Rocky Enterprise Linux 9.2 Manual Pages on command 'OSSL_DECODER_get0_name.3ossl'

$ man OSSL_DECODER_get0_name.3ossl

OSSL_DECODER(3ossl) OpenSSL OSSL_DECODER(3ossl)

NAME

OSSL_DECODER, OSSL_DECODER_fetch, OSSL_DECODER_up_ref,

OSSL_DECODER_free, OSSL_DECODER_get0_provider,

OSSL_DECODER_get0_properties, OSSL_DECODER_is_a,

OSSL_DECODER_get0_name, OSSL_DECODER_get0_description,

OSSL_DECODER_do_all_provided, OSSL_DECODER_names_do_all,

OSSL_DECODER_gettable_params, OSSL_DECODER_get_params - Decoder method

routines

SYNOPSIS

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

typedef struct ossl_decoder_st OSSL_DECODER;

OSSL_DECODER *OSSL_DECODER_fetch(OSSL_LIB_CTX *ctx, const char *name,
                                 const char *properties);

int OSSL_DECODER_up_ref(OSSL_DECODER *decoder);

void OSSL_DECODER_free(OSSL_DECODER *decoder);

const OSSL_PROVIDER *OSSL_DECODER_get0_provider(const OSSL_DECODER *decoder);

const char *OSSL_DECODER_get0_properties(const OSSL_DECODER *decoder);
```

```
int OSSL_DECODER_is_a(const OSSL_DECODER *decoder, const char *name);
const char *OSSL_DECODER_get0_name(const OSSL_DECODER *decoder);
const char *OSSL_DECODER_get0_description(const OSSL_DECODER *decoder);
void OSSL_DECODER_do_all_provided(OSSL_LIB_CTX *libctx,
                                  void (*fn)(OSSL_DECODER *decoder, void *arg),
                                  void *arg);
int OSSL_DECODER_names_do_all(const OSSL_DECODER *decoder,
                              void (*fn)(const char *name, void *data),
                              void *data);
const OSSL_PARAM *OSSL_DECODER_gettable_params(OSSL_DECODER *decoder);
int OSSL_DECODER_get_params(OSSL_DECODER_CTX *ctx, const OSSL_PARAM params[]);
```

DESCRIPTION

OSSL_DECODER is a method for decoders, which know how to decode encoded data into an object of some type that the rest of OpenSSL knows how to handle.

OSSL_DECODER_fetch() looks for an algorithm within the provider that has been loaded into the OSSL_LIB_CTX given by ctx, having the name given by name and the properties given by properties. The name determines what type of object the fetched decoder method is expected to be able to decode, and the properties are used to determine the expected output type. For known properties and the values they may have, please have a look in "Names and properties" in provider-encoder(7).

OSSL_DECODER_up_ref() increments the reference count for the given decoder.

OSSL_DECODER_free() decrements the reference count for the given decoder, and when the count reaches zero, frees it.

OSSL_DECODER_get0_provider() returns the provider of the given decoder.

OSSL_DECODER_get0_properties() returns the property definition associated with the given decoder.

OSSL_DECODER_is_a() checks if decoder is an implementation of an algorithm that's identifiable with name.

OSSL_DECODER_get0_name() returns the name used to fetch the given

decoder.

OSSL_DECODER_get0_description() returns a description of the decoder, meant for display and human consumption. The description is at the discretion of the decoder implementation.

OSSL_DECODER_names_do_all() traverses all names for the given decoder, and calls fn with each name and data as arguments.

OSSL_DECODER_do_all_provided() traverses all decoder implementations by all activated providers in the library context libctx, and for each of the implementations, calls fn with the implementation method and arg as arguments.

OSSL_DECODER_gettable_params() returns an OSSL_PARAM(3) array of parameter descriptors.

OSSL_DECODER_get_params() attempts to get parameters specified with an OSSL_PARAM(3) array params. Parameters that the implementation doesn't recognise should be ignored.

RETURN VALUES

OSSL_DECODER_fetch() returns a pointer to an OSSL_DECODER object, or NULL on error.

OSSL_DECODER_up_ref() returns 1 on success, or 0 on error.

OSSL_DECODER_free() doesn't return any value.

OSSL_DECODER_get0_provider() returns a pointer to a provider object, or NULL on error.

OSSL_DECODER_get0_properties() returns a pointer to a property definition string, or NULL on error.

OSSL_DECODER_is_a() returns 1 if decoder was identifiable, otherwise 0.

OSSL_DECODER_get0_name() returns the algorithm name from the provided implementation for the given decoder. Note that the decoder may have multiple synonyms associated with it. In this case the first name from the algorithm definition is returned. Ownership of the returned string is retained by the decoder object and should not be freed by the caller.

OSSL_DECODER_get0_description() returns a pointer to a decription, or NULL if there isn't one.

OSSL_DECODER_names_do_all() returns 1 if the callback was called for all names. A return value of 0 means that the callback was not called for any names.

NOTES

OSSL_DECODER_fetch() may be called implicitly by other fetching functions, using the same library context and properties. Any other API that uses keys will typically do this.

EXAMPLES

To list all decoders in a provider to a bio_out:

```
static void collect_decoders(OSSL_DECODER *decoder, void *stack)
{
    STACK_OF(OSSL_DECODER) *decoder_stack = stack;
    sk_OSSL_DECODER_push(decoder_stack, decoder);
    OSSL_DECODER_up_ref(decoder);
}
void print_name(const char *name, void *vdata)
{
    BIO *bio = vdata;
    BIO_printf(bio, "%s ", name);
}
STACK_OF(OSSL_DECODER) *decoders;
int i;
decoders = sk_OSSL_DECODER_new_null();
BIO_printf(bio_out, "DECODERs provided by %s:\n", provider);
OSSL_DECODER_do_all_provided(NULL, collect_decoders,
                             decoders);
for (i = 0; i < sk_OSSL_DECODER_num(decoders); i++) {
    OSSL_DECODER *decoder = sk_OSSL_DECODER_value(decoders, i);
    if (strcmp(OSSL_PROVIDER_get0_name(OSSL_DECODER_get0_provider(decoder)),
               provider) != 0)
        continue;
    if (OSSL_DECODER_names_do_all(decoder, print_name, bio_out))
        BIO_printf(bio_out, "\n");
```

```
    }
    sk_OSSL_DECODER_pop_free(decoders, OSSL_DECODER_free);
```

SEE ALSO

provider(7), OSSL_DECODER_CTX(3), OSSL_DECODER_from_bio(3), OSSL_DECODER_CTX_new_for_pkey(3), OSSL_LIB_CTX(3)

HISTORY

The functions described here were added in OpenSSL 3.0.

COPYRIGHT