

cetlib - Feature #25448

Create proper cmake targets for cetlib and cetlib_except

01/28/2021 04:14 PM - Pengfei Ding

Status:	Closed	Start date:	01/28/2021
Priority:	Normal	Due date:	
Assignee:		% Done:	0%
Category:		Estimated time:	0.00 hour
Target version:		Spent time:	0.00 hour
Description			
DUNE DAQ currently run into the following situation that when we package our pre-built packages into UPS product, the absolute path to libcetlib.so and libcetlib_except.so was embedded in the generated cmake Targets.cmake file.			
This limited our ability to relocate the UPS packages. As a temporary solution, we replace the absolute path to those share libraries by using UPS environment variables (e.g. \$ENV{CETLIB_LIB}/libcetlib.so).			
However, a better solution might be if cetlib and cetlib_except can generate proper libraries targets to be used in the first place.			

History

#1 - 02/01/2021 10:27 AM - Kyle Knoepfel

- Status changed from New to Feedback

cetbuildtools, which was used to generate all officially released cetlib(_except) packages does not generate Targets.cmake files. Are you referring to something else? We will need more information.

#2 - 02/01/2021 12:50 PM - Pengfei Ding

Kyle Knoepfel wrote:

cetbuildtools, which was used to generate all officially released cetlib(_except) packages does not generate Targets.cmake files. Are you referring to something else? We will need more information.

I was referring to `cetlib` and `cetlib_except`.

#3 - 02/01/2021 12:58 PM - Kyle Knoepfel

As I said, neither of those UPS packages include Targets.cmake files--how did you obtain the cetlib and cetlib_except packages such that they included targets.cmake files? We only ship config.cmake files.

#4 - 02/01/2021 01:08 PM - Pengfei Ding

Pengfei Ding wrote:

DUNE DAQ currently run into the following situation that when we package our pre-built packages into UPS product, the absolute path to libcetlib.so and libcetlib_except.so was embedded in the generated cmake Targets.cmake file.

This limited our ability to relocate the UPS packages. As a temporary solution, we replace the absolute path to those share libraries by using UPS environment variables (e.g. \$ENV{CETLIB_LIB}/libcetlib.so).

However, a better solution might be if cetlib and cetlib_except can generate proper libraries targets to be used in the first place.

Sorry for the confusing. I meant the generated cmake Targets file for our own package.

Because of `config.cmake` used in cetlib and cetlib_excpet, the absolute path to the libs got embedded into our own packages' generated targets.cmake.

We had to replace the absolute path with UPS environment variables to make the package relocatable.

#5 - 02/01/2021 01:19 PM - Kyle Knoepfel

- Status changed from Feedback to Closed

In that case, I am closing this issue. To my knowledge, the CMake config files we generate for cetlib do not include any hard-coded paths (see an

example below). If you are wanting the installed cetlib package to include a cetlibTargets.cmake file, which includes CMake namespace-qualified targets, then you will get that with art suite 3.07. That version of art will be built with cetmodules, which will support (e.g.) the cetlib::cetlib CMake target.

CMake config file for cetlib 3.11.01

```
set( cetlib_VERSION 3.11.01 )
set( cetlib_UPS_VERSION v3_11_01 )

##### Expanded from @PACKAGE_INIT@ by configure_package_config_file() #####
##### Any changes to this file will be overwritten by the next CMake run #####
##### The input file was product-config.cmake.in #####

get_filename_component(PACKAGE_PREFIX_DIR "${CMAKE_CURRENT_LIST_DIR}/../../../../../../" ABSOLUTE)

macro(set_and_check _var _file)
  set(${_var} "${_file}")
  if(NOT EXISTS "${_file}")
    message(FATAL_ERROR "File or directory ${_file} referenced by variable ${_var} does not exist !")
  endif()
endmacro()

macro(check_required_components _NAME)
  foreach(comp ${${_NAME}_FIND_COMPONENTS})
    if(NOT ${_NAME}_${comp}_FOUND)
      if(${_NAME}_FIND_REQUIRED_${comp})
        set(${_NAME}_FOUND FALSE)
      endif()
    endif()
  endforeach()
endmacro()

#####

if (IS_DIRECTORY "${PACKAGE_PREFIX_DIR}/cetlib/v3_11_01/Modules")
  list(APPEND CMAKE_MODULE_PATH "${PACKAGE_PREFIX_DIR}/cetlib/v3_11_01/Modules")
endif()

## find_library directives
find_ups_boost( v1_50_0 )
find_ups_product( sqlite )
find_ups_product( cetlib_except v1_01_00 )
find_ups_product( hep_concurrency )
find_ups_product( cppunit )
find_ups_product( tbb )
  cet_find_library( CPPUNIT NAMES cppunit PATHS ENV CPPUNIT_LIB NO_DEFAULT_PATH )
  cet_find_library( SQLITE3 NAMES sqlite3_ups PATHS ENV SQLITE_LIB NO_DEFAULT_PATH )
set(CETLIB "$ENV{CETLIB_LIB}/libcetlib.so")
include_directories ( "$ENV{CETLIB_INC}" )
set( CETLIB_CETSKEL_BASICPLUGINCOMMON_PM $ENV{CETLIB_DIR}/perllib/CetSkel/BasicPluginCommon.pm )

check_required_components(cetlib)
```