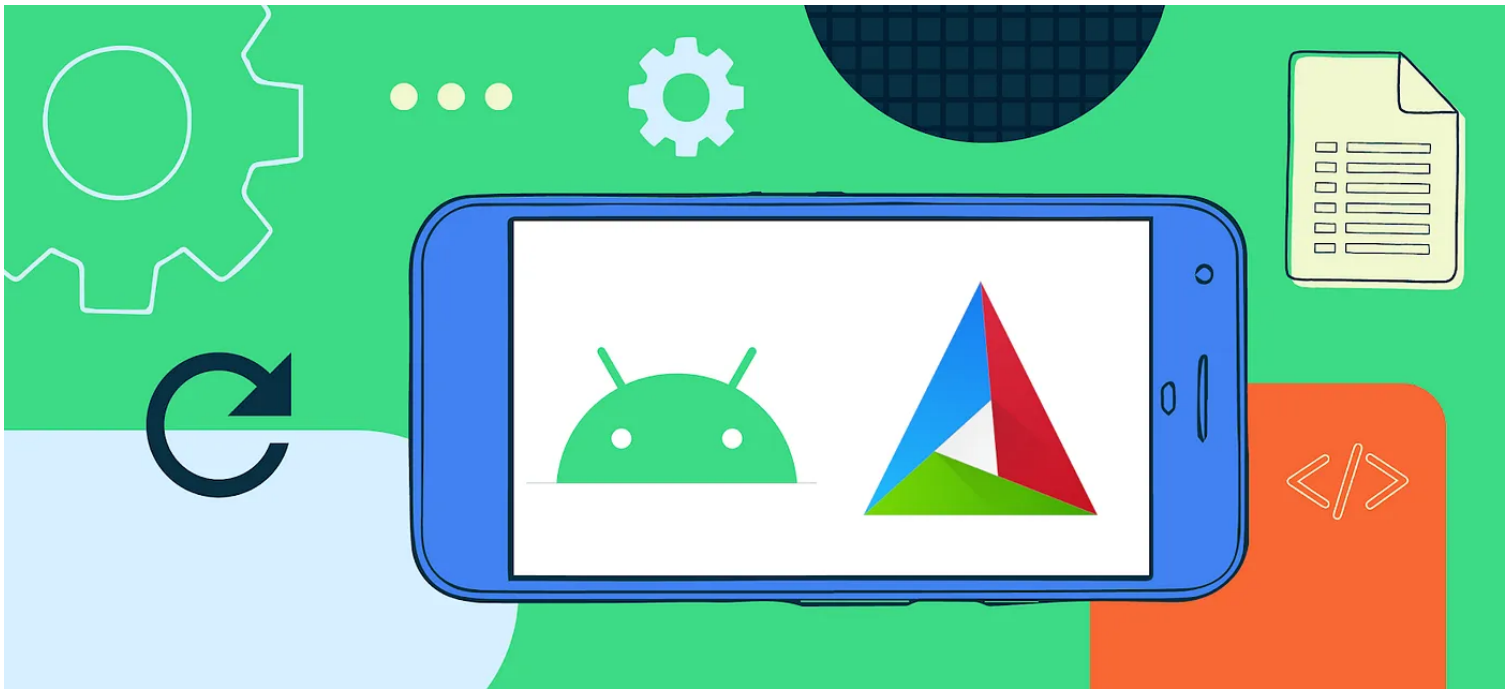
ANDROID NDK

How to link C/C++ library to your Android project with CMake

Depending on if you have the source code, a prebuilt shared or static library, learn how to import a C/C++ library in your project.

Torcheux Frédéric · Follow
4 min read · Jun 12, 2022



First steps to know how to use C/C++ code in an Android application can be hard for an Android developer.

We are used to code in Java/Kotlin, to use Gradle to manage the compilation of our app and now we need to manage C/C++ languages, to use the JNI framework to do the bridge between Java code and C/C++ code and to manage the compilation with CMake.

Particularly, CMake can be pretty hard to use at the beginning.

Goal of this article

To explain how to add a native dependency to your existing C/C++ code in the following cases:

1. You have the **source code** of a library with a CMakeLists file configured to build it.
2. You have a prebuilt **shared** library (**.so**).
3. You have a prebuilt **static** library (**.a**).

Add a dependency to the source code of a library which has a CMakeLists

You have the source code of a library, a CMakeLists to build that library and you want to link it to your library.

We should add a link from the app CMakeLists to the CMakeLists of the library.

To do that, we need to call the command **add_subdirectory:**

```
add_subdirectory(
    /path/to/the/directory/with/CMakeLists/
)
```

In the case where the targeted directory is not a subdirectory of the source, we must add one more parameter:

```
add_subdirectory(
    /path/to/directory/with/CMakeLists/
    ./name_of_the_directory_with_CMakeLists
)
```

For example, if the CMakeLists of the libray is in the directory at the path "*/my-project/subdirectory*" and your own CMakeLists is in the directory at the path "*/my-project/app/src/main/cpp*", the command must be:

```
add_subdirectory(
    ${CMAKE_CURRENT_SOURCE_DIR}/../../../../subdirectory
    ./subdirectory
)
```

If the library doesn't expose its public headers, you should import them by yourself. Call the command **include_directories** with the paths of the directories where the headers are.

In our case, all public headers are located in an *include* directory next to the CMakeLists:

```
include_directories(
    ${CMAKE_CURRENT_SOURCE_DIR}/../../../../subdirectory/include/
```

```
)
```

The last step is to link the library we are adding to our application library. The command here is **target_link_libraries:**

```
target_link_libraries(
    ${PROJECT_NAME}
    subdirectoryLibrary
)
```

- The first parameter is the name of your library, here it's the project name.
- The second parameter is the name of the library we are adding, the name defined in the library CMakeLists.

Now, the dependency will be built at the same time as the native code of your Android application. You can use functions from the library inside your C/C++ code.

Add a dependency to a prebuilt library (.so or .a)

You have a prebuilt library in the form of a **.so** file or **.a** file. The way to depend on one of them is pretty the same way as for the other.

The first step is to indicate that we will add a library to our project. For this, call **add_library** with several parameters:

```
add_library(
    my_prebuilt_library
    SHARED
    IMPORTED
)
```

- The first parameter is the name you want to use to refer to this library in other commands. Now, we will use "my_prebuilt_library"
- The second parameter is the type of library you want to import: **STATIC** (**.a** file) or **SHARED** (**.so** file).
- The word **IMPORTED** is used to indicate that we will refer to a library file outside our project. No build rules are generated for this library due to this parameter, which is logic because we have a prebuilt library file.

For now, nothing has been linked. Following commands will allow to link the prebuilt file to our project.

The next command to call is **set_target_properties:**

```
set_target_properties(
    my_prebuilt_library
    PROPERTIES
    IMPORTED_LOCATION /path/to/prebuilt/file
)
```

- The first parameter is the same name used for the first command.
- The word **PROPERTIES** has to be used as the second parameter.
- The last line allows to define a property with its value. By using the **IMPORTED_LOCATION** property, we can define the path of our prebuilt library file.

The next command is to link the prebuilt library to our library. It is **target_link_libraries:**

```
target_link_libraries(
    ${PROJECT_NAME}
    my_prebuilt_library
)
```

- The first parameter is the name of the our library where we want to add the dependency.
- The second parameter is again the same name to defined the library.

There is one last step to do before to use the library. We need to add the library headers to be able to use them in our code.

Call **include_directories** and pass the directory path with public headers inside as parameter:

```
include_directories(
    ${CMAKE_CURRENT_SOURCE_DIR}/../../my_prebuilt_library/include/
)
```

Now, you can call functions of the prebuilt library inside your code.

The prebuilt library will be linked to the native code of your Android application. If it's a shared library, the **.so** will be visible inside the final APK. If it's a **.a**, it will be merged inside your own libray.

Summary

Now, you know how to add dependencies to a native library for your Android application using CMake.

We looked at the 3 ways to add dependency and we discovered some new commands of CMake to do that.

You are now able to use any native library for your project in order to build the best Android application!

- Codelab to create a Hello-CMake
- Android NDK sample
- Configure CMake from developer android web site
- Configure CMake from android ndk web site

AndroidDev | Android Ndk | Cmake | Developer | Android App Development

Torcheux Frédéric

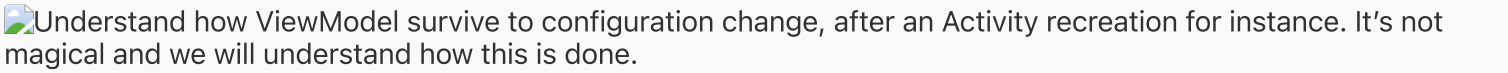
Follow

Written by Torcheux Frédéric

56 Followers

I'm a French Android developer at MWM. I humbly try to contribute to the developer community from which I learnt everything.

More from Torcheux Frédéric



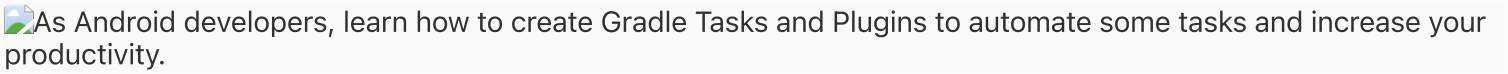
Torcheux Frédéric in ProAndroidDev

How ViewModel works under the hood

Learn how a ViewModel survives to a configuration change

5 min read · Jan 13

-- 5



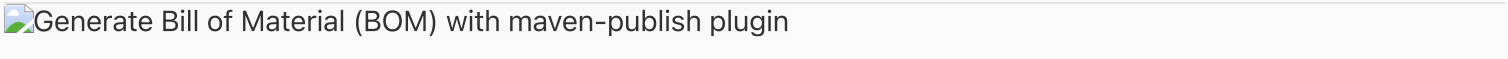
Torcheux Frédéric in ProAndroidDev

Create your own Gradle Tasks and Plugins in less than 10 minutes

Make Gradle work for you

8 min read · Apr 3

-- 1



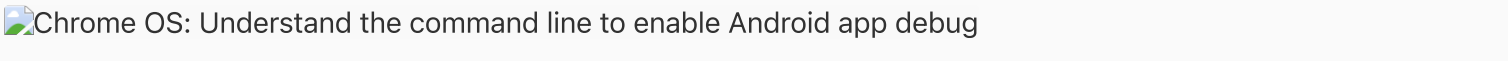
Torcheux Frédéric in by MWM

Generate Bill of Material (BOM) with maven-publish plugin

Define one “cookbook” dependency to provide versions of your libraries compatible between them

4 min read · Oct 22, 2020

--



Torcheux Frédéric

Chrome OS: Understand the command line to enable Android app debug

Discover how to configure your ChromeBook and how USB-C works.

3 min read · Jan 10, 2020

--

See all from Torcheux Frédéric

Recommended from Medium

Maxi Rosson in Dipien

10 ideas to reduce your APK size [Part II]

Less is More: A Comprehensive Guide to Reducing APK Size and Optimizing Your Android App's User Experience

· 7 min read · Mar 10

--

inVerita

12 Best Android Development Tools & Software

The Google Play store offers more than 2.67 million apps available for download as of March 2023, according to Statista, beating Apple App…

· 9 min read · Mar 22

--

Lists

General Coding Knowledge

20 stories · 42 saves

Medium Publications Accepting Story Submissions

145 stories · 186 saves

Now in AI: Handpicked by Better Programming

248 stories · 18 saves

Maxi Rosson in Dipien

10 ideas to reduce your APK size [Part III]

Less is More: A Comprehensive Guide to Reducing APK Size and Optimizing Your Android App's User Experience

· 7 min read · Mar 12

--

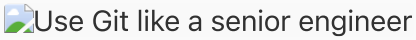
Carlos Aldea

5 Reasons To Become an Embedded SW Developer

And keep yourself as far as possible from web development

✨ · 2 min read · Jan 13

-- 5



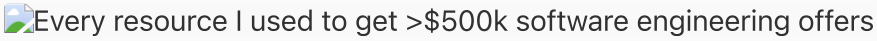
Jacob Bennett in Level Up Coding

Use Git like a senior engineer

Git is a powerful tool that feels great to use when you know how to use it.

✨ · 4 min read · Nov 15, 2022

-- 69



Santal Tech in Tech Pulse

Every resource I used to get >$500k software engineering offers

A list of every resource I used to clear technical interviews from top tech companies. If I can do it, so can you.

✨ · 4 min read · Jan 6

-- 6

See more recommendations