THE METALHEAD TRANSLATION FAQ V 0.8 April 27 2000

Hello people, this is MetalHead. In this document I will try to answer the most common, and perhaps some uncommon, questions about translations, emulation in general and especially about translation/hack-patches and how to make (and use) them.

Index

WHY TRANSLATIONS?

Not even in the world of great, fun videogames is everything perfect. Or at least someone might think so. Thus many people seek to change, and sometimes improve over the original, by changing graphics, like zeldamario, changing words in the same language, like many rude or fun dialogues in many a game, or by actually changing the whole language through translating the game.

The most well known, and obviously most popular translations are from Japanese to English, which give the many people out there (like me) with little or no knowledge of Japanese the chance to play great games that for some reason never were released outside of Japan originally. Excellent examples of this are Final Fantasy 2&3 for NES and Seiken Densetsu 3/Secret of Mana 2 for SNES.

On the other hand, we have people living in different countries, understanding different languages all over the world. Not just between Japan and the English-speaking world. Sweden has very good support, at least Nintendo-wise, for being such a small country. Several games were officially released in Swedish, Shadowgate to name but one, but many games, such as Zelda, Willow, and about a million other games with much text in it, were not.

While this may not be all good, it does present an opportunity for those with a bit of ambition, energy and time (and perhaps an ounce of talent as well) to step in and translate. Even if most people in the western world nowadays do understand English, it is always a thrill to play a game in your own language, and there are also a lot of people (e.g. older people, parents or younger children) who may not understand English very well, but of course want to play fun games anyway. Once again, the translations come to the rescue…

Last, but not least, the translators both gain experience in dealing with computers and different languages, and have fun. Well, it isn’t fun all of the time. Actually you tear your hair quite a bit, but it is always thrilling with a challenge, especially one that you have chosen for yourself, and it is indeed fun when things finally work out. Or when someone happens to appreciate your work. Many reasons for translations, but the major one being to play good games, no matter what language you understand.

HOW DO TRANSLATIONS WORK?

Basic info

In answering this, I will need to explain a little basic information on how games, computers and emulators work. At first you have a game, let us say Final Fantasy, that is in Japanese and on an NES cartridge. Someone “dumps” it, meaning he/she makes a file on his/her computer of it, much like copying from a floppy disk, just horribly much more complicated. Great, now we have half a million bytes of incomprehensible data at our hands. But, as fate would have it, there are many smart programmers out there, who make programs that pretend to be an NES, SNES, Sega Master System or whatever machine you may think of. In doing this the program reads the dumped game and lets you play it on your computer, complete with graphics, sound, gameplay and flaws of the original. It is in this step of emulation that the translation comes into the picture.

Translating and altering

If you do not have a cartridge copier, much technical knowledge and time to spend on trial-and-error, there are not many ways to apply a translation to a "real" game on cartridge/disk or whatever. Instead we use the dumped roms, change them on our computers and play them on our computers.

The translation begins with setting up a goal to achieve, and seeing if it is possible. A Japanese-English translator will have a lot of characters to redraw unless there is an English alphabet available in the game already. In the case of English-to-any other language, chances are that you will run out of space when trying to fit in your translated text. There are only so many characters you can redraw without having to lose others. For example, you can put umlaut over an "O" and use zero instead of "O", which works well in many cases. But you don't have any numbers to replace "A" or "C" with. But, if this step is solved, or you can translate without some of the language-specific characters, or perhaps skip using "Q" or something, the translation may begin.

Now a hex-editor comes in handy. Especially one that can read tables and do relative searches, although you can use multiple programs to achieve the same goal, it is often easier with all-in-one. The game programmers seldom use our standard ASCII character-set and number equivalents. Instead they link the characters to numbers that are unused by graphics and instructions in the game, but almost always in an alphabetic order, making it easier to find. This means that if you manage to locate one letter or word and its hexadecimal value, either through relative searching or through viewing tiles in an emulator like Nesticle, you can build yourself an alphabet.

Look at it this way, you find "K" to be equivalent to the hexadecimal value of 44. Then you can almost certainly say that J=43 and L=45 and so on. The easiest way to find this out is by a relative search. Say you know the word "Nemo" appears in the game. Then you can make a relative search for it, and hopefully find it at non-ASCII-numbers, but you still have found it and can carry on. Numbers and other signs are harder to find this way, but can be located in a tile-editor, where one can see at what distance from the letters they are, and calculate their value from this, although this method does not always work, since the game might not access the characters in the order they are drawn in the graphics... but it works often enough. All these numbers and characters are written into a table that the hex-editor reads upon loading the rom, so the numbers in the game turn to understandable text – unless it is compressed or encoded in some way, which is a different problem to solve.

Great, only a few problems left. Number one: lack of space. If a sentence in the game was 24 characters long, there is in most games no way to make it longer that isn't very, very hard, and you have to translate using short words or bad grammar to fit it in. Remember that dots, commas, dashes and even carriage returns count as characters in almost all games. Number two: making a good, understandable translation, with the feel of the game intact, and without accidentally touching the code of the game and screwing it all up. And last but not least, there are a few problems which might either be big or not, but almost always annoying. One is that the game-text might be laid out as a table of words, without any context, that the game accesses to build it's own sentences. This is generally a problem, and also a case in many games with much text – but not all. Since e.g. there might be the words "THE" and "BOTTLE". In English this works fine, but in Swedish there is no such thing as "THE", and whereas "bottle" means "FLASKA", "the bottle" means "FLASKAN", and you can't use one word to make two... But apart from that it's quite easy…

Next, a patch-file. Once you have edited the game enough, you will most likely want to distribute it. But distributing roms is illegal, and even if it was legal, it is rather tedious to send big files back and forth over the internet. An IPS-patch is much easier, since it only describes what is to be altered in the original file, and the IPS-patcher-program then applies these changes in the original file, on the user's computer. Thus you have done nothing illegal, since distributing a list of changes between two versions of a game can hardly be considered anything but legal.

HOW DO I USE A TRANSLATION PATCH?

In order to use this work of others, you need a dumped file of the game that is being translated (no, don't ask me where to get them). Once you have that, you will need the actual translation patch, which, in combination with a patcher-program, will translate your game to the desired language. No matter what operating system you are using, there is bound to be IPS-patchers available. At our homepage we have linked to patchers for mac and windows/dos, others I do not know of, but they should not be hard to find (otherwise you can always make your own). Anyway, you need the original rom-file from which the game is being translated, the patch that describes the changes and a program to apply it. After this (if you know how to apply the patch, usually stated in the program's help or readme-file), you are all set. Well, then. Apply the patch to a backup copy of your rom, I cannot say this enough times – always backup – play the game and have fun! That's it.

Now you have perhaps played a few games in your own language and feel you want to do something like that, you wonder:

WHAT DO I NEED TO TRANSLATE MYSELF?

So, you want to be a translator, eh? Not easy, please to be followink instructions…

First of all, you need a lot of motivation to keep you going during the dark hours (perhaps some rather large amounts of caffeinated beverage too), a game to translate and a general idea of how to actually perform it. Second, you will need the aid of several programs (unless you are superhuman and can read 6502 assembler and interpret it without thinking twice). Some good programs are linked to at our homepage, including Tilelayer and Massacre for editing the graphics of the game, Hexpose and several others for relative searching, creating character-tables, reading the files and then editing them. You will also need an IPS-patcher once you are done, to create the translation-patch, the goal of your work.

Also, you will need some good documents on rom-hacking, which can be found in many places (among others at Zophar's domain, Emulation Camp and Archaic Ruins), to learn from, since this document merely mentions a few of the subjects. Once you have some knowledge – it's trial and error! The only thing I can teach you that you must remember is: make backups! All of the time! If anything goes wrong and you can't remember which bytes you altered, you're done for, and have to redo it all. Thus, follow my advice, please. It will save you very, very much trouble, I promise.

OTHER QUESTIONS ABOUT TRANSLATIONS

What is not yet answered? You may just find it in this section, who knows?

Can I use saves with newer translations?

Many people have played a long way in games, which are partially translated, and keep the save-files. Sometime later they download a new translation, and wonder if they can use their old saves with this "new" game. There should be no problem at all (unless the game-code itself has been altered, which is usually not the case), the game you are playing is still the same. The difference lies in the sometimes altered graphics and the strings of text in the game, which are changed. But as long as the actual code, which tells the game where to use these graphics and text is intact, the save-files should work fine.

Where can I get these roms you are talking about?

No, don't get me started on that. It will just make me angry and you sad. I will not distribute roms in any way or form, unless they are completely legal, and probably not then either. The only thing I will supply is translation patches and utility programs which I know I am allowed to. Go look somewhere else, please.

What is DTE?

DTE stands for Dual Tile Encoding, and it describes the way some text is accessed in many games. It means that one hexadecimal value in the code actually refers to several characters or a whole word, and is a smart way to save space for programmers, but makes translation harder. E.g. in Super Mario RPG, the words "to" "Mario" " in" and a lot of others, are accessed by the game with just one hex-value. It saves space in the rom, but makes a lot of the available hex-values worthless for the translator, since "the" or some other English word might not be used at all in the language translated to, thus making the available space for the translator even smaller.

Most of the good hex-editors out there support this, by typing something like "7B=Mario" in the table-file, the editor knows that it should type a full word at this value.

Also, many hex-editors support an extended variant of this, like in many adventure-games where the hero's/heroine's name is accessed by a sequence of values, like so: "7B55=[hero]", and the editor shows [hero] for you to know that the game will insert the hero's name there when playing, since it is a variable (user-defined when playing) it cannot be stored in the rom from the beginning.

What is script extraction/insertion?

A “script” is defined as part of, or all of, the text in a game, accessed through a table-file. Dumping, or extracting, a script is done by saving this text into a separate file on your computer to allow easier editing in a text-editor. After translation, the text must be put back into the rom, which is where the insertion takes place. This is a commonly used method of translation, but personally I think that with today’s good hex-editors, it is easier to edit the text directly in the program. The main reason why I think this is that it quickly becomes very hard to keep track of exactly how many characters you can use in a sentence when you look at it in a text-editor, mainly since the text moves when it’s length is changed, instead of being fixed byte by byte, and replaced character by character.

DISCLAIMER/GENERAL INFO

I am in no way affiliated with the companies mentioned here (such as Square, Nintendo and Sega), but I do enjoy their work.

This document is written by MetalHead, and may be freely copied/referred to/extracted from as long as the original source (me) is mentioned. Don’t even try to call this work your own, since I will track you down and beat you senseless if you do, or something like that anyway...

If you feel that something is missing here, that you have something to add, or that I should rewrite something – feel free to email me! (kalle@eyecom.se).

I hope that someone finds this document useful, because I wrote it with two intentions: 1 – that it should become useful for those who possess little or no knowledge of the subjects covered, 2 – that it would be a good read for anyone, interested, knowing or just happening upon it. Thank you for taking the time to download and read it.

CREDITS

Thanks to, in no particular order:

- RoboBengt (death_march@hotmail.com) for being a good friend, a source of support and inspiration and for working with me on our translations.
- Zoop and Uzplayer of Emulation Camp (www.emucamp.com) for kindly hosting our site.
- John Stiles (emulation.net) and Richard Bannister (www.bannister.org) for being great guys and excellent programmers/emulator authors/emulator porters.
- The Snes9x team, for – you guessed it – Snes9x – the greatest SNES emulator available. (www.snes9x.com)
- Bloodlust software, for making Nesticle and Genecyst, which I use quite a bit.
- Nintendo, Sega, Konami, Capcom, Square and about a million other companies for their cool games and machines.

- Johannes Holmberg (www.emucamp.com/massacre) for Massacre among other things.
- Dan of Archaic Ruins (archaic-ruins.parodius.com).
- SGC – Snakeeye’s Gaming Corporation (sgc.emu2k.com)
- My family for being what they are.
- All the good heavy-metal bands out there for giving me something nice to listen to.
- Anyone who feels they need a “thank you” and are not included here.

LAST WORDS

-The more people know, the less they will ask. (Generally true when making FAQs).
-It’s nothing that a couple of beers won’t solve. (Me looking at the world).
-I don’t care if they don’t care, as long as I care it’s OK. (Me regarding peoples’ reactions to my projects).
-Do you want a spork or a foon? (Hungover friend of mine asking what I wanted to have breakfast with).

/MetalHead September 1999 through April 2000.