


[print](#) | [close](#)

APIs by Example: User Interface Manager APIs

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 05/18/2006 (All day)

The UIM APIs and List Panel Groups deliver robust, consistent, and highly functional applications and utilities with a minimum of programming effort. Much of the complexity and screen dialog handling is performed and controlled by the User Interface Manager (UIM), so you can concentrate on the core functionality of your program.

To demonstrate how you can use the UIM APIs and List Panel groups, I wrote the Work with Command Log (WRKCMDLOG) command. This command uses the Retrieve Request Message (QMHRTVRQ) API's ability to retrieve previously executed commands from your job log.

iSeries users are familiar with the F9=Retrieve facility from menus and command lines. The WRKCMDLOG utility is basically the same thing, except that it also displays a scrollable list of previously run commands and subsequently lets you prompt and rerun the commands. You can further optionally specify a search argument if you want to display only a subset of the commands in your job log.

From the operating system's perspective, a command is a message of type request (*RQS), and a variety of message handling APIs can be used to retrieve that request message from a job's call message queue or external message queue. I chose the QMHRTVRQ API because it is designed specifically to retrieve request messages, which are the only messages needed for this utility.

At the end of this article, I provide a link to a discussion of job message queues and related concepts. An important aspect of this topic that relates to the utility here is that a job's logging level needs to be set appropriately to be able to retrieve request messages from the job message queue. (This is true of the F9=Retrieve function as well.) Running the following command takes care of this requirement:

```
CHGJOB LOG(4 00 *N)
```

Back to the WRKCMDLOG command. Here's the command's prompt display:

```

                                Work with Command Log (WRKCMDLOG)
Type choices, press Enter.
Include string . . . . . *ALL
Commands to display first . . . *LAST          *LAST, *FIRST
```

For example, if you run the following command:

```
WRKCMDLOG INCLUDE(*ALL) START(*LAST)
```

It brings up the following display:

```

                                Work with Command Log                                WYNDHAMW
                                                                                   12-05-06  18:46:48
Job . . . . . : DSP007A
User . . . . . : GARY          Include . . . .
Type options, press Enter.
  1=Run command
Opt  Command
  _   DSPMSG QSYSOPR
  _   WRKACTJOB
                                                                                   Bottom
Parameters or command

===>

F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F10=Command entry
F12=Cancel  F16=Display job log   F22=Display entire command

```

If you specify option 1 next to one or more commands in the list and press Enter, the selected commands are run again. Or press F4 to have the command(s) prompted first. If a command string is too long to fit within the list's Command column, you can place the cursor on the command string and press F22 to have the full command string displayed in a pop-up window. Both the command and the list panel group come with full cursor-sensitive help support.

In upcoming issues of APIs by Example, I will discuss the anatomy of list panel groups in more detail. Today, I focus on the UIM APIs. To continue this investigation on your own, you need the following two manuals: *User Interface Manager APIs* and *Application Display Programming*. Links to both manuals in PDF format are at the end of this article.

I need to explain three basic UIM concepts to help make sense of today's UIM API example.

Application handle. A UIM application must be initialized with either the Open Display Application (QUIOPNDA) API or the Open Print Application (QUIOPNPA) API. Both APIs return an application handle to use in subsequent calls to other UIM APIs that load and manipulate data as well as activate, manage, or close the panel group's different display or print panels.

The application handle is also the key identifier of the resources allocated for application initiation and execution, and the Close Application (QUICLOA) API uses the handle to properly release these resources when the application ends. An application handle is an 8-byte-long character field.

UIM exit programs. The User Interface Manager can use a number of different exit programs to communicate with the application and have the application perform a variety of tasks as part of or following different panel group actions. These exit programs can be specified for various UIM panel group elements, such as the PANEL (Display panel), KEYI (Key item) and LISTACT (List action) tags.

A general panel group exit program specified on the PANEL tag, for example, identifies a program to be called every time Enter or a function key is pressed. You can also define an exit program to be called every time a list option has been run. For each type of exit program, an individual parameter data structure is defined and documented in the UIM API manual.

In essence, you can use the UIM exit programs to perform extended validation of input data or update a list entry that has been changed — and much more.

Variable pool. Part of the panel group is the variable pool, where all input and output variables are stored. Many of these variables of course contain the data displayed or input through the different display panels in the panel group. There's also another set of variables that help control the panel group behavior and communicate information between panel group and application.

One example of the latter is the variables that contain the date and time information shown in the panel group header. Another example is the variable that contains the name and library of an exit program to call. UIM APIs exist to perform different input and output operations on the variable pool. UIM APIs also use specific variables to return information to the application, as you see next.

Now, on to the sequence of API events in CPP CBX155:

1. The Open Display Application API is called to initialize the panel group session, and it returns an application handle as I just described.
2. Using this application handle, the Put Dialog Variable (QUIPUTV) API initializes in two calls the list header information (e.g., date, time, job name, user profile, etc.) and the list exit program name and library, respectively.
3. The Retrieve Request Message API is called to return, one by one, all request messages in the current job's job log. Oldest first, newest last.
4. Each request message is formatted and subsequently added to the panel group's list records using the Add List Entry (QUIADDLE) API. Another similar API lets you add a block of entries in one call, but for this utility, single-entry adds suffice. Each entry is assigned a list entry handle, which is UIM's unique identification of a specific list entry. Note that the entry handle does not survive a list rebuild, and UIM can reuse it in the event of a list entry deletion.
5. The Set List Attributes (QUISETLA) API sets the display position of the list to either first or last record, as specified on the WRKCMDLOG command's START parameter.
6. Now it is time to actually display the list, and the Display Panel API handles this task. The actions that can be performed from the panel are handled by either the panel itself or the panel exit program. The only exceptions are:
 - a. If F5 is pressed, the return value 5 is returned from the Display Panel API and causes the application to rebuild and reposition the list.
 - b. If a value is entered in the Include field on the panel, the value is retrieved using the Get Dialog Variable (QUIGETV) API, and the list is rebuilt, selecting only command strings that contain the entered string.
 - c. If either F3 or F12 is pressed, the command ends, but first the Close Application API is called to free all allocated resources.

The CPP is compiled with activation group attribute *NEW to let the command be called repeatedly from the WRKCMDLOG command's command line.

A similar account for the List Exit Program shows the following steps:

1. The entry parameter structure is checked, and if it is the correct type (Function Key) and event (F22 pressed) the Get Dialog Variable API is called to retrieve the internal variable Cursor Entry Handle. This variable contains the list entry handle of the list entry that the cursor was positioned on when the function key was pressed, and it is updated with this information by the UIM.

2. Using the Get List Entry API and the list entry handle returned, the actual list entry data is retrieved, including the full command string, which is then displayed using the Display Long Text (QUILNGTX) API.
3. The exit program returns.

As you see momentarily, the actual code required to build the WRKCMDLOG command is pretty limited. Much of the functionality is built into and taken care of by the User Interface Manager, thereby speeding up the development process and improving the end result in terms of both functionality and consistency.

For more UIM API examples, a further discussion of the UIM panel group tag language, and a suite of new work management CL commands, watch for upcoming installments of APIs by Example.

This APIs by Example includes the following sources:

```
CBX155  -- Work with Command Log - CPP
CBX155E -- Work with Command Log - UIM General Exit
CBX155H -- Work with Command Log - Help
CBX155P -- Work with Command Log - Panel Group
CBX155X -- Work with Command Log

CBX155M -- Create Command Objects
```

To create all these objects, compile and run CBX155M. Compilation instructions are in the source headers, as usual.

Discussion of job message queues:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/rbam6/jmsgq.htm>

UIM APIs — printable PDF:

http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis_web/uim.pdf

Application Display Programming (see pages 271–660) — printable PDF:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/books/sc415715.pdf>

[Azubike Oguine's UIM articles are available by clicking this link.](#)

This article demonstrates the following APIs:

Retrieve Request Message (QMHRTVRQ) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/QMHRTVRQ.htm>

Retrieve Message (QMHRTVM) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/QMHRTVM.htm>

Convert Case (QlgConvertCase) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/QLGCNVCS.htm>

Open Display Application (QUIOPNDA) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiopnda.htm>

Close Application (QUICLOA) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quicloa.htm>

Display Panel (QUIDSPP) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quidspp.htm>

Display Long Text (QUILNGTX) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quilngtx.htm>

Put Dialog Variable (QUIPUTV) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiputv.htm>

Get Dialog Variable (QUIGETV) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quigetv.htm>

Add List Entry (QUIADDLE) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiaddle.htm>

Get List Entry (QUIGETLE) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quigetle.htm>

Update List Entry (QUIUPDLE) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiupdle.htm>

Remove List Entry (QUIRMVLE) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quirmvle.htm>

Retrieve List Attributes (QUIRTVLA) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiirtvla.htm>

Set List Attributes (QUISETLA) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quiisetla.htm>

Delete List (QUIDLTL) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/quidltrl.htm>

Send Program Message (QMHSNDPM) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/QMHSNDPM.htm>

Retrieve Object Description (QUSROBJD) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r3/topic/apis/qusrobjd.htm>

You can retrieve the source code for this API example from the following link:

http://www.pentontech.com/IBMContent/Documents/article/52574_69_WrkCmdLog.zip

Source URL: <http://iprodeveloper.com/rpg-programming/apis-example-user-interface-manager-apis>