NAME

cnvlist_get, **cnvlist_take**, **cnvlist_free** - API for managing name/value pairs by cookie.

LIBRARY

Name/value pairs library (libnv, -lnv)

SYNOPSIS

#include <sys/cnv.h>

*const char **
cnvlist_name(*const void *cookie*);

int
cnvlist_type(*const void *cookie*);

bool
cnvlist_get_bool(*const void *cookie*);

uint64_t
cnvlist_get_number(*const void *cookie*);

*const char **
cnvlist_get_string(*const void *cookie*);

*const nvlist_t **
cnvlist_get_nvlist(*const void *cookie*);

*const void **
cnvlist_get_binary(*const void *cookie, size_t *sizep*);

*const bool **
cnvlist_get_bool_array(*const void *cookie, size_t *nitemsp*);

*const uint64_t **
cnvlist_get_number_array(*const void *cookie, size_t *nitemsp*);

*const char * const **
cnvlist_get_string_array(*const void *cookie, size_t *nitemsp*);

*const nvlist_t * const **

cnvlist_get_nvlist_array(*const void *cookie, size_t *nitemsp*);

int
cnvlist_get_descriptor(*const void *cookie*);

*const int **
cnvlist_get_descriptor_array(*const void *cookie, size_t *nitemsp*);

bool
cnvlist_take_bool(*void *cookie*);

uint64_t
cnvlist_take_number(*void *cookie*);

*const char **
cnvlist_take_string(*void *cookie*);

*const nvlist_t **
cnvlist_take_nvlist(*void *cookie*);

*const void **
cnvlist_take_binary(*void *cookie, size_t *sizep*);

*const bool **
cnvlist_take_bool_array(*void *cookie, size_t *nitemsp*);

*const uint64_t **
cnvlist_take_number_array(*void *cookie, size_t *nitemsp*);

*const char * const **
cnvlist_take_string_array(*void *cookie, size_t *nitemsp*);

*const nvlist_t * const **
cnvlist_take_nvlist_array(*void *cookie, size_t *nitemsp*);

int
cnvlist_take_descriptor(*void *cookie*);

*const int **
cnvlist_take_descriptor_array(*void *cookie, size_t *nitemsp*);

void
cnvlist_free_null(*void *cookie*);

void
cnvlist_free_bool(*void *cookie*);

void
cnvlist_free_number(*void *cookie*);

void
cnvlist_free_string(*void *cookie*);

void
cnvlist_free_nvlist(*void *cookie*);

void
cnvlist_free_descriptor(*void *cookie*);

void
cnvlist_free_binary(*void *cookie*);

void
cnvlist_free_bool_array(*void *cookie*);

void
cnvlist_free_number_array(*void *cookie*);

void
cnvlist_free_string_array(*void *cookie*);

void
cnvlist_free_nvlist_array(*void *cookie*);

void
cnvlist_free_descriptor_array(*void *cookie*);

DESCRIPTION

The **libnv** library permits easy management of name/value pairs and can send and receive them over sockets. For more information, also see nv(9).

The concept of cookies is explained in **nvlist_next**(), **nvlist_get_parent**(), and **nvlist_get_pararr**() from nv(9).

The **cnvlist_name**() function returns the name of an element associated with the given cookie.

The **cnvlist_type**() function returns the type of an element associated with the given cookie. Types which can be returned are described in nv(9).

The **cnvlist_get** family of functions obtains the value associated with the given cookie. Returned strings, nvlists, descriptors, binaries, or arrays must not be modified by the user, since they still belong to the nvlist. The nvlist must not be in an error state.

The **cnvlist_take** family of functions returns the value associated with the given cookie and removes the element from the nvlist. When the value is a string, binary, or array value, the caller is responsible for freeing the returned memory with **free**(*3*). When the value is an nvlist, the caller is responsible for destroying the returned nvlist with **nvlist_destroy**(). When the value is a descriptor, the caller is responsible for closing the returned descriptor with the **close**(*2*).

The **cnvlist_free** family of functions removes an element of the supplied cookie and frees all resources. If an element of the given cookie has the wrong type or does not exist, the program is aborted.

EXAMPLES

The following example demonstrates how to deal with cnvlist API.

```
int type;
void *cookie, *scookie, *bcookie;
nvlist_t *nvl;
char *name;

nvl = nvlist_create(0);
nvlist_add_bool(nvl, "test", 1 == 2);
nvlist_add_string(nvl, "test2", "cnvlist");
cookie = NULL;

while (nvlist_next(nvl, &type, &cookie) != NULL) {
	switch (type) {
	case NV_TYPE_BOOL:
		printf("test: %d\n", cnvlist_get_bool(cookie));
		bcookie = cookie;
		break;
```

```
        case NV_TYPE_STRING:
                printf("test2: %s\n", cnvlist_get_string(cookie));
                scookie = cookie;
                break;
        }
}

name = cnvlist_take_string(scookie);
cnvlist_free_bool(bcookie);

printf("test2: %s\n", name);
free(name);

printf("nvlist_empty = %d\n", nvlist_empty(nvl));
nvlist_destroy(nvl);

return (0);
```

SEE ALSO

close(2), free(3), nv(9)

AUTHORS

The **cnv** API was created during the Google Summer Of Code 2016 by Adam Starak.