Get started with the NDK

The Native Development Kit (NDK) is a set of tools that allows you to use C and C++ code with Android, and provides platform libraries (/ndk/guides/stable_apis) you can use to manage native activities and access physical device components, such as sensors and touch input. The NDK may not be appropriate for most novice Android programmers who need to use only Java code and framework APIs to develop their apps. However, the NDK can be useful for cases in which you need to do one or more of the following:

- Squeeze extra performance out of a device to achieve low latency or run computationally intensive applications, such as games or physics simulations.
- Reuse your own or other developers' C or C++ libraries.

Using Android Studio 2.2 and higher (/studio), you can use the NDK to compile C and C++ code into a native library and package it into your APK using Gradle, the IDE's integrated build system. Your Java code can then call functions in your native library through the Java Native Interface (JNI) (http://docs.oracle.com/javase/7/docs/technotes/guides/jni/spec/jniTOC.html) framework. To learn more about Gradle and the Android build system, read Configure Your Build (/studio/build).

Android Studio's default build tool to compile native libraries is CMake (https://cmake.org/). Android Studio also supports ndk-build (/ndk/guides/ndk-build) due to the large number of existing projects that use the build toolkit. However, if you are creating a new native library, you should use CMake.

This guide gives you the information you need to get up and running with the NDK on Android Studio. If you don't have the latest version of Android Studio, download and install it now (/studio).

Attention experimental Gradle users: Consider migrating to plugin version 2.2.0 or higher (http://tools.android.com/tech-docs/new-build-system/gradle-experimental/migrate-to-stable), and using CMake or ndk-build to build your native libraries if any of the following apply to you: Your native project already uses CMake or ndk-build; you would

rather use a stable version of the Gradle build system; or you want support for add-on tools, such as CCache (https://ccache.samba.org/). Otherwise, you can continue to use the experimental version of Gradle and the Android plugin (http://tools.android.com/tech-docs/new-build-system/gradle-experimental).

Download the NDK and tools

To compile and debug native code for your app, you need the following components:

- The Android Native Development Kit (NDK): a set of tools that allows you to use C and C++ code with Android.
- CMake: an external build tool that works alongside Gradle to build your native library. You do not need this component if you only plan to use ndk-build.
- *LLDB*: the debugger Android Studio uses to debug native code.

For information on installing these components, see Install and configure the NDK and CMake (/studio/projects/install-ndk).

Create or import a native project

Once you set up Android Studio, you can simply Create a New Project with C/C++ Support (/studio/projects/add-native-code#new-project). However, if you want to add or import native code to an existing Android Studio project, you need to follow this basic process:

1. Create new native source files (/studio/projects/add-native-code#create-sources) and add them to your Android Studio project.
    - You can skip this step if you already have native code or want to import a prebuilt native library.
2. Create a CMake build script (/studio/projects/configure-cmake#create_script) to tell CMake how to build your native sources into a library. You also require

this build script if you are importing and linking against prebuilt or platform libraries.

- You can skip this step if your existing native library already has a `CMakeLists.txt` build script, or uses ndk-build and includes an `Android.mk` (/ndk/guides/android_mk) build script.

3. Link Gradle to your native library (/studio/projects/add-native-code#link-gradle) by providing a path to your CMake or ndk-build script file. Gradle uses the build script to import source code into your Android Studio project and package your native library (the SO file) into the APK.

★ **Note:** If your existing project uses the deprecated **ndkCompile** tool, you should open your **build.properties** file and remove the following line of code before configuring Gradle to use CMake or ndk-build:

```
// Remove this line
android.useDeprecatedNdk = true
```

4. Build and run your app (/studio/run) by clicking **Run**  . Gradle adds your CMake or ndk-build process as a dependency to compile, build, and package your native library with your APK.

Once your app is running on a physical device or the emulator, you can use Android Studio to Debug your app (/studio/debug). Otherwise, to learn more about the NDK and its components, read the Concepts (/ndk/guides/concepts) page.

Last updated 2020-09-30 UTC.