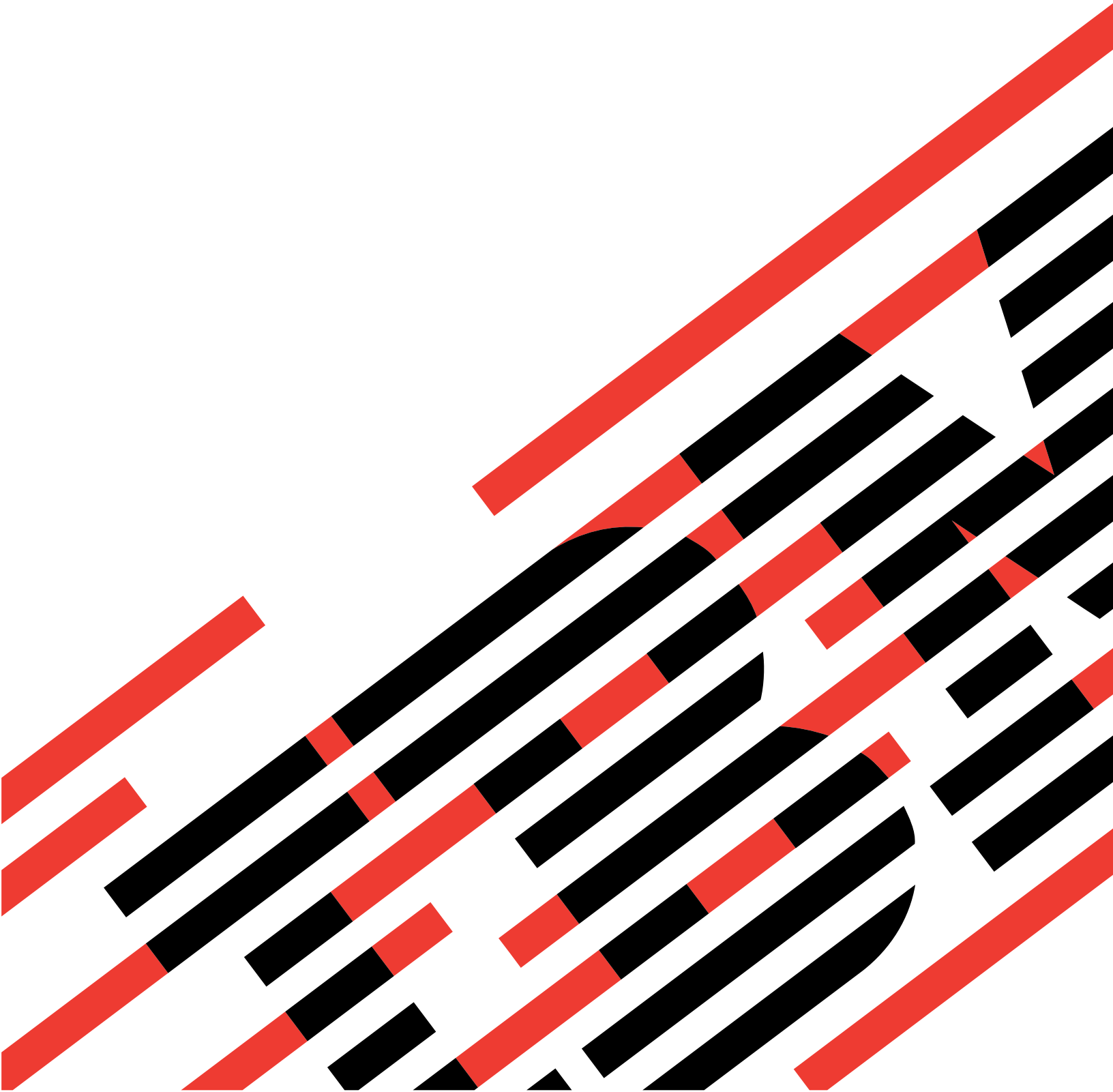
IBM

eserver

iSeries

Process Open List APIs

Version 5 Release 3



IBM

@server

iSeries

Process Open List APIs

Version 5 Release 3

Note

Before using this information and the product it supports, be sure to read the information in "Notices," on page 17.

Sixth Edition (August 2005)

This edition applies to version 5, release 3, modification 0 of Operating System/400 (product number 5722-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

© Copyright International Business Machines Corporation 1998, 2005. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 1998, 2005

Process Open List APIs

The process open list APIs are used to access the data returned by the open list APIs. You can get list entries, find entry numbers in lists and in message lists, find field numbers in lists, retrieve server job information, and close lists. Some examples of these open list APIs are:

- Open List of Job Log Messages (QGYOLJBL)
- Open List of Messages (QGYOLMSG)
- Open List of Objects (QGYOLOBJ)
- Open List of Objects to be Backed Up (QEZOLBKL)
- Open List of Printers (QGYRPRTL)
- Open List of Spooled Files (QGYOLSPL)

These list APIs can improve perceived performance when creating lists. The APIs create and make available to the caller a partial listing of the total set of files, messages, or objects. This list is immediately available to be acted upon, while the remainder of the list is being created. The user does not have to wait for the entire list to be created.

» **Note:** When using Open List APIs, you should use the Close List API to close any open lists after they have been processed and are no longer needed. This frees any internal storage associated with that list. Failure to close open lists when finished may result in the inability to process subsequent Open Lists APIs. «

The open list APIs are available only if the Host Servers option of (R)OS/400 is installed. You can install this option by using the GO LICPGM function of OS/400. Select the Install Licensed Programs option on the Work with Licensed Programs display, and select the Host Servers option on the Install Licensed Programs display. » Beginning with OS/400 Version 5 Release 3, these and other Open List APIs are available in OS/400. However, the Host Servers option may still be required so that applications from previous releases continue to function properly. Routing programs are provided with the Host Servers option to transfer requests to the corresponding program in OS/400. «

The process open list APIs and their functions are:

- "Change Server Job (QGYCHGSJ) API" on page 2 (QGYCHGSJ) sets the maximum number of auxiliary server jobs allowed for a server job with the iSeries.
- "Close List (QGYCLST) API" on page 3 (QGYCLST) closes a previously opened list. Any internal storage associated with that list is freed.
- "Find Entry Number in List (QGYFNDE) API" on page 4 (QGYFNDE) returns the number of the entry in a list of information for a given key value.
- "Find Entry Number in Message List (QGYFNDME) API" on page 6 (QGYFNDME) returns the number of the entry in a list of message information for a given key value.
- "Find Field Numbers in List (QGYFNDF) API" on page 8 (QGYFNDF) returns the number of the entries in a list of information and the value of that entry whenever the value of that field changes.
- "Get List Entries (QGYGTLE) API" on page 12 (QGYGTLE) allows requests to get entries from previously opened lists on the server.
- "Retrieve Server Job Information (QGYRTVSJ) API" on page 14 (QGYRTVSJ) returns information about auxiliary server jobs started for the current job to the system.

Top | APIs by category

© Copyright IBM Corp. 1998, 2005

APIs

These are the APIs for this category.

Change Server Job (QGYCHGSJ) API

Required Parameter Group:	
1	Number of auxiliary server jobs allowed
Input	Binary(4)
2	Error code
I/O	Char(*)
Default Public Authority: *USE	
Threadsafe: No	

The Change Server Job (QGYCHGSJ) API sets the maximum number of auxiliary server jobs allowed for a server job on the iSeries server. At least one auxiliary server job is allowed; up to five auxiliary server jobs may be allowed. An **auxiliary server job** is used to do work asynchronously from the job that started the auxiliary server job. For example, the auxiliary server job is used to complete building lists of information. All auxiliary server jobs end automatically when the submitting job ends.

The Retrieve Server Job Information (QGYRTVSJ) API can be called to retrieve the number of active auxiliary server jobs, the number of auxiliary server jobs allowed, and the job names for each active auxiliary server job.

Required Parameter Group

Number of auxiliary server jobs allowed
INPUT; BINARY(4)

The number of auxiliary server jobs that may be started for the current server job. If the number specified is less than the number that is currently allowed, no change will be made. No more than five auxiliary server jobs may be allowed.

Error code
I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0113 E	Number of auxiliary server jobs, &1, not valid.

API introduced: V3R6

Close List (QGYCLST) API

Required Parameter Group:	
1	Request handle
Input	Char(4)
2	Error code
I/O	Char(*)

Default Public Authority: *USE

Threadsafe: No

The Close List (QGYCLST) API closes a previously opened list. Any internal storage associated with that list is freed. The handle specified on the call to this API is no longer valid after the call completes.

Required Parameter Group

Request handle

INPUT; CHAR(4)

The handle to the list that is to be closed. The handle is generated by one of the following list APIs:

- Open List of Job Log Messages (QGYOLJBL)
- Open List of Messages (QGYOLMSG)
- Open List of Objects (QGYOLOBJ)
- Open List of Printers (QGYRPRTL)
- Open List of Spooled Files (QGYOLSPL)

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0001 E	Invalid handle specified.

API introduced: V3R6

Find Entry Number in List (QGYFNDE) API

Required Parameter Group:	
1	Request handle
Input	Char(4)
2	Number of keys
Input	Binary(4)
3	Key field information
Input	Array(*) of Char(12)
4	Key field values
Input	Array(*) of Char(30)
5	Entry number
Output	Binary(4)
6	Key found
Output	Char(1)
7	Error code
I/O	Char(*)

Default Public Authority: *USE

Threadsafe: No

The Find Entry Number in List (QGYFNDE) API returns the number of the entry in a list of information for a given key value.

Required Parameter Group

Request handle

INPUT; CHAR(4)

The handle of the list. This handle is generated by one of the following open list APIs:

- Open List of Objects (QGYOLOBJ)
- Open List of Printers (QGYRPRTL)
- Open List of Spooled Files (QGYOLSPL)

Number of keys

INPUT; BINARY(4)

The number of elements, within the key field information array, to search.

Key field information

INPUT; ARRAY(*) of CHAR(12)

The offset and length of the information to search.

Key field offset INPUT; BINARY(4)
The offset within the list entry to search.

Key field length INPUT; BINARY(4)
The length of the field within the list entry to search.

Reserved INPUT; CHAR(4)
An ignored field. This field must be set to hexadecimal zeros.

Key field values
INPUT; ARRAY(*) of CHAR(30)

The value of the fields indicated in the key field information parameter for which to search.

Entry number
OUTPUT; BINARY(4)

The number of the first entry in the list in which the key is found. If the key is not found in a sorted list, the number of the entry previous to where the requested entry would have been is returned (a 1 is returned if the entry not found is the first entry in the list). If the key is not found in an unsorted list, a 1 is returned. If the list is empty, a 0 is returned.

Key found
OUTPUT; CHAR(1)

Whether the entry returned is for the key requested or the key was not found. The possible values are:

0 The key was not found. The entry number returned is not associated with the key given.
1 The key was found. The entry number returned is associated with the key given.

Error code
I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0001 E	Invalid handle specified.
GUI0066 E	Number of keys must be at least &2.

API introduced: V3R6

Top | APIs by category

Find Entry Number in Message List (QGYFNDME) API

Required Parameter Group:	
1	Request handle
Input	Char(4)
2	Number of keys
Input	Binary(4)
3	Key field information
Input	Array(*) of Char(12)
4	Key field values
Input	Array(*) of Char(30)
5	Entry number
Output	Binary(4)
6	Key found
Output	Char(1)
7	Message type information
Input	Char(10)
8	Error code
I/O	Char(*)
Default Public Authority: *USE	
Threadsafe: No	

The Find Entry Number in Message List (QGYFNDME) API returns the number of the entry in a list of messages for a given key value. This API may be used on a list generated by the following APIs:

- Open List of Job Log Messages (QGYOLJBL)
- Open List of Messages (QGYOLMSG)

Required Parameter Group

Request handle
INPUT; CHAR(4)

The handle of the list, generated by one of the list messages APIs.

Number of keys
INPUT; BINARY(4)

The number of elements, within the key field information array, to search.

Key field information
INPUT; ARRAY(*) of CHAR(12)

The offset and length of information to search.

Key field offset BINARY(4)
The offset within the list entry to search.

Key field length BINARY(4)
The length of the field within the list entry to search.
Reserved CHAR(4)
An ignored field. This field must be set to hexadecimal zeros.

Key field values
INPUT; ARRAY(*) of CHAR(30)

The value of the fields indicated in the key field information parameter to search for.

Entry number
OUTPUT; BINARY(4)

The number of the first entry in the list in which the key is found. If the key is not found in a sorted list, the number of the entry previous to where the requested entry would have been is returned (a 1 is returned if the entry not found is the first entry in the list). If the key is not found in an unsorted list, a 1 is returned. If the list is empty, a 0 is returned.

Key found
OUTPUT; CHAR(1)

Whether the entry returned is for the key requested or the key was not found. The possible values are:

0 The key was not found. The entry number returned is not associated with the key given.
1 The key was found. The entry number returned is associated with the key given.

Message type information
INPUT; CHAR(10)

The type of message to search for. A valid value must be specified if the messages have been grouped. The possible values are:

**MNR* The group of messages that need a reply are searched.
**MNNR* The group of messages that do not need a reply are searched.
**SCNR* The group of sender's copy messages that need a reply are searched.

Error code
I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0001 E	Invalid handle specified.
GUI0067 E	&1 is not valid for message type.
GUI0102 E	Offset of field, &1, is not valid.

API introduced: V3R6

Top | APIs by category

Find Field Numbers in List (QGYFNDF) API

Required Parameter Group:

1	Request handle
Input	Char(4)
2	Receiver variable
Output	Char(*)
3	Length of receiver variable
Input	Binary(4)
4	Format
Input	Char(8)
5	Field specification
Input	Char(*)
6	Total number returned
Output	Binary(4)
7	Record number
I/O	Binary(4)
8	Error code
I/O	Char(*)

Default Public Authority: *USE

Threadsafe: No

The Find Field Numbers in List (QGYFNDF) API returns the number of the entry in a list of information and the value of that field whenever the value of that field changes. Two types of find operations are supported.

- Generic find operations: The caller specifies which field in the record is used to cause a break.
- Formatted find operations: The format selected determines which field or fields cause a break.

Required Parameter Group

Request handle
INPUT; CHAR(4)

The handle of the request. When a list API is called, a handle is returned upon successful completion. One of these handles is required as input to this API.

Receiver variable
OUTPUT; CHAR(*)

The receiver variable that receives the information requested. You can specify the size of the area to be smaller than the format requested as long as you specify the length parameter correctly. As a result, the API returns only the data that the area can hold.

Length of receiver variable
INPUT; BINARY(4)

The amount of data the application program is prepared to receive. If the length specified is larger than the amount of data available, the receiver is not changed beyond the amount of data available.

The receiver variable must be large enough to hold one array entry. For format FNDF0100 the minimum receiver length must be:

```
4 + field length rounded up to a multiple of 4
```

For format FNDF0200 the minimum receiver length must be:

```
40
```

Format name

INPUT; CHAR(8)

The content and format of the information returned. The possible format names are:

FNDF0100 Generic find operation is performed.

The calling program specifies the field offset and length in the field specification parameter.

Each time the field changes in the list, a record entry is added to the receiver variable.

FNDF0200 Spooled file find operation by printer is performed.

This format can only be used against a list of spooled files with format OSPL0200 that contains the following fields:

- Device name (field key 208)
- Output queue library (field key 207)
- Output queue name (field key 206)

Each time the device name field changes in the list, a record is added to the receiver variable. In addition, when the printer assigned value is 2, then each time the output queue changes in the list a record is added to the receiver variable.

Field specification

INPUT; CHAR(*)

The fields to search for a break. For the FNDF0100 format this parameter must be an array of field offsets and field lengths. The fields specified are considered one logical field for comparison when searching for changes in the fields. The values for the fields will be returned in the receiver variable concatenated in the same order they are specified in this parameter. For the FNDF0100 format, this parameter must have the following layout:

Offset		Type	Field
Dec	**Hex**		
0	0	BINARY(4)	Number of fields to search on. Valid values are 1 through 3.
Offsets vary. These fields repeat, in the order listed, for each field to search on.		BINARY(4)	Field offset
		BINARY(4)	Field length

For the FNDF0200 format, this parameter must have the following layout:

Offset		Type	Field
Dec	**Hex**		
0	0	BINARY(4)	Number of fields to search on. This must be set to zero (0).

Total number returned
OUTPUT; BINARY(4)

The total number of array entries returned in the receiver variable. This is not necessarily the total number of entries available. If the receiver variable is not large enough to hold all available entries, the record number parameter is set to the record number where the next break occurs.

Record number
I/O; BINARY(4)

On input, the record to begin searching for field breaks. This should be set to 1 for the initial call for a list so that searching begins with the first record. This program always assumes a field break to have occurred at the record specified by this parameter. Thus, the record specified is always returned in the receiver variable.

On output, this parameter indicates whether the information in the receiver variable is partial or complete. If the receiver variable is complete, this parameter will be set to zero. If the receiver variable contains partial information, this parameter will be set to the record number where the next field break occurs. This value can be specified as input on a subsequent call to this program to continue the search for field breaks.

Error code
I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Format of Receiver Variable

Format FNDF0100

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Record size
Offsets vary. These fields repeat, in the order listed, for each entry returned.		BINARY(4)	Record number
		CHAR(*)	Field value
		CHAR(*)	Padding for boundary alignment

Format FNDF0200

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Record size
Offsets vary. These fields repeat, in the order listed, for each entry returned.		BINARY(4)	Record number
		CHAR(10)	Output queue name
		CHAR(10)	Output queue library
		CHAR(1)	Printer assigned value
		CHAR(10)	Device name
		CHAR(*)	Padding for boundary alignment

Field Descriptions

Device name. The name of the printer device where the spooled file is printed.

Field value. The value of the field where the break occurs.

This is a concatenation of the fields specified to search on. The fields are concatenated in the order they were specified in the field specification parameter. The length of this field is the sum of all field lengths specified in the field specification parameter.

Output queue library. The name of the library containing the output queue that the spooled file is assigned.

Output queue name. The name of the output queue that the spooled file is assigned.

Padding for boundary alignment. A reserved field.

Printer assigned value. The value specifying whether this spooled file is assigned to a printer. Valid values are 1 through 3.

Printer Value	Description
1	Spooled file is assigned to a specific printer. The device name field contains the name of the printer.
2	Spooled file is assigned to multiple printers. The device name field is set to blanks.
3	Spooled file is not assigned to a printer. The device name field is set to blanks.

Record number. The record number in the list where the break occurs.

Record size. The size of the record in the array shown.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C19 E	Error occurred with receiver variable specified.
CPF3C21 E	Format name &1 is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0001 E	Invalid handle specified.
GUI0002 E	&2 is not valid for length of receiver variable.
GUI0101 E	List information is not valid. Reason code &1.
GUI0102 E	Offset of field, &1, is not valid.
GUI0103 E	Invalid handle specified. Length of field, &2, is not valid. Reason code &1.
GUI0104 E	Record number, &1, is not valid.
GUI0106 E	List does not contain required fields.
GUI0107 E	Number of fields, &2, is not valid. Reason code &1.

API introduced: V3R6

Top | APIs by category

Get List Entries (QGYGTLE) API

Required Parameter Group:	
1	Receiver variable
Output	Char(*)
2	Length of receiver variable
Input	Binary(4)
3	Request handle
Input	Char(4)
4	List information
Output	Char(80)
5	Number of records to return
Input	Binary(4)
6	Starting record
Input	Binary(4)
7	Error code
I/O	Char(*)
Default Public Authority: *USE	
Threadsafe: No	

The Get List Entries (QGYGTLE) API allows requests to get entries from previously opened lists on the iSeries server. A list will exist if an initial request has already been made and the list was not closed using the Close List (QGYCLST) API.

Initial requests are made by calling the following APIs:

- Open List of Job Log Messages (QGYOLJBL)
- Open List of Messages (QGYOLMSG)
- Open List of Objects (QGYOLOBJ)
- Open List of Printers (QGYRPRTL)
- Open List of Spooled Files (QGYOLSPL)
- Open List of User Certificates (QSYOLUC)
- Open List of Validation List Entries (QSYOLVLE)
- Retrieve Objects Secured by Authorization List (QGYRATLO)

Required Parameter Group

Receiver variable
OUTPUT; CHAR(*)

The receiver variable that receives the information requested. You can specify the size of the area to be smaller than the format requested as long as you specify the length parameter correctly. As a result, the API returns only the data that the area can hold.

Length of receiver variable
INPUT; BINARY(4)

The length of the receiver variable provided. The length of receiver variable parameter may be specified up to the size of the receiver variable specified in the user program. If the length of receiver variable parameter specified is larger than the allocated size of the receiver variable specified in the user program, the results are not predictable. The minimum length is 8 bytes.

Request handle
INPUT; CHAR(4)

The handle of the request. The value of this determines from which user space to retrieve the information.

List information
OUTPUT; CHAR(80)

Information about the list from which entries are being returned. For a description of the layout of this parameter, see Format of Open List Information.

Number of records to return
INPUT; BINARY(4)

The number of records in the list (starting with the record indicated in the starting record parameter) to take from the user space and put into the receiver variable. This value must be greater than or equal to zero.

If the value zero is specified, then only the list information is returned and no actual list entries are returned.

Starting record
INPUT; BINARY(4)

The entry in the list that will be the first entry to be put into the receiver variable. The value must be greater than zero or one of the special values of 0 or -1.

The special value of 0 indicates that the list information should be returned to the caller immediately. The special value 0 is only allowed when the number of records to return parameter is zero.

The special value of -1 indicates that the whole list should be built before the list information is returned to the caller.

The following table shows how the number of records to return and the starting record parameters interact with each other. The record parameter is represented by an X. The number of records to return parameter is represented by a Y.

Starting Record (X)	Number of records to return (Y=0)	Number of records to return (Y>0)
X = 0	Immediately return only the list information	Invalid combination. An error message is sent.
X = -1	Return only the list information, but wait until the whole list is built	Wait until the whole list is built and then return the number of records requested from the end of the list. See note.
Note: If the receiver variable is not large enough to hold the number of records requested, then only those that will fit into the receiver variable will be returned, but they will always be the last ones in the list.		
X > 0	Return only the list information, but wait until list entries have been built, as specified in the starting record parameter.	Return the number of list entries specified in the number of records to return parameter starting with the entry specified in the starting record parameter.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

Format of Receiver Variable

The format of the receiver variable was specified when the list was originally created.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C19 E	Error occurred with receiver variable specified.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.
GUI0001 E	Invalid handle specified.
GUI0002 E	&2 is not valid for length of receiver variable.
GUI0006 E	&1 is not valid for starting record number.
GUI0027 E	&1 is not valid for number of records to return.
GUI0114 E	The list cannot be completed. No server jobs are running or can be started.
GUI0115 E	The list has been marked in error. See the previous messages.
GUI0118 E	Starting record cannot be 0 when records have been requested.

API introduced: V3R6

Top | APIs by category

Retrieve Server Job Information (QGYRTVSJ) API

Required Parameter Group:

1	Receiver variable
Output	Char(*)
2	Length of receiver variable
Input	Binary(4)
3	Format name
Input	Char(8)
4	Error code
I/O	Char(*)

Default Public Authority: *USE

Threadsafe: No

The Retrieve Server Job Information (QGYRTVSJ) API returns information about auxiliary server jobs started for the current job on the iSeries server. This API will return the number of auxiliary server jobs and the job names for each auxiliary server job. This information can be used to:

- Retrieve information about the auxiliary server jobs by calling the Retrieve Job Information (QUSRJOBI) API.
- Change the run-time attributes of one or more auxiliary server jobs using the Change Job (CHGJOB) CL command.

An **auxiliary server job** is used to do work asynchronously from the job that started the auxiliary server job. For example, the auxiliary server job is used to complete building lists of information.

The Change Server Job (QGYCHGSJ) API can be used to change the maximum number of auxiliary server jobs that can be active at any one time.

Required Parameter Group

Receiver variable
OUTPUT; CHAR(*)

The receiver variable that receives the information requested. You can specify the size of the area to be smaller than the format requested as long as you specify the length parameter correctly. As a result, the API returns only the data that the area can hold.

Length of receiver variable
INPUT; BINARY(4)

The length of the receiver variable provided. The length of receiver variable parameter may be specified up to the size of the receiver variable specified in the user program. If the length of receiver variable parameter specified is larger than the allocated size of the receiver variable specified in the user program, the results are not predictable. The minimum length is 8 bytes.

Format name
INPUT; CHAR(8)

The format of the information to be returned. You can use this format:

SJBI0100 Basic auxiliary server job information. For details see "SJBI0100 Format."

Error code
I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter.

SJBI0100 Format

The following table describes the information returned in the receiver variable for the SJBI0100 format. For detailed descriptions of the fields, see "Field Descriptions" on page 16.

Offset			
Dec	**Hex**	**Type**	**Field**
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	BINARY(4)	Number of active auxiliary server jobs
12	C	BINARY(4)	Number of auxiliary server jobs allowed
16	10	BINARY(4)	Offset to auxiliary server job information
20	14	BINARY(4)	Job information record size

Offset		Type	Field
Dec	Hex		
Offsets vary. These fields repeat for each active auxiliary server job.		CHAR(26)	Qualified auxiliary server job name
		CHAR(16)	Internal job identifier

Field Descriptions

Bytes available. The number of bytes of data available to be returned. All available data is returned if enough space is provided.

Bytes returned. The number of bytes of data returned.

Internal job identifier. A value sent to other APIs to speed the process of locating the job on the system. Only OS/400 APIs use this identifier. The identifier is not valid following an initial program load (IPL). If you attempt to use it after an IPL, an exception occurs.

Job information record size. The length of the auxiliary server job information record.

Number of active auxiliary server jobs. The number of auxiliary server jobs currently running for the current job.

Number of auxiliary server jobs allowed. The number of auxiliary server jobs allowed to be running at one time. The values are 1 through 5.

Offset to auxiliary server job information. The offset from the beginning of the receiver variable to the beginning of the auxiliary server job names.

Qualified auxiliary server job name. The qualified job name of the auxiliary server job. The qualified job name consists of the following fields:

CHAR(10)	Job name
CHAR(10)	User name
CHAR(6)	Job number

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF3C19 E	Error occurred with receiver variable specified.
CPF3C21 E	Format name &1 is not valid.
CPF3C24 E	Length of the receiver variable is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V3R6

Top | APIs by category

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

```
IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY  10504-1785
U.S.A.
```

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

```
IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan
```

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

© Copyright IBM Corp. 1998, 2005

IBM Corporation
Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, IBM License Agreement for Machine Code, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:
Advanced 36
Advanced Function Printing
Advanced Peer-to-Peer Networking
AFP
AIX
AS/400
COBOL/400
CUA
DB2
DB2 Universal Database
Distributed Relational Database Architecture
Domino
DPI

DRDA
eServer
GDDM
IBM
Integrated Language Environment
Intelligent Printer Data Stream
IPDS
iSeries
Lotus Notes
MVS
Netfinity
Net.Data
NetView
Notes
OfficeVision
Operating System/2
Operating System/400
OS/2
OS/400
PartnerWorld
PowerPC
PrintManager
Print Services Facility
RISC System/6000
RPG/400
RS/6000
SAA
SecureWay
System/36
System/370
System/38
System/390
VisualAge
WebSphere
xSeries

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Terms and conditions for downloading and printing publications

Permissions for the use of the information you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

Personal Use: You may reproduce this information for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of this information, or any portion thereof, without the express consent of IBM(R).

Commercial Use: You may reproduce, distribute and display this information solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of this information, or reproduce, distribute or display this information or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the information or any data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the information is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations. IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THIS INFORMATION. THE INFORMATION IS PROVIDED ″AS-IS″ AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

All material copyrighted by IBM Corporation.

By downloading or printing information from this site, you have indicated your agreement with these terms and conditions.

Code disclaimer information

This document contains programming examples.

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, IBM(R), ITS PROGRAM DEVELOPERS AND SUPPLIERS MAKE NO WARRANTIES OR CONDITIONS EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, REGARDING THE PROGRAM OR TECHNICAL SUPPORT, IF ANY.

UNDER NO CIRCUMSTANCES IS IBM, ITS PROGRAM DEVELOPERS OR SUPPLIERS LIABLE FOR ANY OF THE FOLLOWING, EVEN IF INFORMED OF THEIR POSSIBILITY:

1. LOSS OF, OR DAMAGE TO, DATA;
2. SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR ANY ECONOMIC CONSEQUENTIAL DAMAGES; OR
3. LOST PROFITS, BUSINESS, REVENUE, GOODWILL, OR ANTICIPATED SAVINGS.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

Printed in USA