

clipboard

COLLABORATORS

	<i>TITLE :</i> clipboard	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>
WRITTEN BY		March 14, 2022
<i>SIGNATURE</i>		

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	clipboard	1
1.1	clipboard.doc	1
1.2	clipboard.device/CBD_CHANGEHOOK	1
1.3	clipboard.device/CBD_CURRENTREADID	2
1.4	clipboard.device/CBD_CURRENTWRITEID	2
1.5	clipboard.device/CBD_POST	2
1.6	clipboard.device/CMD_READ	3
1.7	clipboard.device/CMD_UPDATE	4
1.8	clipboard.device/CMD_WRITE	5

Chapter 1

clipboard

1.1 clipboard.doc

```
CBD_CHANGEHOOK
CBD_CURRENTREADID
CBD_CURRENTWRITEID
CBD_POST
CMD_READ
CMD_UPDATE
CMD_WRITE
```

1.2 clipboard.device/CBD_CHANGEHOOK

NAME

CBD_CHANGEHOOK -- Add or remove a clip change hook

FUNCTION

CBD_CHANGEHOOK allows specification of a hook to be called when the data on the clipboard has changed.

IO REQUEST

```
io_Message  mn_ReplyPort set up
io_Device   preset by OpenDevice
io_Unit     preset by OpenDevice
io_Command  CBD_CHANGEHOOK
io_Length   -      0 to remove, 1 to install this hook
io_Data     -      struct Hook *, the clip change hook
```

HOOK ENVIRONMENT

```
hook message - a ClipHookMsg, as defined in devices/clipboard.h
chm_Type     - 0, indicating that the message has the
```

following fields:
chm_ClipID - the clip ID of the clip triggering the change
hook object - io_Unit

1.3 clipboard.device/CBD_CURRENTREADID

NAME

CBD_CURRENTREADID - Determine the current read identifier.

FUNCTION

CBD_CURRENTREADID fills the io_ClipID with a clip identifier that can be compared with that of a post command: if greater than the post identifier then the post data held privately by an application is not valid for its own pasting.

IO REQUEST

io_Message mn_ReplyPort set up
io_Device preset by OpenDevice
io_Unit preset by OpenDevice
io_Command CBD_CURRENTREADID

RESULTS

io_ClipID the ClipID of the current write is set

1.4 clipboard.device/CBD_CURRENTWRITEID

NAME

CBD_CURRENTWRITEID -- Determine the current write identifier.

FUNCTION

CBD_CURRENTWRITEID fills the io_ClipID with a clip identifier that can be compared with that of a post command: if greater than the post identifier then the post is obsolete and need never be satisfied.

IO REQUEST

io_Message mn_ReplyPort set up
io_Device preset by OpenDevice
io_Unit preset by OpenDevice
io_Command CBD_CURRENTWRITEID

RESULTS

io_ClipID the ClipID of the current write is set

1.5 clipboard.device/CBD_POST

NAME

CBD_POST -- Post availability of a clip to the clipboard.

FUNCTION

Indicate to the clipboard device that data is available for use by accessors of the clipboard. This is intended to be used when a cut is large, in a private data format, and/or changing frequently, and it thus makes sense to avoid converting it to an IFF form and writing it to the clipboard unless another application wants it. The post provides a message port to which the clipboard device will send a satisfy message if the data is required.

If the satisfy message is received, the write associated with the post must be performed. The act of writing the clip indicates that the message has been received: it may then be re-used by the clipboard device, and so must actually be removed from the satisfy message port so that the port is not corrupted.

If the application wishes to determine if a post it has performed is still the current clip, it should check the post's `io_ClipID` with that returned by the

`CBD_CURRENTREADID`

command. If the current read `io_ClipID` is greater, the clip is not still current.

If an application has a pending post and wishes to determine if it should satisfy it (e.g. before it exits), it should check the post's `io_ClipID` with that returned by the

`CBD_CURRENTWRITEID`

command. If the current write `io_ClipID` is greater, there is no need to satisfy the post.

IO REQUEST

`io_Message` `mn_ReplyPort` set up
`io_Device` preset by `OpenDevice`
`io_Unit` preset by `OpenDevice`
`io_Command` `CBD_POST`
`io_Data` pointer to satisfy message port
`io_ClipID` zero

RESULTS

`io_Error` non-zero if an error occurred
`io_ClipID` the clip ID assigned to this post, to be used in the write command if this is satisfied

1.6 clipboard.device/CMD_READ

NAME

`CMD_READ` -- Read from a clip on the clipboard.

FUNCTION

The read function serves two purposes.

When `io_Offset` is within the clip, this acts as a normal read

request, and `io_Data` is filled with data from the clipboard. The first read request should have a zero `io_ClipID`, which will be filled with the ID assigned for this read. Normal sequential access from the beginning of the clip is achieved by setting `io_Offset` to zero for the first read, then leaving it untouched for subsequent reads. If `io_Data` is null, then `io_Offset` is incremented by `io_Actual` as if `io_Length` bytes had been read: this is useful to skip to the end of file by using a huge `io_Length`.

When `io_Offset` is beyond the end of the clip, this acts as a signal to the clipboard device that the application is through reading this clip. Realize that while an application is in the middle of reading a clip, any attempts to write new data to the clipboard are held off. This read past the end of file indicates that those operations may now be initiated.

IO REQUEST

`io_Message` `mn_ReplyPort` set up
`io_Device` preset by `OpenDevice`
`io_Unit` preset by `OpenDevice`
`io_Command` `CMD_READ`
`io_Length` number of bytes to put in data buffer
`io_Data` pointer to buffer of data to fill, or null to skip over data
`io_Offset` byte offset of data to read
`io_ClipID` zero if this is the initial read

RESULTS

`io_Error` non-zero if an error occurred
`io_Actual` filled with the actual number of bytes read
`io_Data` (the buffer now has `io_Actual` bytes of data)
`io_Offset` updated to next read position, which is beyond EOF if `io_Actual` != `io_Length`
`io_ClipID` the clip ID assigned to this read: do not alter for subsequent reads

1.7 clipboard.device/CMD_UPDATE

NAME

`CMD_UPDATE` -- Terminate the writing of a clip to the clipboard.

FUNCTION

Indicate to the clipboard that the previous write commands are complete and can be used for any pending pastes (reads). This command cannot be issued while any of the write commands are pending.

IO REQUEST

`io_Message` `mn_ReplyPort` set up
`io_Device` preset by `OpenDevice`
`io_Unit` preset by `OpenDevice`
`io_Command` `CMD_UPDATE`
`io_ClipID` the ClipID of the write

RESULTS
io_Error non-zero if an error occurred

1.8 clipboard.device/CMD_WRITE

NAME

CMD_WRITE -- Write to a clip on the clipboard.

FUNCTION

This command writes data to the clipboard. This data can be provided sequentially by clearing io_Offset for the initial write, and using the incremented value unaltered for subsequent writes. If io_Offset is ever beyond the current clip size, the clip is padded with zeros.

If this write is in response to a SatisfyMsg for a pending post, then the io_ClipID returned by the

CBD_POST
command must

be used. Otherwise, a new ID is obtained by clearing the io_ClipID for the first write. Subsequent writes must not alter the io_ClipID.

IO REQUEST

io_Message mn_ReplyPort set up
io_Device preset by OpenDevice
io_Unit preset by OpenDevice
io_Command CMD_WRITE
io_Length number of bytes from io_Data to write
io_Data pointer to block of data to write
io_Offset usually zero if this is the initial write
io_ClipID zero if this is the initial write, ClipID of the Post if this is to satisfy a post

RESULTS

io_Error non-zero if an error occurred
io_Actual filled with the actual number of bytes written
io_Offset updated to next write position
io_ClipID the clip ID assigned to this write: do not alter for subsequent writes