# Project-level configuration.

cmake\_minimum\_required(VERSION 3.14)

project(project\_manager LANGUAGES CXX)

# The name of the executable created for the application. Change this to change

# the on-disk name of your application.

set(BINARY\_NAME "project\_manager")

# Explicitly opt in to modern CMake behaviors to avoid warnings with recent

# versions of CMake.

cmake\_policy(SET CMP0063 NEW)

# Define build configuration option.

get\_property(IS\_MULTICONFIG GLOBAL PROPERTY GENERATOR\_IS\_MULTI\_CONFIG)

if(IS\_MULTICONFIG)

set(CMAKE\_CONFIGURATION\_TYPES "Debug;Profile;Release"

CACHE STRING "" FORCE)

else()

if(NOT CMAKE\_BUILD\_TYPE AND NOT CMAKE\_CONFIGURATION\_TYPES)

set(CMAKE\_BUILD\_TYPE "Debug" CACHE

STRING "Flutter build mode" FORCE)

set\_property(CACHE CMAKE\_BUILD\_TYPE PROPERTY STRINGS

"Debug" "Profile" "Release")

endif()

endif()

# Define settings for the Profile build mode.

set(CMAKE\_EXE\_LINKER\_FLAGS\_PROFILE "${CMAKE\_EXE\_LINKER\_FLAGS\_RELEASE}")

set(CMAKE\_SHARED\_LINKER\_FLAGS\_PROFILE "${CMAKE\_SHARED\_LINKER\_FLAGS\_RELEASE}")

set(CMAKE\_C\_FLAGS\_PROFILE "${CMAKE\_C\_FLAGS\_RELEASE}")

set(CMAKE\_CXX\_FLAGS\_PROFILE "${CMAKE\_CXX\_FLAGS\_RELEASE}")

# Use Unicode for all projects.

add\_definitions(-DUNICODE -D\_UNICODE)

# Compilation settings that should be applied to most targets.

#

# Be cautious about adding new options here, as plugins use this function by

# default. In most cases, you should add new options to specific targets instead

# of modifying this function.

function(APPLY\_STANDARD\_SETTINGS TARGET)

target\_compile\_features(${TARGET} PUBLIC cxx\_std\_17)

target\_compile\_options(${TARGET} PRIVATE /W4 /WX /wd"4100")

target\_compile\_options(${TARGET} PRIVATE /EHsc)

target\_compile\_definitions(${TARGET} PRIVATE "\_HAS\_EXCEPTIONS=0")

target\_compile\_definitions(${TARGET} PRIVATE "$<$<CONFIG:Debug>:\_DEBUG>")

endfunction()

# Flutter library and tool build rules.

set(FLUTTER\_MANAGED\_DIR "${CMAKE\_CURRENT\_SOURCE\_DIR}/flutter")

add\_subdirectory(${FLUTTER\_MANAGED\_DIR})

# Application build; see runner/CMakeLists.txt.

add\_subdirectory("runner")

# Generated plugin build rules, which manage building the plugins and adding

# them to the application.

include(flutter/generated\_plugins.cmake)

# === Installation ===

# Support files are copied into place next to the executable, so that it can

# run in place. This is done instead of making a separate bundle (as on Linux)

# so that building and running from within Visual Studio will work.

set(BUILD\_BUNDLE\_DIR "$<TARGET\_FILE\_DIR:${BINARY\_NAME}>")

# Make the "install" step default, as it's required to run.

set(CMAKE\_VS\_INCLUDE\_INSTALL\_TO\_DEFAULT\_BUILD 1)

if(CMAKE\_INSTALL\_PREFIX\_INITIALIZED\_TO\_DEFAULT)

set(CMAKE\_INSTALL\_PREFIX "${BUILD\_BUNDLE\_DIR}" CACHE PATH "..." FORCE)

endif()

set(INSTALL\_BUNDLE\_DATA\_DIR "${CMAKE\_INSTALL\_PREFIX}/data")

set(INSTALL\_BUNDLE\_LIB\_DIR "${CMAKE\_INSTALL\_PREFIX}")

install(TARGETS ${BINARY\_NAME} RUNTIME DESTINATION "${CMAKE\_INSTALL\_PREFIX}"

COMPONENT Runtime)

install(FILES "${FLUTTER\_ICU\_DATA\_FILE}" DESTINATION "${INSTALL\_BUNDLE\_DATA\_DIR}"

COMPONENT Runtime)

install(FILES "${FLUTTER\_LIBRARY}" DESTINATION "${INSTALL\_BUNDLE\_LIB\_DIR}"

COMPONENT Runtime)

if(PLUGIN\_BUNDLED\_LIBRARIES)

install(FILES "${PLUGIN\_BUNDLED\_LIBRARIES}"

DESTINATION "${INSTALL\_BUNDLE\_LIB\_DIR}"

COMPONENT Runtime)

endif()

# Fully re-copy the assets directory on each build to avoid having stale files

# from a previous install.

set(FLUTTER\_ASSET\_DIR\_NAME "flutter\_assets")

install(CODE "

file(REMOVE\_RECURSE \"${INSTALL\_BUNDLE\_DATA\_DIR}/${FLUTTER\_ASSET\_DIR\_NAME}\")

" COMPONENT Runtime)

install(DIRECTORY "${PROJECT\_BUILD\_DIR}/${FLUTTER\_ASSET\_DIR\_NAME}"

DESTINATION "${INSTALL\_BUNDLE\_DATA\_DIR}" COMPONENT Runtime)

# Install the AOT library on non-Debug builds only.

install(FILES "${AOT\_LIBRARY}" DESTINATION "${INSTALL\_BUNDLE\_DATA\_DIR}"

CONFIGURATIONS Profile;Release

COMPONENT Runtime)