Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'ASN1_ex_print_func.3ossl' command

$ man ASN1_ex_print_func.3ossl

ASN1_EXTERN_FUNCS(3ossl) OpenSSL ASN1_EXTERN_FUNCS(3ossl)

NAME

ASN1_EXTERN_FUNCS, ASN1_ex_d2i, ASN1_ex_d2i_ex, ASN1_ex_i2d,
ASN1_ex_new_func, ASN1_ex_new_ex_func, ASN1_ex_free_func,
ASN1_ex_print_func, IMPLEMENT_EXTERN_ASN1 - ASN.1 external function
support

SYNOPSIS

```
#include <openssl/asn1t.h>

typedef int ASN1_ex_d2i(ASN1_VALUE **pval, const unsigned char **in, long len,
                        const ASN1_ITEM *it, int tag, int aclass, char opt,
                        ASN1_TLC *ctx);
typedef int ASN1_ex_d2i_ex(ASN1_VALUE **pval, const unsigned char **in, long len,
                           const ASN1_ITEM *it, int tag, int aclass, char opt,
                           ASN1_TLC *ctx, OSSL_LIB_CTX *libctx,
                           const char *propq);
typedef int ASN1_ex_i2d(const ASN1_VALUE **pval, unsigned char **out,
                        const ASN1_ITEM *it, int tag, int aclass);
typedef int ASN1_ex_new_func(ASN1_VALUE **pval, const ASN1_ITEM *it);
typedef int ASN1_ex_new_ex_func(ASN1_VALUE **pval, const ASN1_ITEM *it,
                                OSSL_LIB_CTX *libctx, const char *propq);
```

```
typedef void ASN1_ex_free_func(ASN1_VALUE **pval, const ASN1_ITEM *it);
typedef int ASN1_ex_print_func(BIO *out, const ASN1_VALUE **pval,
                               int indent, const char *fname,
                               const ASN1_PCTX *pctx);

struct ASN1_EXTERN_FUNCS_st {
   void *app_data;
   ASN1_ex_new_func *asn1_ex_new;
   ASN1_ex_free_func *asn1_ex_free;
   ASN1_ex_free_func *asn1_ex_clear;
   ASN1_ex_d2i *asn1_ex_d2i;
   ASN1_ex_i2d *asn1_ex_i2d;
   ASN1_ex_print_func *asn1_ex_print;
   ASN1_ex_new_ex_func *asn1_ex_new_ex;
   ASN1_ex_d2i_ex *asn1_ex_d2i_ex;
};
typedef struct ASN1_EXTERN_FUNCS_st ASN1_EXTERN_FUNCS;

#define IMPLEMENT_EXTERN_ASN1(sname, tag, fptrs)
```

DESCRIPTION

ASN.1 data structures templates are typically defined in OpenSSL using a series of macros such as ASN1_SEQUENCE(), ASN1_SEQUENCE_END() and so on. Instead templates can also be defined based entirely on external functions. These external functions are called to perform operations such as creating a new ASN1_VALUE or converting an ASN1_VALUE to or from DER encoding.

The macro IMPLEMENT_EXTERN_ASN1() can be used to create such an externally defined structure. The name of the structure should be supplied in the sname parameter. The tag for the structure (e.g. typically V_ASN1_SEQUENCE) should be supplied in the tag parameter. Finally a pointer to an ASN1_EXTERN_FUNCS structure should be supplied

in the fptrs parameter.

The ASN1_EXTERN_FUNCS structure has the following entries.

app_data

A pointer to arbitrary application specific data.

asn1_ex_new

A "new" function responsible for constructing a new ASN1_VALUE object. The newly constructed value should be stored in *pval. The it parameter is a pointer to the ASN1_ITEM template object created via the IMPLEMENT_EXTERN_ASN1() macro.

Returns a positive value on success or 0 on error.

asn1_ex_free

A "free" function responsible for freeing the ASN1_VALUE passed in *pval that was previously allocated via a "new" function. The it parameter is a pointer to the ASN1_ITEM template object created via the IMPLEMENT_EXTERN_ASN1() macro.

asn1_ex_clear

A "clear" function responsible for clearing any data in the ASN1_VALUE passed in *pval and making it suitable for reuse. The it parameter is a pointer to the ASN1_ITEM template object created via the IMPLEMENT_EXTERN_ASN1() macro.

asn1_ex_d2i

A "d2i" function responsible for converting DER data with the tag tag and class class into an ASN1_VALUE. If *pval is non-NULL then the ASN_VALUE it points to should be reused. Otherwise a new ASN1_VALUE should be allocated and stored in *pval. *in points to the DER data to be decoded and len is the length of that data.

After decoding *in should be updated to point at the next byte
after the decoded data. If the ASN1_VALUE is considered optional in
this context then opt will be nonzero. Otherwise it will be zero.
The it parameter is a pointer to the ASN1_ITEM template object
created via the IMPLEMENT_EXTERN_ASN1() macro. A pointer to the
current ASN1_TLC context (which may be required for other ASN1
function calls) is passed in the ctx parameter.

The asn1_ex_d2i entry may be NULL if asn1_ex_d2i_ex has been
specified instead.

Returns <= 0 on error or a positive value on success.

asn1_ex_i2d

An "i2d" function responsible for converting an ASN1_VALUE into DER
encoding. On entry *pval will contain the ASN1_VALUE to be
encoded. If default tagging is to be used then tag will be -1 on
entry. Otherwise if implicit tagging should be used then tag and
aclass will be the tag and associated class.

If out is not NULL then this function should write the DER encoded
data to the buffer in *out, and then increment *out to point to
immediately after the data just written.

If out is NULL then no data should be written but the length
calculated and returned as if it were.

The asn1_ex_i2d entry may be NULL if asn1_ex_i2d_ex has been
specified instead.

The return value should be negative if a fatal error occurred, or 0
if a non-fatal error occurred. Otherwise it should return the
length of the encoded data.

asn1_ex_print

A "print" function. out is the BIO to print the output to. *pval is the ASN1_VALUE to be printed. indent is the number of spaces of indenting to be printed before any data is printed. fname is currently unused and is always "". pctx is a pointer to the ASN1_PCTX for the print operation.

Returns 0 on error or a positive value on success. If the return value is 2 then an additional newline will be printed after the data printed by this function.

asn1_ex_new_ex

This is the same as asn1_ex_new except that it is additionally passed the OSSL_LIB_CTX to be used in libctx and any property query string to be used for algorithm fetching in the propq parameter. See "ALGORITHM FETCHING" in crypto(7) for further details. If asn1_ex_new_ex is non NULL, then it will always be called in preference to asn1_ex_new.

asn1_ex_d2i_ex

This is the same as asn1_ex_d2i except that it is additionally passed the OSSL_LIB_CTX to be used in libctx and any property query string to be used for algorithm fetching in the propq parameter. See "ALGORITHM FETCHING" in crypto(7) for further details. If asn1_ex_d2i_ex is non NULL, then it will always be called in preference to asn1_ex_d2i.

RETURN VALUES

Return values for the various callbacks are as described above.

SEE ALSO

ASN1_item_new_ex(3)

HISTORY

The asn1_ex_new_ex and asn1_ex_d2i_ex callbacks were added in OpenSSL 3.0.

COPYRIGHT