Session("SQPlexView") = Server.CreateObject("QPlexViewT27x.QPlexViewUTS")
 OpenStatus = Session("SQPlexView").OpenSession("KMSMCP", Request.ServerVariables("SERVER_NAME"), 23)
'     The line above is used to open a session to the MCP. See earlier occurrence
'     for more information.
 Session("SignedOn") = "N"
End if
if OpenStatus = 1 then
'     The line above is used to check if we have a session to the MCP. If not
'     then we know that the OpenSession call above failed and we should display
'     an error message. The OpenStatus is not a session variable because we can
'     determine the session status from the control by checking the SessionOpen
'     property.
```

```
' Response.Write "Session opened<BR>"
'     Once again a debug display.
   If Session("SignedOn") <> "Y" Then
  Rslt = Session("SQPlexView").WaitForSpecificString(11,1," LOGON - Menu-Assisted Resource Control ",5000)
'     The line above will wait for the logon prompt.
  If Rslt = 0 Then
   Msg = "Didn't get ' LOGON - Menu-Assisted Resource Control ' string<BR>"
  Else
   Rslt = Session("SQPlexView").WaitForString(44,10,"                    ",5000)
   Session("SQPlexView").SetScreenText 44,10,"QPlexView"
   Session("SQPlexView").SetScreenText 42,12, "Demo"
   Session("SQPlexView").DoTerminalKey 172 ' TRANSMIT
   Rslt = Session("SQPlexView").WaitForString(2,2,"Action:",5000)
   If Rslt = 0 Then
    Msg = "Didn't get signed on<BR>"
   Else
    Session("SignedOn") = "Y"
'     We have a session open to the MCP and, if requried,
'     signed on. Now we can run our first transaction. The
'     code here should be used to create input to run
'     the transaction, look at the output and generate
'     the HTML for information display.
   End If
  End If
 Else
  Msg = "Previous session found to be signed on.<BR>"
 Response.Write Msg
'     The line above will display the contents of the message variable. This is used
'     as the primary means of indicating to the user the state of connection and
'     data transfer.
 End If
 If Session("SignedOn") = "Y" Then
  Response.Write "Session established, signed on and ready to use.<BR>"
%>
<!--Comment.  Put HTML code here and more script code here to get data
 from the control-->
<%
  ' Run initial transaction
  Session("SQPlexView").DoTerminalKey 128 ' CLEAR HOME
        .
        .
        .
  Session("SQPlexView").DoTerminalKey 172 ' TRANSMIT
  Rslt = Session("SQPlexView").WaitForString(1, 1, "????? ", 5000)
  If Rslt = 0 Then
   Msg = "Didn't get expected screen response"
  Else
   ' Enter the necessary info for next screen
       .
       .
       .
   Session("SQPlexView").DoTerminalKey 172 ' TRANSMIT
   rslt = Session("SQPlexView").WaitForStringNot(1, 1, "????? ", 5000)
   If Rslt = 0 Then
    Msg = "No response to query"
   Else
    ' Get response from the screen.
    Msg = Session("SQPlexView").GetScreenText(???Col???, ???Row???, ???Len???)
    QueryDone = True
   End If
  End If
 End If


 Else
  Response.Write "Could not establish a session with the MCP host.<BR>"
 End If
```

```
'     At this point the script has completed. This page can be accessed again with
'     different input values to force it to do a query on specfic data etc. It will
'     reuse the same control and open session to the MCP for each use. You must
'     have a method for terminating the session to the MCP and destroying the
'     QPlexView contril within the ASP script. This action could be acomplished by
'     quering the value of a button that is used to activate the page and script.
'     The action would be acomplished by the following code.
' Session("SQPlexView").CloseSession()
'     There is not way for the browser to indicate to ASP that it's session is to be
'     terminated.  Sessions will terminate when their idle time expires. (This value
'     is set in the Application configuration in IIS.) When the session terminates,
'     code in Global.asa will be called.
End If
BtnVal = "Query"
%>

<body>

%>
<!--Comment.  Your HTML code continues here.-->
<%

</body>
</html>
```

Note the <% sequence that starts several lines of VBScript code.

The first "if" statement (*if Request("Btn_Query") = "Query" then*) acts as a first time switch, if the caption of Btn_Query is equal to "Query", the rest of the script will be executed. On first entry, the caption of Btn_Query will be blank so the rest of the VBScript will not be executed and only the initial HTML will display a blank input form to the end user (Figure 1).

The rest of the VBScript at the beginning of the ASP contains all the logic required to complete the transaction. In this case, a session is opened to a Unisys MCP host and a usercode and password is entered to establish a session.

The remaining query code is not shown and is dependent upon the host program executed to continue the dialog.

This is a very simple example, but QPlexView can be used to perform very complex transaction sequences involving the navigation of many screens and even performing updates.

Remember that the host is only accessed by the QPlexView ActiveX component running on the Web server. The end-user never has any knowledge of how the information was obtained. QPlexView and ASP provide a very secure method of providing legacy application access to users across the World Wide Web.

Global.asa

Code in Global.asa gets control 4 different times.
1) – When an Application is initialized.
2) – When an Application is terminated.
3) – When a Session is initialized.
4) – When a Session is terminated.

```
<script LANGUAGE="VBScript" RUNAT="Server">
SUB Application_OnStart
  Application("ApplicationName") = "QPlexView"
' Line above sets an Application level variable (similar to a Session level
' variable) so that the name of the Application can be determined. This is
' mostly for debug purposes.
END SUB
</script>

<script LANGUAGE="VBScript"
RUNAT="Server">
SUB Application_OnEnd
END SUB
</script>

<script LANGUAGE="VBScript" RUNAT="Server">
SUB Session_OnStart
' This routine is called when a new session is started. It is a good place to initialize
' variables that all users of an application will need.
END SUB
</script>

<script
LANGUAGE="VBScript" RUNAT="Server">
SUB Session_OnEnd
' This routine is called when a new session is terminated. Sessions can be terminated under
' two conditions.
' 1) - The session idle time is exceeded.
' 2) - The ASP script requests that the session be aborted.
' We have clean-up code here, because there is no way to
' know the state of the QPlexView control at time of termination.
  if IsObject(Session.Contents("SQPlexView")) then
' Determine if we have a control.
    if Session("SQPlexView").SessionOpen = 1 then
' Determine if we have a session to the 2200. (We don't care if we are
' signed-on to the 2200 because the session close will terminate the
' sign-on.
    Session("SQPlexView").CloseSession()
'   Close the session.
    end if
    Set Session("SQPlexView") = Nothing
' This line is VERY important. This is the only way to make the copy of the object that we
' just finished actually go away. If this is not done, memory will be orphaned until such
' time as IIS is stopped and re-started. Do not put this line of code into another web page.
' Doing so will cause numerous problems.
  end if
END SUB
</script>
```

QPlexView Host Connection

QPlexView connects to the host through the Gost Gateway Server (HGS) in the same manner as QPlex Client.

The QPlexView ActiveX Component Reference

This section defines the properties and methods provided by the QPlexView ActiveX components.

Properties

Property	Type	Description
AllowUserFKey	Integer Read/Write	Default is False (0). Controls the end users ability to do function keys (a form of transmit). If set to 0 (False), the script must perform this function. If set to 1 (True), the user can initiate this function; however, there is no way for the script to detect when or if the function has occurred. This property may be set when running in ClientMode and should be set before the OpenSession method is called.
AllowUserTransmit	Integer Read/Write	Default is False (0). Controls the end users ability to do a transmit. If set to 0 (False), the script must perform this function. If set to 1 (True), the user can initiate this function; however, there is no way for the script to detect when or if the function has occurred. This property may be set when running in ClientMode and should be set before the OpenSession method is called.
AutoScaleMode	Integer Read/Write	0 = No autoscale, 1 = Size screen to font size, 2 = Size font to fill screen (default). This property may be set when running in ClientMode and should be set before the OpenSession method is called.
BasicLogging	Integer Write Only	Default is 0. Set this property to 1 to get basic Windows/NT logging of actions performed by the QPlexView component.
ClientMode	Integer Read/Write	A value of 1 indicates that QPlexView is running on the client side and is allowed to print. Defaults to 0 indicating server mode operation.
ColorSettings	String Read/Write	A string that specifies the configured color settings. This property may be set when running in ClientMode and should be set before the OpenSession method is called.

Property	Type	Description
CursorColumn	Integer Read Only	The current cursor column. If a session is not open, a 0 is returned.
CursorRow	Integer Read Only	The current cursor row. If a session is not open, a 0 is returned.
DatacomOptions	String Read/Write	This property contains a string of settings that are generated in the QPlex Client T27 User Configuration Manager (QPlexCfgT27.exe) and govern communications between the host and PC. The WEB developer can generate the HTML for the settings then copy/paste them into the ASP script.
DetailLogging	Integer Write Only	Default is 0. Set this property to 1 to get detailed Windows/NT logging of actions performed by the QPlexView component.
HGSSecurityFlags	Integer Read/Write	0 = None 1 = Authenticate 3 = Authenticate and encrypted
HostGatewayType	Integer Read/Write	Set to "1" for HGS, and "0" (default) for QPlex. See the QuickStartHGS.asp for an example.
KeyboardOptions	String Read/Write	This property contains a string of settings that are generated in the QPlex Client T27 User Configuration Manager (QPlexCfgT27.exe) and govern keyboard actions. The WEB developer can generate the HTML for the settings then copy/paste them into the ASP script.
LastErrorCode	Integer Read Only	This property can be displayed in the HTML after an error is encountered. Do not use this property to determine whether an error has occurred, as the property is not cleared after the error.
LastErrorMessage	String Read Only	This property can be displayed in the HTML after an error is encountered. Do not use this property to determine whether an error has occurred, as the property is not cleared after the error.
Pages	Integer Read/Write	This property may be set to enable the paging feature of QPlexView. You may specify from one (default) to nine screen pages for each UTS screen. In conjunction with the paging feature, the PAGE DOWN and PAGE UP UTS keys are mapped to the Page Down and Page Up keyboard keys, respectively. Note: You must first click somewhere on the screen page before utilizing the Page Up/Page Down keys. Paging is most useful where only one screen is available. Paging provides the means to

Property	Type	Description
		maintain multiple output screen pages for reference while running additional transactions. WARNING: The host is completely unaware of the paging feature. Applications, especially DPS, often check the screen input for specific context, meaning the application thinks a certain screen was displayed and expects data to come from that screen. Use of the Page Down and Page Up keys with this feature can leave the screen in a state not expected by the application.
PrintTimeOut	Integer Read/Write	-1 (or any negative value) = No timeout, 0 = Timeout immediately after printing, *n* (any positive number) = Timeout *n* seconds after printing. The default is 15 seconds.
ReadOnly	Integer Read/Write	Default is 1 (True). ReadOnly controls whether or not the end-used is allowed to type directly into the screen. If false the only way to enter anything in the screen is programmatically in the script. This property may be set when running in ClientMode and should be set before the OpenSession method is called.
ScreenFont	String Read/Write	A string that specifies the configured screen font, style and size. This property may be set when running in ClientMode and should be set before the OpenSession method is called.
SessionOpen	Integer Read Only	Indicates whether a host session is currently open. 1 = Open, 0 = Not Open.
ShowStatusBar	Integer Write only	Use to show a status bar at the bottom of the screen. 1 = show, 0 = do not show.
TraceOption	Integer Write only	Default is 0 (none). Set this property to 2 to a trace to file. Please see "Where to Look for the Trace Files" below.
TransmitType	Integer Write Only	Set this property to the transmit type expected by the host software. Allowed values are VAR, CHAN, ALL. Most host applications expect VAR.
TransportTrace	Integer Write Only	Default is 0 (none). To start a Transport trace the new property TransportTrace is set to 1. This property can be set at any point,

Property	Type	Description
		but it is recommended that it be set before the OpenSession. Please see "Where to Look for the Trace Files" below.
Version	String Read Only	Use to retrieve the current version of the QPlexView ActiveX component.
VideoOptions	String Read/Write	This property contains a string of settings that are generated in the QPlex Client T27 User Configuration Manager (QPlexCfgT27.exe) and govern the state of the environment display. The WEB developer can generate the HTML for the settings then copy/paste them into the ASP script.

Where to Look for the Trace Files:

The software uses a Windows registry key for a default eXpress trace directory. The key is "Software\KMSystems\eXpressTrace" under HKEY_LOCAL_MACHINE. The value name is "DefaultTraceFilePath". The value data is initially null and must be MANUALLY entered as a stardard drive and path specification (e.g., "C:\trace\").

Note: The account that the web service is running under must have read access to the registry key and modify access to the directory specified in the registry key.

If no default directory is specified and the application is not running as a service (QPlexView runs as a service) then a standard File Save dialog will be displayed.

If the user cancels the File Save dialog or the trace file cannot be created in the specified directory, the trace will not be performed. A message will be displayed if not running as a service.

In the case of a server application (QPlexView) where no dialogs or messages can be used, the default directory must be specified in the registry and default file names will be used. The default file names are: "KMSTransEx*nnnnnnnn*.trc" for a transport trace, "eXpressTrace*nnnnnnnn*.txt" for an emulator trace and "ETEMTrace*nnnnnnnn*.txt" for a T27 ETEM trace, where *nnnnnnnn* is represents a unique date/time stamp.

Methods

AddKeyDef

Procedure **AddKeyDef** (*KeyDefinitionString*)

Description:

This procedure allows the keyboard to be customized from the Web page. The *KeyDefinitionString* is generated by the QPlex Client Configuration Manager and pasted into the script (see the QPlex Client Configuration Manager help in the QPlex Client).

CloseSession

Function **CloseSession** () as Integer

Description:

Close the current host session. If successful, this method returns a 1; else, it returns a 0.

DoDataKey

Sub **DoDataKey** (*KeyCode* as Integer)

Description:

Enter a data key into the internal terminal emulator as if it were typed from the keyboard by an end user. *KeyCode* is the ASCII character value (ex A = 65).

DoTerminalKey

Sub **DoTerminalKey** (*KeyCode* as integer)

Description:

Cause the specified Terminal Key sequence to be executed by the internal terminal emulator. All terminal key functions are available (see Key Code Values).

Note: Use the DoTerminalKey method to do a transmit, send a function key, erase display, etc.

GetScreenAttribute

Function GetScreenAttributes(Column as Integer, Row as Integer) as Integer

Description:

Retrieve certain screen attributes at the specified screen row and column coordinates. The following attribute values may be returned:

Constant	Value	Description
SATTR_NORMAL	0	Normal field
SATTR_FIELD	1	Start of field (set on 1st position of field)
SATTR_TAB	2	Tab stop (at start of field only)
SATTR_PROTECTED	8	Protected-Output only
SATTR_VIDEO_OFF	16	Video off (Data is present in

Constant	Value	Description
		screen buffer
SATTR_BLINK	128	Blinking field
SATTR_RIGHT	256	Right justified data
SATTR_REV	1024	Reverse video
Unique T27 Attributes:		
SATTR_BRIGHT	512	T27 Bright
SATTR_ULINE	4	T27 Underline
Unique UTS Attributes:		
SATTR_NUMERIC	32	Numeric only input
SATTR_ALPHA	64	Alpha only input
SATTR_LOWINT	512	UTS Low intensitity
SATTR_CHANGED	4	Data field changed flag

Returns:

The function returns the attribute bits. If an error is encountered, a 0 is returned.

GetScreenColor

Function GetScreenColor(Column as Integer, Row as Integer) as Integer

Description:

Retrieve the color attributes of the screen at the specified screen row and column. The following color attribute codes are used:

Returns:

This function returns the color attribute code. If an error was encountered, a 0 is returned.

GetScreenLine

Function **GetScreenLine** (*Row* as Integer) as String

Description:

Retrieve an entire line of text, including leading and trailing spaces, from the internal screen at the specified *row*. If successful, the function returns the line as a string; otherwise, it returns an empty string.

GetScreenText

Function **GetScreenText** (*Column* as Integer, *Row* as Integer, *Length* as Integer) as String

Description:

Retrieve a string of text, including leading and trailing spaces, from the internal screen at the specified *row* and *column* and for the specified *length*. If successful, the function returns the text as a string; otherwise, it returns an empty string.

HoldMessages

Sub HoldMessages

Description:

Force messages from the host to be held until a Receive, UnholdMessages is issued. Note: This subroutine only applies to UTS emulation.

OpenSession

Function **OpenSession** (*OpenId* as String, *IPAddress* as String, *IPPort* as Integer) as Integer

Description:

Open a host session using the specified *OpenId*, Host Gateway Server *IPAddress* and Host Gateway Server *IPPort*. This method returns a 1 if successful; else, it returns a 0.

OpenSessionStation

Function **OpenSessionStation** (*OpenId* as String *IPAddress* as String, *IPPort* as Integer, *StationName* as String) as Integer

Description:

This method is used in lieu of the OpenStation method when you want to specify the specific station name to use for the open.
Open a host session using the specified *OpenId*, Host Gateway Server *IPAddress*, Host Gateway Server *IPPort* and *StationName*. This method returns a 1 if successful; else, it returns a 0.

Receive

Function **Receive**(*TimeOutValue* as Integer) as Integer

Description:

The Receive method informs the user that a message, containing text and/or control sequences, has been received from the transport and mapped to the internal screen. The Receive method does **NOT** cause anything to actually happen, because the UTS terminal (which is being emulated) receives data without any action taken by the user. To retrieve the actual text received, use the GetScreenText or GetScreenLine method.

When the Receive function is called and used in conjunction with the HoldMessages method, one message will be accepted and processed (moved to the screen, etc.).

Used with the HoldMessages method, this function is useful when a stream of messages are expected from the host and the script needs to process each one separately. Receiving each message explicitly guarantees that no messages will be missed due to the asynchronous nature of message receipt from the host.

The *TimeOutValue* specifies, in milliseconds, how long the Receive should wait for a message to arrive before returning.

Returns:

If no messages are received from the host within the specified TimeOut value, the function will return a False (zero). If a message is received the function returns as True (-1).

RefreshScreen

Sub RefreshScreen

Description:

Force the visible screen to be repainted. This procedure should be called after entering text into the screen or moving the cursor. Only use this procedure in ClientMode.

Send

Function **Send**(*TextToSend* as Srtring) as Integer

Description:

In ClientMode, the Send method transmits a specified string directly to the transport without affecting the content of the screen. The string is passed unmodified—no control sequences are added. Keep in mind that the host application may not function correctly using this method, because it may require control sequences that separate fields, messages, etc. Normally, when data is sent from the screen to the host, it will contain additional control sequences (FCCs, field separation, start-of-entry position, etc.) which may be essential to the host application.

Returns:

The return value can be 0 or 1. A return of 0 means the send was not done because the session is not currently open. A return of 1 indicates the Send completed at the transport level.

SetCursorPosition

Sub **SetCursorPosition** (*Column* as Integer, *Row* as Integer)

Description:

Move the screen cursor to the specified *row* and *column* in the internal screen.

SetScreenText

Sub **SetScreenText** (*Column* as Integer, *Row* as Integer, Text as String)

Description:

Replace *text* in the internal screen starting at the specified *row* and *column*.

UnholdMessages

Sub UnholdMessages

Description:

Releases messages stopped by the HoldMessages statement. Messages will be received and processed in the normal manner.

Wait

Sub **Wait** (*WaitTime* as Integer)

Description:

Wait does a timed wait for the specified *WaitTime*. The *WaitTime* is specified in milliseconds.

WaitForString

Function **WaitForString** (*Column* as Integer, *Row* as Integer, *Text* as String,
TimeOut as Integer) as Integer

Description:

Wait for specified *text* to appear in the screen at the specified *column* and *row* position. If the *text* appears before the *TimeOut* expires, this function returns the value 1, otherwise, it returns a 0.

WaitForStringNot

Function **WaitForStringNot** (*Column* as Integer, *Row* as Integer, *Text* as String,
TimeOut as Integer) as Integer

Description:

Wait for specified *text* to be no longer present in the screen at the specified *column* and *row* position. If the *text* changes before the *TimeOut* expires, this function returns the value 1; otherwise, it returns a 0.

Key Code Values

The following list of key code values may be used with on the DoTerminalKey method. Note that the same list may be displayed in a separate window when working with the QPlexView Script Editor (see View Terminal Key Code Values on the Options menu).

T27 Keys:

Code Value	Key
249	ARROWDN
247	ARROWLEFT
248	ARROWRIGHT
246	ARROWUP
8	BACKSPACE
196	BACKTAB
218	BOUND
13	CARRIAGERTN
16442	CLRALLVTAB
134	CLREOL
135	CLREOP
159	CLRFORMS
128	CLRHOME
16432	COPY
164	CTRL
16431	CUT
234	DBLZERO
132	DELCHAR
16425	DELCHARPAGE
133	DELLINE
174	HOME
130	INSCHAR
16424	INSCHARPAGE
131	INSLINE
168	LOCAL
165	LOCKCTRL
16415	LOGICALEOL
217	MARK
138	MOVELINEDOWN
139	MOVELINEUP
253	NEXTPAGE
16434	PASTE
252	PREVPAGE
157	PRINTALL
156	PRINTUNPROT
214	RECALL
170	RECEIVE
136	ROLLDN
137	ROLLUP
158	SETFORMS
166	SPECIFY
213	STORE
198	TAB

Code Value	Key
141	TOGGLEFORMS
16441	TOGGLETAB
172	TRANSMIT
16428	TRANSMITLINE
236	TRIPZERO
210	UPPERONLYON
211	UPPERONLYUOFF
16426	WRITEESC
3	WRITEETX
16427	WRITEGS