DAAD
Adventure Writer

Version 2 - Release 4

A multi-machine adventure writing system.

Revised in June 2019

https://github.com/daad-adventure-writer/daad

Target machines
C64, ZX Spectrum, Amstrad CPC, MSX, PCW, Atari ST, Amiga, IBM PC (DOS).

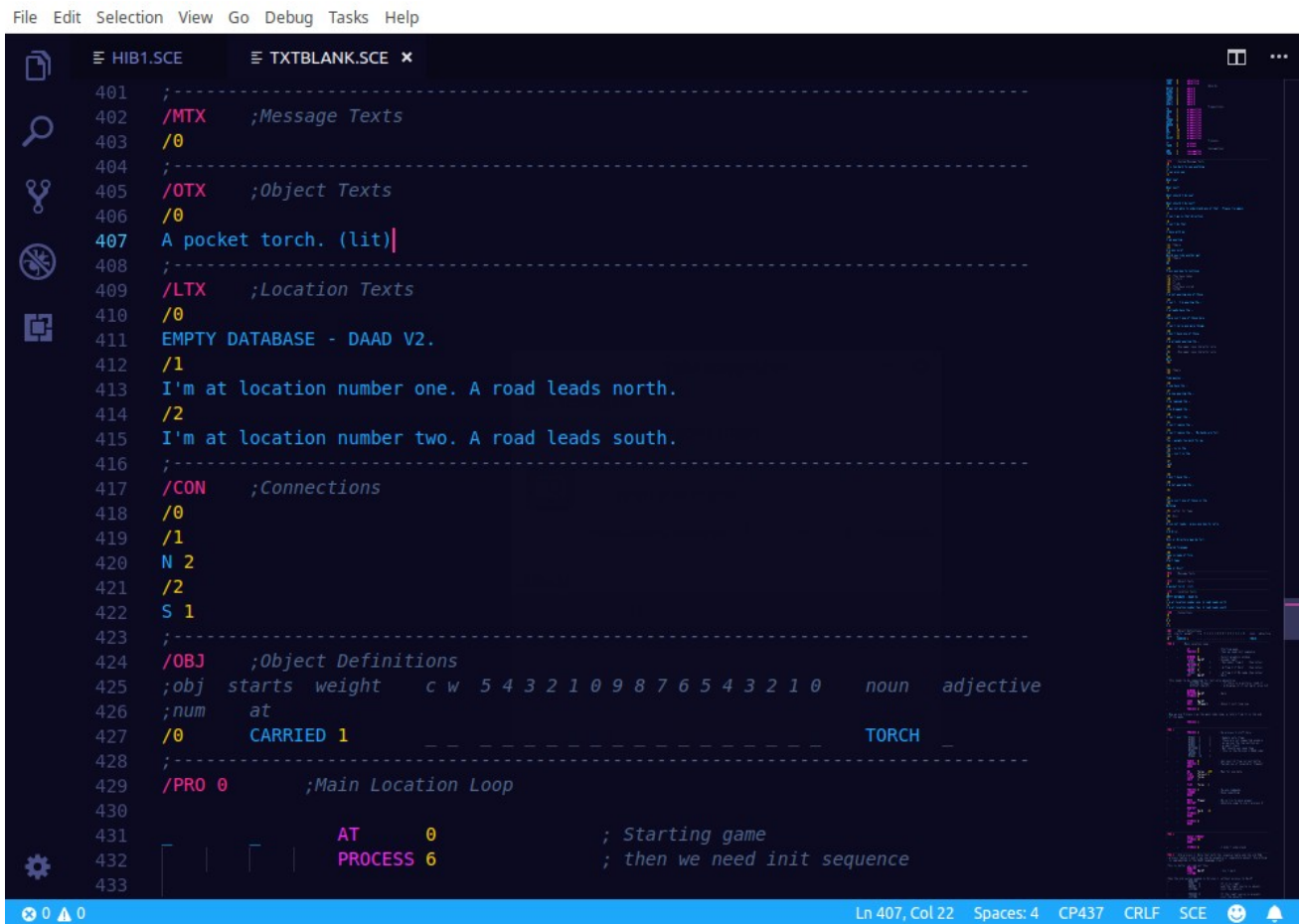## System requirements and recommendations
We wanted to ensure that you can work with DAAD in a (mostly) modern environment. Here are the mandatory system requirements to start developing your own adventures with an <u>unexpanded</u> DAAD system:

- a modern operating system (Linux, MacOS, Windows)
- Visual Studio Code
- DOSBox

To properly setup Visual Studio Code, you should also install the .SCE Syntax Highlighter that Chris Ainsley created for this project. You may download the extension from the VSCode Marketplace: [SCE Syntax Highlighter (DAAD/PAWs)](#).

We highly recommend using it together with the gorgeous "Outrun" theme, that resembles the wonderful colours of the 80s. It conveys the proper retro look and feel. Get Outrun from here: [Outrun theme](#).

This is how a SCE file will look after you applied these changes:

```
File  Edit  Selection  View  Go  Debug  Tasks  Help

     ≡ HIB1.SCE        ≡ TXTBLANK.SCE  ×

401  ;--------------------------------------------------------------
402  /MTX     ;Message Texts
403  /0
404  ;--------------------------------------------------------------
405  /OTX     ;Object Texts
406  /0
407  A pocket torch. (lit)|
408  ;--------------------------------------------------------------
409  /LTX     ;Location Texts
410  /0
411  EMPTY DATABASE - DAAD V2.
412  /1
413  I'm at location number one. A road leads north.
414  /2
415  I'm at location number two. A road leads south.
416  ;--------------------------------------------------------------
417  /CON     ;Connections
418  /0
419  /1
420  N 2
421  /2
422  S 1
423  ;--------------------------------------------------------------
424  /OBJ     ;Object Definitions
425  ;obj  starts  weight   c w  5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0    noun   adjective
426  ;num     at
427  /0      CARRIED 1      _ _  _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _   TORCH   _
428  ;--------------------------------------------------------------
429  /PRO 0      ;Main Location Loop
430
431          _      _      AT      0              ; Starting game
432      |  |   |  |   |   PROCESS 6              ; then we need init sequence
433
```
```
Ln 407, Col 22   Spaces: 4   CP437   CRLF   SCE
⊗ 0 ⚠ 0
```

Note that DAAD .SCE files need to be processed in DOS. To ensure compatibility, we suggest you to open them with the code page 437 encoding. You can override the default settings in user settings to always open .SCE files with CP437 encoding. Please also mount your DAAD directory to DOSBox.

Directory structure DAAD v2 R3 (2019)

We significantly changed the directory structure and the bundled files compared to the incomplete release in Spain a few years ago. The reason behind this was the intention to provide a ready to use distribution.

DAAD   → DAAD root dir where the compiler (DC.EXE) files are located. It's recommended to put the contents of this directory into the root of a DOS drive, D:\ for example

DAAD\TOOLS   → contains all the system utilities in one place, you should add this folder to your PATH variable in DOSBox.

DAAD\TEST   → quick DOS test environment for your adventures

DAAD\TAPEMAST   → directory for mastering tape files (Spectrum, CPC)
DAAD\SCE   → contains all the database template files and examples

DAAD\OBJ   → as referenced in the 1991 documentation, no changes

DAAD\INTERP   → contains all the DOS interpreter files, see 1991 docs

LIB   → interpreters and tools to roll out your games on the supported platforms

DOCS   → the documentation

Image editing – pixel graphics

The DAAD 1991 documentation often refers to the well-known ST program DEGAS for editing pixel graphics and loading screens. While it's fine to still use DEGAS, we suggest you to take a look at some of the modern and convenient solutions. Here are two programs we highly recommend, **Grafx2** as replacement for editing PI1 files (the common ST format), **Multipaint** for editing loading screens for Spectrum, C64, CPC and MSX. It is very easy to bundle a loading screen with the 16-bit versions of your DAAD games (Commodore Amiga, Atari ST and DOS), which you will notice that when you come to the worked example sections. Adding a loading screen to the 8-bit versions is much harder to achieve so we created DAAD loader templates for all 8-bit machines, which also extend DAAD's functionality. The process how to use these 8-bit templates is described in a separate chapter after the worked example section of this documentation.

A worked example in modern times

Please note that the DAAD compiler **DC.EXE** has been updated for the first time since 1990. The new version is 2.42. While the old compiler did only compress location texts, the new compiler also supports compressing messages and system messages, which has a significant impact on the database sizes and thus allows you to create bigger games.

The [-c] switch of the DAAD compiler now has options:
- 0: do not compress
- 1: compress locations
- 2: compress locations and messages
- 3: compress locations, messages and system messages

Example: **dc MYFILE.SCE PART1 -c3 -l0 -m0** would compress LTX + MTX + STX

Note: -**l0** is the English language compiler option. DAAD's default language is Spanish, so make sure to always add language 0 for compiling an English game.

The old compiler version 2.40 remains as **LECACY.EXE** in the DAAD root directory.

ATARI ST
This is one is quite easy. Do exactly as the 1991 documentation says. Don't do the suggested cable transfer though. Use an emulator of choice to get your game files on an empty disk image. **Hatari** is a good solution as it allows mounting directories as TOS hard drives. Don't forget to copy the interpreter files from the DAAD ST disk image: EDI for English and SDI for Spanish adventures.

AMIGA
Compile as described in the 1991 documentation. Don't use any of the transfer programs unless you really want to do it the oldschool way. We highly recommend getting yourself **ADF Opus**, a great explorer and editor for Amiga disk images. It is a Windows application but works fine under Linux with Wine. Make a copy of the MinOS Amiga disk image (in LIB\AMIGA\). Replace PART1.DDB on the image with your own game database. Do the same with the PART1.DAT file if your adventure is not txt-only. If you don't want to add a loading screen you're basically done already. Go ahead like this if you want to add one: instead of creating a loading screen as described in the documentation, create an IFF image with a modern tool like **GIMP**. Use the provided S-Pic utility (see bundled documentation) to create a compressed executable from your image. Add it to your game disk and add the image executable name as an entry in "s/startup-sequence". It needs to be entered before the line that's loading the interpreter. The English interpreter (EDI) had been renamed to INTERP in the Minimum OS template. If you're creating a Spanish game, you need to delete the interpreter that's already on disk and copy SDI from the DAAD Amiga disk to your game disk. Rename it to INTERP and you're done. Copying and renaming can all be done with **ADF Opus**.

PC (DOS)
No changes to the 1991 documentation. We just want to add that you definitely should stick with the method described as "using new system multi-machine graphics". After 28 years it's safe to say that this method will work best for you.

CPC
You can stay close to the 1991 documentation. We recommend using **CPCDiskXP** to transfer your game files to an Amstrad disk. CPCDiskXP is a Windows application but it will run perfectly with Wine under Linux. On Linux we recommend also **Arnold** as the primary Emulator, for Windows there are quite a few emulators available, so pick the one you like the most. In LIB\CPC is a minimal disk that is setup with the essential files ready to compile your game (DAAD_compile.DSK). You should always make a copy of the compile-disk first rather than using the original image. This has the advantage that you can delete

all the files except your game binary from it after the compilation process completed. The disk contains a file GFX.BIN which is an empty graphics database you can use for text-only adventures. Replace it with your own file if you created graphics. DAAD always wants graphic files, it won't work without. Compilation is handled by MCRF which is a CP/M program. So you need to boot into CP/M for the compilation itself. The 1991 documentation says that you may use CP/M 2.2 and CP/M 3 (Plus), which is wrong unfortunately. Only CP/M 3 (Plus) will work. You'll find an image in LIB\CPC. Here is the synopsis:

MCRF oufile{.BIN} interp{.Z80} text{.DDB|.BIN} graphics{.BIN}

Note that you must specify the type of text database. DDB is from the compiler direct (which is recommended) and BIN is assumed to have a CPC disc header.

After you created a native CPC binary, you may use the Windows-Console application 2CDT.EXE (in DAAD\TAPMAST) to also create a tape file. It will work with Wine CMD under Linux. Careful though, as 2CDT.EXE is not a DOS executable. The docs for 2CDT are in the directory. To grab your game file from the Amstrad disk image use CPCDiskXP, or alternatively the console utility iDSK.

ZX Spectrum
This workflow is a bit different to what it was in the past so we completely replace the 1991 documentation with this.

Compile a version of your source without debug information compressed called xxx.DDB (etc) using option -m1. Use ASH to add a Spectrum header to the .DDB file, rename it PARTx.DDB. Make a copy of your DAAD_Spectrum.dsk. Use **WinAPE** to transfer the game files to it. Don't be confused we are using an **CPCDiskXP** again to move files to a Spectrum +3 disk image. Both machines use the same disk system so it will work. Save and close CPCDiskXP after transferring the files. Now open the image in your Spectrum emulator. We recommend **ZEsarUX** or **Fuse**. Make sure you selected a +3 machine for emulation.

ZX Spectrum: preparing the Spectrum +3 release
You are already prepared. We recommend to go with the brand-new DAAD loader template, usage is explained in the next chapter of the documentation. The old loader templates on the DAAD Spectrum disk are still there but deprecated.

ZX Spectrum: preparing the tape release (.TAP file)
You probably want to distribute a TAP file as these can be easily used to create real tapes and may also be used in common interfaces like the divMMC. The steps to achieve this are somewhat different from a +3 release. On the DAAD Spectrum disk image, you find a file called MERGE (and MERGES) for Spanish games. Run it via LOAD "MERGE". It will load the interpreter, PART1.DDB and PART1.SDG into memory. The Basic prompt will reappear after that happened.

Now use the following command to save the memory contents to disk as a single executable: **SAVE "MYFILE" CODE 24576,40960**

Replace MYFILE with your game's name of course. You could also save it PART1 and PART2 in case you created a multi-part adventure.
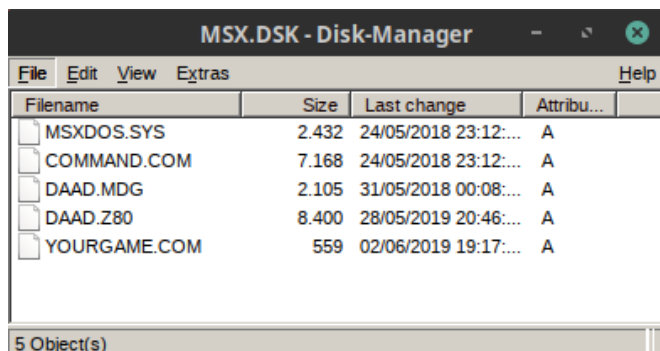
Now you need a tape loader. We recommend to use the advanced DAAD tape loader template. It's properly described in the next chapter of this documentation.

CBM 64
The Commodore 64 was our problem child due to corrupted disks, outdated interpreters, missing headers. We are going to replace the 1991 docs completely with process: Compile a version of your source without debug information compressed called PART1.DDB (etc) using option -m2. Use our shiny new tool ACHTUNG (only Tim can tell you what the acronym stands for and yes, it is one) to add a C64 header to your database. The output format of ACHTUNG is *.DDC. Rename the file to **BPART1** without any suffix. Make a copy of the DAAD_C64.D64 and put the file on it. Use **Droid64** for this purpose. Note there is a similar file on the disk image (APART1), which is the graphics database. You need to leave that on the image, even if you create a text-only adventure as the file contains the charset for your game. If you want to use a different charset, there is one ripped from an Aventuras AD game on disk. Enter the DG editor, select the load charset option, provide the name NEWCHRSET when you're asked for a name. DAAD has also facilities to make you import your graphics from the Spectrum graphics database. There are two interpreters on disk, EDI (English) and SDI (Spanish). We added loaders for English and Spanish games. So you can conveniently rename a loader. LE = Loader English and LS = Loader Spanish. LE1 for example will load an English game BPART1. LS2 will load a Spanish game BPART2. Finally delete from disk what you don't need anymore. This finally brings a sophisticated parser on par with PAWs to the Commodore 64. Enjoy! If you fancy adding a loader image to your adventure, have a look at the next section of this document, where the process is explained in detail. The information you've read here is the base of using the DAAD loader template supporting HiRes and Koala images.

MSX
Note that creating MSX tape releases is deprecated, so we will only cover disk here. You find a template disk in LIB/MSX with the following content:

DAAD.Z80 is the English interpreter (MSXEDI.Z80) on the template disk. Be sure to swap the file with MSXSDI.Z80 for Spanish games. DAAD.MDG is the minimal graphics database containing the charset.

Now compile a version of your source without debug information compressed called PART1.DDB using option -m4. You will need to add Maluva as an EXTERN for making Disk / MSX-DOS support work. More information about Maluva in the bundled README.md file and in the expansions section of this documentation. We suggest a conditional compilation so you won't need a separate source. First add Maluva binary to the /CTL section.

```
13    /CTL     ;Control Section (null char is an underline)
14    #if MSX
15     #extern MLV_MSX.BIN
16    #endif
17    _
18    /TOK     ;Tokens as supplied with PAW under CP/M
19     the
```

Also implement Maluva disk saving and loading like this.

```
2004    #if MSX
2005     SAVE       _       EXTERN 0 1              ;ditto
2006                                RESTART
2007
2008     LOAD       _       EXTERN 0 2
2009                                RESTART
2010    #else
2011     SAVE       _       SAVE    0               ;ditto
2012                                RESTART
2013
2014     LOAD       _       LOAD    0
2015                                RESTART
2016    #endif
```

After compilation, put PART1.DDB on the template disk using the freeware tool MSX Disk-Manager. Rename YOURGAME.COM to the name of your adventure and save the disk image. Start your adventure with the renamed YOURGAME.COM loader. That's it.

The directory at LIB/MSX contains the loader for PART1.DDB and a possible PART2.DDB, as well as both the English and Spanish MSX interpreters.

PCW
We have fully recovered the PCW interpreters but have not found time to test them. Feel free to drop us an email in case you were able to create a PCW release with DAAD.

Using the 8-bit DAAD loader templates for DISK/TAPE

Out of the box, it is not easy to add a loading screen to a DAAD 8-bit adventure. We wanted to ease the process for you and created templates you may now adopt with your own adventure games for that extra bit of professional look and feel. Use **Multipaint** as advised earlier in this documentation for creating your C64, ZX Spectrum, CPC or MSX loading screens and use **Grafx2** for creating your screens for Amiga, Atari ST and DOS. There is also **png2scr.py** which could create a Speccy screen from a given PNG image for you.

Spectrum: TAPE MASTER
Put your Spectrum binary (or binaries) into the TAPEMAST directory. You've created said binaries when following the "preparing a Spectrum tape release" instructions earlier in this documentation. Now put your Spectrum loading screen as **SCREEN.SCR** into the directory. Make sure it has a Spectrum header. Multipaint SCR files for example don't have one but you can add it simply by using the **ASH** tool. Rename your binary, which may be called PART1 for example, to **GAME**. In DOSBox, go to the TAPEMAST directory and enter the following command:

tapcat -ftap sidea.tap -b10 SPECCY.BAS SCREEN.SCR GAME

Repeat these steps with a possible PART2 binary but name the output file **sideb.tap**, so that the already exisiting file won't be overwritten. That's it, your Spectrum tape is mastered and ready for distribution. You can use **tape2wav** from the fuse-emulator-utilitlies in case you also want to create a WAV file.

Spectrum: DISK MASTER
Create a copy of the DAAD_Sloader_Temp.dsk image in LIB\SPECTRUM. Use CPCDiskXP to add your game databases (PART1.DDB, PART2.DDB) on the disk. Also add your loading screen as SCREEN.SCR. Make sure the .SCR file has a Spectrum header as described in the Spectrum: TAPE section. Rename DAADLOAD to the name of your game, SHERLOCK for example. Done.

In case you just have a one part adventure, use the MERGE command from +3 BASIC to modify DAADLOAD so it just loads part 1 instead of prompting which part to load. In that case, the PART2.SDG can be deleted, too. Note the template is made for English games but you can of course edit the template one time and modify the loader so that it loads the Spanish interpreter.

Amstrad CPC: GENERAL

The loaders for CPC are special as they do not just load an adventure with a loading screen. They also patch the game in memory before it gets executed. DAAD on CPC had two issues. You can only save to drive A: (when saving to disk) and a DAAD adventure won't run on a French AZERTY CPC, which is a major problem (the M key can't be pressed). The loader now takes care of the issues. So when using the DAAD CPC Disk loader you can save to any drive you've started the game from and your game will run just fine on a French AZERTY CPC, both from tape and disk. Please note that these loaders very likely only work with the English interpreters. One could delete the machine code patch (pretty much) at the end of the loader code though to make it work. We guess making a Spanish adventure run on a French machine is a pretty uncommon scenario anyway. But just for the loader itself it is definitely possible to adopt it for Spanish games. Just use the MERGE command from CPC BASIC and alter as needed so it fits your game.

Amstrad CPC: CREATING THE .SCR FILE
Note that DAAD uses a special format that appends the palette bytes at the end of the screen data, so normal SCR files won't work unless the palette is not appended. We added the **MARS** utility to tools, which does the job.

This is the recommended way: use Multipaint to create a CPC mode 0 image, save as PNG, you might need a separate image editor like GIMP to index it afterwards. We would recommend **png2crtc** (not bundled with DAAD) to create a .SCR file from your indexed PNG image:

png2crtc MYSCREEN.PNG SCREEN.SCR 7 0

Now dump the palette using **dump-pal.py** found in the TAPEMAST directory:

dump-pal.py MYSCREEN.PNG COLOUR.PAL

Use the **MARS** tool to merge SCREEN.SCR and COLOUR.PAL to **LOADING.SCR**, which is the format and the name the CPC loader needs. Use AAH to add an Amstrad header if needed (can also be done with CPCDiskXP).

Amstrad CPC: TAPE MASTER

Put your LOADING.SCR into the TAPEMAST directory. Also add your game binaries named PART1.BIN and PART2.BIN. Now use 2CDT to create the tape masters, replace GAMENAME with a short equivalent to your game's name, QUEST for example:

PART1:
2cdt -s 1 -n -r GAMENAME DLPART1.BAS side1.cdt
2cdt -s 1 -r DAAD.FNT DAAD.FNT side1.cdt
2cdt -s 1 -F 2 -L 0xc000 -r LOADING.SCR LOADING.SCR side1.cdt
2cdt -s 1 -r PART1.BIN PART1.BIN side1.cdt

PART2:
2cdt -s 1 -n -r GAMENAME DLPART2.BAS side2.cdt
2cdt -s 1 -r DAAD.FNT DAAD.FNT side2.cdt
2cdt -s 1 -F 2 -L 0xc000 -r LOADING.SCR LOADING.SCR side2.cdt
2cdt -s 1 -r PART2.BIN PART2.BIN side2.cdt

That's it. You may once again use tape2wav from the fuse-utilities to create a WAV file if you have the desire to get this on real tape, too.

Amstrad CPC: DISK MASTER
This one is easy. Make a copy of the DAAD_Loader.dsk image in LIB/CPC. Put your game binaries as PART1.BIN and PART2.BIN on it. Add your LOADING.SCR. Done. Use MERGE from CPC Basic to edit "DISC" in case you just have a one part game.

CBM 64: DISK MASTER
The C64 loaders support both HiRes and Multicolour images. HiRes is something you only see from the best artists in the scene, so when you're new to pixel artworks on the Commodore, we suggest to stick with Multicolour first. From Multipaint, save your work as a PNG image, then use a tool like Pixel Polizei or Pixcen to save it to the desired format. Rename your image to **LOADERPIC**. Make a copy of the DAAD_Loader_Temp.d64 image in LIB/C64. Put your LOADERPIC on it. When using DROID64, the PRG suffix is added automatically. Now add your game databases using the naming convention you learned in the worked example section, earlier in this documentation. DLKOALA1 will load a Koala artwork before it launches BPART1 of your adventure. DLKOALA2 will do the same with BPART2.

The DLHIRES1 and DLHIRES2 are the loader equivalents for HiRes images (Art Studio Format). Delete everything from disk you don't need anymore. Rename the remaining loaders to something that represents your game better, e.g. DLKOALA1 to QUEST1 or whatever. Use HiRes2DAAD.py utility in TAPEMAST to prepare the Art Studio image for DAAD (place LOADERPIC in dir and execute). Done.

Note that there is only EDI on the loader disk, the English interpreter. The quickest way to create a Spanish release would be to delete EDI, put SDI instead on disk and then rename it to EDI, since the loaders search for EDI

CBM 64: TAPE MASTER

C64 tape mastering can be quite complex but you find an extensive blog post about it here, which also covers the usage of DAAD's C64 tape loader templates: http://8-bit.info/2019/04/07/daad-c64-tape-mastering/

MSX: DISK MASTER
Since you already followed the steps in the worked example section, there is not much else to do. Just put an MSX1 mode 2 image on disk if you want to show loading screen before your game start. Spectrum screens can be easily converted with Multipaint. Name it LOADPIC.SC2. That's it.

Known issues

Spectrum +3 games can only save to tape

Saving to disk had never been implemented to the DAAD Spectrum interpreters. So the game works, but it will ask you for a tape when you type the SAVE command. Uto is currently adding +3 support to Maluva, so stay tuned.

DAAD's official expansions

Maluva https://github.com/Utodev/MALUVA

An EXTERN which adds pixel graphic support to the 8-bit targets, as an alternative to DAADs own vector graphics editors. It also adds essential, missing features to DAAD, for example loading from and saving to +3 Disc on ZX Spectrum as well as MSX-DOS support.

DRC (DAAD Reborn Compiler) https://github.com/Utodev/DRC

A modern variant of the DAAD compiler which comes with lots of new features as well as an improved syntax. The biggest advantage of DRC is that building your adventures can be automated for most of the target machines. Also has a wonderful tape mastering tool for Spectrum bundled.

Triz2DAAD https://pypi.org/project/triz2DAAD

Command line utility which is priceless for world design and prototyping. It transforms maps created with Trizbort desktop app and Trizbort.io into code which can be compiled straight away with DAAD's DOS compiler as well as DRC.

EAAD https://github.com/Utodev/EAAD

An advanced editor for DAAD sources, supports both the classic .SCE files as well as DRC's new DSF syntax. Also has an integrated code generator for riddles, which makes it very easy to create complex statments of adventure logic.

Credits

Some of the software we bundled with DAAD is not made by us. The project also received amazing contributions by even more amazing individuals. We want to give credit to the authors and advise you to support them in any way you can.

TAPCAT	→ Written by John Elliot as part of TAPTOOLS. You may find other programs in the TAPTOOLS bundle interesting.
2CDT	→ Written by Kevin Thacker.
C64 HiRes Loader	→ Written by Vanja Utne, based on Stefan's Koala loader
CPC Loader code	
DAAD AZERTY patch	
DAAD drive patch	→ All written by Nich Campbell - cpcgamereviews.com
dump-pal.py	
png2scr.py	→ All written by Juan J. Martinez – usebox.net

Legal notes

The binaries of DAAD had been kindly gifted to the public domain by **Andrés Samudio**, the founder of Aventuras AD. It was his company that held the single right to use this software. The sources of DAAD's tools and the DOS compiler are still archived by Tim Gilberts and are not distributed.

The source code of Hibernated 1 – This Place is Death is copyright Stefan Vogt and Pond Software. Read the license details in the header of the SCE file to learn more. Neither the name of the author nor the names of other contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "as is" And any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the author or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services loos of use, data, or profits or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

You are responsible for any legal issues arising from your use of this software.

Final words

It took us many hours, tea and fairy dust to craft this new DAAD release, all for the sake of preservation. DAAD is the final and most sophisticated tool emerging from the code that once started as "The Quill" on a rubber key ZX Spectrum. Now complete again for the first time in nearly 30 years, it never was available to the public and never to an English language audience. DAAD is a significant milestone in text adventure history, a heritage that is reflected by the wonderful "aventuras conversacional" from Aventuras AD. But rather than a "thank you", we want you to use the system. Far too few adventures were written with it and the time couldn't be better to change that. Imagine worlds!

Tim, Stefan and Uto
June 2019