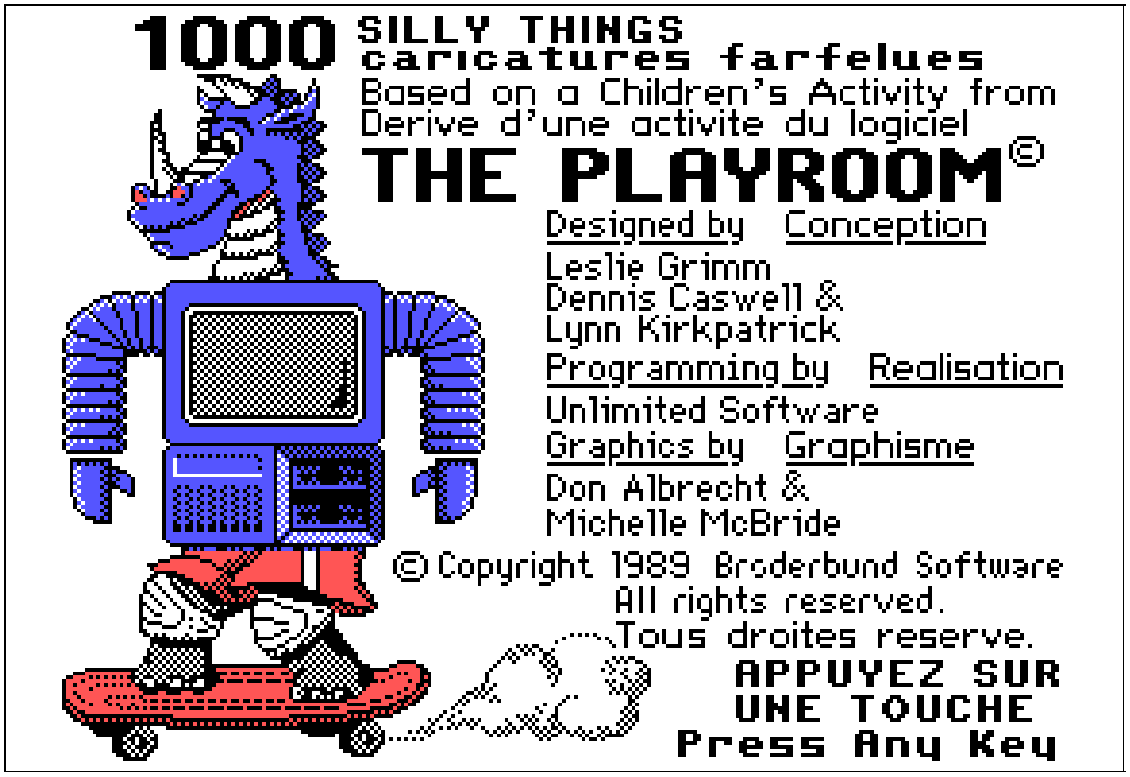
1000 Silly Things aka 1000 caricatures farfelues
ProDOS Version 2.0
By Bill Buckels May 2008

Written in AppleX Manx Aztec C65 Version 3.2b
Windows XP Cross-development environment for Apple //e ProDOS 8

Available for Download at: http://www.clipshop.ca/DiskImages/1000SillyThingsP8.zip

Program Description



1000 SILLY THINGS
caricatures farfelues
Based on a Children's Activity from
Derive d'une activite du logiciel
THE PLAYROOM©
Designed by Conception
Leslie Grimm
Dennis Caswell &
Lynn Kirkpatrick
Programming by Realisation
Unlimited Software
Graphics by Graphisme
Don Albrecht &
Michelle McBride
© Copyright 1989 Broderbund Software
All rights reserved.
Tous droites reserve.
APPUYEZ SUR
UNE TOUCHE
Press Any Key

Children's Play Activity

"1000 Silly Things" is a bilingual play activity for children from 1-100. It runs in either English or French and language can be changed "on the fly" during game play.

This program is based on "The Mixed-Up Toy game" from Broderbund's PlayRoom program. "Three different sections of the body--the head, torso, and legs spin around and its up to the kids to change the body parts to match the cartoon characters…"

It also offers a BSAVE option to save the Silly Things. From the 10 cartoon characters 1000 combinations are possible.

Graphics Demo

From the programming side, this is an Aztec C demo program to show how run length encoded image fragments can be used to save disk space and increase file load speed since the disk files are smaller and load more quickly than a raw file, despite the extra time required to decode the file onto the screen.

Origin of the Graphics

The graphics images were captured from the 1989 MS-DOS CGA 4 color version of The PlayRoom then saved to BSaved IBM images using my ClipShop program. Following that they were subsequently chopped into the required pieces using a modified version of my Fraggle utility, I then modified my FragRAG utility to provide a run length encoded version of my RAG format that I call the RAX format. This format is the same as the RAG format (which is an Apple II HIRES native mode format) with the exception that the graphics image data following the width and height header is encoded as one chunk using the ZSoft PCX encoding algorithm (which is relatively decent and unpacks quickly).

History

This program has a little history attached to it as well. Around 1990 when my son was just a little guy I would give him Apple II programs to take to school.

My kids had The Playroom on their IBM-PC. A ComputerLand Salesman had given me an Okey-Dokey Licenced Copy as a gift at one point. I was purchasing a fair amount of software like compilers around that time.

So I got busy and created "Billy's Silly Things" since like me my son's name is Bill. I did so similarly to what I have done here and wrote it in Aztec C for DOS 3.3, and then finally (again) for ProDOS 8. I long ago lost the original code, and the program.

But with the resurrection of my Aztec C compiler I decided that it would be nice to recreate this program for a graphics demo, and of course for any kids that you might be kicking around.

I first created an Apple II DOS 3.3 Version which is functionally identical to this one. However I was asked to provide a ProDOS version so with a few changes I did the ProDOS version as well. Both versions behave the same way but the ProDOS version has some characteristics of its own.

While creating the French Version, "Billy's Silly Things" became "1000 Silly Things". This happened for several reasons.

From the production side of things I did not want to complicate the programming by asking the child (user) to enter their name to personalize the program, and I did not want the program to become rebranded and redistributed with the last user's name.

I could also see that young children might be stuck with someone else's name simply because it was too complicated to enter their own name or worse yet they or someone else could enter "garbage" or "bad words".

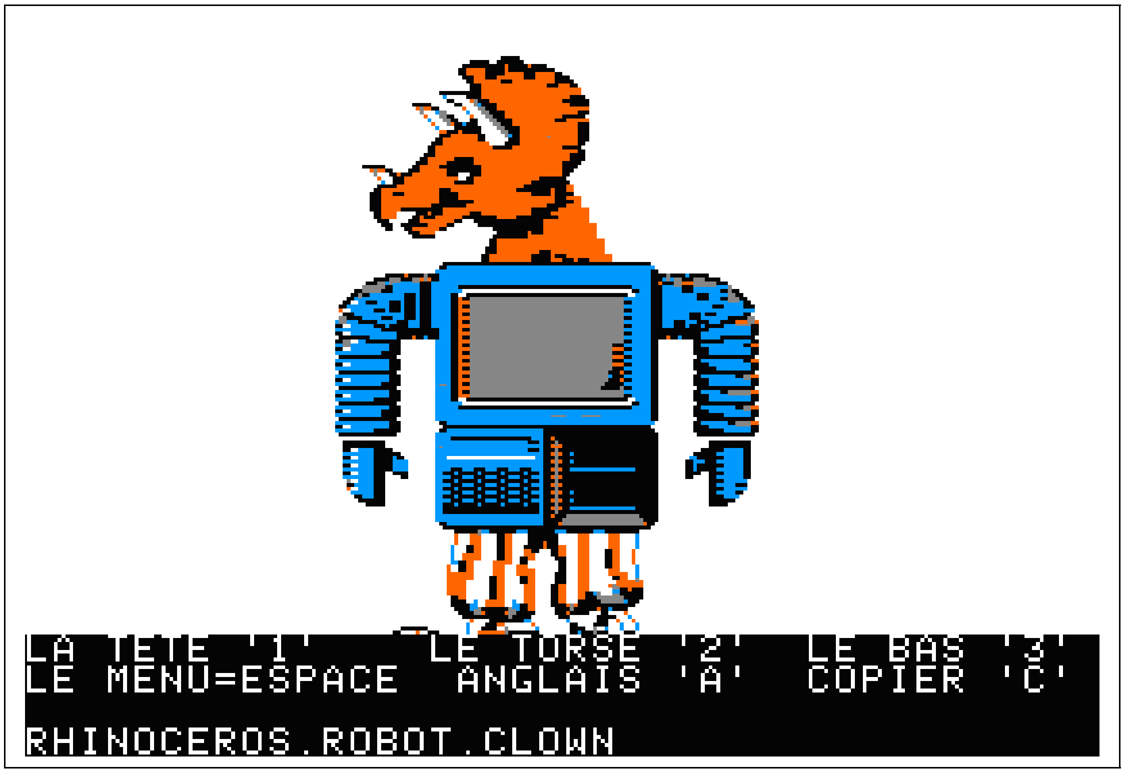
At the same time, a boy's name is not gender inclusive, and my son's name is not even culturally recognizable in most of the world even though English or French may be understood.

The third thing that I did not want was to make the child feel that the program is for someone else (Billy) besides themselves, like an older brother called "Billy" or worse yet, a school-yard bully called "Billy". Even worse yet could be a bad "Uncle Billy" or a bad priest called "Father Billy".

Well you get the point I am sure, so with some reluctance but a realization that my son is no longer a "little guy" and we still have our memories of our history together, and given my other reasons, I changed "Billy's Silly Things" to "1000 Silly Things".

I am sure that both you and Bill can understand why.

Program Details



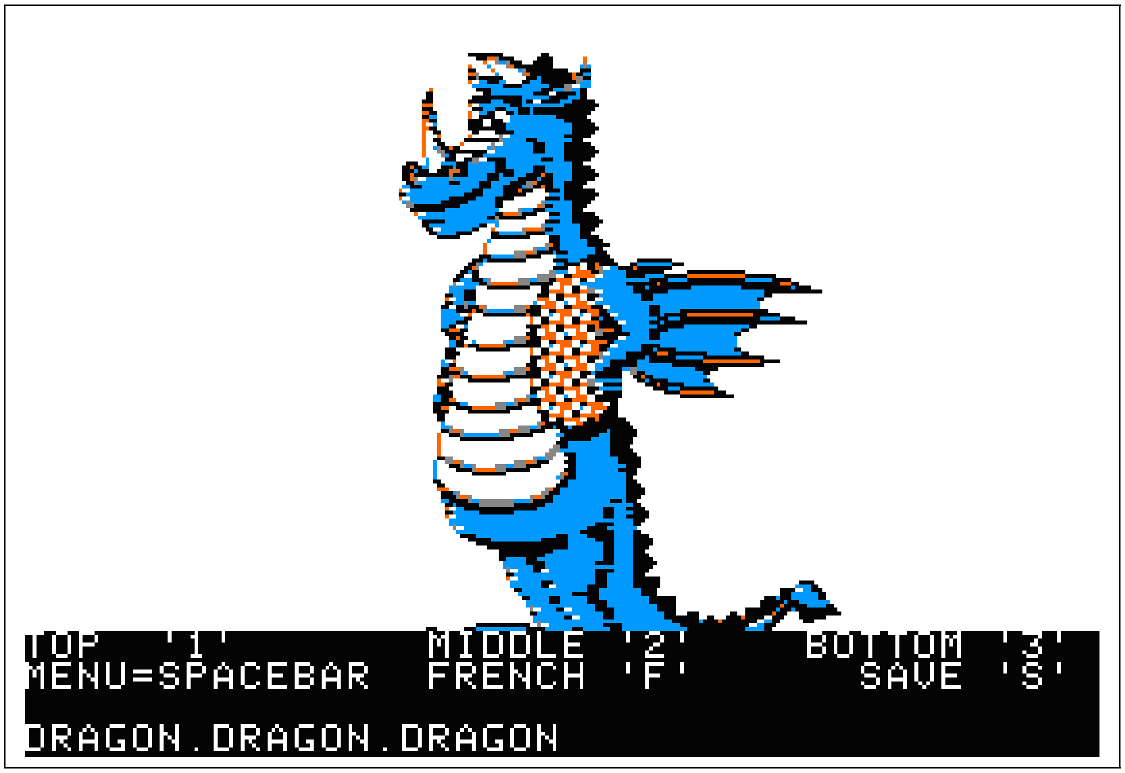
```
LA TETE '1'     LE TORSE '2'   LE BAS '3'
LE MENU=ESPACE   ANGLAIS 'A'   COPIER 'C'

RHINOCEROS.ROBOT.CLOWN
```

Silly Things runs in standard Apple II HIRES 280 x 192 x 6 colour Mixed 40 column Text and Graphics Mode.

It is not necessary for your child to read in order to use Silly Things. The Menu Commands are very straight forward.

Note also that Silly Things remembers the last language used and will start-up the next time using that language. English and French are toggled during game play by pressing the 'E' or the 'F' key respectively

Getting Started



Commands and Navigation

The arrow keys and number keys 1,2,3 are used interchangeably toggle the next Silly body part.

1 or T or [UPARROW] - Get Next Silly Top
2 or M or [LEFTARROW] or [RIGHTARROW] - Get Next Silly Middle
3 or B or [DOWNARROW] - Get Next Silly Bottom

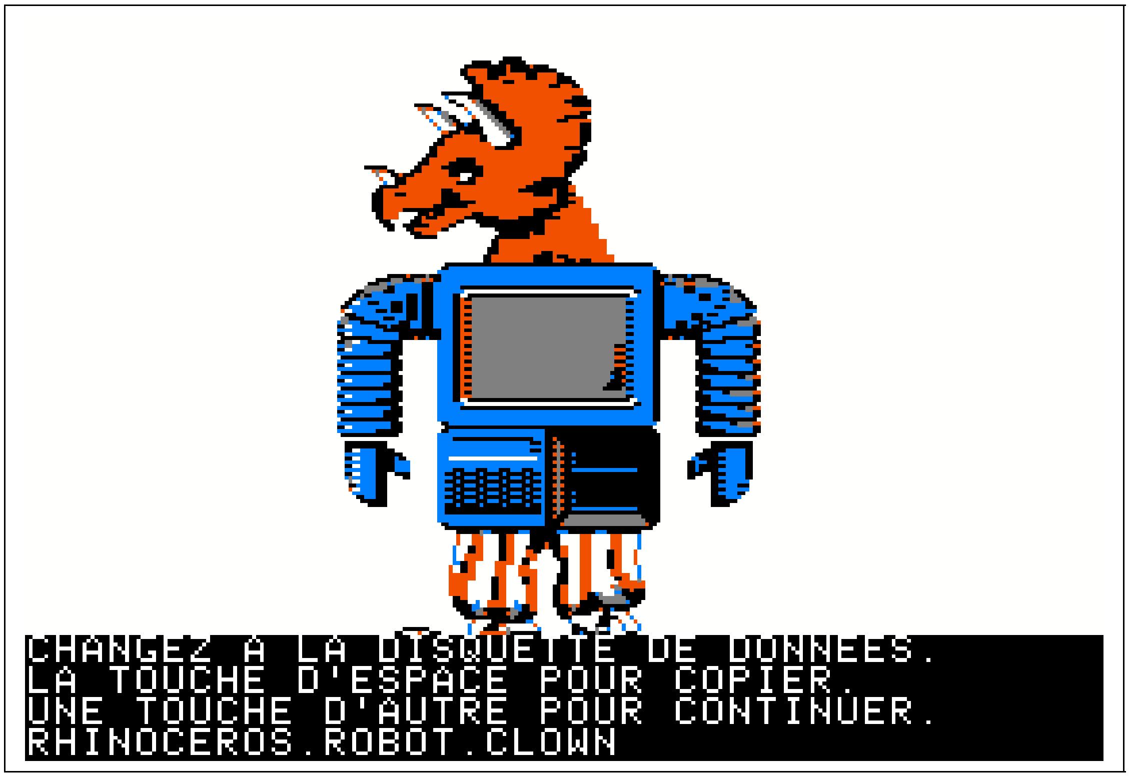
[RETURN] or [SPACEBAR] - toggle between full-screen mode and mixed-screen mode (displays the menu at the bottom of the screen).

F - Run program in French. 1000 caricatures farfelues.
E or A - Run program in English (Anglais). 1000 Silly Things.
S or C - Save Silly Thing

[ESC] - Exit

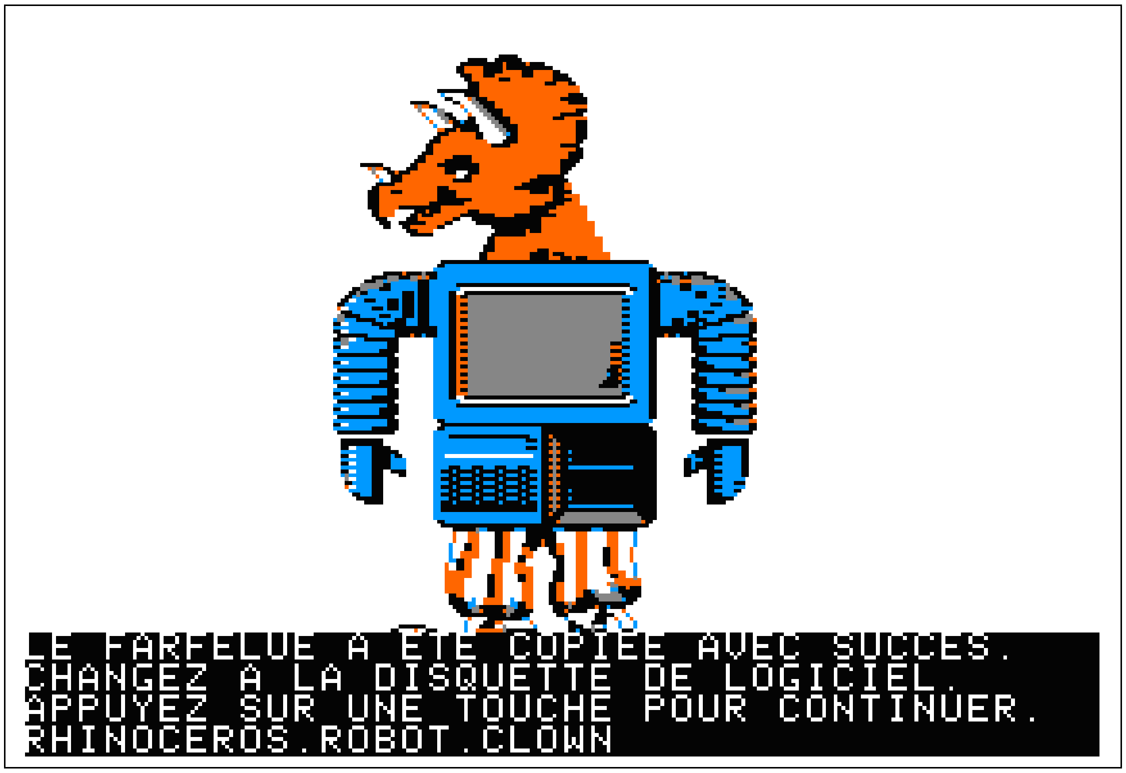
A mouse is not used.

Saving



```
CHANGEZ A LA DISQUETTE DE DONNEES.
LA TOUCHE D'ESPACE POUR COPIER.
UNE TOUCHE D'AUTRE POUR CONTINUER.
RHINOCEROS.ROBOT.CLOWN
```

Silly Things are saved using an automatic naming based on the three body parts. If the head is from a ROBOT and the middle is from an ELEPHANT and the bottom is from a DRAGON, the resulting BSaved image file will be called ROBOTELEPHDRAGO (5 characters from each Silly Thing's Name).



```
LE FARFELUE A ETE COPIEE AVEC SUCCES.
CHANGEZ A LA DISQUETTE DE LOGICIEL.
APPUYEZ SUR UNE TOUCHE POUR CONTINUER.
RHINOCEROS.ROBOT.CLOWN
```

During saving you will be given the opportunity to put a Data Disk into the drive and then you will be prompted to press [SPACEBAR] to save. The [RETURN] key and the [SPACEBAR] can be used interchangeably just as they are during game play.

See the section below on Automatic File Naming and Data Disks for more details about saving.

Differences between DOS 3.3 and ProDOS 8 Versions

In DOS 3.3 "1000 Silly Things" is called SILLY.PRG and is BRUN from the HELLO program on Side A of the disk or from DOS 3.3. On Side B of the DOS 3.3 disk, the BSaved Image Viewer, ABINLOAD.PRG, is BRUN from the HELLO program or from DOS 3.3.

In ProDOS 8 "1000 Silly Things" is called SILLY.SYSTEM and runs on startup on side A of the ProDOS disk. ABINLOAD.SYSTEM runs on startup on side B of the ProDOS disk, or either SYS program can be run directly from ProDOS.

In either DOS 3.3 or ProDOS you do not need the ABINLOAD BSaved Image viewer to view the BSAVED images that "1000 Silly Things" creates. It is provided as a convenience. You can write your own in BASIC or view these in any program that accepts a standard Apple II HIRES BSaved Image.

The ProDOS version of the ABINLOAD BSaved Image viewer wants a text file called "PICLIST" for the BSAVED images that are created by SILLY.SYSTEM. So when SILLY.SYSTEM saves a BSaved Image it appends the image name to the PICLIST on the SILLYPROGRAM data disk. If the BSaved Image is already in the PICLIST a duplicate image name will not be added. If the PICLIST does not exist it will be created. If a BSaved Image file is on the disk but not listed in the PICLIST, and the filename is entered in ABINLOAD it will still be loaded,

The reason that ABINLOAD in DOS 3.3 doesn't want a PICLIST is because it calls the DOS 3.3 Catalog routine directly to list disk contents. Other than that ABINLOAD.PRG in DOS 3.3 is the same program as ABINLOAD.SYSTEM in ProDOS 8.

Since ProDOS has a 15 character file name rather than the 30 character filename in DOS 3.3 I have limited the automatic naming that is used by SILLY.SYSTEM to 15 characters when saving.

On the DOS 3.3 version, if a silly thing was called DRAGON.CARROT.ROBOT the ProDOS equivalent is DRAGOCARROROBOT which is descriptive enough considering the differences between the two systems. This is one difference between the two versions that was unavoidable.

Automatic File Naming

The filename is always in English regardless of language. In the case of recognizing the saved French files afterwards:

DRAGON=DRAGON
ROBOT=ROBOT
CAROTTE=CARROT - CAROTTE will gain a second R.
RHINOCEROS=RHINO - truncates to RHINO
ELEPHANT=ELEPHANT
CLOWN=CLOWN
FEE=FAIRY - unrecognizable
SOURIS=MOUSE - unrecognizable
POISSON=FISH - unrecognizable
LICORNE=UNICORN - somewhat recognizable

While this compromises the French in 60% of the set this is a programming consideration to avoid a number of potential problems and to make programming easier:

1. The initial compressed image fragments require a relatively large amount of disk space despite the fact that they are compressed. It is not practical to add 60% more image fragment files (additional duplicates) that are named in French so that our loader uses a differently named file. It serves no purpose except that a French person looking at the disk would see the extra files in French. They would still see the English files, so doing this would serve no real purpose and also would add unnecessary code to the program without improving the user experience.

2. Adding 6 files would increase the number of combinations from 10 x 10 x 10 = 1000 to 16 x 16 x 16 = 4096 and would also result in potential duplicates of 3096 files if the program was flipped between English and French. And "4096 Silly Things" does not have the same "ring" to a child's ear as "1000 Silly Things".

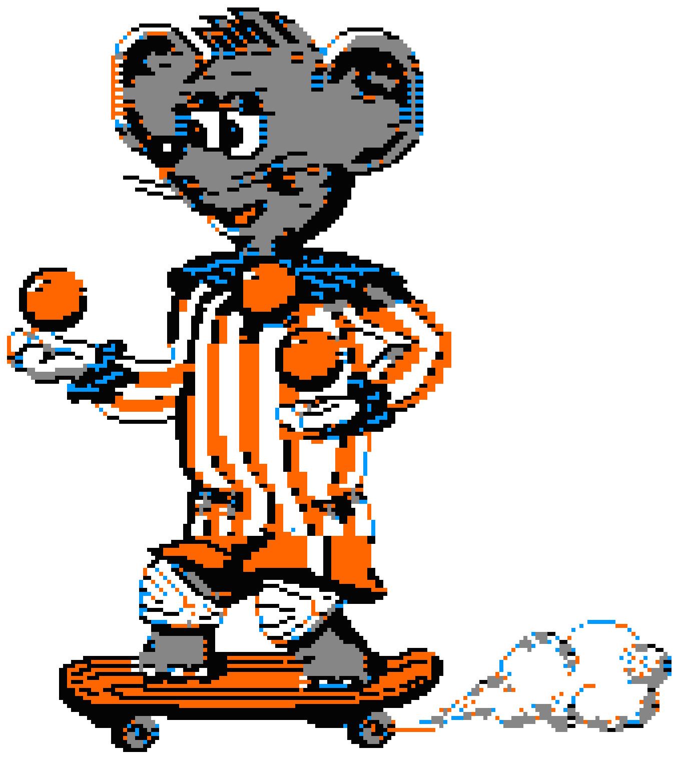
A potential elegant workaround for most of this would be to rename the initial compressed files "on the fly" every time the language is changed. However there are a couple of reasons why this would not be a good idea:

1. If a program was flipped back and forth between English and French duplicate output files could still occur. The potential for this happening can't be ignored and since the user experience would not be improved because the filename never appears on screen anyway. If the same program is shared between users this might be more likely to happen.

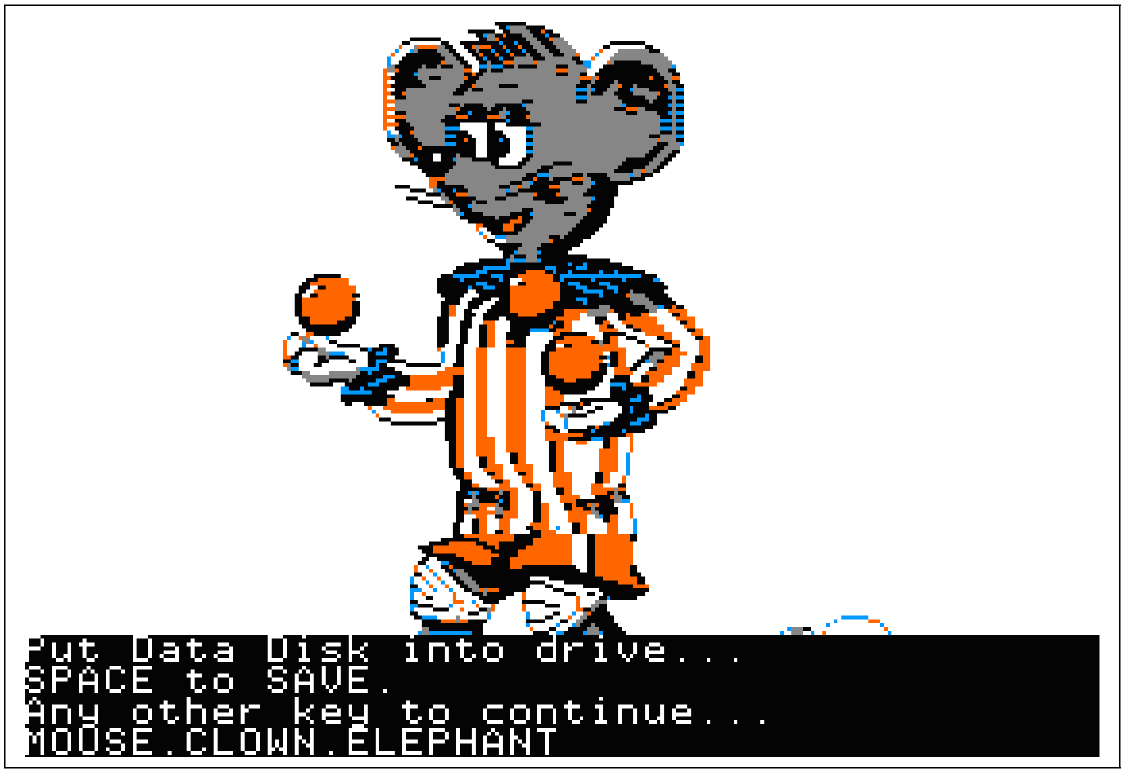
2. Whenever disk level commands are done, there is both a risk of corruption and error checking code must be added. When loading or saving error checking is simple and already in place. Yes the program does depend on the initial image fragments all being available to read and there is a risk that this may not be the case, but the program will still work with some files missing the way I have programmed it, and I distributed a functional disk image with everything in place, and loading files is a read operation, not a write operation.

If I start renaming things, it's like saving a file. Except when a file save goes bad it's because the disk is full or corrupt or because the volume label (ProDOS) has changed from that of the program disk (Program and Data Disks require the same name in ProDOS).

I don't see the benefit in renaming warranting all the extra code that I would need to write, and also to base the architecture on a rename command (a write operation) would unnecessarily complicate the program logic, complicate testing, and all associated code. It's just not a clean solution and having explained enough of my reasons that you might understand, 'nuff said.



Data Disks



```
Put Data Disk into drive...
SPACE to SAVE.
Any other key to continue...
MOUSE.CLOWN.ELEPHANT
```

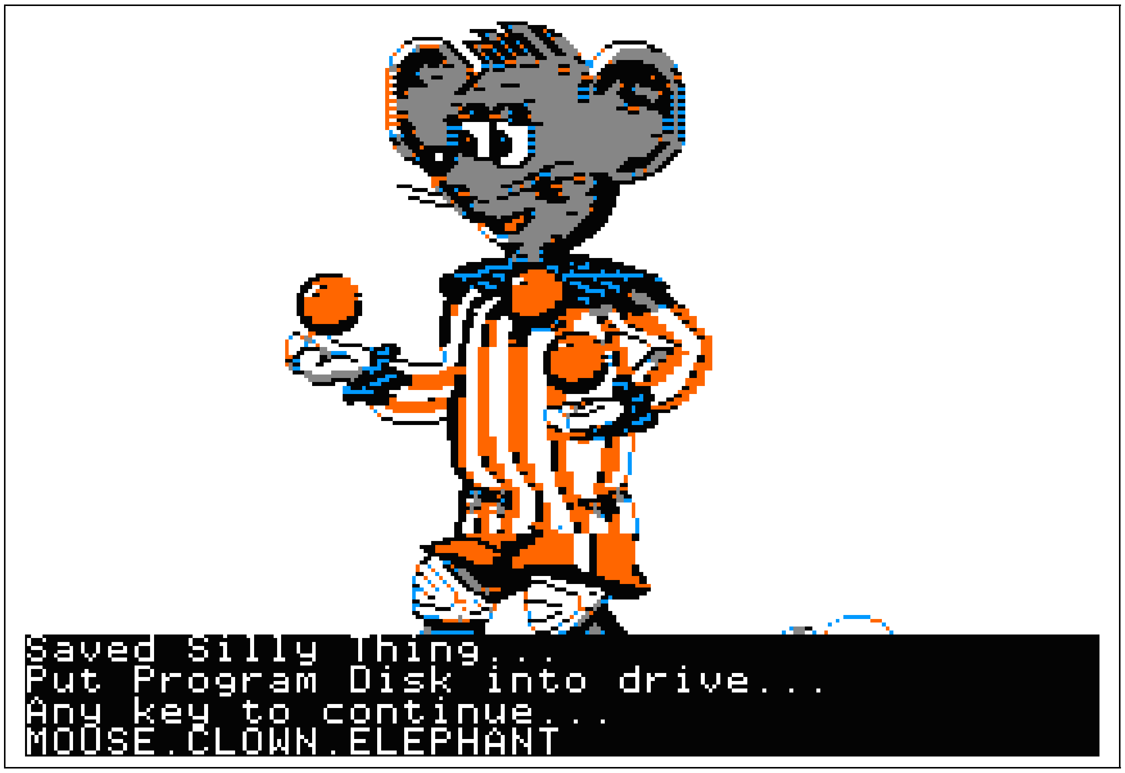
When saving in ProDOS the volume name on the DATA disks must be the same as the volume name on the program disk. The ProDOS volume on the disks that I have provided is SILLYPROGRAM.

In either the DOS 3.3 or ProDOS versions, when you are prompted to put the data disk in the drive you can skip this step and save directly to the program disk but you will clutter-up your program disk and it will soon fill-up since 1000 different Silly Things are possible.

If you are running SILLY.SYSTEM in ProDOS from a hard drive, just put the contents of both disks together and skip the changing of disks when prompted.

Since 1000 Silly Things are possible your data disk will fill-up quickly. Each BSaved Silly Thing is 8K (8192 bytes) in size.

You will need several clean copies of the Silly Things data disk (with PICLIST and BSaved Images removed) or considerable space on your hard drive if you are doing a lot of saving. On a hard drive, the PICLIST will grow quite large and you may find it more practical to run from a floppy disk and let the disk fill-up. When the disk is full, the program will still work except that it won't be able to save to the full disk.



```
Saved Silly Thing...
Put Program Disk into drive...
Any key to continue...
MOUSE.CLOWN.ELEPHANT
```

You will have several options when the data disk fills-up and you can no longer save and some of these are listed below:

1. Ignore the problem. The program will still work well otherwise.
2. Use a fresh data disk with PICLIST and BSaved Images removed.
3. Remove unwanted BSaved Images from the data disk and PICLIST

The PICLIST (a text file) can be edited manually to remove or add images to but this option may be too much work for you.

The APPLEVU Slide Show program can also be used with the PICLIST.