Appendix 11A
PrintCore API

Contents

PRINTAbortJob 3

PRINTCloseDevice 4

PRINTCloseJob 5

PRINTEnumDevices 6

PRINTFlushJob 8

PRINTGetAliasWithName 9

PRINTGetDeviceData 10

PRINTGetDeviceStrData 12

PRINTGetJobData 14

PRINTGetJobStrData 16

PRINTOpenDevice 18

PRINTOpenJob 20

PRINTOpenJobWithHandle 21

PRINTSetDeviceData 22

PRINTSetDeviceStrData 24

PRINTSetJobData 26

PRINTSetJobStrData 28

PRINTWriteToJob 30

PRINTAbortJob

Description

Aborts an open print job. Does not flush, but immediately closes the queue or file.

Syntax

```
UINT32
PRINTAbortJob(
    UINT32    pgID,
    UINT32    processID
    UINT32    pJobAliasHandle)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Alias print job handle.

Return values

SUCCESS_CODE
INVALID_JOB_HANDLE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTCloseDevice

Description

Closes a print device.

Syntax

```
UINT32
PRINTCloseDevice(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pDevAliasHandle)
```

Input

pgID	Process group ID of the calling process group.
processID	Process ID of the calling process.
pDevAliasHandle	Alias print device handle

Return values

SUCCESS_CODE

INVALID_PARAMETER	*pgID* or *processID* equal 0 or 0xFFFFFFFF

Remarks

If the caller has sufficient rights (based on *modHandle*, *pgID*, and *processID*) this API closes all the print jobs associated with this print device, deregisters the print device as a system device, and deallocates the internal structures.

PRINTCloseJob

Description

Flushes any buffers associated with a print job and closes the print job.

Syntax

```
UINT32
PRINTCloseJob(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle

Output

None.

Return values

SUCCESS_CODE
INVALID_JOB_HANDLE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTEnumDevices

Description Enumerates all accessible print devices

Syntax

```
UINT32
PRINTEnumDevices(
    UINT32          pgID,
    UINT32          processID,
    UINT32          *sIndex,
    PrintDevInfo *pdinfo)
```

Input

pgID Process group ID to match

processID Process ID of the calling process

Index Address of buffer to hold search index. This must be 0xFFFFFFFF to start and will be modified for subsequent calls to enumerate through all print devices.

Note: This value should not be examined nor should any attempt be made to translate this value in any way. Simply pass in 0xFFFFFFFF for the first search, and continue to pass the unmodified buffer for subsequent interations of the search.

Output

pdinfo Address of buffer to hold print device information in the following format:

```
UINT32      pDevAliasHandle;
UINT32      pgID;
UINT32      processID;
CONN_HANDLE qConnHandle;
UINT8       queueName[MAX_QUEUE_NAME];
```

Return values

SUCCESS_CODE

NO_MORE_ENTRIES	Search is complete
INVALID_PARAMETER	*pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTFlushJob

Description Flushes all buffers associated with a print job.

Syntax

```
UINT32
PRINTFlushJob(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle

Return values

SUCCESS_CODE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTGetDeviceData

Syntax

```
UINT32
PRINTGetDeviceData(
    UINT32    processID,
    UINT32    pDevAliasHandle,
    UINT32    *entryIDList,
    UINT8     *getList)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pDevAliasHandle
Alias print device handle returned from **PRINTOpenDevice**.

entryIDList Zero-terminated array of UINT32s containing any combination of the following typedefs, along with their respective sizes in bytes:

```
GET_DEVICE_DEF_FLAGS,    4
GET_DEVICE_DEF_COPIES,   4
GET_DEVICE_DEF_LINES,    4
GET_DEVICE_DEF_COLS,     4
GET_DEVICE_DEF_TAB,      4
GET_DEVICE_DEF_FORMTYPE, 4
GET_DEVICE_DEF_CONTROL,  2
GET_DEVICE_DEF_SETUP_ALLOCLEN,  4
GET_DEVICE_DEF_RESET_ALLOCLEN,  4
GET_DEVICE_DEF_SETUP_BUFFLEN,  4
GET_DEVICE_DEF_RESET_BUFFLEN,  4
GET_DEVICE_DEF_SETUP_PTR, 4
GET_DEVICE_DEF_RESET_PTR, 4
```

getList Caller-defined structure defined by the size and ordering of the *entryIDList*. If *entryIDList* is defined as:

```
entryIDList[]= {GET_DEVICE_DEF_FLAGS, 4,
                GET_DEVICE_DEF_COPIES,4,
               0};
```

then getList would be a pointer to an array that looks like:

```
typedef getListTag {
UINT32   flags;
UINT32   copies;
} getListStruct;
```

And the call to **PRINTGetDeviceData** would look like:

```
PRINTGetDeviceData(
    modHandle,
    pgID,
    processID,
    pDevAliasHandle,
    entryIDList,
    &getList);
```

Output *getList* Contains the requested data

Return values

SUCCESS_CODE	
INVALID_PDEVICE_HANDLE	
INVALID_PARAMETER	*pgID* or *processID* equal 0 or 0xFFFFFFFF

See also PRINTSetDeviceData

PRINTGetDeviceStrData

Description

Gets data associated with default print device strings.

Syntax

```
UINT32
PRINTGetDeviceStrData(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pDevAliasHandle,
    UINT32    elementNumber,
    UINT8     *buffer,
    UINT32    size,
    UINT32    *bRead)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pDevAliasHandle
Alias print device handle returned from PRINTOpenDevice.

elementNumber
GET_DEVICE_DEF_BANNER
GET_DEVICE_DEF_FORMNAME
GET_DEVICE_DEF_BANNERNAME
GET_DEVICE_DEF_DESCRIPTION
GET_DEVICE_DEF_RESET_STR
GET_DEVICE_DEF_SETUP_STR

size Size in bytes of buffer

Output

buffer Buffer to copy device string field data into

bRead Number of bytes read from device string field. If *bRead* is less than *size*, then the value returned in *bRead* is the size in bytes of the requested string field.

Return values

SUCCESS_CODE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

See also

PRINTSetDeviceStrData, PRINTGetJobStrData, PRINTSetJobStrData

PRINTGetJobData

Description

Retrieves data associated to a print job and places it in the caller-supplied buffer.

Syntax

```
UINT32
PRINTGetJobData(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle,
    UINT32    *entryIDList,
    UINT8     *getList)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle returned from **PRINTOpenJob**.

entryIDList Zero-terminated array of UINT32s containing any combination of the following typedefs along with their respective sizes, in bytes:

GET_JOB_FLAGS, 4
GET_JOB_COPIES, 4
GET_JOB_LINES, 4
GET_JOB_COLS, 4
GET_JOB_TAB, 4
GET_JOB_FORMTYPE, 4
GET_JOB_CONTROL, 2
GET_JOB_TARGETTIME, 6
GET_JOB_SETUP_ALLOCLEN, 4
GET_JOB_RESET_ALLOCLEN, 4
GET_JOB_SETUP_BUFFLEN, 4
GET_JOB_RESET_BUFFLEN, 4
GET_JOB_SETUP_PTR, 4
GET_JOB_RESET_PTR, 4

getList A caller defined structure defined by the size and ordering of the *entryIDList*. If *entryIDLis*t is defined as:

```
entryIDList[] = {GET_JOB_FLAGS, 4L,
        GET_JOB_COPIES, 4L, 0};
```

then *getList* would be a pointer to an array that looks like this:

```
typedef getListTag {
    UINT32flags;
    UINT32copies;
} getListStruct;
```

And the call to **PRINTGetJobData** would look like:

```
PRINTGetDeviceData(modHandle, pgID,
processID, pJobAliasHandle, entryIDList, &getList);
```

Output *getList* Contains requested data

Return values SUCCESS_CODE

INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

See also PRINTSetDeviceData, PRINTGetDeviceData, PRINTSetJobData

PRINTGetJobStrData

Description

Sets data associated to a print job from a caller supplied buffer.

Syntax

```
UINT32
PRINTGetJobStrData(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle,
    UINT32    elementNumber,
    UINT8   *buffer,
    UINT32    size,
    UINT32  *bRead)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle

elementNumber
SET_JOB_BANNER
SET_JOB_FORMNAME
SET_JOB_BANNERNAME
SET_JOB_DESCRIPTION
SET_JOB_RESET_STR
SET_JOB_SETUP_STR

buffer Buffer to copy job string field data into

size Size in bytes of buffer

bRead Buffer that *SetJobStrData* fills with number of bytes read.

Output

bRead The number of bytes read from job string field

Return values

SUCCESS_CODE

INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

See also

PRINTSetJobStrData, PRINTGetDeviceStrData, PRINTSetDeviceStrData

PRINTOpenDevice

Description Opens a print device.

Syntax

```
UINT32
PRINTOpenDevice(
    UINT32      pgID,
    UINT32      processID,
    CONN_HANDLE qConnHandle,
    UINT8       *queueName,
    UINT32      queueID,
    UINT32      *pDevAliasHandle)
```

Input

pgID	Process group ID of the calling process group
processID	Process ID of the calling process
qConnHandle	Connection handle associated with the queue. If servername is included in the queueName, qConnHandle can be zero.
queueName	Name of queue to associate print device. Full UNC path may be specified. (\\SERVERNAME\QUEUENAME).
queueID	Queue ID where print jobs are to be sent. If queueName is passed, queueID can be 0

Output

pDevAliasHandle	Address to return alias device handle

Return values

SUCCESS_CODE	Successful
PRINT_DEVICE_NOT_FOUND	Device was not registered
DEVICE_ALREADY_OPEN	Device already opened
OUT_OF_CLIENT_MEMORY	Out of memory
INVALID_PARAMETER	*pgID*/*processID*=0 or 0xFFFFFFF
INVALID_CONN_HANDLE	Specified connHandle is invalid
INVALID_QUEUE_SPECIFIED	Specified queue does not exist
QUEUE_NAME_ID_MISMATCH	Both queue name and queue ID were specified but do not refer to the same object.

Remarks

If device is already opened, it fills the *pDevAliasHandle* buffer and returns PRINT_DEVICE_ALREADY_OPEN. Otherwise, it registers the print device as a system device and allocates the necessary structures.

PRINTOpenJob

Description

Opens a print job and allocates a print job structure.

Syntax

```
UINT32
PRINTOpenJob(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pDevAliasHandle,
    UINT32    dirHandle,
    UINT8 nameSpace,
    UINT8     *path
    UINT32    actionFlag,
    UINT32    *pJobAliasHandle)
```

Input

pgID Calling process group ID

processID Calling process ID

pDevAliasHandle
Alias print device handle

dirHandle Alias directory handle, 0 if fully specified.

nameSpace Name space type. (See API.H "NAME_SPACE")

path Address of input path and filename, NULL if no file redirection.

actionFlag Bits that match DOS open/create action bits.
0 = Fail, 1 = Create,
bits 1,2 indicate action if exists,
00 = Fail, 01 = Open, 10 = Create, 11 = Invalid.
See "ACTION_" in API.H for equates.

Output

pJobAliasHandle Handle to a print job or 0 if error

Return values

SUCCESS_CODE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTSetDeviceData

Description

Sets numeric data associated with a print device.

Syntax

```
UINT32
PRINTSetDeviceData(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pDevAlias,
    UINT32    *entryIDList,
    UINT8     *setList)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pDevAliasHandle
Alias print device handle

entryIDList Zero-terminated array of UINT32 containing any combination of the following typedefs along with their respective size in bytes:

SET_DEVICE_DEF_FLAGS, 4
SET_DEVICE_DEF_COPIES, 4
SET_DEVICE_DEF_LINES, 4
SET_DEVICE_DEF_COLS, 4
SET_DEVICE_DEF_TAB, 4
SET_DEVICE_DEF_FORMTYPE, 4
SET_DEVICE_DEF_CONTROL, 2
SET_DEVICE_DEF_SETUP_ALLOCLEN, 4
SET_DEVICE_DEF_RESET_ALLOCLEN, 4
SET_DEVICE_DEF_SETUP_BUFFLEN, 4
SET_DEVICE_DEF_RESET_BUFFLEN, 4
SET_DEVICE_DEF_SETUP_PTR, 4
SET_DEVICE_DEF_RESET_PTR, 4

setList Caller-defined structure defined by the size and ordering of the *entryIDList*. If *entryIDList* is defined as:

```
entryIDList[]= {SET_DEVICE_DEF_FLAGS, 4,
                SET_DEVICE_DEF_COPIES,4,
               0};
```

then getList would be a pointer to an array that looks like:

```
typedef getListTag {
UINT32   flags;
UINT32   copies;
} setListStruct;
```

And the call to **PRINTSetDeviceData** would look like:

```
PRINTSetDeviceData(
    modHandle,
    pgID,
    processID,
    pDevAliasHandle,
    entryIDList,
    &getList);
```

Return values

SUCCESS_CODE

INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTSetDeviceStrData

Description Sets string data associated with a print device.

Syntax

```
UINT32
PRINTSetDeviceStrData(
    UINT32 pgID,
    UINT32 processID,
    UINT32 pDevAliasHandle,
    UINT32 elementNumber,
    UINT8 *buffer,
    UINT32 size,
    UINT32 *bWritten)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pDevAliasHandle
Alias print device handle

elementNumber
SET_DEVICE_DEF_BANNER
SET_DEVICE_DEF_FORMNAME
SET_DEVICE_DEF_BANNERNAME
SET_DEVICE_DEF_DESCRIPTION
SET_DEVICE_DEF_RESET_STR
SET_DEVICE_DEF_SETUP_STR

buffer Buffer to copy into device string field

size Size in bytes of string in buffer

Output

bWritten Number of bytes written into the device string field.

Return values

SUCCESS_CODE
INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

Remarks The *bufferLen* field in the setup or reset string structures will be set

to the value that is returned in *bWritten* when this is called to establish or change a setup or reset string.

See also PRINTGetDeviceStrData, PRINTGetJobStrData, PRINTSetJobStrData

PRINTSetJobData

Description

Sets data associated to a print job from a caller supplied buffer.

Syntax

```
UINT32
PRINTSetJobData (
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle,
    UINT32   *entryIDList,
    UINT8    *setList)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle

entryIDList Zero-terminated array of UINT32s containing any combination of the following typedefs along with their respective sizes, in bytes:

SET_JOB_FLAGS, 4
SET_JOB_COPIES, 4
SET_JOB_LINES, 4
SET_JOB_COLS, 4
SET_JOB_TAB, 4
SET_JOB_FORMTYPE, 4
SET_JOB_CONTROL, 2
SET_JOB_TARGETTIME, 6
SET_JOB_SETUP_ALLOCLEN, 4
SET_JOB_RESET_ALLOCLEN, 4
SET_JOB_SETUP_BUFFLEN, 4
SET_JOB_RESET_BUFFLEN, 4
SET_JOB_SETUP_PTR, 4
SET_JOB_RESET_PTR, 4

setList Caller-defined structure defined by the size and ordering of the *entryIDList*. If *entryIDList* is defined as:

```
entryIDList[]= {SET_JOB_FLAGS, 4,
               SET_JOB_COPIES,4, 0};
```

then *getList* would be a pointer to an array that looks like:

```
typedef setListTag {
UINT32   flags;
UINT32   copies;
} setListStruct;
```

And the call to **PRINTSetDeviceData** would look like:

```
PRINTSetDeviceData(
    modHandle,
    pgID,
    processID,
    pDevAliasHandle,
    entryIDList,
    &getList);
```

Return values

SUCCESS_CODE

INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

PRINTSetJobStrData

Description Sets data associated with a print job from a caller-supplied buffer.

Syntax

```
UINT32
PRINTSetJobStrData (
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAliasHandle,
    UINT32    elementNumber,
    UINT8   *buffer,
    UINT32    size,
    UINT32  *bWritten)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle

elementNumber
SET_JOB_BANNER
SET_JOB_FORMNAME
SET_JOB_BANNERNAME
SET_JOB_DESCRIPTION
SET_JOB_RESET_STR
SET_JOB_SETUP_STR

buffer Buffer to copy into job string field

size Size in bytes of string in buffer

bWritten Buffer that SetJobStrData fills with number of bytes written.

Output

bWritten Number of bytes written into the job string field

Return values

SUCCESS_CODE

INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF

Remarks

The *bufferLen* field in the setup or reset string structures will be set to the value that is returned in *bWritten* when this is called to establish or change a setup or reset string.

See also

PRINTGetJobStrData, PRINTGetDeviceStrData, PRINTSetDeviceStrData

PRINTWriteToJob

Description

Writes a buffer to a print job

Syntax

```
UINT32
PRINTWriteToJob(
    UINT32    pgID,
    UINT32    processID,
    UINT32    pJobAlias,
    UINT32    size,
    UINT8     *printBuffer
    UINT32    *reserved)
```

Input

pgID Process group ID of the calling process group.

processID Process ID of the calling process.

pJobAliasHandle
Print job handle.

size Number of bytes being written.

printBuffer Buffer containing data to be written.

reserved Reserved. Set to NULL.

Return values

SUCCESS_CODE
- PRN_FILE_CREATED
- INVALID_JOB_HANDLE
- INVALID_PDEVICE_HANDLE
- INVALID_FILE_HANDLE
- OUT_OF_CLIENT_MEMORY
- NCP_NO_MORE_FILE_HANDLES
- INVALID_PARAMETER *pgID* or *processID* equal 0 or 0xFFFFFFFF