

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

## APIs by Example: Journal APIs Solving Spooled Files Mysteries

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 09/24/2009 (All day)

The collection of journal and commit APIs includes a number of very useful and powerful APIs. Some of the most prominent candidates for this designation include the Retrieve Journal Entries (QjoRetrieveJournalEntries) API, the Retrieve Journal Information (QjoRetrieveJournalInformation) API and the Retrieve Journal Receiver Information (QjoRtvJrnReceiverInformation) API. Although the API names certainly are long, the code included with today's APIs by Example column is aimed at making the time needed for you to take advantage of these APIs correspondingly short.

To provide a somewhat realistic context for my presentation of the three APIs, I've created the Work with Spooled File Audit (WRKSPLFAUD) command. On systems having the system audit journal configured to track spooling activity (how-to link provided at the end of this article) the WRKSPLFAUD command also provides a quick and practical tool to investigate events related to printer output as well as document all the stages of a spooled file's life cycle. More details to follow in a moment.

The QjoRetrieveJournalEntries, QjoRetrieveJournalInformation, and QjoRtvJrnReceiverInformation APIs offer access to journal, journal receiver, and journal entry information similar to that provided by the Display Journal (DSPJRN), Work with Journal Attributes (WRKJRNA), and Display Journal Receiver Attr (DSPJRNRCVA) CL commands, respectively. Put together, that's an impressive amount of information covering the journaling infrastructure and event and data recording on IBM i.

My example today has its origin in the system audit journal (QAUDJRN) because that seems to be the most common denominator across systems as far as journaling is concerned, but apart from various system uses of the journal facility you can journal database files, data areas and data queues. So using the journal and commit APIs, you could also create monitoring, logging, accounting, and auditing applications and utilities. At the end of this article, I've included a couple of links pointing to articles and code examples demonstrating the former two employments of journal entries and APIs.

The QjoRetrieveJournalEntries API provides the foundation for the core function of the WRKSPLFAUD command because it is capable of selecting and retrieving journal entries based on the same comprehensive set of parameters and selection criteria as the DSPJRN and Receive Journal Entry (RCVJRNE) commands. The basic parameter offering includes the range of journal receivers, date and time interval, number of journal entries, qualified job name, user name, as well as program name and journal entry code and type. This array of criteria allows you to narrow your selection of journal entries quite efficiently, and consequently additional parameters can be supported and enforced by the program calling the QjoRetrieveJournalEntries API without incurring a significant performance penalty.

The complete set of the WRKSPLFAUD command's parameters are documented by the command prompt below:

Work with Spooled File Audit (WRKSPLFAUD)			
Type choices, press Enter.			
Range of journal receivers:			
Starting journal receiver . .	*CURRENT	Name, *CURRENT,	
*CURCHAIN			
Library . . . . .		Name, *LIBL,	
*CURLIB			
Ending journal receiver . . .		Name, *CURRENT	
Library . . . . .		Name, *LIBL,	
*CURLIB			
Starting date and time:			
Starting date . . . . .	*FIRST	Date, *FIRST	
Starting time . . . . .		Time	
Ending date and time:			
Ending date . . . . .	*LAST	Date, *LAST	
Ending time . . . . .		Time	
Number of journal entries . . .	*ALL	Number, *ALL	
Job name . . . . .	*ALL	Name, *ALL	
User . . . . .		Name	
Number . . . . .		000000-999999	
User profile . . . . .	*ALL	Name, *ALL	
Program . . . . .	*ALL	Name, *ALL	
Spooled file . . . . .	*ALL	Name, *ALL	
Spooled file job . . . . .	*ALL	Name, *ALL	
User . . . . .		Name	
Number . . . . .		000000-999999	

```

Spooled file number . . . . . *ALL          1-999999, *ALL

Output queue . . . . . *ALL          Name, *ALL

Library . . . . . Name, *ALL

Audit type . . . . . *ALL          *ALL, *PRTO,
*SPLF, *DIRP...
      + for more values

Spooled file event . . . . . *ALL          *ALL, *READ, *CRT,
*DLT...
      + for more values

Output . . . . . *          *, *PRINT

```

The command and its parameters are documented in great detail in the accompanying online help text panel group, but the *Range of journal receivers* deserves a little explanation here as well. Specifying the special value *\*CURRENT* for this parameter limits the journal entry selection process to the currently attached journal receiver only, whereas the alternate option of *\*CURCHAIN* extends the selection process to the complete online journal receiver directory.

To accommodate the need for a receiver selection somewhere in between, I've written a so-called command parameter *Choice Program* (CP). A CP allows you to tailor both the *Choice text* appearing to the right of the parameter entry line, as well as the function key *F4 Prompt list* displaying the permissible input values if you place the cursor on the parameter entry line and press F4. For each of these events, if you've specified a CP for the parameter in question, the CP is called and asked to return the information needed. There's a CP input parameter data structure defining the command name, the keyword name for which the details are requested as well as a code specifying whether a choice text or a list of permissible values is required. You can then return the desired information in the CP's second output parameter.

So instead of the command operator having to execute the WRKJRNA command to locate the currently online journal receiver directory, then make a note of the journal receiver range of interest based on the receiver's attach date and time, and eventually enter the relevant range on the WRKSPLFAUD command prompt, I use the QjoRetrieveJournalInformation and QjoRtvJrnReceiverInformation APIs in the CP to retrieve all this information and make it immediately available whenever anyone presses the F4 function key for any of the receiver range parameter elements, as in the following example, in which F4 has been pressed for the *Starting journal receiver* parameter:

```

Specify Value for Parameter RCVRNG

Type choice, press Enter.

```

```

Type . . . . . : NAME

Starting journal receiver . . . *CURRENT


AUDRCV4197  15-09-09  02:00:20

AUDRCV4198  15-09-09  06:38:57

AUDRCV4199  16-09-09  02:00:33

AUDRCV4200  16-09-09  06:35:00

AUDRCV4201  17-09-09  02:00:49

AUDRCV4202  17-09-09  06:37:26

AUDRCV4203  18-09-09  03:34:27

AUDRCV4204  18-09-09  06:35:55

AUDRCV4205  19-09-09  02:00:15

AUDRCV4206  19-09-09  06:36:27

AUDRCV4207  20-09-09  02:00:30

AUDRCV4208  20-09-09  06:36:22

```

You have the journal receiver name, attach data, and attach time in the three columns listed for each receiver available, and corresponding information if you use F4 for one of the journal receiver *Library* entry fields. Apart from serving as an example of how to call the two involved journal APIs the CBX208C choice program is specifically written with the aim of establishing a choice program template for anyone to use as a starting point for their own choice programs, should anyone have that requirement.

To show you an example of the WRKSPLFAUD list panel and the information and options available from that panel, I ran the following command on my system:

```
WRKSPLFAUD RCVRNG(*CURRENT) USRPRF(QSECOFR)
```

Said command in turn produced the list panel below:

```

                                     Work with Spooled File Audit

WYNDHAMW                                                                    19-09-09

08:27:04
Type options, press Enter.

```

5=Display journal    7=Work with job    8=Work with audit for spooled file						
-----Entry-----    -----Job-----    User						
Entry    Event						
Opt    Date    Time    Name    User    Number    Profile						
Type    Type						
	19-09-09	06:47:28	ZITSEC	QSECOFR	764780	QSECOFR
*SPLF    *CRT						
	19-09-09	06:47:29	ZITSEC	QSECOFR	764780	QSECOFR
*SPLF    *DLT						
	19-09-09	06:47:30	ZITSEC	QSECOFR	764780	QSECOFR
*SPLF    *CRT						
	19-09-09	06:47:30	ZITSEC	QSECOFR	764780	QSECOFR
*SPLF    *DLT						
8    19-09-09	06:47:30	ZITSEC	QSECOFR	764780	QSECOFR	
*SPLF    *CRT						
Bottom						
F3=Exit	F4=Prompt	F5=Refresh	F10=Command line	F11=View 2		
F12=Cancel						
F17=Top	F18=Bottom					

The list panel offers a total of four different views, each conveying various information about the spooled file audit entries presented and all in all providing a solid overview of the audit event in question. As usual, you use function key F11 to toggle between the views. All list columns are also explained in the online help text panel group available. The list option *5=Display journal* runs the DSPJRN command for the selected audit entry, positioning the DSPJRN list panel to the same entry in context with subsequent spooled file journal entries and allowing you to display more details about each individual journal entry.

Option *7=Work with job* likewise runs the Work with job (WRKJOB) command for the job referenced in the spooled file audit entry header section, returning a message if the job no longer is active on the system. And option *8=Work with audit for spooled file* executes the WRKSPLFAUD command recursively, specifying the journal entry's journal receiver, the spooled file name, qualified job name, and spooled file number as the only parameters. This combination allows you to see all events related to that particular spooled file, irrespective of their origin. For the spooled file audit entry selected in the panel above, option 8 returns the following information:

```

                                Work with Spooled File Audit

WYNDHAMW

                                19-09-09

08:27:42
Type options, press Enter.

    5=Display journal    7=Work with job    8=Work with audit for
spooled file

      -----Entry-----  -----Job-----  User
Entry  Event
Opt   Date    Time      Name      User      Number  Profile
Type  Type
19-09-09 06:47:30  ZITSEC    QSECOFR    764780  QSECOFR
*SPLF  *CRT
19-09-09 06:47:30  CPHP01    QSPLJOB    754862  QSPLJOB
*SPLF  *READ
19-09-09 06:47:31  CPHP01    QSPLJOB    754862  QSPLJOB
*SPLF  *READ
19-09-09 06:47:34  CPHP01    QSPLJOB    754862  QSPLJOB
*PRTO  *DLT
19-09-09 06:47:34  CPHP01    QSPLJOB    754862  QSPLJOB
*SPLF  *DLT

      Bottom
      F3=Exit   F4=Prompt   F5=Refresh   F10=Command line   F11=View 2
      F12=Cancel
      F17=Top    F18=Bottom

```

As the above example demonstrates, option 8 gives you an overview of the full life cycle of the spooled file in question, of course depending on the timeframe being covered by the online journal receiver directory. I hope you'll find the WRKSPLFAUD command helpful in the event that a spooled file mystery should cross your path. And I hope that the code presented today will give you a headstart should a programming assignment involving the journal APIs do so, too.

**This APIs by Example includes the following sources:**

```

CBX208    -- RPGLE    -- Work with Spooled File Audit - CPP
CBX208C   -- RPGLE    -- Work with Spooled File Audit - Choice Program

```

```
CBX208H  -- PNLGRP  -- Work with Spooled File Audit - Help
CBX208P  -- PNLGRP  -- Work with Spooled File Audit - Panel Group
CBX208V  -- RPGLE   -- Work with Spooled File Audit - VCP
CBX208X  -- CMD     -- Work with Spooled File Audit

CBX208M  -- CLP     -- Work with Spooled File Audit - Build command
```

To create all these Work with Spooled File Audit command objects, compile and run the CBX208M program, following the instructions in the source header. As always, the compilation instructions are also included in the respective source headers.

### **Previously published related articles:**

[Retrieve Journal Entry Exit Program](#)

### **IBM Documentation:**

[Using Auditing to Track Spooling Activity](#)

[Setting Up Security Auditing](#)

### **This article demonstrates the following Journal and Commit APIs:**

[Retrieve Journal Entries \(QjoRetrieveJournalEntries\) API](#)

[Retrieve Journal Information \(QjoRetrieveJournalInformation\) API](#)

[Retrieve Journal Receiver Information \(QjoRtvJrnReceiverInformation\) API](#)

[Journal and Commit APIs](#)

[\*\*Retrieve the source code for this API example.\*\*](#)

**Source URL:** <http://iprodeveloper.com/rpg-programming/apis-example-journal-apis-solving-spooled-files-mysteries>