NAME

i2t_ASN1_OBJECT, OBJ_length, OBJ_get0_data, OBJ_nid2obj, OBJ_nid2ln, OBJ_nid2sn, OBJ_obj2nid, OBJ_txt2nid, OBJ_ln2nid, OBJ_sn2nid, OBJ_cmp, OBJ_dup, OBJ_txt2obj, OBJ_obj2txt, OBJ_create, OBJ_cleanup - ASN1 object utility functions

SYNOPSIS

```
#include <openssl/objects.h>

ASN1_OBJECT *OBJ_nid2obj(int n);
const char *OBJ_nid2ln(int n);
const char *OBJ_nid2sn(int n);

int OBJ_obj2nid(const ASN1_OBJECT *o);
int OBJ_ln2nid(const char *ln);
int OBJ_sn2nid(const char *sn);

int OBJ_txt2nid(const char *s);

ASN1_OBJECT *OBJ_txt2obj(const char *s, int no_name);
int OBJ_obj2txt(char *buf, int buf_len, const ASN1_OBJECT *a, int no_name);

int i2t_ASN1_OBJECT(char *buf, int buf_len, const ASN1_OBJECT *a);

int OBJ_cmp(const ASN1_OBJECT *a, const ASN1_OBJECT *b);
ASN1_OBJECT *OBJ_dup(const ASN1_OBJECT *o);

int OBJ_create(const char *oid, const char *sn, const char *ln);

size_t OBJ_length(const ASN1_OBJECT *obj);
const unsigned char *OBJ_get0_data(const ASN1_OBJECT *obj);
```

Deprecated:

```
#if OPENSSL_API_COMPAT < 0x10100000L
void OBJ_cleanup(void)
#endif
```

DESCRIPTION

The ASN1 object utility functions process ASN1_OBJECT structures which are a representation of the ASN1 OBJECT IDENTIFIER (OID) type. For convenience, OIDs are usually represented in source code as numeric identifiers, or **NID**s. OpenSSL has an internal table of OIDs that are generated when the library is built, and their corresponding NIDs are available as defined constants. For the functions below, application code should treat all returned values — OIDs, NIDs, or names — as constants.

OBJ_nid2obj(), *OBJ_nid2ln()* and *OBJ_nid2sn()* convert the NID **n** to an ASN1_OBJECT structure, its long name and its short name respectively, or **NULL** if an error occurred.

OBJ_obj2nid(), *OBJ_ln2nid()*, *OBJ_sn2nid()* return the corresponding NID for the object **o**, the long name <ln> or the short name <sn> respectively or NID_undef if an error occurred.

OBJ_txt2nid() returns NID corresponding to text string <s>. **s** can be a long name, a short name or the numerical representation of an object.

OBJ_txt2obj() converts the text string **s** into an ASN1_OBJECT structure. If **no_name** is 0 then long names and short names will be interpreted as well as numerical forms. If **no_name** is 1 only the numerical form is acceptable.

OBJ_obj2txt() converts the **ASN1_OBJECT a** into a textual representation. The representation is written as a null terminated string to **buf** at most **buf_len** bytes are written, truncating the result if necessary. The total amount of space required is returned. If **no_name** is 0 then if the object has a long or short name then that will be used, otherwise the numerical form will be used. If **no_name** is 1 then the numerical form will always be used.

i2t_ASN1_OBJECT() is the same as *OBJ_obj2txt()* with the **no_name** set to zero.

OBJ_cmp() compares **a** to **b**. If the two are identical 0 is returned.

OBJ_dup() returns a copy of **o**.

OBJ_create() adds a new object to the internal table. **oid** is the numerical form of the object, **sn** the short name and **ln** the long name. A new NID is returned for the created object.

OBJ_length() returns the size of the content octets of **obj**.

OBJ_get0_data() returns a pointer to the content octets of **obj**. The returned pointer is an internal pointer which **must not** be freed.

In OpenSSL versions prior to 1.1.0 *OBJ_cleanup()* cleaned up OpenSSLs internal object table and was called before an application exits if any new objects were added using *OBJ_create()*. This function is deprecated in version 1.1.0 and now does nothing if called. No explicit de-initialisation is now required. See *OPENSSL_init_crypto(3)* for further information.

NOTES

Objects in OpenSSL can have a short name, a long name and a numerical identifier (NID) associated with them. A standard set of objects is represented in an internal table. The appropriate values are defined in the header file **objects.h**.

For example the OID for commonName has the following definitions:

```
#define SN_commonName "CN"
#define LN_commonName "commonName"
#define NID_commonName 13
```

New objects can be added by calling *OBJ_create()*.

Table objects have certain advantages over other objects: for example their NIDs can be used in a C language switch statement. They are also static constant structures which are shared: that is there is only a single constant structure for each table object.

Objects which are not in the table have the NID value NID_undef.

Objects do not need to be in the internal tables to be processed, the functions *OBJ_txt2obj()* and *OBJ_obj2txt()* can process the numerical form of an OID.

Some objects are used to represent algorithms which do not have a corresponding ASN.1 OBJECT IDENTIFIER encoding (for example no OID currently exists for a particular algorithm). As a result they **cannot** be encoded or decoded as part of ASN.1 structures. Applications can determine if there is a corresponding OBJECT IDENTIFIER by checking *OBJ_length()* is not zero.

These functions cannot return **const** because an **ASN1_OBJECT** can represent both an internal, constant, OID and a dynamically-created one. The latter cannot be constant because it needs to be freed after use.

EXAMPLES

Create an object for **commonName**:

```
ASN1_OBJECT *o;
o = OBJ_nid2obj(NID_commonName);
```

Check if an object is **commonName**

```
if (OBJ_obj2nid(obj) == NID_commonName)
/* Do something */
```

Create a new NID and initialize an object from it:

```
int new_nid;
ASN1_OBJECT *obj;
```

```
 new_nid = OBJ_create("1.2.3.4", "NewOID", "New Object Identifier");

 obj = OBJ_nid2obj(new_nid);
```

Create a new object directly:

```
 obj = OBJ_txt2obj("1.2.3.4", 1);
```

BUGS

OBJ_obj2txt() is awkward and messy to use: it doesn't follow the convention of other OpenSSL functions where the buffer can be set to **NULL** to determine the amount of data that should be written. Instead **buf** must point to a valid buffer and **buf_len** should be set to a positive value. A buffer length of 80 should be more than enough to handle any OID encountered in practice.

RETURN VALUES

OBJ_nid2obj() returns an **ASN1_OBJECT** structure or **NULL** is an error occurred.

OBJ_nid2ln() and *OBJ_nid2sn()* returns a valid string or **NULL** on error.

OBJ_obj2nid(), *OBJ_ln2nid()*, *OBJ_sn2nid()* and *OBJ_txt2nid()* return a NID or **NID_undef** on error.

SEE ALSO

ERR_get_error(3)

HISTORY

OBJ_cleanup() was deprecated in OpenSSL 1.1.0.

COPYRIGHT

Copyright 2002-2018 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (the ‘‘License’’). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.