-Translation Guide-

by Amko/Amel

This guide is meant to teach anyone interested in translating “The DENPA Men FREE”, which never got an official release outside of Japan, regardless of their knowledge about editing Nintendo 3DS game assets and general IT skills. It’s recommended to have basic knowledge about Windows Explorer. If you already used tdmextractor for game moding with Citra before, it is only recommended to read through chapter 1.1, 2.2, 2.4 and full chapter 3

Some screenshots to explain certain steps may not be in English, in that case relevant buttons which need to be pressed will be highlighted in red, so that it is clear where you need to click.

Supported OS: Windows 10 and 11

WARNING: No liability assumed! Please be careful when following the explanations not to cause any harm to your computer!

Contents

1. Preparations and Basics

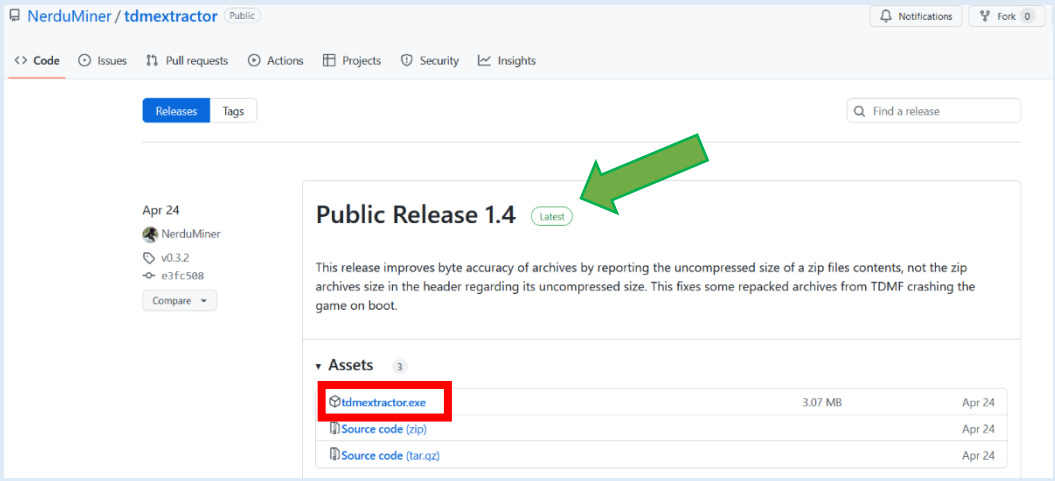
1.1 Required software

To start with in-game translations, certain programs are required (Download link in brackets):

ATTENTION: Please check for yourself if the following links are trustworthy (check Page Certificate) and the downloaded software is secure/malwarefree (by using an Anti-Virus Program). No liability is assumed for any damage caused!

- Tdmextractor (https://github.com/NerduMiner/tdmextractor/releases)
- Tdmfscripteditor (https://github.com/NerduMiner/tdmfscripteditor/releases)

To download either tool on GitHub:



Just click on the exe file and make sure to download the latest build. You will find the downloaded exe files in the download folder (explanations on how to use them appear later!).

- Citra Nightly (https://citra-emu.org/) with TDMF and 1.16 update installed (alternative: Install TDMF and update on real Nintendo 3ds and test out with LayeredFS)
- Text editor
  (Notepad++ is highly recommended: https://notepad-plus-plus.org/downloads/)

Additional (not required) software:
-Hex Editor (e.g. HxD: https://mh-nexus.de/en/hxd/)

-Hex Editor with UTF-16 Bit encoding option
(e.g. Wxmedit: https://sourceforge.net/projects/wxmedit/)

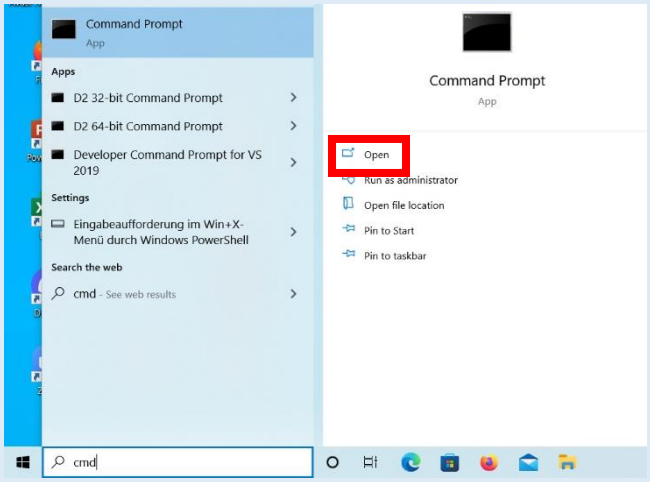
1.2 Basics in cmd

To be able to properly use “tdmextractor” and “tdmfscripteditor”, you should know the basics about “cmd”. “cmd” is part of every Windows OS and allows you to edit files on your computer via command lines and much more (**Attention: you can theoretically delete important files too, so be careful when using it**)!

It’s also used to run .exe files like “tdmextractor” and “tdmfscripteditor”, to open it, follow the next steps:

1. Type “cmd” on your search bar on the left corner of the Display (right to the Windows Symbol

2. And then, open cmd (“Command Prompt”)



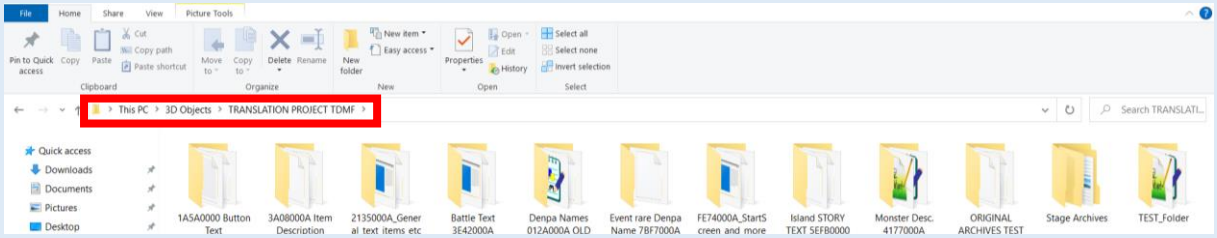
Important command:

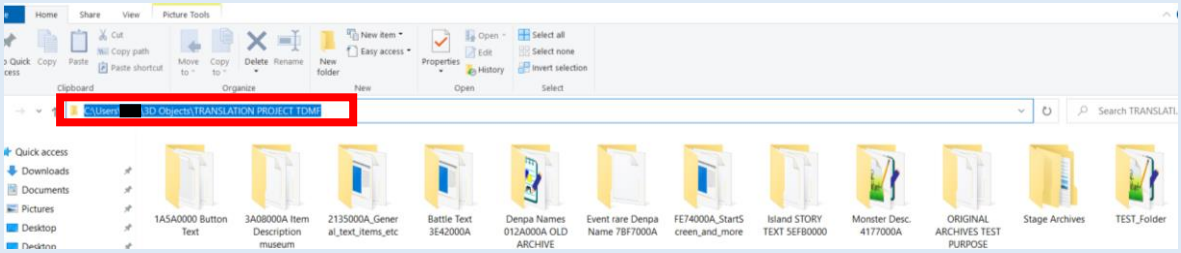
To properly use “cmd” and keep your files sorted, you should use the “cd” command which changes the directory. The directory is basically the file path in which new files will be added/overwritten etc.

When you later use “tdmextractor” or “tdmfscripteditor” it puts the output file in the chosen directory. If you don’t use “cd”, then all the generated files will be put in your very first directory (profile directory in which all basic folders like “Download” and “Programs” are located).

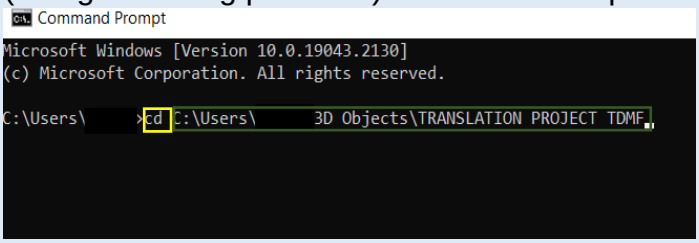
To avoid this, always use “cd” to put new generated files in the folders you want to put.

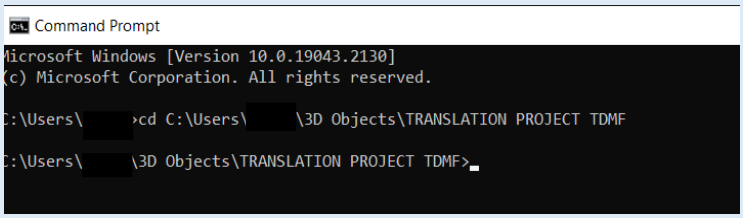
To copy a directory, you can open Windows Explorer. Navigate to the folder where you'd like to put the output files (and which contains the .exe files too!!!) and then copy the following part:





Now, simply copy the directory with "ctrl+c" (or right-click and then "copy"), open cmd, type in "cd", add a space and then paste the copied directory with "ctrl+v" (no right-clicking possible!). Press enter to perform this command.





The directory changed from "user" to the folder "TRANSLATION PROJECT TDMF" (which is in the folder called "3D Objects"). All the commands now will always refer to that folder, to perform .exe programs, the .exe files must be in that folder, and the files that you want to edit must be in that folder, too! Keep in mind, if you close "cmd" and open it afterwards, you'll start in the user directory again.

1.3 The Denpa Men FREE and Citra

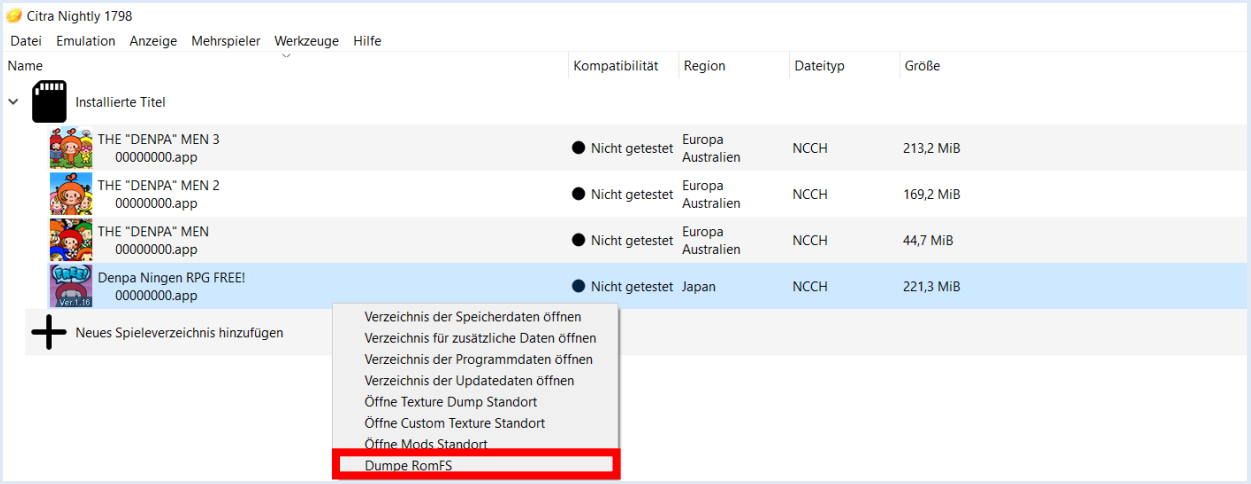
First, if you have not already installed the game and update, look up where you can obtain the .cia file for both the update and the game (be careful about potential malware!), you may have to decrypt those cia files via a real Nintendo 3ds, otherwise you may ask someone else to provide a decrypted cia files for the game + update.

Citra is an emulator for Nintendo 3ds games and enables the modification of those. It is quite important for the translation process as it…

- is a way to test out translation patches in a short time
- has an option to **dump romfs**

Romfs is the part of a Nintendo 3ds game which contains all the assets (models, textures, sound, **text**, animations…) used in-game, especially in-game text which is why this part of the game is vital for the game translation.

In order to dump the game, you need to open Citra (with TDMF + 1.16 update already installed), right-click on the game and choose the last option.

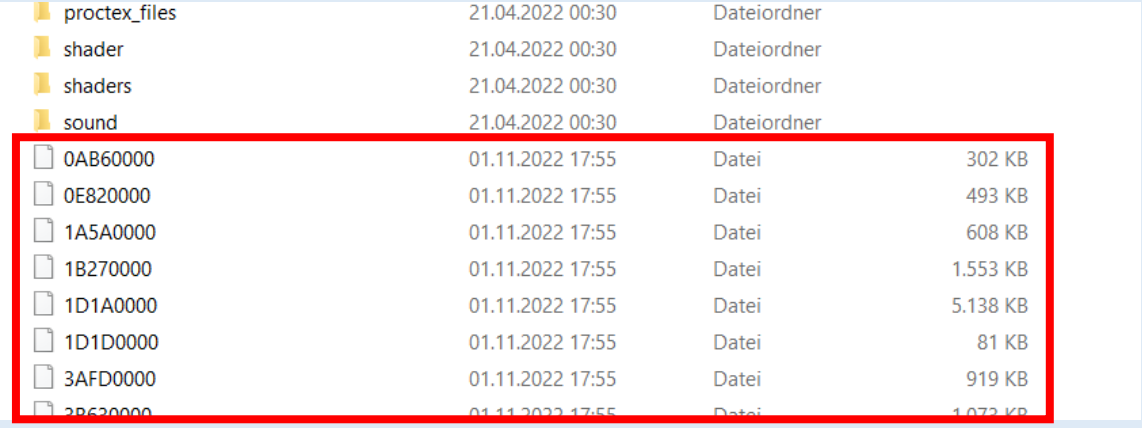


After “dumping” the RomFS two windows should pop up:

- Window with file path containing “0004000000125D00” is the “Dump” from the actual game

- Window with file path containing “0004000E00125D00” is the “Dump” from the update

The update usually just overrides the contents of the game and adds new content to it, thus most of the time only the update “Dump” is relevant.



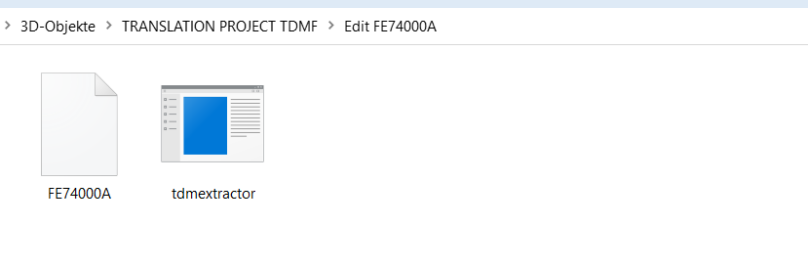
The red marked files in the “Dump” directory are so-called “Archives”. They contain most of the game’s assets, text files too! In order to make use of them, “tdmextractor” will be needed to extract the contents of these archives (there is also a list entailed to this tutorial containing which “Archives” are relevant for text editing and what these archives contain). This will be explained in the next chapter.

You can also dump the game and update via a **homebrewed Nintendo 3ds**, please refer to guides online on how to do that.

2. Modification of text files

2.1 Extracting “Archives”

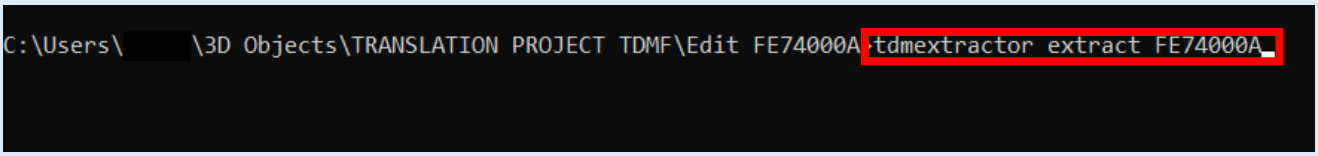
As a preparation for extracting archives, you should copy the archive you want to edit and paste it to a folder in which you specifically want to edit files of that archive. Additionally, you need to paste in “tdmextractor” (the .exe file) in that folder, too! For instance, if you want to edit the “Archive” FE74000A which contains a lot of text files (and is the last “Archive” you find in the “Dump”) the contents of the folder should look like this for now:



Now, in order to “extract” this archive, cmd can be used to perform “tdmextractor”.

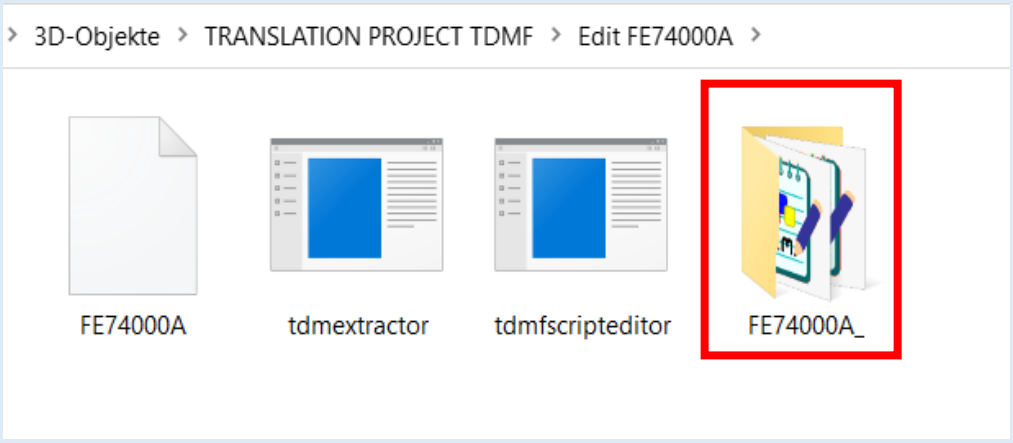
Use the “cd” command in cmd (refer to 1.2 Basics in cmd and “Important command”) to change the directory to the one folder in which the archive and “tdmextractor” can be found and perform following command:

`C:\Users\DIRECTORY>`**`tdmextractor extract FE74000A`**



Then, just press enter.

“tdmextractor” starts extracting all the files from the archive which will generate a lot of text in cmd (so don’t be surprised about that, it just lists which files are extracted and how many). Now, if you look at your folder, the following new folder should appear.



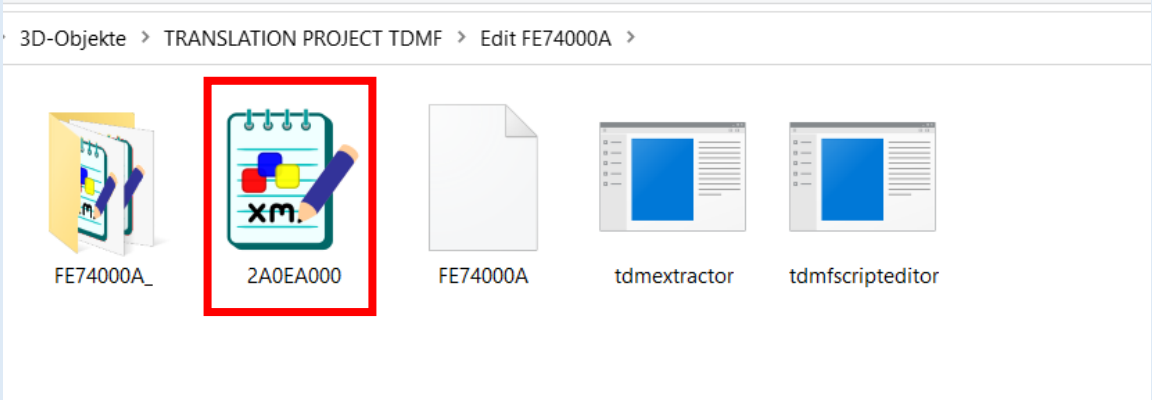
If an error message pops up instead, make sure you correctly typed in the command and used “cd” in the right directory. Also, check whether you have the DMD compiler for “tdmextractor” installed. Additionally, make sure that the name of the .exe “tdmextractor” is literally “tdmextractor” (not “tdmextractor(2)” and so on)!

This new folder contains files, in this case, text files only (besides one .json file which the “tdmextractor” always generates when extracting archives, no need to edit or remove it). These text files always have the file ending “.dat” but not all “.dat” files are text files!

Now these files inside the new folder can be edited in order to translate text or for other purposes (editing textures for instance, but this archive doesn’t contain any).

2.2 Editing text files

In order to properly translate text files, we need to use “tdmfscripteditor”. To prepare the usage of this program, select one of the “.dat” files in the newly generated folder from 2.1 and paste it in the current folder including “tdmfscripteditor”. The folder should look like this now:



In general, it is recommended to create a new folder for each text file to keep all the text sorted, in this case it’s only meant to teach the use of “tdmfscripteditor”.

In this case, the file “2A0EA000.dat” (text for Battle UI) will be edited (the file symbol represents wxmedit, one of the mentioned hex-editors in 1.1, it allows you to view the Japanese text by changing the encoding option in that hex editor to “UTF-16 Bit Little Endian”, this software is not necessary to edit any of the .dat text files).

Now, in order to put this .dat file into a file format that can be easily edited, “tdmfscripteditor” needs to be used. Just like “tdmextractor”, this program can be run with “cmd”, to convert (“extract”) this file into an editable form, perform following command in “cmd” (use “cd” command to be in the right directory!):

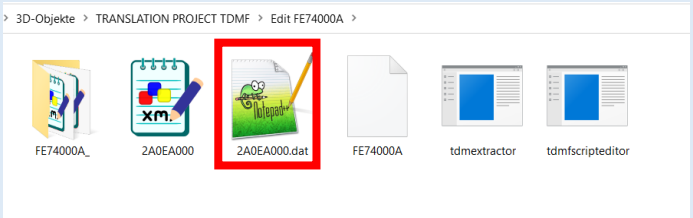
`C:\Users\DIRECTORY>`**`tdmfscripteditor extract 2A0EA000.dat`**

```
C:\Users      cd C:\Users\      3D Objects\TRANSLATION PROJECT TDMF\Edit FE74000A

C:\Users\      D Objects\TRANSLATION PROJECT TDMF\Edit FE74000A>tdmfscripteditor extract 2A0EA000.dat
```

Don’t forget to add the file type extension “.dat” every time you extract a dat file with “tdmfscripteditor”, also make sure the exe program is named “tdmfscripteditor” (and not something like “tdmfscripteditor(2)”).

Now, the following .json file has been created:



<u>WARNING:</u> If you perform this command again, it will overwrite the current .json file, be aware that this may cause any progress to be gone, make sure to backup any of the edited text files.

Those type of files can be edited via various text editors, as mentioned in 1.1, “Notepad++” is one of these that is highly recommended to use for .json files.

After opening the .json file, you’ll notice a certain structure of it. To not cause any issues, only edit the part after "text_contents": “…

All other lines with “hasEntryInOffsetOffsets” and so on, are not relevant for text translation and editing any of those lines will make it impossible for “tdmfscripteditor” to repack those files into .dat files again! It is highly recommended to check if you can repack the file after finishing or pausing editing it and to create a backup after every successful repack.

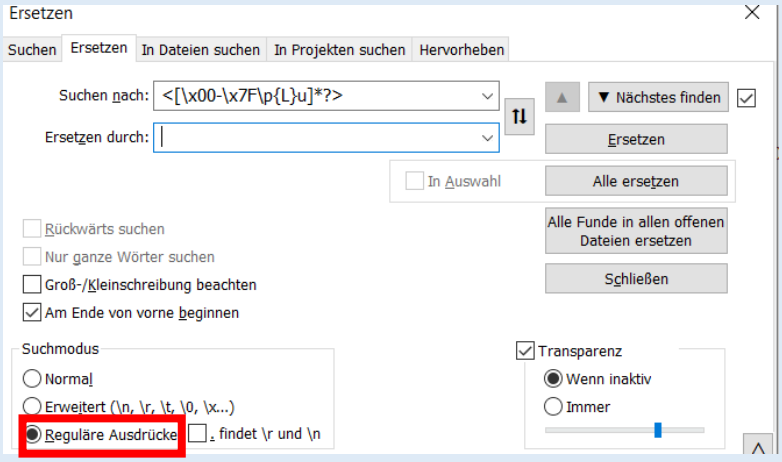
Those functional prefixes such as “<18\u0000>” or “<1ĉ\u0001>{3, name, +}” serve certain purposes which may not all be needed for a translation (purpose will be shown in 3.1). While translating the Japanese text contents via other programs (e.g., DeepL or Google Translate), these prefixes may be annoying, in order to remove them, you can do the following steps (in Notepad++):

1. Create a copy of the .json file (called something like FILENAMETextReference) in which you will remove those prefixes (some of the prefixes may be relevant for translation or are functionally important, thus don’t remove them on the actual .json file you want to repack again!).

2. Open that copy and press “Ctrl+F” for the “Find” option in Notepad++

3. Switch to the “replace” register and paste in the following expression in the search term: <[\x00-\x7F\p{L}u]*?>

4. Keep the “replace with” box empty to effectively delete all the found prefixes and click on “Regular Expressions” below the “search mode”.



After pressing “replace all”, you’ll see that most of the prefixes are gone (very few remain such as “<1ĉ\u0001>{3, name, +}” but these shouldn’t cause any huge translation problems with mentioned software).

“\n” is used to add a line break (like in most programming languages).

\” is used to add quotation marks in the text (simply adding “ won’t work, as these alone are used to mark the start and end of the “text_contents” section).
Example: `\”Special Item\”` will be displayed as “Special item” in-game

Keep in mind that you do this on a copy of the actual .json file as you may need some of those prefixes for functional purposes (such as mentioning item names or npc names)

Repacking a .json file

To repack an edited .json file (to convert it back to a “.dat” file), perform the following code:

C:\Users\DIRECTORY>**tdmfscripteditor repack 2A0EA000.dat.json**

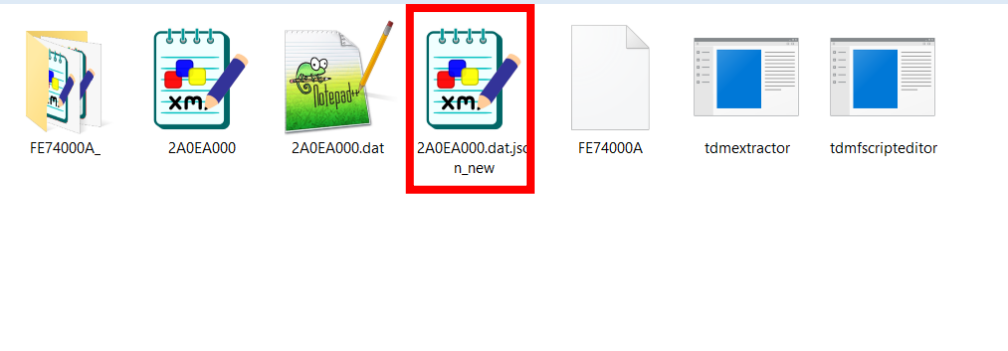
(don’t forget to use “cd” to be in the right directory!!!)

```
ine 59 at offset 3292 parsed.
ine 60 at offset 3410 parsed.
ine 61 at offset 3520 parsed.
ine 62 at offset 3638 parsed.

:\Users\      \3D Objects\TRANSLATION PROJECT TDMF\Edit FE74000A>tdmfscripteditor repack 2A0EA000.dat.json
ine 1 parsed
ine 2 parsed
ine 3 parsed
ine 4 parsed
ine 5 parsed
```

If you improperly edited the .json file, an error message will appear explaining what went wrong (“error exception message”), sometimes it displays the line in which this happened (two line numbers should appear then, don’t interchange it with the exe program line which is in brackets!). But for other errors (such as placing a simple “\” without any use -> causes “invalid string escape sequence”) no lines will be displayed (only the line from the exe program), then you need to search where the error might be. Also, such error messages may appear if you simply forgot to add “.json” to the file you want to repack!

Now, if no error message occurred, the following “.dat” file should be in the folder:



This new .dat file can now be used to replace the old .dat file in the “FE74000A_” folder, by doing this, the text will be replaced stored in that file for this archive (don’t forget to delete the old .dat file in the archive extraction folder and to rename the new one by deleting the “.dat.json_new” part of the new file’s name).

Not every text .dat file will be correctly extracted and repacked with the command “extract” / “repack” in “tdmfscripteditor”, some .dat files have a unique offset format and therefore must be repacked with other commands (such as “extract-old”, “extract-v3” and so on), an additional text file will list those files with their respective commands to be used on them. Also, chapter 3.3 will cover this. In this example, the “extract” command causes no issue.

It is important to note that the tdmfscripteditor may not be able to extract/repack all text relevant .dat files in its current version, this is planned to be fixed when any incompatible .dat file formats are encountered.

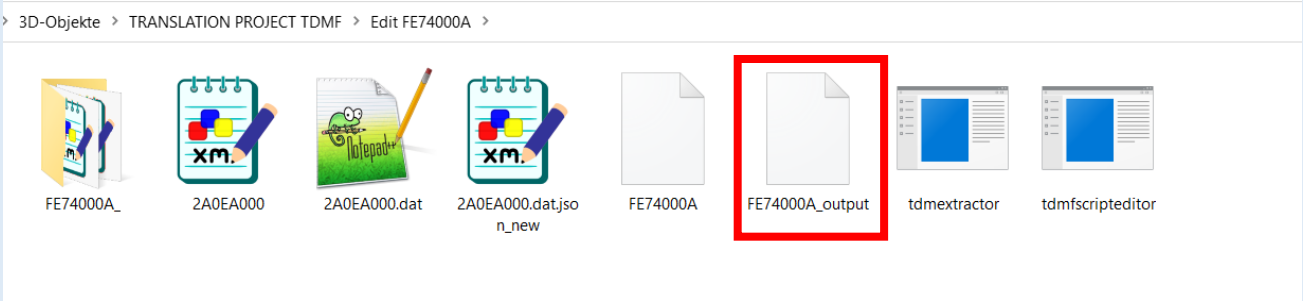
2.3 Repacking “Archives”

In order to test and apply the translated text files, the whole archive must be repacked. To do that, simply perform this command (in this example) in cmd in the right directory:

```
C:\Users\DIRECTORY>tdmextractor repack FE74000A_
```

```
C:\Users\      \3D Objects\TRANSLATION PROJECT TDMF\Edit FE74000A>tdmextractor repack FE74000A_
Repacking Folder...
Archive Version: TDMF
File Index: 1
File ID: 22CF9000
File unk: 11
File extension: dat
Is file compressed?: true
```

Don’t forget the underscore “_” after the archives name (if you didn’t rename the folder). This new archive file should now appear in your folder:



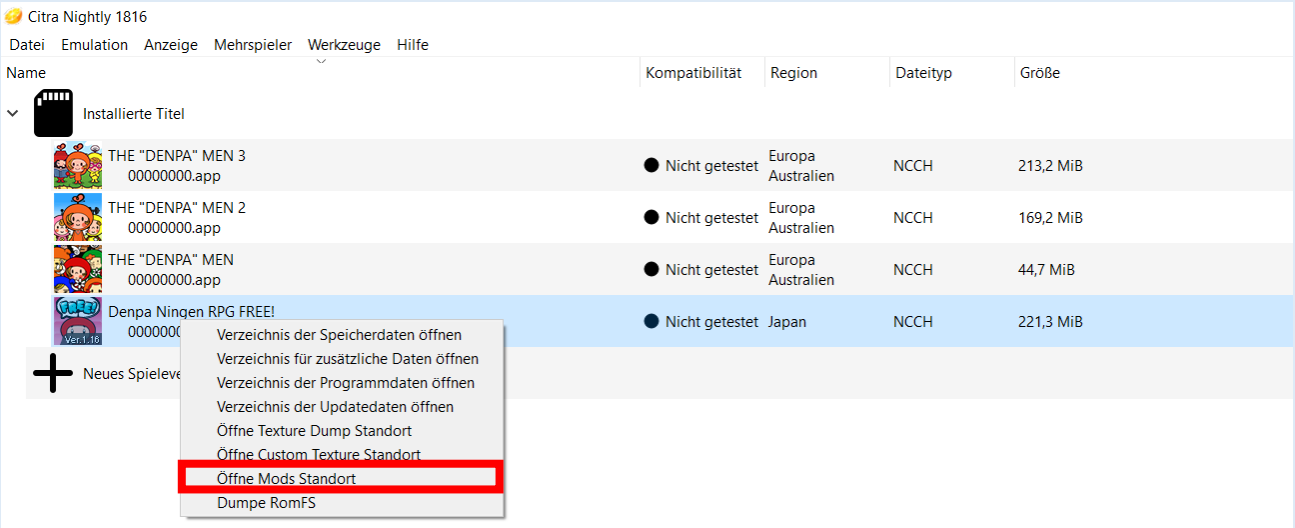
This new archive can now be injected for “LayeredFS” mods in Citra or on a homebrewed 3ds, in the next chapter the method for applying “LayeredFS” on Citra will be demonstrated (guides on how to apply it on a homebrewed 3ds are available online).

2.4 Testing out translation patches on Citra: “LayeredFS”

Preparation:

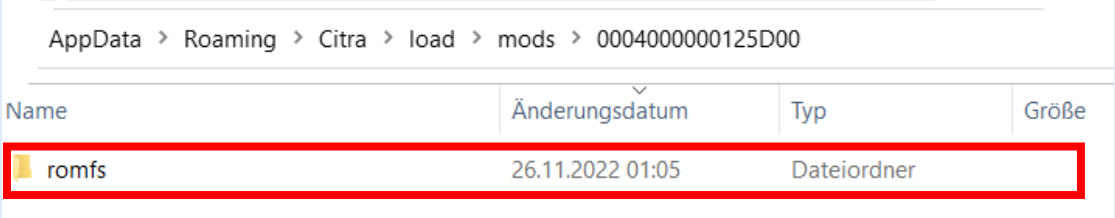
If you haven’t applied mods on TDMF before in Citra via LayeredFS, you need to do the following steps:

1. Open Citra and right click on TDMF



2. Choose “Open Mods Location” to open the file path for mods
3. Windows Explorer will open in the mods directory for TDMF

4. Now simply create a new folder called “romfs” in this directory



Now you can simply paste any modified archive from the game in this folder (for instance, “FE74000A_output” from 2.3). Just make sure that the names of the modified archives are changed back to the original archives’ names (in this case, rename “FE74000A_output” to “FE74000A” after pasting it into the romfs folder), otherwise Citra won’t detect the archive.

3. Advanced translation advices

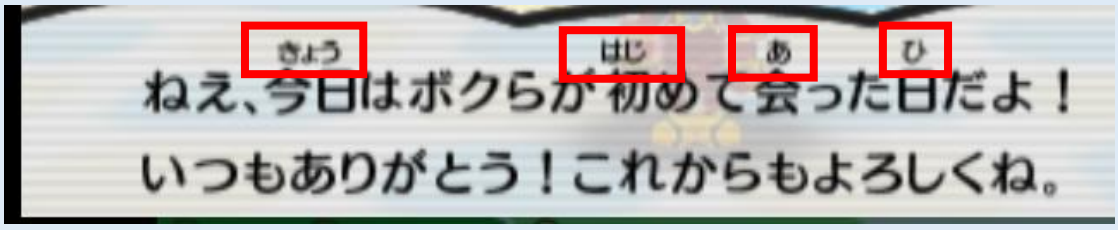
This part is meant to show editing advices which will make it much easier and more efficient to properly translate the game.

3.1 Functional prefixes in text files

As already mentioned in 2.2, in most extracted text files (.json files) various prefixes can be found such as <18\u0000> or <1\"\u0000>. These serve different purposes and may not be necessarily needed for a translation in English. The following ones will be categorized:

Upper text prefixes: <18\u0000> <19\u0000> <1:\u0000>

These prefixes are usually not needed for an English translation, they serve the purpose of indicating when some sort of text in Japanese is put over other text, as in this case:



<18\u0000>: Start of text section which gets “text above it”

<19\u0000>: End of text section which gets “text above it”, Start of text section which is above previous marked text section

<1:\u0000>: End of text section that is above previous marked text section

Example:

```
Welcome to<18\u0000>this v.s. place.<19\u0000>very secret<1:\u0000>
```

Output in game:

```
Welcome to this very secret v.s. place.
```

The upper text will be centered in the marked text section.

As already mentioned, these prefixes are irrelevant for an English translation, therefore can be deleted without fearing any functional problems. But keep in mind, this can be useful if text space is limited in some cases (such as stage place names). Also, for some text files these prefixes won't work (item names for instance), there you simply won't see the upper text.

Denpa Name Placement Prefix: <1Ĉ\u0000> <1ċ\u0001>

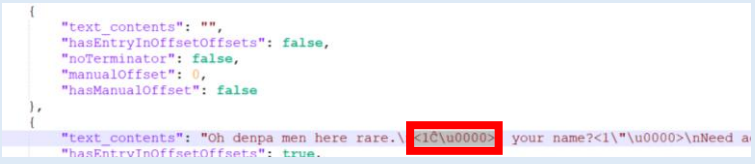
These prefixes are "placeholders" for Denpa names (e.g. the Hero leading the party when talking to NPC which mention the hero's name) but also for enemies during battles ("Enemy took … damage"), the game will automatically insert the right names.

<1Ĉ\u0000>: For a single name (hero name, or when one Denpa Man or monster gets attacked/attacks during a battle for the battle text which is in white).

<1ċ\u0001>: For a group of monsters or Denpa Men (in case of Denpa Men, it will always insert "Denpa Men"/ the name which is inserted in the text file DAFCF000.dat, line 337, Archive 2135000A)

Sometimes, you can add those prefixes in a text spot even though the original file doesn't contain one at that text spot. This can be used to let NPCs say the Hero's/Leader's name if needed.

Example:



(Keep in mind that in this screenshot the Hero's name is in Japanese, hence the Japanese signs)

Text dialogue extension: <1\"\u0000>\n

This prefix is quite useful and important for text dialogue from Denpa Men, NPCs, Monsters and so on. It will add another "speech page" which allows to extend the spoken text as much as needed (no need to limit oneself regarding dialogue

translation). It is important to note that after using the prefix <1\"\u0000> a line break (\n) must be added as well.

If you take a closer look at the example from above, you'll find that the prefix <1\"\u0000>\n was also used in that case. This adds another text page which can be seen in the in-game screenshot as a yellow triangle indicating the player that the NPC still has something to say (prompting to press A). On the antenna tower during catch mode, such text dialogue will usually be played automatically (no prompting the player to press A).

Item/Place/Monster/NPC name references:
<17\u0003>{X, DATA, +}<XYZ>{X, name, Y}

The text files of this game don't always directly mention items, NPC names and so on but instead reference them indirectly with such prefixes (the game then loads the names of other text files in which they can be found e.g., Item Name reference → loads Item Name from file that contains all the item names)

Examples:

<17\u0003>{5, ItemData, +}<2Ç耀>{3, name, +}

→To reference item names

<17\u0003>{8, MessageNpcName, +}<2\u0001耀>{4, message}

→To reference NPC names

<17\u0003>{8, MessageMapName, +}<2\r耀>{4, message}

→To reference place names

<17\u0003>{9, MonsterParameter, +}<2ÿ耀>{3, name, +}

→To reference monster names

Please note that it is not necessary to implement those references yourself, you just need to recognize and make use of them (writing your text in a way that makes sense when the actual text content of the reference is put in).

You may have noticed that all those references contain the "<17\u0003>" prefix, it's important to not delete this prefix, otherwise the game will put out only question

marks and other weird signs, also on a real 3ds console, the game may crash if you forget that prefix!

Other prefixes

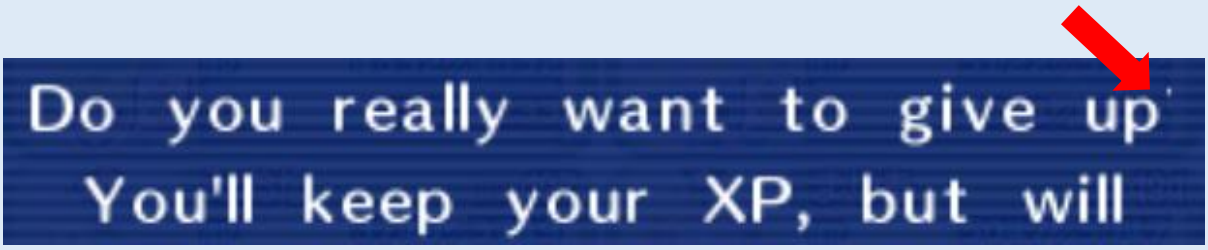
There are also more prefixes that can be found. Most of those though are self-explanatory (e.g., prefix which puts out a number) if you look at the context of the text file. Thus, keep in mind which purpose those prefixes have if they are unknown for you and maybe it is not required at all to edit them.

The only prefixes (and the most common) that should be deleted are <18\u0000>, <19\u0000> and <1:\u0000> as they are only used for upper text sections. Other prefixes serve an important function and therefore shouldn't be removed.

3.2 Text length

When editing a .json text file in Notepad++, you won't know whether the text you write is too long to be displayed in-game if you are not aware of the char limits for a text line. This chapter is meant to demonstrate how long a text line should be for various text spots in-game. If you do not keep the text limit in mind, then the output translation may not be properly displayed in-game.

Example:

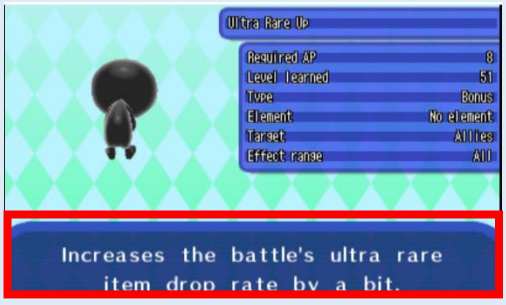


The text for the first line is too long, thus the question mark cannot be properly seen

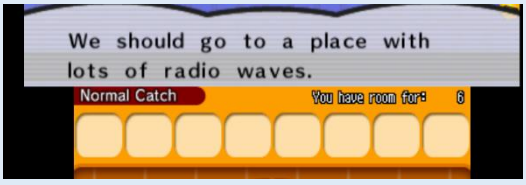
WARNING: The following expressions will refer to a limit of "chars", this also includes spaces! So, keep in mind to count them too, to make sure to not exceed the line limit.

Text dialogue and descriptions

The limit for a line here is **30 chars** (a line can also have less than 30 chars). Descriptions are limited to two lines, while text dialogue can always be extended with <1\"\u0000>\n, so there is no need to limit yourself on text dialogue.



Descriptions

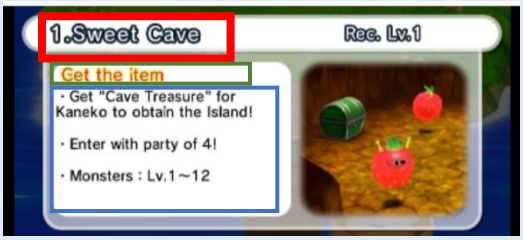


Text dialogue

The same also goes for system messages, such as the screen that appears when you want to save the game or load the save file. In that case though, the number of lines shouldn't exceed **6 (char limit per line is still 30)**.

This text limit also seems to work with battle text (which also has a line limit of 2):

Stage descriptions and titles (Archive: 2135000A File: D3422400.dat)



Stage title (red marked part): After the Stage number (here "1."), **10 chars**

The number of allowed chars is limited, therefore sometimes the stage titles must be abbreviated (e.g., "Evil Hideout" gets abbreviated to "E. Hideout").

Stage goal (green marked part in orange font color): **9 chars**

Stage description (blue marked part): **24 chars per line**, **100 chars in total**

After a line break, another line break should be added so that the description looks clear as in the screenshot above.

Although it looks like there is more space in this screenshot, adding too many chars will result in the overhanging chars not being displayed.

Also, to test out stage description and titles translation, it is necessary to use a new save file. It is assumed that the game's text for stage titles and descriptions gets saved onto the save file, thus you won't notice a change when translating stage descriptions and testing it out on an advanced save file.

Item names (Archive: 2135000A File: 1CABF400.dat)

USE "extract-old" /"repack-old" in tdmfscripteditor for this .dat file!!!

Item names do not seem to have a text limit. But it should be kept below **15 chars**. If the game displays too many chars in one screen (e.g., a lot of translated items in the item list have quite long names) the game will start to glitch out and eventually crash when switching the screen!

Other text

There are no clear indications how long text lines should be for other text, it depends on how much space is available for these (requires in-game testing), particularly for buttons which have a variable size it is not clear how much text they can display.

Regardless of that, most of the in-game text is already covered with the shown types of text.

3.3 Text file location and tdmfscripteditor commands

Text file location

Locations of text files are noted on the attached txt file

(“TDMF Text File LocationsV2.txt”)

This txt file may be updated in the future as it doesn’t contain all translation relevant archives (doesn’t list all stage archives).

tdmfscripteditor commands

As already mentioned in 2.2 some .dat files require certain commands to be properly extracted and repacked. Here is a list of the commands and files:

-extract/repack: for all non-specific .dat files

-extract-old/repack-old: for item names .dat file (Archive: 2135000A File: 1CABF400.dat)

-extract-v2/repack-v2: for antenna .dat file (Archive: 2135000A File: 44D5000.dat)

-extract-v3/repack-v3: for dialogue texts in general (if .dat file contains any text dialogue, use this command instead of “extract”/”repack”)

-extract-v4/repack-v4: Archive: 2135000A File: 2BD17000.dat (EQUIP Effects)

-extract-v5/repack-v5: Archive: 2135000A File: 4C2BD400.dat (Monster Names)

-extract-v6/repack-v6: Archive. 2135000A File: 50BEB000.dat (Shop description)

Make sure to have the most current tdmfscripteditor in order to be able to use all of the mentioned commands properly.

You may also notice that some commands require you to press Enter to finish the extraction process. If the output text in cmd is looking odd, then you may have used the wrong command for extracting (repacking will always look normal). It is incredibly important to always use the right command for extraction and repacking, otherwise translated .json files may not be correctly repacked into functioning .dat files!!!

This list may expand in the future when more commands for tdmfscripteditor will be added to it.

3.4 Further information

- If the game crashes while having applied a translation patch, make sure you used the right commands for extracting and repacking the .json files and that you also correctly repacked the archive

- Some text such as the title screen, certain buttons, text sprites can't be edited via tdmfscripteditor. Instead, other tools must be used (bflim extractors and bflyt editors/hex editors), this only affects small parts of the game

- The Denpa Men's Voice will sometimes still be played (in Japanese, assumption: prerecorded voice) even if the text is translated/changed. Besides, they won't have an English voice when having dialogue translated.

- For collaboration, a cloud link will be provided leading to a possible file structure for the translation which can be downloaded on your computer, this will make it easier for anyone to contribute to translating the game: https://1drv.ms/u/s!Ag4Ze-NRZB9AgQsi1P6NT5pCMZ55?e=ug4vho (some of the TextReference json files are extracted with an outdated version of tdmfscripteditor, they are only meant for text reference to be translated via Google Translate for instance, not meant to be repacked again)

- Check the Discord server of The Denpa Men community to see which text you can translate to contribute to the game translation