## UF_CLONE_add_assembly (view source)

**Defined in: uf_clone.h**

### Overview

This routine adds an assembly to the current clone operation. Any load errors will be placed in the load_status output argument. Possible errors that will be recorded here are
UF_CLONE_err_bvr_out_of_sync - meaning that the BVR has been modified in NX Manager since the part was last saved, and
UF_CLONE_err_comp_not_found, meaning a component could not be found using the current load options.

### Environment

Internal and External

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_add_assembly
(
    const char * part_name,
    UF_PART_load_status_p_t load_status
)
```

| | | | |
|---|---|---|---|
| const char * | **part_name** | Input | The name of the top-level assembly file to add;<br>for native, or import, an O/S filename; in NX Manager or for export, a CLI name |
| UF_PART_load_status_p_t | **load_status** | Output to UF_*free* | The load status for the initialisation operation |

## UF_CLONE_add_part (view source)

**Defined in: uf_clone.h**

### Overview

This routine adds a part to an initialised clone operation. If the part is an assembly part, any components of the assembly not already a part of the clone operation will be added as name only references

### Environment

Internal and External

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_add_part
(
    const char * part_name
)
```

| | | | |
|---|---|---|---|
| const char * | **part_name** | Input | The name of the part file to add; for native, or import, an O/S filename, in NX Manager or for export, a CLI name |

---

## UF_CLONE_apply_defaults (view source)

**Defined in: uf_clone.h**

### Overview

This routine applies default values to all parts in the clone
operation without performing the clone. Defaults application
may generate naming rule failures. In this case the clone will
not be performed, the return code will be
UF_CLONE_err_naming_failures and the
naming_failures argument will be filled in with:
a count of the number of errors (n_failures)
an array of error codes (statuses) which must be freed
an array of input names (input_names) which must be freed
with UF_FREE_string_array
an array of output_names (output_names), entries may be
null, must be freed with UF_FREE_string_array

The function UF_CLONE_init_naming_failures should be used to
initialise the naming_failures structure before it is passed to
this function.

### Return

Return code :
= 0 : successful
UF_CLONE_err_naming_failures: naming failures occurred,
details reported in the naming_failures output
argument
= not 0: Error code, use UF_get_fail_message to obtain message string

### Environment

Internal and External

### See Also

UF_CLONE_naming_failures_t
UF_CLONE_init_naming_failures

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_apply_defaults
(
    UF_CLONE_naming_failures_p_t naming_failures
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_naming_failures_p_t | **naming_failures** | Input / Output | Pointer to a naming failures structure.If a naming failure occurs the return code will be UF_CLONE_err_naming_failures and this structure will be filled in |

---

## UF_CLONE_apply_selective_export_xml (view source)

**Defined in: uf_clone.h**

### Overview
This routine uses the PLMXML structure specified in "filename" to define a selective Clone Export operation. It is mainly intended for use by the ug_selective_export utility. It has no meaning for a Clone or Clone Import operation.

### Environment
Internal and External

### History
Originally released in NX2.0

### Required License(s)
assemblies

```
int UF_CLONE_apply_selective_export_xml
(
    const char* xml_file,
    UF_PART_load_status_p_t load_status
)
```

| | | |
|---|---|---|
| const char* | **xml_file** | Input |
| UF_PART_load_status_p_t | **load_status** | Output to UF_*free* |

---

## UF_CLONE_ask_action (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the action which has been applied to the specified component in the current clone operation.

### Environment
Internal and External

### See Also
UF_CLONE_action_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_action
(
    const char * input_part_name,
    UF_CLONE_action_t * action_type
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | name of the part for which the action is to be returned,<br>in NX Manager or for export a CLI form name, otherwise a native file name |
| UF_CLONE_action_t * | **action_type** | Output | the action currently applied to this part |

---

## UF_CLONE_ask_assoc_file_copy (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the value of the associated file copy flag for the given part.

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_assoc_file_copy
(
    const char * input_part_name,
    logical * copy_associated
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation, in NX Manager clone, or for an export operation, this should be a CLI name, otherwise a native O/S file name |
| logical * | **copy_associated** | Output | whether to copy associated files for this part |

---

## UF_CLONE_ask_assoc_file_dir (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns the directory in which associated files for
the specified part will be placed, if relative, then relative to
the associated file root directory

### Environment

Internal and External

### See Also

UF_CLONE_set_assoc_file_root_dir
UF_CLONE_ask_assoc_file_root_dir
UF_CLONE_set_assoc_file_dir

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_assoc_file_dir
(
    const char * input_part_name,
    char * * assoc_file_dir
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import,<br>an O/S filename; for export, a CLI name |
| char * * | **assoc_file_dir** | Output to UF_*free* | directory to be used, a native file specification, relative or absolute, or the empty string |

---

## UF_CLONE_ask_assoc_file_root_dir (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns the root directory below which part specific associated
file directories will be placed (for export) or looked for (for import)

If this option has not been set, the empty string will be returned

### Environment

Internal and External

### See Also

UF_CLONE_set_assoc_file_root_dir

### History

Originally released in V16.0

### Required License(s)
gateway

**int UF_CLONE_ask_assoc_file_root_dir**
**(**
    **char * * root_directory**
**)**

| | | | |
|---|---|---|---|
| char * * | **root_directory** | Output to UF_*free* | the root directory which will be used, or the empty string |

---

## UF_CLONE_ask_attach_log_file (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the attach_log_file flag for the current operation. It has no meaning outside of the clone import operation.

### Environment
Internal and External

### History
Originally released in NX5.1

### Required License(s)
gateway

**int UF_CLONE_ask_attach_log_file**
**(**
    **logical * attach_log_file**
**)**

| | | |
|---|---|---|
| logical * | **attach_log_file** | Input |

---

## UF_CLONE_ask_ci (view source)

**Defined in: uf_clone.h**

### Overview
This routine asks the checkin options to be used for the specified part in the current import operation.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_ci
UF_CLONE_set_def_ci

UF_CLONE_set_ci

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_ci
(
    const char * input_part_name,
    UF_CLONE_checkin_data_p_t * checkin_data
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | the name of a part in the current import operation, a native O/S filename |
| UF_CLONE_checkin_data_p_t * | **checkin_data** | Output to UF_*free* | the checkin options which will be applied or null if none have been set |

---

## UF_CLONE_ask_ci_comment_checking (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the comment checking behaviour on checkin for the specified part.

### Return
Return Code:
=0: success
>0: failure code

### Environment
Internal or External

### History
Originally released in v19.0

### Required License(s)
gateway

```
int UF_CLONE_ask_ci_comment_checking
(
    const char * input_part_name,
    logical * error_unless_comments_match,
    char * * comment
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of the part in the current export operation, a CLI name |

| | | | |
|---|---|---|---|
| logical * | **error_unless_comments_match** | Output | Whether to report an error if the checkout comment does not match the given comment |
| char * * | **comment** | Output to UF_*free* | string containing the comment to be checked |

---

## UF_CLONE_ask_clone_related_cae (view source)

**Defined in: uf_clone.h**

### Overview

This routine asks the option to clone/export the related CAE parts when cloning any CAD parts
It has no meaning in the Clone Import Operation

### Environment

Internal and External

### History

Originally released in NX9.0

### Required License(s)

assemblies

```
int UF_CLONE_ask_clone_related_cae
(
    UF_CLONE_clone_rel_cae_p_t rel_cae
)
```

| | | |
|---|---|---|
| UF_CLONE_clone_rel_cae_p_t | **rel_cae** | Output |

---

## UF_CLONE_ask_clone_related_dwgs (view source)

**Defined in: uf_clone.h**

### Overview

This routine asks the current boolean option to clone/export the related drawings when cloning any CAD parts
It has no meaning in the Clone Import Operation

### Environment

Internal and External

### History

Originally released in NX9.0

### Required License(s)

assemblies

```
int UF_CLONE_ask_clone_related_dwgs
(
    logical* rel_dwgs
)
```

| | | |
|---|---|---|
| logical* | **rel_dwgs** | Output |

---

## UF_CLONE_ask_co (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the checkout options to be used for the specified part in the current export operation.

### Return
Return code :
= 0 : successful
= not 0: Error code, use UF_get_fail_message to obtain message string

### Environment
Internal and External

### See Also
UF_CLONE_set_def_co
UF_CLONE_ask_def_co
UF_CLONE_set_co
UF_CLONE_checkout_data_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_co
(
    const char * input_part_name,
    UF_CLONE_checkout_data_p_t * checkout_data
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of the part in the current export operation, a CLI name |
| UF_CLONE_checkout_data_p_t * | **checkout_data** | Output to UF_*free* | checkout options which will be applied or NULL if none have been set |

---

## UF_CLONE_ask_cvt_callbacks (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the currently registered convert callbacks in three parallel arrays containing the callbcack points, the names of the callbacks and the descriptions of the callbacks

### Environment
Internal and External

### See Also
UF_CLONE_register_cvt_callback
UF_CLONE_remove_cvt_callback

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_cvt_callbacks
(
    int * n_callbacks,
    UF_CLONE_convert_cb_p_t * points,
    char * * * names,
    char * * * descriptions
)
```

| | | | |
|---|---|---|---|
| int * | **n_callbacks** | Output | number of callbacks registered |
| UF_CLONE_convert_cb_p_t * | **points** | Output to UF_*free* | callback points |
| char * * * | **names** | Output to UF_*free* | n_callbacks (with UF_free_string_array) names of callbacks |
| char * * * | **descriptions** | Output to UF_*free* | n_callbacks (with UF_free_string_array) descriptions of callbacks |

---

## UF_CLONE_ask_def_action (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default action for the current clone operation

### Environment
Internal and External

### See Also
UF_CLONE_action_t

### History

Originally released in V16.0

### Required License(s)

gateway

**int UF_CLONE_ask_def_action**
**(**
**UF_CLONE_action_t * action**
**)**

| | | | |
|---|---|---|---|
| UF_CLONE_action_t * | **action** | Output | default action for the current clone |

---

## UF_CLONE_ask_def_assoc_file_copy (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a logical indication whether by default associated files will be copied, imported or exported (as appropriate) in this clone operation.

### Environment

Internal and External

### See Also

UF_CLONE_set_def_assoc_file_copy

### History

Originally released in V16.0

### Required License(s)

gateway

**int UF_CLONE_ask_def_assoc_file_copy**
**(**
**logical * copy_associated**
**)**

| | | | |
|---|---|---|---|
| logical * | **copy_associated** | Output | whether to copy associated files by default |

---

## UF_CLONE_ask_def_ci (view source)

**Defined in: uf_clone.h**

### Overview

This routine asks the default checkin options to be used in the current import operation.

### Environment

Internal and External

### See Also

UF_CLONE_set_def_ci
UF_CLONE_set_ci
UF_CLONE_ask_ci

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_def_ci
(
    UF_CLONE_checkin_data_p_t * checkin_data
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_checkin_data_p_t * | **checkin_data** | Output to UF_*free* | checkin options which will be applied, or null if non have been set |

---

## UF_CLONE_ask_def_ci_comment_checking (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default comment checking behaviour on checkin for the import operation

### Return

Return Code:
=0: success
>0: failure code

### Environment

Internal or External

### History

Originally released in v19.0

### Required License(s)

gateway

```
int UF_CLONE_ask_def_ci_comment_checking
(
    logical * error_unless_comments_match,
    char * * comment
)
```

| | | | |
|---|---|---|---|
| logical * | **error_unless_comments_match** | Output | Whether to report an error if the checkout comment<br>does not match the given comment |
| char * * | **comment** | Output to UF_*free* | string containing the comment to be checked |

## UF_CLONE_ask_def_co (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default checkout options to be used for the current export operation.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_co
UF_CLONE_ask_co
UF_CLONE_set_co
UF_CLONE_checkout_data_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_co
(
    UF_CLONE_checkout_data_p_t * checkout_data
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_checkout_data_p_t * | **checkout_data** | Output to UF_*free* | checkout options which will be applied or NULL if none have been set |

## UF_CLONE_ask_def_directory (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default directory that parts created during a native clone or a clone based export will be placed in.

If this option is not set then the empty string will be returned used

### Environment
Internal and External

### See Also
UF_CLONE_set_def_directory

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_directory
(
    char * * directory_name
)
```

| | | | |
|---|---|---|---|
| char * * | **directory_name** | Output to UF_*free* | directory to place created parts in |

---

## UF_CLONE_ask_def_folder (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the value of the default folder that parts
generated during an NX Manager clone or clone based import will be placed
in.

The folder name will be specified as
<user-name>:[<containing-folder>:...]<folder-name>
e.g. "smith:Imported Parts:machinehead" or "smith:My Parts"

If this option is not set, then the empty string will be returned

### Environment
Internal and External

### See Also
UF_CLONE_set_def_folder

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_folder
(
    char * * folder_name
)
```

| | | | |
|---|---|---|---|
| char * * | **folder_name** | Output to UF_*free* | the default folder parts will be placed in |

---

## UF_CLONE_ask_def_group (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns the default group that will be applied to Items, ItemRevisions and Datasets created during this operation. If this is the empty string, then the group under which the executing user is logged in will be used.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_group

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_group
(
    char * * group
)
```

| | | | |
|---|---|---|---|
| char * * | **group** | Output to UF_*free* | the group ownership which will be assigned by default to Items, ItemRevisions and datasets created by this clone operation |

---

## UF_CLONE_ask_def_item_type (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default item type that will be used in this import operation. This item type will be applied to newly created Items, ItemRevisions or Datasets for which the part being imported has not had its item type set by some other method (such as by having a user name applied, or via a convert callback)

This setting is ignored for export operations (where it is irrelevant), and for clone operations (where the type of the source part will be used if none has been otherwise set). However it will be recorded in the clone log-file.

### Environment
Internal and External

### See Also
UF_CLONE_set_item_type
UF_CLONE_set_def_item_type

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_item_type
(
```

**char * * item_type**
**)**

| | | | |
|---|---|---|---|
| char * * | **item_type** | Output to UF_*free* | string containing the default item type for this import operation |

---

## UF_CLONE_ask_def_naming (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns the default naming technique of the current clone operation. This is the naming technique that will be applied during an apply_defaults operation to all parts which have the naming type UF_CLONE_default_naming.

### Environment

Internal and External

### See Also

UF_CLONE_naming_technique_t
UF_CLONE_set_def_naming
UF_CLONE_apply_defaults

### History

Originally released in V16.0

### Required License(s)

gateway

**int UF_CLONE_ask_def_naming**
**(**
**UF_CLONE_naming_technique_t * naming**
**)**

| | | | |
|---|---|---|---|
| UF_CLONE_naming_technique_t * | **naming** | Output | the default naming technique of the current clone operation |

---

## UF_CLONE_ask_def_nm_copy (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a list of nonmaster copy options. Each entry contains:
nonmaster_type: the name of the nonmaster type to which this
flag applies, must be freed with UF_free
copy: logical indicating whether it will be copied
next: pointer to next element in list, NULL to end list

each entry must also be freed after use using UF_free

### Environment

Internal and External

### See Also
UF_CLONE_set_def_nm_copy
UF_CLONE_copy_nm_opt_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_nm_copy
(
    UF_CLONE_copy_nm_opt_p_t * copy_nonmaster_opts
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_copy_nm_opt_p_t * | **copy_nonmaster_opts** | Output to UF_*free* | list of default nonmaster copy options, which must each be freed |

---

## UF_CLONE_ask_def_owner (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default owner that will be applied to Items, ItemRevisions and Datasets created during this operation. If this is the empty string, then the user under which the executing user is logged in will be used.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_owner

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_owner
(
    char * * owner
)
```

| | | | |
|---|---|---|---|
| char * * | **owner** | Output to UF_*free* | default owner that will be assigned to Items, ItemRevisions and Datasets created during this operation |

---

## UF_CLONE_ask_def_pdm_desc (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a string containing the default description which will be applied during the import operation. This description will be applied to newly created Items and ItemRevisions for which the part being imported has not had its description set by some other method (such as by having a user name applied, or via a convert callback).

This setting is ignored for export operations (where it is irrelevant), and for clone operations (where the description of the source part will be used if none has been otherwise set).
However it will be recorded in the clone log-file.

### Environment

Internal and External

### See Also

UF_CLONE_convert_cb_t
UF_CLONE_set_def_pdm_desc

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_def_pdm_desc
(
    char * * pdm_desc
)
```

| | | | |
|---|---|---|---|
| char * * | **pdm_desc** | Output to UF_*free* | string containing the default PDM description for this import operation |

## UF_CLONE_ask_def_pdm_name (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a string containing the default PDM name which will be applied during the import operation. This name will be applied to newly created Items, ItemRevisions or Datasets for which the part being imported has not had its description set by some other method (such as by having a user name applied, or via a convert callback

This setting is ignored for export operations (where it is irrelevant), and for clone operations (where the name of the source part will be used if none has been otherwise set). However it will be recorded in the clone log-file.

### Environment

Internal and External

### See Also
UF_CLONE_convert_cb_t
UF_CLONE_set_def_pdm_name

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_def_pdm_name
(
    char * * pdm_name
)
```

| | | | |
|---|---|---|---|
| char * * | **pdm_name** | Output to UF_*free* | string containing the default PDM name for this import operation |

---

## UF_CLONE_ask_def_validation_options (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default validation related options

### Environment
Internal and External

### See Also
UF_CLONE_set_def_validation_options
UF_CLONE_validation_opts_t

### History
Originally released in V5.0

### Required License(s)
assemblies

```
int UF_CLONE_ask_def_validation_options
(
    UF_CLONE_validation_opts_p_t * validation_options
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_validation_opts_p_t * | **validation_options** | Output to UF_*free* | Default validation options<br>Must be freed using<br>UF_CLONE_free_validation_options |

---

## UF_CLONE_ask_dryrun (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns the value of the dryrun flag for the current operation.
If this flag is true, then when UF_CLONE_perform_clone is called, a log file
will be written but the operation will not actually be performed. A log
file must be specified for the dryrun to work.

### Environment

Internal and External

### See Also

UF_CLONE_set_dryrun
UF_CLONE_perform_clone
UF_CLONE_set_logfile

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_dryrun
(
    logical * dryrun
)
```

| | | | |
|---|---|---|---|
| logical * | **dryrun** | Output | true for a dryrun, false if the operation will actually take place |

---

## UF_CLONE_ask_family_treatment (view source)

**Defined in: uf_clone.h**

### Overview

This routine asks what the treatment of Part Family Members by Clone is.

### Environment

Internal and External

### See Also

UF_CLONE_set_family_treatment

### History

Originally released in V18.0

### Required License(s)

gateway

```
int UF_CLONE_ask_family_treatment
(
    UF_CLONE_family_treatment_t * treatment
)
```

| | | |
|---|---|---|
| UF_CLONE_family_treatment_t * | **treatment** | Output |

---

## UF_CLONE_ask_group (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the group that will be applied to the Item, ItemRevision and Dataset created if needed for this part in this operation. If this is the empty string, then the group under which the executing user is logged in will be used. If the Item, Itemrevision or dataset already exist then they will not be affected.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_group

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_group
(
    const char * input_part_name,
    char * * group
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import,<br>an O/S filename; in NX Manager or for export, a CLI name |
| char * * | **group** | Output to UF_*free* | the group ownership which will be assigned to Items, ItemRevisions and datasets created for this part, or the empty string |

---

## UF_CLONE_ask_item_type (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the item type that has been set for this part in the current operation. It may be the empty string, in which case the default behaviour will be applied. No attempt is made to verify that the item type exists in the database, if it does not, then again the default behaviour will be applied.

See UF_CLONE_ask_def_item_type for a description of the default behaviour.

### Environment

Internal and External

### See Also

UF_CLONE_set_item_type
UF_CLONE_set_def_item_type
UF_CLONE_ask_def_item_type
UF_CLONE_convert_cb_t

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_item_type
(
    const char * input_part_name,
    char * * item_type
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| char * * | **item_type** | Output to UF_*free* | The item type that has been applied to the part |

## UF_CLONE_ask_logfile (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a string containing the name of the log file that will be written when UF_CLONE_perform_clone is called. If the value returned is NULL, no log file has been specified, and so no log file will be written.

### Environment

Internal and External

### See Also

UF_CLONE_set_logfile
UF_CLONE_perform_clone

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_logfile
(
    char * * log_file_name
```

)

| | | | |
|---|---|---|---|
| char * * | **log_file_name** | Output to UF_*free* | the name of the logfile that will be written by this clone operation or NULL if one has not been specified |

---

## UF_CLONE_ask_naming (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the naming technique currently specified for the specified part in the current clone operation. If the naming technique has already been applied (by calling UF_CLONE_set_naming, or UF_CLONE_apply_defaults) then the output name currently applied wil be returned in the output_part_spec argument. If the naming technique has not been applied, then an empty string will be returned in the output_part_spec

### Environment
Internal and External

### See Also
UF_CLONE_set_naming
UF_CLONE_apply_defaults
UF_CLONE_naming_technique_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_naming
(
    const char * input_part_name,
    UF_CLONE_naming_technique_t * naming_technique,
    char * * output_part_spec
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation,<br>in NX Manager clone, or for an export operation , this should be a CLI name, otherwise a native O/S file name |
| UF_CLONE_naming_technique_t * | **naming_technique** | Output | naming technique currently assigned to the specified part |
| char * * | **output_part_spec** | Output to UF_*free* | output part name if set otherwise NULL returned in NX Manager clone, or for an import operation , this will be a CLI |

| | | | |
|---|---|---|---|
| | | | name, otherwise a native O/S file name |

## UF_CLONE_ask_nm_copy (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a list of nonmaster copy options for the specified part.
Each entry contains:
nonmaster_type: the name of the nonmaster type to which this
flag applies, must be freed with UF_free
copy: logical indicating whether it will be copied
next: pointer to next element in list, NULL to end list

each entry must also be freed after use using UF_free

### Environment

Internal and External

### See Also

UF_CLONE_set_def_nm_copy
UF_CLONE_set_nm_copy
UF_CLONE_ask_def_nm_copy

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_ask_nm_copy
(
    const char * input_part_name,
    UF_CLONE_copy_nm_opt_p_t * copy_nonmaster_opts
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation, in NX Manager clone, or for an export operation, this should be a CLI name, otherwise a native O/S file name |
| UF_CLONE_copy_nm_opt_p_t * | **copy_nonmaster_opts** | Output to UF_*free* | list of nonmaster copy options |

## UF_CLONE_ask_ntfy_callbacks (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the currently registered notify callbacks in three parallel arrays containing the callback points, the names of the callbacks and the descriptions of the callbacks.

### Environment
Internal and External

### See Also
UF_CLONE_register_ntfy_callback
UF_CLONE_remove_ntfy_callback

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_ntfy_callbacks
(
    int * n_callbacks,
    UF_CLONE_notify_cb_p_t * points,
    char * * * names,
    char * * * descriptions
)
```

| | | | |
|---|---|---|---|
| int * | **n_callbacks** | Output | number of callbacks registered |
| UF_CLONE_notify_cb_p_t * | **points** | Output to UF_*free* | callback points |
| char * * * | **names** | Output to UF_*free* | n_callbacks (with UF_free_string_array) names of callbacks |
| char * * * | **descriptions** | Output to UF_*free* | n_callbacks (with UF_free_string_array) descriptions of callbacks |

---

## UF_CLONE_ask_operation_class (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the class of the current clone operation

### Environment
Internal and External

### See Also
UF_CLONE_operation_class_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_operation_class
(
    UF_CLONE_operation_class_p_t operation_class
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_operation_class_p_t | **operation_class** | Output | the class of the current clone operation |

---

## UF_CLONE_ask_owner (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the owner that will be applied to the Item, ItemRevision and Dataset created if needed for this part in this operation. If this is the empty string, then the owner under which the executing user is logged in will be used. If the Item, Itemrevision or dataset already exist then they will not be affected.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_group

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_owner
(
    const char * input_part_name,
    char * * owner
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| char * * | **owner** | Output to UF_*free* | the owner that will be used when creating PDM data for this part, or the empty string |

---

## UF_CLONE_ask_part_attrs (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns an array of attributes for specified part from the current clone operation

### Environment
Internal and External

### History
Originally released in NX6.0

### Required License(s)
gateway

```
int UF_CLONE_ask_part_attrs
(
    const char * part_name,
    int * n_attributes,
    UF_ATTR_part_attr_p_t * attributes
)
```

| | | | |
|---|---|---|---|
| const char * | **part_name** | Input | The name of a part in the current clone operation;<br>for native, or import, an O/S filename; in NX Manager or for export, a CLI name |
| int * | **n_attributes** | Output | the number of attributes returned |
| UF_ATTR_part_attr_p_t * | **attributes** | Output to UF_*free* | Allocated array of structures holding attribute titles and values. This must be freed by the caller using UF_free. |

---

## UF_CLONE_ask_part_state (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the state of the specified part in the current clone operation.

### Environment
Internal and External

### See Also
UF_CLONE_part_state_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_part_state
(
    const char * input_part_name,
    UF_CLONE_part_state_p_t state
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for native, or import, an O/S filename; in NX Manager or for export, a CLI name |
| UF_CLONE_part_state_p_t | **state** | Output | the part's state |

---

## UF_CLONE_ask_pdm_desc (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the PDM description that has been set for this part, or the empty string if none has been supplied in which case the default description will be used

### Environment
Internal and External

### See Also
UF_CLONE_set_def_pdm_desc
UF_CLONE_convert_cb_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_pdm_desc
(
    const char * input_part_name,
    char * * pdm_desc
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| char * * | **pdm_desc** | Output to UF_*free* | the description which will be applied or the empty string |

---

## UF_CLONE_ask_pdm_name (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the PDM name that has been set for this part, or the empty string if none has been supplied in which case the default description

will be used

### Environment
Internal and External

### See Also
UF_CLONE_set_def_pdm_name
UF_CLONE_convert_cb_t

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_ask_pdm_name
(
    const char * input_part_name,
    char * * pdm_name
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| char * * | **pdm_name** | Output to UF_*free* | the name which will be applied or the empty string |

---

## UF_CLONE_ask_rev_up (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the rev_up flag for the current operation. It has no meaning outside of the clone import operation.

### Environment
Internal and External

### History
Originally released in NX4.0

### Required License(s)
gateway

```
int UF_CLONE_ask_rev_up
(
    logical * rev_up
)
```

| | | |
|---|---|---|
| logical * | **rev_up** | Input |

## UF_CLONE_ask_validation_abort_option (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the global validation abort options

### Environment
Internal and External

### See Also
UF_CLONE_set_def_validation_options

### History
Originally released in V5.0

### Required License(s)
assemblies

```
int UF_CLONE_ask_validation_abort_option
(
    logical * abort_import
)
```

| | | | |
|---|---|---|---|
| logical * | **abort_import** | Output | whether to abort part import when validation failed |

---

## UF_CLONE_ask_validation_options (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns validation options for the specified part.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_validation_options
UF_CLONE_set_validation_options
UF_CLONE_ask_def_validation_options

### History
Originally released in V5.0

### Required License(s)
assemblies

```
int UF_CLONE_ask_validation_options
(
    const char * input_part_name,
    UF_CLONE_validation_opts_p_t * validation_options
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | Part name of part in the current cloning<br>operation. It is a native O/S file name.<br>Only relevant for import operation<br>in an NX Manager environment. |
| UF_CLONE_validation_opts_p_t * | **validation_options** | Output to UF_*free* | Default validation options<br>Must be freed using<br>UF_CLONE_free_validation_options |

## UF_CLONE_create_log_file (view source)

**Defined in: uf_clone.h**

### Overview

This routine receives the name of an input assembly and the name of
a log file and performs a dryrun clone operation with the logging
output written to the given log file. This is suitable for
generating a basic log file for subsequent editing external to
NX before being used as input to UF_CLONE_execute_log_file

### Environment

Internal and External

### See Also

UF_CLONE_execute_log_file

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_create_log_file
(
    UF_CLONE_operation_class_t operation_class,
    const char * input_name,
    const char * logfile_name,
    UF_CLONE_gen_log_opts_p_t options
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_operation_class_t | **operation_class** | Input | class of operation to execute dryrun for |
| const char * | **input_name** | Input | input assembly |
| const char * | **logfile_name** | Input | name of log file to be generated |
| UF_CLONE_gen_log_opts_p_t | **options** | Input | options to apply to dryrun operation |

## UF_CLONE_ensure_def_directory (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default directory that parts created during a native clone or a clone based export will be placed in. This routine additionally tries to create the specified default directory on the fly if it does not exist.

If this option is never set then the current directory will be used

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_directory

### Required License(s)

assemblies

```
int UF_CLONE_ensure_def_directory
(
    const char * directory_name
)
```

| | | | |
|---|---|---|---|
| const char * | **directory_name** | Input | directory to place created parts in |

---

## UF_CLONE_execute_log_file (view source)

**Defined in: uf_clone.h**

### Overview

This routine receives the name of a log file and an operation class performs a clone operation using the log-file to specify the assemblies to clone and to supply the action and naming operations for the clone operation.

### Environment

Internal and External

### See Also

UF_CLONE_ex_log_opts_p_t

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_execute_log_file
(
    UF_CLONE_operation_class_t operation_class,
    const char * logfile_name,
    UF_CLONE_ex_log_opts_p_t options
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_operation_class_t | **operation_class** | Input | class of operation to perform |
| const char * | **logfile_name** | Input | name of log file to use to specify assemblies to clone |
| UF_CLONE_ex_log_opts_p_t | **options** | Input | options to apply to operation |

## UF_CLONE_free_validation_options (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the default validation related options

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_validation_options
UF_CLONE_ask_validation_options
UF_CLONE_validation_opts_t

### History
Originally released in V5.0

### Required License(s)
assemblies

```
int UF_CLONE_free_validation_options
(
    UF_CLONE_validation_opts_p_t validation_options
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_validation_opts_p_t | **validation_options** | Input | validation options to be freed |

## UF_CLONE_generate_report (view source)

**Defined in: uf_clone.h**

### Overview
This routine calls the report callbacks for each part in the clone operation, possibly multiple times, as follows:

call the begin occurrences report callback.

For each root part, call the report callback pre-order depth
first on the components

call the begin non-occurrences report callback

Then call the report callback for name-only references, and
non-masters that have assigned output names, in an undefined
order

call the end report callback.

For Interactive NX a default callback is registered for
these to generate a report to the listing window. For
non-interactive NX no default callback is registered.

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_generate_report
(
    void
)
```

---

## UF_CLONE_init_log_file_failure (view source)

**Defined in: uf_clone.h**

### Overview
This routine initialises a logfile_failures structure. It should be called
before passing the structure to UF_CLONE_load_logfile

### Environment
Internal and External

### See Also
UF_CLONE_log_file_failure_t
UF_CLONE_load_logfile

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_init_log_file_failure
(
    UF_CLONE_log_file_failure_p_t logfile_failures
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_log_file_failure_p_t | **logfile_failures** | Output to UF_*free* | pointer to structure to be initialised |

## UF_CLONE_init_naming_failures (view source)

**Defined in: uf_clone.h**

### Overview
This routine initialises a naming failures structure. It should be called before passing the structure to UF_CLONE_apply_defaults or UF_CLONE_perform_clone

### Environment
Internal and External

### See Also
UF_CLONE_apply_defaults
UF_CLONE_perform_clone
UF_CLONE_naming_failures_t

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_init_naming_failures
(
    UF_CLONE_naming_failures_p_t failures
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_naming_failures_p_t | **failures** | Output | pointer to structure to be initialised |

## UF_CLONE_initialise (view source)

**Defined in: uf_clone.h**

### Overview
This routine initialises a clone operation. If a clone operation has already been started this routine will return UF_CLONE_err_active.

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_initialise
(
    UF_CLONE_operation_class_t operation_class
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_operation_class_t | **operation_class** | Input | class of operation to initialise |

---

## UF_CLONE_iterate (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the name of the next part in an iteration, or NULL if there are no more. If the end of the iteration is reached, the iteration will have been terminated. To terminate an iteration before reaching the end, call UF_CLONE_stop_iteration()

### Environment
Internal and External

### See Also
UF_CLONE_start_iteration
UF_CLONE_stop_iteration

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_iterate
(
    char * * part_name
)
```

| | | | |
|---|---|---|---|
| char * * | **part_name** | Output to UF_*free* | The name of a part in the current clone operation; for native, or import, an O/S filename; in NX Manager or for export, a CLI name |

---

## UF_CLONE_load_logfile (view source)

**Defined in: uf_clone.h**

### Overview
This routine reads the specified log file and applies the data in it to the current operation.

A log file contains <keyword> <value> pairs, in a defaults section, followed by a sequence of Part: sections. See the Assembly Modelling User Manual or the NX Manager User Manual for a description of how to generate log-files automatically during a clone operation, and of their structure.

Part sections with an action will be added to the clone operation if not already present. Part sections without an action will be added as NameOnly references if not already present.

If any naming failures occur during loading the log file, these will be reported via the naming_failures output argument (which should have been initialised via a call to UF_CLONE_init_naming_failures) and UF_CLONE_err_naming_failures returned. If the logfile is invalid, UF_CLONE_err_invalid_logfile will be returned, and details of the problem will be placed in the logfile_failure output argument (which should be initialised by a call to UF_CLONE_init_log_file_failure).

If processing the log file causes new assemblies to be loaded, then any errors while loading the assembly (parts not found, or warnings about out of sync BVRs) will be reported through the load_status output argument.

If both naming failures and load failures occur, it is not defined which error code the return value of the function will be - if you wish to report both you should check the n_failures member of the naming_failures structure and the failed member of the load_status structure to see if errors have occurred

## Environment

Internal and External

## See Also

UF_CLONE_log_file_failure_t
UF_CLONE_naming_failures_t
UF_PART_load_status_t
UF_CLONE_init_naming_failures
UF_CLONE_init_log_file_failure
UF_CLONE_set_logfile

## History

Originally released in V16.0

## Required License(s)

assemblies

```
int UF_CLONE_load_logfile
(
    const char * log_file_name,
    UF_CLONE_naming_failures_p_t naming_failures,
    UF_CLONE_log_file_failure_p_t logfile_failure,
    UF_PART_load_status_p_t load_status
)
```

| | | | |
|---|---|---|---|
| const char * | **log_file_name** | Input | log file to load |
| UF_CLONE_naming_failures_p_t | **naming_failures** | Input / Output | naming failures if any |
| UF_CLONE_log_file_failure_p_t | **logfile_failure** | Output to UF_*free* | details of the log file invalidity |
| UF_PART_load_status_p_t | **load_status** | Output to UF_*free* | reports any failures to load parts |

## UF_CLONE_part_under_specified (view source)

**Defined in: uf_clone.h**

### Overview

This routine returns a logical indicating whether the part in question is lacking necessary information. After a successful call to UF_CLONE_apply_defaults this should be true for no parts in the operation.

It should not be necessary to call this function during an operation, but it might be useful in order to give feedback to the user, for example a notify callback registered against UF_CLONE_beg_apply_defs_cb might iterate the parts in the clone calling this function in order to count those parts which need defaults applying, (since defaults application can take some time for an NX Manager clone or import, where accesses to the database are required to fill in values)

### Environment

Internal and External

### History

Originally released in V16.0

### Required License(s)

assemblies

**int UF_CLONE_part_under_specified**
**(**
**const char * part_name,**
**logical * is_under_specified**
**)**

| | | | |
|---|---|---|---|
| const char * | **part_name** | Input | name of component |
| logical * | **is_under_specified** | Output | true if node is underspecified, false otherwise |

## UF_CLONE_perform_clone (view source)

**Defined in: uf_clone.h**

### Overview

This routine performs the clone, if necessary completing defaults application first. Defaults application may generate naming rule failures, in which case the clone will not be performed, the failures reported through the naming_failures argument, and UF_CLONE_err_naming_failures will be returned

### Environment

Internal and External

### See Also

UF_CLONE_naming_failures_t
UF_CLONE_init_naming_failures

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_perform_clone
(
    UF_CLONE_naming_failures_p_t naming_failures
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_naming_failures_p_t | **naming_failures** | Input / Output | naming failures if any |

---

## UF_CLONE_register_cvt_callback (view source)

**Defined in: uf_clone.h**

### Overview
This routine registers a callback which will be called when the clone operation needs a string value for a piece of information, for example a user name, an associated file directory or an item type. The convert callback list will be called in order before other default behaviours are applied.

If a callback of the same name is already registered at this callback point, then it will be removed before the function supplied is registered. Names are case independent.

The relative_callback and before_relative arguments can be used to control where the supplied callback function is placed in the list.

If the function returns UF_CLONE_use_supplied, then the callback function should have filled in its answer argument with the needed information. Callback functions later in the list will not be called.

If the function returns UF_CLONE_not_converted then the next function in the callback list will be called, until the list runs out, at which point the default behaviour will be applied

If the function returns UF_CLONE_no_conversion, then no further callback functions in the list will be called, and the default behaviour will be applied.

### Environment
Internal and External

### See Also
UF_CLONE_remove_cvt_callback
UF_CLONE_ask_cvt_callbacks
UF_CLONE_convert_cb_t

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_register_cvt_callback
(
    UF_CLONE_convert_cb_t cb,
    UF_CLONE_convert_callback_t callback,
    char * name,
    char * description,
    char * relative_callback,
    logical before_or_after_relative
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_convert_cb_t | **cb** | Input | callback point at which to call function being registered |
| UF_CLONE_convert_callback_t | **callback** | Input | function pointer of the callback function |
| char * | **name** | Input | the name of this callback - should be unique to this callback point |
| char * | **description** | Input | A description of the callback's function |
| char * | **relative_callback** | Input | The name of a callback to position this one relative to, can be NULL, meaning the entire list of callbacks it this point |
| logical | **before_or_after_relative** | Input | Whether this callback should be placed before the relative callback (if true) or after it (if false). If the relative callback is null, true places the new callback at the beginning of the list, false at the end |

## UF_CLONE_register_ntfy_callback (view source)

**Defined in: uf_clone.h**

### Overview

This routine registers a callback function to be called at the notify callback points specified by the cb argument. The name argument is used to identify the callback for later removal, or insertion of other callback functions before or after this function in the callback list. The name should therefore be unique (for this particular callback point). It is recommended that the name should be prefixed with a site specific code. Some callbacks are registered by NX itself, and the names of these all begin with "UGS". Names are considered to be case independent- i.e. "UGS NOTIFY" is the same as "UGS Notify"

If a callback of the same name is already registered at this call back point, then it will be removed before the function supplied is registered.

The Clone operation will call these callbacks in the order in which they are present in the list at the appropriate point in the clone operation. The callback function should generally return UF_CLONE_continue. Calling further callbacks in the list may be prevented by returning UF_CLONE_cut.

Most of the notify callback points have begin and end callbacks: begin (..._beg_...) callbacks will be called before an operation is about to be performed, allowing a callback to check and forbid it, or perform necessary pre-actions; end callbacks will be called after an operation has completed succesfully, allowing a callback to perform necessary post-actions. Note the end callback corresponding to a begin callback may not be called if the operation in question fails for some reason.

It is possible to specify that the supplied callback should be placed before or after another named callback using the relative_callback and before_relative arguments. If the specified relative callback is not present, the function will return the error UF_CLONE_err_no_relative_cb and place the callback being registered at the beginning or end of the callback list depending on the value of the before_relative flag.

### Environment
Internal and External

### See Also
UF_CLONE_notify_response_t
UF_CLONE_notify_cb_t
UF_CLONE_ask_ntfy_callbacks
UF_CLONE_remove_ntfy_callback

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_register_ntfy_callback
(
    UF_CLONE_notify_cb_t cb,
    UF_CLONE_notify_callback_t callback,
    const char * name,
    const char * description,
    const char * relative_callback,
    logical before_relative
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_notify_cb_t | **cb** | Input | callback point at which to call function being registered |
| UF_CLONE_notify_callback_t | **callback** | Input | function pointer of the callback function |
| const char * | **name** | Input | the name of this callback - should be unique to this callback point |
| const char * | **description** | Input | A description of the callback's function, for information only |

| | | | |
|---|---|---|---|
| const char * | **relative_callback** | Input | The name of a callback to position this one relative to, can be NULL, meaning the entire list of callbacks at this point. |
| logical | **before_relative** | Input | Whether this callback should be placed before the relative callback (if true) or after it (if false). If the relative callback is NULL or not present, true places the new callback at the beginning of the list, after at the end |

## UF_CLONE_remove_cvt_callback (view source)

**Defined in: uf_clone.h**

### Overview
This routine removes the callback of the specified name from the given callback list and returns a pointer to the function that was registered under that name.

### Environment
Internal and External

### See Also
UF_CLONE_notify_response_t
UF_CLONE_notify_cb_t
UF_CLONE_ask_cvt_callbacks
UF_CLONE_register_cvt_callback

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_remove_cvt_callback
(
    UF_CLONE_convert_cb_t cb,
    const char * name,
    UF_CLONE_convert_callback_t * callback_removed
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_convert_cb_t | **cb** | Input | callback point from which to remove function |
| const char * | **name** | Input | name of callback to remove |
| UF_CLONE_convert_callback_t * | **callback_removed** | Output | the function that was removed, so you can reinstall it if desired may be NULL if no callback of that name was registered |

## UF_CLONE_remove_ntfy_callback (view source)

**Defined in: uf_clone.h**

### Overview

This routine removes the callback of the specified name from the given callback list and returns a pointer to the function that was registered under that name.

### Environment

Internal and External

### See Also

UF_CLONE_notify_response_t
UF_CLONE_notify_cb_t
UF_CLONE_ask_ntfy_callbacks
UF_CLONE_register_ntfy_callback

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_remove_ntfy_callback
(
    UF_CLONE_notify_cb_t cb,
    char * name,
    UF_CLONE_notify_callback_t * callback_removed
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_notify_cb_t | **cb** | Input | callback point from which to remove function |
| char * | **name** | Input | name of callback to remove |
| UF_CLONE_notify_callback_t * | **callback_removed** | Output | the function that was removed, so you can reinstall it if desired, may be NULL if no callback of that name was registered at the specified callback point |

## UF_CLONE_reset_to_default (view source)

**Defined in: uf_clone.h**

### Overview

This routine Resets the state of the current clone operation so that no parts have exceptions or defaults applied to them, but leaving all the parts already in the clone operation present.

### Environment
Internal and External

### See Also
UF_CLONE_unapply_defaults

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_reset_to_default
(
    void
)
```

---

## UF_CLONE_set_action (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the action for the specified part to be the given action.
In a clone operation this may result in this action being cascaded to parent parts (if it is a clone action) or child parts (if it is a retain action) whose action is currently UF_CLONE_default_action. (See the section "resolving conflicts" in the "Cloning Assemblies" chapter of the Assemblies User Manual for details of this behaviour.)

If the action being applied is UF_CLONE_replace, you must specify the name of the replacement part, otherwise this argument should be null.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_action

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_action
(
    const char * input_part_name,
    UF_CLONE_action_t action_type,
    const char * replacement_part
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | name of the part for which the action is to be set, in NX Manager or for export a CLI form name, otherwise a native file name |
| UF_CLONE_action_t | **action_type** | Input | action to be applied |
| const char * | **replacement_part** | Input | name of the replacement part if action is UF_CLONE_replace, otherwise null in NX Manager or for export a CLI form name, otherwise a native file name |

---

## UF_CLONE_set_assoc_file_copy (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets a logical for the specified part in the current clone operation indicating whether associated files should by default be copied (for a clone), exported or imported in the current operation.

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_assoc_file_copy
(
    const char * input_part_name,
    logical copy_associated
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation, in NX Manager clone, or for an export operation, this should be a CLI name, otherwise a native O/S file name |
| logical | **copy_associated** | Input | whether to copy associated files for this part |

---

## UF_CLONE_set_assoc_file_dir (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the directory in which associated files for the specified part will be placed, if relative, then relative to the associated file root

directory

### Environment
Internal and External

### See Also
UF_CLONE_set_assoc_file_root_dir
UF_CLONE_ask_assoc_file_root_dir
UF_CLONE_ask_asso_file_dir

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_assoc_file_dir
(
    const char * input_part_name,
    const char * assoc_file_dir
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import,<br>an O/S filename; for export, a CLI name |
| const char * | **assoc_file_dir** | Input | directory to be used, a native file specification, relative or absolute |

---

## UF_CLONE_set_assoc_file_root_dir (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the root directory below which part specific associated file directories will be placed (for export) or lookd for (for import)

### Environment
Internal and External

### See Also
UF_CLONE_ask_assoc_file_root_dir

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_assoc_file_root_dir
(
    const char * root_directory
)
```

| | | | |
|---|---|---|---|
| const char * | **root_directory** | Input | root directory to be used, a native file specification |

---

## UF_CLONE_set_attach_log_file (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the attach_log_file flag for the current operation.
It has no meaning outside of the clone import operation.

### Environment
Internal and External

### History
Originally released in NX5.1

### Required License(s)
assemblies

```
int UF_CLONE_set_attach_log_file
(
    logical attach_log_file
)
```

| | | |
|---|---|---|
| logical | **attach_log_file** | Input |

---

## UF_CLONE_set_ci (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the checkin options to be used for the
specified part in the current import operation.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_ci
UF_CLONE_set_def_ci
UF_CLONE_ask_ci

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_ci
(
    const char * input_part_name,
    UF_CLONE_checkin_data_p_t checkin_data
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | the name of a part in the current import operation, a native O/S filename |
| UF_CLONE_checkin_data_p_t | **checkin_data** | Input | the checkin options to be applied |

---

## UF_CLONE_set_ci_comment_checking (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the comment checking behaviour on checkin for the specified part.

### Return
Return Code:
=0: success
>0: failure code

### Environment
Internal or External

### History
Originally released in v19.0

### Required License(s)
assemblies

```
int UF_CLONE_set_ci_comment_checking
(
    const char * input_part_name,
    logical error_unless_comments_match,
    const char * comment
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of the part in the current export operation, a CLI name |
| logical | **error_unless_comments_match** | Input | Whether to report an error if the checkout comment<br>does not match the given comment |
| const char * | **comment** | Input | string containing the comment to be checked |

---

## UF_CLONE_set_clone_related_cae (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the option to clone/export the related CAE parts when cloning any CAD parts
It has no meaning in the Clone Import Operation

### Environment

Internal and External

### History

Originally released in NX9.0

### Required License(s)

assemblies

```
int UF_CLONE_set_clone_related_cae
(
    UF_CLONE_clone_rel_cae_t rel_cae
)
```

| | | |
|---|---|---|
| UF_CLONE_clone_rel_cae_t | **rel_cae** | Input |

## UF_CLONE_set_clone_related_dwgs (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the boolean option to clone/export the related drawings when cloning any CAD parts
It has no meaning in the Clone Import Operation

### Environment

Internal and External

### History

Originally released in NX9.0

### Required License(s)

assemblies

```
int UF_CLONE_set_clone_related_dwgs
(
    logical rel_dwgs
)
```

| | | |
|---|---|---|
| logical | **rel_dwgs** | Input |

## UF_CLONE_set_co (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the checkout options to be used for the specified part in the current export operation.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_co
UF_CLONE_ask_def_co
UF_CLONE_ask_co
UF_CLONE_checkout_data_t

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_co
(
    const char * input_part_name,
    UF_CLONE_checkout_data_p_t checkout_data
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of the part in the current export operation, a CLI name |
| UF_CLONE_checkout_data_p_t | **checkout_data** | Input | checkout options to be applied |

## UF_CLONE_set_def_action (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default action for the current clone operation, an action inappropriate to the current operation will return an error. UF_CLONE_replace may not be used as a default action.

### Environment
Internal and External

### See Also
UF_CLONE_set_action
UF_CLONE_action_t

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_action
(
    UF_CLONE_action_t action
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_action_t | **action** | Input | any action except UF_CLONE_default_action and UF_CLONE_replace |

---

## UF_CLONE_set_def_assoc_file_copy (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets a logical in the current clone operation indicating whether associated files should by default be copied (for a clone), exported or imported in the current operation.

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_assoc_file_copy
(
    logical copy_associated
)
```

| | | | |
|---|---|---|---|
| logical | **copy_associated** | Input | whether to copy associated files by default |

---

## UF_CLONE_set_def_ci (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default checkin options to be used in the current export operation.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_ci
UF_CLONE_set_ci
UF_CLONE_ask_ci

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_ci
(
    UF_CLONE_checkin_data_p_t checkin_data
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_checkin_data_p_t | **checkin_data** | Input | checkin options to be applied |

---

## UF_CLONE_set_def_ci_comment_checking (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default comment checking behaviour on checkin for the import operation

### Return
Return Code:
=0: success
>0: failure code

### Environment
Internal or External

### History
Originally released in v19.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_ci_comment_checking
(
    logical error_unless_comments_match,
    const char * comment
)
```

| | | | |
|---|---|---|---|
| logical | **error_unless_comments_match** | Input | Whether to report an error if the checkout comment<br>does not match the given comment |
| const char * | **comment** | Input | string containing the comment to be checked |

---

## UF_CLONE_set_def_co (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default checkout options to be used for the current export operation.

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_co
UF_CLONE_ask_co
UF_CLONE_set_co
UF_CLONE_checkout_data_t

### History

Originally released in V16.0

### Required License(s)

gateway

```
int UF_CLONE_set_def_co
(
    UF_CLONE_checkout_data_p_t checkout_data
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_checkout_data_p_t | **checkout_data** | Input | checkout options to be applied |

---

## UF_CLONE_set_def_directory (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default directory that parts created during a native clone or a clone based export will be placed in.

If this option is never set then the current directory will be used.
Note that this option is ignored for user name numbering technique.

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_directory

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_def_directory
(
    const char * directory_name
)
```

| | | | |
|---|---|---|---|
| const char * | **directory_name** | Input | directory to place created parts in |

---

## UF_CLONE_set_def_folder (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the value of the default folder that parts generated during
a NX Manager clone or clone based import will be placed in.

The folder name should be specified as
<user-name>:[<containing-folder>:...]<folder-name>
e.g. "smith:Imported Parts:machinehead" or "smith:My Parts"

If this option is never set, the executing users home
folder will be used.

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_folder

### History

Originally released in V16.0

### Required License(s)

assemblies

**int UF_CLONE_set_def_folder**
**(**
**const char * folder_name**
**)**

| | | | |
|---|---|---|---|
| const char * | **folder_name** | Input | The default folder to place parts in |

---

## UF_CLONE_set_def_group (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default group that will be applied to Items,
ItemRevisions and Datasets created during this operation. If this is the empty
string, then the group under which the executing user is logged in will be used.

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_group

### History
Originally released in V16.0

### Required License(s)
assemblies

**int UF_CLONE_set_def_group**
**(**
**const char * group**
**)**

| | | | |
|---|---|---|---|
| const char * | **group** | Input | the group ownership which will be assigned by default to Items, ItemRevisions and datasets created by this clone operation |

---

## UF_CLONE_set_def_item_type (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default PDM item type that will be used in this import operation. See UF_CLONE_ask_item_type for details of how this will be applied

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_item_type
UF_CLONE_set_item_type
UF_CLONE_ask_item_type

### History
Originally released in V16.0

### Required License(s)
assemblies

**int UF_CLONE_set_def_item_type**
**(**
**const char * item_type**
**)**

| | | | |
|---|---|---|---|
| const char * | **item_type** | Input | item type to be used by default |

---

## UF_CLONE_set_def_naming (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default naming technique for the current clone operation.
See UF_CLONE_ask_def_naming for details of how this will be applied.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_naming
UF_CLONE_set_naming
UF_CLONE_ask_naming

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_naming
(
    UF_CLONE_naming_technique_t naming_technique
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_naming_technique_t | **naming_technique** | Input | any naming_technique except UF_CLONE_default_naming |

---

## UF_CLONE_set_def_nm_copy (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the nonmaster copy option for the nonmaster types specified in the input list. Each list entry contains:
nonmaster_type: the name of the nonmaster type to which the entry applies
copy: logical indicating whether it will be copied
next: pointer to next element in list, NULL to end list

### Environment
Internal and External

### See Also
UF_CLONE_set_def_nm_copy

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_nm_copy
(
    UF_CLONE_copy_nm_opt_p_t copy_nonmaster_opts
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_copy_nm_opt_p_t | **copy_nonmaster_opts** | Input | list of default nonmaster copy options |

---

## UF_CLONE_set_def_owner (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default owner that will be applied to Items, ItemRevisions and Datasets created during this operation. If this is the empty string, then the user under which the executing user is logged in will be used.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_owner

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_owner
(
    const char * owner
)
```

| | | | |
|---|---|---|---|
| const char * | **owner** | Input | default owner that will be assigned to Items, ItemRevisions and Datasets created during this operation |

---

## UF_CLONE_set_def_pdm_desc (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default description which will be applied during the import operation. This description will be applied to newly created Items and ItemRevisions for which the part being imported has not had its description set by some other method (such as by having a user name applied, or via a convert callback).

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_pdm_desc
UF_CLONE_ask_pdm_desc

UF_CLONE_set_pdm_desc

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_pdm_desc
(
    const char * pdm_desc
)
```

| | | | |
|---|---|---|---|
| const char * | **pdm_desc** | Input | description to be applied by default |

---

## UF_CLONE_set_def_pdm_name (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the default name which will be applied during the import operation. See UF_CLONE_ask_def_pdm_name for details of how this will be applied.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_pdm_name
UF_CLONE_ask_pdm_name
UF_CLONE_set_pdm_name

### History
Originally released in V16.0

### Required License(s)
gateway

```
int UF_CLONE_set_def_pdm_name
(
    const char * pdm_name
)
```

| | | | |
|---|---|---|---|
| const char * | **pdm_name** | Input | the name to be applied |

---

## UF_CLONE_set_def_validation_options (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the default validation related options

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_validation_options

### History
Originally released in NX5.0

### Required License(s)
assemblies

```
int UF_CLONE_set_def_validation_options
(
    UF_CLONE_validation_opts_p_t validation_options
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_validation_opts_p_t | **validation_options** | Input | default validation options |

## UF_CLONE_set_dryrun (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the dryrun flag for the current operation. See UF_CLONE_ask_dryrun for details of how this behaves

### Environment
Internal and External

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_dryrun
(
    logical dryrun
)
```

| | | | |
|---|---|---|---|
| logical | **dryrun** | Input | true for a dryrun, false for the clone to actually take place |

## UF_CLONE_set_family_treatment (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the treatment of Part Family Members by Clone.

UF_CLONE_treat_as_lost: family members are regarded as lost parts.
The operation will proceed.
UF_CLONE_strip_family_status: family members are imported/exported as
normal parts: i.e. if you import a part
family member, the original NX part remains
a part family member, but NX Manager part
is not.
UF_CLONE_give_error: the operation will return an error if it
encounters a part family member and will
not proceed.

### Environment

Internal and External

### History

Originally released in V18.0

### Required License(s)

assemblies

```
int UF_CLONE_set_family_treatment
(
    UF_CLONE_family_treatment_t treatment
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_family_treatment_t | **treatment** | Input | treatment to be applied to part family members when added toclone operations. |

---

## UF_CLONE_set_group (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the group that will be applied to the Item, ItemRevision and Dataset created if needed for this part in this operation. If this is the empty string, then the group under which the executing user is logged in will be used. If the Item, Itemrevision or dataset already exist then they will not be affected.

### Environment

Internal and External

### See Also

UF_CLONE_ask_def_group

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_group
(
    const char * input_part_name,
    const char * group
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| const char * | **group** | Input | the group ownership which will be assigned to Items, ItemRevisions and datasets created for this part |

---

## UF_CLONE_set_item_type (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the item type to be used for this part in the current operation. See UF_CLONE_ask_item_type for details of how this will be applied

### Environment
Internal and External

### See Also
UF_CLONE_ask_item_type
UF_CLONE_set_def_item_type
UF_CLONE_ask_def_item_type
UF_CLONE_convert_cb_t

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_item_type
(
    const char * input_part_name,
    const char * item_type
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| const char * | **item_type** | Input | The item type that should be applied to the part |

---

## UF_CLONE_set_logfile (view source)

**Defined in: uf_clone.h**

### Overview

This routine specified the name of the log-file to be used to record this operation. If the value specified is NULL, no logfile will be written.

The log file will be attached as a named reference to any root parts (whether input ot output) that are stored in the PDM.

### Environment

Internal and External

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_logfile
(
    const char * log_file_name
)
```

| | | | |
|---|---|---|---|
| const char * | **log_file_name** | Input | name of logfile to be written (an O/S file name, even in NX Manager) |

---

## UF_CLONE_set_name_rule (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the name rule for the clone operation. If a previously existing name rule has already been applied to any parts (via apply_defaults or setting exceptions) then the new name rule will be applied to those parts; any name rule applications which fail wil be reported via the name rule failures argument and UF_CLONE_err_naming_failures will be returned

### Environment

Internal and External

### See Also

UF_CLONE_name_rule_def_t
UF_CLONE_naming_failures_t
UF_CLONE_name_rule_type_t
UF_CLONE_init_naming_failures

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_name_rule
(
    UF_CLONE_name_rule_def_p_t name_rule,
    UF_CLONE_naming_failures_p_t naming_failures
)
```

| | | | |
|---|---|---|---|
| UF_CLONE_name_rule_def_p_t | **name_rule** | Input | structure describing a name rule |
| UF_CLONE_naming_failures_p_t | **naming_failures** | Input / Output | naming failures if any |

---

## UF_CLONE_set_naming (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the naming technique to be applied to the specified part
in the current clone operation. If the naming technique is UF_CLONE_user_name,
then an output part name (the user name) must also be supplied, otherwise
this argument should be NULL;

### Environment

Internal and External

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_naming
(
    const char * input_part_name,
    UF_CLONE_naming_technique_t naming_technique,
    const char * output_part_name
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation, in NX Manager clone, or for an export operation , this should be an encoded name, otherwise a native O/S file name |
| UF_CLONE_naming_technique_t | **naming_technique** | Input | naming technique to be assigned to the specified part |
| const char * | **output_part_name** | Input | output part name to be set if naming_technique is UF_CLONE_user_name, otherwise NULL. In NX Manager clone, or for an import operation, this |

| | | | |
|---|---|---|---|
| | | | should be an encoded name,<br>otherwise a native O/S<br>file name |

## UF_CLONE_set_nm_copy (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the nonmaster copy option for the nonmaster types specified
in the input list for the specified part.
Each list entry contains:
nonmaster_type: the name of the nonmaster type to which the entry applies
copy: logical indicating whether it will be copied
next: pointer to next element in list, NULL to end list

### Environment
Internal and External

### See Also
UF_CLONE_set_def_nm_copy
UF_CLONE_ask_nm_copy
UF_CLONE_ask_def_nm_copy

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_nm_copy
(
    const char * input_part_name,
    UF_CLONE_copy_nm_opt_p_t copy_nonmaster_opts
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | part name of part in the current cloning operation, in NX Manager clone, or for an<br>export operation, this should be a CLI name,<br>otherwise a native O/S file name |
| UF_CLONE_copy_nm_opt_p_t | **copy_nonmaster_opts** | Input | list of nonmaster copy options |

## UF_CLONE_set_owner (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the owner that will be applied to the Item, ItemRevision and
Dataset created if needed for this part in this operation. If this is the

empty string, then the owner under which the executing user is logged in will be used. If the Item, Itemrevision or dataset already exist then they will not be affected.

### Environment
Internal and External

### See Also
UF_CLONE_ask_def_group

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_owner
(
    const char * input_part_name,
    const char * owner
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| const char * | **owner** | Input | the owner to be used when creating PDM data for this part |

---

## UF_CLONE_set_pdm_desc (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the PDM description to be applied to the specified part in the current operation.

### Environment
Internal and External

### See Also
UF_CLONE_ask_pdm_desc
UF_CLONE_set_def_pdm_desc
UF_CLONE_ask_def_pdm_desc

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_set_pdm_desc
(
    const char * input_part_name,
    const char * pdm_desc
```

)

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import,<br>an O/S filename; in NX Manager or for export, a CLI name |
| const char * | **pdm_desc** | Input | the description to be applied |

---

## UF_CLONE_set_pdm_name (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the PDM name to be applied to the specified part in the current operation.

### Environment

Internal and External

### See Also

UF_CLONE_ask_pdm_name
UF_CLONE_set_def_pdm_name
UF_CLONE_ask_def_pdm_name

### History

Originally released in V16.0

### Required License(s)

assemblies

```
int UF_CLONE_set_pdm_name
(
    const char * input_part_name,
    const char * pdm_name
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | The name of a part in the current clone operation; for import, an O/S filename; in NX Manager or for export, a CLI name |
| const char * | **pdm_name** | Input | the name to be applied or the empty string |

---

## UF_CLONE_set_propagate_actions (view source)

**Defined in: uf_clone.h**

### Overview

This routine sets the propagation behaviour for actions in an export operation (for Clone propagation is always on, and it

is not relevant to an import operation.
When propagation is on:
specifying an Exclude action for a part will cause all parts
it references that are not referenced by other non-excluded
parts to also have the Exclude action applied, and so on for their referenced parts
specifying an Overwrite or Use Existing action for a part will cause all parts
which reference it which currently have an Exclude action applied (directly or by default)
to have the same Overwrite or Use Existing action applied to them, and so on for their
referencing parts

### Return
Return Code:
= 0 : success
UF_CLONE_err_invalid_operation : the current operation is not an export operation
> 0 : other failure code

### Environment
Internal or external

### See Also
UF_CLONE_set_def_action
: UF_CLONE_set_action

### History
Originally released in v19.0

### Required License(s)
assemblies

```
int UF_CLONE_set_propagate_actions
(
    logical propagate_actions
)
```

| | | | |
|---|---|---|---|
| logical | **propagate_actions** | Input | whether to propagate actions in this export operation |

---

## UF_CLONE_set_rev_up (view source)

**Defined in: uf_clone.h**

### Overview
This routine returns the rev_up flag for the current operation. It has no
meaning outside of the clone import operation.

### Environment
Internal and External

### History
Originally released in NX4.0

### Required License(s)
assemblies

```
int UF_CLONE_set_rev_up
(
    logical rev_up
)
```

| | | |
|---|---|---|
| logical | **rev_up** | Input |

## UF_CLONE_set_validation_abort_option (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the global validation abort options

### Environment
Internal and External

### See Also
UF_CLONE_ask_validation_abort_option

### History
Originally released in NX5.0

### Required License(s)
assemblies

```
int UF_CLONE_set_validation_abort_option
(
    logical abort_import
)
```

| | | | |
|---|---|---|---|
| logical | **abort_import** | Input | whether to abort part import when validation failed |

## UF_CLONE_set_validation_options (view source)

**Defined in: uf_clone.h**

### Overview
This routine sets the validation option for for the specified part.

### Environment
Internal and External

### See Also
UF_CLONE_set_def_validation_options
UF_CLONE_ask_validation_options
UF_CLONE_ask_def_validation_options

### History
Originally released in V5.0

### Required License(s)
gateway

```
int UF_CLONE_set_validation_options
(
    const char * input_part_name,
    UF_CLONE_validation_opts_p_t validation_options
)
```

| | | | |
|---|---|---|---|
| const char * | **input_part_name** | Input | Part name of part in the current cloning operation. It is a native O/S file name. Only relevant for import operation in an NX Manager environment. |
| UF_CLONE_validation_opts_p_t | **validation_options** | Input | validation options |

---

## UF_CLONE_start_iteration (view source)

**Defined in: uf_clone.h**

### Overview
This routine initialises an iteration over the parts currently in the clone operation. While an iteration is in progress, parts or assemblies should not be added to the clone operation, or the iterator's behaviour is undefined, functions which may add parts to the operation will therefore return UF_CLONE_err_iterator_active.

### Environment
Internal and External

### See Also
UF_CLONE_stop_iteration
UF_CLONE_iterate

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_start_iteration
(
    void
)
```

---

## UF_CLONE_stop_iteration (view source)

**Defined in: uf_clone.h**

### Overview
This routine stops an iteration. It is only necessary to call this routine if an iteration is to be stopped before all parts have been iterated over

### Environment
Internal and External

### See Also
UF_CLONE_start_iteration
UF_CLONE_iterate

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_stop_iteration
(
    void
)
```

---

## UF_CLONE_terminate (view source)

**Defined in: uf_clone.h**

### Overview
This routine terminates the current clone operation if there is one, returns no error if there is not

### Environment
Internal and External

### See Also
UF_CLONE_initialise

### History
Originally released in V16.0

### Required License(s)
assemblies

```
int UF_CLONE_terminate
(
    void
)
```

---

## UF_CLONE_unapply_defaults (view source)

**Defined in: uf_clone.h**

### Overview
This routine removes previously applied defaults, but leaves exceptions which have been specified intact. Naming failures may occur if a registered notify callback forbids the

operation.

### Environment
Internal and External

### See Also
UF_CLONE_reset_to_default
UF_CLONE_naming_failures_t
UF_CLONE_init_naming_failures

### History
Originally released in V16.0

### Required License(s)
assemblies

**int UF_CLONE_unapply_defaults**
**(**
**UF_CLONE_naming_failures_p_t naming_failures**
**)**

| | | | |
|---|---|---|---|
| UF_CLONE_naming_failures_p_t | **naming_failures** | Input / Output | Pointer to a naming failures structure. If a naming failure occurs the return code will be UF_CLONE_err_naming_failures and this structure will be filled in |