NAME

FN_status_t, fn_status_create, fn_status_destroy, fn_status_copy, fn_status_assign, fn_status_code, fn_status_remaining_name, fn_status_resolved_name, fn_status_resolved_ref, fn_status_diagnostic_message, fn_status_link_code, fn_status_link_remaining_name, fn_status_link_resolved_name, fn_status_link_resolved_ref, fn_status_link_diagnostic_message, fn_status_is_success, fn_status_set_success, fn_status_set, fn_status_set_code, fn_status_set_remaining_name, fn_status_set_resolved_name, fn_status_set_resolved_ref, fn_status_set_diagnostic_message, fn_status_set_link_code, fn_status_set_link_remaining_name, fn_status_set_link_resolved_name, fn_status_set_link_resolved_ref, fn_status_set_link_diagnostic_message, fn_status_append_resolved_name, fn_status_append_remaining_name, fn_status_advance_by_name, fn_status_description – an XFN status object

SYNOPSIS

cc [*flag* ...] *file* ... **–lxfn** [*library* ...]

#include <xfn/xfn.h>

FN_status_t *fn_status_create(void);

void fn_status_destroy(FN_status_t **stat*);

FN_status_t *fn_status_copy(const FN_status_t **stat*);

FN_status_t *fn_status_assign(FN_status_t **dst*, **const FN_status_t** **src*);

unsigned int fn_status_code(const FN_status_t **stat*);

const FN_composite_name_t *fn_status_remaining_name(const FN_status_t **stat*);

const FN_composite_name_t *fn_status_resolved_name(const FN_status_t **stat*);

const FN_ref_t *fn_status_resolved_ref(const FN_status_t **stat*);

const FN_string_t *fn_status_diagnostic_message(const FN_status_t **stat*);

unsigned int fn_status_link_code(const FN_status_t **stat*);

const FN_composite_name_t *fn_status_link_remaining_name(
    const FN_status_t **stat*);

const FN_composite_name_t *fn_status_link_resolved_name(const FN_status_t **stat*);

const FN_ref_t *fn_status_link_resolved_ref(const FN_status_t **stat*);

const FN_string_t *fn_status_link_diagnostic_message(const FN_status_t **stat*);

int fn_status_is_success(const FN_status_t **stat*);

int fn_status_set_success(FN_status_t **stat*);

int fn_status_set(FN_status_t **stat*, **unsigned int** *code*, **const FN_ref_t** **resolved_ref*,
    const FN_composite_name_t **resolved_name*,
    const FN_composite_name_t **remaining_name*);

int fn_status_set_code(FN_status_t **stat*, **unsigned int** *code*);

int fn_status_set_remaining_name(FN_status_t **stat*,
    const FN_composite_name_t **name*);

int fn_status_set_resolved_name(FN_status_t **stat***,**
const FN_composite_name_t **name***);**

int fn_status_set_resolved_ref(FN_status_t **stat***, const FN_ref_t** **ref***);**

int fn_status_set_diagnostic_message(FN_status_t **stat***,**
const FN_string_t **msg***);**

int fn_status_set_link_code(FN_status_t **stat***, unsigned int** *code***);**

int fn_status_set_link_remaining_name(FN_status_t **stat***,**
const FN_composite_name_t **name***);**

int fn_status_set_link_resolved_name(FN_status_t **stat***,**
const FN_composite_name_t **name***);**

int fn_status_set_link_resolved_ref(FN_status_t **stat***, const FN_ref_t** **ref***);**

int fn_status_set_link_diagnostic_message(FN_status_t **stat***,**
const FN_string_t **msg***);**

int fn_status_append_resolved_name(FN_status_t **stat***,**
const FN_composite_name_t **name***);**

int fn_status_append_remaining_name(FN_status_t **stat***,**
const FN_composite_name_t **name***);**

int fn_status_advance_by_name(FN_status_t **stat***,**
const FN_composite_name_t **prefix***, const FN_ref_t** **resolved_ref***);**

FN_string_t ***fn_status_description(const FN_status_t** **stat***,**
unsigned int *detail***, unsigned int** **more_detail***);**

MT-LEVEL

Safe.

DESCRIPTION

The result status of operations in the context interface and the attribute interface is encapsulated in an **FN_status_t** object. This object contains information about how the operation completed: whether an error occurred in performing the operation, the nature of the error, and information that helps locate where the error occurred. In the case that the error occurred while resolving an XFN link, the status object contains additional information about that error.

The context status object consists of several items of information.

primary status code	An **unsigned int** code describing the disposition of the operation.
resolved name	In the case of a failure during the resolution phase of the operation, this is the leading portion of the name that was resolved successfully. Resolution may have been successful beyond this point, but the error might not be pinpointed further.
resolved reference	The reference to which resolution was successful (in other words, the reference to which the resolved name is bound).
remaining name	The remaining unresolved portion of the name.
diagnostic message	This contains any diagnostic message returned by the context implementation. This message provides the context

implementation a way of notifying the end-user or administrator of any implementation-specific information related to the returned error status. The diagnostic message could then be used by the end-user or administrator to take appropriate out-of-band action to rectify the problem.

link status code
: In the case that an error occurred while resolving an XFN link, the primary status code has the value **FN_E_LINK_ERROR** and the link status code describes the error that occurred while resolving the XFN link.

resolved link name
: In the case of a link error, this contains the resolved portion of the name in the XFN link.

resolved link reference
: In the case of a link error, this contains the reference to which the resolved link name is bound.

remaining link name
: In the case of a link error, this contains the remaining unresolved portion of the name in the XFN link.

link diagnostic message
: In the case of a link error, this contains any diagnostic message related to the resolution of the link.

Both the primary status code and the link status code are values of type **unsigned int** that are drawn from the same set of meaningful values. XFN reserves the values 0 through 127 for standard meanings. The values and interpretations for the codes are determined by XFN. See **xfn_status_codes**(3N).

fn_status_create( ) creates a status object with status **FN_SUCCESS. fn_status_destroy( )** releases the storage associated with *stat*. **fn_status_copy( )** returns a copy of the status object *stat*. **fn_status_assign( )** makes a copy of the status object *src* and assigns it to *dst*, releasing any old contents of *dst*. A pointer to the same object as *dst* is returned.

fn_status_code( ) returns the status code. **fn_status_remaining_name( )** returns the remaining part of name to be resolved. **fn_status_resolved_name( )** returns the part of the composite name that has been resolved. **fn_status_resolved_ref( )** returns the reference to which resolution was successful. **fn_status_diagnostic_message** returns any diagnostic message set by the context implementation.

fn_status_link_code( ) returns the link status code. **fn_status_link_remaining_name( )** returns the remaining part of the link name that has not been resolved. **fn_status_link_resolved_name( )** returns the part of the link name that has been resolved. **fn_status_link_resolved_ref( )** returns the reference to which resolution of the link was successful. **fn_status_link_diagnostic_message( )** returns any diagnostic message set by the context implementation during resolution of the link.

fn_status_is_success( ) returns **1** if the status indicates success, **0** otherwise.

fn_status_set_success( ) sets the status code to **FN_SUCCESS** and clears all other parts of *stat*. **fn_status_set( )** sets the non-link contents of the status object *stat*. **fn_status_set_code( )** sets the primary status code field of the status object *stat*.

fn_status_set_remaining_name( ) sets the remaining name part of the status object *stat* to *name*. **fn_status_set_resolved_name( )** sets the resolved name part of the status object *stat* to *name*. **fn_status_set_resolved_ref( )** sets the resolved reference part of the status object*stat* to *ref*. **fn_status_set_diagnostic_message( )** sets the diagnostic message part of the status object to *msg*.

fn_status_set_link_code( ) sets the link status code field of the status object *stat* to indicate why resolution of the link failed. **fn_status_set_link_remaining_name( )** sets the remaining link name part of the status object *stat* to *name*. **fn_status_set_link_resolved_name( )** sets the resolved link name part of the status object *stat* to *name*. **fn_status_set_link_resolved_ref( )** sets the resolved link reference part of the status object *stat* to *ref*. **fn_status_set_link_diagnostic_message( )** sets the link diagnostic message part of the status object to *msg*.

fn_status_append_resolved_name( ) appends as additional components *name* to the resolved name part of the status object *stat*. **fn_status_append_remaining_name( )** appends as additional components *name* to the remaining name part of the status object *stat*. **fn_status_advance_by_name( )** removes *prefix* from the remaining name, and appends it to the resolved name. The resolved reference part is set to *resolved_ref*. This operation returns **1** on success, **0** if the *prefix* is not a prefix of the remaining name.

RETURN VALUE

The **fn_status_set_*( )** operations return **1** if the operation succeeds, **0** if the operation fails.

SEE ALSO

FN_composite_name_t(3N), **FN_ref_t**(3N), **FN_string_t**(3N), **xfn_status_codes**(3N), **xfn**(3N)