# Project-level configuration.

cmake\_minimum\_required(VERSION 3.10)

project(runner 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 "todo\_list\_app")

# The unique GTK application identifier for this application. See:

# https://wiki.gnome.org/HowDoI/ChooseApplicationID

set(APPLICATION\_ID "com.example.todo\_list\_app")

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

# versions of CMake.

cmake\_policy(SET CMP0063 NEW)

# Load bundled libraries from the lib/ directory relative to the binary.

set(CMAKE\_INSTALL\_RPATH "$ORIGIN/lib")

# Root filesystem for cross-building.

if(FLUTTER\_TARGET\_PLATFORM\_SYSROOT)

set(CMAKE\_SYSROOT ${FLUTTER\_TARGET\_PLATFORM\_SYSROOT})

set(CMAKE\_FIND\_ROOT\_PATH ${CMAKE\_SYSROOT})

set(CMAKE\_FIND\_ROOT\_PATH\_MODE\_PROGRAM NEVER)

set(CMAKE\_FIND\_ROOT\_PATH\_MODE\_PACKAGE ONLY)

set(CMAKE\_FIND\_ROOT\_PATH\_MODE\_LIBRARY ONLY)

set(CMAKE\_FIND\_ROOT\_PATH\_MODE\_INCLUDE ONLY)

endif()

# Define build configuration options.

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()

# 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\_14)

target\_compile\_options(${TARGET} PRIVATE -Wall -Werror)

target\_compile\_options(${TARGET} PRIVATE "$<$<NOT:$<CONFIG:Debug>>:-O3>")

target\_compile\_definitions(${TARGET} PRIVATE "$<$<NOT:$<CONFIG:Debug>>:NDEBUG>")

endfunction()

# Flutter library and tool build rules.

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

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

# System-level dependencies.

find\_package(PkgConfig REQUIRED)

pkg\_check\_modules(GTK REQUIRED IMPORTED\_TARGET gtk+-3.0)

add\_definitions(-DAPPLICATION\_ID="${APPLICATION\_ID}")

# Define the application target. To change its name, change BINARY\_NAME above,

# not the value here, or `flutter run` will no longer work.

#

# Any new source files that you add to the application should be added here.

add\_executable(${BINARY\_NAME}

"main.cc"

"my\_application.cc"

"${FLUTTER\_MANAGED\_DIR}/generated\_plugin\_registrant.cc"

)

# Apply the standard set of build settings. This can be removed for applications

# that need different build settings.

apply\_standard\_settings(${BINARY\_NAME})

# Add dependency libraries. Add any application-specific dependencies here.

target\_link\_libraries(${BINARY\_NAME} PRIVATE flutter)

target\_link\_libraries(${BINARY\_NAME} PRIVATE PkgConfig::GTK)

# Run the Flutter tool portions of the build. This must not be removed.

add\_dependencies(${BINARY\_NAME} flutter\_assemble)

# Only the install-generated bundle's copy of the executable will launch

# correctly, since the resources must in the right relative locations. To avoid

# people trying to run the unbundled copy, put it in a subdirectory instead of

# the default top-level location.

set\_target\_properties(${BINARY\_NAME}

PROPERTIES

RUNTIME\_OUTPUT\_DIRECTORY "${CMAKE\_BINARY\_DIR}/intermediates\_do\_not\_run"

)

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

# them to the application.

include(flutter/generated\_plugins.cmake)

# === Installation ===

# By default, "installing" just makes a relocatable bundle in the build

# directory.

set(BUILD\_BUNDLE\_DIR "${PROJECT\_BINARY\_DIR}/bundle")

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

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

endif()

# Start with a clean build bundle directory every time.

install(CODE "

file(REMOVE\_RECURSE \"${BUILD\_BUNDLE\_DIR}/\")

" COMPONENT Runtime)

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

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

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)

foreach(bundled\_library ${PLUGIN\_BUNDLED\_LIBRARIES})

install(FILES "${bundled\_library}"

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

COMPONENT Runtime)

endforeach(bundled\_library)

# Copy the native assets provided by the build.dart from all packages.

set(NATIVE\_ASSETS\_DIR "${PROJECT\_BUILD\_DIR}native\_assets/linux/")

install(DIRECTORY "${NATIVE\_ASSETS\_DIR}"

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

COMPONENT Runtime)

# 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.

if(NOT CMAKE\_BUILD\_TYPE MATCHES "Debug")

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

COMPONENT Runtime)

endif()