Reference

ClosePipe
Syntax

Pipe ClosePipe(Pipe)

Parameter	Type/Description	
Pipe	**PIPE**	Specifies a handle to a pipe.

Closes the pipe designated by Pipe.

Return Value

Handle to the closed pipe. No significance.

CreatePipe
Syntax

PIPE CreatePipe(hWnd, lpszPipeName, wStyle, wBufferSize, wNotify);

Parameter	Type/Description	
hWnd	**HWND**	Identifies the Window who owns the pipe.
lpszPipeName	**LPSTR**	Name of pipe to be created. (under 16 bytes)
wStyle	**WORD**	Bitwise ORing of Pipe styles.
wBufferSize	**WORD**	Size of Pipe buffer, zero for default.
wNotify	**WORD**	wParam passed to window with a WM_USER message.

This function creates a pipe. A buffer is allocated within the DLL and information is stored about the pipe user. Access to the pipe is regulated by **wStyle.** If **wNotify** is non-zero this value will be sent to **hWnd** as wParam in a WM_USER message if **wStyle** is PIPE_READ or PIPE_WRITE. See **Pipe Notification** for more details**.**

Return Value

A handle to a pipe, or an error. An error is indicated by a number less than zero.

OpenPipe
Syntax

PIPE OpenPipe(hWnd, lpszPipeName, wStyle, wNotify);

Parameter	Type/Description	
hWnd	**HWND**	Identifies the Window who owns the pipe.
lpszPipeName	**LPSTR**	Name of pipe to be created. (under 16 bytes)
wStyle	**WORD**	Bitwise ORing of Pipe styles.
wNotify	**WORD**	wParam passed to window with a WM_USER message.

This function is similar to CreatePipe except that a pipe is not actually created. This function sets the access rights a window has to a previously created pipe.

Return Value

A handle to the pipe or an error. An error is indicated by a number less than zero.

PeekPipe

Syntax

WORD PeekPipe(lpBuffer, iNum, Pipe);

Parameter	**Type/Description**	
lpBuffer	**LPSTR**	A buffer to contain data from pipe.
iNum	**int**	The number of bytes to copy from pipe
Pipe	**PIPE**	Valid pipe handle, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

PeekPipe functions and behaves as a Pread except data is not removed from the pipe.

Return Value

The number of bytes actually copied into the pipe or an error. An error is indicated by a number less than zero.

PurgePipe

Syntax

PurgePipe(Pipe);

Parameter	**Type/Description**	
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

PurgePipe purges all data in a pipe, resetting internal indexes to zero.

Return Value

Zero or error. Error is indicated by a number less than zero.

QueryPipe

Syntax

WORD QueryPipe(Pipe);

Parameter	**Type/Description**	
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

QueryPipe checks the status of Pipe.

Return Value

The number of bytes in a pipe or error. An error is indicated by a number less than zero.

ReleasePipe

Syntax

WORD ReleasePipe(Pipe);

Parameter	Type/Description	
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

ReleasePipe releases any READ or WRITE ownership on a pipe that a particular window may have.

Return Value

The original pipe handle or an error. An error is indicated by a number less than zero.

Pputc

Syntax

WORD Pputc(c, Pipe);

Parameter	Type/Description	
c	**char**	A character byte to be written to a pipe.
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

Pputc puts a single character into a pipe.

Return Value

The number of bytes written or error. An error is indicated by a number less than zero.

Pputs

Syntax

WORD Pputs(lpszString, Pipe);

Parameter	Type/Description	
lpszString	**LPSTR**	A null terminated string.
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

Pputs copies the contents of lpszString to the buffer of Pipe, up to but not including the terminating NULL character.

Return Value

The number of bytes written or error. An error is indicated by a number less than zero.

Pgetc

Syntax

WORD Pgetc(Pipe);

Parameter	Type/Description	
Pipe	**PIPE**	A handle to a valid pipe, can be a standard pipe or

a pipe handle obtained from an OpenPipe or a CreatePipe.

Reads and removes a single byte from a pipe.

Return Value

The character read or an error. An error is indicated by a number less than zero.

Pgets

Syntax

WORD Pgets(lpszString, iNum, Pipe);

<u>**Parameter**</u>	<u>**Type/Description**</u>	
lpszString	**LPSTR**	A buffer to contain data from pipe.
iNum	**int**	The number of bytes to copy from pipe.
Pipe	**PIPE**	Valid pipe handle, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

Pgets reads and removes up to iNum bytes from Pipe and copies them into the buffer pointed to by lpszString. The data is capped off with a terminating NULL character.

Return Value

The number of bytes actually read or an error. An error is indicated by a number less than zero.

Pread

Syntax

Pread(lpBuffer, iNum, Pipe);

<u>**Parameter**</u>	<u>**Type/Description**</u>	
lpBuffer	**LPSTR**	A buffer to contain data from pipe.
iNum	**int**	The number of bytes to copy from pipe.
Pipe	**PIPE**	Valid pipe handle, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

Pread reads and removes iNum bytes from Pipe.

Return Value

The number of bytes actually read or an error. An error is indicated by a number less than zero.

Pwrite

Syntax

WORD Pwrite(lpBuffer,iItemSize, iCount, Pipe);

<u>**Parameter**</u>	<u>**Type/Description**</u>	
lpBuffer	**LPSTR**	A buffer of iCount data objects.
iItemSize	**int**	The size of objects contained in buffer.

iCount	**int**	The number of objects in lpBuffer.
Pipe	**PIPE**	Valid pipe handle, can be a standard pipe or a pipe handle obtained from an OpenPipe or a CreatePipe.

Pwrite will write iCount objects of iItemSize to Pipe.

Return Value

The number of bytes actually written or an error. An error is indicated by a number less than zero.

Wgetc

Syntax

WORD Wgetc();

Wgetc reads and removes one byte from the standard pipe "Stdin."

Return Value

The character read or an error. An error is indicated by a number less than zero.

Wgets

Syntax

Wgets(lpszString, iNum);

Parameter	Type/Description	
lpszString	**LPSTR**	A buffer to contain returned string.
iNum	**int**	The number of bytes to read.

Wgets reads and removes iNum characters from the standard pipe "Stdin."

Return Value

The number of bytes actually read or an error. An error is indicated by a number less than zero.

Wputc

Syntax

WORD Wputc(c);

Parameter	Type/Description	
c	**char**	The byte to be written.

Wputc writes one byte, c, to the standard pipe "Stdout."

Return Value

The number of bytes actually written or an error. An error is indicated by a number less than zero.

Wputs

Syntax

Wputs(lpszString);

Parameter	Type/Description	
lpszString	**LPSTR**	The string to be written.

Wputs writes a null terminated string, up to but not including the terminating NULL character to the standard pipe "Stdout."

Return Value

The number of bytes actually written or an error. An error is indicated by a number less than zero.

Wprintf

Syntax

WORD Wprintf(lpszFmt,[*argument*]...);

Parameter	Type/Description	
lpszFmt	**LPSTR**	The printf format string.
argument	One or more optional parameters. See page 4-465 of the Windows SDK reference Volume #1. This function has the same arguments and syntax as "wsprintf," except for lpOutput which in this case is the standard pipe Stdout.	

Writes a printf format string and following arguments to the standard pipe "Stdout."

Return Value

The number of bytes written or an error. An error is indicated by a number less than zero.

Comments

This function uses the Windows function wvsprintf for the string formatting. All limitations that apply to wvsprintf apply to this function also.Furthermore, a temporary buffer is used as storage for the wvsprintf call; its maximum size is 1024 bytes.