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 'UI_destroy_method.3ossl' command

$ man UI_destroy_method.3ossl

UI_CREATE_METHOD(3ossl) OpenSSL UI_CREATE_METHOD(3ossl)

NAME

UI_METHOD, UI_create_method, UI_destroy_method, UI_method_set_opener, UI_method_set_writer, UI_method_set_flusher, UI_method_set_reader, UI_method_set_closer, UI_method_set_data_duplicator, UI_method_set_prompt_constructor, UI_method_set_ex_data, UI_method_get_opener, UI_method_get_writer, UI_method_get_flusher, UI_method_get_reader, UI_method_get_closer, UI_method_get_data_duplicator, UI_method_get_data_destructor, UI_method_get_prompt_constructor, UI_method_get_ex_data - user interface method creation and destruction

SYNOPSIS

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

typedef struct ui_method_st UI_METHOD;

UI_METHOD *UI_create_method(const char *name);
void UI_destroy_method(UI_METHOD *ui_method);
int UI_method_set_opener(UI_METHOD *method, int (*opener) (UI *ui));
int UI_method_set_writer(UI_METHOD *method,
                         int (*writer) (UI *ui, UI_STRING *uis));
```

```
int UI_method_set_flusher(UI_METHOD *method, int (*flusher) (UI *ui));
int UI_method_set_reader(UI_METHOD *method,
                         int (*reader) (UI *ui, UI_STRING *uis));
int UI_method_set_closer(UI_METHOD *method, int (*closer) (UI *ui));
int UI_method_set_data_duplicator(UI_METHOD *method,
                                  void *(*duplicator) (UI *ui, void *ui_data),
                                  void (*destructor)(UI *ui, void *ui_data));
int UI_method_set_prompt_constructor(UI_METHOD *method,
                                     char *(*prompt_constructor) (UI *ui,
                                                                  const char
                                                                  *object_desc,
                                                                  const char
                                                                  *object_name));
int UI_method_set_ex_data(UI_METHOD *method, int idx, void *data);
int (*UI_method_get_opener(const UI_METHOD *method)) (UI *);
int (*UI_method_get_writer(const UI_METHOD *method)) (UI *, UI_STRING *);
int (*UI_method_get_flusher(const UI_METHOD *method)) (UI *);
int (*UI_method_get_reader(const UI_METHOD *method)) (UI *, UI_STRING *);
int (*UI_method_get_closer(const UI_METHOD *method)) (UI *);
char *(*UI_method_get_prompt_constructor(const UI_METHOD *method))
    (UI *, const char *, const char *);
void *(*UI_method_get_data_duplicator(const UI_METHOD *method)) (UI *, void *);
void (*UI_method_get_data_destructor(const UI_METHOD *method)) (UI *, void *);
const void *UI_method_get_ex_data(const UI_METHOD *method, int idx);
```

DESCRIPTION

A method contains a few functions that implement the low-level of the User Interface. These functions are:

an opener

This function takes a reference to a UI and starts a session, for example by opening a channel to a tty, or by creating a dialog box.

a writer

This function takes a reference to a UI and a UI String, and writes the string where appropriate, maybe to the tty, maybe added as a field label in a dialog box. Note that this gets fed all strings associated with a UI, one after the other, so care must be taken which ones it actually uses.

a flusher

This function takes a reference to a UI, and flushes everything that has been output so far. For example, if the method builds up a dialog box, this can be used to actually display it and accepting input ended with a pressed button.

a reader

This function takes a reference to a UI and a UI string and reads off the given prompt, maybe from the tty, maybe from a field in a dialog box. Note that this gets fed all strings associated with a UI, one after the other, so care must be taken which ones it actually uses.

a closer

This function takes a reference to a UI, and closes the session, maybe by closing the channel to the tty, maybe by destroying a dialog box.

All of these functions are expected to return 0 on error, 1 on success, or -1 on out-off-band events, for example if some prompting has been cancelled (by pressing Ctrl-C, for example). Only the flusher or the reader are expected to return -1. If returned by another of the functions, it's treated as if 0 was returned.

Regarding the writer and the reader, don't assume the former should only write and don't assume the latter should only read. This depends

on the needs of the method.

For example, a typical tty reader wouldn't write the prompts in the write, but would rather do so in the reader, because of the sequential nature of prompting on a tty. This is how the UI_OpenSSL() method does it.

In contrast, a method that builds up a dialog box would add all prompt text in the writer, have all input read in the flusher and store the results in some temporary buffer, and finally have the reader just fetch those results.

The central function that uses these method functions is UI_process(), and it does it in five steps:

1. Open the session using the opener function if that one's defined. If an error occurs, jump to 5.

2. For every UI String associated with the UI, call the writer function if that one's defined. If an error occurs, jump to 5.

3. Flush everything using the flusher function if that one's defined. If an error occurs, jump to 5.

4. For every UI String associated with the UI, call the reader function if that one's defined. If an error occurs, jump to 5.

5. Close the session using the closer function if that one's defined.

UI_create_method() creates a new UI method with a given name.

UI_destroy_method() destroys the given UI method ui_method.

UI_method_set_opener(), UI_method_set_writer(),
UI_method_set_flusher(), UI_method_set_reader() and
UI_method_set_closer() set the five main method function to the given
function pointer.

UI_method_set_data_duplicator() sets the user data duplicator and
destructor. See UI_dup_user_data(3).

UI_method_set_prompt_constructor() sets the prompt constructor. See
UI_construct_prompt(3).

UI_method_set_ex_data() sets application specific data with a given
EX_DATA index. See CRYPTO_get_ex_new_index(3) for general information
on how to get that index.

UI_method_get_opener(), UI_method_get_writer(),
UI_method_get_flusher(), UI_method_get_reader(),
UI_method_get_closer(), UI_method_get_data_duplicator(),
UI_method_get_data_destructor() and UI_method_get_prompt_constructor()
return the different method functions.

UI_method_get_ex_data() returns the application data previously stored
with UI_method_set_ex_data().

RETURN VALUES

UI_create_method() returns a UI_METHOD pointer on success, NULL on
error.

UI_method_set_opener(), UI_method_set_writer(),
UI_method_set_flusher(), UI_method_set_reader(),
UI_method_set_closer(), UI_method_set_data_duplicator() and
UI_method_set_prompt_constructor() return 0 on success, -1 if the given
method is NULL.

UI_method_set_ex_data() returns 1 on success and 0 on error (because CRYPTO_set_ex_data() does so).

UI_method_get_opener(), UI_method_get_writer(), UI_method_get_flusher(), UI_method_get_reader(), UI_method_get_closer(), UI_method_get_data_duplicator(), UI_method_get_data_destructor() and UI_method_get_prompt_constructor() return the requested function pointer if it's set in the method, otherwise NULL.

UI_method_get_ex_data() returns a pointer to the application specific data associated with the method.

SEE ALSO

UI(3), CRYPTO_get_ex_data(3), UI_STRING(3)

HISTORY

The UI_method_set_data_duplicator(), UI_method_get_data_duplicator() and UI_method_get_data_destructor() functions were added in OpenSSL 1.1.1.

COPYRIGHT

Copyright 2001-2020 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (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>.