## UF_UNDO_ask_any_mark_exist (view source)

**Defined in: uf_undo.h**

### Overview
Queries whether at least one mark of visibility exists.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_ask_any_mark_exist
(
    UF_UNDO_user_visibility_t visibility,
    int * any_exists
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | visibility of mark of interest |
| int * | **any_exists** | Output | if at least 1 mark of 'visibility' exists then 1 else 0 |

## UF_UNDO_ask_mark_exist (view source)

**Defined in: uf_undo.h**

### Overview
Queries whether a mark exists.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_ask_mark_exist
(
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_mark_name_c_t mark_name,
    int * exists
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | id of mark to inquire or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | Mark name to inquire. Only used when mark_id is set to UF_UNDO_USE_NAME_ID. |
| int * | **exists** | Output | Is 0 if mark_id DOES NOT exist, 1 otherwise |

## UF_UNDO_ask_mark_visibility (view source)

**Defined in: uf_undo.h**

### Overview
Finds the visibility of a previously set mark.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_ask_mark_visibility
(
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_mark_name_c_t mark_name,
    UF_UNDO_user_visibility_t * visibility
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | Mark id to get the visibility of or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | Mark name to get visibility of. This is used when mark_id is set to UF_UNDO_USE_NAME_ID. |
| UF_UNDO_user_visibility_t * | **visibility** | Output | The visibility of the specified mark. |

---

## UF_UNDO_ask_next_vis_mark (view source)

**Defined in: uf_undo.h**

### Overview
Passes back the mark_id of the Next Visible Mark. This is the mark that is undone to, on a call to UF_UNDO_undo_to_next_vis_mark.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_ask_next_vis_mark
(
    UF_UNDO_mark_id_t * mark_id
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t * | **mark_id** | Output | The mark id of the next visible mark. This may also be the value UF_UNDO_NO_VIS_MARK_PRESENT in which case there are no more visible marks to undo to. |

## UF_UNDO_ask_number_of_marks (view source)

**Defined in: uf_undo.h**

### Overview
Finds the number of marks for the specified visibility.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_ask_number_of_marks
(
    UF_UNDO_user_visibility_t visibility,
    int * how_many
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | The visibility of the specified mark. |
| int * | **how_many** | Output | Number of marks that exist with the specified visibility. |

## UF_UNDO_delete_all_marks (view source)

**Defined in: uf_undo.h**

### Overview
Deletes all marks. It instructs all logged on data managers to delete their marks associated with each mark. It deals with each data manager in the appropriate way. This function frees up all mark space.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_delete_all_marks
(
    void
)
```

## UF_UNDO_delete_all_misc_cbs (view source)

**Defined in: uf_undo.h**

### Overview

Deletes all the miscellaneous callbacks currently registered.

### Environment

Internal and External

### Required License(s)

gateway

```
int UF_UNDO_delete_all_misc_cbs
(
    void
)
```

---

## UF_UNDO_delete_mark (view source)

**Defined in: uf_undo.h**

### Overview

Flags the specified mark as deleted. It does not always remove the mark. The mark is logically deleted from UF_UNDO and therefore no one can call any UF_UNDO function with the mark id of this deleted mark successfully.

Since this function does free up some mark space (and sometimes all mark space) it is useful to call if one is trying to conserve space.

### Environment

Internal and External

### Required License(s)

gateway

```
int UF_UNDO_delete_mark
(
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_mark_name_c_t mark_name
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | Mark to delete or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | Mark name to delete. This is used when mark_id is set to UF_UNDO_USE_NAME_ID. |

---

## UF_UNDO_delete_to_mark (view source)

**Defined in: uf_undo.h**

### Overview

Performs a delete to a specific mark (inclusively) that was previously set by UF_UNDO_set_mark. It instructs all data managers that are logged on to delete their marks associated with the deleted UF_UNDO marks. It deletes from the last mark set to the specified mark.

### Environment

Internal and External

### Required License(s)

gateway

```
int UF_UNDO_delete_to_mark
(
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_mark_name_c_t mark_name
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | Mark to delete to or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | Mark name to delete to. This is used when mark_id is set to UF_UNDO_USE_NAME_ID. |

---

## UF_UNDO_disable_misc_cbs (view source)

**Defined in: uf_undo.h**

### Overview

Disables all miscellaneous callbacks. That is, no miscellaneous callback is called by UF_UNDO after this function is called until UF_UNDO_enable_misc_cbs is called. Miscellaneous callbacks are enabled when UF_UNDO starts up.

### Environment

Internal and External

### Required License(s)

gateway

```
int UF_UNDO_disable_misc_cbs
(
    void
)
```

---

## UF_UNDO_enable_misc_cbs (view source)

**Defined in: uf_undo.h**

### Overview

Enables miscellaneous callbacks. That is, all miscellaneous callbacks are called by UF_UNDO after this function is called until UF_UNDO_disable_misc_cbs is called. Miscellaneous callbacks are enabled when UF_UNDO starts up.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_enable_misc_cbs
(
    void
)
```

---

## UF_UNDO_register_misc_cb (view source)

**Defined in: uf_undo.h**

### Overview
This function allows the Open API programmer to register a callback to be executed before or after the setting/undoing of a mark.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_register_misc_cb
(
    UF_UNDO_misc_cb_t cb_type,
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_user_visibility_t visibility,
    UF_UNDO_misc_cb_f_t func,
    void * closure,
    UF_UNDO_misc_cb_id_t * id
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_misc_cb_t | **cb_type** | Input | Call back types:<br>UF_UNDO_misc_cb_set_pre to call the function just before a mark is set. The function will always be passed a mark_id of -2.<br><br>UF_UNDO_misc_cb_set_post to call the function just after any mark is set but before UF_UNDO returns back to the application that set the mark. The passed mark_id is the mark id of the mark just set.<br><br>UF_UNDO_misc_cb_undo_pre to call the function just before UF_UNDO performs an undo to mark. The passed mark_id is the mark id of the mark we are about to undo to.<br><br>UF_UNDO_misc_cb_undo_post to call the function just after UF_UNDO performs an undo to mark but before UF_UNDO returns to the application that requested the undo. The passed mark_id is the |

| | | | |
|---|---|---|---|
| | | | mark id of the mark we just undid to.<br><br>UF_UNDO_misc_cb_chg_vis to call the function just after a mark has its visibility changed. The passed mark_id is the mark id of the mark whose visibility just changed. |
| UF_UNDO_mark_id_t | **mark_id** | Input | If registering a function that should only be called if processing a particular mark then enter that mark_id here. If its a SET_PRE or SET_POST then this argument is ignored. If you want a function called whenever any mark is processed then enter UF_UNDO_MISC_CB_ANY_MARK here.<br><br>If you enter a mark here and UF_UNDO is requested to undo over that mark then the callback is not called. It is only called if the given mark_id is the one undone to. |
| UF_UNDO_user_visibility_t | **visibility** | Input | Take action based on the specific visibility:<br><br>UF_UNDO_visible<br>Only execute the given function if the mark being set or undone to is user visible or a mark was just made visible.<br>UF_UNDO_invisible<br>Only execute the given function if the mark being set or undone to is user invisible or a mark was just made invisible.<br>UF_UNDO_any_vis to execute the function regardless of the mark's visibility. |
| UF_UNDO_misc_cb_f_t | **func** | Input | The function to call.<br>When this function is executed, the UF text mode will be set to whatever the mode is when the function is registered.<br>If your SET_PRE or UNDO_PRE callback returns UF_UNDO_misc_cb_stop when called then no further callbacks are called and the setting or undoing of the mark is not done.<br>If other callbacks return UF_UNDO_misc_cb_stop when called then the operation will be partly complete and an error will be returned. This action is not recommended. |
| void * | **closure** | Input | The argument to pass to the function when the function is called. |
| UF_UNDO_misc_cb_id_t * | **id** | Output | The identifier assigned to this registration. This identifier can be used to remove this registration via UF_UNDO_unregister_misc_cb(). |

## UF_UNDO_set_mark (view source)

**Defined in: uf_undo.h**

### Overview

Instructs every data manager which is logged on to set a mark.
Returns an identifier which the application can use to identify the
state of the data managers at the time the mark is set. Additionally,

the application can register a name by which this mark can later be referenced.

Note: The maximum number of undo marks in an Open API program is 100.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_set_mark
(
    UF_UNDO_user_visibility_t visibility,
    UF_UNDO_mark_name_c_t mark_name,
    UF_UNDO_mark_id_t * mark_id
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | The user visibility of this mark. |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | optional name for this mark. If NULL then not used. If not NULL then must be '\0' terminated. UF_UNDO copies this name to some space of its own, therefore, the caller may do whatever it wants with 'mark_name' after the call. |
| UF_UNDO_mark_id_t * | **mark_id** | Output | identifier associated with this mark. |

---

## UF_UNDO_set_mark_visibility (view source)

**Defined in: uf_undo.h**

### Overview
Changes the visibility of a previously set mark to visibility.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_set_mark_visibility
(
    UF_UNDO_mark_id_t mark_id,
    UF_UNDO_mark_name_c_t mark_name,
    UF_UNDO_user_visibility_t visibility
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | Mark to change the visibility of or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | Mark name to set visibility of. This is used when mark_id is set to UF_UNDO_USE_NAME_ID. |

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | The new visibility for specified mark. |

---

## UF_UNDO_set_to_mark_visibility (view source)

**Defined in: uf_undo.h**

### Overview

An application calls this function to change the visibility of all existing
marks from the current mark to the specified "to_mark", inclusively.
The visibility of these marks is set to "visibility".

### Environment

Internal and External

### Required License(s)

gateway

```
int UF_UNDO_set_to_mark_visibility
(
    UF_UNDO_mark_id_t to_mark,
    UF_UNDO_mark_name_c_t mark_name,
    UF_UNDO_user_visibility_t visibility
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **to_mark** | Input | The "to_mark" to change the visibility of, or UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | The mark name of the "to mark" to set visibility. Only used when to_mark is UF_UNDO_USE_NAME_ID. |
| UF_UNDO_user_visibility_t | **visibility** | Input | The new visibility for "to mark". |

---

## UF_UNDO_undo_to_last_mark (view source)

**Defined in: uf_undo.h**

### Overview

Performs an undo to the last mark of the specified visibility that was
previously set by UF_UNDO_set_mark. It instructs all data managers
that are logged on to return to their state associated with that mark.
The visibility referred to is the mark's current visibility which may
differ from the mark's original visibility (see UF_UNDO_set_mark_visibility).

### Environment

Internal and External

### See Also

UF_UNDO_set_mark_visibility

### Required License(s)

gateway

**int UF_UNDO_undo_to_last_mark**
**(**
**UF_UNDO_user_visibility_t visibility,**
**UF_UNDO_mark_id_t * mark_id**
**)**

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | The desired visibility of the mark to undo to. |
| UF_UNDO_mark_id_t * | **mark_id** | Output | id of mark undone to |

## UF_UNDO_undo_to_mark (view source)

**Defined in: uf_undo.h**

### Overview

Performs an undo to a specific mark that was previously set by UF_UNDO_set_mark. It instructs all data managers that are logged on to return to their state associated with the specified mark.

### Environment

Internal and External

### Required License(s)

gateway

**int UF_UNDO_undo_to_mark**
**(**
**UF_UNDO_mark_id_t mark_id,**
**UF_UNDO_mark_name_c_t mark_name**
**)**

| | | | |
|---|---|---|---|
| UF_UNDO_mark_id_t | **mark_id** | Input | id of mark to undo to if != UF_UNDO_USE_NAME_ID |
| UF_UNDO_mark_name_c_t | **mark_name** | Input | name of mark to undo to if mark_id == UF_UNDO_USE_NAME_ID |

## UF_UNDO_undo_to_next_vis_mark (view source)

**Defined in: uf_undo.h**

### Overview

UNDO to the Next Visible Mark. The Next Visible Mark (NVM) can be described as follows:
1. Whenever a new mark is set AND that mark is visible, then it becomes the NVM.
2. If the NVM is undone to or over (i.e. the NVM is mark j and we undo to mark i, i <= j) then the first visible mark

prior to mark undone to becomes the NVM.
3. Same as 2 except for deleting marks instead of undoing to marks.
4. If the NVM is made invisible then the first visible mark previous to that NVM is made the new NVM.
5. If an existing invisible mark is made visible, then it is made the NVM if it was set after the previous NVM.
UF_UNDO's NVM is updated upon a successful call to this function. Therefore, you may step back thru the visible marks by successive calls to this function. For example:
while( UF_UNDO_undo_to_next_vis_mark() == UF_UNDO_REQ_OK )

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_undo_to_next_vis_mark
(
    void
)
```

---

## UF_UNDO_undo_to_prev_mark (view source)

**Defined in: uf_undo.h**

### Overview
Performs an undo to the last mark of the specified visibility that was set before the previous_to mark. E.g., if previous_to is ID 4 and the desired visibility is visible then this function finds the last visible mark that appears before mark 4.

### Environment
Internal and External

### Required License(s)
gateway

```
int UF_UNDO_undo_to_prev_mark
(
    UF_UNDO_user_visibility_t visibility,
    UF_UNDO_mark_id_t previous_to,
    UF_UNDO_mark_id_t * mark_id
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_user_visibility_t | **visibility** | Input | The desired visibility of the mark to undo to. |
| UF_UNDO_mark_id_t | **previous_to** | Input | find the last mark of "visibility" appearing before this mark |
| UF_UNDO_mark_id_t * | **mark_id** | Output | The ID of the mark undone to. |

## UF_UNDO_unregister_misc_cb (view source)

**Defined in: uf_undo.h**

### Overview
Removes a registration previously made by UF_UNDO_register_misc_cb.

### Environment
Internal and External

### See Also
UF_UNDO_register_misc_cb

### Required License(s)
gateway

```
int UF_UNDO_unregister_misc_cb
(
    UF_UNDO_misc_cb_id_t cb_id
)
```

| | | | |
|---|---|---|---|
| UF_UNDO_misc_cb_id_t | **cb_id** | Input | The id to remove. |