NAME

snmp_pdu_add_variable, snmp_varlist_add_variable, snmp_add_null_var, snmp_clone_varbind, snmp_set_var_objid, snmp_set_var_value, snmp_set_var_typed_value, snmp_set_var_typed_integer, print_variable, fprint_variable, snprint_variable, print_value, fprint_value, snprint_value, snmp_free_var, snmp_free_varbind - netsnmp_varbind_api functions

SYNOPSIS

#include <net-snmp/varbind_api.h>

Creation

netsnmp_variable_list *snmp_pdu_add_variable(
netsnmp_pdu ***pdu*,**
const oid ***objid*, size_t** *objidlen***,**
u_char *type***, const void ***value***, size_t** *len***);**

netsnmp_variable_list *snmp_varlist_add_variable(
netsnmp_variable_list ***varlist*,**
const oid ***objid*, size_t** *objidlen***,**
u_char *type***, const void ***value***, size_t** *len***);**

netsnmp_variable_list *snmp_add_null_var(
netsnmp_pdu ***pdu*,**
const oid ***objid*, size_t** *objidlen***);**

netsnmp_variable_list *snmp_clone_varbind(
netsnmp_variable_list ***varlist*);**

Setting Values

int snmp_set_var_objid(netsnmp_variable_list* *variable***,**
const oid * *objid***, size_t** *objidlen***);**

int snmp_set_var_value(netsnmp_variable_list* *variable***,**
const void * *value***, size_t** *vallen***);**

int snmp_set_var_typed_value(netsnmp_variable_list* *variable***,**
u_char *type***,**
const void * *value***, size_t** *vallen***);**

int snmp_set_var_typed_integer(netsnmp_variable_list* *variable***,**
u_char *type***, long** *value***);**

Output

void print_variable(const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

void fprint_variable(FILE * *fp***,**
const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

int snprint_variable(char ***buf*, size_t** *len***,**
const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

void print_value(const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

void fprint_value(FILE * *fp***,**
const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

int snprint_value(char ***buf*, size_t** *len***,**
const oid ***objid*, size_t** *objidlen***,**
const netsnmp_variable_list ***variable*);**

Deletion

void snmp_free_var(netsnmp_variable_list ***variable*);**

void snmp_free_varbind(netsnmp_variable_list ***variables*);**

DESCRIPTION

The functions dealing with variable bindings fall into four groups - dealing with the creation, setting of values, output and deletion of varbinds.

Creation

snmp_pdu_add_variable will create a new varbind structure, initialised with the name (*objid*, *objidlen*), syntax (*type*) and value (*value* , *len*) provided. This varbind is then added to the end of the varbind list in the given PDU.

snmp_varlist_add_variable is similar, but appends the new varbind to the end of the varbind list provided. When adding the first varbind to an empty list, simply pass the address of the head of the list:

```
netsnmp_variable_list *vl = NULL;
snmp_varlist_add_variable(
        &vl, name1, name1_len,
        ASN_TYPE, &val1, val1_len);
snmp_varlist_add_variable(
        &vl, name2, name2_len,
        ASN_TYPE, &val2, val2_len);
```

In both cases, the routine will return a pointer to the new varbind structure (or NULL if the varbind creation fails).

snmp_add_null_var is a convenience function to add an empty varbind to the PDU. without needing to specify the NULL value explicitly. This is the normal mechanism for constructing a GET (or similar) information retrieval request.
Again, this returns a pointer to the new varbind, or NULL.

snmp_clone_varbind creates a copy of each varbind in the specified list, returning a pointer to the head of the new list (or NULL if the cloning fails).

Setting of values

snmp_set_var_objid sets the name of the varbind structure to the specified OID.
snmp_set_var_typed_value sets the syntax type and value of the varbind structure.
snmp_set_var_value sets the value of the varbind structure, leaving the syntax type unchanged.
snmp_set_var_typed_integer is a convenience function to set the syntax type and value for a 32-bit integer-based varbind.

All four of these return 0 if the assignment is successful, or 1 if it is not.

Output

print_variable will take an object identifier (as returned by **read_objid**, **snmp_parse_oid** or **get_module_node**) and an instance of such a variable, and prints to the standard output the textual form of the object identifier together with the value of the variable.

fprint_variable does the same, but prints to the FILE pointer specified by the initial parameter.

snprint_variable prints the same information into the buffer pointed to by *buf* which is of length *len*. It returns the number of characters printed, or -1 if the buffer was not large enough. In the latter case, *buf* will typically contained a truncated version of the information (but this behaviour is not guaranteed). This function replaces the obsolete function **sprint_variable**.

print_value, **fprint_value**, and **snprint_value** do the same as the equivalent **print_variable** routines, but only displaying the value of the variable, without the corresponding object identifier.

For displaying the OID of a varbind, see **netsnmp_mib_api**(3).

Deletion

snmp_free_var releases all memory used by the given varbind structure.
snmp_free_varbind releases all memory used by each varbind structure in the varbind list provided.

SEE ALSO

netsnmp_pdu_api(3) **netsnmp_mib_api**(3)