



**QuickServe**

24 May 2007



## Table of Contents

QuickServe .....	1
QuickServe Introduction.....	1
Client System Requirements .....	1
Host Requirements .....	1
QuickServe Installed File List .....	1
QuickServe Overview.....	1
The QuickServe Environment .....	1
Developing a QuickServe Application .....	2
QuickServe Developer Overview.....	4
Components of QuickServe Developer.....	4
Getting Started with QuickServe Developer .....	4
QuickServe Developer.....	5
Q-LINK Server Settings .....	5
QuickServe Developer Tasks .....	5
File Menu .....	6
View Menu.....	7
Help Menu .....	7
QuickServe Sign On to Host.....	7
Host Connection Route.....	7
Your User-id .....	7
Your Password .....	7
Auto Sign-On Script.....	7
Debug Trace .....	8
QuickServe Request Editor .....	8
File Menu .....	8
Edit Menu.....	8
Search Menu.....	9
Bookmarks Menu.....	9
Options menu .....	9
Help .....	9
Editor Properties .....	9
Edit Window Font .....	9
Tab Size.....	9
Highlight Colors.....	9
OK.....	10
Cancel .....	10
Help .....	10
QuickServe Result Viewer.....	10
File Menu .....	10
Edit Menu.....	10
Help .....	10
The Samples .....	11
A QuickServe Application.....	11
Establishing s QuickServe Host Server Session.....	11
Main Subroutine .....	11

Connect Button .....	12
Sign_On Function .....	13
Get Accounts Button .....	14
Get Account List .....	14
Put & Get Host Message .....	16
Long Requests .....	16
General Sign-On Script .....	17
Script Files .....	18
Sample Sign-on Script .....	18
QuickServe Commands.....	21
QuickServe Command Processing .....	21
QuickServe Command Format.....	21
QuickServe Reply Format .....	21
QuickServe Command Processing .....	21
QuickServe Command Format.....	22
QuickServe Reply Format .....	22
RL Command - Retrieve Q-LINK Server Log .....	22
SR Command - Send Request .....	24
SS Command - Send Short Request .....	26
TS Command - Test Server Response .....	28
XX Command - Terminate Server .....	29
Index .....	31

## QuickServe

### QuickServe Introduction

The QuickServe application programming interface (API) provides a client/server interface that enables Microsoft Windows applications to access files and databases that reside on a Unisys 2200 Enterprise Server. The QuickServe API gives a Windows application the ability to read and update data on the Unisys 2200 through a simple, yet robust programming language (Q-LINK). The data on the Unisys 2200 can reside in PCIOS files, TIP/FCSS files, Unisys SDF files, DMS 2200 databases, RDMS databases, SFS files, and MAPPER reports (via DTM).

All communications between the Windows application and the Unisys 2200 Enterprise Server are handled by the WinQ Application Development Tools that comes with QuickServe. The WinQ API provides a channel through which the Microsoft Windows application (the client) can send requests and receive responses from the QuickServe Host Server running on the Unisys 2200 Enterprise Server. The QuickServe Host Server receives requests written in the Q-LINK language from the Microsoft Windows application and processes these requests through the Q-LINK environment on the Unisys 2200 Enterprise Server.

QuickServe also comes with a working QuickServe Windows application called QuickServe Developer. QuickServe Developer provides a quick and easy way to communicate with the QuickServe Host Server. QuickServe Developer can be used for developing and testing the Q-LINK procedures that will be used in your QuickServe applications. QuickServe Developer provides text file transfer between the Microsoft Windows PC and the Unisys 2200 Enterprise Server. QuickServe Developer can also be used to quickly extract data from the Unisys 2200 Enterprise Server when developing a permanent application is not necessary.

### Client System Requirements

- Windows NT 3.5 or higher; Windows ME, Windows 2000 or Windows XP,
- Any PC hardware platform capable of running one of the above Microsoft Windows versions, with at least 4MB of RAM (a minimum of 8MB is recommended);
- Approximately 1 MB of available disk space (local hard drive or network logical disk);
- UTS eXpress Pro.

### Host Requirements

- A Unisys 1100/2200 Enterprise Server with KMSYS Worldwide's Q-LINK product installed.

### QuickServe Installed File List

The following is a list of files that may be found upon completion of the QuickServe installation:

File	Description
Program Directory (normally, C:\Program Files\KMSystems\QuickServe\2.0):	
QSERVE.EXE	QuickServe Developer Program
QSERVE.HLP	QuickServe Help File
Example Directory (normally, C:\Program Files\KMSystems\QuickServe\2.0\EXAMPLEAPP):	
DEMO1.FRM	Demo Application Main Form
DEMO2.FRM	Demo Application Form
DEMO.MAK	Demo Application Project File
DEMO.BAS	Demo Application Code Module
CUSTDEMO.TXT	Demo Application Q-LINK Request
ORDDEMO.TXT	Demo Application Q-LINK Request

### QuickServe Overview

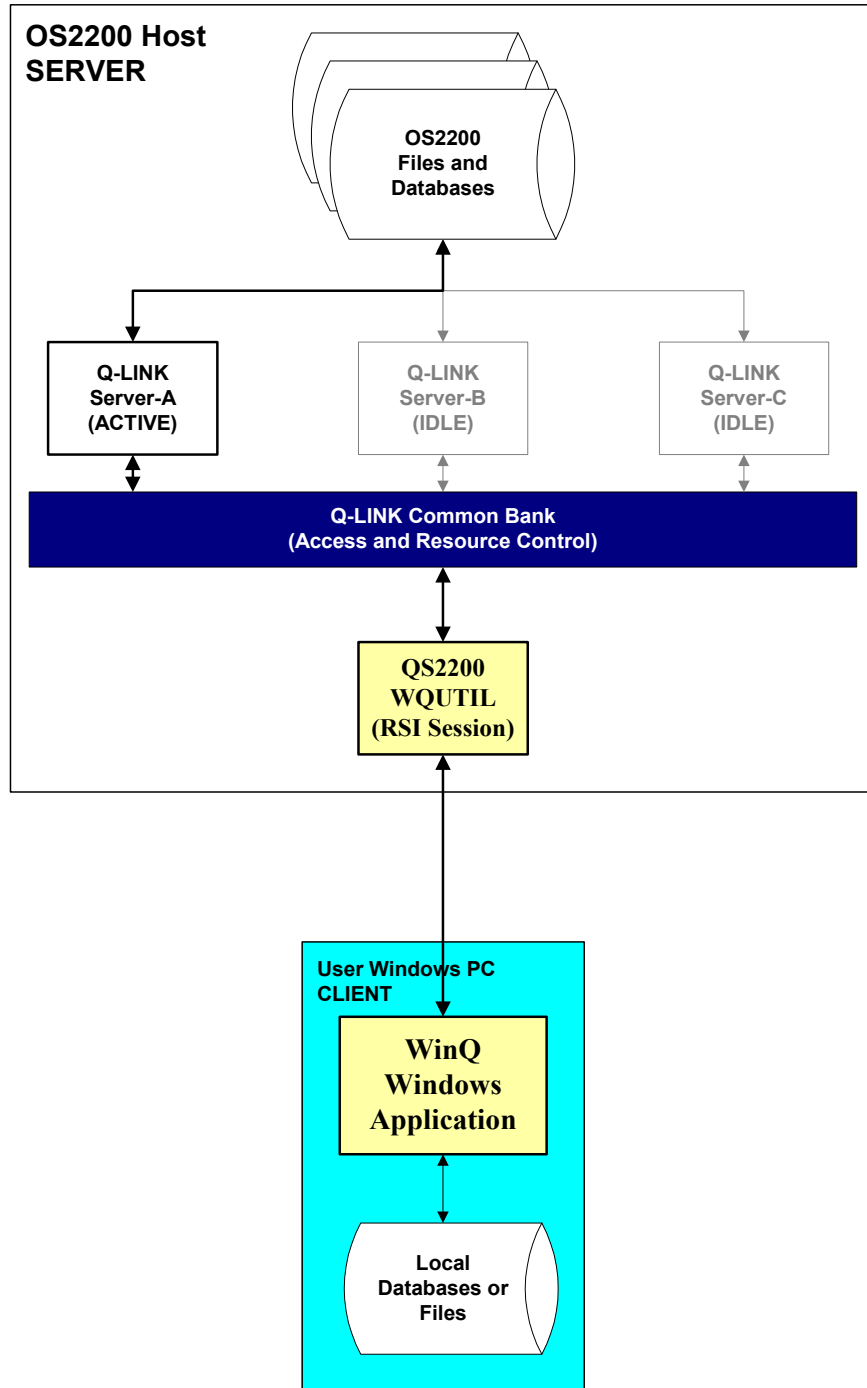
This chapter describes how QuickServe, the WinQ Development Tools, and the QuickServe Host Server program interface with each other across the Windows and Unisys 2200 software architectures.

### The QuickServe Environment

A QuickServe request passes through several different environments before the request is processed and a response is returned from the Unisys 2200 Enterprise Server. The QuickServe API, WinQ Development Tools, and QuickServe Host Server enable a Windows programmer to traverse these disparate environments.

The Windows application uses the WinQ functions to send Q-LINK requests (programs) to the QuickServe Host Server program. If the Q-LINK request is over ten lines, the windows application uses the QuickServe Host Server's IN-FILE as a staging area for the Q-LINK program. The WinQ file transfer

function uploads the files containing Q-LINK programs directly to the QuickServe Host Servers IN-FILE. The QuickServe Host Server links to the Q-LINK common bank which allocates the request to an idle Q-LINK server. If all the Q-LINK servers are busy, a new Q-LINK server is started to handle the request.



The QuickServe Host Server sends request status information that the Windows application retrieves with a WinQ function. The data that is returned, as a result of the QuickServe request, is placed in the QuickServe Host Server's OUT-FILE. The Windows application uses the WinQ API to download the result from the QuickServe Host Server's OUT-FILE to a PC file. The host data, which now resides in the PC file, can be displayed or processed by the Windows application.

**Developing a QuickServe Application**

A Windows application sends QuickServe commands to the QuickServe Host Server and receives replies and data from the QuickServe Host Server through the WinQ Applications functions. The following

sections describe the windows components that are required to use the WinQ Application functions and the general steps that normal Windows applications perform when using QuickServe to access databases and files on the host.

### **Windows Components Required to Use QuickServe**

The following Windows component must exist within the Windows application's project before using the WinQ Application functions to send QuickServe commands to the QuickServe Host Server:

- WinQ Custom Communications Control -WinQUTS32X.OCX from the UTS eXpress Pro install.

### **Host Components Required to Use QuickServe**

The only host component required by a QuickServe application is the QuickServe Host Server program. The QuickServe Host Server is executed by the WinQ sign-on script. After the WinQ sign-on script executes the QuickServe Host Server it should check for a reply of "<OK:QUTILREADY" (use the WaitForSpecificString function) which is QuickServe's confirmation that the QuickServer Host Server is initialized and ready to receive QuickServe commands. The string will begin in column 25 of the row containing the text cursor (normally, row 23 of a 24-row screen).

After confirmation has been received that the QuickServe Host Server is initialized, it sends a reply of "<OK:QUTILREADY" back to the script. The WinQ script must be coded to execute the SetSignOnResult subroutine, setting it to "True", and exit the script after the script receives the "ready" reply. Control then returns back to the procedure in the Windows application that executed the RunScript function.

### **QuickServe Order of Processing**

This section introduces the general steps that a Windows application will take to access files and databases on a Unisys 2200 Enterprise Server using QuickServe.

#### **Open a Session with the Communications Manager**

The Windows application uses the WinQ function, OpenSession, to establish a session with the communications manager.

#### **Sign On to the 2200 Host and Start QuickServe Host Server**

Next, the Windows application must use a WinQ sign-on script to connect to the host with a DEMAND (RSI) session. The eXpress Plus Script Editor program/icon is included with UTS eXpress Pro to allow the development of custom sign-on scripts (subroutines), as sign-on procedures may vary from site to site. The Windows application uses the WinQ API RunScript function to execute the object sign-on script.

The last action of the WinQ sign-on script before relinquishing control to the Windows application should be to execute the QuickServe Host Server program. After the QuickServe Host Server is initialized, it will send a reply of "<OK:QUTILREADY" back to the script. The script, in preparation for receiving this reply, should issue a WaitForSpecificString function. Upon receiving the confirmation, the sign-on script sets SetSignOnResult to "True" and exits the subroutine, thus returning control back to the procedure in the Windows application that executed the RunScript function.

#### **Send QuickServe Requests and Process Replies**

The Windows application can now use WinQ API functions to send QuickServe commands and files containing Q-LINK requests to the QuickServe Host Server. The Windows application also uses the WinQ API functions to get the host replies and data files that result from the QuickServe commands that were sent.

Normally, the Windows application creates a Q-LINK request in a file and uses the WinQ API SendFile function to upload the request to the QuickServe Host Server's IN-FILE. The Windows application then uses the WinQ API SendApp function to send a QuickServe SR command which tells the QuickServe Host Server to process the Q-LINK request in the IN-FILE.

However, if the Q-LINK request is ten (10) lines or less, the request can be sent as part of the QuickServe SS command. Using SS command makes the step of uploading the Q-LINK request to the QuickServe Host Server's IN-FILE unnecessary. The SS command is especially useful for running Q-LINK requests that have been saved in object form on the Unisys 2200 Enterprise Server (see the SAVE directive in the Q-LINK Programmer Reference). These stored procedures can be executed by the Windows application using an SS command that contains a single Q-LINK RUN directive (see the RUN directive in the Q-LINK Programmer Reference).

The QuickServe Host Server sends a reply which is read by the Windows application using the WinQ API GetMessage function. The Windows application processes the reply based on the value in the Q-LINK status code in the QuickServe Request Status indicating whether the request successfully completed or failed.

If necessary, the Windows application uses the GetFile function to retrieve the QuickServe Host Server's OUT-FILE which contains the result of the Q-LINK request. The Windows application then displays or processes the downloaded data. At this point, the Windows application can send requests and process replies until no more data is needed from the Unisys 2200 Enterprise Server.

#### **Terminate the QuickServe Host Server**

Before the Windows application ends, it must signal the QuickServe Host Server to close. The Windows application uses the SendApp function to send a QuickServe XX command (see XX Command - Terminate Server) which instructs the QuickServe Host Server to terminate

#### **Sign Off Host and Close the Communications Manager Session**

The Windows application next calls the WinQ API CloseSession function to close the session with the communications manager.

### **QuickServe Developer Overview**

QuickServe Developer is a QuickServe application that provides a generalized way to communicate with a QuickServe Host Server on the host 2200. A Windows application designer will find that QuickServe Developer's functions can be used to eliminate many time-consuming tasks. These uses might include:

- Q-LINK program testing and development
- Transferring text files between a Windows PC and the Unisys 2200 Enterprise Server
- Performing quick one-time data extractions when a permanent application is not necessary
- Confirming that a Q-LINK Server is functioning
- Retrieving a Q-LINK Server Log

QuickServe Developer can be very beneficial to a developer because it gives hands on experience using the QuickServe and WinQ application interfaces. Using QuickServe Developer's functions enables developers to visualize how the QuickServe and WinQ functions work, making the implementation of QuickServe and WinQ functions into Windows applications easily understood. Turning on WinQ Message Tracing (see the WinQ User Guide ) while running QuickServe Developer shows the developer the conversation that takes place between a QuickServe Windows application and the QuickServe Host Server. QuickServe Developer gives the developer the chance to gain familiarity with a working QuickServe application immediately after installing QuickServe on the PC.

#### **Components of QuickServe Developer**

The QuickServe Developer application has four main components:

- The QuickServe Developer Main Window
- The Request Editor
- The Result Viewer
- File Transfer

#### **Getting Started with QuickServe Developer**

Clicking the QuickServe Developer icon causes the QuickServe Developer main window to display. Initially, the QuickServe Developer main window contains only one enabled command button: Sign On to Host.

Clicking the Sign On to Host command button starts the QuickServe Sign On to Host dialog. On this dialog, you select a route, enter a host user-id and password and select a sign-on script. The route must be for a DEMAND session and is configured with the Configure function of the UTS eXpress Plus emulator (press the Configure button on the UTS eXpress Plus control panel and press the Configure Routes button on the Screen Connections tab). A sign-on script establishes a DEMAND session on the host 2200 and initiates the QuickServe Host Server. Most QuickServe Developer functions require that a QuickServe Host Server is active on the Unisys 2200. A sample sign-on script is installed with QuickServe and may be altered as needed (see <drive>:\Program Files\KMSystems\QuickServe\SampleScript\QServ.bas). If your site requires information for sign-on other than user-id and password (e.g., project-id, clearance level, etc.), the sample sign-on script will prompt you for that information via additional dialogs (not shown); however. Note: if your site requires you to enter a @RUN card or if you are using TOMF (Terminal Operator Menu Facility), you will need to alter the sample script accordingly.

## QuickServe Developer

The QuickServe Developer program contains the necessary controls to develop and prototype Q-LINK requests. For those familiar with the MAPPER run, QUTIL, that is included with Q-LINK, QuickServe Developer has the same functionality, but from Windows instead of MAPPER.

After a session is established with a QuickServe Host Server, two group boxes appear on the QuickServe Developer main window. These group boxes contain the Q-LINK Server Settings and the Tasks that QuickServe Developer can perform.

### Q-LINK Server Settings

The Q-LINK Server Settings are used to send required and optional information to the Q-LINK common bank concerning the Q-LINK Server which will be used to process the Q-LINK requests. These parameters include:

#### Q-LINK Server Class (Required)

Specifies the Q-LINK Server Class under which the request is to run.

#### Request Max Time (Required)

Specifies the maximum time that the request will be allowed to run before the server terminates the request.

#### Max Result Lines (Optional)

Specifies the maximum number of lines to be returned by the Q-LINK request.

#### Q-LINK Password (Optional)

A password that allows the Q-LINK configuration limits to be overridden.

#### Server Run-Id (Optional)

The run-id of a specific server to which the request is being directed.

### QuickServe Developer Tasks

Five tasks are initiated by clicking one of the command buttons in the Basic Q-LINK Functions group.

#### Send Request from Editor

Clicking the Send Request from Editor command button sends the Q-LINK request in the Request Editor (use the F3 key to activate the QuickServe Request Editor dialog) to the QuickServe Host Server. The QuickServe Host Server routes the request to the Q-LINK common bank where the request is processed by a Q-LINK server from the Q-LINK server class specified in QuickServe Developer's Q-LINK Server Class parameter. After the Q-LINK request is processed, QuickServe Developer displays a message stating "Q-LINK request processing complete" and the Q-LINK Status Codes are displayed on the QuickServe Developer main window status bar at the bottom of the window. Note: If you do not see these codes in the status bar, make sure the Auto-Retrieve Reply check box is not checked. You can subsequently view the result using the Retrieve Result button.

The Q-LINK Status Codes returned are:

- STAT1 - The QLK error number (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- STAT2 - The number of lines in the result.
- STAT3 - A user status code returned by the Q-LINK STOP command (see the STOP command in the Q-LINK Programmer Reference).

#### Send Request from File

The Send Request from File command button sends a Q-LINK request that is stored in a PC file to the QuickServe Host Server. After clicking the Send Request from File command button, the user is asked to select the PC file that contains the Q-LINK request to be run. The selected request is sent to the QuickServe Host Server, which in turn, routes it to the Q-LINK common bank. The Q-LINK common bank allocates a Q-LINK server to process the request from the server class specified in the Q-LINK Server Class parameter. After the request is processed, QuickServe Developer displays a message stating "Q-LINK request processing complete" and displays the Q-LINK Error Status Codes in the QuickServe Developer main window status bar. Note: If you do not see these codes in the status bar, make sure the Auto-Retrieve Reply check box is not checked. You can subsequently view the result using the Retrieve Result button.

The Q-LINK Status Codes returned are:

- STAT1 - The QLK error number (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- STAT2 - The number of lines in the result.
- STAT3 - A user status code returned by the Q-LINK STOP command (see the STOP command in the Q-LINK Programmer Reference).

### Quick Request

This control, presents a simple dialog for entering a small Q-LINK request on a single line. Multiple commands may be entered in the text box shown but must be separated by the caret (^) symbol. For example:

```
D DATE^ D TIME^RUN
```

All other Q-LINK conventions must be observed.

### Retrieve Result

The Retrieve Result command button downloads the result produced by a Q-LINK request to an intermediate file on the PC. After the result is downloaded, the QuickServe Result Viewer dialog is activated to display the contents of this file.

### Test Server Response

The Test Server Response command button is used to check to see if QuickServe Developer is communicating properly with the QuickServe Host Server and whether the specified Q-LINK server on the host is able to process requests. The Test Server Response command sends a short request to a Q-LINK server of the class specified in the Q-LINK Server Class parameter. If the QuickServe Host Server successfully delivers the request to the Q-LINK common bank and the Q-LINK request is processed, a message box is posted stating that the test of the server is complete. If the QuickServe Host Server cannot deliver the request to the Q-LINK common bank or the Q-LINK common bank cannot start a Q-Link server to process the request, QuickServe Developer posts an error message. The Q-LINK Status Codes are displayed in the QuickServe Developer main window status bar. Note: If you do not see these codes in the status bar, make sure the Auto-Retrieve Reply check box is not checked. You can subsequently view the result using the Retrieve Result button.

The Q-LINK Status Codes returned are:

- STAT1 - The QLK error number (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- STAT2 - The number of lines in the result.
- STAT3 - A user status code returned by the Q-LINK STOP command (see the STOP command in the Q-LINK Programmer Reference).

### Retrieve Server Log

Clicking the Retrieve Server Log command button causes the QuickServe Host Server to download the Q-LINK server log of the Q-LINK server specified in the Q-LINK Server Class and Server Run-id parameters. The server log may then be viewed with the Result Viewer.

### File Transfer

The QuickServe Developer Text File Transfer dialog allows text files to be uploaded to the 2200 host computer and downloaded from the 2200 host computer. After the File Transfer command button is clicked on the QuickServe Developer Main Window, QuickServe Developer presents this dialog box. This dialog box includes a text box for the name of the host file involved in the file transfer procedure and two radio buttons that are used to specify the transfer direction.

After the 2200 EXEC file name is entered in the host file text box and the transfer direction is selected, click the OK button to bring up another dialog box. This second dialog box asks for the PC file involved in the file transfer operation.

The file transfer is initiated when the OK command button on this dialog is clicked. When the file transfer operation is finished, a message box is displayed to inform the user whether the file transfer was complete or unsuccessful.

Note: The host file must exist on the host 2200 in order for the file transfer to complete successfully.

## File Menu

### Configure File Paths

This control may be used to specify the drive and path to the directory where the temporary transfer files and Q-LINK requests are to be located.

**Close**

Use this selection to exit the QuickServe Developer.

**View Menu****Request Editor (F3)**

Use this selection to open the QuickServe Request Editor.

**Result Viewer (F4)**

The Retrieve Result command button downloads the result produced by a Q-LINK request to an intermediate file on the PC. After the result is downloaded, the QuickServe Result Viewer dialog is activated to display the contents of this file.

**WinQ Trace to Window**

Use this selection to trace communications between the QuickServe Developer PC component (QServe.exe) and the host program (SYS\$LIB\$\*QSERVE.QSERV). The communications trace will be displayed in a separate window.

**WinQ Trace to File**

Use this selection to write the trace to a text file.

**WinQ Trace to Both Window and File**

Use this selection to display the trace in a window as well as writing it to a file.

**Close WinQ Trace**

Use this selection to terminate the trace.

**Help Menu****Contents**

This selection displays this topic.

**About**

Use this selection to display the version and build number of QuickServe Developer.

**QuickServe Sign On to Host**

This window is used to initially connect, sign on and execute the host component on the host server. QuickServe uses a sign-on script that mimics the sign-on steps that you would take when using an emulator. The window contains the minimum controls (user-id and password) that you need to sign on to the 1100. If your sign-on requires that you enter such things as account number and project-id, the script should prompt you for those values.

Note: There is a generalized sign-on script (QServe.bas) developed at KMSYS Worldwide that should meet most sign-on requirements at any site. The script should require little or no modification at most sites; however, if your site requires you to enter an @RUN statement or uses TOMF to make the host connection, you will need to add the necessary script commands to enter the run card or simulate the keystrokes required by TOMF. The generalized script may be found in the QuickServe installation directory (normally, C:\Program Files\KMSystems\QuickServe\2.0\SampleScripts\QServe.bas).

**Host Connection Route**

With this drop-down list box, select a previously configured, DEMAND route. A route, open id. and virtual destination must be configured on the QPort TCP/IP Visual Configuration window of UTS eXpress Plus. To open the QPort TCP/IP Visual Configuration window, press the **Configure** button on the UTS eXpress Plus Control Panel, select the **Screen Connections** tab and click the **Configure Routes** button.

**Your User-id**

In this text box, enter your OS-1100 user-id.

**Your Password**

In this text box, enter your OS-1100 password.

**Auto Sign-On Script**

This text box allows you to select a Sign-On Script file (.BAX or .BAS). The sign-on script is required when connecting to the 2200 host. To develop a sign-on script or alter the sample script provided, use the **Express Plus32 Script Editor** program/icon located in the **UTS eXpress Pro** folder/program group. Complete on-line help for the **eXpress Plus Script Editor** is provided.

You may enter the complete drive, path and file specification directly into the box or use the **Browse** button to locate a new script.

### **Debug Trace**

Use these controls to trace communications between the QuickServe Developer client on the PC and the QuickServe host server during sign-on.

#### **None**

With this option button set (the default), no trace will occur.

#### **Trace to Window**

Set this option to displace the trace in a separate window.

#### **Trace to File**

Set this option to write the trace to a text file.

#### **Trace to Window and File**

Set this option to trace to both a window and a file.

### **QuickServe Request Editor**

From this window, Q-LINK source programs (requests) may be developed and saved to PC files. A request can be submitted directly from the QuickServe Developer main window to the QuickServe Host Server using the Send Request from File function

#### **File Menu**

##### **New (Ctrl+N)**

Create a new request.

##### **Open (Ctrl+O)**

Load an existing request in the Request Editor.

##### **Save (Ctrl+S)**

Save changes to the open request.

##### **Save As**

Allow the request to be saved into another PC file.

##### **Clear File List**

Clear the list of most recently accessed files located at the bottom of the File menu.

##### **Print (Ctrl+P)**

Start the Print dialog to print the request. Settings that may be changed a dependant upon to print drivers installed in Windows.

##### **Page Setup**

This selection allows you to choose the font to be used when printing the request. You may also change the paper orientation and add a page header and/or footer.

##### **Editor Properties**

Use this command to edit the properties of the request editor: window font, highlight colors and tab stops.

##### **Close (Alt+F4)**

Return control back to the QuickServe Developer Main Window.

#### **Edit Menu**

##### **Undo (Ctrl+Z)**

Cancel a change.

##### **Redo Shift+Ctrl+Z)**

Reapply a change.

##### **Cut (Ctrl+X)**

Move selected text to the Windows Clipboard.

##### **Copy (Ctrl+C)**

Copy selected text to the Windows Clipboard.

**Paste (Ctrl+V)**

Paste selected text from the Windows Clipboard.

**Delete (Ctrl+D)**

Delete selected text.

**Select All (Ctrl+A)**

Select all lines in the request.

**Word Wrap (Ctrl+W)**

Making this selections causes words to wrap to the next line when they are not viewable within the text area.

**Search Menu****Find (Ctrl+F)**

Display a dialog for entering search parameters and initiating the search.

**Repeat Find (F3)**

Repeats the last Find.

**Replace (Ctrl+R)**

Display a dialog for entering replacement parameters and initiate the search for values to be replaced.

**Go to Line (CtrlG)**

Display a dialog to enter a line number and go to that line in the code.

**Bookmarks Menu****Set Bookmark 1 through 5 (Shift+F1 through Shift+F5)**

Use these selections to bookmark lines in the request. To bookmark a line, place the text cursor anywhere on the line and make one of the five Set Bookmark selections.

**Go to Bookmark 1 through 5 (Ctrl+F1 through Ctrl+F5)**

Use these selections to go to a bookmark set by one of the Set Bookmark commands.

**Options menu****Show Toolbar**

Use this toggle to show or hide the Request Editor Toolbar..

**Syntax Highlight**

Use this toggle to show or hide highlighted syntax..

**Help**

Get help for this dialog.

**Editor Properties**

This dialog is used to change the properties or appearance of a request/result displayed in the text area of the dialog.

**Edit Window Font**

The controls in this group affect the font typeface, size and intensity used to display the request.

**Font Name**

From this drop-down list box, choose from the list of non-proportional, fixed fonts installed on your PC.

**Size**

With this spin wheel, increase or decrease the font size.

**Bold**

Check this box to increase the font intensity.

**Tab Size**

With this spin wheel, increase or decrease the number characters between tab characters.

**Highlight Colors**

Use this group to assign colors to different parts of the request text.

**Set Text Color**

To change color, select the type of text (Normal text, Strings, etc.) and select from this drop-down list box to change the foreground.

**Set Background Color**

To change the background color, select from this drop-down list box.

**OK**

Click this button to accept the changes made and exit the dialog.

**Cancel**

Click this button to discard the changes made and exit the dialog.

**Help**

Click this button to receive on-line help for this dialog.

**QuickServe Result Viewer**

The QuickServe Result Viewer is used to see the result returned from a Q-LINK request submitted by QuickServe Developer. The result is loaded to the Result Viewer when the Retrieve Result function successfully. The Result Viewer allows the user to scroll down and across the result if the result does not fit within the window.

**File Menu****Save As (Ctrl+S)**

Open the Save Result as Text File dialog to save the result to a PC file.

**Print (Ctrl+P)**

Open the Print dialog for the default Windows printer and print the result.

**Page Setup**

Open the Print Page Setup dialog to make font, header/footer and page orientation selections prior to printing.

**Editor Properties**

Change the properties or appearance of a request/result displayed in the text area of the dialog

**Close (Alt+F4)**

Close the viewer and return to the QuickServe Developer dialog.

**Edit Menu****Copy (Ctrl+C)**

Copy selected text to the Windows Clipboard.

**Find (Ctrl+F)**

Display a dialog for entering search parameters and initiating the search.

**Repeat Find (F3)**

Repeats the last Find.

**Help**

Get help for this dialog.

## The Samples

### A QuickServe Application

A demonstration Visual Basic application called DEMO is included with QuickServe. All the necessary forms and code libraries for DEMO are present in the QuickServe installation directory after installation of the QuickServe software. To see how DEMO works, open the project file DEMO.MAK in Visual Basic. The following sections of this chapter will use the DEMO application to demonstrate how a QuickServe application should be developed.

### Establishing a QuickServe Host Server Session

Before any Windows applications can access data on the host computer, it must first establish a demand session on the Unisys 2200 Enterprise Server and execute the QuickServe Host Server. The Windows application uses a sign-on script to establish this link to a Unisys 2200 Enterprise Server. A developer must create a WinQ Script tailored to the site's sign-on procedures before the QuickServe application can be run. The developer may examine and alter the sample script provided by KMSYS Worldwide that can be found in the installation directory.

View the sample sign-on script.

### Main Subroutine

This routine initializes several variables and displays the first form, DEMO1:

```
Sub Main()
    App.Title = "Q-LINK Demo"
    DEMO1.WinQ.Visible = False

    SvrClass = "PROD1" ' Default Q-LINK server class
    Maxlines = 2000    ' Max lines from Q-LINK
    Password = "      "
    RunId = "          "
    Work_Dir = App.Path
    ' DEMO1.Txt_Script = "" ' Default sign-on script
    DEMO1.Show
End Sub
```

Notice that the application must have the WinQ custom control placed somewhere on the first form (see the graphic at the bottom left-hand corner of the form). The WinQ control does not have any code associated with it and may be made invisible when the form is displayed (see above).

Like most controls, the WinQ controls have properties, events and methods that apply to WinQ (see the WinQ User Guide on KMSYS Worldwide's Software & Documentation CD). To include a WinQ custom control in your application, you must add the file WinQUTS32X.OCX to your project (e.g., in Visual Basic, use the Custom Controls selection from the Tools menu). These files will have been installed and registered with Windows when you installed UTS eXpress Pro.

### Connect Button

The WinQ sign-on script for the DEMO application expects two input variables: an EXEC 2200 user-id and password. The first form loaded by the DEMO application contains edit boxes for a sign-on script name, the user-id and the password. After these values are entered, the command button, Btn\_Connect, which is labeled, Connect, is clicked to initiate the session establishment:

```

Private Sub Btn_Connect_Click()
    If Left$(Btn_Connect.Caption, 1) = "D" Then
        If Signed_On Then Sign_Off
        End
    End If

    Usr = UCase$(RTrim$(Txt_User))
    Pw = UCase$(RTrim$(Txt_Password))
    WinQ_Path = UCase$(RTrim$(Txt_Script))

    If Len(Usr) = 0 Or Len(Pw) = 0 Or Len(WinQ_Path) = 0 Then
        MsgBox "User id., Password and Script entries are required to connect to host.", MB_ICONEXCLAMATION
        Txt_User.SetFocus
        Exit Sub
    End If
    WinQ.OperationMode = 0
    If Not Sign_On() Then
        Exit Sub
    End If
    Signed_On = True

```

```

WinQ.OperationMode = 1
Btn_Connect.Caption = "Dis&connect"
Chk_Trace.Enabled = False
Txt_User.Enabled = False
Txt_Password.Enabled = False
Txt_Script.Enabled = False
Frm_Acct_Sel.Visible = True
Btn_Connect.DEFAULT = False
Btn_Accts.DEFAULT = True
Txt_State.SetFocus
DoEvents
End Sub

```

### Sign\_On Function

The Sign\_On function from the DEMO application (see DEMO.BAS) demonstrates how to establish the session on the Unisys 2200 with the RunScript function in WinQ.

```

Function Sign_On() As Integer
    Dim ans As Integer
    Dim rep As String
    Dim Path As String
    Dim Rslt As Integer

    Hold True, "Processing Sign On."
    DEMO1.WinQ.ScreenVisible = 0
    DEMO1.WinQ.OpenSession ("ROUTE1")

    If DEMO1.Chk_Trace.Value = CHECKED Then
        DEMO1.WinQ.TraceOption = 1
    End If

    Rslt = DEMO1.WinQ.RunScript(WinQ_Path, 2, Usr, Pw, "", "", "", "")
    If Rslt = 0 Then
        MsgBox "Sign-on script processing failed.", MB_ICONEXCLAMATION
        Rslt = DEMO1.WinQ.CloseSession
        Sign_On = False
    Else
        Sign_On = True
    End If

    Hold False, "Connected to host."
End Function

```

### Get Accounts Button

Once sign-on is complete, the user can initiate the database search by enter a state abbreviation and city then clicking the Get Accounts button. The Btn\_Accts\_Click subroutine will call the Get\_Account\_List Subroutine:

```
Private Sub Btn_Accts_Click()
    Dim X As Integer

    st = Txt_State.Text
    city = Txt_City.Text
    If st = " " Then
        MsgBox "Plese enter a State abbreviation.", MB_ICONEXCLAMATION
        Txt_State.SetFocus
        Exit Sub
    End If
    Lst_Accounts.Clear
    Get_Account_List
    If QS1 <> 0 Then
        MsgBox "Q-LINK Error:" + Format$(QS1), MB_ICONEXCLAMATION
        Exit Sub
    End If
    If QS2 = 0 Then
        MsgBox "No customers found", MB_ICONEXCLAMATION
        Exit Sub
    End If
    For X = 1 To Acct_List_Cnt
        Lst_Accounts.AddItem Right$(Acct_list(X), Len(Acct_list(X)) - 10) + "(" + Left$(Acct_list(X), 9) + "
    Next X
End Sub
```

### Get Account List

The Get\_Account\_List subroutine passes the state and city as part of a message sent to the host server program, QSERV. The complete message is formatted as a Send Short request (SS command) consisting 10 or fewer Q-LINK lines, the Q-LINK server class, maximum number of result lines, etc.

The following Type statement is used to describe the format of QuickServe commands. This code is in the General Declarations Section of the code module, DEMO.BAS.

```
Type QUTIL_CMD_TYPE
    Cmd As String * 2
    SvrClass As String * 6
    Maxlines As String * 8
    Password As String * 6
    RunId As String * 6
    CsfImg As String * 80
End Type
```

At the beginning of the Get\_Account\_List subroutine, the application sets up the QuickServe request. Since the Q-LINK request being sent is less than ten lines long, the Send Short Request (SS command) is used. Notice that the Q-LINK command lines in the data portion of the request are delimited by tab characters - CHR\$(9). In reality, the Q-LINK program that is run on the Unisys 2200 Enterprise Server is much longer. The SS command is only acting as a trigger to run a saved Q-LINK procedure. The only Q-

LINK command being sent is the first tab delimited line (RUN CUSTDEMO ...), the other line is the input criteria (state and city). DEMO uses the WinQ\_Send\_App function (see SS Command - Send Short Request) to send the SS command to the QuickServe Host Server.

Once the message is constructed, the Put\_Get\_Host\_Msg function (see below) will send the message to the host for processing by the QuickServe host program.

If the execution is successful, the subroutine uses the WinQ\_GetFile method to return the result to the PC and store it into a temporary file. The PC file will be read and the contents placed in a list box on the form for viewing.

```

Sub Get_Account_List()
    Dim Msg As QUTIL_CMD_TYPE
    Dim Reply As String
    Dim Fn As Integer
    Dim Rslt As Integer
    Dim X As Integer
    ' Create and send a short request.
    Hold True, "Selecting accounts for specified State/City"
    Msg.Cmd = "SS" ' Send short request.
    Msg.SvrClass = SvrClass
    Msg.Maxlines = Format$(Maxlines, "00000000")
    Msg.Password = Password
    Msg.RunId = RunId
    Msg.CsfImg = "RUN CUSTDEMO FROM BOB*QSERVE" & Chr$(9) & st & city ' & Chr$(9) & "#EOF"
    If Not Put_Get_Host_Msg(Msg, Reply, MISC_TIMEOUT) Then
        Hold False, "WinQ Error"
        Exit Sub
    End If
    Get_QLINK_Stats Reply
    If QS1 <> 0 Or QS2 = 0 Then ' Error or Nothing found
        Hold False, ""
        Exit Sub
    End If
    Hold True, "Downloading accounts."

    ' Retrieve results of request.
    Rslt = DEMO1.WinQ.GetFile("OUT-FILE", Work_Dir + "\QLINK$$$TMP", 0, 300)
    If Rslt = 0 Then
        MsgBox "WinQ_Get_File error.", "", MB_ICONEXCLAMATION
        Hold False, "Get file failed"
        QS3 = 0
        Exit Sub
    End If

    ' Get downloaded data into a list box.
    Fn = FreeFile
    Open Work_Dir + "\QLINK$$$TMP" For Input As Fn
    ReDim Acct_list(1 To QS2)
    Acct_List_Cnt = 0
    Do While Not EOF(Fn)
        Line Input #Fn, Reply

```

```

    If Left$(Reply, 1) <> "." And Left$(Reply, 1) <> " " Then
        Acct_List_Cnt = Acct_List_Cnt + 1
        Acct_list(Acct_List_Cnt) = RTrim$(Reply)
    End If
Loop
Close Fn

Hold False, ""
Exit Sub

Get_Account_List_File_Err:
MsgBox "Error one temporary request file." + Chr$(13) + Error$, MB_ICONEXCLAMATION
Hold False, ""
Exit Sub

End Sub

```

### Put & Get Host Message

This function uses the WinQ SendApp method to send the constructed message to the host server for processing. The host server program makes the call to the QLSIM processor (see the Q-LINK PRM) to execute the Q-LINK request.

Upon completion of the request, the function uses the GetApp method to return the execution status of the Q-LINK request.

```

Function Put_Get_Host_Msg(Cmd As QUTIL_CMD_TYPE, Reply As String, TimeOut As Integer) As Integer
    Dim Tmp As GEN_TYPE
    Dim Rslt As Integer

    LSet Tmp = Cmd
    Rslt = DEMO1.WinQ.SendApp(Tmp.Rec, TimeOut)
    If Rslt = 0 Then
        MsgBox "WinQ_Send_App error.", MB_ICONEXCLAMATION
        Put_Get_Host_Msg = False
        Exit Function
    End If
    Reply = DEMO1.WinQ.GetApp(TimeOut)
    If DEMO1.WinQ.LastStatus <> 0 Then
        MsgBox "WinQ_Get_Msg error.", MB_ICONEXCLAMATION
        Put_Get_Host_Msg = False
        Exit Function
    End If
    Put_Get_Host_Msg = True
End Function

```

There is more to this sample Windows application than is documented here; however, the methodology is pretty much the same for the other application functions.

The following two sections describe how to handle Q-LINK requests that are longer than 10.

### Long Requests

If the Q-LINK request being sent is greater than ten lines, DEMO uses the WinQ\_Send\_File function (see the WinQ User Guide) to upload the Q-LINK request to the QuickServe Host Server's IN-FILE. DEMO then uses the WinQ\_Send\_App function to send a QuickServe Send Request (SR command). The SR command signals the QuickServe Host Server to submit the contents of its IN-FILE to Q-LINK for

processing. This code fragment from the Get\_Details subroutine of DEMO.BAS shows an example of the SR command processing:

```

' Create a request in a file. Multiple accounts may be selected.
Fn = FreeFile
Open Work_Dir + "\QLINK$$$TMP" For Output As Fn
Print #Fn, "RUN ORDDMO FROM BOB*QSERVE"
For X = 0 To DEMO1!Lst_Accounts.ListCount - 1
    If DEMO1!Lst_Accounts.Selected(X) Then
        Print #Fn, Left$(Acct_list(X + 1), 9)
        Cnt = Cnt + 1
    End If
Next X
Print #Fn, "#EOF"
Close Fn
' Make sure user selected at least one account from list.
If Cnt = 0 Then
    MsgBox "Nothing selected.", MB_ICONEXCLAMATION
    Hold False, ""
    Exit Sub
End If
' Transfer request file to host.
Rslt = DEMO1.WinQ.SendFile("IN-FILE", Work_Dir + "\QLINK$$$TMP", 0, 30)
If Rslt = 0 Then
    MsgBox "WinQ_Send_File error.", "", MB_ICONEXCLAMATION
    Hold False, ""
    Exit Sub
End If

' Tell QUTIL to run it.
Hold True, "Selecting details for specified accounts."
Msg.Cmd = "SR"
Msg.SvrClass = SvrClass
Msg.Maxlines = Format$(Maxlines, "00000000")
Msg.Password = Password
Msg.CsfImg = ""
If Not Put_Get_Host_Msg(Msg, Reply, MISC_TIMEOUT) Then
    Hold False, "WinQ Error"
    Exit Sub
End If
Get_QLINK_Stats Reply
If QS1 <> 0 Or QS2 = 0 Then
    Exit Sub ' Error or Nothing found
End If

```

### General Sign-On Script

Sign-on scripts are used to automatically establish the session between the WinQ client and server application programs. Session establishment can be very different from site to site due to differences in

security needs, communication networks and many other factors. The sign-on script makes the task of session establishment and application initialization transparent to the QuickServe application.

Once a sign-on script is set up, the QuickServe application simply calls the WinQ RunScript function (see, above, and the WinQ User Guide) that processes the script. The WinQ RunScript function returns to the application when the sign-on is complete. Once the sign-on process is complete, the WinQ application can proceed with its conversation. When the WinQ application is ready to exit, the WinQ RunScript function is called once more to process the sign-off portion of the script. Using WinQ Scripts, the application can run in any environment without regard to differences in session establishment procedures.

### Script Files

WinQ scripts are stored in two formats: a source file with an extension of WQS and an executable file with an extension of WQX. The script source files can be viewed and edited using the WinQ Script Manager. You can also view (but not update) a script source file with any other text editor. The executable script file is the one that the application will use to sign on. Executable scripts are stored in an encrypted format so that sensitive information such as user ids. and passwords cannot be viewed. Only the executable script file is needed by the application. The source script need not (and should not) be distributed with the application.

### Sample Sign-on Script

The following script is provided by KMSYS Worldwide as an example and generally will handle most sign-on scenarios. If your site uses the Terminal Operator Menu Facility (TOMF), the script will need to be slightly altered to handle the required screen selections. The same is true if your site requires a @RUN statement to sign-on.

The sign-on script below is compatible with the DEMO application provided with this product. The DEMO application displays a sign-on dialog that allows the user to enter the minimum parameters required to sign-on to most 2200s: user-id and password. If, however, the script detects that there are other parameters that the user is required to enter (e.g., proj-id, account number, etc.), the script is set to prompt for those values with the Input\$ function.

Notice at the bottom of the script that the host server program is executed (@xqt SYS\$LIB\$\*QSERVE.QSERV). The host server program then announces that it is ready to converse with the client (<OK:QUTILREADY) and the sign-on script is exited.

```

Sub Main()
  Dim PrompRow as Integer
  dim Txt As String
  dim blankline As String
  dim Unfinished As integer
  dim x as integer
  Dim NewString as String
  dim Rslt As Integer
  dim TCnt As Integer

' *** Automatic Sign On Script

'Moves cursor to find the correct start position
UTSKey UK_CURSOR_TO_HOME
UTSKey UK_CURSOR_UP
PromptRow = GetCursorRow() - 2

'Look for "Enter Session Establishment"
Txt = GetScreenText(2, PromptRow + 1, 10)
If UCase$(Txt) = "ENTER SESS" Then
  NewString = InputBox$("Enter Session Establistment.", "$$Open")
  EnterText Chr(30) + NewString
  UTSKey UK_TRANSMIT_KEY
  Wait 1000

```

```

end if

TCnt = 0
Unfinished = True

While Unfinished
  'Get screen Text from 3rd line and text in 2nd line
  Txt = UCase$(GetScreenText(2, PromptRow, 20))
  blankline = Trim$(GetScreenText(2, PromptRow + 1, 10))

  if Trim$(blankline) = "" then
    Select Case Txt
      Case "ENTER YOUR USER-ID/P" 'Look for "Enter Your User-Id/Password:"
        ' prompts user for UserId and password in a dialog box if necessary
        if GotSignOnPrompt = True then
          MsgBox "Sign-On Recieved Incorrect Password", mb_IconExclamation, "Bad Sign-on from Script"
          SetSignOnResult False
          Unfinished = False
          Exit Sub
        end if

        GotSignOnPrompt = True

        UP = GetUserParam(1) + "/" + GetUserParam(2)

        EnterText Chr$(30) + UP
        UTSKey UK_TRANSMIT_KEY
        TCnt = 0

      Case "ENTER YOUR NEW PASSW" 'Look for "Enter your new password"
        NewString = ""
        NewString = InputBox$("Enter Your New Password.", "New Password")
        If NewString = "" Then
          MsgBox "You did not enter a new password so sign-on script is quitting."
          SetSignonResult False
          Unfinished = False
          Exit Sub
        End If
        EnterText Chr(30) + NewString
        UTSKey UK_TRANSMIT_KEY
        TCnt = 0

      Case "ENTER YOUR CLEARANCE" 'Now input clearance level if necessary
        NewString = InputBox$("Enter Your Clearance Level.", "Clearance Level")
        EnterText Chr(30) + NewString
        UTSKey UK_TRANSMIT_KEY
        TCnt = 0
    End Select
  End If
End While

```

```
Case "ENTER YOUR ACCOUNT N" 'Now input Account number if necessary
  NewString = InputBox$("Enter Your Account Number.", "Account Number")
  EnterText Chr(30) + NewString
  UTSKey UK_TRANSMIT_KEY
  TCnt = 0

Case "CHOOSE YOUR ACCOUNT " 'Now input Account Index if necessary
  NewString = InputBox$("Enter Your Account Index.", "Account Index(1-5)")
  EnterText Chr(30) + NewString
  UTSKey UK_TRANSMIT_KEY
  TCnt = 0

Case "ENTER YOUR PROJECT-I" 'Now input a project Id if necessary
  NewString = InputBox$("Enter Project Id.", "Id")
  EnterText Chr(30) + NewString
  UTSKey UK_TRANSMIT_KEY
  TCnt = 0
End Select

'Look for correct text that you are signed on
If Left$(Txt, 6) = "DATE: " Then
  Wait 1000
  EnterText "@xqt SYS$LIB$*QSERVE.QSERV "
  UTSKey UK_TRANSMIT_KEY
  If not WaitForSpecificString(23, 25, 14, "<OK:QUTILREADY") Then Exit Sub

  SetSignOnResult True
  Unfinished = False
  Exit Sub
End If

End If

TCnt = TCnt + 1
If TCnt > 60 Then
  MsgBox "Sign-On Timed Out", mb_IconExclamation, "Bad Sign-on from Script"
  SetSignonResult False
  Unfinished = False
End If

Wait 500

Wend

End Sub
```

## QuickServe Commands

### QuickServe Command Processing

The Windows application communicates with the QuickServe Host Server by sending QuickServe command records through the WinQ\_Send\_App function (see the WinQ User Guide). The QuickServe Host Server responds to these commands by sending a reply record back to the Windows application. The Windows application uses the WinQ\_Get\_Msg function to receive the message. A normal reply is indicated by "<OK:" in the Indicator field (see below).

### QuickServe Command Format

All QuickServe commands have a common format:

	<u>Position</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
Command	1	2	Alphabetic	A valid QuickServe command.
Server Class	3	6	Alphanumeric	A valid Q-LINK Server Class name to which the current request is directed.
Max Lines	9	8	Numeric	Optional. Maximum reply lines allowed for this Q-LINK request.
Password	17	6	Alphanumeric	Optional. Q-LINK password that allows Q-LINK configuration limits to be overridden.
Run-Id	23	6	Alphanumeric	Optional. The run-id of a specific Q-LINK server to which the request is to be directed.
Data	29	972	Alphanumeric	Optional. Data furnished for this command. Currently only the SS and CS commands require data. - For the SS command the data portion may contain from 1 to 10 tab delimited Q-LINK request lines - For the CS command the data is a complete CSF image.

### QuickServe Reply Format

The QuickServe Host Server replies to all commands with a message in the following format:

	<u>Position</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
Field Name Indicator	1	4	Alphanumeric	Indicates the type of reply. A normal QuickServe reply is indicated by "<OK:" in this field.
Request Status	5	11	Alphanumeric	Contains status information for the specific command that caused the server to send the reply.
Request line count	16	11	Numeric	After processing a Q-LINK request this field contains the number of lines in the Q-LINK result. This does not include lines written to alternate print files or other data files.
User Exit Code	27	11	Numeric	After processing a Q-LINK request, this field contains the exit code set by the Q-LINK program (see the STOP command in the Q-LINK Programmer Reference).

### QuickServe Command Processing

The Windows application communicates with the QuickServe Host Server by sending QuickServe command records through the WinQ\_Send\_App function (see the WinQ User Guide). The QuickServe Host Server responds to these commands by sending a reply record back to the Windows application. The Windows application uses the WinQ\_Get\_Msg function to receive the message. A normal reply is indicated by "<OK:" in the Indicator field (see below).

**QuickServe Command Format**

All QuickServe commands have a common format:

	<b><u>Position</u></b>	<b><u>Length</u></b>	<b><u>Type</u></b>	<b><u>Description</u></b>
Command	1	2	Alphabetic	A valid QuickServe command.
Server Class	3	6	Alphanumeric	A valid Q-LINK Server Class name to which the current request is directed.
Max Lines	9	8	Numeric	Optional. Maximum reply lines allowed for this Q-LINK request.
Password	17	6	Alphanumeric	Optional. Q-LINK password that allows Q-LINK configuration limits to be overridden.
Run-Id	23	6	Alphanumeric	Optional. The run-id of a specific Q-LINK server to which the request is to be directed.
Data	29	972	Alphanumeric	Optional. Data furnished for this command. Currently only the SS and CS commands require data. - For the SS command the data portion may contain from 1 to 10 tab delimited Q-LINK request lines - For the CS command the data is a complete CSF image.

**QuickServe Reply Format**

The QuickServe Host Server replies to all commands with a message in the following format:

	<b><u>Position</u></b>	<b><u>Length</u></b>	<b><u>Type</u></b>	<b><u>Description</u></b>
Field Name Indicator	1	4	Alphanumeric	Indicates the type of reply. A normal QuickServe reply is indicated by "<OK:" in this field.
Request Status	5	11	Alphanumeric	Contains status information for the specific command that caused the server to send the reply.
Request line count	16	11	Numeric	After processing a Q-LINK request this field contains the number of lines in the Q-LINK result. This does not include lines written to alternate print files or other data files.
User Exit Code	27	11	Numeric	After processing a Q-LINK request, this field contains the exit code set by the Q-LINK program (see the STOP command in the Q-LINK Programmer Reference).

**RL Command - Retrieve Q-LINK Server Log**

The RL command retrieves the Q-LINK Server log of the specified Q-LINK Server. After the RL command successfully completes, the Q-LINK Server log will reside in the QuickServe Host Server's OUT-FILE. The Windows application should use the WinQ\_Get\_File function (see the WinQ User Guide) to download the QuickServe Host Server's OUT-FILE to a PC file where the contents of the log can be displayed or processed.

The RL command returns the following QuickServe Host Server reply fields:

- Indicator – "<OK:" indicates that the RL command was successfully passed to the QuickServe Host Server.
- Request Status – Q-LINK request status (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- Request Line Count – The number of lines in the log returned to QuickServe Host Server's OUT-FILE.

Example:

```

Type QUTIL_CMD_TYPE
  Cmd As String * 2
  SvrClass As String * 6
  Maxlines As String * 8
  Password As String * 6
  RunId As String * 6
  QServData As String * 80
End Type

Dim Msg As QUTIL_CMD_TYPE
Dim Server_Class As String
Dim Tmp As String
Dim TimeOut As Integer
Dim Reply As String

' Setup QuickServe RL command
  Msg.Cmd = "RL"
  Server_Class = "PROD1"
  Msg.SvrClass = Server_Class
  Msg.Maxlines = Format$(Maxlines, "00000000")
  Msg.Password = "QLINK"
  Msg.RunId = ""
  Msg.QServData = ""
  TimeOut = 120
  Tmp = Msg.Cmd + Msg.SvrClass + Msg.Maxlines + Msg.Password + Msg.RunId + Msg.QServData

' Send QuickServe RL command with WinQ_Send_App function
  Rslt = WinQ_Send_App(1, Tmp, TimeOut)
  ' Check status of WinQ_Send_App function
  If Rslt <> 0 Then
    If Rslt >= -2 Then
      MsgBox "WinQ_Send_App timed out.", MB_ICONEXCLAMATION
    Else
      WinQ_Status_Msg "WinQ_Send_App error.", "", MB_ICONEXCLAMATION
    End If
  Else
    ' Use WinQ_Get_Msg to get QuickServe host reply
    Rslt = WinQ_Get_Msg(1, Reply, TimeOut, True)
    If Rslt <> 0 Then
      If Rslt >= -2 Then
        MsgBox "WinQ_Get_Msg timed out.", MB_ICONEXCLAMATION
      Else
        WinQ_Status_Msg "WinQ_Get_Msg error.", "", MB_ICONEXCLAMATION
      End If
    Else
      If Left$(Reply, 4) <> "<OK:" Then
        ' Bad reply from QuickServe Host Server
        MsgBox "Error Retrieving Q-LINK Server Log for" & Server_Class, MB_ICONEXCLAMATION
      End If
    End If
  End If

```

```

Else
    ' Retrieve Q-Link ServeLog
    Rslt = WinQ_Get_File(1, "OUT-FILE", "C:\" &
        Trim$(server_Class) & ".LOG", False, 300)
    ' Check WinQ_Get_File status
    If Rslt <> 0 Then
        If Rslt >= -2 Then
            MsgBox "WinQ_Get_File timed out.", MB_ICONEXCLAMATION
        Else
            WinQ_Status_Msg "WinQ_Get_File error.", "", MB_ICONEXCLAMATION
        End If
        MsgBox "Log download failed", MB_ICONEXCLAMATION
    End If
End If
End If
End If

```

### SR Command - Send Request

The SR command tells the QuickServe Host Server to send the Q-LINK request in its IN-FILE to the Q-LINK server class specified in the command. The Windows application must upload the request to the QuickServe Host Server's IN-FILE with the WinQ\_Send\_File function (see the WinQ User Guide) before sending the SR command. The result of the SR command is placed in the QuickServe Host Server's OUT-FILE and must be downloaded by the Windows application using the WinQ\_Get\_File function.

The SR command returns the following QuickServe Host Server reply fields:

- Indicator - "<OK:" indicates that the SR command was successfully passed to QuickServe Host Server.
- Request Status - Q-LINK request status (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- Request Line Count - Q-LINK line count returned to the QuickServe Host Server's OUT-FILE.
- User Exit Code - The exit code set by the Q-LINK program (see the STOP command in the Q-LINK Programmer Reference).

Example:

```

Cmd As String * 2
Type QUTIL_CMD_TYPE
    SvrClass As String * 6
    Maxlines As String * 8
    Password As String * 6
    RunId As String * 6
    QServData As String * 80
End Type

Dim Msg As QUTIL_CMD_TYPE
Dim Rslt As Integer
Dim Reply As String
Dim TimeOut As Integer
Dim Tmp As String

' Transfer file containing Q-LINK Request to QuickServe
' Host's IN-FILE with WinQ_Send_File function

```

```

Rslt = WinQ_Send_File(1, "IN-FILE", "C:\QLINK.REQ", False, 300)
' Check WinQ_Send_File status
If Rslt <> 0 Then
  If Rslt >= -2 Then
    MsgBox "WinQ_Send_File timed out.", MB_ICONEXCLAMATION
    Exit Sub
  Else
    WinQ_Status_Msg "WinQ_Send_File error.", "", MB_ICONEXCLAMATION
    Exit Sub
  End If
End If

' Setup SR command to tell QuickServer Host Server to run
' the request in its IN-FILE.
Msg.Cmd = "SR"
Msg.SvrClass = "PROD1"
Msg.Maxlines = Format$(Maxlines, "00000000")
Msg.Password = ""
Msg.QServData = ""
TimeOut = 120
Tmp = Msg.Cmd + Msg.SvrClass + Msg.Maxlines + Msg.Password + Msg.RunId + Msg.QServData

' Send SR command to QuickServe Host Server with
' WinQ_Send_App command
Rslt = WinQ_Send_App(1, Tmp, TimeOut)
' Check WinQ_Send_App status
If Rslt <> 0 Then
  If Rslt >= -2 Then
    MsgBox "WinQ_Send_App timed out.", MB_ICONEXCLAMATION
    Exit Sub
  Else
    WinQ_Status_Msg "WinQ_Send_App error.", "", MB_ICONEXCLAMATION
    Exit Sub
  End If
End If

' Use WinQ_Get_Msg to get QuickServe Host Server reply
Rslt = WinQ_Get_Msg(1, Reply, TimeOut, True)
' Check WinQ_Get_Msg status
If Rslt <> 0 Then
  If Rslt >= -2 Then
    MsgBox "WinQ_Get_Msg timed out.", MB_ICONEXCLAMATION
    Exit Sub
  Else
    WinQ_Status_Msg "WinQ_Get_Msg error.", "", MB_ICONEXCLAMATION
    Exit Sub
  End If
End If

```

```

' Check QuickServe Host Server reply
If Left$(Reply, 4) <> "<OK:" Then
    MsgBox "Error Sending Q-LINK Request to " & Server_Class, MB_ICONEXCLAMATION
    Exit Sub
End If

' Check Q-LINK Status
If Mid$(Reply, 5, 11) <> "0000000000" Then
    MsgBox "Error Returned from Q-LINK Server " & Trim$(Mid$(Reply, 5, 11))
    Exit Sub
End If

' Check Lines Returned from Q-LINK
If Mid$(Reply, 16, 11) > "0000000000" Then
' Get Result
    Rslt = WinQ_Get_File(1, "OUT-FILE", "C:\Q_LINK.RST", False, 300)
    If Rslt <> 0 Then
        If Rslt >= -2 Then
            MsgBox "WinQ_Get_File timed out.", MB_ICONEXCLAMATION
        Else
            WinQ_Status_Msg "WinQ_Get_File error.", "", MB_ICONEXCLAMATION
        End If
        MsgBox "Download of Q-LINK Result failed", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

```

### SS Command - Send Short Request

The SS command is used to send a Q-LINK request of ten (10) lines or less to a Q-LINK server. A server class must be specified in the QuickServe command record's Server Class command field. The Q-LINK request is placed in the QuickServe command record's data field. Each line of the Q-LINK command must be delimited by a tab character (CHR\$(9)). The result of the SS command is placed in the QuickServe Host Server's OUT-FILE and must be downloaded using the WinQ\_Get\_File function (see the WinQ User Guide) by the Windows application for processing.

The SS command returns the follows QuickServe Host Server reply fields:

- Indicator - "<OK:" indicates that the SS command was successfully passed to QuickServe Host Server.
- Request Status - Q-LINK request status (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- Request Line Count - Q-LINK line count returned to QuickServe Host Server's OUT-FILE.
- User Exit Code - The exit code set by the Q-LINK program (see the STOP command in the Q-LINK Programmer Reference).

Example:

```

Dim Msg As QUTIL_CMD_TYPE
Dim Rslt As Integer
Dim Reply As String
Dim TimeOut As Integer
Dim Tmp As String
Dim State As String * 2
Dim City As String * 30

```

```

' Setup QuickServer SS command
Msg.Cmd = "SS"
Msg.SvrClass = "PROD1"
Msg.Maxlines = Format$(Maxlines, "00000000")
Msg.Password = ""
State = "OR"
City = "Corvallis"
Msg.QServData = "RUN CUSTORD1" + Chr$(9) + State + City + Chr$(9) + "#EOF"
TimeOut = 120
Tmp = Msg.Cmd + Msg.SvrClass + Msg.Maxlines + Msg.Password + Msg.RunId + Msg.QServData

' Use WinQ_Send_App to send SS command to QuickServe Host
' Server
Rslt = WinQ_Send_App(1, Tmp, TimeOut)
' Check WinQ_Send_App status
If Rslt <> 0 Then
    If Rslt >= -2 Then
        MsgBox "WinQ_Send_App timed out.", MB_ICONEXCLAMATION
        Exit Sub
    Else
        WinQ_Status_Msg "WinQ_Send_App error.", "", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

' Use WinQ_Get_Msg to get QuickServe Host Server reply
Rslt = WinQ_Get_Msg(1, Reply, TimeOut, True)
' Check WinQ_Get_Msg Status
If Rslt <> 0 Then
    If Rslt >= -2 Then
        MsgBox "WinQ_Get_Msg timed out.", MB_ICONEXCLAMATION
        Exit Sub
    Else
        WinQ_Status_Msg "WinQ_Get_Msg error.", "", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

' Check QuickServe Host Server reply
If Left$(Reply, 4) <> "<OK:" Then
    MsgBox "Error Sending Q-LINK Request to " & Server_Class, MB_ICONEXCLAMATION
    Exit Sub
End If

' Check Q-LINK Status
If Mid$(Reply, 5, 11) <> "00000000000" Then
    MsgBox "Error Returned from Q-LINK Server " & Trim$(Mid$(Reply, 5, 11))
    Exit Sub

```

```

End If

' Check Lines Returned from Q-LINK
If Mid$(Reply, 16, 11) > "00000000000" Then
    ' Transfer Result using WinQ_Get_File
    Rslt = WinQ_Get_File(1, "OUT-FILE", "C:\Q_LINK.RST", False, 300)
    ' Check WinQ_Get_File Status
    If Rslt <> 0 Then
        If Rslt >= -2 Then
            MsgBox "WinQ_Get_File timed out.", MB_ICONEXCLAMATION
        Else
            WinQ_Status_Msg "WinQ_Get_File error.", "", MB_ICONEXCLAMATION
        End If
        MsgBox "Download of Q-LINK Result failed", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

```

### TS Command - Test Server Response

The TS command is used to check the status of the QuickServe Host Server and Q-LINK servers. The TS command causes the QuickServe server to generate a short Q-LINK request and send it to the server class specified in the QuickServe command's Server Class command field.

The TS command returns the following QuickServe Host Server's reply fields:

- Indicator - "<OK:" indicates that the TS command was successfully passed to QuickServe Host Server.
- Request Status - Q-LINK request status from the request generated by the QuickServe host server (see Appendix C, "QLK External Function Errors," in the Q-LINK Programmer Reference).
- Request Line Count - Q-LINK line count returned to the QuickServe Host Server's OUT-FILE.

Example:

```

Dim Msg As QUTIL_CMD_TYPE
Dim Rslt As Integer
Dim Reply As String
Dim TimeOut As Integer
Dim Tmp As String

' Setup QuickServe TS command
Msg.Cmd = "TS"
Msg.SvrClass = "PROD1"
Msg.Maxlines = Format$(Maxlines, "00000000")
Msg.Password = ""
Msg.QServData = ""
TimeOut = 120
Tmp = Msg.Cmd + Msg.SvrClass + Msg.Maxlines + Msg.Password + Msg.RunId + Msg.QServData

' Use WinQ_Send_App to send TS command to QuickServe Host Server
Rslt = WinQ_Send_App(1, Tmp, TimeOut)
' Check WinQ_Send_App status
If Rslt <> 0 Then

```

```

    If Rslt >= -2 Then
        MsgBox "WinQ_Send_App timed out.", MB_ICONEXCLAMATION
        Exit Sub
    Else
        WinQ_Status_Msg "WinQ_Send_App error.", "", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

' Get QuickServe Host Server's reply
Rslt = WinQ_Get_Msg(1, Reply, TimeOut, True)
' Check WinQ_Get_Msg reply
If Rslt <> 0 Then
    If Rslt >= -2 Then
        MsgBox "WinQ_Get_Msg timed out.", MB_ICONEXCLAMATION
        Exit Sub
    Else
        WinQ_Status_Msg "WinQ_Get_Msg error.", "", MB_ICONEXCLAMATION
        Exit Sub
    End If
End If

' Check QuickServe Host Server's reply
If Left$(Reply, 4) <> "<OK:" Then
    MsgBox "QuickServe Host Server return bad status" & Left$(Reply, 4), MB_ICONEXCLAMATION
    Exit Sub
End If

' Check Q-LINK Status
If Mid$(Reply, 5, 11) <> "0000000000" Then
    MsgBox "Error Returned from Q-LINK Server " & Trim$(Mid$(Reply, 5, 11)), MB_ICONEXCLAMATION
Else
    MsgBox "QuickServe Host Server and Q-LINK Server " & Trim$(Msg.SvrClass) & " responding", MB_ICONINFORMATION
End If

```

### **XX Command - Terminate Server**

The XX command is used to tell the QuickServe Host Server to terminate. No other QuickServe command fields are required with the XX command. Normally after performing the XX command, the Windows application will use the WinQ CloseSession (see the WinQ User Guide) function to terminate the demand session on the Unisys.

The XX command does not return a reply to the Windows application.

Example:

```

Sub Sign_Off()
    Dim Rslt As Integer

    Screen.MousePointer = HOURLASS
    Rslt = DEMO1.WinQ.CloseSession

```

```
Signed_On = False  
Screen.MousePointer = DEFAULT  
End Sub
```

**Index**

<b>A</b>		QuickServe Installed File List	1
A QuickServe Application	11	QuickServe Introduction	1
<b>C</b>		QuickServe Overview	1
Client System Requirements	1	QuickServe Request Editor	8
CS Command - Submit CSF Request	21	QuickServe Result Viewer	10
<b>E</b>		QuickServe Sign On to Host	7
Editor Properties	9	<b>R</b>	
Example Directory	1	RL Command - Retrieve Q-LINK Server Log	22
<b>G</b>		<b>S</b>	
General Sign-On Script	17	SR Command - Send Request	24
<b>H</b>		SS Command - Send Short Request	26
Host Requirements	1	<b>T</b>	
<b>Q</b>		TS Command - Test Server Response	28
QuickServe Command Processing	21	<b>X</b>	
QuickServe Developer	5	XX Command - Terminate Server	29
QuickServe Developer Overview	4		