NAME

FN_ctx_t – an XFN context

MT-LEVEL

Safe.

DESCRIPTION

An XFN context consists of a set of name to reference bindings. An XFN context is represented by the type **FN_ctx_t** in the client interface. The operations for manipulating an **FN_ctx_t** object are described in detail in separate reference manual pages.

These are the interfaces:

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

#include <xfn/xfn.h>

FN_ctx_t *fn_ctx_handle_from_initial(FN_status_t *status);

FN_ctx_t *fn_ctx_handle_from_ref(const FN_ref_t **ref*, **FN_status_t** **status***);**

FN_ref_t *fn_ctx_get_ref(const FN_ctx_t **ctx*, **FN_status_t** **status***);**

void fn_ctx_handle_destroy(FN_ctx_t **ctx***);**

FN_ref_t *fn_ctx_lookup(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  FN_status_t **status***);**

FN_namelist_t *fn_ctx_list_names(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  FN_status_t **status***);**

FN_string_t *fn_namelist_next(FN_namelist_t **nl*, **FN_status_t** **status***);**

void fn_namelist_destroy(FN_namelist_t **nl*, **FN_status_t** **status***);**

FN_bindinglist_t *fn_ctx_list_bindings(FN_ctx_t **ctx*,
  const FN_composite_name_t **name*, **FN_status_t** **status***);**

FN_string_t *fn_bindinglist_next(FN_bindinglist_t **iter*, **FN_ref_t** ***ref*,
  FN_status_t **status***);**

void fn_bindinglist_destroy(FN_bindinglist_t **iter_pos*, **FN_status_t** **status***);**

int fn_ctx_bind(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  const FN_ref_t **ref*, **unsigned int** *exclusive*, **FN_status_t** **status***);**

int fn_ctx_unbind(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  FN_status_t **status***);**

int fn_ctx_rename(FN_ctx_t **ctx*, **const FN_composite_name_t** **oldname*,
  const FN_composite_name_t **newname*, **unsigned int** *exclusive*,
  FN_status_t **status***);**

FN_ref_t *fn_ctx_create_subcontext(FN_ctx_t **ctx*,
  const FN_composite_name_t **name*, **FN_status_t** **status***);**

int fn_ctx_destroy_subcontext(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  FN_status_t **status***);**

FN_ref_t *fn_ctx_lookup_link(FN_ctx_t **ctx*, **const FN_composite_name_t** **name*,
  FN_status_t **status***);**

FN_attrset_t *fn_ctx_get_syntax_attrs(FN_ctx_t **ctx*,
const FN_composite_name_t **name*, **FN_status_t** **status*);

The following contains a brief summary of these operations.

fn_ctx_handle_from_initial( ) returns a pointer to an Initial Context that provides a starting point for resolution of composite names. **fn_ctx_handle_from_ref( )** returns a handle to an **FN_ctx_t** object using the given reference *ref.* **fn_ctx_get_ref( )** returns the reference of the context *ctx.* **fn_ctx_handle_destroy( )** releases the resources associated with the **FN_ctx_t** object *ctx;* it does not affect the state of the context itself.

fn_ctx_lookup( ) returns the reference bound to *name* resolved relative to *ctx.* **fn_ctx_list_names( )** is used to enumerate the atomic names bound in the context named by *name* resolved relative to *ctx.* **fn_ctx_list_bindings( )** is used to enumerate the atomic names and their references in the context named by *name* resolved relative to *ctx.*

fn_ctx_bind( ) binds the composite name *name* to a reference *ref* resolved relative to *ctx.* **fn_ctx_unbind( )** unbinds *name* resolved relative to *ctx.* **fn_ctx_rename( )** binds *newname* to the reference bound to *oldname* and unbinds *oldname. oldname* is resolved relative to *ctx; newname* is resolved relative to the target context.

fn_ctx_create_subcontext( ) creates a new context with the given composite name *name* resolved relative to *ctx.* **fn_ctx_destroy_subcontext( )** destroys the context named by *name* resolved relative to *ctx.*

Normal resolution always follow links. **fn_ctx_lookup_link( )** lookups up *name* relative to *ctx* , following links except for the last atomic part of *name* , which must be bound to an XFN link.

fn_ctx_get_syntax_attrs( ) returns an attribute set containing attributes that describes a context's syntax. *name* must name a context.

ERRORS

In each context operation, the caller supplies an **FN_status_t** object as a parameter. The called function sets this status object as described in **FN_status_t**(3N) and **xfn_status_codes**(3N).

APPLICATION USAGE

In most of the operations of the base context interface, the caller supplies a context and a composite name. The supplied name is always interpreted relative to the supplied context.

The operation may eventually be effected on a different context called the operation's *target context.* Each operation has an initial resolution phase that conveys the operation to its target context, and the operation is then applied. The effect (but not necessarily the implementation) is that of doing a lookup on that portion of the name that represents the target context, and then invoking the operation on the target context. The contexts involved only in the resolution phase are called *intermediate contexts.*

Normal resolution of names in context operations always follows XFN links.

SEE ALSO

FN_attrset_t(3N), **FN_composite_name_t**(3N), **FN_ref_t**(3N), **FN_status_t**(3N), **fn_ctx_create_subcontext**(3N), **fn_ctx_bind**(3N), **fn_ctx_destroy_subcontext**(3N), **fn_ctx_handle_destroy**(3N), **fn_ctx_handle_from_initial**(3N),

fn_ctx_handle_from_ref(3N), **fn_ctx_get_ref**(3N), **fn_ctx_get_syntax_attrs**(3N), **fn_ctx_list_names**(3N), **fn_ctx_list_bindings**(3N), **fn_ctx_lookup**(3N), **fn_ctx_lookup_link**(3N), **fn_ctx_rename**(3N), **fn_ctx_unbind**(3N), **xfn_links**(3N), **xfn_status_codes**(3N), **xfn**(3N)