# >Remember

By Hugo Labrande
Issue #11 : Some “demake” ideas

When I was 17, I wrote my first text adventure, which was a hit in the French community, but I still felt like I wasn't very proficient in Inform 6. So over the following 12 months, I ported 2 games by Froggy Software (very popular French Apple II text adventures) to Inform 6, *Même les pommes de terre ont des yeux* (originally a graphical text adventure) and *La femme qui ne supportait pas les ordinateurs*. I'm not sure many people have played them, but I got more experience and confidence in Inform 6 while getting the vague impression that I was helping the community. (And it kind of did: 13 years later, my port of *La femme* was featured in a Polygon video, so that's nice! Here is the link, for those who are interested: https://www.youtube.com/watch?v=vLKZ5lg1qK0 )

What I'm trying to say is: if you are interested in learning how to program a text adventure using a new tool, ports and remakes are extremely useful. You'll get to wrestle with code a little bit without worrying about writing or game design, and you'll get a good idea of how such games are written internally.

Today, I wanted to talk more precisely about “demakes”, because I feel like they have been overlooked in the text adventure genre - and it's a shame, as there are a lot of potentially interesting projects waiting to be unearthed. I've been toying with the idea for a while; but, since I don't have time for it, I am releasing all these thoughts and project ideas in the wild. Maybe one will find a good home?

## What is a demake?

In cinema, some movies are made, then re-made years later, keeping the same story and characters, but changing the visuals, the dialogue, etc. A remake is then expected to be an update of the movie, which is brought to modern times (or, in the case of *Justice League*, 5 years later). A **de**make, then, is when you take a video game, and bring it back to older times - that is, older platforms with fewer processing capabilities and less space, while trying to keep the original spirit and game as intact as possible. Demakes can require ingenuity and technical know-how to push the boundaries of the platform; or sometimes it's just about reimagining the game and keep its most striking elements in a retro package. There is not really a viable market for companies to create demakes of their own games, as there are far fewer retro players; so the demakes of these games are usually made by hobbyists, for the fun of it, for the challenge, or to learn how to code for a platform.

So, here is my pitch: if you'd like some more practice and experience in your text-adventure-writing platform of choice, or if you're looking for a game jam idea, look into demaking text adventures! I'll outline a few ideas here that might be interesting projects. (I would be delighted if you decide to take one up! Please let me know, and I will cheer for you and help you the best I can.)

**Demaking to z3**

Infocom's Z-Machine has a format called version 3, with some constraints on a game's features, including the size of the game, which can be no more than 128kb. This means, realistically, that their z5 games like *Trinity* (which, although not using many of the later features, is 256kb) will never fit in a z3 format. It's not really a big deal: a lot of platforms, from the Commodores to the BBCs, Apple II, Amstrads and all the 16-bit computers, can run z5 games fine. However, not all of them can: there currently isn't a z5 interpreter for the TRS-80 and TRS-CoCo, for the Dragon 64, for a bunch of early CP/M machines (Osborne 1, etc.), for the Oric, for the TI-99/4A, and a few others. This means that demaking an Infocom game into a z3 game would allow such platforms to play one more Infocom game.

That game would most probably be *Nord & Bert*. It is a z4 game of 166kb; all the other z4 and z5 games are larger. Here's my thinking. Looking at the code, it seems like the only time z4 features are used are when displaying the InvisiClues, and of course the larger file size. First, if you take out the InvisiClues, you remove 12kb of space. Then, you can optimize abbreviations, using the few scripts that can find optimal abbreviations; here is a link to mine (which also links to two more, by Matthew Russotto and Henrik Asman, in the first few lines of comment):
https://github.com/hlabrand/retro-scripts/blob/master/abbreviations.py
Judging by our experiments on intfiction.org
(https://intfiction.org/t/highly-optimized-abbreviations-computed-efficiently/48753),
you might save about 4-7kb doing that. Then, further savings might come from code efficiencies: some people have noticed that Inform 6 and PunyInform's code structure is more mature and more streamlined than some of Infocom's code, which could lead to better Z-Machine code. Furthermore, Infocom didn't aggressively use space saving techniques (especially on *Nord & Bert*, as they were 100kb under the z4 limit); there's a bunch of them (just ask me or Fredrik Ramsberg!), and they could save some space too. A back-of-the-envelope calculation (comparing Tristam Island to Infocom games) makes me think that 10-15kb savings could be reached. This means that the game might be 132-140kb in size, which is in the same ballpark as 128kb… Is it possible? Is it interesting? (Do people really want to be able to play *Nord & Bert* on their TRS-80 Model III?) I don't really know, but it would be an interesting project!

There's another possible avenue to explore for demakes: take modern z5 games that are already smaller than 128kb, and try to port them into z3 by removing the z5 features or finding workarounds so you that don't end up using them. The example I have in mind here is *Shade*, by Andrew Plotkin; this great horror short story is around 90kb, and I'm pretty sure the only z5 feature that is used is custom room names in the status line, which could be worked around (by having two identical rooms with the same objects for instance). Or *Hunter, in Darkness*, a chase in a cavern that's a tribute to retro text adventures, and has a file size of 114kb. Plus, the source code for both games are available on Andrew Plotkin's website! It really feels like with just a bit of work, and Stefan Vogt's Puddle Build Tools, *Shade* or *Hunter, in Darkness* could be made playable on dozens of retro platforms (if Andrew Plotkin is ok with it, of course).

If someone wants a challenge, you can also try to shrink games down so they can be played on cassette tape. For C64, you're aiming to get a game to a size smaller than 52kb; this is challenging, but doable, as highlighted by the ZIL or PunyInform version of *Craverly Heights* by Ryan Veeder, which are both around 50kb. It is still rather punitive, and requires having a tight game, but it could be done for some games!

## Demaking to single-side disks

Here, I'm more particularly thinking of the C64: using Ozmoo, you can fit a z5 game of about 160kb on a standard (C1541) single-sided drive, meaning you can play hundreds of Inform 6 adventures. But some of them are larger than that, and will require larger disks and the associated disk readers. What if these games could be shrunk to fit on these readers? This is tricky, as the simplest way to do it requires both to have a game with a file size close to 160kb, and with the source available. I haven't found any; most games I could find with source available, such as Andrew Plotkin's *Spider and Web* (a classic frequently voted one of the best text adventures of all time), or Gareth Rees' *Christminster* are in the 190kb-220kb size, if not larger.

For the more technically proficient, there is in theory another thing that can be done: shrinking games for which the source code is not available. One way to do this requires mastering the z5 file format, managing to parse it, modify it, and still get it to run. But in theory, a game such as *Winter Wonderland*, a 190kb Christmas-themed adventure by Laura Knauth which won IFComp 2000, could be shrunk if one made a tool that extracted the strings from the game file, found optimal abbreviations, recoded these strings using these abbreviations, then very carefully reinjected the strings inside the game so the file still runs fine, but is smaller. (I don't have a good enough idea of the Z-Machine format to know if this would actually work!) Alternatively, you could try to recreate the whole game, and you don't necessarily need to start from scratch: there are disassembly/decompiling tools (Garry Francis tells me that Ztools' infodump.exe and txd.exe are good ones) that can help you extract a lot of data, and you would then just need to stitch it back together, aided by a playthrough perhaps. (This also means you could switch from the standard I6 library to PunyInform, and potentially save dozens of kilobytes.)

This could also open the door to more projects to work around some platforms' limitations: the Atari 8-bit disks can only fit a 124kb game, which means Plundered Hearts had to be distributed on 2 disks; could efficiencies be found such that it could fit on one disk without changing anything in the game?

## Demaking from Inform 7

Inform 7 is not an evolution of Inform 6 as much of a revolution. Contrary to I6, it is a rule-based language, with a syntax meant to emulate the English language; however, the I7 compiler converts this code to Inform 6 code, then uses the regular I6 compiler. The result is powerful, with lots of places for people to plug complicated code in the game's flow; but the generation of I6 code is not efficient at all, going for generality above efficiency. (You can look at the I6 output of an I7 compilation run, and it can be fairly incomprehensible.) In fact, Inform 7 does not support compilation to the Z-Machine any more, as even the most basic game runs past the maximum memory size; it must compile to the Glulx format instead.

This means there is significant potential in rewriting Inform 7 into Inform 6, if one is so inclined. You would need to check first how large the game is and how much text there is, of course; but there are a lot of games written in I7 not because of the in-game features it provides, but because the author liked that language better. Games like *Hoosegow*, *Aotearoa*, and *Hunger Daemon* would probably fit in Z-Machine; whereas games like *Blue Lacuna* or *Counterfeit Monkey* wouldn't.

To give an idea of the size difference, people have recently ported the game *Craverly Heights*, by Ryan Veeder, to various programming languages, as you will see on this intfiction.org thread:

https://intfiction.org/t/craverly-heights-in-inform-6/51780

The original Inform 7 game fits in Z-Machine, and is 256kb big. A Dialog reimplementation is 127kb, while the Inform 6 version is 84kb. If you write it in Inform 6 using PunyInform, it is 49kb; with ZILF (Infocom's internal language), it's 48kb. This 5-fold size reduction means the game fits in z3, and (using the Puddle Tools) could be readily played on several dozens of retro platforms.

And I am sure there are more that could fit in z3, or a small z5! From the filesize alone, I think Arthur DiBianca's *Inside the Facility* would be one; from my memory of playing it, *Hunger Daemon* would be another one! If you are looking for a project to level up in any of the languages mentioned above, there are hundreds of Inform 7 games that could be ported to your favorite retro computer.

## Demaking to The Quill

The Quill, and PAWS, are systems that can handle fewer verbs and custom responses than Infocom's Z-Machine, but this doesn't mean they would not be able to handle complex games - *The Beast of Torrack Moor*, by Linda Doughty, comes to mind, and was written in PAWS, as are Larry Horsfield's games, which are sprawling and complex. The Quill games are smaller, but there are recent small games that could be interesting to The Quill; a particular game that comes to mind is *Darkiss!*, by Marco Vallarino, which straddles the line between horror and over-the-top B-movie in a very entertaining fashion, and whose size is small enough to probably fit in Quill. Another kind of games which might be suited to be Quilled are optimization games, where the game is short but replayable and maximizing the score is a tough act, requiring many tries; so *Captain Verdeterre's Plunder* by Ryan Veeder, and *Sugar Lawn* by Mike Spivey might just squeeze in Quill. Other games could be ported in the roomier PAWS or DAAD systems, in particular ones that have a structure in several parts (as discussed in Issue #10 of this newsletter); the excellent *Chlorophyll* by Steph Cherrywell might be a good fit for that format.

## Demaking to the Scott Adams format

Writing games using Scott Adams' format got easier and more popular in the last few years, using the engine ScottKit. However, you're still faced with very harsh constraints: for instance, the maximum number of messages that can be displayed is capped at 99! There are a few other constraints (and be warned that, in general, the format is less understood than Z-Machine or DAAD), which you can see a recent discussion of in these two intfiction.org threads:

https://intfiction.org/t/scottkit-producing-misplaced-out-of-order-strings/45507
https://intfiction.org/t/scott-adams-interpreter-discrepancies/48864

This means that the text likely needs to be significantly condensed and cut; if you want to make the most out of the limited space, the broken syntax of Admiral Jota's *Lost Pig* might make it a hilarious and great fit. (Garry Francis' *Igor's Quest*, on the other hand, makes such an efficient use of broken syntax that it would probably not need to be condensed!) This kind of demake might thus become an exercise in distilling a game into its core concepts or scenes, which would be a pretty fun deconstruction or reinvention. Can you make a compressed version of *Tristam Island*? How small can you make *Mini Zork*? Personally, I'd be interested in playing such odd ports!

## There’s more!

So much more! The games and the systems that are named here are just the ones I have familiarity with! But there are more systems to pick from; for instance, more modern systems like TADS or ADRIFT, and the games made in these tools, are not playable on retro computers, but they are still text adventures, and they could be “de-made” to retro platforms! And Adventuron is an interesting case here: it is a modern system, with the ability to export to DAAD (and then onto retro computers), but I’m sure that it isn’t possible of all games and some creative programming is sometimes needed to get around these platforms’ constraints. And this article also doesn’t mention the ports to modern systems that sprouted recently, including lots of Speccy games remade in Adventuron, and Larry Horsfield’s remakes of his games into ADRIFT.

Anyway, this article sprung from the idea of remaking games under tighter constraints, and I hope I gave some ideas to a few of you. But even if you’re unlikely to take on such an undertaking, I hope you’ve enjoyed this kind of daydreaming, and maybe this article gave you ideas of fun modern text adventures to play next! And if you can think of any more games that could be ported in the same way, let me know; I’m always interested in talking about that topic!