OBEdit

COLLABORATORS			
	TITLE : OBEdit		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 27, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 OBEdit **1**

1.1 OBEdit - an X-COM I and II weapons editor . . . 1
1.2 Oy, what's this for then? . . . 2
1.3 Why bother? . . . 2
1.4 You'll be needing some stuff to use this . . . 3
1.5 Gettin' stuff up an' runnin' . . . 3
1.6 'Ere... how's this work then? . . . 4
1.7 Examples of usage . . . 4
1.8 Viewing binary files . . . 4
1.9 Viewing ASCII files . . . 5
1.10 Exporting ASCII files . . . 6
1.11 Importing ASCII files . . . 6
1.12 Changing stuff . . . 6
1.13 Advanced info. . . . 7
1.14 Using custom RFF headers . . . 8
1.15 About the author . . . 8
1.16 Distribution . . . 9
1.17 Credits . . . 9
1.18 Program development history . . . 9
1.19 Frequently Asked Questions . . . 10
1.20 Index . . . 10

Chapter 1

OBEdit

1.1 OBEdit - an X-COM I and II weapons editor

```
OBEdit 1.2


 Introduction
 - What does it do?

 Motivation
 - Why did I write it?

 Requirements
 - What you need to use it.

 Installation
 - How to install it. (Duh!)

 Usage
 - How to use it.

 Examples
 - Examples (Obviously!)

 Author
 - About the author.

 Distribution
 - Copyright info and such stuff.

 Aknowledgements
 - Credits where credits are due.

 History
 - Development history.

 FAQ
 - Frequently Asked Questions.
```

1.2 Oy, what's this for then?

INTRODUCTION

OBEdit is a handy little proggy for editing an important aspect of MicroProse's excellent games X-COM: UFO Defense (or UFO: Enemy Unknown) and X-COM II: Terror From The Deep - The weapons' performances.

Well, what it actually does is read the file OBDATA.DAT and extract information about the different weapons and other equipment stored there. The info is then saved to either an ASCII standard CSV (Comma Separated Variables) or a TSV (Tab Separated Variables) file with or without an appropriate RFF header.

The CSV format (or ADCH, if you are familiar with FinalData) can be read by most database programs and spreadsheets. This is also true for the TSV format that uses TABs to separate the variables rather than commas. The RFF format is identical to the TSV format, except that it also contains an RFF header that determines how the data will be presented by the program reading it. There is only one program that reads RFF headers, as far as I know, and that is DB 3.5 by David Ekholm.

RFF files are by far the best alternative to be mucking around with. To use them, however, you'll need DB v3.5 or newer. DB is a fully-functional Shareware program that can be D/L'ed from aminet at biz/dbase/db.lha. You could also try to get the newest version from the DB homepage at http://wfmh.man.szczecin.pl/db/

At this point you may ask yourself: "Awright, so I'll have a nice and readable database file with interesting stuff from OBDATA.DAT, but this still doesn't make the game any different. Surely there has to be more?" And the answer is Yes, of course there's more!

OBEdit is not limited to just churning out CSV/TSV and RFF files, it can also read them and apply their data to OBDATA.DAT. The logical editing process will then be: Extract data to CSV or RFF file -> Load CSV or RFF file into appropriate program and make some changes -> Import CSV/TFV/RFF file and apply changes to OBDATA.DAT => Cool!

Additionally (yes, there is even more!) OBEdit will list records to the console window in various ways: Index list, single object info and info of all objects.

Now isn't that just dandy...

1.3 Why bother?

MOTIVATION

The primary reason I made this program was because I wanted to learn how to program in AmigaE. I also wanted to learn more object-oriented programming and file I/O in general. When I decided to undertake this project, I was also becoming increasingly addicted to X-COM: UFO Defense, and in my search for utils for this game I noticed that the only weapons editors were for the PC version. Using AZap and/or FileX I had tracked down the offsets for damage, accuracy and TU usage. Now, using AZap to change these values is very cumbersome and time consuming, so I though I'd write my own X-COM editor, me being a student of computer science and all.

My first major decision was wether to write it in C or AmigaE, or even JAVA. The latter was ruled out rather quickly, since there were no really functioning compilers for AmigaOS at the time. Too bad, really, since I do most of my educational programming in JAVA. Turning to C, I had some basic knowledge of the language and DICE 1.10 that came with an Amiga Format cover-CD. Unfortunaly, DICE was too outdated to do the stuff I wanted to, and upgrading it would be too much of an effort.

Enter AmigaE 3.3: A fully-featured AmigaOS programming environment with good documentation and OS 2.0+ support. And well within my price range too... After skimming through Jason Hulance's beginner's guide and some example source code, I started programming right away.

To find more info about the OBDATA.DAT file, I turned to Robert T. Scott, the creator of XCOMUtil for the PC. He graciously sent me the source code of the part of XCOMUtil that deals with the OBDATA.DAT file, filling out the gaps in my knowledge about it.

Soon after that, the first test version of OBEdit was compiled and successfully executed. At this point it could only read OBDATA.DAT and print out the info it found there to the console. Either as a list of names and indexes, or detailed information about a specific object. These abilities have remained literaly unchanged up to the current version.

Then, my education required more of my attention for a while, and the project was put "on ice" for a couple of months.

Returning to the project, I faced yet another important decision: To GUI or not to GUI! I could make it a command-line program only, implement some kind of console, or study MUI or GadTools. Looking over the beginner's guide for some clue about the use of GadTools, I noticed the string and I/O examples that used the common CSV format to store data. Something went "ding" in the back of my head, and I realized that making OBEdit export and import CSV files that could be edited by dedicated spreadsheet or database programs would save me a shitload of work!

A couple of hours' worth of programming later, OBEdit exported its first comma-separated ASCII file that was readable by FinalData. Taking the idea still further, the cunning implementation of the file's delimiter as an argument to the procline() method and the insertion of an RFF header enabled the program to import/export files that were readable by DB as well. And by yet another stroke of genius, I enabled OBEdit to read this RFF header from an external file, making it possible for the user to have DB display the data as he/she sees fit.

And the rest, as they say, is history ...

1.4 You'll be needing some stuff to use this

```
REQUIREMENTS

The following is required to use OBEdit:

  - An Amiga, or at least a half-decent PC running UAE.
  - OS 2.0 or higher. (Developed under OS 3.0)
  - X-COM I and/or X-COM II (Any versions)

Also, the following is STRONGLY recommended:

  - A database or spreadsheet program capable of reading and writing CSV files
    - or -
  - DB 3.5 or newer.

  - A hard drive.

It is theoretically possible to edit the CSV/TSV/RFF files with a normal
text-editor, but this is only marginally better than to change the OBDATA.DAT
file directly with f.ex. AZap or some other binary editor.
```

1.5 Gettin' stuff up an' runnin'

```
INSTALLATION

Installing OBEdit is quite easy:

  - Copy the main program file to anywhere you want to use it from. The
    easiest would be your c: assign.

  - Copy OBEdit.guide anywhere. Like locale:help or your X-COM drawer.

It is also strongly recommended that you make a backup of your
original OBDATA.DAT file. Under no circumstances should the program destroy the
OBDATA.DAT file, save for the extremely unlikely event that there is a power
failure during the microseconds it takes to write it to your HD. However, you
may one day wish to restore the game's original settings.
```

1.6 'Ere... how's this work then?

USAGE

OBEdit is a command-line only program. There is no GUI, no MUI or anything like that; everything is controlled by the arguments you give it. The arguments are case insensitive and must be in exactly the right order. There's no need to despair, however, as the commands are very logical, and there's really not that many of them. The commands are: "exportcsv", "exportrff", "import" and the optional "all". Here's a list of what they do:

- exportcsv Export a comma-separated ASCII file. The command can be shortened to "c".

-exporttsv Export a tab-separated ASCII file. Short version: T.

- exportrff Export a tab-separated ASCII file with an RFF header. Short version: R.

- import Import either a CSV or RFF file (the program will recognize the format automatically). Short version: I.

- all An optional switch that is always the last argument. When used with exportcsv or exportrff, it will cause ALL 80 records in OBDATA.DAT to be exported rather than just the first 50, which contain the useful stuff. And when used instead of the index number directly after the .DAT file to scan, it will list ALL records in the file to the console.

- verbose Another optional switch that is always the last argument. Used with exportcsv, exportrff or import it causes the program to report its activities to the console window. I.e. it will print the name and index numbers of the weapons it exports or imports. This command can not be used with the all option. Short version: V.

To sum it all up, the syntax is:

obedit <file to scan> <record number/exportcsv/exportrff/import/all> <file to export to/file to import from> <all> <(v)erbose>

At any time during execution you can press CTRL+C to abort whatever OBEdit is doing. This is most useful during file contents listings, as importing and exporting normally takes less than one second unless Verbose mode is active.

NOTE: In the Amiga version of X-Com: UFO Defense the OBDATA.DAT file is kept in two separate directories: GEODATA, where it is read by the GEO program (including the UFOPaedia) and UFOGRAF, where it is read by the TACTICAL program (the action sequences). I'm not sure if this is the case with the PC floppy version, but in the CD version, only the file in GEODATA is read.

Confused? Don't worry, after reading the next bit , everything will (hopefully) seem like a walk in the park.

1.7 Examples of usage

EXAMPLES

Viewing contents of binary .DAT files

Viewing contents of ASCII data files

Exporting ASCII files

Importing ASCII files

1.8 Viewing binary files

obedit

This will only list the commands, in case you've forgotten them.

obedit obdata.dat

This will read the file "obdata.dat" and type out the index numbers and names of the records contained therein to the console.

Result:

*** UFO Weapons editor v1.2 *** File: obdata.dat Size: 4320 bytes. File seems to be of correct lenght.

Index: Name: ---------- 00 PISTOL 01 PISTOL CLIP 02 RIFLE 03 RIFLE CLIP 04 HEAVY CANNON 05 CANNON AP-AMMO 06 CANNON HE-AMMO 07 CANNON I-AMMO 08 AUTO-CANNON 09 AUTO-CANNON AP-AMMO 10 AUTO-CANNON HE-AMMO

obedit obdata.dat 12

Details about record #12 in the file "obdata.dat" will be printed to the console. Records are numbered from 0 to 79 (not from 1 to 80).

Result:

*** UFO Weapons editor v1.2 *** File: obdata.dat Size: 4320 bytes. File seems to be of correct lenght.

Index: 12 Name: ROCKET LAUNCHER Damage: 0 DmgType: 255 Rounds: 10 Ammo 1: 13 Ammo 2: 14 Ammo 3: 15 Acc. Auto: 0 Acc. Snap: 55 Acc. Aimed: 115 TU Auto: 0 TU Snap: 45 TU Aimed: 75

obedit obdata.dat all

This will print out details about ALL the 80 records within OBDATA.DAT to the console.

1.9 Viewing ASCII files

From v1.2 on OBEdit is capable of scanning exported ASCII files in much the same manner as OBDATA.DAT. The ASCII-scanning mode is activated when the supplied #1 file is not 4320 bytes long. The algorithms for viewing ASCII files are the same as the ones used for import, only the results are printed to the console rather than written to OBDATA.DAT. This feature is not all that useful as such, but for all you slightly paranoid people out there it is a reassuring way to check your files' integrities before using that dreaded "import" command.

obedit obdata.csv

This will scan the file "obdata.csv", determine its type and print the records therein to the console. In this case, it is a comma-separated file, but keep in mind that the suffix has no effect on how OBEdit identifies files. Note that line numbering starts from 1 and not 0. Any records that contain errors will be printed to the console along with a helpful error message.

Result:

*** UFO Weapons editor v1.2 *** File: obdata.csv Size: 2555 bytes. - File is not of correct lenght for OBDATA.DAT. Examining... File "obdata.csv" is CSV-ADCH format.

Line Index Name ---------- 2 0 PISTOL 3 1 PISTOL CLIP 4 2 RIFLE 5 3 RIFLE CLIP 6 4 HEAVY CANNON 7 5 CANNON AP-AMMO 8 6 CANNON HE-AMMO 9 7 CANNON I-AMMO 10 8 AUTO-CANNON >--- snip ---< 48 46 MIND PROBE 49 47 >>UNDEFINED << 50 48 >> empty << 51 49 >> empty <<

SUMMARY: ---------- File lenght (lines): 52 Valid objects found: 50 Invalid objects found: 0 _____

obedit obdata.csv all

This will scan the file "obdata.csv" and dump detailed information of all objects therein to the console. This may go on for a very long time if the CSV/TSV/RFF file in question has been exported with the "all" command!

Result:

*** UFO Weapons editor v1.2 *** File: obdata.csv Size: 2555 bytes. - File is not of correct lenght for OBDATA.DAT. Examining... File "obdata.csv" is CSV-ADCH format.

Line 2: 0,PISTOL,0,1,255,255,255,0,60,78,0,18,30,8 Index: 0 Name: PISTOL Damage: 0 DmgType: 255 Rounds: 8 Ammo 1: 1 Ammo 2: 255 Ammo 3: 255 Acc. Auto: 0 Acc. Snap: 60 Acc. Aimed: 78 TU Auto: 0 TU Snap: 18 TU Aimed: 30

Line 3: 1,PISTOL CLIP,26,255,255,255,0,0,0,0,0,0,12 Index: 1 Name: PISTOL CLIP Damage: 26 DmgType: 0 Rounds: 12 Ammo 1: 255 Ammo 2: 255 Ammo 3: 255 Acc. Auto: 0 Acc. Snap: 0 Acc. Aimed: 0 TU Auto: 0 TU Snap: 0 TU Aimed: 0

--- And so on... ---

1.10 Exporting ASCII files

obedit obdata.dat exportcsv obdata.adch

Scan "obdata.dat" and make a comma-separated ASCII file (with header names) called "obdata.adch" that contains data for the first 50 (#0 to 49) records in the data file. This file can be imported and edited by most half-decent database (f.ex. FinalData) or spreadsheet (f.ex. TurboCalc) programs.

obedit obdata.dat exportrff obdata.db

As above, only that the file "obdata.db" will be a tab-separated ASCII file with an RFF header that will make the resulting file readable to DB .

obedit obdata.dat exporttsv obdata.tsv

Very similar to the exportrff command, but the RFF-header will be omitted.

obedit obdata.dat exportcsv obdata.adch all obedit obdata.dat exportrff obdata.db all

When used with either "exportXXX"-statement, the ALL switch will cause the resulting CSV or RFF files to contain ALL 80 records within "obdata.dat". Unless you REALLY want to make some radical changes that I can't possibly think of as useful, there is no reason to ever use this option. All the interesting stuff (as well as a lot of uninteresting), like weapons and ammo is within the first 50 records.

1.11 Importing ASCII files

obedit obdata.dat import obdata.csv

The "import" command is the only one that changes the contents of "obdata.dat". In the above case, the file obdata.csv would be read to memory, and the data it contained would be applied to "obdata.dat". As previously stated, the program will determine wether the file it reads is of RFF or CSV or TSV format itself. If the file is neither, the program will exit, and nothing will be changed in "obdata.dat".

NOTE: The OBDATA.DAT file will always be checked for being exactly 4 320 bytes long. If this is not the case, then the program will assume it is a CSV, TSV or RFF file and scan this as if it was being imported, and a brief summary will printed to the console. If the Verbose option is checked, the program will also list details about the valid objects it finds.

Now that you know how to import and export RFF and CSV/TSV files, let's look at how to edit them .

1.12 Changing stuff

Editing exported CSV, TSV and RFF files

Assuming you have successfully extracted a CSV, TSV or RFF file from OBDATA.DAT and loaded it into a suitable database/spreadsheet program, now there are some things you should know before you change stuff.

Let's start with a list of the different columns you see, and what can/can not be written in them:

Index

This is the object's logical placement in OBDATA.DAT. Do not change this number!

Name

The name of the object. This is for reference only, and will not be imported to OBDATA.DAT.

NOTE: If you use DB, both Index and Name will be unchangable. (Another reason to download it.)

Damage

The amount of damage the ammo will cause. Must be between 0 and 255. Notice that this is applicable for ammo types, grenades and laser weapons only. You should not try to change the damage amount for f.ex. the rifle.

Ammo

Potentially, each weapon in X-COM can use three types of ammo. Ammo1, Ammo2 and Ammo3 are the indexes of three ammo types that can be used. F.ex. the rocket launcher (index 12) will use small, large and incendiary rockets. These are at index 13, 14 and 15, respectively. As a general rule, a weapon's ammo type(s) are on the directly following place(s). The exceptions are gauss weapons in X-COM II, which ammo types are at #47, 48 and 49. A value of 255 indicates that no ammo is usable in that particular slot.

DmgType

This is the damage type of the ammo. Valid types are:

0 - Armour Piercing

1 - Incendiary/Phosporous

2 - High Explosive

3 - Laser Beam/Gauss Beam

4 - Plasma Beam/Sonic Beam

5 - Stun/Freeze

6 - Hand-To-Hand

255 - NONE

Accauto/Accsnap/Accaimed

The accuracy of the different firing modes are expressed in % chance to hit the target. This may exceed 100% and is also influenced by the operator's firing accuracy.

TUAuto/TUSnap/TUAimed

This number is the % of the operator's total Time Units that will be required to fire the weapon. If a value is 0, then the firing mode will not be available. If a value exceeds 100, the operator will obviously never have enough TUs to use that firing mode.

Rounds

This is the amount of rounds contained within one clip of the ammo type. For HTH weapons (stun rod/thermal tazer, vibroblade, thermic lance and heavy thermic lance) this is the amount of damage inflicted to the target.

That just about wraps it up for the things you can edit in the CSV/RFF files. Also notice that you can delete unchanged equipment from the CSV/RFF file to save space (and importing time).

Go to the next section to learn more about the limitations, error handling and structures of the CSV/RFF files.

1.13 Advanced info.

Advanced information about the exported files

A lot of work (well, a couple of hours at least) has been invested to prevent errors in the import files to slip through to OBDATA.DAT. ALL data is examined very closely by the import routines before it is even read into memory. Then, to be completely sure, the values are checked again before they are written to the copy of OBDATA.DAT that is in memory. If everything checks out OK, the OBDATA.DAT file on the disk is finally overwritten.

Let's look at the whole process of importing CSV/RFF files in detail:

First, the import file is read into memory. This is when the first error checking takes place:

- To determine the file's format, OBEdit reads the first line and then searches it for a comma. If a comma is found, the file is regarded a CSV file. If a comma is not found, the program checks if the line contains a tab stop. If this is the case, then the file is regarded an RFF file. If neither a comma or a tab stop is present in the first line, then the file format cannot be determined, and the operation is aborted.

- In the case of CSV files, the first line is ALWAYS skipped, regardless of what it contains.

- In the case of RFF and TSV files, the program will start reading from the first occurence of a linefeed after a "@" is encountered. (The RFF header always starts with a "@".) If an RFF header is not found, the program will skip the first line only.

- In any case, if a line is less than 20 characters long, it will be skipped. This prevents empty lines (two consecutive linefeeds) from being treated as data for object #0.

- If multiple instances of the same object (matching indexes) are found, the LAST one read will be imported.

- If an index number is invalid (less than 0 or higher than 79), the program will inform you about this, and skip the entire line. (No point in reading values that you don't know where to put, is there?)

- If any other data is out of range, the program will skip the object (line in RFF/CSV file) and print it to the console with a helpful error message. Specific data ranges are:

- Damagage: 0 to 255

- Ammo1/2/3: 0 to 79

- DmgType: 0 to 6 or 255

- AccAuto/Snap/Aimed: 0 to 255

- TimeAuto/Snap/Aimed: 0 to 100

- Rounds: 0 to 255

- If a line has an incomplete or superfluous set of data (more or less than 14 comma- or tab separated values) the line will be skipped. In practice, this amounts to checking if there are more or less than 13 commas or tab stops on the line. Two consecutive commas or tab stops will make the value that was expected between them be set to 0. This error is not reported, as it also allows comments within the CSV/RFF files.

Now, there is just one more thing you should know about: the custom RFF header feature.

1.14 Using custom RFF headers

Using custom RFF headers

The procedure for exporting RFF files will look for a file called "OBEdit.hdr" in the S: assign. This file should contain an appropriate RFF header to be inserted in the exported file. If "S:OBEdit.hdr" is not found, an internal default header is used.

The big advantage of this is to let you - the user - export RFF files with the header of your choise. The recommended way to do this:

- Export an RFF file from OBEDIT.DAT

- Load the resulting RFF file into DB and use the "Edit design" menu command to change the appearance of the window to your preference.

- Save the RFF file and open it with a text editor (f.ex. CygnusEd).

- Copy the RFF header (the line that starts with a "@") and make a new file that contains this header and one linefeed. Save this file to "S:obedit.hdr".

1.15 About the author

About myself

Just the basic facts:

On the west coast of an utterly insignificant country called "Norway", there is an even more insignificant city called "Bergen". This is where I live. It rains a lot here, and that's all there is to say about it, really.

Although this program is freeware, I figure there is a one-in-1200120120^1212 chance that somebody will actually send me a postcard, money, inflatable women or anything else. So here's the adress:

Kay Ove Ovesen

Carl Konowsgate 36, L514

5161 LAKSEVAAG

NORWAY

My homepage: http://home.sol.no/~kaovesen

My e-mail adress: ai97koo@stud.hib.no or kayove@hotmail.com

1.16 Distribution

Distribution

OBEdit is FREEWARE. This means that you can copy it, use it and spread it around as long as the original archive is intact. The Copyright remains with me, however. Disassembling and changing the executable file is a complete waste of time, since the source code can be downloaded from Aminet in the dev/e directory anyway. (dev/e/obedit_src.lha)

The only thing I request for this program is that any bugs/errors are reported to me as soon as possible. In other words: complaints are welcome!

Disclaimer:

If this program messes up your hard disk, melts your CPU, blows your fuses, degrades your vision or reduces the quality of your sex life, it is not my problem!

1.17 Credits

Credits

Thanks go to:

Wouter van Oortmerssen for the excellent programming language Amiga E.

Jason Hulance for an excellent beginner's guide to Amiga E.

Scott T. Jones (Author of XcomUtil for the PC) for sharing his knowledge about the OBDATA.DAT file.

MicroProse for X-COM I and II.

The subscribers of the AmigaE mailing list for invaluable advice about E-programming.

Software used:

CygnusEd 4.2 by CygnusSoft Software.

Amiga E v3.3a

DB v3.5 by Martin Ekholm and Marchin Orlowski.

FinalData relase 2 by SoftWood.

TurboCalc 5.01 by Michael Friedrich.

AZap v2.40 by Denis Gounelle

1.18 Program development history

```
HISTORY


v1.0: (15.05.99) Initial relase. Yay!

v1.1: (28.05.99) Minor upgrade.
  - Successfully implemented "$VER:" version string. (Finally got it working!)
  - Improved import-file exception handling. No longer aborts if procline() or
    impvalues() encounter out-of-range data, but rather skips the object
    (line in CSV/RFF file) entirely and prints it to the console.
  - Added the "Verbose" option just for the heck of it.
  - No bugs fixed because: no bugs found.

v1.2: (05.07.99) Minor upgrade.
  - Implemented short versions of most commands.
  - Added the 'exporttsv' argument to export TAB-separated ASCII files without
    RFF headers.
  - Added the ability to scan exported files and report errors encountered in
    them. Not terribly useful as such, since the import methods apply the same
    comprehensive error checking routines. However, it provides a quick way of
    comparing exported files without actually printing them out.
```

1.19 Frequently Asked Questions

F.A.Q.

Q: Your program totally sucks and it messed up my hard drive big-time when I tried to run it! Where can I find your adress, so I can come to your house and punch you in the face?

A: You can find my adress here .

Q: Your program is simply divine! It has elevated X-COM to a whole new level of playability and my life has been pure bliss ever since I downloaded it. Where can I find your adress, so that I can send my 18 year old virgin sister over?

A: Again, you can find my adress here .

Q: What is the answer to the ultimate question about Life, the Universe and Everything?

A: Forty-two.

Q: Why aren't there any real questions in this FAQ?

A: Nobody has asked me anything yet.

1.20 Index

```
                INDEX


                Introduction

                Motivation

                Requirements
```

Installation

Usage

Examples

   Listing Binary files

   Listing ASCII files

   Exporting ASCII files

   Importing ASCII files

Exportfiles editing

Advanced info

Custom RFF headers

About the author

Distribution

Credits

Development history

FAQ