WBrain

COLLABORATORS			
	TITLE : WBrain		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 6, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

WBrain

1.1 Index

```
                ***************************************************************************

                                    WBrain v1.0

                                    02 AUG 1993

*******************************************************************************

                                    Index

            0 -
                 General intro
                             1 -
                   Game idea
                             2 -
                  Installation
                 System requirements
            3 -
                      Rules
                 Playing the game (Rules)
            4 -
                   Legal crap
                             5 -
                  Source code
                 About the source code
            6 -
                    Bug list
                             7 -
                     History
                 and future

*******************************************************************************
```

1.2 General intro

General intro

Brain for the WorkBench.
This program is based on Brain, by Andre Wichmann.
WBrain takes no code from Brain, and is written entirely in Amiga_E.

Amiga_E
is a programming language by Wouter van Oortmerssen which produces very small, fast code, and is designed to simplify the creation of user interfaces.

Brain v1.01 can be obtained from the Fred Fish PD disk #652.
Amiga_E v2.1b can be found on Fred Fish #848, or you can contact Wouter by mail:

Wouter van Oortmerssen
Levendaal 87
2311 JG leiden
HOLLAND

or by EMail:

Wouter@alf.let.uva.nl
Wouter@mars.let.uva.nl
Oortmers@gene.fwi.uva.nl

You can contact me only through snailmail:

Sean Russell
Claude-Lorrain-Str 31
81543 Mnchen
GERMANY

See the
Bug list
, if nothing else (nothing disasterous)!!!

****

1.3 Game idea

Game idea

WBrain is short for Workbench Brain. Brian is a game in which the player is given a randomly generated pattern which SHe must reproduce by choosing a correct order of moves. WBrain is implemented in a WorkBench environment; this means that it opens a window which can be resized to determine the size of the playing field. Moves are made by clicking on squares.

The game is fairly simple after learning the general pattern rules; it is very low-stress but still requires some thought to complete. Personally, I like the game because I can always solve it (if I think enough) without becoming too frustrated. At the same time, the game is not so mindless that it becomes boring; there is always a little challange.

Since the computer does not have to play against you, this program requires very little CPU time, and also very little memory. As the author of the original Brain said, it is a good game for when you're doing something like compiling with a slow compiler or with a big program, or while generating fractals, or while ray tracing...

****

1.4 Installation & system requrements

Installation & system requirements

WBrain requires the reqtools and gadtools libraries, of version 37 or greater. I've also made the game unuable on OS<2.0x as a safety measure, since I don't know if it would run on anything less. Sorry. I won't tell you that you should buy OS2.0 since you probably already know that by now; half the PD stuff out there these days won't run on 1.3 or 1.2 anymore. If you don't have 2.0 yet, I can only assume that you either don't have enough money or motivation to upgrade; in the first case I can't help you, and in the second case that's your own problem. If you are the unfortunate possessor of an A1000 (for which you require more than $400 to upgrade to 2.0), buy an A600 instead: it's cheaper than upgrading. Or better yet, buy an A1200, which is also an unreasonably good deal.

I've wandered from the topic of this section. To install, simply copy reqtools and gadtools into libs: (if they aren't already there). This program uses topaz 8 as it's font, so that must also be in the fonts: directory. Unless you've deleted it, it should be there as it comes with all versions of WorkBench. You need to have a 3 bitplane screen going; this is 8 colors. I imagine you could launch this onto about any public screen if you wanted to, but I wrote it with the WorkBench in mind. Double click on the WBrain icon to start the program and you're off. Or if you're a fanatical icon hater, launch it from a CLI; I couldn't care less.

You can also install this by double-clicking on the InstallMe! icon. This launches the Commodore "Installer" program, which comes with WB2.0 or greater. I've included that program in the package as well, but it will have been deleted if it's not freely distributable.

The programs included in this release are:

```
WBrain          (The program)
InstallMe!      (The installation script)
WBrain.guide    (The AmigaGuide format documentation)
WBrain.doc      (The text format documentation)
installer       (hopefully; the Commodore install program)
WBrain.e        (The Amiga_E sourcecode)
libs
   gadtoolsbox.library
   reqtools.library
```

The total space for these files is 375 blocks.

****

1.5 Playing the game

Playing the game (Rules)

Start the game (by double clicking on the icon, or from a shell).
The rules are simple: your goal is to make the left hand grid look like the right hand grid. There are three game level, which determine the difficulty of play.
Level 0: When you click on an empty square, a 1 appears there. If there are any numbers in neighboring squares (these are the 4 squares up, down, right, and left of the target square), 1 is added to the number in that square. If a square is 4 and one is added to it, it becomes a 1 again.
Level 1: Same as level 0, only all eight neighbors are effected by adding a new box.
Level 2: Same as level 0, but now neighbors two square away in the four primary directions are affected.
Level 3: Same as level 1, but now all neighbors in a range of 2 squares in all directions are affected.

The window contains (in addition to the two grids) three buttons, one slider, and three text boxes, which have the following functions:

Undo - Take back the last move. The Undo function remembers all of your moves, which means you could back up to the beginning from the end if you wanted to.
Retry - Start over with the same goal grid from the beginning. This is the same as Undo-ing back to the beginning.
New - Make a new goal
Slider - Changes the game level
Rows : - Number of rows in each grid
Colms: - Number of columns in each grid
Level: - The current game level

In addition are the standard zip, close, and back gadgets, and the size gadget. WBrain will produce the largest grids it can in the window you have sized. The starting dimensions are 8x8; I suggest the first time you play you reduce this to 5x5 or less. The largest size is 10x10, the smallest is 2x4. Every time you resize the window, a new goal will be produced. There are also two boxes under the gadgets telling you what the dimensions of the grids are, in columns and rows.

When the right grid has the same configuration as the left, a window will appear telling you that you have won the game. If you fill all the square and no such window appears, check your grid, because somewhere you've made a mistake.

The menu contains two sections, Project and Game. The project menu contains the usual Info and Quit options, and the game menu contains copies of the window buttons Undo, Retry, and New. These can be chosen by hot-key (RAmiga-u, RA-r, and RA-n, respectively). The game can be quit over the menu, the hotkey RAmiga-q, or by sending a Ctrl-C break to the program (for example, through ARTM or similar program, or if you started the program from a shell).

****

1.6 Legal crap

Legal crap

This program is Selfish-Ware: you can do anything with it you like, but you can't make any money from it, aside from a reasonable duplication fee. By "reasonable" I mean up to two US dollars. This mostly applies to those immoral characters who like to take advantage of us generally selfless programmers and market other people's software without providing due compensation. PD collections, such as Fred Fish and SAAR-AG-PD are hereby granted permission to do whatever they want with WBrain; this includes distribution and the making of money thereof (to support their endeavors).

Use and abuse the sourcecode as much as you want, but if you add a virus to it, you'll earn an eternity of bad-karma. I'd also appreciate it if you mention me somewhere (in small print) in the Info block of your modified program.

If you really want to do a good deed, or just want to do me a favor for providing you with something you like, then subscribe to the Fred Fish collection. You'll be doing yourself a favor as well. SAAR-AG-PD is also a good collection, but I think it's a reasonable choice only if you live in Germany.

The mandatory disclaimer: I make no claims that this program won't bomb your system, burn out your roms, start World War III, or do any other nasty things. I also take no resposnibility if you use this program to crack into the NSA databanks or give Bill Clinton a bad credit rating. On the other hand, if any of this stuff happens, drop me a letter so that I can brag about it in the next version.

An aside note: What I've really become sour to lately is people who expect you to send them money for their shareware package while at the same time not guaranteeing that it wont reformat your harddrive when you're not looking. Requests for money, suggestions of food gifts, and desperate pleas for contact are one thing, but "crippled"-until-you-pay-but-no-guarantees software is revolting. Yea, programmers deserve a little compensation for what they provide us with, but like I've always said:

Real programmers code for the love of programming.
Anything else is prostitution

(Don't let that stop you from sending me your recent inheritance in gratitude =-)

"This little thingy is ofcourse FreeWare, but if you REALLY like it,
you can send me your sister as payment."
-Arthur Hagen

****

1.7 About the source code

About the source code

If you want to change this code, I suggest the first thing you start with are the variable names. In my personal programs, I tend to make local variable names rather short and ambiguous, since my routines are usually pretty short and it's not that hard to remember what the variables stand for. On the other hand, that practice is considered "bad programming," ranking up there with using 'goto's. Don't do it if you want to pass your CIS class.

I don't know how much this conforms to the Commodore guides, but since it's such a small program, I couldn't have strayed too far. It's fairly multi-tasking friendly, with a WaitTOF in the main loop, and it doesn't do too much more than wait for input anyway.

I didn't program this in C, for two reasons:
1) I did more work trying to install GCC and getting it to run correctly than I did programming. One of these days I'll buy a commercial version...
2) It would have been overkill. The game consists mostly of GUI and other graphics stuff; these functions are usually the only ones that need to be changed in porting. Also, I'm a firm believer in the dogma that the best tool to use is the easiest one that gets the job done. Amiga_E is not only the easiest for interfaces for the Amiga, it also produces very compact, fast code.

Enough said.

One of the bad things about Amiga_E v2.1 (or perhaps I just missed something somewhere) is the inability to easily define (and manipulate) multi-dimensional arrays. You'll notice this if you look at the code. I had to make an array of pointers to an array. Normally, this wouldn't be so bad, but to access this array we can't simply say k:=t[5].h[6]; we have to say g:=t[5].h and then k:=g[6]. Ah well. Aside from that, programming in E is a dream.

****

1.8 Bug list

Bug list

None that I know of.

Er, uh... due to the way the Amiga_E Rnd() routine works, every time you start the program, you will get the same series of random numbers. Although this is inconvenient, resizing the window a couple of times and pressing the New button a few times more will ensure that you get a somewhat original Goal grid. In the next release I'll bind this routine to the time and date, and get a real random number. Sorry, but I was just too lazy to do it for this version.

****

1.9 History & future

```
    History and future

   Rather short history, I'm afraid...

   Version 0.0 - Not released.  Window resizing was inconvenient and GURUed
                 my machine routinely.

   Version 1.0 - \textdegree{} Resizing revamped
                 \textdegree{} Grid size displayer added
                 \textdegree{} Buttons moved to the left of the goal grid
                 \textdegree{} Made multitasking friendly.
   Version 1.2 - \textdegree{} Levels 1-3 added.

   Future:
          \textdegree{} Add a timer and scores.  This will be disablable to  ↩
             maintain a
            lowest possible CPU usage rating.
          \textdegree{} Add locale support for other languages (As I'm an American
            living in Germany it's appropriate that this is high on my
            list. =-)
          \textdegree{} Make font sensitive (not likely)
          \textdegree{} Fix the random routine (I'll probably do that tomorrow)

*****************************************************************************
```