iSecurity CodeScope

User Guide

www.razlee.com

Contents

About this Manual

This user guide is intended for system administrators and security administrators responsible for the implementation and management of security on IBM i systems. However, any user with basic knowledge of IBM i operations will be able to make full use of this product after reading this book.

Raz-Lee takes customer satisfaction seriously. Our products are designed for ease of use by personnel at all skill levels, especially those with minimal IBM i experience. The documentation package includes a variety of materials to get you familiar with this software quickly and effectively.

This user guide, together with the iSecurity Installation Guide, is the only printed documentation necessary for understanding this product. It is available in HTML form as well as in user-friendly PDF format, which may be displayed or printed using Adobe Acrobat Reader version 6.0 or higher. If you do not have Acrobat Reader, you can download it from the Adobe website: http://www.adobe.com/. You can also read and print pages from the manual using any modern web browser.

This manual contains concise explanations of the various product features as well as step-by-step instructions for using and configuring the product.

Raz-Lee's iSecurity is an integrated, state-of-the-art security solution for all System i servers, providing cutting-edge tools for managing all aspects of network access, data, and audit security. Its individual components work together transparently, providing comprehensive "out-of-the- box" security. To learn more about the iSecurity Suite, visit our website at http://www.razlee.com/.

Intended Audience

The CodeScope User Guide document was developed for users, system administrators and security administrators responsible for the implementation and management of security on IBM® AS/400 systems. However, any user with a basic knowledge of System i operations is able to make full use of this document following study of this User Guide.

NOTE: Deviations from IBM® standards are employed in certain circumstances in order to enhance clarity or when standard IBM® terminology conflicts with generally accepted industry conventions.

This document may also serve for new versions' upgrade approval by management.

Native IBM i (OS/400) User Interface

CodeScope is designed to be a user-friendly product for auditors, managers, security personnel and system administrators. The user interface follows standard IBM i CUA conventions. All product features are available via the menus, so you are never required to memorize arcane commands.

Many features are also accessible via the command line, for the convenience of experienced users.

Conventions Used in the Document

Menu options, field names, and function key names are written in **Courier New Bold**.

Links (internal or external) are emphasized with underline and blue color as follows: "About this Manual" on the previous page.

Commands and system messages of IBM i® (OS/400®), are written in ***Bold Italic***.

Key combinations are in Bold and separated by a dash, for example: **Enter**, **Shift-Tab**.

Emphasis is written in **Bold.**

A sequence of operations entered via the keyboard is marked as

***STRAV* > 81 > 32**

meaning: Syslog definitions activated by typing ***STRAV***and selecting option: **81** then option: **32**.

Menus

Product menus allow easy access to all features with a minimum of keystrokes. Menu option numbering and terminology is consistent

throughout this product and with other Raz-Lee products. *To* select a menu option, simply type the option number and press **Enter**. The command line is available from nearly all product menus. If the command line does not appear (and your user profile allows use of the command line), press **F10** to display it.

Data Entry Screens

Data entry screens include many convenient features such as:

- Pop-up selection windows
- Convenient option prompts
- Easy-to-read descriptions and explanatory text for all parameters and options
- Search and filtering with generic text support

The following describes the different data entry screens.

- To enter data in a field, type the desired text and then press Enter or Field Exit
- To move from one field to another without changing the contents press Tab
- To view options for a data field together with an explanation, press F4
- To accept the data displayed on the screen and continue, press Enter

The following function keys may appear on data entry screens.

- **F1**: **Help** Display context-sensitive help
- **F3**: **Exit** End the current task and return to the screen or menu from which the task was initiated
- **F4**: **Prompt** Display a list of valid options for the current field or command. For certain data items, a pop-up selection window appears
- **F6**: **Add New** Create a new record or data item
- **F8**: **Print** Print the current report or data item
- **F9**: **Retrieve** Retrieve the previously-entered command
- **F12**: **Cancel** Return to the previous screen or menu without updating

Legal Notice

While Raz-Lee is doing its best to coordinate between this document and Raz-Lee's products, changes might occur. In case a change has been encountered, please inform Raz-Lee. Raz-Lee keeps its right to modify the software or the document as per its sole discretion Usage of this document, and all information (including product information) provided within, are subject to the following terms and conditions, and all applicable laws. If you do not agree with these terms, please do not access or use the remainder of this document.

This document contains highly confidential information, which is proprietary to Raz-Lee Security Ltd. and/or its affiliates (hereafter, "Raz-Lee"). No part of this document's contents may be used, copied, disclosed or conveyed to any third party in any manner whatsoever without prior written permission from Raz-Lee. The information included in this document is intended for your knowledge and for negotiation purposes only. Raz-Lee makes no implicit representations or warranties with respect to such information. The information included in this document is subject to change without notice. Any decision to rely on the information contained herein shall be at your sole responsibility, and Raz-Lee will not accept any liability for your decision to use any information or for any damages resulting therefrom. Certain laws do not allow limitations on implied warranties or the exclusion or limitation of certain damages. If these laws apply to you, some or all of the above disclaimers, exclusions, or limitations may not apply to you.

All registered or unregistered trademarks, product names, logos and other service marks mentioned within this document are the property of Raz-Lee or their respective owners. Nothing contained herein shall be construed as conferring by implication, estoppels, or otherwise any license or right, either express or implied, under any patent or trademark of Raz-Lee or any third party. No use of any trademark may be made without the prior written authorization of Raz-Lee. This document and all of its contents are protected intellectual property of Raz-Lee. Any copying, reprinting, reuse, reproduction, adaptation, distribution or translation without the prior written permission of Raz-Lee is prohibited.

Please check your End User License Agreement (EULA) for terms and Conditions.

Manual Revised: Monday, January 2, 2023

Contacts

Raz-Lee Security Inc. www.razlee.com
Marketing: marketing@razlee.com 1-888-RAZLEE-4 (1-888-7295334)
Support: support@razlee.com 1-888-RAZLEE-2 (1-888-7295332)

About This User Guide

Who should use this guide

This guide is for people who use CodeScope to generate query and menu windows on IBM i computers. The guide is primarily intended for technical and partially technical personnel.

What you should know

Using windows created with CodeScope is very simple and requires no special technical background. The user needs to know only the basics of display station operation. Help is only a keystroke away.

Creating query and menu windows and calling them from within application programs (manual or automatic implementation) requires a basic knowledge of IBM i application programming.

Welcome To CodeScope

Why CodeScope

Users of application programs are continually faced with the problem of having to come up with the correct code or value to enter into parameter and field prompts. Data entry programs in particular, present the user with a whole host of fields demanding the precise response - Customer and Account Numbers, Product and Payment Codes, etc. Human memory has its limits, as does computer memory, and is therefore ill equipped to remember every possible and permissible field code and value. How do your users locate the exact pieces of information they need, without having to interrupt their work? How can you make their application programs easier to use? Common approaches to cope with this situation can be found in every data processing environment:

- Multi-colored Post-It notes pasted all over user terminals listing the most often used codes and field values (as well as the user's password)
- Stacks of master file listings parked on each user's desk with place-holding pens, paper clips and 80 column cards (if you are not sure what these are, don't worry) protruding at every angle
- Stuck at the less than helpful "Incorrect field value" message, our brave user exits the application program, wanders through a series of menus until reaching the appropriate inquiry program, finds the required value, writes it down, exits the inquiry program, navigates through the menus back to the original program and enters the value - and then takes a break due to exhaustion
- A not so common approach - provide each user with two terminals - one for data entry and the other for inquiries
- Provide each user with a nice, friendly PC with pop-up window prompts - if you are ready for an extended vacation

What is CodeScope?

CodeScope is a query and menu window generator that lets you provide your users with the ability to open a window over their application screen to browse files, transfer code values, display menus and execute menu options - all without ever leaving the program they are working on.

With little or no programming effort you can give your users:

- Added power and functions through instant file queries, code look-ups and menuing.
- Applications that are easier to use by providing windows that let users isolate and transfer the data they want with only a minimal number of keystrokes, thereby making users more productive.
- Faster answers without lengthy searches, note taking and re-keying.
- The ability to join up to ten files to produce the data in a single window. The use of more than one file is transparent to the user when activating a window.

CodeScope functions can quickly and easily be adapted to existing application software - whether program source is available or not. CodeScope automatically gives users the same functionality and ease of use they have become accustomed to in PC programs.

How CodeScope Works

CodeScope lets programmers define query and menu windows, which can then be activated by users needing to search for values and execute additional functions from application programs

Query Windows

Whenever a user needs to enter a specific value (such as an Account Number) in an application program field, a query window can be automatically displayed, overlapping a portion of the user's screen.

The user can browse the file searching for the desired record and when the needed code or value is located, the user can select the code by moving the cursor to the displayed record and pressing Enter. The query window disappears and the original display is restored, with the selected code appearing in the appropriate field. Any number of windows can be activated from within an application.

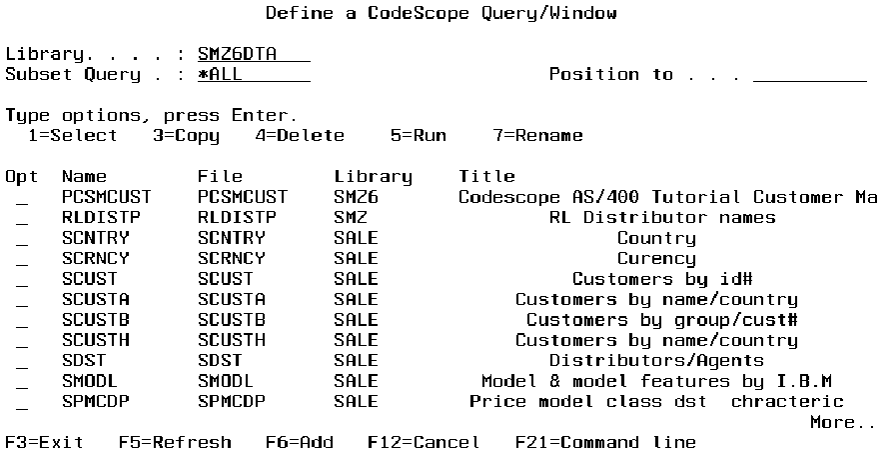
Menu Windows

Menu windows give the user the ability to display a menu associated with the current program and from that menu activate any system command, call other programs or invoke submenus.

A user can also activate any number of CodeScope query windows from CodeScope menu window.

What the Programmer Does

To generate a file query window, a programmer specifies the file and fields to display using the CodeScope window definition process.



Any keyed file or files can be displayed - physical, logical, joined, ***OPNQRYF***, externally defined or not. Any field within the file can be displayed. You can also display additional fields, call internal fields that are defined through the CodeScope window definition function.

During the window definition process, the programmer can optionally specify other parameters, such as default window size and location on the screen. Fields can be arranged in any order, field titles can be modified and limits can be set on the range of records the user can display.

Menu windows can also be defined to activate system commands, other programs, submenus or file query windows.

Once a query or menu window is defined, it can be called from any source program, including CL programs, with a simple call sequence. The programmer decides what type of user interface is most appropriate for detecting the need to activate a window. Among the most commonly used techniques are having the user press a predetermined function key (typically **F4**), key a question mark in the entry field, or leave the application entry field blank. When the application program detects a window request, it calls CodeScope, telling it which window to activate.

It takes just a few minutes to add windows to application programs - basically the entire work is done automatically. The only thing the user has to do is specify (using SDA) where he wants windows.

The application program can optionally override window parameters such as window size, location, range of records and starting key. After the call, CodeScope passes the selected record, if any, back to the calling program. Menu windows can also be activated without any modification to source programs. Programs can be associated with a particular menu, which can automatically be invoked via the Attention key.

Menu windows can also serve as an intermediate step toward activating a file query.

1. While using an application, the user presses the **Attn** key.
2. CodeScope displays the menu associated with the active application program.
3. The user places the cursor at a query window selection and presses **Enter**.
4. CodeScope displays the selected query window.

This technique is especially useful when source code is unavailable or where it is necessary to integrate CodeScope quickly into an existing application without having to make any source code modifications.

Time needed to define and add windows by the Programmer

It may take about one hour to define windows for 100 files.

It may take about one day to add windows to 100 files in the source level.

It may take about one day to add windows to a 250-program system in the object level.

Adding option windows manually may extend the time for each file window, or a few minutes per source program.

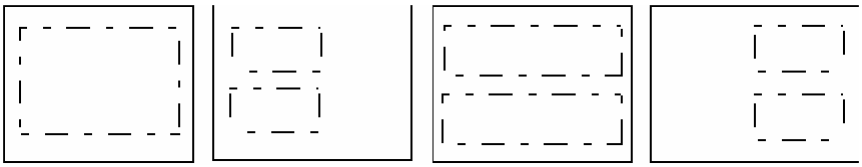
What the User sees

A user activates a window from within an application program through the interface method (function key, question mark, blank field, etc.) established by the application programmer.

** Accounts **		
Locate- 7234654:N Search- Account | Act | Short Name Code | Type |		
7234654 N		Jet Components
7234658 N		NASA Satellites
7234660 S		BOEING Jets
7234665 N		Comm. Services.
7234700 N		E.S.P. Comm. Inc.
7234775 N		Jet's Fuel Ltd.
7234778 S		Isaac Asimov- Cons.
7234779 N		Houston flights
7234789 S		E.T.Creatures Inc.
7234989 S		Exoft Systems
F3=Exit	F6=Print	F15/16/17=Search

The default window size is quarter screen and the default window location is the corner diagonally opposite to the current cursor position. These defaults can be overridden by the programmer either when the window is defined or as a call parameter. Once the window is displayed, the user can modify the window's appearance by moving it across the screen (up, down, left and right) and by changing its size (quarter screen, half screen, full screen).

To move the window to a new screen location or to change its size, the user presses the associated function key. The default window control function keys are:



Full Screen	Quarter	Half	Quarter
F21 = Full Screen	F22 = Left UP	F23 = Screen Up	F24 = Right Up
	F10 = Left Down	F11 = Screen Down	F12 = Right Down

Pressing the **Help** key displays the currently assigned function keys.

The user can page up and down through the queried file using the roll keys and can scroll across record fields using function keys.

Records can be retrieved by key and by partial key, a string of characters can be scanned across all alphanumeric fields and records can be searched according to a given field pattern or patterns.

Position By Key

The user can position the file browse by entering a key value in the **Position** field. If the file's key consists of several fields, the user separates the key values by commas.

Search by Record or by Pattern

The user can search across all alphanumeric window fields in each record by entering a search string in the **Search** field. If the string exists in a particular record, then that record is displayed at the beginning of a new window. The user enters the search string in upper case, but the search will match on any combination of upper case and lower case.

The pattern search scans for the value that the user enters in the reserved line below the field titles. Each character of the pattern is compared to the character directly below it, scanning for an exact match. Pattern search can be used for both numeric and alphanumeric fields and can be specified for multiple fields at once. The pattern can consist of upper case, lower case, or both. A match will only be found when the character and case are the same.

The user can search forwards and backwards through the file by pressing F16 or F15 respectively. Record and pattern searches can be combined in the same operation.

Using F17 will produce a window of just those records that match the scan criteria.

After a pre-specified number of records are searched without a match, the search process halts with an appropriate message. The user can continue searching from that point by pressing the search forward or backward function keys.

Selecting a record

To select a record, the user moves the cursor to the record and presses **Enter**. If the file query window was called from within an application program, then the record selected by the user is passed back to the application program.

Menu window options are also selected by moving the cursor to the appropriate line and pressing **Enter**.

.

Configuring CodeScope

To configure CodeScope, select **81. Configuration** from the main CodeScope menu (*STRCS*).

The **Define CodeScope global defaults** screen appears.

```
                  Define CodeScope global defaults          1/01/23 16:18:15

Standard line colors  . . . . . . . N            Y=Green & bold keys, N=No
If N=No:                        Underline  Color
  Odd lines . . . . . . . . . . . . N        B     B=Blue, P=Pink, G=Green
  Even lines  . . . . . . . . . . . N        P     B=Blue, P=Pink, G=Green

Window frame type . . . . . . . . . 2            1=Standard
                                                 2=Reverse image
Key values are in upper case  . . . Y            Y=Yes, N=No
Maximum number of records to SCAN . ___0         0=*NOMAX

Query from Menu returns to menu . . Y            Y=Yes, N=No
Default F-key for prompt  . . . . . F04          F01-F24
Request Enter after selecting by 1. N            Y=Yes, N=No

Generator remains ACTIVE in memory. N            Y=Yes, N=No


Release ID. . . . . . . . . . . . . 12.00 20-05-04   788C500  41A EP10       2
Authority Code  . . . . . . . . . .                                  2

F3=Exit       F19=Translate         F22=Enter Authority Code
```

The body of the screen contains the following fields:

Standard line colors

Y: The screen uses the standard green screen, with key symbols bolded.

N: The screen uses colors indicated in these fields:

Odd lines

This field contains subfields indicating how **Underline** and **Color** characters are shown in odd-numbered lines. If set to N, they are the default colors. If set to B, they are Blue; to n P, they are Pink, to G, they are Green.

Even lines

This field contains subfields indicating how **Underline** and **Color** characters are shown in even-numbered lines. If set

to N, they are the default colors. If set to B, they are Blue; to n P, they are Pink, to G, they are Green.

Window Frame Type

1: Standard

2: Reverse image

Key values are in upper case

Y: Yes

N: No

Maximum number of records to SCAN

The maximum numbers of records to scan. If set to 0 (the default), there is no maximum.

Query from Menu returns to menu

Y: Yes

N: No

Default F-key for prompt

The function key to be used for prompts. This can be from F01 to F24. F04 is the default.

Request Enter after selecting by 1

Y: Yes

N: No

Generator remains ACTIVE in memory.

Y: Yes

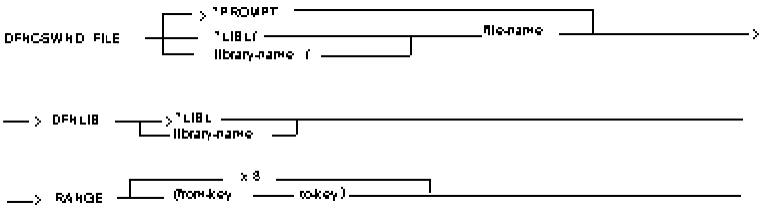
N: No

Defining Windows

The ***DFNCSWND*** command is used to define file query windows automatically or manually.

The ***DFNWND*** command is used to build query and menu windows and to set CodeScope environment parameters.

DFNCSWND - Define CodeScope window



The Define CodeScope window (*DFNCSWND*) command allows you to define a query window with or without any interactive intervention or with very limited interactive processing.

Note: this command can be run directly from the PDM screen by selecting option "**DW**" next to items in the PDM list screens.

FILE

The qualified name of a data file the window is based on.

***PROMPT**

Enter a conversational definition process. The possible library values are:

***LIBL**

search the library list to locate the file.

Library-name

the library where the file is located.

DFNLIB

The name of the library to store the definition. Proper definition files must exist in such a library. See Appendix B - CodeScope and Library.

***LIBL**

Use the job library list. If no definition files appear, the default library-SMZ6 is used.

Library name

Library name.

RANGE

Specify the range of keys that is included in this query window.

Up to 8 pairs of from-value and to-value can be specified for up to the first 8 key fields. The entire result from-key and to-key can be used for comparison.

The possible values for from-key are:

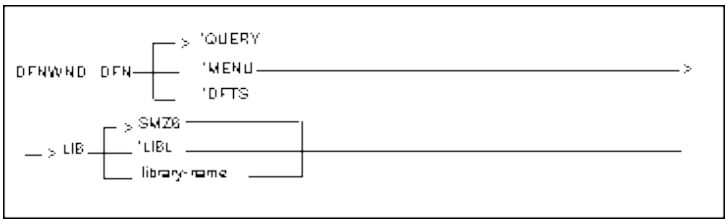
key-field-value

The lowest possible value for this key field.

***LDAnnnn**

The value starting from position nnnn in the *LDA (Local Data Area) is used.

The DFNWND (Define window) command



The *DFNWND* (Define window) command is used to build query and menu windows and to set CodeScope environment parameters.

DFN

The type of the required definition

***QUERY**

Build or change a query window definition

***MENU**

Build or change a menu window definition

***DFTS**

Modify system default parameters that apply to all CodeScope windows for the entire system or for a particular library.

LIB

The name of the library to store the definition. Proper definition files must exist in such a library. This enables the use of different languages, global defaults (such as scan scope, color of frame) or different menus for different users.

SMZ6

The system default library is used.

***LIBL**

Use the job library list. If no definition files appear, the default library, SMZ6, is used.

library name

Library name.

Building Query Windows

Query windows are built interactively by filling in the window definition fields. The query window definition becomes effective immediately - no compilation is required.

To build, update or delete a query window, enter ***DFNWND *QUERY*** or ***DFNCSWND *PROMPT*** on the command line. The following screen is displayed:

```
                    Define a CodeScope Query/Window

Library. . . . : SMZ6DTA___
Subset Query . : *ALL_____                    Position to . . . __________

Type options, press Enter.
  1=Select   3=Copy   4=Delete    5=Run    7=Rename

Opt  Name        File        Library    Title
 _   PCSMCUST    PCSMCUST    SMZ6       Codescope AS/400 Tutorial Customer Ma
 _   RLDISTP     RLDISTP     SMZ               RL Distributor names
 _   SCUST       SCUST       SALEINT             Customers by id#




                                                                       Botto
F3=Exit   F5=Refresh   F6=Add   F12=Cancel   F21=Command line
```

Options

More than one option can be chosen at the same time.

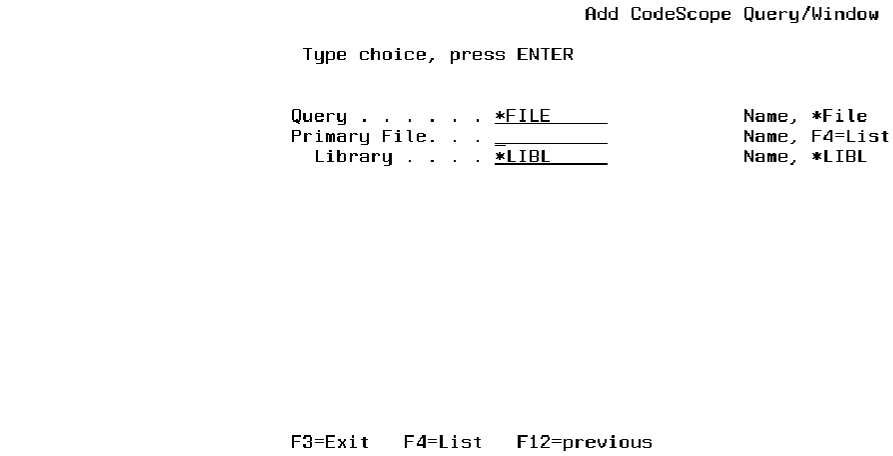
- **1 - Select**: Update an existing query window definition.
- **3 - Copy** : Copy a query window definition from an existing window definition.
- **4 - Delete** : Delete a query window definition.
- **5 - Run**: Activates a Window.
- **7 - Rename**: Rename a query window definition.

Key Functions

- F3 = Exit
- F5 = Refresh
- F6 = Add a new query window
- F12 = Cancel
- F21 = Command Line

Copy/Rename from

The name of the existing window that will be used for a copy or rename action.



File

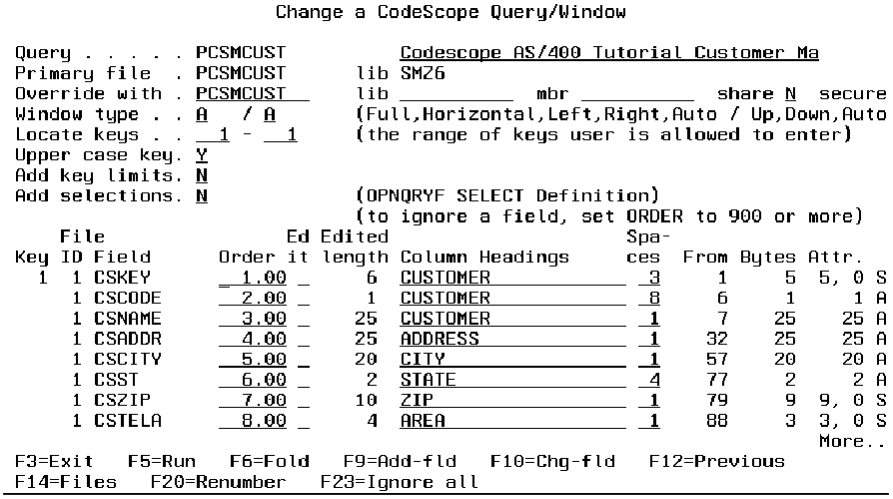
The name of the file displayed in the window.

F4

list of files within the library.

Library

Press **Enter**. The remainder of the window definition parameters are displayed.



Window Title

Specify the title that will appear on the top bar.

Override

If you wish to display a file by referring to a different file/s definition, use the override feature to specify the override File, library, member, Share, and Secure parameters. For OPNQRYF (Open Query File) files, the SHARE parameter must be **Y**.

Window Type

This two part parameter determines the default window size and position on the screen. After the window is opened, the user can change window size and position by pressing function keys.

Size

F: Full Screen Window

H: Horizontal, half screen windows

L: Quarter size, left side of screen

R: Quarter size, right side of screen

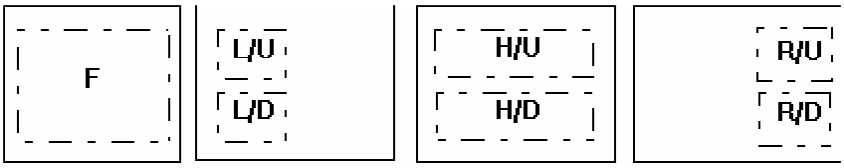
A: Quarter size, automatic placement. The location on screen is determined by the position of the cursor at execution time. The window is displayed on the side of the screen opposite to the cursor. If the cursor is on the left side of the screen, the window displays on the right. If the cursor is on the right side of the screen, the window displays on the left.

Position

U: Up window displays at upper part of the screen.

D: Down widow displays on the lower part of the screen.

A: Automatic positioning. The horizontal positioning of the window is determined by the position of the cursor at execution time. The window is displayed on the part of the screen opposite to the cursor. If the cursor is on the upper half of the screen, the window displays on the lower half of the screen. If the cursor is on the lower half of the screen, the window displays on the upper half of the screen.



Fullscreen	Quarter Left Up	Half Screen Up	Quarter Right Up
	Left Down	Screen Down	Right Down

Locate keys

Specify which key fields can be accessed by the user for the purpose of positioning within the file. Up to eight key fields can be specified for positioning.

Example 1: For a Customer Order File keyed by Customer Number, Order Number and Line Number, the default range for the "Locate keys" parameter is 1 - 3. If the average customer order contains only a few line items, it would be advisable to omit the Line Number from the displayed key by changing the range to 1 - 2. This would make it easier for the user to reposition the file query by only keying two key field values, rather than three.

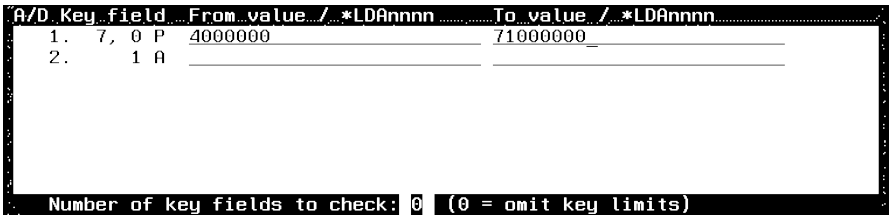
Example 2: In the example above, if you want to restrict the window to one particular customer, change the range to 2 - 3 and use the "Add Key Limits" option (the next parameter) to specify the Customer Number range limit.

Add Key Limits / Change Key Limits

N: No key limits apply to this query window.

Y: This lets you limit the window query display to a range of key values.

If this parameter is Y, then a window is displayed at the top of the screen which allows you to enter the key value range limits and the number of key fields to be checked.



Only records within the range that you specify are displayed in the window. Example: You wish to build a window that only shows records from the Product File with a key range of 4000000 to 7100000. By specifying those key limits, you will restrict the user from viewing any other records in the file.

Add Selections

Selecting '**Y**' will bring up the criteria screen, to allow you to display only those records in your window.

Field Names

The data base field names are displayed at the lower half of the screen. Use the Roll keys to page through the list of fields.

If you accept the defaults, all data base fields will be available for display in the query window, as they appear in the external definition file. CodeScope supports a maximum of 25 fields per query window. You can modify the defaults by specifying:

- Which fields to display in the window
- The order in which the fields will be displayed
- The number of spaces between fields
- The field column headings.

You can also add internal fields not contained in the file's definition and you can modify the definition of any field - whether internal or external.

Order

The order in which the fields are displayed in the window can be specified by modifying the default "Order" value. To prevent a field from being displayed, enter an "Order" value greater than or equal to **900**.

Spaces

This value represents the number of spaces between the end of this field and the beginning of the next. CodeScope begins with a default value that allows the entire column heading to be displayed with a delimiter symbol, " ", between each field. To shorten or lengthen the displayed field, modify the "Spaces" value.

A "Spaces" value of **0** causes the delimiter symbol to be eliminated, allowing you to connect separate fields together under one heading.

Column Headings

Column headings in the query window are displayed on two lines. Pressing the fold/unfold function key will let you modify both lines of the headings. In the "unfold" mode, the third line under "Column Headings" is available for comments.

What are Internal Fields?

CodeScope lets you define additional fields that can be displayed in a query window file. Internal fields can be defined for files without external definitions (DDS), as well as for files with external definitions. This is especially useful in allowing you to build windows to query files without external definitions. For files containing large fields, you can split the large fields into smaller ones to ease viewing in a query window. Creating new fields does not affect the external definition file.

Creating an Internal Field

From the Define Window Screen, press **F9** to define a new internal field. CodeScope opens a window at the top of the screen allowing you to define the field. You are prompted for field name, type, starting position, length, decimals and column headings. While defining a new internal field, you can also use **Page Up** and **Page Down** keys to review the currently defined fields.

Valid field types are:

A: alphanumeric

B: binary

P: packed

S: signed

The new field is added at the beginning of the query window's fields.

Changing Field Definitions

CodeScope allows you to change the field definition of both externally and internally defined fields. These changes do not affect the file's external definition.

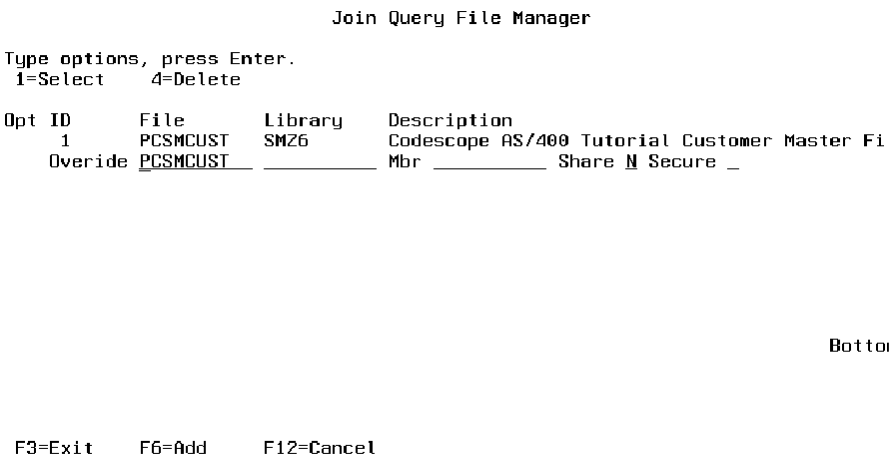
To change the definition of a field, press **F10** from the **Define Window** screen. CodeScope prompts you for the name of the field to be changed and responds with a display of the current definition, allowing you to change it.

To remove an internal field definition from the query window, change the "Order" number to a value greater than or equal to **900.00**

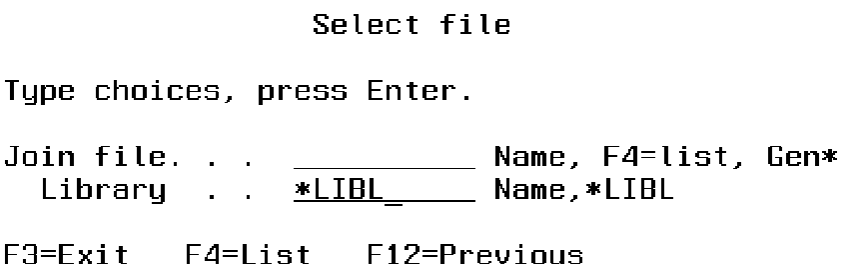
Press the **Exit** function key to exit the window definition process.

Join Query File Manager

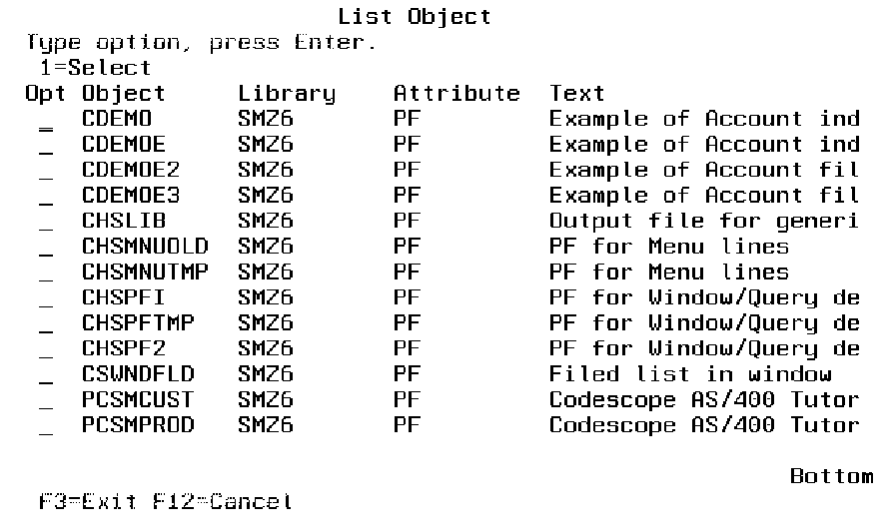
F14 (Files) from the **CodeScope Query/Window** screen guides you to a list of files.



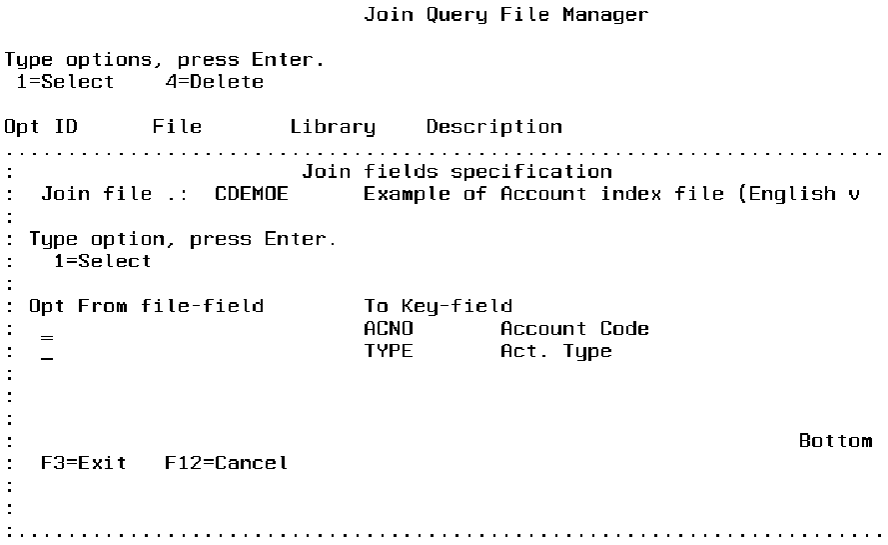
A File can be changed, deleted or added from this screen. A file is added by using the **F6** Command.



A "**Select file**" window will appear. The user then enters the file name. **F4** can be used to display a list of files.



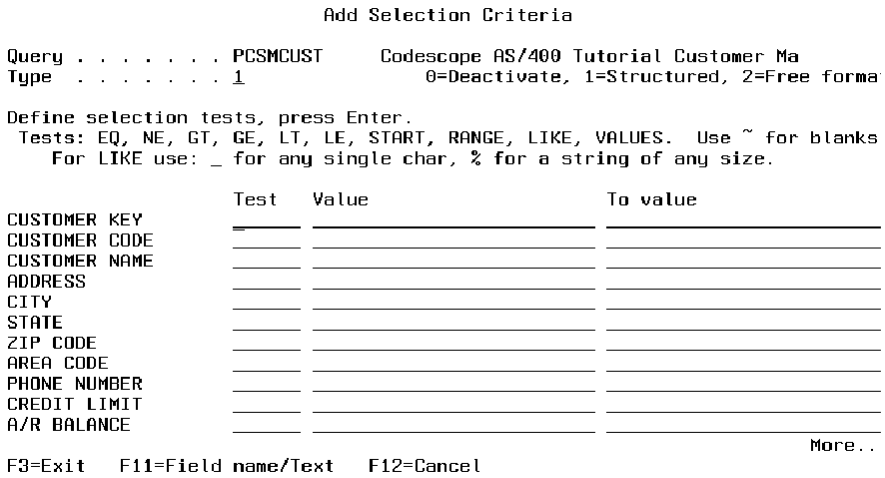
Once a file has been defined the "**Join Fields**" screen will appear with a list of key fields for this file.



At least one of the fields must be selected, a window will appear with a list of suitable fields. One of these fields must be selected to join the two files together. Once the fields have been selected, press **Enter** to return to the list of files. **F3** will end the process of the addition with a choice of "yes" or "no" for accepting the file selection, Returning to the **CodeScope Query/Window** screen which then shows all fields from all the files defined.

Display Selected Records to Include

The Add Selection Criteria Screen is shown if the 'Add selections' is set to 'Y' from the Change a CodeScope Query/Window screen.



Through the feature of CodeScope, a working set of records can be defined, eliminating those records that physically exist but are not necessary for the subsequent operations. The "Display Selected Records" is implemented through a high level interface to the OPNQRYF (Open Query File) command.

Records are dynamically selected based upon the specified tests.

Internal fields which do not cross the field boundaries i.e. start and finish within one external field) can be used in selection tests for a record selection.

No significant delay is encountered, unless the specified selection defines a small number of records in a large file.

Types of Selections

There are 2 types of selection interfaces available in CodeScope.

Structured Style

Structured Style provides a list of field names or descriptions in the user interface along with selection tests. When multiple selection tests are used the relationship between them is an AND relationship. It is only possible to use each field for one test.

Free Format

Free Format enables the entry of the query selection string to be used for the *OPNQRYF*. This is very useful for more complex selection criteria. The selection criteria can be created initially in Structured Style to obtain the correct syntax and then enhanced in free style.

Changing Style

It is possible to change between styles while a SUBSET is being defined. This may result in the loss of some selection criteria when moving to a style (such as Structured Style) which does not support all features, which have been used.

Test Types

LIKE	The field contains, starts or ends with a character or a string of characters. Wild cards are supported for single character and string of any length. A detailed explanation follows
START	Field starts with characters
RANGE	Limit from... -to ...
EQ	Equal to
NE	Not equal to
GT	Greater than
GE	Greater than or equal to
LE	Less than
VALUES	Field equals one of the values supplied (only one value is required)

LIKE Test

The LIKE test is similar to the one used in IBM QUERY. This test helps to find strings in the character files. The "%" sign replaces a string of characters, of any length. The sign "_" replaces any single character. The special signs used can be changed - see Product Defaults.

If we are looking for the name "DAVID" which appears somewhere in the name field, we will use the find option as follows: NAME LIKE %DAVID% The result - the Displayed Selection will contain all the records containing "DAVID" regardless of where this appears in the field.

Upper and Lower Case Differences

Upper and lower case differences are ignored (default). Optionally the differences in pronunciation of the same letter can also be ignored. See Product Defaults.

Building Menu Windows

Menu windows are built interactively by filling in the menu definition fields. The window menu definition becomes effective immediately - no compilation is required.

To build, update or delete a window menu, enter: ***DFNWND *MENU*** on the command line.

The following screen is displayed:

```
                          Define a CodeScope Menu

Library. . . . : SMZ6DTA
Subset menu  . : *ALL                          Position to . . . __________

Type options, press Enter.
 1=Select   3=Copy    4=Delete    7=Rename

    Menu
Opt Name       Title
 _  CSTUT1       ** Tutorial Menu **
 _  CSTUT4     *** Tutorial Menu Windwo
 _  EITAN      Eitan's Menu
 _  JOBS         ** Jobs **
 _  MSGS         ** Messages **
 _  OUTPUT       ** Output **
 _  QCL          ** Main Menu **
 _  QCLSAV       ** Main Menu **
 _  QCMD         ** Main Menu **
 _  QCMDSAV      ** Main Menu **
                                                                      More..
F3=Exit   F5=Refresh   F6=Add   F12=Cancel   F21=Command line
```

- **1**: Update an existing window menu definition.
- **4:** Delete a window menu definition.
- ***7:*** Rename a window menu definition.
- ***3:*** Copy a window menu definition from an existing menu definition.
- **Option F6** : Add a new window menu.

```
                              Add CodeScope Menu

 Type choice, press ENTER

Menu name . . . . _________




F3=Exit   F12=Cancel
```

Menu Name

The menu window name. From the list of definitions you can select one to work with. If this menu is opened as an automatic menu, the menu name must be identical to the program with which it is associated. For more information on automatic menus, see "Opening Windows" on page 58.

Copy/Rename from

The name of the existing menu window that will be used for a copy or rename action.

Press **Enter**. The remainder of the window definition parameters are displayed.

```
                        Change a CodeScope menu

Menu name  . . . CSTUT1

Menu title . . .  ** Tutorial Menu **
Menu size. . . . L / U        (Full,Horizontal,Left,Right / Up,Down)

       S=Sub-title                          Override
       C=Command                            next      Command,
       Q=Query                              window         Menu name,
Order  M=Menu        -- T e x t --          size              Query name
 1.00  Q   Customer Master Inquiry           _ _    CUSTMAST
 2.00  Q   Product Master Inquiry            _ _    PRODMAST
 3.00  S                                     _ _
 4.00  C   Display Active Jobs               _ _    WRKACTJOB
 5.00  C   Display Output Queues            _ _    WRKOUTQ
 6.00  C   Display Messages                 _ _    DSPMSG
 7.00  C   Display Library List             _ _    DSPLIBL
 8.00  S                                     _ _
 9.00  M   System Operations Menu           _ _    SYSOP
10.00  S                                     _ _

F3=exit                          F6=fold/unfold                 F12=previous
```

Menu Title

The title you specify will appear in the top bar of the generated menu window.

Window Type

This two part parameter determines the default window size and position on the screen. See "Building Query Windows" on page 28 for details on the **Window Type** field.

Menu detail lines

Specify options that the user can select. Menu options can:

- Open a query window
- Open another menu window
- Execute any system command
- Call a program

For each line, specify the line position, line type, user text to be displayed and the window menu to be activated or the command to be executed:

Order

The position of this line in relation to other lines in the menu can be reordered.

S Subtitle

C Command

Q Query window

M Menu window

Text

The text that will be displayed on the menu line.

Window Type

This two part parameter determines the override window size and position on the screen. This parameter applies to Line Types Q and M only.

Command

Any valid system command, including the *CALL* command to activate other programs. Press the "Fold/Unfold" function key to extend this field.

Query Name

The name of the CodeScope query window that will be activated when this line is selected.

Menu Name

The name of the CodeScope menu window that will be activated when this line is selected.

To remove a menu line, simply blank out the fields on the line. Press the exit function key to exit the menu definition process.

To remove a menu line, simply blank out the fields on the line.

Press the exit function key to exit the menu definition process.

Automatic Insertion of Windows into RPG Programs (Source Level)

All the changes that are needed in display files and RPG source programs are now done automatically.

The entire process

The only user intervention needed is to define where windows should be enabled in the display file. This is done using SDA and includes the following steps:

- Using SDA : define where windows are enabled.
- Using PDM over the display file source:-activate option **AW** (*ADDCSWND*) to compile the display file
- Using PDM over the RPG source: activate option **AW** (*ADDCSWND*) to: compile the RPG program

PDM options for ease of use

In order to speed up the process for large quantities of members, the new commands can also be run via codes entered next to items in the PDM list screens.

Options **AW** for *ADDCSWND*, **DW** for *DFNCSWND* and **RW** for *RMVCSWND* are added during the installation to the PDM.

Defining where windows are enabled

Follow this procedure to enable a window for a field: While in SDA, type "*****" in front of each field for which a window is needed and press Enter.

In the "**TEXT keyword**" field at the bottom of the screen, enter "**$$CSW window field**" , where window is a CodeScope window name,and field is the window field that contains the value.

(On early versions of V2R2 there was a bug in the SDA that made it ignore TEXT entries for referenced fields. PTFs were issued to solve this problem. In such cases, install the latest PTF cumulative package).

Adding necessary changes to the display file source automatically

Using PDM over the display file source, activate option **AW**. This runs the *ADDCSWND* command for the display file.

The command adds:

- *CF04* at the file level.
- The parameters *RTNCSRLOC* and *CSRLOC* for each record format.
- The hidden fields needed for these parameters.

All statements that were previously added on a prior run of this command will be removed before insertion of the new statements.

Adding necessary changes to the RPG source automatically

Using PDM over the RPG source activate option **AW**. This runs the *ADDCSWND* command for the RPG source.

(Ensure successful compilation of the display file beforehand)

Based upon the definitions made earlier in the display file, the following statements are automatically inserted in the RPG source:

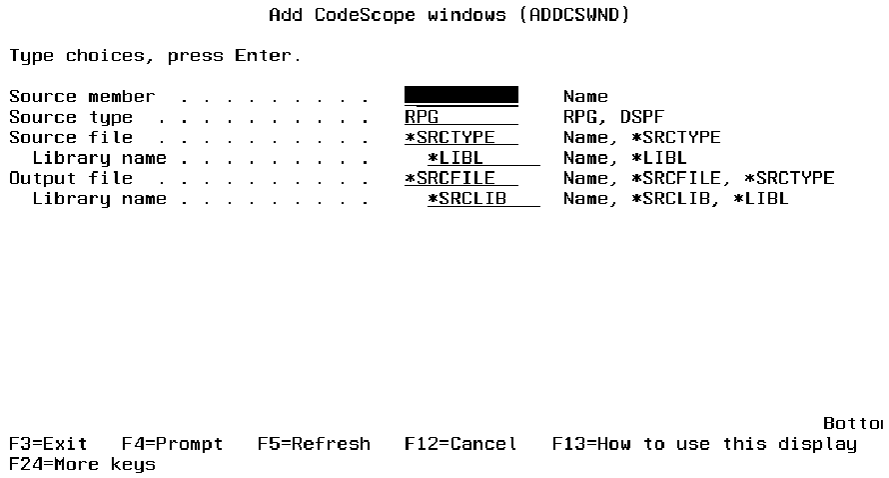
- Statements to identify a user request for a window (pressing **F4** in the field)
- Statements to activate the window
- Statements to fill the display file field with the selected code
- Statements to display the modified record format again

The insertions are for all *EXFMT*, *READ* and *WRITE* verbs which are used to display regular formats, window formats and subfile control formats which contain fields connected to Codescope windows

Programmer intervention is needed to handle special cases such as a record format which is written to the screen in one subroutine and read from it in another.

Note: The additions change the program sequence, especially as each record format has to be re-displayed after it receives a value from a window. It is the programmer's responsibility to ensure that the changes made automatically do not interfere with the operation of the program.

ADDCSWND - Add CodeScope window



The ***ADDCSWND*** (Add CodeScope Windows) command adds the necessary modifications to RPG and DSPF sources.

Source member (SRCMBR)

The name of the source member.

Source Type (SRCTYPE)

The type of the source member.

***RPG**: this is RPG source.

***DSPF**: this is a display file source.

Source File (SRCFILE)

The name of the file from which the source is taken.

***SRCTYPE**: the name is according to the **SRCTYPE**:

QRPGSRC for RPG members

QDDSSRC for DSPF members.

The possible library values are:

***LIBL**: search the library list to locate the file.

Library-name: the library where the file is located.

Output File (OUTFILE)

The name of the file to receive the modified source. The member name remains the same. If an identical member already appears, it is cleared before new data is stored in it.

The possible file values are:

***SRCFILE**: the name of the source file.

***SRCTYPE**: the name is according to the SRCTYPE:

QRPGSRC for RPG members and
QDDSSRC for DSPF members.

File-name: file name.

The possible library values are:

***SRCLIB**: the name of the source file library.

***LIBL**: search the library list to locate the file.

Library-name: the library where the file is located.

DSPF

The possible file values are:

*RPGNAME:

File-name: file name.

The possible library values are:

***LIBL**: search the library list to locate the file.

Library-name : the library where the file is located.

DFNLIB

The name of the library to store the definition. Proper definition files must exist in such a library.

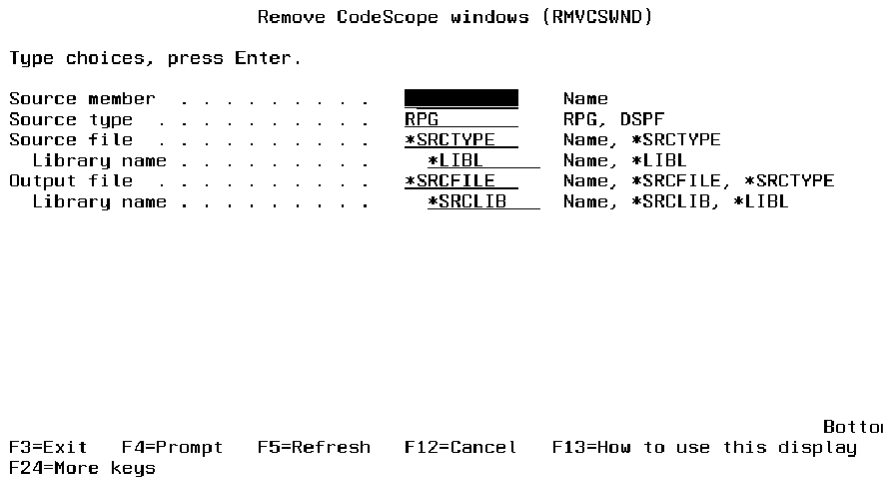
***LIBL**: use the job library list. If no definition files appear, the default library- SMZ6 is used.

library name : library name.

Removing previously added statements

You can use the command ***RMVCSWND*** (Remove CodeScope Windows) to remove the additions made via the ***ADDCSWND*** command.

As program sequences change after running this command, it is the programmer's responsibility to ensure proper operation of the program.



The ***RMVCSWND*** (Remove CodeScope Windows) command removes the modifications previously added by the ADDCSWND command.

Source Member (SRCMBR)

The name of the source member.

Source Type (*SRCTYPE)

The type of the source member.

***RPG**: this is RPG source.

***DSPF**: this is a display file source.

Source File (SRCFILE)

The name of the file from which the source is taken.

***SRCTYPE:** the name is according to the SRCTYPE:

QRPGSRC for RPG members

QDDSSRC for DSPF members.

The possible library values are:

***LIBL**: search the library list to locate the file.

Library-name: the library where the file is located.

Output File (OUTFILE)

The name of the file to receive the modified source. The member name remains the same. If an identical member already appears, it is cleared before new data is stored in it.

The possible file values are:

***SRCFILE**: the name of the source file.

***SRCTYPE**: the name is according to the SRCTYPE: **QRPGSRC** for RPG members and **QDDSSRC** for DSPF members.

File-name: file name.

The possible library values are:

***SRCLIB**: the name of the source file library.

***LIBL**: search the library list to locate the file.

Library-name: the library where the file is located.

Opening Windows

Once you have built your query windows (see "Building Query Windows" on page 28) and/or menu windows (see "Building Menu Windows" on page 43), you can then link the windows to your application programs.

The various methods for opening windows are:

- Call a query window directly from an application program. This involves minor modifications to the application source program.
- Call a menu window directly from an application program. This also involves source modifications similar to those used in calling query windows. From the displayed menu the user can select menu options to open query windows.
- Let the user activate an "automatic menu" by pressing the **Attention** key. This does not involve any modifications to the application source program. From the displayed menu the user can select menu options to open query windows.
- A window can also be opened by issuing a call in a CL program or by issuing a call at the command line.

To simplify programming, we have provided examples of source code needed to interface with CodeScope. You can include this code into your programs and make the necessary modifications. The examples can be found in library SMZ6, file SOURCE.

Source members CSTUT1 and CSTUT2 contain examples of RPG program coding, and source member RPG contains "skeleton" code that can easily be incorporated and modified into your application programs. Source member COBOL contains "skeleton" code that can be incorporated and modified into your application programs. You can also call CodeScope from a PL/1 program.

Calling Query Windows

To call a query window from an application program, the following source code must be added to the application program:

- Detect the user's request for opening a window
- Set up parameters for passing data to and from the window
- Code the CALL statement with associated parameters
- Test for the record returned from the query window.

Set Up Data Parameters

Parameter No	Field Name	Description
1	window name	the name of the window as specified in it
2	place	Where to position the window. In RPG this must be the INFDS if automatic positioning is required. in COBOL, this is the I/O FEEDBACK AREA, if automatic positioning is used. To override the positioning specified in the DFNWND command, enter an asterisk ("*") in position 1, window size in position 2 and window position in positions 3-5. (For valid size and position values, see Chapter 4, Building Q Windows: Window Definition Fields). An additional window position value which can be entered here is "Lnn", where nn is the line number from which the window starts, eg. *RL07 will display a right window from line 7.
3	field name	the name of the field where the cursor was placed when the user selected the record.
	return code	if greater than or equal to zero, the user selected a record from the query window.
	return key	the full key of the selected record (first 99 bytes).
	return rec	record the full selected record (first =-1000 bytes)
4	stype	contant value = "S"
	start key	the key of the record from which the query display should begin

5	rtype from key	constant value = "R"
	from key	The lowest value key that the user will be allowed to access. This overrides the key range specified in the DFNWND command
	to key	The highest value key that the user will be allowed to access. This overrides the key range specified in the DFNWND command

Parameters can be omitted from the end of the list when they are not used. It is common to call CodeScope with 3 parameters only. If a parameter is passed it must be passed, in the correct length, otherwise unpredictable results may occur.

RPG

Parameter passing to and from CodeScope is done by defining the appropriate data structures.

If you wish to have CodeScope automatically position windows based on cursor location, you must code an INFDS (Information Data Structure) so that CodeScope can determine the cursor location. The coding consists of a workstation continuation statement and a data structure statement:

The following data structures define the values passed back from the call to CodeScope, the starting value displayed in the query window and the range of record keys the user may browse:

```
FCDEMOFMECF  E                    WORKSTN
F                                          KINFDS $INFDS
 *
E                    $REC        4250
 *
I$INFDS      DS                            520
I$OUTVL      DS
I                                       31  40 $FLD
I                                       41  550$OK
I                                       56 154 $KEY
I                                      1551154 $REC
I$START      DS
I                                        1   1 $S
I                                        2 100 $SKEY
I$RANGE      DS
I                                        1   1 $R
I                                        2 100 $FRKEY
I                                      101 199 $TOKEY
I$DSKEY      DS                              99
I                                    P   1   40$ACNO
```

COBOL

If you wish to have CodeScope automatically position windows based on cursor location, you must set up an I-O Feedback link for the display file:

```
0044.00       ****************************************************************
0045.00        CONFIGURATION SECTION.
0046.00        SPECIAL-NAMES.   I-O-FEEDBACK IS I-O-FB.
0047.00       ****************************************************************
0048.00        01  I-O-FB-AREA.
0049.00            03 FILLER                PIC X(147).
0050.00            03 I-O-CURSER.
0051.00               05 I-O-ROWCOL         PIC 9(2) COMP-4.
0052.00            03 FILLER                PIC X(51).
0053.00        01  I-O-ROW                  PIC 9(3).
0054.00        01  I-O-COL                  PIC 9(3).
```

Data fields used for passing parameters to the CodeScope query window should be set up in working storage:

```
0056.00        01  WINDOW-NAME              PIC X(10).
0057.00        01  VALUE-RETURNED.
0058.00            03 FILLER                PIC X(30).
0059.00            03 FIELD-NAME            PIC X(10).
0060.00            03 FILLER                PIC X(12).
0061.00            03 RETURN-CODE           PIC S9(3).
0062.00            03 RETURNED-KEY          PIC X(99).
0063.00            03 RETURNED-REC          PIC X(1000).
0064.00        01  START-PRM.
```

Detect User Request

The method for detecting a user request for a query window depends on the user interface design. The user interface method you use is independent of CodeScope.

Common interface techniques are:

- Function Key: A predetermined function key (typically F4) can be used to indicate that a query is desired. If more than one window can be called from the application screen, then you have to test the cursor location to know which window is being requested. Alternatively, each query window request can be assigned to a particular function key.
- Blank Field: When the user leaves a field blank it indicates that a query window is desired. If more than one window can be called from the application screen, then you have to test the cursor location to know which window is being requested.
- Special Field Value: A predetermined field value can be used for indicating a query window request. This is typically a question mark character ("?"). In numeric fields the user could use a negative one ("-1").

Call CodeScope

Once the program detects a user query window request, a call is used to CodeScope. The source code necessary for the call is as follows.

RPG

```
C    CALL 'C#R'
C    PARM 'windownm'WINDOW 10
C    PARM      PLACE
C    PARM      VALUE
C    PARM      START
C    PARM      RANGE
```

COBOL

```
C CALL 'C#R' USING WINDOW PLACE
C PARM 'windownm' WINDOW 10
C START RANGE
```

The optional parameters - start and range - may be omitted when the are needed. In COBOL if you require automatic positioning, use I-O-FB-AREA instead of PLACE parameter.

Returned Record

After the call to CodeScope, a non-negative value in there "return code" field indicates that the user selected a record. the program would typically move the selected record to a data-structure describing the record fields.

```
C     RETCDE IFGE 0
C     MOVEARETREC REC
C     END
```

The definition of the returned record can be contained in an externally described data structure (DS), so that the defined fields can be easily accessed by the program.

```
IF RETURN-CODE>=ZERO
   MOVE RETURNED-REC TO REC
```

Calling Menu Windows

To call a menu window from an application program, the following source code must be added to the application program.

- Detect the user's request for opening a window
- Set up parameters for passing data to the window
- Code the CALL statement with associated parameters

Setting Up Data Parameters

Required Parameters:

- **Window**: constant value = "*MENU"
- **place**: where to position the window
- **menu name**: the name of the menu as specified in the DFNWND command

RPG

If you wish to have CodeScope automatically position the menu window based on cursor location, you must code an INFDS (Information Data Structure) so that CodeScope can determine the cursor location. The coding consists of a workstation continuation statement and a data structure statement.

```
FDSPFILE CF E         WORKSTN
F                     KINFDS PLACE
IPLACE      DS                  520
```

COBOL

If you wish to have CodeScope automatically position windows based on cursor location, you must set up an I-O feedback link for the display file.

```
CONFIGURATION SECTION
SPECIAL-NAMES. I-O FEEDBACK IS I-O-FB

01 I-0-FB AREA     PIC X(200)

ACCEPT I-0-FB-AREA FROM
  I-0-FB FOR DISPLAY FILE
```

Data fields used for passing parameters to the CodeScope menu window should be set up in working storage.

```
01 WINDOW      PIC X(10).
01 PLACE       PIC X(10).
01 MENU-NAME   PIC X(10).
```

Detect User Request

The method for detecting a user request for a menu window depends on the user interface design. The user interface method you use is independent of CodeScope.

The most common user interface technique is by means of a predetermined function key that can be used to indicate that a menu is desired. If more than one menu window can be called from the application screen, then you have to test the cursor location or a predetermined field value to know which window is being requested. Alternatively, each menu window request can be assigned to a particular function key.

Call CodeScope

Once the program detects a user menu window request, a call is issued to CodeScope. The source code necessary for the call is as follows:

RPG

```
...10....+...20....+...30....+...40....+...50....+
C                   CALL 'C#R'
C                   PARM '*MENU' WINDOW 10
C                   PARM         PLACE
C                   PARM 'menuname' MENU 10
```

COBOL

```
...10....+...20....+...30....+...40....+...50....+
  C                   CALL 'C#R' USING WINDOW PLACE
MENU.
```

Activating Automatic Menus

In the "automatic menu" method, the menu generator is activated by pressing the Attention key. CodeScope activates the menu automatically without any modifications to the application program.

CodeScope activates a menu with the same name as the currently active application program. If no such menu exists, CodeScope attempts to display the menu defined for the program that preceded the current one. If not successful, CodeScope keeps searching the invocation stack until it finds a match.

To give a user access to automatic menus, the user environment should be initialized for CodeScope. See Appendix B, Technical Matters: Initializing Automatic Menu Environment, to see how this is done.

Examples

1. Open a query name ITEMNO as a full screen window from within a CL program:

```
CALL 'C#R' (ITEMNO *FA)
```

2. Open the ITEMNO window from a RPG program, using default positioning:

```
C CALL 'C#R'
C PARM 'ITEMNO' WINDOW
C PARM INFDS
C PARM VALUE
```

3. Open the ITEMNO window from a COBOL program, overriding the positioning to a horizontal window starting on line 6:

```
MOVE "ITEMNO" TO WINDOW-NAME.
MOVE "*HL06" TO PLACE.
CALL "C#R" USING WINDOW-NAME PLACE VALUE.
```

4. Open the ITEMNO window from a RPG program, specifying a start key and key range:

```
C* Define START key value
C MOVE 'S' STYPE
C MOVE *LOVAL SKEY
C MOVEL 'A5000' SKEY
C* Define RANGE key values
C MOVE 'R' RTYPE
C MOVE *LOVAL FRMKEY
C MOVEL 'A0000' FRMKEY
C MOVE *HIVAL TOKEY
C MOVEL 'A9999'
C* Open the window
C CALL 'C#R'
C PARM 'ITEMNO' WINDOW
C PARM INFDS
C PARM VALUE
C PARM START
C PARM RANGE
```

5. Open a menu window named ORDMENU from an RPG program:

```
C CALL 'C#R'
C PARM '*MENU' WINDOW
C PARM INFDS
C PARM 'ORDMENU' MENUNM
```

For full examples of source programs with CodeScope window calls, see the source members in library SMZ6, file SOURCE.

Appendix: Technical Matters

Supported Environments

Computer types

CodeScope is supported on the IBM i.

Files

CodeScope query windows can be defined on indexed files only. The file can only contain one record format and the record length cannot exceed 4096 characters. Files do not require an external definition to be used by CodeScope, although externally described files make building query windows much easier. When building a new query window, CodeScope will automatically retrieve the first 200 fields of an externally described file. The maximum number of fields that can be defined within a query window is 25.

Record Search Limit

When the user enters a string to be searched across all alphanumeric window fields (by entering the string in the CodeScope window "Search" field), the system only scans the first 9999 characters of the fields defined for the window.

Security Notes

In order to use CodeScope, users need authority to the commands *CRTDTAARA* and *CRTMSGQ*. In order to grant authority for these commands to all users, the security officer can issue the following commands:

```
GRTOBJAUT CRTDTAARA *CMD *PUBLIC
GRTOBJAUT CRTMSGQ *CMD *PUBLIC
```

Performance Tips

CodeScope requires that certain objects must exist in library QTEMP. We recommend that you prepare a work environment at the beginning of the users work session. To see how to prepare the user environment, read Initializing Automatic Menu Environment later in this chapter. If you do not initialize the environment, CodeScope will automatically generate the

necessary objects during the first CodeScope operation, consequently slowing down the response time on the first operation.

For CodeScope pop-up windows to display correctly, the screen files of the programs that will use CodeScope windows should have the RSTDSP(*YES) option. The following command can be used to ensure that all screens include this option:

```
CHGDSPF FILE(libname/*ALL) RSTDSP(*YES)
```

This command may be issued for any user library. Do not perform the command on IBM system libraries, nor on the CodeScope system library - SMZ6.

CodeScope Global Defaults affect system performance. For best performance, the "New Window" parameter should be set to L, the "Scan Scope" should be between 500-1000, and the Generator remains ACTIVE in Y. See Global Defaults later in this chapter for more details.

Initializing Automatic Menu Environment

The following commands must be added to the beginning of the user profile initial program or to the beginning of the application driver program:

```
SETWND
SETATNPGM SMZ6/C#R
```

Alternatively, you can change the user profile, using command CHGUSRPRF, to set the attention key program to SMZ6/C#R.

When the user presses the attention key, Codescope displays the menu window whose name is identical to the currently executing program. If Codescope cannot find a menu matching the current program name, or any other program name in the user's invocation stack, then the QCMD menu will be displayed (if it exists).

CodeScope Files and Libraries

CodeScope programs are located in library SMZ6. CodeScope files are located in SMZ6DTA.

Neither SMZ6 nor SMZ6DTA need to be included in the user's library list.

CodeScope window definition files can be created in user libraries so that separate query and/or menu windows can be defined for different applications. In order to replicate the query window definition file, copy the file CHSPF in library SMZ6DTA to a user library. For menu definitions, copy file ***CHSMNU*** in library SMZ6DTA.

Files CHSPF and CHSMNU should be backed up after the ***DFNWND*** command has been used to modify window definitions.

Global Defaults

By using the *DFTS paramenter of the ***DFNWND*** command, you can modify CodeScope parameters for the entire system or for a particular user library:

Window Frame

You can specify the characters to be used to fill the horizontal and vertical bars of window and help frames. They are normally blank. You can also

specify whether frame characters should display in reverse image, high intensity and/or with column separators. The remaining parameter lets you specify whether the frame is displayed for all windows, query windows only, or none.

To achieve a window frame similar to IBM standard, use Horizontal fill character=. , Vertical fill character= : and Reverse Image= N.

New Window

Specify whether displaying a new window causes a currently displayed window to disappear or to remain on the screen. The options are that the new window replaces the old one (Y), does not replace the old one (N), or only replaces the old one if the user is on a local terminal (L). Since the replace function requires that extra data be sent to the terminal, response time on remote terminals will be improved if replace is not used.

Scan Scope

This parameter is used to prevent a search from running too long, thereby slowing down the system. The "Scan Scope" is the number of records that CodeScope will search before issuing a not-found message. After the message is issued, the user can continue the search by pressing the search function key again.

Window Active

The window generator program displays all queries and menus. You can specify that it remain active in memory in order to improve performance.

Command Syntax

You can specify whether menu commands will be entered in AS/400 or S/38 format.

Authority Code

The Authorization Code can optionally be changed by pressing the appropriate function key. See Appendix A, Installation: Entering Authorization Codes for an explanation of authorization code

Using CodeScope with OPNQRYF Files

CodeScope can display files opened through the OPNQRYF command. The OPNQRYF command parameters must be coded as follows:

```
OPNQRYF KEYFILE(*FILE) OPTIMIZE(*FIRST 10) SEQONLY(*NO)
```

In addition, the CodeScope window must be defined with "Share" = Y