TPGETCTXT(3)

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

SYNOPSIS

#include <atmi.h>

int tpgetctxt(TPCONTEXT_T *_context_, long _flags_);

Link with _-latmisrv|-latmisrvnomain|-latmisrvinteg -latmi -lubf -lnstd -lpthread -lrt -lm_

Chapter 2

DESCRIPTION

Function is used to get thread local storage data (TLS) associated with current thread. The current thread after calling this function becomes disassociated from any TLS and is running in NULL context. How every, any ATMI call will make to initialize new client TLS. Due to nature of the function, that it is suspending full ATMI TLS setup, it should not be used from ATMI server's main thread (service dispatcher), how ever it is safe to use it in ATMI server's client threads. The *context* variable points to space where to save the current thread's TLS data. *flags* shall be set to 0. Any global XA transaction currently associated with thread becomes suspended, and it's suspend data is saved in returned context data.

If the context data is not restored by **tpsetctxt(3)** then you have to make the data free by **tpfreectxt(3)** call, otherwise memory will be leaked.

To pass the request from server main thread to worker thread, use **tpsrvgetctxdata(3)** and **tpsrvsetctxdata(3)**.

This function uses underlaying thread local storage infrastructure which is provided separately for each of the major Enduro/X libraries - libnstd (Standard library), libufb (UBF buffer library) and libatmi (ATMI library). If operations at library levels are required, then following functions can be used:

1. ndrx_nstd_tls_new(), ndrx_ubf_tls_new(), ndrx_atmi_tls_new() - allocate TLS data for library.
2. ndrx_nstd_tls_get(), ndrx_ubf_tls_get(), ndrx_atmi_tls_get() - get the TLS data for library (currently associated with thread).
3. ndrx_nstd_tls_set(), ndrx_ubf_tls_set(), ndrx_atmi_tls_set() - set the thread local data from saved pointer.
4. ndrx_nstd_tls_free(), ndrx_ubf_tls_free(), ndrx_atmi_tls_free() - free the thread local data.

Chapter 3

RETURN VALUE

On success, **tpgetctxt()** return **TPMULTICONTEXTS** if context/TLS data was set, **TPNULLCONTEXT** if running in NULL context (no TLS initialized - there was no ATMI calls); on error, -1 is returned, with **tperrno** set to indicate the error.

Chapter 4

ERRORS

Note that **tpstrerror()** returns generic error message plus custom message with debug info from last function call. Error data (tperrno) is valid only if return was -1. In case if process was running in **TPNULLCONTEXT**, new client context will be made to store the error code.

TPEINVAL *context* parameter was NULL or flags was not 0.

Chapter 5

EXAMPLE

See **atmitest/test016_contextsw/atmiclt16.c** for sample code.

Chapter 6

BUGS

Report bugs to support@mavimax.com

Chapter 7

SEE ALSO

tpsetctxt(3) tpfreectxt(3) tpsrvsetctxdata(3) tpsrvgetctxdata(3) tpcontinue(3) tpinit(3)

Chapter 8

COPYING