

# Setting Up Code Modifications

So, you're ready to dive into the code of the game? Let's get started!

In this tutorial, you will learn how to:

- Set up the NSMB DS Code Template
- Set up ARM GCC
- Set up NCPatcher
- Extract/Build your ROM

This guide will cover the "NCPatcher Standalone" method described in the code template as the steps are more synchronized between all operating systems.

## Setting Up the Code Template

1. Head over to the [code template's GitHub](#)
2. Click on **Code -> Download ZIP**
3. Now extract this zip and rename the folder to what you want to call your project (this will be referred to as your **project root**)
4. Don't put any spaces in your folder name!

You have now set up the code template

## Setting up ARM GCC

1. Head to the [Arm GNU Toolchain Download Page](#)
2. Now search (using CTRL/CMD + F) for **AArch32 bare-metal target (arm-none-eabi)** and download the correct installer for your operating system.
3. Open the installer and install the toolchain.
4. Pick a location without spaces to install the toolchain!

You have now set up ARM GCC

## Setting Up NCPatcher

1. Head over to the [NCPatcher GitHub releases page](#)
2. Download the latest release for your operating system
3. Extract NCPatcher

Now, NCPatcher depends on **ncpatcher.json**, so lets make it!

If you'd like to learn more about this file, head over to the [NCPatcher GitHub!](#)

In your **project root**, create the following files:

- **ncpatcher.json**
  - Copy/Paste the following JSON into the file

```
{
  "$arm_flags": "-masm-syntax-unified -mno-unaligned-access -mfloat-abi=soft -mabi=aapcs",
  "$c_flags": "-Os -fomit-frame-pointer -ffast-math -fno-builtin -nostdlib -nodefaultlibs -nostartfiles -DSDK_GCC -DSDK_FINALROM",
  "$cpp_flags": "-fno-rtti -fno-exceptions -std=c++20",
  "$asm_flags": "-Os -x assembler-with-cpp -fomit-frame-pointer",
  "$ld_flags": "-lgcc -lc -lstdc++ --use-blx",

  "backup": "backup",
  "filesystem": "nsmb",
  "toolchain": "arm-none-eabi-",

  "arm7": {},
  "arm9": {
    "target": "arm9.json",
    "build": "build"
  },

  "pre-build": [],
  "post-build": [],

  "thread-count": 0
}
```

You have now set up NCPatcher

## Extracting, Building, and Repackaging Your ROM

### If you're on Windows

1. Download [fireflower.zip](#) and extract it.
2. Move **nds-build.exe** and **nds-extract.exe** out from the folder

### If you're on macOS/Linux

1. Download [nds-extract.zip](#)
2. Download [nds-build.zip](#)
3. Extract both ZIPs

Now, you need to build the tools.

For NDS Extract:

1. Open a new **Terminal** window in the folder of the code
2. Run this command: `g++ nds-extract.cpp -o nds-extract -std=c++20`

For NDS Build:

1. Open a new **Terminal** window in the folder of the code
2. Run this command: `g++ nds-build.cpp -o nds-build -std=c++20`

From here on, the instructions will work for all operating systems. If you are on Windows 10, you can use **Command Prompt** instead of **Terminal**

## Extracting Your ROM

1. Open a **Terminal** window in your **project root**
2. Run this command: `/path/to/nds-extract rom.nds nsmb`

Replace `/path/to/` with the actual file path to nds-extract. Also replace "rom" with the actual name of your .nds file

This will extract the contents of your ROM into a folder named nsmb

You have extracted your ROM

## Building Your ROM

This step will compile and patch your ROM with any code files found in the **source** directory in your **project root**. The Code Template comes with a few examples included in the **source** directory.

1. Open a **Terminal** window in your **project root**
2. Run this command: `/path/to/ncpatcher`

Replace `/path/to/` with the actual file path to `ncpatcher`

You have built your ROM

## Repackaging Your ROM

`nds-extract` depends on **buildrules.txt**, so let's create it!

- **buildrules.txt**
  - Copy/Paste the following text into the file:

```
rom_header NSNDY/header.bin
arm9_entry KEEP
arm9_load KEEP
arm7_entry KEEP
arm7_load KEEP
fnt nsmb/fnt.bin
file_mode ADJUST
arm9 nsmb/arm9.bin
arm7 nsmb/arm7.bin
arm9ovt nsmb/arm9ovt.bin
arm7ovt nsmb/arm7ovt.bin
icon nsmb/banner.bin
rsa_sig nsmb/rsasig.bin
data nsmb/root
ovt_repl_flag 0xFF
ov9 nsmb/overlay9
ov7 nsmb/overlay7
```

This step will take the files from the **nsmb** folder and repackage them into a `.nds` file

1. Open a **Terminal** window in your **project root**
2. Run this command: `/path/to/nds-build buildrules.txt NSMB.nds`

Replace /path/to/ with the actual file path to nds-build.

You have repackaged your ROM

---

Revision #5

Created 1 March 2024 22:52:08 by Ndymario

Updated 19 February 2025 03:06:34 by Ndymario