The DDLC Mod Template Guide to Android Mod Making

Works on Ren'Py 6.99.12.4 and Ren'Py 7.4.5

Guide Version 1.41
Author: GanstaKingofSA#0235 | u/GanstaKingofSA

For DDLC Mod Template Versions: 2.4.0 and higher

Disclaimer: This guide is not endorsed by Team Salvato. Modding should be done when you have completed DDLC. This is not meant to port mods over to Android without mod author approval nor port DDLC by itself. Do not release other people's mods without their permission or release DDLC onto the Google Play Store, or any other first/third party Android App Store.

This guide will not work for Doki Doki Literature Club Plus.

Contents

Setting Up The Template

To setup the template on a fresh copy of DDLC to use the Android feature, you will need the following.

Note for MacOS Users: To access the contents of *DDLC.app*, right-click *DDLC.app* in MacOS and click Show Package Contents.

1. A system with at least **8 GB** of RAM minimum.
    a. Depending on the size of your mod, the RAM requirements listed here may increase or decrease. Mods which mod files, along with extracted DDLC assets that exceed around 500MB will need more than 4 GB RAM in order to build properly.

2. A **fresh copy** of DDLC from either DDLC.moe or from Steam.

3. **DDLC Mod Template Version 2.4.0** or higher from GanstaKingofSA/DDLCModTemplate2.0's repository.
    a. This will ***not*** work on the *Monika-After-Story/RationalPi* template or by decompiling the game yourself. For more information, see **Upgrading Ren'Py/The Template**.

4. **Ren'Py 6.99.12.4 (Ren'Py 6)** or **Ren'Py 7.4.5 (Ren'Py 7.4) SDK**
    a. Ren'Py 7.3.5 – 7.4.4 and Ren'Py 7.4.6 and higher will not work for Android Mod Building. For more information, see **Upgrading Ren'Py/The Template**.
    b. If you are using *Ren'Py 6* (6.99.12.4)
        i. Download the Ren'Py 6.99.12.4 SDK by downloading it here. Download either the *7z exe*, *dmg*, or *tar.bz2 package* for your respective system and extract it somewhere.
    c. If you are using *Ren'Py 7* (7.4.5)
        i. Download the Ren'Py 7.4.5 SDK by downloading it here. Download either the *7z exe*, *dmg*, or *tar.bz2 package* for your respective system and extract it somewhere.
    d. Make a folder for your mod inside the extracted Ren'Py SDK folder.

5. **Visual Studio Code (VSC) +** the **Ren'Py Extension** from Microsoft.
    a. To install the Ren'Py Extension, go to *Extensions* (Button with 4 boxes on the left side), search *Ren'Py* and install first one listed in the results.

6. **Eclipse Temurin OpenJDK 8** from Adoptium.
    a. Building your mod while having Eclipse Temurin OpenJDK 8 & 11 installed may cause problems. See **Notes** Section 4 for more information.

7. **RPATool** from Shizmob/rpatool's repository.

8. **Python** from Python.org.
    a. Make sure to check *Add python.exe to path* when installing Python to your system.

9. **An android phone** with USB Debugging **On** or **Android Studio**
    a. If you are using your own android phone:
        i. To turn on USB debugging, find *Build Number*, and tap it 5-8 times repeatedly until it says Developer mode has been enabled. Then go to *Developer Options* and find *USB debugging*, then turn it on.
            1. A prompt may appear on your phone saying if you want to trust your computer for USB debugging. Say *Yes* to it.
            2. To find the build number or developer options, Google your device and USB debugging (Galaxy S9 USB Debugging) for more information.
    b. If you prefer to use Android Studio instead:
        i. Make sure to have *Android Virtual Device* checked during setup and install everything (including Intel HAXM or the AMD equivalent). Next, go to *Tools*, then *AVD Manager* and click the *Play* button on the side of the first option to start a virtual android phone for debugging use.

Installation Steps

Note: If you are upgrading your template to 2.4.X, followed a guide, or used a tool to make your mod project, please read the **Upgrading Ren'Py/The Template** before proceeding.

1. Open the *ddlc-win.zip* file and double-click the *DDLC-1.1.1-pc* folder. Copy its contents to the folder you made in the Ren'Py SDK folder.
    a. MacOS Users with a *ddlc-mac* folder
        i. Copy the contents inside the *ddlc-mac* folder to the folder you made in the Ren'Py SDK folder.
    b. MacOS Users with a *ddlc-mac* ZIP file
        i. Open the *ddlc-mac.zip* file and copy its contents to the folder you made in the Ren'Py SDK folder.

2. Open the *mod template ZIP file* and copy its contents to the folder you made in the Ren'Py SDK folder. Accept any replaces.
    a. MacOS Users with a *DDLCModTemplate-2.4.X* folder
        i. Copy the contents inside the *DDLCModTemplate-2.4.X* folder to *DDLC.app/Contents/Resources/autorun*. Accept any replaces.
    b. MacOS Users with a *DDLCModTemplate-2.4.X* ZIP file
        i. Open the *ddlc-mac.zip* file and copy its contents to *DDLC.app/Contents/Resources/autorun*. Accept any replaces.

3. Open the *rpatool-master ZIP* file and double click the *rpatool-master* folder. Copy *rpatool* to the *game* folder.
    a. MacOS Users with a *rpatool-master* folder
        i. Copy *rpatool* from within the *rpatool-master/rpatool*-master folder to *DDLC.app/Contents/Resources/autorun/game*.
    b. MacOS Users with a *rpatool-master* ZIP file
        i. Open the *rpatool-master ZIP* file and double click the *rpatool-master* folder inside the ZIP. Copy *rpatool* to *DDLC.app/Contents/Resources/autorun/game*.

4. Run DDLC via DDLC.exe, DDLC.app, or via the Linux Terminal by typing ./DDLC.sh in the directory it is in.

5. If the game runs without any errors, you can exit the game and proceed to make your mod and follow the rest of the guide to setting up your mod for Android.
    a. If DDLC doesn't start for the first time, try launching the game again. If that doesn't work, look for any traceback or error files in the game's base folder; else, report the issue you are facing to the DDLCModTemplate2.0 repository.

Upgrading Ren'Py/The Template

Upgrading from the Monika-After-Story/RationalPi Template (Versions 0.2.0 to 1.1.2)

This update has major code changes to be like DDLC which the Monika-After-Story/RationalPi template lacks; especially the android code needed to run the game on Android. It is highly suggested to move all your existing code to the 2.4.X files to retain all DDLC functionality, your past work, and have Android compatibility in your mod.

Upgrading from Versions 2.1.X to 2.3.1 (includes 2.3.1-uX versions)

Due to the Android code, some functionality will require a transfer of code to 2.4.X. I suggest transferring the code you have done to the 2.4.X files to retain your past work and have Android compatibility in your mod.

Upgrading from Versions 2.4.0 to 2.4.5 (for templates above 2.4.5)

Templates above 2.4.5 shouldn't be affected too much as the Android code in the template has just been optimized and fixed slightly. If you do plan to upgrade, please note the following.

- *package_name* in options.rpy has become obsolete in favor of environment paths.
- *script-poemgame*, *splash*, *script*, *options*, and *definitions.rpy* have changed a bit with its Android code; similar to *options.rpy*.

Android by following a DDLC Guide (Tormuse's Mod Guide, etc.)

Some DDLC guides will show you that in order to make your first DDLC mod, you will have to download decompiled scripts or show you how to decompile the whole game with RPATool. This, however, will not make your mod work on Android as it needs Android specific code for the game to run properly on the platform. If you plan to make your mod work on Android, do not decompile the game or download these files. Rely on the mod template files itself for your Android mod and this guide to help you make your mod work on Android.

Android via a tool-made project (DDMM/DDMMaker)

If you made your project in DDMMaker, you should be fine following this guide as the tool uses the correct mod template. Any future tools made may or may not work with this guide's steps, it's provided mod template and such.

Android under Ren'Py 7.3.5 – 7.4.4 or 7.4.6 and higher.

Due to BinTray shutting down, Ren'Py versions older than 7.4.5 will not be able to compile mods on Android anymore. There is no solution to resolve this problem apart from upgrading the SDK to 7.4.5 for now.

Using Ren'Py versions higher than 7.4.5 will not work as they break DDLC heavily with it's transforms and certain versions of the mod template will lock you out from modding on these versions of Ren'Py.

Building the Mod for Android

Read the steps carefully before doing anything. This section assumes you only followed the steps in **Installation Steps**.

Note for DDMM/DDMMaker users: When following this guide, refer to Ren'Py Launcher as DDMM/DDMMaker.

1. Open Terminal, Command Prompt or PowerShell in the game folder and run the following commands. Make sure you **backup** your mod files before your run them or they might get overridden with DDLC's own files.
    a. *python rpatool -x audio.rpa*
    b. *python rpatool -x fonts.rpa*
    c. *python rpatool -x images.rpa*
        i. If you get a *Python was not found* and not a '*python' is not recognized* error, see **Notes** Section 7 for more information.
        ii. If you get a '*python' is not recognized* error, then Python is not installed properly. Refer to **Setting up the Template** Section 8 and its details for more information.
        iii. **Keep the RPA files in the game folder!** At minimum, remove the archive check in *splash.rpy* if you must or Ren'Py will not compile your mod due to a DDLC archive error. Don't worry, the RPAs won't be included in the final APK release.

2. Start Ren'Py Launcher and select your mod from the Projects Sidebar, then click *Android*.
    a. You will be asked to download RAPT (Ren'Py Android Platform Tools). Select *Yes* and it will download it. It will restart Ren'Py Launcher after it installs.

3. Go back to the *Android* section and click *Install SDK and Select Keys*. It will download the Android SDK and other files. Accept the Android Terms and say *Yes* to setting up a key and to backing it up. Set your organization name to your name.
    a. If the *Install SDK and Select Keys* is grayed-out, you are either missing AdoptOpenJDK8 or misconfigured it. Refer to **Setting Up The Template** Section 6 for more information along with **Notes** Section 4 for issues related to AdoptOpenJDK8 and AdoptOpenJDK11 on the same system.

4. Click *Configure* and follow the steps below to setup the mod configuration for Android.
        i. For Full Name, set it as your mod name.
        ii. For Short Name, set it to your mod name's abbreviation.

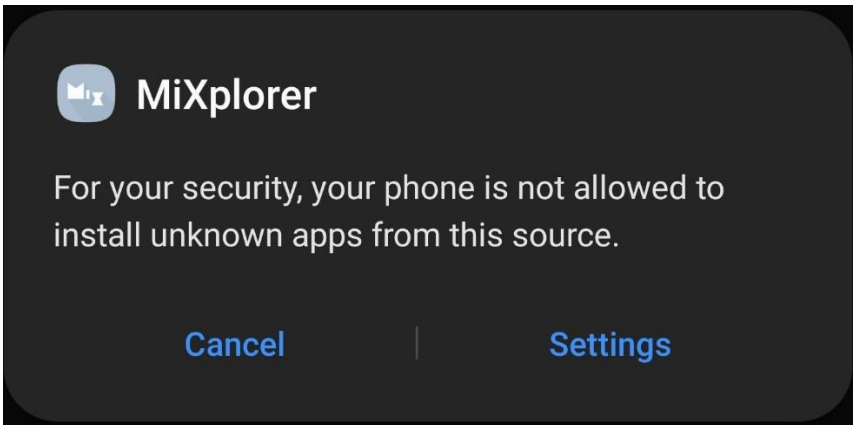
iii. For Package Name, use *com.* then your name with a period and the build name of your mod. For example, if your name is Eileen and your build name for your mod is TheQuestion in *options.rpy* (under *build.name*), it will be formatted like this: *com.eileen.thequestion*
iv. For Application Version, set it to the version number of your mod.
v. For Version Code, leave it as is.
vi. (For 7.4.5) For RAM Allocation to Gradle, leave it as-is.
    1. If you run into *java.lang.OutOfMemoryError* when building your mod, return to *Configure* and increase this number. Remember to not exceed your computer's RAM size. Refer to **Notes** Section 5 for more information.
vii. For Display, set it to *landscape orientation*.
viii. For app store in-app purchasing, select *Neither*.
ix. Select *No* to creating an expansion APK.
x. (For 6.99.12.4) For version of Android, select *Android 4.0*.
xi. Select *No* to allowing the mod to access the internet.
xii. (For 7.4.5) Select *Yes* to automatically update your project.

5. Select *Build Package* and your mod should compile to a APK to be used on Android. You may then sideload it to your phone and test it for errors before release.
    a. For Ren'Py 6, your mod will be compiled with the following name format. Here is an example with mod name *Cookie Club* and version number *2.3.0*
        i. CookieClub-2.3.0-release.apk
    b. For Ren'Py 7, your mod will be compiled and output 4 APK files. For universal compatibility between different phones, use the following name format APK. Here is an example with author name *Natsuki*, mod build name *mlhnstlc* and version number *1.0.0*
        i. com.natsuki.mlhnstlc-10000-universal-release.apk
    c. Refer to **Notes** Section 5, if you encountered a *java.lang.OutOfMemoryError* when building your mod.

6. After you tested that your mod works as is, upload it to Google Drive, Mega, etc. and your port should be complete.
    a. Make sure to add to your post that the user must turn on *Unknown Sources* on their phones temporarily and of a Google Play Protect warning that may occur and to tap *Install Anyways*. Refer to **Notes** Sections 1-3 for more details.

Notes

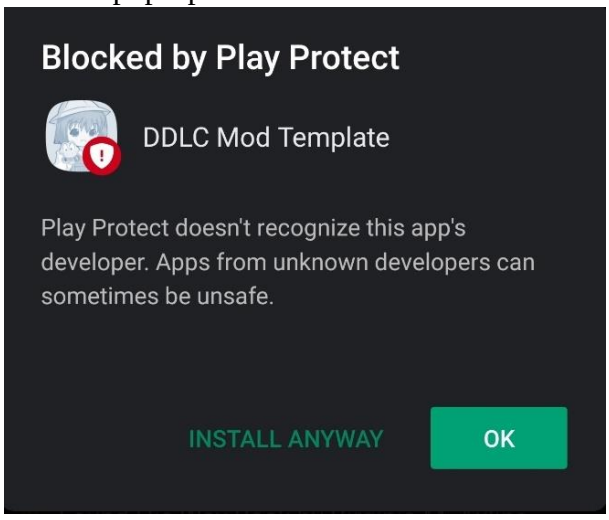
1. When installing your mod either via a file manager or by downloading off your browser, etc. you may get this pop-up by Android or something similar.



To fix this, tap *Settings* and turn on Unknown Sources. Android will warn you that installing apps from unknown sources may be malicious and some devices allow you to turn this on temporarily which is fine. Select either *OK* or *Install Just Once* then select *Install*.

- This step will differ from device to device. Consult Google for more information on turning on unknown sources for your device.

2. When installing your mod either via a file manager or by downloading off Chrome, etc. you may get this Play Protect pop-up.



This is Google detecting that you are someone they do not recognize as a developer and think the app may be malicious. Tap Install Anyway and the installation should proceed as normal.

3. While running the game you might get a pop-up that asks whether to send your mod's APK file to Google for analysis. You can choose either option presented as this is just a Google Security Message.

4. If Eclipse Temurin OpenJDK 8 and 11+ are installed on the same computer, they may cause an error in Ren'Py in which it cannot build your mod due to outdated code. Please make sure that only Eclipse Temurin OpenJDK 8 is installed (re-install it if Eclipse Temurin OpenJDK 11+ was uninstalled) or if you still need Eclipse Temurin OpenJDK 11+ on your system, to build your mod on another device or on a virtual machine with just Eclipse Temurin OpenJDK 8.

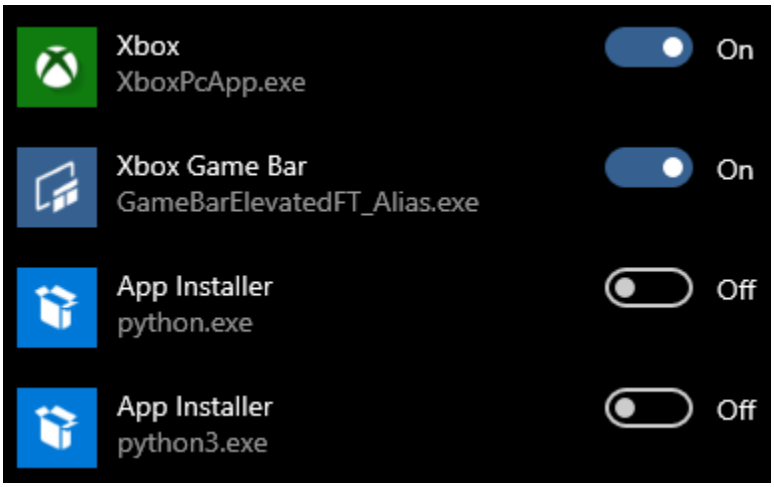
5. Sometimes when building your mod, you may run into an error saying *Caused by: java.lang.OutOfMemoryError*. This error is caused due to Java using all the space it can use for building the mod which is 3 GB. To fix this, go to *rapt/project* and open *gradle.properties* with VSC. Change the three in *-Xmx3g* to a different number. I recommend 5-8 but this depends on how much RAM there is on your system. See Task Manager -> Performance -> RAM for Windows, Apple Logo -> About this Mac for Mac OS, or *neofetch* for Linux (*neofetch* requires you to install it on your system) to see how much RAM you have. Allocate all but 1 GB of the system memory to it i.e 7 GB on a 8GB system.

    - If you are running Ren'Py 7.4.5 or DDMM/DDMMaker 1.2.6 or higher, you can fix this within the Android Menu via *Configure* (Ren'Py Launcher and later DDMM/DDMMaker versions) or *Change RAPT Settings* (DDMM/DDMMaker 1.2.6)

6. When executing the RPATool commands in Windows, you might get this error.

    Python was not found; run without arguments to install from the Microsoft Store, or disable this shortcut from Settings > Manage App Execution Aliases.

    To fix this, go to Windows Search and type *Manage App Execution Aliases*. In it, turn off App Installer for *python.exe*. Next, reinstall Python and check on *Will be installed on local hard drive* for *Add python.exe to path* and continue the installation. Once it completes, re-open Command Prompt or Powershell and type the commands in again.



7. In some rare instances, RPATool may not work if you run the commands with Python 3 and Python 2 installed. To fix this, do *py -2* instead of *python* and try typing the commands again with it.

FAQ

Q. Where are all the game/character files located now?

A. They are located in the APK itself, however some files that can't be loaded from it are copied over to the *Android/data/com.[yourname].[modbuildname]* for deleting files, adding files, etc.

Q. A bunch of my scripts is broken. Why is it the case now?

A. Ren'Py on Android runs differently than on PC. There are different logical patterns needed in Python in Android and such to run properly on all devices. Refer to the template comments or the Ren'Py Documentation for more information.

Q. Does it have to be *com.[yourname].[modbuildname]* to build the mod? It looks cheezy and bland.

A. No. You can change it to be whatever (except the *modbuildname* part), but it must follow the Android naming scheme of *x.y.[modbuildname]* **and** be all lowercased. Consult the other folder naming patterns in the Android/data folder on how to properly name an Android APK data folder.

Q. My GUI elements are all broken. What gives here?

A. The GUI loaded is of DDLC but modified to work on mobile devices. The code for it is located at the bottom section of *gui.rpy*. Change the variables there to adjust your GUI to how it is in-game on PC.

Q. Isn't distributing the APK with DDLC's own files against Team Salvato's IPG?

A. No. Team Salvato acknowledges that this is a limitation on making DDLC mods available on Android and allows it for the time being, however they will keep an eye on how these APK files are distributed and Android modding may be revoked at any time by Team Salvato.

Q. What is the difference between this and JoiPlay?

A. This runs natively on Android rather than emulating a computer. This means it is prone to less bugs and crashes than running mods on JoiPlay; which is known to break many mods and games. Any errors that appear are either the mod itself being packaged incorrectly or not coded to run differently formatted code for Android.

Q. How would I add my files that need to be read, written to or modified in *Android/data/com.[yourname].[modbuildname]* folder like the character files?

A. The mod template files shows some examples on how these files work in the game under Android. Below are a few examples of code that writes files into the Android directory.

This code below shows an example of writing *CAN YOU HEAR ME.txt* to the game's Android directory.

```
python:
    if renpy.android:
        # For Android, the try and excepts must be formatted like so with this but replace
        # hxppy thxughts.png with the file you want to write.
        ## try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/hxppy thxughts.png")
        ## except: open(os.environ['ANDROID_PUBLIC'] + "/hxppy thxughts.png"), "wb").write(renpy.file("hxppy thxughts.png").read())
        try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/hxppy thxughts.png")
        except: open(os.environ['ANDROID_PUBLIC'] + "/hxppy thxughts.png", "wb").write(renpy.file("hxppy thxughts.png").read())
    else:
        try: renpy.file(config.basedir + "/hxppy thxughts.png")
        except: open(config.basedir + "/hxppy thxughts.png", "wb").write(renpy.file("hxppy thxughts.png").read())
```

This code below shows an example of writing the character files into the *characters* folder if a Doki's character file is missing from within the Android directory.

```
import os
if renpy.android: #checks if the platform is android
    try:
        # writes character files if missing and correct playthrough to Android/data/[your mod]/characters
        if not os.access(os.environ['ANDROID_PUBLIC'] + "/characters/", os.F_OK):
            os.mkdir(os.environ['ANDROID_PUBLIC'] + "/characters")
        if persistent.playthrough <= 2:
            try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/characters/monika.chr")
            except: open(os.environ['ANDROID_PUBLIC'] +  "/characters/monika.chr", "wb").write(renpy.file("monika.chr").read())
        if persistent.playthrough <= 1 or persistent.playthrough == 4:
            try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/characters/natsuki.chr")
            except: open(os.environ['ANDROID_PUBLIC'] + "/characters/natsuki.chr", "wb").write(renpy.file("natsuki.chr").read())
            try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/characters/yuri.chr")
            except: open(os.environ['ANDROID_PUBLIC'] + "/characters/yuri.chr", "wb").write(renpy.file("yuri.chr").read())
        if persistent.playthrough == 0 or persistent.playthrough == 4:
            try: renpy.file(os.environ['ANDROID_PUBLIC'] + "/characters/sayori.chr")
            except: open(os.environ['ANDROID_PUBLIC'] + "/characters/sayori.chr", "wb").write(renpy.file("sayori.chr").read())
    except:
        pass
else:
    try:
        if not os.access(config.basedir + "/characters/", os.F_OK):
            os.mkdir(config.basedir + "/characters")
        if persistent.playthrough <= 2:
            try: renpy.file("../characters/monika.chr")
            except: open(config.basedir + "/characters/monika.chr", "wb").write(renpy.file("monika.chr").read())
        if persistent.playthrough <= 1 or persistent.playthrough == 4:
            try: renpy.file("../characters/natsuki.chr")
            except: open(config.basedir + "/characters/natsuki.chr", "wb").write(renpy.file("natsuki.chr").read())
            try: renpy.file("../characters/yuri.chr")
            except: open(config.basedir + "/characters/yuri.chr", "wb").write(renpy.file("yuri.chr").read())
        if persistent.playthrough == 0 or persistent.playthrough == 4:
            try: renpy.file("../characters/sayori.chr")
            except: open(config.basedir + "/characters/sayori.chr", "wb").write(renpy.file("sayori.chr").read())
    except:
        pass
```

This FAQ is not an exhaustive list of questions for Android DDLC modding development. For further questions, ask in the *#mod_help* channel or see the *#mod_faq* channel on the DDMC Discord linked here or contact me in DM for any analysis of what might be wrong. Be advised that not many members will know about Android modding for DDLC and support for it is limited. Explain in depth the issue and someone will try to assist you.

DDMC Discord Invite: https://discord.gg/Gx7m9xE

Acknoledgements

I want to acknowledge the following people for assistance, either voluntary or in-voluntary.

Tom Rothamel i.e. PyTom for Ren'Py and Ren'Py Android functionality.

Dan Salvato for making DDLC and allowing us to mod it to our hearts content that is IP compliant.

My past projects to optimize the mod template with better code.

You for reading this whole guide.

Weiss Schnee for support. (Weiss: 😊)