



iSeries

Miscellaneous APIs

Version 5 Release 3





iSeries

Miscellaneous 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 43.

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

Miscellaneous APIs	1
APIs	1
General Miscellaneous APIs	1
Add Seed for Pseudorandom Number Generator (QC3ADDS, Qc3AddPRNGSeed) API	2
Authorities and Locks	2
Required Parameter Group	2
Error Messages	3
Check Communications Trace (QSCCHKCT) API	3
Authorities and Locks	3
Required Parameter Group	4
Error Messages	4
Control Device (QTACTLDV) API	5
Authorities and Locks	5
Required Parameter Group	6
CTL0100 Format	7
Field Descriptions	7
Error Messages	8
Reason Codes	8
Usage Notes	9
Usage Example	9
Convert Date and Time Format (QWCCVTD) API	10
Required Parameter Group	11
Optional Parameter Group 1.	12
Optional Parameter Group 2.	13
Input and Output Variable Formats	13
16-Byte Character Date and Time Value Structure	13
17-Byte Character Date and Time Value Structure	14
19-Byte Character Date and Time Value Structure	14
20-Byte Character Date and Time Value Structure	14
DOSGetDateTime Value Structure	14
Time Zone Information Value Structure	15
Field Descriptions	15
Usage Notes	16
Error Messages	18
Generate Pseudorandom Numbers (QC3GENRN,Qc3GenPRNs) API	18
Authorities and Locks	19
Required Parameter Group	19
Error Messages	20
Remove All Bookmarks from a Course (QEARMVBM) API.	20
Authority	20
Required Parameter Group	20
Error Messages	20
Retrieve Data (QPARTVDA) API	21
Required Parameter Group	21
Error Messages	22
Start Pass-Through (QPASTRPT) API.	22


Authorities and Locks	22
Required Parameter Group	23
PAST0100 Format	23
PAST0200 Format	24
Field Descriptions	25
Error Messages	26
Update Device Microcode (QTAUPDDV) API	27
Authorities and Locks	27
Required Parameter Group	28
Optional Parameter Group	28
Error Messages	28
Using the WebFacing Environment API (QqfEnvironment)	29
Examples	29
User Application APIs.	30
Remove User Application Information (QsyRemoveUserApplicationInfo) API	31
Authorities and Locks	31
Required Parameter Group	31
Error Messages	32
Retrieve User Application Information (QsyRetrieveUserApplicationInfo) API	32
Authorities and Locks	33
Required Parameter Group	33
Receiver Variable Description	34
Format of Returned Records Feedback Information	34
Field Descriptions	34
Error Messages	35
Update User Application Information (QsyUpdateUserApplicationInfo) API.	36
Authorities and Locks	36
Required Parameter Group	36
Error Messages	37
Exit Programs	38
Device Selection Exit Program	38
Authorities and Locks	38
Required Parameter Group	39
PDSC0100 Format	39
PDSR0100 Format	39
Field Descriptions	40
Coding Guidelines	40

Appendix. Notices	43
Trademarks	44
Terms and conditions for downloading and printing publications	45
Code disclaimer information.	46

Miscellaneous APIs

The miscellaneous APIs are APIs that do not logically fall in a specific part of the OS/400^(R) reference information.

The miscellaneous APIs consist of:

- “General Miscellaneous APIs”
-  “User Application APIs” on page 30 

APIs by category



APIs

These are the APIs for this category.

General Miscellaneous APIs

The general miscellaneous APIs perform a variety of functions, including removing bookmarks from a course, converting date and time, starting pass-through, and retrieving data on a target system.

The general miscellaneous APIs are:

- “Add Seed for Pseudorandom Number Generator (QC3ADDS, Qc3AddPRNGSeed) API” on page 2 (QC3ADDS, Qc3AddPRNGSeed) allows the user to add seed into the server’s pseudorandom number generator system seed digest.
- “Check Communications Trace (QSCCHKCT) API” on page 3 (QSCCHKCT) returns, in bytes, the maximum size configured for the communications trace tool.
- “Control Device (QTACTLDV) API” on page 5 (QTACTLDV) provides a direct command interface to a device.
- “Convert Date and Time Format (QWCCVTD) API” on page 10 (QWCCVTD) allows you to convert date and time formats from one format to another format.
- “Generate Pseudorandom Numbers (QC3GENRN,Qc3GenPRNs) API” on page 18 (QC3ADDS, Qc3GenPRNs) generates a pseudorandom binary stream.
- “Remove All Bookmarks from a Course (QEARMVBM) API” on page 20 (QEARMVBM) allows you to remove the bookmarks from a Tutorial System Support course.
- “Retrieve Data (QPARTVDA) API” on page 21 (QPARTVDA) retrieves up to 1KB of user data, which was passed to this system with the Start Pass-through (QPASTRPT) API.
- “Start Pass-Through (QPASTRPT) API” on page 22 (QPASTRPT) starts a 5250 pass-through session and optionally passes up to 1KB of user data from the source system to the target system. This data can be accessed on the target system with the Retrieve Data (QPARTVDA) API.
-  “Update Device Microcode (QTAUPDDV) API” on page 27 (QTAUPDDV) provides an interface for updating device microcode from a code image stored in an Integrated File System (IFS) stream file. 
- “Using the WebFacing Environment API (QqfEnvironment)” on page 29 (QqfEnvironment) 0

The exit program within the general miscellaneous APIs is:

- “Device Selection Exit Program” on page 38 provides an interface to control virtual device selection and automatic creation used by the system for connection requests from clients using virtual device support.

Top | “Miscellaneous APIs” | APIs by category

Add Seed for Pseudorandom Number Generator (QC3ADDSD, Qc3AddPRNGSeed) API

Required Parameter Group:	
1	Seed data
Input	Char(*)
2	Seed data length
Input	Binary(4)
3	Error Code
I/O	Char(*)
	Service Program Name: QC3PRNG
	Default Public Authority: *USE
	Threadsafe: Yes

The Add Seed for Pseudorandom Number Generator [»](#) (OPM, QC3ADDSD; ILE, Qc3AddPRNGSeed) [«](#) API allows the user to add seed into the server's pseudorandom number generator system seed digest.

The pseudorandom number generator is composed of two parts: pseudorandom number generation and seed management. Pseudorandom number generation is performed using the FIPS 186-1 algorithm. (See the Generate Pseudorandom Numbers (Qc3GenPRNs) API.) Cryptographically-secure pseudorandom numbers rely on good seed. The FIPS 186-1 key and seed values are obtained from the system seed digest. The server automatically generates seed using data collected from system information or by using the random number generator function on a cryptographic coprocessor, such as a 4758, if one is available. System-generated seed can never be truly unpredictable. If a cryptographic coprocessor is not available, you can use this API to add your own random seed to the system seed digest. This should be done as soon as possible any time the Licensed Internal Code is installed.

Authorities and Locks

All object (*ALLOBJ) special authority is needed to use this API.

User Profile Authority
*ALLOBJ

Required Parameter Group

Seed data

INPUT; CHAR(*)

The input seed data for the system seed digest.

It is important that the seed data be unpredictable and have as much entropy as possible. Entropy is the minimum number of bits needed to represent the information contained in some data. For seeding purposes, entropy is a measure of the amount of uncertainty or unpredictability of the seed. The system seed digest holds a maximum of 160 bits of entropy. You should add at

least that much entropy to refresh the system seed digest totally. Possible sources of seed data are coin flipping, keystroke or mouse timings, or a noise source such as the one available on the 4758 Cryptographic Coprocessor.

Seed data length

INPUT; BINARY(4)

The length of the seed data, in bytes. If this length is 0, no seed data is added.

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
CPF222E E	*ALLOBJ special authority is required.
CPF3C17 E	Error occurred with input data parameter.
CPF3CF1 E	Error code parameter not valid.

API introduced: V5R1

[Top](#) | [“Miscellaneous APIs,” on page 1](#) | [APIs by category](#)

Check Communications Trace (QSCCHKCT) API

Required Parameter Group:	
1	Storage allocated
Output	Binary(8)
2	Storage in use
Output	Binary(8)
3	Error code
I/O	Char(*)
	Default Public Authority: *USE
Threadsafe: No	

The Check Communications Trace (QSCCHKCT) API returns, in bytes, the maximum size configured for the communications trace tool and the portion of that size that is currently in use for all communications traces active (running or stopped state), or zero if no traces are active.

Authorities and Locks

Caller must have *SERVICE special authority, or be authorized to the Service Trace function of OS/400 through iSeries Navigator’s Application Administration support.

Required Parameter Group

Storage allocated

OUTPUT; Binary(8)

The variable containing the maximum bytes configured for the communications trace tool after the QSCCHKCT API has completed processing.

Storage in use

Output; Binary(8)

The variable containing the total bytes in use for all communications traces active (running or stopped state), or zero if no traces are active, after the QSCCHKCT API has completed processing.

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

CPF222E E	&1 special authority is required.
CPF39A8 E	Not authorized to communications trace service tool.
CPF39B6 E	Communications trace function cannot be performed.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V5R2

[Top](#) | [“Miscellaneous APIs,” on page 1](#) | [APIs by category](#)

Control Device (QTACTLDV) API

Required Parameter Group:	
1	Device name
Input	Char(10)
2	Requested function
Input	Binary(4)
3	Send buffer
Input	Char(*)
4	Length of send buffer
Input	Binary(4)
5	Receive buffer
Output	Char(*)
6	Length of receive buffer
Input	Binary(4)
7	Command format
Input	Char(8)
8	Command data
Input	Char(*)
9	Length of command data
Input	Binary(4)
10	Error code
I/O	Char(*)
	Default Public Authority: *EXCLUDE
Threadsafe: Conditional; see "Usage Notes" on page 9.	

The Control Device (QTACTLDV) API provides a direct command interface to a device. The caller of this API can issue any command directly to a device and transfer data to or from the device.

Note: For tape devices, the Retrieve Device Capabilities (QTARDCAP) API can be used to determine if the tape device supports the QTACTLDV API. Other kinds of devices currently do not support this API. This API can be used for a tape device within a media library if the device is deallocated from the library.

Note: Incorrect use of this API can cause damage to data saved on tape media or can interfere with I/O processor function.

Authorities and Locks

API Public Authority
*EXCLUDE

Special Authority
*SERVICE

Device Description Authority

*CHANGE

Device Description Lock

*EXCLRD

Required Parameter Group

Device name

INPUT; CHAR(10)

The device description to which the request is sent.

Requested function

INPUT; BINARY(4)

The function to perform. The possible values are:

- 1 Open a connection to the device. This function must be performed before any commands can be sent to the device. The device must be varied on before this function is performed. No other job can use the device while the connection is open.
- 2 Send a command to the device. When running in a multithreaded environment, a command sent by a thread must be complete before another command can be sent by another thread.
- 3 Close the connection to the device. This function must be performed after the user has completed sending commands to the device. No other job can use the device until the connection is closed.

Send buffer

INPUT; CHAR(*)

A buffer containing data to send to the device when a data transfer command is sent. No support is provided to send and receive data on the same command.

This parameter is ignored if the length of the send buffer is 0.

Length of send buffer

INPUT; BINARY(4)

The length of the send buffer.

This parameter must be 0 for the open connection and close connection functions.

Receive buffer

OUTPUT; CHAR(*)

A buffer to store data received from the device after a data transfer command is sent. No support is provided to send and receive data on the same command.

This parameter is ignored if the length of the receive buffer is 0.

Length of receive buffer

INPUT; BINARY(4)

The length of the receive buffer.

This parameter must be 0 for the open connection and close connection functions.

Command format

INPUT; CHAR(8)

The format of the command data. The following format is supported.

CTLD0100

Issue command to a tape device.

Note: The connection must be opened using the open function before a command can be issued to the device.

See “CTLD0100 Format” for more information on the command data format.

Command data

INPUT; CHAR(*)

The variable that contains the command data.

This parameter is ignored if the length of command data is set to 0.

Length of command data

INPUT; BINARY(4)

The length of the command data to be sent to the device. The command data must be 0, or a minimum of 32 bytes long and a maximum of 56 bytes long.

This parameter must be 0 for the open connection and close connection functions.

Error code

I/O; CHAR(*)

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

CTLD0100 Format

The following table shows the command information that is required for the CTLD0100 format. For more details about the fields in the following table, see “Field Descriptions.”

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Data transfer direction
4	4	BINARY(4)	Requested transfer length
8	8	BINARY(4)	Ignore length errors
12	C	BINARY(4)	Command timeout value
16	10	BINARY(4)	Type of command
20	14	BINARY(4)	Offset to command string
24	18	BINARY(4)	Length of command string
28	1C	BINARY(4)	Reserved
		CHAR(*)	Command string

Field Descriptions

Command string. The command string to send to the device. See the device specifications to determine what command strings are supported by the device.

Command timeout value. The time, in seconds, to wait for the command to complete. Valid values are 1 through 7200.

Data transfer direction. The direction of any data transfer associated with the command. The possible values are:

- 0 No data transfer.
- 1 Receive data from the device.
- 2 Send data to the device.

Ignore length errors. The possible values are:

- 0 Report length errors.
- 1 Ignore length errors.

Length of command string. The length of the command string to send to the device. Valid values are 0 through 24.

Offset to command string. The offset from the start of the command data, in bytes, to the start of the command string. Valid values are 32 and greater.

Requested transfer length. The expected length of the data to be transferred by the command. The requested transfer length must be less than or equal to the length of the buffer parameter that will be used to send or receive the data.

Note: This field must be 0 for commands with no data transfer.

Reserved. An ignored field. This value must be set to 0.

Type of Command. The type of command. The possible values are:

- 0 Small Computer System Interface (SCSI) command.
- 1 Reset the device.

Error Messages

For descriptions of the reason codes in CPF67C8, see “Reason Codes.”

Message ID	Error Message Text
CPF222E E	&1 special authority is required.
CPF24B4 E	Severe error while addressing parameter list.
CPF3C1D E	Length specified in parameter &1 not valid.
CPF3C21 E	Format name &1 is not valid.
CPF3C39 E	Value for reserved field not valid.
CPF3C3C E	Value for parameter &1 not valid.
CPF3C4C E	Value not valid for field &1.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF6708 E	Command ended due to error.
CPF67C8 E	Command failed for device &1. Reason code &2.
CPF9814 E	Device &1 not found.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

Reason Codes

This topic contains the description of the reason codes returned in message CPF67C8.

Note: For command timeouts, I/O processor errors, system bus errors, and device bus errors, a reset operation might occur on the device bus. Other devices attached to the same bus may be affected by this reset operation.

Possible reason code values are:

'010300'x	Command length not valid: The command length was too long for existing device interface specifications.
'010700'x	Data length not valid: The IOP does not support the size of the data transfer. Use the Retrieve Device Capabilities (QTARDCAP) API to determine the maximum block size supported.
'020900'x	Insufficient data: The data transfer buffer is not large enough.
'02C0yy'x	Device detected error: The tape device reported an error condition. For an SCSI type of command, yy is set to the value of the completion status.
'02C100'x	Selection timeout: The tape device did not respond to the command. Check device power and cables and retry the command. If the problem persists, contact your hardware service provider.
'02C200'x	I/O processor length error: Length error on the data transfer. The ignore length errors field was not set in the command data.
'02C300'x	I/O processor error: The I/O processor card detected an internal hardware failure. Contact your hardware service provider.
'02C400'x	Command timed out: The tape device did not complete the requested command within the specified time. Correct the command timeout value and retry the command. If the problem persists, contact your hardware service provider.
'02C500'x	System bus error: The host internal system bus failed. Contact your hardware service provider.
'02C600'x	Device bus error: The I/O processor detected a failure in the device interface. Check the device power and cables and retry the command. If the problem persists, contact your hardware service provider.
'100000'x	Open failure: A connection could not be opened to the device. The device may not be varied on or is being used by another job.
'100001'x	Open failure: A connection could not be opened to the device. The device does not support the QTACTLDV API.
'100002'x	Open failure: A connection could not be opened to the device. The device is in a failed state.
'200000'x	Close failure: The connection to the device could not be closed. The device may not be varied on or is being used by another job.
'200002'x	Close failure: The connection to the device could not be closed. The device is in a failed state.
'300000'x	Device not valid: The device specified is not a tape device.
'300001'x	Resource not valid: The resource name associated with the specified device is not valid or does not exist.
'400000'x	Connection not open: The command could not be completed because there is not an open connection.

Usage Notes

When running in a multithreaded environment, a command sent by a thread to a device must be complete before a command can be sent by another thread to the same device.

Usage Example

See Using the QTACTLDV API in API examples for an example of how to use the QTACTLDV API.

API introduced: V4R4

“Control Device (QTACTLDV) API” on page 5 | “Miscellaneous APIs,” on page 1 | APIs by category

Convert Date and Time Format (QWCCVTDT) API

Required Parameter Group:

1 Input format

Input Char(10)

2 Input variable

Input Char(*)

3 Output format

Input Char(10)

4 Output variable

Output Char(*)

5 Error code

I/O Char(*)



Optional Parameter Group 1:

6 Input time zone

Input Char(10)

7 Output time zone

Input Char(10)

8 Time zone information

Output Char(*)

9 Length of time zone information

Input Bin(4)

10 Precision indicator

Input Char(1)

Optional Parameter Group 2:

11 Input time indicator

Input Char(1)



Default Public Authority: *USE

Threadsafe: Yes

The Convert Date and Time Format (QWCCVTDT) API converts date and time values from one format to another format. The QWCCVTDT API lets you:

- Convert a time-stamp (*DTS, for system time-stamp) value to character format
- Convert a character date and time value to time-stamp format
- Convert a date from one character format to another

- **»** Convert a date and time based on input and output time zone values and return the time zone information that is associated with the converted output
- Specify a precision of milliseconds or microseconds for your input and output variables
- Retrieve a current clock time based on the output time zone and return it based on the output format you specify **«**

» For additional information on converting dates and times, see “Usage Notes” on page 16.

«

Required Parameter Group

Input format

INPUT; CHAR(10)

The format of the data you give QWCCVTDT to convert. Valid values are:

<i>*CURRENT</i>	» The current system time. «
<i>*DTS</i>	System time-stamp.
<i>*JOB</i>	The format given in the DATFMT job attribute.
<i>*SYSVAL</i>	The format given in the QDATFMT system value.
<i>*YMD</i>	YYMMDD (year, month, day) format.
<i>*YYMD</i>	YYYYMMDD (4-digit year, month, day) format.
<i>*MDY</i>	MMDDYY (month, day, year) format.
<i>*MDYY</i>	MMDDYYYY (month, day, 4-digit year) format.
<i>*DMY</i>	DDMMYY (day, month, year) format.
<i>*DMYY</i>	DDMMYYYY (day, month, 4-digit year) format.
<i>*JUL</i>	Julian format (YYDDD (year, day of year)).
<i>*LONGJUL</i>	Long Julian format (YYYYDDD (4-digit year, day of year)).

Input variable

INPUT; CHAR(*)

The data to be converted. If the input format is **CURRENT*, then this parameter is not used. See “Input and Output Variable Formats” on page 13 to determine the structure of the input variable for all other input formats.

Output format

INPUT; CHAR(10)

The format to convert the data to. Valid values are:

<i>*DTS</i>	System time-stamp.
<i>*JOB</i>	The format given in the DATFMT job attribute
<i>*SYSVAL</i>	The format given in the QDATFMT system value
<i>*YMD</i>	YYMMDD format
<i>*YYMD</i>	YYYYMMDD format
<i>*MDY</i>	MMDDYY format
<i>*MDYY</i>	MMDDYYYY format
<i>*DMY</i>	DDMMYY format
<i>*DMYY</i>	DDMMYYYY format
<i>*JUL</i>	Julian format (YYDDD)
<i>*LONGJUL</i>	Long Julian format (YYYYDDD)
<i>*DOS</i>	DOSGetDateTime format. The <i>*DOS</i> value can be specified only when <i>*CURRENT</i> or <i>*DTS</i> is specified for the input format parameter.

Output variable

OUTPUT; CHAR(*)

The converted data. » If the output format is *DOS, the first 11 characters of this parameter are used. For details, see “DOSGetDate/Time Value Structure” on page 14. See “Input and Output Variable Formats” on page 13 to determine the structure of the output variable for all other output formats. «

Error code

I/O; CHAR(*)

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



Optional Parameter Group 1

Input time zone

INPUT; CHAR(10)

Specifies the time zone associated with the input variable. If the input format is *CURRENT, then this parameter is not used. The default value is *SYS. Valid values are:

- *SYS The input variable is a local system time value and the associated time zone is specified by the time zone system value.
- *UTC The input variable is a Coordinated Universal Time (UTC) value.
- *JOB The input variable is a local job time value and the associated time zone is specified by the time zone job attribute.
- Time zone name* Specifies the name of a time zone description (*TIMZON) object.

Output time zone

INPUT; CHAR(10)

Specifies the time zone associated with the output variable. The default value is *SYS. Valid values are:

- *SYS The output variable is a local system time value and the associated time zone is specified by the time zone system value.
- *UTC The output variable is a Coordinated Universal Time (UTC) value.
- *JOB The output variable is a local job time value and the associated time zone is specified by the time zone job attribute.
- Time zone name* Specifies the name of a time zone description (*TIMZON) object.

Time zone information

OUTPUT; CHAR(*)

Specifies the time zone information associated with the output time zone. If 0 is specified for the length of time zone information, then this parameter is not used. For the format of the structure, see “Time Zone Information Value Structure” on page 15.

Length of time zone information

INPUT; BIN(4)

Specifies the length of the time zone information to be returned. The minimum length is 0 which indicates to not return any time zone information.

Precision indicator

INPUT; CHAR(1)

Specifies the precision of the input and output variables. The default value is 0 or milliseconds. Valid values are:

- 0 The input and output variables will have a precision in milliseconds.

1 The input and output variables will have a precision in microseconds.

Optional Parameter Group 2

Input time indicator

INPUT; CHAR(1)

Specifies which segment of time to use when the input variable has a date and time value that matches a repeated time. Otherwise, this parameter is not used. Repeated times occur when time changes from Daylight Saving Time (DST) to Standard Time (ST). For example, if DST ends on a given day at 02:00AM, then the segment of time from 01:00:00.000000 to 01:59:59.999999 on that day repeats. The first segment of time is considered in DST and the second segment is considered in ST. The default value is 1 or use the DST segment. For additional information on this parameter, see “Usage Notes” on page 16.

- 0 The input variable contains a date and time value that is contained in the second or Standard Time segment.
- 1 The input variable contains a date and time value that is contained in the first or Daylight Saving Time segment.

Input and Output Variable Formats

This table shows the format used for the input or output variable parameters.

Input or Output Format	Input or Output Variable
*DTS	System time-stamp. The first 8 characters are used.
*YYMD, *MDYY, *DMYY, *LONGJUL in milliseconds	The first 17 characters are used. See “17-Byte Character Date and Time Value Structure” on page 14.
All other character formats in milliseconds	The first 16 characters are used. See “16-Byte Character Date and Time Value Structure.”
*YYMD, *MDYY, *DMYY, *LONGJUL in microseconds	The first 20 characters are used. See “20-Byte Character Date and Time Value Structure” on page 14.
All other character formats in microseconds	The first 19 characters are used. See “19-Byte Character Date and Time Value Structure” on page 14.



16-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *JOB, *SYSVAL, *YMD, *MDY, *DMY, and *JUL and the precision indicator specifies milliseconds.

Offset	Description
0	Century. Possible values are 0, which indicates years 19xx, 1, which indicates years 20xx and so forth through 9, which indicates years 28xx.
1-6	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
7-12	Time, in HHMMSS (hours, minutes, seconds) format.

Offset	Description
13-15	Milliseconds. This value cannot be blanks.

17-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *YYMD, *MDYY, *DMYY, and *LONGJUL >> and the precision indicator specifies milliseconds. <<

Offset	Description
0-7	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
8-13	Time, in HHMMSS (hours, minutes, seconds) format.
14-16	Milliseconds. This value cannot be blanks.



19-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *JOB, *SYSVAL, *YMD, *MDY, *DMY, and *JUL and the precision indicator specifies microseconds.

Offset	Description
0	Century. Possible values are 0, which indicates years 19xx, 1, which indicates years 20xx and so forth through 9, which indicates years 28xx.
1-6	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
7-12	Time, in HHMMSS (hours, minutes, seconds) format.
13-18	Microseconds. This value cannot be blanks.

20-Byte Character Date and Time Value Structure

This table shows the structure used for the input and output variables when the format is *YYMD, *MDYY, *DMYY, and *LONGJUL and the precision indicator specifies microseconds..

Offset	Description
0-7	Date, left-justified. This value cannot be all blanks or all zeros. Left-justify Julian dates, using blanks to fill the space.
8-13	Time, in HHMMSS (hours, minutes, seconds) format.
14-19	Microseconds. This value cannot be blanks.



DOSGetDateTime Value Structure

This table shows the structure used for the output variable.

Offset	Description
0	Hours (0-23) ¹
1	Minutes (0-59) ¹
2	Seconds (0-59) ¹

Offset	Description
3	Hundredths of seconds (0-99) ¹
4	Day (1-31) ¹
5	Month (1-12) ¹
6-7	Year (for example, 1995) ²
8-9	Time zone offset (in minutes) ^{2, 3}
10	Day of the week, where 0 is Sunday (0-6) ¹
Notes:	
¹	A 1-byte integer.
²	A 2-byte integer.
³	» This is the negative value of the offset associated with the specified output time zone. If *UTC is specified for the output time zone, then this value will be 0. If an output time zone is not specified, then this is the negative value of the system value QUTCOFFSET. «



Time Zone Information Value Structure

This table shows the structure used for the time zone information output parameter. If *UTC is specified for the output time zone, or if the input and output time zone parameter values are the same and the input variable contains a date that is outside the supported date range (from August 25, 1928, 00:00:00.000000 to May 09, 2071, 00:00:00.000000), then all binary fields will be set to 0 and all character fields will be set to blanks.

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	CHAR(10)	Time zone description name
18	12	CHAR(1)	Reserved
19	13	CHAR(1)	Current Daylight Saving Time indicator
20	14	BINARY(4)	Current offset
24	18	CHAR(50)	Current full name
74	4A	CHAR(10)	Current abbreviated name
84	54	CHAR(7)	Current message identifier
91	5B	CHAR(10)	Message file name
101	65	CHAR(10)	Message file library

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.

Current abbreviated name. The abbreviated, or short, name for the time zone. This field will contain either the Standard Time or Daylight Saving Time abbreviated name depending on whether or not Daylight Saving Time is in effect. If the time zone description uses a message to specify the current abbreviated name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the message file.

Current Daylight Saving Time indicator. Indicates whether or not the output date and time (output variable converted based on the output time zone) is observing Daylight Saving Time or not. Valid values that are returned are:

0	Daylight Saving Time is not being observed (Standard Time).
1	Daylight Saving Time is being observed.

Current full name. The full, or long, name for the time zone. This field will contain either the Standard Time or Daylight Saving Time full name depending on whether or not Daylight Saving Time is in effect. If the time zone description uses a message to specify the current full name and the message cannot be retrieved, this field returns *N. This can occur when the caller of the API is not authorized to the message file or its library, the message file cannot be found or the message does not exist in the message file.

Current message identifier. The identifier of the message that contains the current full and abbreviated names. This field will be *NONE if a message was not specified when the time zone description was created.

Current offset. The time difference, in minutes, between the output time zone and Coordinated Universal Time (UTC). This value has been adjusted for Daylight Saving Time, if necessary.

Message file library. The name of the library that contains the message file. The field will contain all blanks if the current message identifier is *NONE.

Message file name. The name of the file that contains the current message. The field will contain *NONE if the current message identifier is *NONE.

Reserved. An unused field.

Time zone description name. The name of the time zone description that is associated with the output time zone. If *SYS or *JOB was specified for the output time zone and a time zone has not been set for the Time zone (QTIMZON) system value, this field returns *N.

Usage Notes

When converting an input date from a 2-digit year format to a *DTS time-stamp format without time zone conversion, the supported date range is from August 23, 1928, 12:03:06.314752 (.315 for milliseconds) to May 10, 2071, 11:56:53.685240 (.685 for milliseconds). Converting an input date that is outside this range will result in an output date within this range.

When converting an input date from a 4-digit year format to a *DTS time-stamp format without time zone conversion, the supported date range is from August 24, 1928, 00:00:00.000000 to May 09, 2071, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from a 4-digit year format to a 2-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from a 4-digit year format to a 4-digit year format without time zone conversion, the supported date range is from January 1, 0001, 00:00:00.000000 to December 31, 9999, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When converting an input date from 2-digit year format to a 2-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds). The century digit of the input variable is copied into the output variable without validation.

When converting an input date from 2-digit year format to a 4-digit year format without time zone conversion, the supported date range is from January 1, 1900, 00:00:00.000000 to December 31, 2899, 23:59:59.999999 (.999 for milliseconds).

When converting an input date from a *DTS time-stamp format to an output date of any format without time zone conversion, the supported date range is from August 23, 1928, 12:03:06.314752 (.315 for milliseconds) to May 10, 2071, 11:56:53.685240 (.685 for milliseconds).

When converting an input date of any format to an output date of any format that involves time zone conversion as well, the supported date range is from August 25, 1928, 00:00:00.000000 to May 08, 2071, 23:59:59.999999 (.999 for milliseconds). Converting an input date that is outside this range will result in error message CPF1060.

When moving from Standard Time (ST) to Daylight Saving Time (DST) there is a window of time (1 hour) that does not occur. Any time zone conversion where the input variable date and time value is within this window will result in error message CPF1060.

When moving from Daylight Saving Time (DST) to Standard Time there is a window of time (1 hour) that repeats. For example, if DST ends on a given day at 02:00AM, then the segment of time from 01:00:00.000000 to 01:59:59.999999 on that day repeats. The first segment of repeated time is the DST segment. The second segment of repeated time is the Standard Time segment. It is possible using time zone conversion to have the output variable date and time value end up in either segment. If you are retrieving time zone information, the current Daylight Saving Time indicator will be set accordingly. By default, for any time zone conversion the input variable that is within this window of time that repeats is considered part of the DST segment. However, you can use the optional Input time indicator parameter to cause the input variable to be considered within the Standard Time segment. You can copy the resultant current DST indicator into the Input time indicator parameter when converting back and forth between time zones. For example, when converting a date and time value from *UTC to time zone A, the resultant time is 01:15:00 AM and the current DST indicator returned is 0, which means the resultant time is Standard Time. In order to obtain the original *UTC value when converting back to *UTC from time zone A, the current DST indicator value should be copied to the Input time indicator parameter. This will cause the date and time value to be treated as Standard Time rather than as the default, Daylight Saving Time.

You can convert any input format except *CURRENT to the same output format without receiving an error (time zone conversion is not specified, or if specified, the input and output time zone parameter values must be the same and the time zone information length must be 0 as well). For these cases, the input variable is copied into the output variable without validation.

When converting one character date format (that is, anything other than *CURRENT, the current machine-clock time, *DTS, the system time-stamp, or any specified time zone conversion) to another character date format, the date information is validated and converted. However, the time portion of the input variable is copied into the output variable without validation.

When requesting time zone conversion with different input and output time zone values, or when requesting time zone information, the time portion is validated and converted as well as the date portion.

When converting a character date and time value to *DTS and back to character format using microseconds precision, there is a rounding error of minus 1 to minus 7 microseconds. If you specify a precision of microseconds, it is recommended that you use a microsecond value that is evenly divisible by 8.



Error Messages

Message ID	Error Message Text
CPF1060 E	Date not valid.
CPF1061 E	Time not valid.
CPF1848 E	Century digit &1 not valid.
CPF1849 E	Millisecond » or microsecond value « not valid.
CPF1850 E	Format &1 not valid
CPF24B4 E	Severe error while addressing parameter list.
» CPF3C36 E	Number of parameters, &1, for API not valid. «
» CPF3C3C E	Value for parameter &1 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: V2R1

Top | "Miscellaneous APIs," on page 1 | APIs by category

Generate Pseudorandom Numbers (QC3GENRN,Qc3GenPRNs) API

Required Parameter Group:	
1	PRN data
Output	Char(*)
2	PRN data length
Input	Binary(4)
3	PRN type
Input	Char(1)
4	PRN Parity
Input	Char(1)
5	Error code
I/O	Char(*)
Service Program Name: QC3PRNG	
Default Public Authority: *USE	
Threadsafe: Yes	

The Generate Pseudorandom Numbers [»](#) (OPM, QC3GENRN; ILE, Qc3GenPRNs) [«](#) API generates a pseudorandom binary stream.

The pseudorandom number generator is composed of two parts: pseudorandom number generation and seed management. Pseudorandom number generation is performed using the FIPS 186-1 algorithm. Cryptographically-secure pseudorandom numbers rely on good seed. The FIPS 186-1 key and seed values are obtained from the system seed digest. The server automatically generates seed using data collected from system information or by using the random number generator function on a cryptographic coprocessor, such as a 4758, if one is available. System-generated seed can never be truly unpredictable. If a cryptographic coprocessor is not available, you can use the Add Seed for PRNG (Qc3AddPRNGSeed) API to add your own random seed to the system seed digest. This should be done as soon as possible any time the Licensed Internal Code is installed.

Authorities and Locks

None.

Required Parameter Group

PRN data

OUTPUT; CHAR(*)

The generated pseudorandom binary stream.

PRN data length

INPUT; BINARY(4)

The number of pseudorandom number bytes to return in the PRN data parameter. If 0 is specified, no pseudorandom numbers are returned.

PRN type

INPUT; CHAR(1)

The API can generate a real pseudorandom binary stream or a test binary stream.

The FIPS 186-1 algorithm obtains the initial key and seed values from the system seed digest when generating a real pseudorandom binary stream. When generating a test binary stream, the algorithm uses preset values for the key and seed. Valid values are:

- 0 Generate real pseudorandom numbers.
- 1 Generate test pseudorandom numbers.

PRN Parity

INPUT; CHAR(1)

The API sets each byte of the pseudorandom number binary stream to the specified parity by altering the low order bit in each byte as necessary. Valid values are:

- 0 Do not set parity.
- 1 Set each byte to odd parity.
- 2 Set each byte to even parity.

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
CPF3C19 E	Error occurred with receiver variable specified.
CPF3CF1 E	Error code parameter not valid.
CPFBAF1 E	PRN type not valid.
CPFBAF2 E	Parity not valid.
CPFBAF3 E	The system seed digest is not ready.

API introduced: V5R1

Top | "Miscellaneous APIs," on page 1 | APIs by category

Remove All Bookmarks from a Course (QEARMVBM) API

Required Parameter Group:	
1	Course ID
Input	Char(10)
2	Error code
I/O	Char(*)
	Default Public Authority: *USE
Threadsafe: No	

The Remove All Bookmarks from a Course (QEARMVBM) API removes all bookmarks from a Tutorial System Support course. This API provides support similar to option 9 (Remove bookmarks) on the Work with Courses display within the Start Education (STREDU) command.

Authority

The user must be an education administrator and have one of the following authorities:

Authority

*ALLOBJ or *SECADM

Required Parameter Group

Course ID

INPUT; CHAR(10)

The ID of the course that is to have all bookmarks removed.

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
CPF1D50 E	Not authorized to remove bookmarks.
CPF1D51 E	Not all bookmarks removed.

Message ID	Error Message Text
CPF1D52 E	Course not found.
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: V2R2

[Top](#) | [“Miscellaneous APIs,” on page 1](#) | [APIs by category](#)

Retrieve Data (QPARTVDA) API

Required Parameter Group:	
1	Receiver variable
Output	Char(*)
2	Length of receiver variable
Input	Binary(4)
3	Actual length of user data
Output	Binary(4)
4	Error code
I/O	Char(*)
	Default Public Authority: *USE
Threadsafe: No	

The Retrieve Data (QPARTVDA) API retrieves up to 1KB of user data, which was passed to this system with the Start Pass-through (QPASTRPT) API.

Required Parameter Group

Receiver variable

OUTPUT; CHAR(*)

User data associated with a pass-through session. You can specify the size of the area to be smaller than the data sent by the source system as long as you specify the length parameter correctly. As a result, the API returns only the data the area can hold.

Length of receiver variable

INPUT; BINARY(4)

The length of the receiver variable. If the length is larger than the size of the receiver variable, the results are not predictable.

Actual length of user data

OUTPUT; BINARY(4)

The actual length of user data associated with this pass-through session. If this value is greater than the length of receiver variable parameter, then truncation occurred.

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.
CPF3CF1 E	Error code parameter not valid.
CPF3C19 E	Error occurred with receiver variable specified.
CPF3C24 E	Length of the receiver variable is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF8942 E	Command or API call not allowed on source system.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V3R6

Top | "Miscellaneous APIs," on page 1 | APIs by category

Start Pass-Through (QPASTRPT) API

Required Parameter Group:	
1	Pass-through information
Input	Char(*)
2	Length of pass-through information
Input	Binary(4)
3	Format name
Input	Char(8)
4	Data
Input	Char(*)
5	Length of data
Input	Binary(4)
6	Error code
I/O	Char(*)
	Default Public Authority: *USE
Threadsafe: No	

The Start Pass-Through (QPASTRPT) API starts a 5250 pass-through session and optionally passes up to 1KB of user data from the source system to the target system. This data can be accessed on the target system with the Retrieve Data (QPARTVDA) API.

Authorities and Locks

APPC Device on Source System

*CHANGE

APPC Device on Target System

*CHANGE

Virtual Controller on Target System

*USE

Virtual Device on Target System

*CHANGE

Program Specified in QRMTSIGN System Value on Target System

*USE

Required Parameter Group

Pass-through information

INPUT; CHAR(*)

Information associated with establishing the 5250 pass-through session.

Length of pass-through information

INPUT; BINARY(4)

The length, in bytes, of the pass-through information parameter. This value must be greater than or equal to 8 and less than or equal to 580.

Format name

INPUT; CHAR(8)

The format of the pass-through information. The supported format names are:

PAST0100 Pass-through with up to 10-byte password

PAST0200 Pass-through with up to 128-byte password

See “PAST0100 Format” and “PAST0200 Format” on page 24 for details.

Data INPUT; CHAR(*)

User-defined data to be passed to the target system. The format of this data is not defined by the API, and is sent to the target system as is.

Length of data

INPUT; BINARY(4)

The length of the data parameter.

Error code

I/O; CHAR(*)

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

PAST0100 Format

The following table is the layout of the pass-through information for format PAST0100, which controls how the pass-through session is established for a pass-through with up to a 10-byte password.

Offset		Type	Field
Dec	Hex		
0	0	CHAR(8)	Remote location name
8	8	CHAR(10)	Virtual controller
18	12	CHAR(8)	Mode name
26	1A	CHAR(8)	Local location name
34	22	CHAR(8)	Remote network ID
42	2C	CHAR(10)	System request program name

Offset		Type	Field
Dec	Hex		
52	34	CHAR(10)	System request library name
62	3E	CHAR(10)	Remote user ID
72	48	CHAR(10)	Remote password
82	52	CHAR(10)	Initial program
92	5C	CHAR(10)	Initial menu
102	66	CHAR(10)	Current library
112	70	CHAR(1)	Display option
113	71	CHAR(3)	Reserved
116	74	BINARY(4)	Offset to virtual devices
120	78	BINARY(4)	Number of virtual devices
		CHAR(*)	Array of virtual devices

PAST0200 Format

The following table is the layout of the pass-through information for format PAST0200, which controls how the pass-through session is established for a pass-through with up to a 128-byte password.

Offset		Type	Field
Dec	Hex		
0	0	CHAR(8)	Remote location name
8	8	CHAR(10)	Virtual controller
18	12	CHAR(8)	Mode name
26	1A	CHAR(8)	Local location name
34	22	CHAR(8)	Remote network ID
42	2C	CHAR(10)	System request program name
52	34	CHAR(10)	System request library name
62	3E	CHAR(10)	Remote user ID
72	48	CHAR(10)	Reserved
82	52	CHAR(10)	Initial program
92	5C	CHAR(10)	Initial menu
102	66	CHAR(10)	Current library
112	70	CHAR(1)	Display option
113	71	CHAR(3)	Reserved
116	74	BINARY(4)	Offset to virtual devices
120	78	BINARY(4)	Number of virtual devices
124	7C	BINARY(4)	Offset to remote password
128	80	BINARY(4)	Phrase length of remote password
		CHAR(*)	Remote password
		CHAR(*)	Array of virtual devices

Field Descriptions

Array of virtual devices. An array of 0 through 32 virtual devices on the target system. A device on the target system is selected from this list based on a comparison of device type and model.

Current library. The library to be the current library on the target system. The special value *RMTUSRPRF can be used to indicate that the current library found in the remote user profile should be used.

Display option. Whether the pass-through and associated status messages appear. Special values follow:

0 Do not display pass-through information.
1 Display pass-through information.

Initial menu. The menu that is initially shown at the target system. This runs after the initial program. Special values follow:

*RMTUSRPRF Show the initial menu as specified by the remote user profile.
*SIGNOFF Sign off the target system after running the initial program.

Initial program. The program that is called immediately after sign-on to the target system. Special values follow:

*RMTUSRPRF Call the initial program as specified by the remote user profile.
*NONE There is no initial program to call.

Local location name. The name by which the local iSeries server is known to other devices in the network. Special values follow:

*LOC The local location name is chosen by the system.
*NETATR The local location name, a system network attribute, is used.

Mode name. The mode to be used. The special value *NETATR can be used to indicate that the system network attribute mode should be used.

Number of virtual devices. The number of virtual devices in the array of virtual devices. If the virtual controller field is not *NONE, this field must be set to 0.

Offset to remote password. The offset from the beginning of the format to the start of the remote password. The phrase length of the remote password field must be a valid value.

Offset to virtual devices. The offset from the beginning of the format to the start of the array of virtual devices. If the virtual controller field is not *NONE, this field must be set to 0.

Phrase length of remote password. The length, in bytes, of the remote password. This value must be greater than 0 and less than or equal to 128.

Remote location name. The name of the location that is the target of the pass-through session.

Remote network ID. The network ID of the network where the remote location resides. Special values follow:

*LOC Any remote location name will be used.
*NETATR The local network ID, a system network attribute, is used.

*NONE The remote system does not support network IDs.

Remote password. The password being sent to the target system. The special value allowed is *NONE. If a profile is specified for the remote user ID field and password security is active on the target system, *NONE is not allowed.

Remote user ID. The user profile for automatic sign-on to the target system. Special values follow:

*NONE No user ID is passed to the target system; automatic sign-on is not used.
*CURRENT The user ID that is active in the current job is passed to the remote system.

Reserved. An ignored field. This field must be blanks.

System request library name. The library in which the system request program can be found. Special values are *LIBL and *CURLIB. If the system request program is *SRQMNU, this field must be set to blanks.

System request program name. The program that is to be called on the source system when system request option 10 is selected. The special value *SRQMNU causes the system-supplied system request menu to be displayed.

Virtual controller. The name of the virtual controller on the target system that is used to do pass-through. If you specify a virtual controller, one of the virtual display devices attached to it is selected for the pass-through job. This entry is mutually exclusive of the array of virtual devices, and must be *NONE if the number of virtual devices field is not 0. The default is *NONE.

Error Messages

Message ID	Error Message Text
CPF24B4 E	Severe error while addressing parameter list.
CPF2702 E	Device description &1 not found.
CPF2703 E	Controller description &1 not found.
CPF3CF1 E	Error code parameter not valid.
CPF3C1D E	Length specified in parameter &1 not valid.
CPF3C21 E	Format name &1 is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF5383 E	Mode &7 specified for device &4 not valid.
CPF5536 E	System cannot automatically select an APPC device description for the remote location.
CPF5546 E	Class-of-service for device &4 not valid.
CPF8901 E	Virtual device &1 not varied on.
CPF8902 E	Virtual device &1 not available.
CPF8904 E	Pass-through request not accepted.
CPF8905 E	Pass-through not allowed on this system.
CPF8906 E	Error during session initialization. Reason code &1.
CPF8907 E	Communications failure for device &1.
CPF8908 E	Controller &1 not varied on.
CPF8911 E	Communications failure. Session was not started.
CPF8912 E	Pass-through session ended. Reason code &1.
CPF8913 E	Pass-through ended abnormally.
CPF8916 E	Cannot select virtual device &1 at system &2.
CPF8918 E	Job canceled at system &1.
CPF8919 E	Device &1 not accessed by system &2.
CPF8920 E	Pass-through failed. &1 must be varied off and on.
CPF8921 E	APPC failure. Failure code is &3.

Message ID	Error Message Text
CPF8922 E	Negative response from device &1 at system &2.
CPF8935 E	Pass-through not allowed to system &1.
CPF8936 E	Pass-through failed for security reasons.
CPF8937 E	Automatic sign on not allowed.
CPF8939 E	Trying to send too much data.
CPF8944 E	Device &1 no longer communicating with system &2.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.

API introduced: V3R6

Top | “Miscellaneous APIs,” on page 1 | APIs by category

Update Device Microcode (QTAUPDDV) API

Required Parameter Group:	
1	Device name
Input	Char(10)
2	Source path name
Input	Char(*)
Optional Parameter Group:	
3	Length of source path name
Input	Binary(4)
4	Format Name
Input	Char(8)
5	Error code
I/O	Char(*)
	Default Public Authority: *EXCLUDE
Threadsafe: No	

The Update Device Microcode (QTAUPDDV) API provides an interface for updating device microcode from a code image stored in an Integrated File System (IFS) stream file.

The QTAUPDDV API only supports self-configuring, stand alone tape devices. The Retrieve Device Capabilities (QTARDCAP) API can be used to determine if the tape device is a self-configured tape device. Other types of devices are currently not supported by this API. This API can be used for a tape device within a media library if the device is deallocated from the library.

Note: Incorrect use of this API can cause damage to the tape device.

Authorities and Locks

API Public Authority
*EXCLUDE

Special Authority
*SERVICE

Directory Access Authority
*X

Stream File Access Authority
*R

Device Description Authority
*CHANGE

Device Description Lock
*EXCLRD

Required Parameter Group

Device name

INPUT; CHAR(10)

The name of the device for which for which the code image is to be updated.

Source path name.

INPUT; CHAR(*)

The path name for the code image.

Optional Parameter Group

Length of source path name.

INPUT; BINARY(4)

The length of the source path name provided. Valid values range from 1 through 32048 or -1. If this parameter is omitted, the source path name is assumed to be a simple blank terminated path name. Use -1 if the length is in the LG-type structure.

Format name

INPUT; CHAR(8)

The format of the source path name parameter. If this parameter is omitted the source path name is assumed to be a simple blank terminated path name in the current CCSID.

TAUD0100

The source path name is a simple path name in the current CCSID.

TAUD0200

The source path name is a LG-type path name structure. For more information on this structure, see Path name format.

Error code

I/O; CHAR(*)

The structure in which to return error information. For the format of the structure, see Error Code Parameter. If this parameter is omitted, any problems will result in escape messages to the application.

Error Messages

Message ID	Error Message Text
CPFA0A9 E	Object not found. Object is *.
CPFA09C E	Not authorized to object.

Message ID	Error Message Text
CPF222E E	* special authority is required.
CPF24B4 E	Severe error while addressing parameter list.
CPF3C1D E	Length specified in parameter * not valid.
CPF3C21 E	Format name * is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF3CF1 E	Error code parameter not valid.
CPF6B35 E	File member is in use.
CPF6708 E	Command ended due to error.
CPF671F E	Parameter list not correct.
CPF67C2 E	Path name structure field * not valid.
CPF67C8 E	Command failed for device *. Reason code *.
CPF8068 E	Error detected while processing file.
CPF9802 E	Not authorized to object * in *.
CPF9814 E	Device &1 not found.



API introduced: V5R3 with PTF

Top | “Miscellaneous APIs,” on page 1 | APIs by category

Using the WebFacing Environment API (QqfEnvironment)

The WebFacing Environment (QqfEnvironment) API enables you to check whether a user is accessing your application through a Web browser or through 5250 emulation. Use this API when you would like to change the behavior of your program according to the type of access a user has. For example, there may be an extra field or different text that you would like to display if your program is being accessed through a browser, but you would like to suppress display of the field or text if 5250 emulation is being used.

The WebFacing Environment API is called QqfEnvironment and is part of the WebFacing server runtime. The external procedure name QqfEnvironment is case sensitive. It is a procedure packaged in a service program called QQFENV that is located in the QSYS library. The API returns 1 if the application is running under WebFacing and 0 if it is running under 5250 emulation.

Examples

The following examples show how to use this API. In the RPG sample, the external procedure QqfEnvironment is defined with the DSpec QQFENV. In this example, the QQFENV DSpec has been given the same name as the service program and has been defined as an integer since the procedure returns 0 or 1. A DSpec rc has also been defined to hold the value 0 or 1 when the Eval rc = QQFENV is performed. The RPG program then uses the value of rc to conditionally determine the behaviour of the program and what will get displayed on the DDS display.

In the DDS sample below, if the value for rc in the RPG module is NOT 1, the text “Application is not running in the Webfacing environment” will be displayed. If the value for rc is 1, the text “Application is running in the Webfacing environment” will be displayed.

When you are creating a program to use this API:

1. Use the CRTRPGMOD command to create a module with your RPG code that is calling the API. An RPG module needs to be created because it is using a procedure not in the program.
2. When you create your program (CRTPGM) use the BNDSRVPGM keyword to bind your RPG module with the QQFENV service program in QSYS.

See Code disclaimer information for information pertaining to code examples.

Table 1. RPGLE sample

```

.....1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7.....+.....8
FCHKENVFM CF E Workstn
*
DQQFENV PR 10I 0 Extproc('QqfEnvironment')
*
Drc S 10I 0
*
C Eval rc = QQFENV
C Eval FLD001 = rc
*
C Dow NOT *IN03
*
C If rc = 1
C Eval *in01 = *on
*
C Else
C Eval *IN01 = *off
C EndIf
*
C Exfmt FMT01
C EndDo
*
C Eval *inlr = *on

```

Table 2. DDS Sample

```

.....+A*..1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7.....+.....8
A*%TS SD 20010924 150104 USERID REL-V4R4M0 5769-PW1
A*%EC
A DSPSIZ(24 80 *DS3)
A R FMT01
A*%TS SD 20010924 150104 USERID REL-V4R4M0 5769-PW1
A CA03(03)
A 1 24'Testing Webfacing Environment API'
A DSPATR(HI)
A 10 3'F3=Exit'
A COLOR(BLU)
A N01 5 13'Application is not running in the -
A Webfacing environment'
A 01 6 15'Application is running in the Webf-
A acing environment'
A COLOR(RED)
A FLD001 4S 00 7 39
A 7 22'QqfEnvironment:'
A
A*%GP SCREEN1 01

```

API introduced: V5R1

[Top](#) | [“Miscellaneous APIs,” on page 1](#) | [APIs by category](#)

User Application APIs

The user application APIs are:

- [»](#) [“Remove User Application Information \(QsyRemoveUserApplicationInfo\) API” on page 31](#) (QsyRemoveUserApplicationInfo) removes the specified application information from the specified user profile. [«](#)
- [»](#) [“Retrieve User Application Information \(QsyRetrieveUserApplicationInfo\) API” on page 32](#) (QsyRetrieveUserApplicationInfo) retrieves the application information for a user profile. [«](#)

- [»](#) “Update User Application Information (QsyUpdateUserApplicationInfo) API” on page 36 (QsyUpdateUserApplicationInfo) updates the specified application information for a user profile. [«](#)



Top | “Miscellaneous APIs,” on page 1 | APIs by category

Remove User Application Information (QsyRemoveUserApplicationInfo) API

Syntax for QsyRemoveUserApplicationInfo:

```
#include <qsyusrin.h>

void QsyRemoveUserApplicationInfo
    (char      *User_profile,
     char      *Application_information_ID,
     int       *Length_of_application_information_ID,
     void      *Error_code);
```

Service Program: QSYUSRIN

Default Public Authority: *USE

Threadsafe: No

The Remove User Application Information (QsyRemoveUserApplicationInfo) API removes the specified application information from the specified user profile.

The Change User Profile exit programs are not called from this API.

Authorities and Locks

If the user profile parameter is not *CURRENT or the user profile currently running, then the user profile that calls this API must have *SECADM special authority and *OBJMGT and *USE authorities to the user profile.

Required Parameter Group

User profile

INPUT; CHAR(10)

The user profile for which the application information will be removed. The special value *CURRENT may be specified to remove application information for the user profile that calls this API.

Application information ID

INPUT; CHAR(*)

The ID for the application information to be removed. The following can be specified for the application information ID:

generic*

All application information IDs that have IDs beginning with the generic string will be removed.

application information ID

Specific application information ID will be removed.

Length of application information ID

INPUT; BINARY(4)

The length of the application information ID that is specified in the application information ID parameter. The length of the application information ID must be a value from 1 to 200.

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

CPF2204 E	User profile &1 not found.
CPF2213 E	Not able to allocate user profile &1.
CPF226C E	Not authorized to perform function.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF3C1D E	Length specified in parameter &1 not valid.
CPF3C90 E	Literal value cannot be changed.
CPF4AA2 E	Application information ID &1 does not exist.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.



API introduced: V5R3

[Top](#) | [Security APIs](#) | [APIs by category](#)

Retrieve User Application Information (QsyRetrieveUserApplicationInfo) API

Syntax for QsyRetrieveUserApplicationInfo:

```
#include <qsyusrin.h>

void QsyRetrieveUserApplicationInfo
    (void          *Receiver_variable,
     int           *Length_of_receiver_variable,
     void          *Return_records_feedback_information,
     char          *Format_name,
     char          *User_profile,
     char          *Application_information_ID,
     int           *Length_of_application_information_ID,
     void          *Error_code);
```

Service Program: QSYUSRIN

Default Public Authority: *USE

Threadsafe: No

The Retrieve User Application Information (QsyRetrieveUserApplicationInfo) API returns a list of application information entries for a user profile.

Authorities and Locks

User Profile Authority
*READ

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.

Returned records feedback information

OUTPUT; CHAR(12)

Information about the entries that are returned in the receiver variable.

See “Format of Returned Records Feedback Information” on page 34 for details.

Format name

INPUT; CHAR(8)

The name of the format that is used to retrieve application information entries for the user profile.

You can specify this format:

RUAI0100 For a detailed description of this format, see “RUAI0100 Format” on page 34.

User profile

INPUT; CHAR(10)

The name of the user profile for which the application information will be retrieved. The special value *CURRENT may be specified to retrieve application information for the user profile that calls this API.

Application information ID

INPUT; CHAR(*)

The ID for the application information to retrieve. The following can be specified for the application information ID:

generic*

All application information IDs that have IDs beginning with the generic string will be retrieved.

application information ID

Specific application information ID will be retrieved.

Length of application information ID

INPUT; BINARY(4)

The length of the application information ID. The length of the application information ID may be from 1 to 200.

Error code

I/O; CHAR(*)

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

Receiver Variable Description

The following tables describe the order and format of the data returned in the receiver variable. For detailed descriptions of the fields in the tables, see "Field Descriptions."

RUAI0100 Format

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Length of entry
4	4	CHAR(200)	Application information ID
204	CC	BINARY(4)	Displacement to user application information
208	D0	BINARY(4)	Length of user application information
212	D4	BINARY(4)	CCSID of user application information
216	D8	CHAR(6)	First valid release
		CHAR(*)	User application information

Format of Returned Records Feedback Information

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Bytes returned
4	4	BINARY(4)	Bytes available
8	8	BINARY(4)	Number of user application information entries

Field Descriptions

Application information ID. The application information ID that identifies this entry.

Bytes available. The number of bytes of data available to be returned to the user in the receiver variable. If all data is returned, bytes available is the same as the number of bytes returned. If the receiver variable was not large enough to contain all of the data, this value is estimated based on the total number of application information entries for the user profile and the format specified.

Bytes returned. The number of bytes of data returned to the user in the receiver variable. This is the lesser of the number of bytes available to be returned or the length of the receiver variable.

CCSID of user application information. The CCSID of the user application information. This will be the default job CCSID of the job that last updated the user application information.

Displacement to user application information. The displacement in the entry to the start of the user application information.

First valid release. The first release that this application information is valid. This field will be in the format VxRxMx (for example, V5R3M0).

Length of entry. The length (in bytes) of the current entry. This length can be used to access the next entry.

Length of user application information. The length (in bytes) of the user application information.

Number of user application information entries. The number of complete entries returned in the list of user application information entries. A value of zero is returned if the list is empty.

User application information. The user application information that is associated with the user profile.

Error Messages

Message ID	Error Message Text
------------	--------------------

CPFA0AA E	Error occurred while attempting to obtain space.
CPF2204 E	User profile &1 not found.
CPF2213 E	Not able to allocate user profile &1.
CPF2217 E	Not authorized to user profile &1.
CPF2222 E	Storage limit is greater than specified for user profile &1.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF3C1D E	Length specified in parameter &1 not valid.
CPF3C21 E	Format name &1 is not valid.
CPF3C90 E	Literal value cannot be changed.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.



API introduced: V5R3

[Top](#) | [Security APIs](#) | [APIs by category](#)

Update User Application Information (QsyUpdateUserApplicationInfo) API

Syntax for QsyUpdateUserApplicationInfo:

```
#include <qsyusrin.h>

void QsyUpdateUserApplicationInfo
(char          *User_profile,
 char          *Application_information_ID,
 int          *Length_of_application_information_ID,
 char          *Application_information,
 int          *Length_of_application_information,
 char          *First_valid_release,
 void         *Error_code);
```

Service Program: QSYUSRIN

Default Public Authority: *USE

Threadsafe: No

The Update User Application Information (QsyUpdateUserApplicationInfo) API updates the specified application information for a user profile. The specified information is stored in an object that is saved and restored with the user profile.

The Change User Profile exit programs are not called from this API.

Authorities and Locks

If the user profile parameter is not *CURRENT or the user profile currently running, then the user profile that calls this API must have *SECADM special authority and *OBJMGT and *USE authorities to the user profile.

Required Parameter Group

User profile

INPUT; CHAR(10)

The user profile for which the application information will be updated. The special value *CURRENT may be specified to update application information for the user profile that calls this API.

Application information ID

INPUT; CHAR(*)

The ID for the application information entry to update. IBM-supplied AS/400 application information IDs are named QIBM_ccc_name, where ccc is the component identifier. User-supplied application information IDs should not preface their application information ID with QIBM. User-supplied application information IDs should start with the company name to eliminate most problems that involve unique names. Application information IDs should use an underscore (_) to separate parts of the name. Also, IDs for related applications should start with the same name.

The first character of the application information ID must be one of the following:

A-Z

Uppercase A-Z

The remaining characters in the application information ID must be made up of the following characters:

A-Z	Uppercase A-Z
0-9	Digits 0-9
.	Period
_	Underscore

Length of application information ID

INPUT; BINARY(4)

The length of the application information ID that is specified in the application information ID parameter. The length of the application information ID must be a value from 1 to 200.

Application information

INPUT; CHAR(*)

The application information to be associated with the specified user profile.

Length of application information

INPUT; BINARY(4)

The length of the application information that is specified in the application information parameter. The length of the application information must be a value from 1 to 1700.

First valid release

INPUT; CHAR(6)

The first release that this application information is valid. This field is used to determine the earliest release this user application information is valid when saving a user profile to a previous release. If the user profile is saved to a release previous to the release specified in this field, this information will not be saved with the user profile information. This field must be in the format VxRxMx (for example, V5R3M0). The release specified must be V5R3M0 or greater.

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
CPF2204 E	User profile &1 not found.
CPF2213 E	Not able to allocate user profile &1.
CPF2222 E	Storage limit is greater than specified for user profile &1.
CPF226C E	Not authorized to perform function.
CPF3CF1 E	Error code parameter not valid.
CPF3CF2 E	Error(s) occurred during running of &1 API.
CPF3C1D E	Length specified in parameter &1 not valid.
CPF3C90 E	Literal value cannot be changed.
CPF4AA0 E	Application information ID &1 not valid.
CPF4AA1 E	First release value &1 not valid.
CPF9872 E	Program or service program &1 in library &2 ended. Reason code &3.



API introduced: V5R3

Exit Programs

These are the Exit Programs for this category.

Device Selection Exit Program

Required Parameter Group:	
1	Format name
Input	Char(8)
2	Device selection information
Input/Output	Char(*)
3	Return code
Output	Binary(4)
	Exit point name: QIBM_QPA_DEVSEL
	Exit point format name: PADS0100
	QSYSINC Member Name: EPADSEL

The Device Selection exit program provides an interface to control virtual device selection and automatic creation used by the system for connection requests from clients using virtual device support. The interface allows the user to write an exit program to specify the naming conventions used for automatically created virtual devices and virtual controllers and to specify the automatic creation limit to be used for the specific request.

The exit program can:

1. Reject a specific name for a device connection request.
2. Specify a naming pattern to be used for the automatic creation of a virtual device. This is used only if a specific device name was not requested on the client's connection request.
3. Specify the naming pattern of the virtual controller to be used:
 - to search in an attempt to select an existing device (if a specific device name was not requested on the client's connection request) or
 - to specify a controller to which to attach the automatically created device.This naming pattern is also used to 'count' the number of existing devices toward the automatic creation limit.
4. Specify a second controller naming pattern to be used to 'count' the number of existing virtual devices.
5. Specify the number of devices that can exist on the virtual controllers whose naming pattern is specified.

Authorities and Locks

You must have *ALLOBJ authority to register an exit program for the QIBM_QPA_DEVSEL exit point.

Required Parameter Group

Format name

INPUT; CHAR(8)

The format of the information provided in the Device selection information parameter. The format name is "PDSC0100 Format."

Device selection information

INPUT/OUTPUT; CHAR(*)

The structure containing the data that is being passed to the exit program and that is returned from the exit program.

Return code

OUTPUT; BINARY(4)

Whether to allow the connection request to continue. The possible values are:

- 0 >> Allow connection request. <<
- 1 >> Do not allow connection request. <<

If any other value is returned, the request for selection and automatic creation of a device description for the client request will be processed using the system defaults for the QAUTOVRT system value and the defaults for the device and controller naming conventions.

This parameter is initialized to 0 on the call to the exit program.

PDSC0100 Format

For details about the fields in the following table, see "Field Descriptions" on page 40

Offset		Type	Field
Dec	Hex		
0	0	BINARY(4)	Size of structure
4	4	BINARY(4)	Function
8	8	BINARY(4)	Specific name requested
12	C	CHAR(10)	Name for requested device
22	16	CHAR(8)	Format of returned data

PDSR0100 Format

For details about the fields in the following table, see "Field Descriptions" on page 40

Offset		Type	Field
Dec	Hex		
30	1E	CHAR(2)	Reserved
32	20	BINARY(4)	Autocreation limit
36	24	CHAR(6)	Naming pattern for device
42	2A	CHAR(6)	Controller naming pattern for device attachment
48	30	CHAR(6)	Additional controller naming pattern

Field Descriptions

Additional controller naming pattern. The naming pattern for the controllers to be used for the device automatic creation limit. When applying the check for automatic creation limit, devices attached to these controllers are also counted when determining if the limit is exceeded. This field is initialized to blanks before the call is made to the exit program.

Autocreation limit. The number to be used for the virtual device automatic creation limit for this connection request. Possible values are:

0	Do not allow any additional virtual device descriptions to be created automatically.
1-32500	The number of devices that can be attached to the controller descriptions whose naming patterns are specified in Controller naming pattern for device attachment and Additional controller naming pattern.
32767	The special value of *NOMAX. Do not limit the automatic creation of virtual devices.

Controller naming pattern for device attachment. The naming pattern for the controller to which an automatically created device is to be attached. These characters must be a valid input to the CRTCTLVWS command. If there are not six characters, the pattern is padded with zeros. If the Autocreation limit is 1-32500, the devices attached to controllers with this pattern are counted and this number is used toward the automatic creation limit. This field is initialized to blanks before the call is made to the exit program.

Format of returned data. The format name specified by the user exit program for the output data returned from the Device Selection exit point. The only format supported currently is PDSR0100. This field is initialized to PDSR0100 on the call to the exit program.

Function. The function being used by the client. Possible values are:

1	APPC
2	TELNET
3	Virtual Terminal Manager API (VTM API)

Name for requested device. The name of the device requested by the client. If Specific name requested is set to 0, this field is blank.

Naming pattern for device. The naming pattern to be used for device automatic creation. These characters must be a valid input to the CRTDEVDSR command. This field is checked only if the Specific name requested field is 0. This field is initialized to blanks before the call is made to the exit program.

Reserved. A reserved field that must be set to hexadecimal zeros.

Size of structure. The size of the structure containing the data being passed to and returned from the exit program.

Specific name requested. Whether a specific name was requested by the client. If a specific name was requested, this name will be passed to the exit program in the Name for requested device field.

0	No specific name was requested.
1	Specific name was requested.

Coding Guidelines

Applications should consider the following when coding this exit program:

- The program should return an exception for the requested operation only if there has been a failure in the operation. If the program signals an escape message to the API, the system assumes there is a failure. A diagnostic message is returned to the calling program. The request for selection and automatic creation of a device description for the client request will be processed using the system defaults for the QAUTOVRT system value and the defaults for the device and controller naming conventions.
- The program must clean up any locks that it acquires.
- The program must handle all potential error conditions associated with its own operations (be fault tolerant).
- The program must avoid infinite looping conditions.

Exit program introduced: V5R2

[Top](#) | [“Miscellaneous APIs,” on page 1](#) | [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:

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