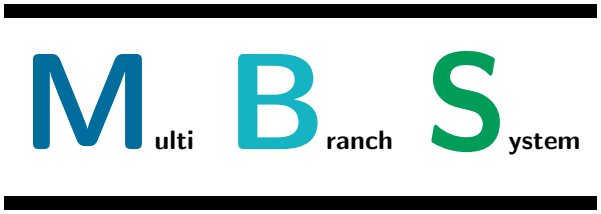
MuBSy
Id.: gm˙eventapi
Version: 3.4
Date: March 20, 2004
Revised: March 20, 2004



MBS Event API

H.G. Essel

March 20, 2004

GSI, Gesellschaft für Schwerionenforschung mbH
Planckstraße 1, D-64291 Darmstadt
Germany
Tel. (0 6159) 71–0

Chapter 1

MBS Event API

f_evt__example

CALLING Examples for calling event API

PURPOSE See f_evt_examples.c

Implementation

User Example	In m_lea_user.c
Channel structure	defined in f_evt.h
File name	f_evt.c
Version	1.01
Author	H.Essel
Created	16-Feb-2000
Updates	Date Purpose

f_evt_get_subevent

CALLING l_sts = f_evt_get_subevent(ve10_1 *,subevent,**head,**data,*lwords)

PURPOSE get subevent pointer

ARGUMENTS

ve10_1 (s_ve10_1 *) event header pointer

subevent subevent number (1,2,3...) If = 0, f_evt_get_subevent returns the number of subevents. In this case the following arguments might be NULL.

head Address of s_ves10_1 subevent header pointer

data Address of INTS4 event data pointer

lwords Address of INTS4 to get number of data longwords

Return type int

GETEVT__SUCCESS Found subevent.

GETEVT__NOMORE No more subevents.

Declaration INTS4 f_evt_get_subevent(s_ve10_1 *, INTS4, INTS4 **, INTS4 **, INTS4 *);

f_evt_type

CALLING l_status = f_evt_type(bufhe,evhe,sid,long,hex,data)

PURPOSE print event

ARGUMENTS

bufhe	(s_bufhe *) buffer header pointer (=NULL no output)
evhe	(s_evhe *) event header pointer (=NULL no output)
sid	subevent ID (-1 is all)
long	output longwords
hex	output hex longwords
data	output data

Return type int

Declaration INTS4 f_evt_type(s_bufhe *,s_evhe *, INTS4, INTS4, INTS4, INTS4);

f_evt_rev_port

CALLING f_evt_rev_port(long l_port)

PURPOSE f_evt_rev_port sets port number for event server

ARGUMENTS

l_port Port number:

Return type int.

Status codes

GETEVT__SUCCESS success.

Declaration INTS4 f_evt_rev_port(INTS4);

f_evt_get_open

CALLING f_evt_get_open(long l_mode, char &c_server[], s_evt_channel &s_chan, char **ps_info, long l_sample,l_para)

PURPOSE f_evt_get_open opens an event stream from specified channel.

ARGUMENTS

l_mode Type of server:

GETEVT__FILE Input from file

GETEVT__STREAM Input from MBS stream server

GETEVT__TRANS Input from MBS transport

GETEVT__EVENT Input from MBS event server

GETEVT__REVSERV Input from remote event server

c_server Node of server or file name.

s_chan structure s_evt_channel, must be allocated.

ps_info address of pointer. If it is not NULL, then try to return file header or other information about server. If it is NULL, then returns nothing.

l_sample used by event server to send only every 'l_sample' event.

l_para currently not used

Return type int.

Status codes

GETEVT__SUCCESS success.

GETEVT__NOFILE file does not exist.

GETEVT__RDERR read server error.

GETEVT__NOSERVER can not connect server.

Declaration	INTS4 f_evt_get_open(INTS4, CHARS *, s_evt_channel *, CHARS **, INTS4, INTS4);
FUNCTION	Opens the input channel and save context in s_chan.
NOTE	Up to four input channels can be opened.

f_evt_get_event

CALLING f_evt_get_event(s_evt_channel &s_chan, long **ppl_buffer, long **ppl_goobuf)

PURPOSE f_evt_get_event returnes address of event

ARGUMENTS

s_chan Input channel from open.

ppl_buffer Address of pointer. Returns address of event.

ppl_goobuf Address of pointer. Returns address of buffer.

Return type int.

Status codes

GETEVT__SUCCESS success.

GETEVT__FRAGMENT Event fragment found.

GETEVT__NOMORE No more events.

GETEVT__RDERR read server or file error

GETEVT__TIMEOUT when enabled by f_evt_timeout

Declaration INTS4 f_evt_get_event(s_evt_channel *, INTS4 **, INTS4 **);

FUNCTION Get next event and returnes pointer. The pointer may point to the event in the buffer or internal event buffer (spanned events). The content of the pointer may be destroyed by next call.

f_evt_get_close

CALLING f_evt_get_close(s_evt_channel &s_chan)

PURPOSE f_evt_get_close closes event stream of specified channel.

ARGUMENTS

s_chan Input channel from open.

Return type int.

Status codes

GETEVT__SUCCESS success.

GETEVT__CLOSE_ERR close server or file error

Declaration INTS4 f_evt_get_close(s_evt_channel *);

FUNCTION Closes the specified input channel.

f_evt_put_open

CALLING f_evt_put_open(char *c_file[], long l_size, long l_stream, long l_type, long l_subtype, s_evt_channel *ps_chan, char *ps_filhe)

PURPOSE f_evt_put_open opens an event output stream.

ARGUMENTS

c_file Name of file.

l_size Size of output buffers in bytes.

l_stream Number of buffers with spanning events.

l_type Buffer type number

l_subtype Buffer subtype number

ps_chan Address of channel structure which will be returned.

ps_filhe Address of user specified file header

Return type int.

Status codes

PUTEVT__SUCCESS success.

PUTEVT__FILE_EXIST file already exists.

PUTEVT__FAILURE failure.

Declaration INTS4 f_evt_put_open(CHARS *,INTS4,INTS4,INTS4,INTS4,s_evt_channel *,CHARS *);

FUNCTION Opens the output channel and save context in s_evt_channel structure.

NOTE Up to four output channels can be opened. User Example : In m_lea_user.c

f_evt_put_event

CALLING f_evt_put_event(s_evt_channel *ps_chan, long &la_evt_buf[])

PURPOSE f_evt_put_event outputs event

ARGUMENTS

ps_chan Address of channel structure as returned from f_evt_put_open.

la_evt_buf event data array. Standard GSI event structure.

Return type int.

Status codes

PUTEVT__SUCCESS success.

PUTEVT__WRERR read server or file error

Declaration INTS4 f_evt_put_event(s_evt_channel *, INTS4 *);

FUNCTION Copies current event into output buffer. Writes buffer to file, when full.

f_evt_put_buffer

CALLING f_evt_put_buffer(s_evt_channel *ps_chan, s_bufhe *ps_bufhe)

PURPOSE f_evt_put_buffer outputs buffer

ARGUMENTS

ps_chan Address of channel structure as returned from f_evt_put_open.

ps_bufhe Buffer.

Return type int.

Status codes

PUTEVT__SUCCESS success.

PUTEVT__WRERR read server or file error

Declaration INTS4 f_evt_put_buffer(s_evt_channel *, s_bufhe *);

FUNCTION Writes buffer to file.

f_evt_put_close

CALLING f_evt_put_close(s_evt_channel *ps_chan)

PURPOSE f_evt_put_close closes specified channel.

ARGUMENTS

channel Channel number.

Return type int.

Status codes

PUTEVT__SUCCESS success.

PUTEVT__FAILURE failure.

PUTEVT__CLOSE_ERR close server or file error

Declaration INTS4 f_evt_put_close(s_evt_channel *);

FUNCTION Closes the specified output channel after writing last buffer.

f_evt_error

CALLING f_evt_error(long l_error , char &c_string[], long l_out)

PURPOSE f_evt_error displays error messages.

ARGUMENTS

l_error The error id as returned from other calls

c_string The string into f_evt_error() copies the message....

l_out specifies the output device for the error message.

out = 1 error message is copied to string.

out = 0 error message is printed on terminal.

Return type int (longword).

Status codes

GETEVT__SUCCESS success.

GETEVT__FAILURE failure

Declaration INTS4 f_evt_error(INTS4 , CHARS * , INTS4);

FUNCTION f_evt_error displays the error message for the error id (l_error). If out = 1 the error message is copied into string, else f_evt_error prints the message on terminal.

f_evt_get_buffer

CALLING f_evt_get_buffer(s_evt_channel &s_chan, INTS4 *pl_buffer)

PURPOSE f_evt_get_buffer read one buffer from server into user buffer.

ARGUMENTS

s_chan structure s_evt_channel.

pl_buffer Address of user buffer

Return type int.

Status codes

GETEVT__SUCCESS success.

GETEVT__FAILURE failure

GETEVT__RDERR read server or file error

GETEVT__NOMORE No more events.

GETEVT__TIMEOUT when enabled by f_evt_timeout

Declaration INTS4 f_evt_get_buffer(s_evt_channel *, INTS4 *);

f_evt_skip_buffer

CALLING f_evt_skip_buffer(s_evt_channel &s_chan, INTS4 l_buffer)

PURPOSE Skip buffers in file.

ARGUMENTS

s_chan structure s_evt_channel.

l_buffer buffers to skip

Return type int.

Status codes

GETEVT__SUCCESS success.

GETEVT__FAILURE failure

GETEVT__RDERR read server or file error

GETEVT__NOMORE No more events.

GETEVT__TIMEOUT when enabled by f_evt_timeout

Declaration INTS4 f_evt_skip_buffer(s_evt_channel *, INTS4);

f_evt_timeout

CALLING	f_evt_timeout(s_evt_channel *ps_chan, seconds)
PURPOSE	Set a timeout for TCP read operations
ARGUMENTS	
ps_chan	Address of channel structure.
seconds	-1: wait (default) >0: if after n seconds no data arrived, read functions return GETEVT__TIMEOUT.
Return type	INTS4
Declaration	INTS4 f_evt_timeout(s_evt_channel *, INTS4);
FUNCTION	Set a timeout for TCP read operations. The calls of f_evt_get_event, f_evt_get_buffer will return GETEVT__TIMEOUT when seconds have been set to positive value.

f_evt_cre_tagfile

CALLING f_evt_cre_tagfile(lmdfile,tagfile,filter)

PURPOSE Create a tag file from lmd file

ARGUMENTS

lmdfile LMD file name

tagfile tag file name

filter optional function for filter

Return type INTS4 .

Status codes

GETEVT__SUCCESS success.

GETEVT__NOFILE file does not exist.

GETEVT__TAGWRERR tag file write error.

GETEVT__RDERR lmd read error.

Declaration INTS4 f_evt_cre_tagfile(CHARS *,CHARS *, INTS4 (*)());

FUNCTION Create a tag file from lmd file

filter The filter function is called at the beginning with a NULL as argument, then for each event with the event pointer. Returning 0 skips the event, 1 takes the event into the tag file. Different tag files can be created from one lmd file.

f_evt_get_tagopen

CALLING f_evt_get_tagopen(channel,tagfile,lmdfile,header,print)

PURPOSE Open tag and lmd file

ARGUMENTS

channel s_evt_channel* , must be allocated.

tagfile Name of tag file

filename LMD file name

header address of CHARS pointer. If it is not NULL, then try to return file header or other information about server. If it is NULL, then returns nothing.

print Print flag: 1=verbose */

Return type INTS4 .

Status codes

GETEVT__SUCCESS success.

GETEVT__NOFILE file does not exist.

GETEVT__TAGRDERR tag file read error.

GETEVT__RDERR read server error.

Declaration INTS4 f_evt_get_tagopen(s_evt_channel *,CHARS *,CHARS *,CHARS **,INTS4);

FUNCTION Open tag file and lmd file. If no tag file is found, a standard f_evt_get_open is called. In this case following tag functions call standard f_evt_get functions.

f_evt_get_tagnext

CALLING f_evt_get_tagnext(channel,skip,**event)

PURPOSE Get tagged event from lmd file

ARGUMENTS

channel s_evt_channel* , must be allocated.

event address of pointer. If it is not NULL, then return event pointer.

Return type INTS4 .

Status codes

GETEVT__SUCCESS success.

GETEVT__NOFILE file does not exist.

GETEVT__TAGRDERR tag file read error.

GETEVT__RDERR lmd read error.

GETEVT__FRAGMENT Event fragment found.

GETEVT__NOMORE No more events.

Declaration INTS4 f_evt_get_tagnext(s_evt_channel *,INTS4,INTS4 **);

FUNCTION Get next event at current position, either in tag file, or in lmd file. Optional <skip> events are skipped.

f_evt_get_tagevent

CALLING f_evt_get_tagevent(*channel, value, mode, **event)

PURPOSE Get tagged event from lmd file

ARGUMENTS

channel s_evt_channel* , must be allocated.

value event number or index

mode 0: value is number, 1: value is index

event address of pointer. If it is not NULL, then return event pointer.

Return type INTS4 .

Status codes

GETEVT__SUCCESS success.

GETEVT__TAGRDERR tag file read error.

GETEVT__RDERR lmd read error.

GETEVT__FRAGMENT Event fragment found.

GETEVT__NOMORE No more events.

GETEVT__NOTAG Specified event not in tag file.

Declaration INTS4 f_evt_get_tagevent(s_evt_channel *,INTS4,INTS4,INTS4 **);

FUNCTION Get tag event. If no tag file is there, skip <value> events, or look for event number <value>

f_evt_get_tagclose

CALLING	f_evt_get_tagclose(s_evt_channel)
PURPOSE	Close tag and lmd file, cleanup s_evt_channel
ARGUMENTS	
channel	s_evt_channel* , must be allocated.
Return type	INTS4
Declaration	INTS4 f_evt_get_tagclose(s_evt_channel *);
FUNCTION	Create a tag file from lmd file

Contents