Memo CMake

Pour plus d'informations, contactez moi à l'adresse : baudouin.leo@gmail.com.

1 Aide en ligne

https://cmake.org/cmake/help/v3.14/

2 Fonctions de base

— Mettre des commentaires :
#Commentaire
— Affiche un message lors de l'appel à CMake :
MESSAGE([FATAL_ERROR|WARNING|...] "message")
— Définir la version minimale de CMake :
CMAKE_MINIMUM_REQUIRED(VERSION 2.6)

2.1 Utiliser les variables

— Définir une variable :
SET(variable_name "value")
— Récupérer la valeur d'une variable :
${variable_name}
— Créer des options :
OPTION(variable_name "Description" default_value)
— Lister les fichiers d'un dossier :
FILE(GLOB variable pattern)
Exemple : FILE(GLOB SRCS src/*.cpp)

2.2 Utiliser les cibles

— Ajouter un exécutable :
ADD_EXECUTABLE(exec_name sources)
— Ajouter une bibliothèque :
ADD_LIBRARY(lib_name [static|shared] sources)
— Linker une cible avec une bibliothèque :
TARGET_LINK_LIBRARIES(exec_name libraries)
— Ajoute des dépendances entre les cibles :
ADD_DEPENDENCIES
— Ajouter une cible personnalisée :
ADD_CUSTOM_TARGET
Exemple :

```
ADD_CUSTOM_TARGET(Name [ALL] [command1 [args1...]]
[COMMAND command2 [args2...] ...]
[DEPENDS depend depend depend ... ]
[WORKING_DIRECTORY dir]
[COMMENT comment] [VERBATIM]
[SOURCES src1 [src2...]]
)
```

2.3 Utiliser des bibliothèques

— Chercher un package :
FIND_PACKAGE(package_name [version] [REQUIRED])
Initialise habituellement les variables :
— package_name_FOUND
— package_name_INCLUDE_DIRS

  - package_name_LIBRARY
  - package_name_LIBRARY_DIRS
- Utiliser un dossier contenant des bibliothèques :
  LINK_DIRECTORIES(folder)
- Ajouter un dossier à la liste des dossiers à inclure :
  INCLUDE_DIRECTORIES(folder)

2.4 Autre

- Installer un fichier :
  INSTALL
  Exemple :
  ```
  INSTALL(FILES my_file
  DESTINATION output_folder
  PERMISSIONS OWNER_READ GROUP_READ WORLD_READ OWNER_WRITE
  )
  ```
- Remplacer les variables par leurs valeurs dans un fichier :
  CONFIGURE_FILE(input_file output_file)
  On utilise généralement des fichiers ".in"
- Ajouter une définition :
  ADD_DEFINITION("-DUSE_MYLIB")
  Vous pouvez utiliser dans votre code C/C++ :
  `#ifdef USE_MYLIB`
- Exécuter une commande personnalisée :
  ADD_CUSTOM_COMMAND
  Exemple :
  ```
  ADD_CUSTOM_COMMAND(OUTPUT output1 [output2 ...]
  COMMAND command1 [ARGS] [args1...]
  [COMMAND command2 [ARGS] [args2...] ...]
  [MAIN_DEPENDENCY depend]
  [DEPENDS [depends...]]
  [IMPLICIT_DEPENDS <lang1> depend1 ...]
  [WORKING_DIRECTORY dir]
  [COMMENT comment] [VERBATIM] [APPEND]
  )
  ```
- Exécuter un programme externe :
  EXEC_PROGRAM
  Exemple :
  ```
  EXEC_PROGRAM(Executable [directory in which to run]
  [ARGS <arguments to executable>]
  [OUTPUT_VARIABLE <var>]
  [RETURN_VALUE <var>]
  )
  ```

2.5 Test unitaires

- Active les tests unitaires :
  ENABLE_TESTING()
- Ajouter un test unitaire :
  ADD_TEST(test_name exec_name [arguments])
  Nécessite d'appeler une fois la fonction ENABLE_TESTING()

3 Variables de base

- Définir le projet :
  ```
  SET(PROJECT_NAME name)
  SET(PROJECT_DESCRIPTION "")
  SET(PROJECT_URL "")
  ```

SET(PROJECT_VERSION "0.0.1")
SET(PROJECT_REQUIREMENTS "")

— Dossier contenant le premier CMakeLists.txt :
CMAKE_SOURCE_DIR
— Dossier contenant le CMakeLists.txt courant :
CMAKE_CURRENT_SOURCE_DIR
— Dossier contenant le projet courant :
PROJECT_SOURCE_DIR
— Dossier de sortie du premier CMakeLists.txt :
CMAKE_BINARY_DIR
— Dossier de sortie du CMakeLists.txt courant :
CMAKE_CURRENT_BINARY_DIR
— Dossier de sortie du projet courant :
PROJECT_BINARY_DIR
— Dossier d'installation :
CMAKE_INSTALL_PREFIX
— Liste des dossiers pour FIND_PATH() :
CMAKE_INCLUDE_PATH
— Liste des dossiers pour FIND_LIBRARY() :
CMAKE_LIBRARY_PATH
— Liste des dossiers pour FIND_PACKAGE(), FIND_PATH(), FIND_PROGRAM() et FIND_LIBRARY() :
CMAKE_PREFIX_PATH
— Options de compilation :
CMAKE_CXX_FLAGS , CMAKE_CXX_FLAGS_DEBUG , CMAKE_CXX_FLAGS_RELEASE
— Choisir le compilateur :
CMAKE_CXX_COMPILER
— Compiler les bibliothèques en mode dynamique :
SET(BUILD_SHARED_LIBS ON)
— Définir le dossier de sortie des exécutables :
RUNTIME_OUTPUT_DIRECTORY
— Définir le dossier de sortie des bibliothèques :
LIBRARY_OUTPUT_DIRECTORY

4 Programmation

4.1 Fonction IF

```
SET( number 4 )
IF( number GREATER 10 )
  MESSAGE( "The number ${number} is too large." )
ELSE()
  MESSAGE( "The number ${number} is ok." )
ENDIF()

SET( x )
IF(DEFINED x)
  MESSAGE("This will never be printed")
ENDIF()
```

4.2 Fonction FOREACH

```
foreach(func ${funcs})
  set(func_name test_${func})
  eval("${func_name}(\"Test\")")
endforeach(func)
```