Shelly

COLLABORATORS

	TITLE : Shelly		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		December 30, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Shelly

1.1 Contents

```
                                                         13.1.1993

Welcome to:

              SHELLY V1.2
              -----------
```

1.2 introduction

```
                INTRODUCTION:
-------------

Shelly is a little tool that generates 3D-Objects of various
shells (Ammonites, Slug-houses etc.) for: POV-V2.0, Real3DV2
and T3Dlib (the last means Imagine,DXF,Rayshade,Vort,Post-
Script etc. support! ). (take a look at "examples.jpg")

It uses an
        algorithm
         found in:
 Computer&Graphics Vol. 17,No. 1,pp. 79-84, 1993
 ("DIGITAL SEASHELLS" by M.B. Cortie.)

It was written in (portable) C using GCC2.3.3.

The POV output of Shelly consists of triangles.
The Real3D output is a RPL-Macro that you can execute via
"Execute named" in the "Macros"-menu and produces a big B-
Spline-mesh.
The T3D-output can be converted via TDDD2xxx to many different
formats (of course you need the converters (available on Aminet)
look in 'gfx/3d' ...)

Have fun with it!
```

1.3 changes

```
Changes:
--------

from V1.0 to V1.2:

- added "autofocus" for POV-output (automatical placement of the
  camera in the right distance)

- the file 'shelly.pov' is never consultet now!

- the RPL-output now generates much (100 times) smaller objects

- added T3Dlib-support (new Keyword is 'T3D'!)

- the silly countdown is gone

- this time Shelly comes with only one guide ! (hope so :))
```

1.4 contents

```
CONTENTS:
---------

shelly.lha contains:

 - shelly (the executable for C= Amiga (should work on all Amigas))

 - shelly20 (a special version of the exe that uses MC68020 and up
             & MC68881 and up)

 - shelly.c, shelly.h (source & header, ready to compile on various
                       machines)

 - examples.jpg (a picture of all examples)
                (except 'Lyria' all Shells rendered on my Amiga
                 (A4000/030, 2C/2F) with Real3D2(Demo:)))

 - Planorbis.shy, Nautilus.shy, Lyria.shy, Ammonite.shy
   Oxystele.shy, Natalina.shy           (some example data-files)

 - Shelly.guide (this document)

 - Blank.shy   (a blank datafile)
```

1.5 requirements

```
Shelly requires atleast:

 (to use the executables provided)

  - an Amiga (Harddisk & fast processor recommended)
  - 'ixemul.library' (not in this package)
  - POV-V2.0 or Real3DV2 to look at the results
  - or the TDDDlib-converters by Glenn Lewis (Shareware) if
    you want Shelly to create objects for Imagine etc.

  Shelly has been tested on the following configurations:
  -A4000/030
  -A2000D+A2630



  Shelly compiled with no problems on
  IBM-RS6000,SUN4,HP9000/345,CONVEX

  You might get problems when trying to run it on
  64Bit-machines (Franky (IRC) reported problems
  (float exceptions) on a 64Bit-MIPS ... )
```

1.6 installation

```
INSTALLATION:
-------------

The installation of Shelly is very easy ...
Just copy the Drawer "Shelly" to a place
where you like to install it.

and give it a '(g)cc shelly.c -o shelly -lm' (not needed on Amiga)
```

1.7 quickstart

```
                To get started quickly :

 - install Shelly (described in
        Installation
        )

 - open a shell (CLI), cd to the directory "Shelly"

 - type 'shelly Planorbis.shy xxx.pov'
   (Planorbis is one of the examples, xxx is the name of the
    POV-Scene Shelly will create)

 - now go and render the file 'xxx.pov'
   (e.g. 'pov -ixxx.pov -f +d' (assuming you have pov in your path))

 perhaps you have to edit the file 'xxx.pov' (camera position etc.)
 and try it again to get the best result...

For detailed information look into the
         Usage
          section.
```

1.8 usage

```
                     Usage
     -----

just type

'Shelly infile outfile'

to run Shelly from a shell (CLI)

-infile is the (path+)name of a datafile
-outfile is the (path+)name of the POV/RPL output

note: outfile will be overwritten (if it exists)!
```

```
- after running Shelly you should be able to
  render the outputfile with POV
  or execute the output as a macro (in Real3D)
  if RPL-output was choosen in the datafile
  or if T3D-output was your choice, convert it and render
  it in Imagine etc. pp.


now to that mysterious DATAFILES:
Shelly uses own Datafiles in a simple
        format
        .
There is a special file ("blank.shy") prepared for you that is blank.

consider:

 -some parameters have to be given in degrees, some not
   (look into the
        algorithm
         section)

 -if you want Shelly to create a RPL file as output add a line
   like "RPL" or "pleazze do it in RPL" to the file
   ("T3D" will switch to T3D-output)
   (POV output is default)

 -be careful with the parameters, don't try to fool Shelly ("what
   does it do if i enter an infinite value :)?") it will end up
   in a mess or coredump or our beloved friend! because the values
   are not checked!
   You should just change the given examples slightly until you
   know what you are doing...

 -smin,smax,sd,omin,omax,od are very critical parameters
   because they determine the size of the output and the memory
   consumption while calculating the shell

 -o must be positive! (omin >=0, omax >omin, od >0)

 -always remember:
   This program has still the status "experimental"!



 - several PROBLEMS may occur:

   - it is nothing to be seen in POV:
      probably the camera/light positions are wrong
      take a look at the data in your pov-file and correct this

   - POV tells me something from "degenerated triangles"
      well this problem did not occur yet (in shelly) but i know
      it could happen (former projects)
      nothing serious, just some triangles with 2 points the same

   - Real3Ds annoying "Stack full" message comes up everytime
     i try to execute a macro:
```

```
     - change the RPL-stacksize (menu: Settings/RPL)
        (increase the "Parameter Stack")
     - open a new RPL-window
     - type: '"(path+)macroname" LOAD'

  - strange numbers (NaN's) occur in the output:
     Well this problem is known to me but no solution (sorry).
     Since the algorithm is somewhat complex i really don't
     want to have to find out which combination of which
     parameters cause this.
     It is also a problem of the sideeffects and (numerical)
     stability of the "mathematic" functions i call.

     note: i suppose zeros are the source of all this
           -> try to avoid them
```

1.9 hints

```
Hints
-----

for the Real3D-user:

- remember that for a mesh the first and last line
  (and in each line the first and last point)
  of the shell will be invisible (unless you switch objecttype
  to Polygon or Phong)
  that means for a shell with smin:10, smax:210, sd:20 that you
  will see a shell created from smin:30 to smax:190!
  (all examples will suffer from this if you just add the RPL
   keyword)
  solution: increase the ranges of s and o.


- if you want nodules in RPL-objects:

  You should choose proper values of od and sd to see the nodules
  at all
   (if you have nodules that are 10\textdegree{} wide (in o-direction) and ↩
       you
    choose an od of 40\textdegree{} you will see probably no nodules!)
   (this is also important for the POV-output)

  You should double the nodule height (L) for B-Spline objects
  to get the same height of the nodules as a POV-output!


- if you want to create a shell without nodules you can
  double the sd and od values for B-Spline objects without loss
  of quality in many cases
```

1.10 algorithm

```
The Algorithm:
--------------

In this section you will find more detailed information on the
algorithm used by Shelly and on the parameters it uses.

- The basic idea of the algorithm is to simulate a shell shape
  by rotating & moving (&copying) an ellipse (or a part of an
  ellipse, or any other curve (a cardiod)) around an axis.
  This will end up in some sort of spiral-shape.

- The shape produced will depend on many things like:
       -starting size/place/orientation of the ellipse
       -exact form of the ellipse (nodules)
       -how fast is the ellipse growing while rotating etc.

- you can find the exact formulas in the original article or
  in the sourcecode (too lazy to write them here again, they are
  very complex)

- here is a list of all parameters that shelly needs to generate
  a shell:

 -angular parameters (given in degrees):
  alpha    :equiangular angle of spiral
  beta     :angle between z-axis and line from aperture local
            origin to xyz-origin
  phi      :tilt of ellipse major axis from horizontal plane
  omega    :amount of azimuthal rotation of aperture
  my       :amount of "leaning over" of aperture

  smin     :angle at which aperture generating curve begins
  smax     :angle at which aperture generating curve ends
  sd       :stepsize in s-direction
  omin     :angle at which spiral begins
  omax     :angle at which spiral ends
  od       :stepsize in o-direction

  P        :position of nodule, in terms of angle s
  W1       :width of nodule in s-direction
  W2       :width of nodule in o-direction

 -linear dimensions
  A        :distance from main origin of aperture at o=0
  a        :major radius (long axis) of ellipse at o=0
  b        :minor radius (short axis) of ellipse at o=0
  L        :height of nodule at o=0

 -other
  N        :number of nodules per whorl


- the parameters smin,smax,sd,omin,omax,od determine
  how many triangles (controlpoints) are generated
  (how smooth is the shell and how many whorls are generated)
  -> be careful with these: memory usage and filesize of Shelly
     depend directly on this parameters

- the parameters alpha,beta,phi,omega,my determine the orientation
  of the ellipse before (and while) rotating

- the parameters A,a,b determine starting place and size of the
  ellipse

- the parameters P,N,L,W1,W2 determine number,size and place
  of nodules
```

1.11 credits

```
Credits:
--------



- M.B. Cortie for his article "Digital Seashells"
- Martin Huttenloher for the icon of the guide
  (Thanks for MagicWB!)

Thanks to the people who ported GCC & CSH to the Amiga
and to Soulman (IRC) who helped me to realize the difference
between 2 and 2.0 :).
```

1.12 distribution

```
                DISTRIBUTION:
-------------

Shelly may be distributed FREELY via any media as long as:

1) The archive shelly.lha and its
        content
         remains unchanged.

2) No money (except a small copying fee) changes hand.



(Although Shelly is Freeware i won't reject gifts like
 money, chocolate, your latest piece of (gfx related) code etc..
 My adress can be found under "
        contacting the author
        ".)
```

1.13 disclaimer

```
DISCLAIMER:
-----------

This program comes with no warranty, either expressed or implied.
The author is in no way responsible for any damage or loss that
may occur due to direct or indirect usage of this software.
Use this software entirely at your own risk.
```

1.14 adress

```
send
chocolate, money, your programs, bug reports (NOOOOOOO!) etc.
to:

Randolf Schultz
Unter den Linden 51
19079 Mirow
GERMANY

i know it is hard to send chocolate via email but you could give
it a try ...

INTERNET: rschultz@informatik.uni-rostock.de

this adress is a bit unsafe, you could also try

          tfb512@hp1.rz.uni-rostock.de
```

1.15 fileformat

```
                   Fileformat:
   -----------

The files are of a very simple (and easy to process :)) format:

- every line of the file is scanned for
        keywords
        .

- if a line contains no keyword it is treated like a comment

- if a line contains a keyword it is interpreted
  (the number behind the keyword is copied into an internal structure
   ("alpha:30" sets the internal alpha value to 30)
  or a flag is set
   (the 'RPL' keyword sets the flag "we_have_to_produce_an_RPL-file")
  )
```

```
- as you can see we have keywords that need parameters behind them
  and keywords that just have to be there to set something

- the only "Flag-keywords" the program knows are: 'POV' 'RPL' 'T3D'
  all other keywords need to be combined with a number (as the ':'
  states)

- everything is casesensitive! ('RPL' != 'rPl')

- the file is not checked for anything else
   (double use of the same keyword cause an overwriting of the
    last set value)
   (lines like "alpha:Blafasel" will cause NO errormsg,
    such things are really your problem)


note: the keyword does not have to stand alone!
      if you write a line like:

      "/*RPL*/" or "BlafaseRPLl"

      the RPL-flag will be set! But you could also write:

      "render this in RPL pleazze :)"
```

1.16 keywords

```
                The following keywords are supported:


'alpha:'
'beta:'
'phi:'
'omega:'
'my:'
'smin:'
'smax'
'sd:'
'omin:'
'omax:'
'od:'
'P:'
'L:'
'A:'
'a:'
'b:'
'W1:'
'W2:'
'N:'
'RPL' (switches to RPL-output)
'POV' (guess)
'T3D' (hmm)
```

```
(note that the ':' belongs to the keyword!  you can use
 for instance the word 'alpha' with no risk in comment lines)

(The meaning of a special parameter (keyword) can be found
 in the section about the
        algorithm
        .)
```