

## APIs by Example: What's the Current State of Your Remote Journals?

[System i Network Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Wed, 02/15/2012 - 3:00pm

The Retrieve Journal Information (QjoRetrieveJournalInformation) API provides a wealth of information about journals, including journal receiver directories, journaled objects, and associated remote journals. Because remote journals to a great extent are employed in solutions providing replication and disaster recovery facilities, it has increasingly become vital to get access to current information detailing remote journal configuration and status. This APIs by Example article shows how to exploit the QjoRetrieveJournalInformation in such an endeavor.

The [code](#) associated with today's article delivers the Retrieve Remote Journal Attribute (RTVRMTJRNA) command, allowing you to access critical information about the remote journals associated with a specified local journal. I've also included sample code showing how to interrogate the current state of all remote journals servicing a local journal, and for any remote journals not currently active to send warning messages or e-mails and ensure they get attention. As a part of the notification methods included in the sample program, I also present a new, native Send SMTP Email (SNSMTPPEMM) command, which IBM currently has in the works, and is making available to release IBM i 6.1 and 7.1 by means of PTFs.

I've provided links to the PTF APARs in the *Compilation Instructions and Read More* sidebar at the end of this article. But let's begin with the Retrieve Journal Information (QjoRetrieveJournalInformation) API, which, according to the [online API documentation at the IBM Information Center](#), has the following parameter list:

Retrieve Journal Information (QjoRetrieveJournalInformation) API

Required Parameter Group:

1	Receiver variable	Output	Char(*)
2	Length of receiver variable	Input	Binary(4)
3	Qualified journal name	Input	Char(20)
4	Format name	Input	Char(8)
5	Journal information to retrieve	Input	Char(*)

Omissible Parameter:

6	Error Code	I/O	Char(*)
---	------------	-----	---------

Default Public Authority: \*USE

The *Receiver variable* parameter is where the QjoRetrieveJournalInformation API puts the journal information returned by the API, up to the length specified by the second parameter. [As mentioned in a previous article](#), starting with release IBM i 6.1, ILE RPG supports data structures and character variables up to lengths of 16773104 bytes, whereas these lengths for earlier releases were limited to 65535 bytes. In order to support a return variable length larger than 65535 bytes at release 5.4, I must manually allocate the storage required by the API.

In cases where the initial storage allocated for the return variable is deemed insufficient by the QjoRetrieveJournalInformation API, the API returns the actual size needed in the *Bytes available* subfield of the return variable, and this amount of storage is then reallocated immediately prior to repeating the API call. The third API parameter is where you submit the qualified name of the journal from which to retrieve information.

The fourth API parameter defines the *Format name* of the receiver variable. Two format names are currently supported, RJRNO100 and RJRNO200. Both formats return the same journal information; the only difference

between the two formats is associated to the return structure subfields *Bytes returned* and *Bytes available*. If RJRNO100 is specified, the values returned in the two fields are bytes, as suggested by the field names; but if you specify format RJRNO200, the values returned are the number of 4K units (4096 bytes) returned by the API. This allows the API to return more information than the maximum value of a 4-byte integer supports.

The fifth parameter identifying the *Journal information to be returned* offers the following array to choose from:

- Journal receiver directory information
- Journalized object information
- Remote journal information
- Auxiliary storage pool device name

One or more or all of these information segments can be returned in one single call, if requested. You indicate to the API which types of journal information to return by submitting one or more numeric keys in the range 1 to 4 as well as the associated parameter value(s). For more details on this parameter, please refer to the online API documentation, to which I've provided a link in the *Compilation Instructions and Read More* sidebar at the end of this article—or check out the CBX245 command processing program.

The sixth, final, and omissible API parameter is the standard API error data structure, which I assume you're well aware of, and therefore will leave out of scope for now. Based on the Remote journal information returned by the QjoRetrieveJournalInformation API, I decided to include the return values documented by the RTVJRMJRNA command and prompt shown below:

```

-----
                Retrieve Remote Journal Attr (RTVRMTJRNA)

Type choices, press Enter.

Journal . . . . . Name
Library . . . . . *LIBL      Name, *LIBL, *CURLIB
Relational database entry:
Reference RDB number . . . . . *FIRST      Number, *FIRST
Relationship . . . . .          *SAME, *NEXT
CL var for RDBNBR      (5 0) . .          Number
CL var for RDB          (18) . .          Character value
CL var for RMTJRNSTT   (10) . .          Character value
CL var for RMTJRN      (10) . .          Character value
CL var for RMTJRNLIB   (10) . .          Character value
CL var for RMTRCVLIB   (10) . .          Character value
CL var for RMTJRNTYP   (1)  . .          Character value
-----

```

Given the specific need, it would be quite simple to add further remote journal-related return values to the RTVRMTJRNA command. All current parameters and return values are documented in the associated help text panel group accessed by placing the cursor on the area or parameter of interest and pressing function key F1. Processing all remote journals associated with the specified journal name is done by using the ordinal number identifying each remote journal and returned in the RDBNBR return value to navigate the remote journal array, as performed in the following CL example taken from the test program included with [this article's code download](#):

```

RtvRmtJrnA Jrn( &JrnLib/&JrnNam )
  RdbEntry( *FIRST )
  RdbNbr( &RdbNbr )
  Rdb( &RdbNam )
  RmtJrnStt( &RmtJrnStt )
  RmtJrn( &RmtJrn )

DoWhile ( &RdbNbr > 0 )

If      ( &RmtJrnStt *NE 'ACTIVE' )      Do

RtvMsg  MsgId( CBX1001)
        MsgF( CBX245M )
        MsgDta( &RmtJrn *Cat &RdbNam *Cat &RmtJrnStt )

```

```

Msg( &MsgTxt )

SndSmtpEmm Rcp(( &EmlAdr *PRI ))
Subject( 'Remote journal event' )
Note( &MsgTxt )

EndDo

RtvRmtJrnA Jrn( &JrnLib/&JrnNam )
RdbEntry( &RdbNbr *NEXT )
RdbNbr( &RdbNbr )
Rdb( &RdbNam )
RmtJrnStt( &RmtJrnStt )
RmtJrn( &RmtJrn )

EndDo

```

In the initial execution of the RTVRMTJRNA command, you specify the special value \*FIRST for the command's RDBENTRY parameter. This returns the first remote journal entry and the entry's ordinal number in the RDBNBR return parameter. You then use the ordinal number returned on the previous call for all subsequent invocations of the RTVRMTJRNA command together with the special value \*NEXT as the relationship value, pointing the RTVRMTJRNA command to the remote journal entry following the one returned on the previous call.

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

### Compilation Instructions and Read More

Below you'll find instructions on how to create the Retrieve Remote Journal Attributes (RTVRMTJRNA) command, as well as links to more examples of and information on the Retrieve Journal Information (QjoRetrieveJournalInformation) API.

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

```

CBX245  -- RPGLE  -- Retrieve Remote Journal Attributes - CPP
CBX245H -- PNLGRP -- Retrieve Remote Journal Attributes - Help
CBX245X -- CMD    -- Retrieve Remote Journal Attributes
CBX245T -- CLP    -- Retrieve Remote Journal Attributes - Test

CBX245M -- CLP    -- Retrieve Remote Journal Attributes - Build command

```

To create all above command objects, compile and run the CBX245M CL program, following the instructions in the source header. You'll also find compilation instructions in the respective source headers of the individual sources. Prior to running the CBX245M program, be sure to set the message queue and email address variables &MsgQue and &EmlAdr, respectively, in the CBX245T program source to your preferred values.

Run the CBX245T test program from a command line specifying two 10-character parameters—a journal name as the first and the journal library as the second parameter—as in the following example:

```
CALL PGM( CBX245T ) PARM( 'JRNNAM' 'JRNLIB' )
```

Use a source debugger to step through the statements executed by the CBX245T program and monitor the action as it unfolds.

### Related articles and documentation:

[APIs by Example: Journal APIs Solving Spooled Files Mysteries](#)

[APIs By Example: Manage Journal Receivers](#)

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

[Sending Messages from RPG to a Syslog Server](#)

[Create and Send MIME E-mail \(QtmsCreateSendEmail\) API](#)

[Using the Create and Send MIME Email API](#)

**New feature support, SNDSMTPEMM command, release 6.1 and 7.1:**

[6.1 APAR SE49380 PTF SI44666](#)

[7.1 APAR SE48980 PTF SI44334](#)

**Source URL:** <http://iprodeveloper.com/rpg-programming/apis-example-whats-current-state-your-remote-journals>