

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

APIs by Example: Retrieve ASP Information, Part 2

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 06/16/2005 (All day)

This installment of APIs By Example will continue the examination of the Open List of ASPs (QYASPOL) API that I started in a previous installment. Part 1 offered an example of retrieving information about an ASP itself. This time around I will show how you can get at information about the disk units configured on a system by means of the QYASPOL API.

You will be familiar with some of this information if you have used the WRKDSKSTS (Work with Disk Status) command. However, the WRKDSKSTS command only offers this information to display or print. If you want to process system disk information programmatically, the QYASPOL API -- or the RTVDSKATR (Retrieve Disk Attributes) command presented in this article -- is the way to go about it.

Disk unit information is returned in the QYASPOL API's YASPO300 return format. You will find that format documented here:

<http://as40obks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/qyaspol.htm#HDRYASPL5>

The information returned in the YASPO300 format includes disk usage and I/O statistics and attributes defining the disk units' current operational status, such as mirrored unit status, disk unit status, and disk compression status.

Note that there is a minor discrepancy in the total disk capacity reported by the QYASPOL API compared to the information returned by the WRKDSKSTS command. The QYASPOL API often returns a total disk capacity that is 1 Mb bigger than the total disk capacity reported by the WRKDSKSTS command for the same disk unit.

Although the difference is insignificant, Bruce Vining of IBM took the time to research this issue and came back with the following explanation: The QYASPOL API and the WRKDSKSTS command both use the MATRMD (Materialize Resource Management Data) MI function to gather the information needed. However, the API and command are the result of two different development efforts, taking place at different points in time. MATRMD returns the disk storage capacity in bytes, whereas the API and command both returns this attribute in megabytes.

As a consequence, the developers were faced with a decision about what to do in the event of a remainder resulting from the scaling process. In this case, two different approaches were chosen: The WRKDSKSTS command does simple truncation of the capacity, while the QYASPOL API looks to see if any data was lost when scaling and if so, adds 1 Mb.

One less mystery to solve! Thanks to Bruce Vining for the above documentation.

To look at the documentation for the MATRMD MI function and learn more about the many kinds of detailed information it is capable of delivering, check out the following link:

http://publib.boulder.ibm.com/iserics/v5r1/ic2924/tstudio/tech_ref/mi/MATRMD.htm

Now on to the RTVDSKATR command; the prompt below displays all the disk attributes that are available through the RTVDSKATR command:

```

= = = = =
              Retrieve Disk Attributes (RTVDSKATR)
Type choices, press Enter.
Disk unit . . . . . *FIRST          *FIRST, *NEXT
CL var for RTNUNIT   (10 0) . .      Number
CL var for ASPNBR    (3 0) . .      Number
CL var for TYPE      (4) . .        Character value
CL var for MODEL     (4) . .        Character value
CL var for SRLNBR    (10) . .       Character value
CL var for RSRNAME   (10) . .       Character value
CL var for DISKCAP   (10 0) . .     Number
CL var for STGAVL    (10 0) . .     Number
CL var for STGSYS    (10 0) . .     Number
CL var for MIRUNITPTC (1) . .       Character value
CL var for MIRUNITRPT (1) . .       Character value
CL var for MIRUNITSTS (1) . .       Character value
CL var for UNITCTL   (2 0) . .     Number
CL var for BLKTFRTO  (10 0) . .     Number
CL var for BLKTFRFRM (10 0) . .     Number
CL var for RQSDTATO  (10 0) . .     Number
CL var for RQSDTAFRM (10 0) . .     Number
CL var for PERMBLKFRM (10 0) . .     Number
CL var for RQSPERMF  (10 0) . .     Number
CL var for SAMPLECNT (10 0) . .     Number
CL var for NOTBUSYCNT (10 0) . .     Number
CL var for COMPSTS   (1) . .       Character value
CL var for DISKPTCTYP (1) . .       Character value
= = = = =

```

As always, a help panel group that explains the individual attributes in detail is included.

Below you will find an example of how the RTVDSKATR command can be put to use in a CL program:

```

RtvDskAtr  Disk( *FIRST )          +
           RtnUnit( &RtnUnit )      +
           AspNbr( &AspNbr )        +
           Type( &Type )            +
           Model( &Model )          +
           SrlNbr( &SrlNbr )        +
           UnitCtl( &UnitCtl )
RtvNext:
  If ( &RtnUnit > 0 )      Do
  If ( &UnitCtl > 1 )      Do
  ChgVar      &RtnUnitC    &RtnUnit
  ChgVar      &UnitCtlC    &UnitCtl
  SndPgmMsg   MsgId( CPF9897 )      +
              MsgF( QCPFMSG )      +
              MsgDta( 'Disk unit'   *Bcat +
                    &RtnUnitC      *Bcat +

```

```

                                &RtnUnitC
                                'currently has unit status'
                                &UnitCtlC
                                '.)
                                ToMsgQ( *SYSOPR )
EndDo
RtvDskAtr  Disk( *NEXT )          +
           RtnUnit( &RtnUnit )    +
           AspNbr( &AspNbr )      +
           Type( &Type )          +
           Model( &Model )        +
           SrlNbr( &SrlNbr )      +
           UnitCtl( &UnitCtl )
GoTo      RtvNext
EndDo

```

The above code is included in the download for this article as a sample program called CBX137T. What it does is report any disk units that are not at the 'Disk unit is active' status to the QSYSOPR message queue. The disk unit may still be fully operational, so it could be worth taking a closer look at the exact status to ensure that the disk unit or subsystem is working as it should.

As a closer look at the code may reveal, the disk units are retrieved one by one (in disk unit number order).

To retrieve the full array of configured disk units, execute the RTVDSKATR command once, specifying the special value *FIRST for the DISK keyword, followed by a number of executions specifying the special value *NEXT.

When all disk units have been retrieved, the end of list condition is signalled by returning the value -1 in the variable specified for the RTNUNIT parameter.

The RTVDSKATR command is created from the following sources:

CBX137 -- Command processing program.

CBX137X -- Command definition source member.

CBX137H -- Command help text panel group.

Compilation instructions are found in the source headers.

This article demonstrates the following APIs:

Open List of ASPs (QYASPOL) API:

<http://as40obks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/qyaspol.htm>

Close List (QGYCLST) API:

<http://as40obks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/qgyclst.htm>

Send Program Message (QMHSNDPM) API:

<http://publib.boulder.ibm.com/iserics/v5r2/ic2924/info/apis/QMHSNDPM.htm>

Normal End (CEETREC) API:

<http://as40obks.rochester.ibm.com/iserics/v5r2/ic2924/info/apis/CEETREC.htm>

You can retrieve the source code for this API example from

http://www.pentontech.com/IBMContent/Documents/article/51090_25_RtvDskAtr.zip.

Source URL: <http://iprodeveloper.com/rpg-programming/apis-example-retrieve-asp-information-part-2>