

agl

COLLABORATORS

	TITLE : agl		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		July 22, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	agl	1
1.1	main	1
1.2	what	1
1.3	disclaimer	2
1.4	who	2
1.5	need	2
1.6	consist	3
1.7	functions	4
1.8	future	5
1.9	excluded	6
1.10	differ	6
1.11	sgi	7

Chapter 1

agl

1.1 main

Amiga GL by Jason Weber
Copyright © 1994, Jason Weber

What is AGL?
What hardware and software is required?
What should the archive contain?
What functions are supported?
What may be implemented in the future?
What is not explicitly not implemented
How is AGL different than IrisGL(TM)?

Who wrote AGL and how do I pay for it?
DISCLAIMER
Trademarks

1.2 what

What is AGL?

This library is a subset of the IrisGL(TM) graphics language. IrisGL(TM) was developed by Silicon Graphics Inc SGI(TM). The Amiga version presented here is not produced, supported, or endorsed by SGI(TM). SGI(TM)'s unofficial release of their API allows me to release my Amiga version to the general public.

This software was produced using IrisGL(TM) user manuals and experience on SGI(TM) systems. No source was obtained or used from the original SGI(TM) source. The goal of this library is total upwards compatibility to a compliant platform.

The user should obtain documents which explain the use of IrisGL(TM). There are no lessons, tutorials, or texts included in this release. I recommend the "Graphics Library Programming Guide" distributed by SGI(TM).

1.3 disclaimer

DISCLAIMER

This version of IrisGL(TM) is non-compliant and unlicensed. It bears no guarantee of fitness for any particular purpose. The user assumes all responsibility for any actions or damage resulting from its use. Many significant parts of the real IrisGL(TM) specification have not been included in this release.

This library provides an introduction to the IrisGL(TM) graphics language and is not intended for serious or commercial use. Serious parties should seek to purchase a compliant licensed version of IrisGL(TM) or OpenGL(TM) on an acceptable platform.

1.4 who

Who wrote AGL?

This source/library is property of Jason Weber. Before the official release, you must obtain permission to use this software by contacting:

Jason Weber
403 Montpelier Dr.
Stafford, VA 22554

No one is permitted to distribute a modified version of this software without permission. Any redistribution must contain all of the original files. Any fix or improvement must be passed through the original author.

There is no fee to copy the software. Permission to use is only granted to users who register the software. This allows me to maintain contact and gives me incentive to make repairs and improvements. You should send the following to the address above:

Name
Address
Occupation
What you are using the software for
e-mail address
\$20 registration fee

If you truly can not afford the fee, you still register without it by adding the following line to your registration: "I can afford all this computer equipment, but I can't afford to support the work of fellow programmers". If you have written a public domain or shareware program that I am using (Vim, Csh, KingCON, HWGRCS, Enforcer, CPUblit, FASTERblit, MagicMenus, MagicWB, Toolmanager, VMM), state this and we can just call it even.

1.5 need

What hardware and software is required?

The library's source can be compiled for any Amiga. I believe I followed all the "rules" and avoided any tempting tricks. The given library was compiled for 68030 with coprocessor under SAS(TM) 6.51. Check "scopts" to see how it was compiled so that your code will match.

I use a 25Mhz Amiga 3000. I would recommend this as a mininum configuration. The performance is poor on a stock Amiga 500.

The compile scripts use csh (Csh 5.37 by Urban D Mueller/Andreas M Kirchwitz).

The middle mouse button is fully supported. If you don't have a middle mouse button, it is very unlikely that you will receive a MIDDLEMOUSE event.

SAS is a trademark of SAS Institute Inc.

1.6 consist

What should the archive contain?

The following files, at least, should be included in this archive. I may have submitted the precompiled library separately or not at all. Except for the font, all these directories/files should be placed in a directory assigned to "usr:".

The library:

```
libs/  
gl.lib
```

My font:

```
fonts/  
JXEN/7  
JXEN.font
```

The required headers:

```
include/  
gl.h  
device.h
```

Some utilities:

```
bin/  
  basename      (executable; replaces csh's incorrect builtin version)  
  prototype     (executable; my auto-prototyper)
```

The source:

```
src/basename
  basename.c

src/prototype
  prototype.c

src/agl
  prototypes.h (will be generated automatically by 'prototype')

  agl.guide    (you're reading it)

  keymap.h
  agl.h

  border.c      (Motif(TM) style borders)
  clip.c        (scrmask clipping using Regions)
  matrix.c      (transforms)
  poly.c        (primary drawing)
  que.c         (event handling)
  rgb.c         (auto-dithering for RGBmode())
  text.c        (text drawing)
  window.c      (window handling)

  mice.c        (test to read second mouse, not really used)
  sprite.c      (test to draw second mouse, not really used)

  makefile
  Compile
  sas.makefile

  RCS/          (revision control)
```

Sample code:

```
src/gltest
  gltest.c
```

"usr:bin" should be added to your path. Note that this not enough to override csh's builtin incorrect "basename" which has to be aliased. The Compile script for AGL does this alias to make sure.

You should also copy the font into your fonts: or AGL will use your default system font. The JXEN font is a reduced version of XEN from MagicWB by Martin Huttenloher. I believe JXEN 7 to be the smallest readable fixed-space font. I use it for my shells.

1.7 functions

What functions are supported?

CONFIGURATION

```
gversion(), getgdesc(), getdisplaymode()
```

WINDOWS

```
foreground()
prefposition(), prefsize()
winopen(), winclose(), winset(), winget(), winpush(), winpop()
winposition(), winmove(), wintitle()
getsize(), getorigin()
singlebuffer(), doublebuffer(), swapbuffers()
cmode()
RGBmode(): EXPERIMENTAL
gconfig()
```

EVENTS

```
getbutton() for keys and mouse buttons
getvaluator() for MOUSEX, MOUSEY

qdevice(), unqdevice(), isqueued(), qreset()
qenter(), qtest(), qread()

tie() to mouse valuator

queue-able events: REDRAW, INPUTCHANGE, MOUSEX, MOUSEY,
                  LEFTMOUSE, MIDDLEMOUSE, RIGHTMOUSE,
                  KEYBD, WINQUIT, most individual keys
```

TRANSFORMS

```
perspective(), ortho2(), ortho()
scrmask(), viewport()
pushmatrix(), popmatrix()
translate(), rot(), scale()
mmode(), getmmode(), loadmatrix(), getmatrix(), multmatrix()
```

DRAWING

```
mapcolor(), getmcolor(), color(), getcolor()
clear()
bgnpoint(), endpoint(), bgnline(), endline(), bgnpolygon(), endpolygon()
v2i(), v3i(), v2s(), v3s(), v2f(), v3f()
cmov2s(), cmovs(), cmov2i(), cmovi(), cmov2(), cmov(),
charstr(), getcpos()
recti(), rectfi(), rects(), rectfs(), rect(), rectf()
```

Check in prototypes.h to see if a specific function is supported. Note that there are many non-IrisGL(TM) support functions in there as well. None of those should be called directly from the user's software.

1.8 future

What may be implemented in the future?

auto-dithered RGBmode()
OpenGL(TM) compatibility
a 24-bit graphics board
pop-up menus

1.9 excluded

What is explicitly not supported?

zbuffer()
lighting
shading
texture mapping

1.10 differ

How else does AGL differ from real IrisGL(TM) on an SGI(TM)?

1. A single solitary optional Amiga-specific function is included to configure the screen:

```
long AGLconfig(short screenx,short screeny,short bitplanes)
    returns nonzero if successful
```

If used, this call must be made before any GL call or it will have no effect and just return 0. It specifies the size and depth of the screen. AGL will automatically use hi-res laced if either dimension exceeds a specific level. The actual screen created may be different than what the user requested. The function getgdesc() should be used to get the actual values for the screen.

2. If there are multiple windows in the application, window-independent double-buffering is simulated using a blit copy from a backbuffer. If there is only one window, true double buffering (of the screen) is used.

3. Interactive window placement is not supported and all programs are run in the foreground. A warning is issued if foreground() doesn't precede the first window opening.

4. Intuition uses a structure "Device". Therefore, the IrisGL(TM) typedef of "Device" is intentionally removed. You should use "long" instead to refer to IrisGL(TM) devices.

5. AGL is run on a separate screen that is created automatically on your first winopen(). It is closed automatically when you winclose() your last window.

6. Intuition does not have a window auto-kill and cleanup like on an SGI(TM). A sudden break (like CTRL-C) may leave the windows and memory in use. Your program should issue a qdevice(WINQUIT) to queue the close gadget and close all

your windows when it is done. AGL automatically shuts down when the final window is closed. As an added precaution, AGL uses the ANSI `atexit()` function to shut down automatically during an `exit()`, hopefully. However, you should not rely on this as it is not clean programming practice.

7. You should use `AutoPoint` and `ClickToFront` to act more like the SGI(TM). There will be a slight difference from the SGI(TM) in that a window will retain focus even after the mouse is moved out and until another window is selected. This should only be a problem in that the middle and right buttons may report outside the window.

8. AGL has its own built-in window borders that should look and act similarly to Motif(TM) as one would have under IRIX(TM) 4.0.X and beyond, including the Maximize button. However, the upper left menu button does not produce a menu, but immediately enters WINQUIT onto the queue. The Minimize button does nothing at this time.

9. MagicMenus by Martin Korndorfer is highly recommended. This would actually be relevant if I were to add pop-up menus.

10. Intuition can sometimes send double Refresh's resulting in double REDRAW's. If you read through the entire queue at once, you can eliminate redundancy by only redrawing once.

11. I cannot directly poll the mouse buttons for `getvaluator()`. Instead, I update a value on every message from intuition. (Intuition messages all give the current mouse button state) The noticable difference is that when a no AGL window is focused, the AGL program gets no messages and, therefore, the mouse button state is not updated. The key states are done similarly, except that they are only updated on key events.

12. I've made no effort to implement the obsolete MSINGLE matrix mode even though it's the default. The user should always call `mmode(MVIEWING)` or `mmode(MPROJECTION)` before doing matrix operations. AGL will just use MVIEWING when in MSINGLE mode, but this may not (should not) be the case when ported to another platform. MTEXTURE mode is not supported.

1.11 sgi

Silicon Graphics Inc., SGI, IrisGL, and OpenGL are trademarks of Silicon Graphics Inc.