Subject: Allegro5 - Allegro game programming library v5.x
Posted by Sender Ghost on Wed, 01 Aug 2012 01:24:38 GMT
View Forum Message <> Reply to Message

Homepage:
http://alleg.sourceforge.net

License:
zlib

Version:
5.1.9 (Jan 3, 2015)

Description:
Allegro 5 is cross-platform library mainly aimed at video game and multimedia programming. It handle common, low-level tasks such as creating windows, accepting user input, loading data, drawing images, playing sounds, etc. and generally abstracting away the underlying platform. However, Allegro is not a game engine: you are free to design and structure your program as you like.

According to the Oxford Companion to Music, Allegro is the Italian for "quick, lively, bright". It is also a recursive acronym which stands for "Allegro Low LEvel Game ROutines". Allegro was started by Shawn Hargreaves in the mid-90's but has since received contributions from hundreds of people over the net.

Allegro 5 is the latest major revision of the library, designed to take advantage of modern hardware (e.g. hardware acceleration using 3D cards) and operating systems. Although it is not backwards compatible with earlier versions, it still occupies the same niche and retains a familiar style.

Allegro v5.1 supports the following platforms:
- Unix/Linux
- Windows (MSVC, MinGW)
- MacOS X
- iPhone
- Android

Allegro only supports 2D graphics primitives natively, but it is perfectly reasonable to use Allegro alongside a 3D API (e.g. OpenGL, Direct3D, and higher level libraries), while Allegro handles the other tasks. Allegro is also designed to be modular; e.g. if you prefer, you can substitute another audio library.

Documentation:
http://alleg.sourceforge.net/a5docs/refman/index.html

In the attachments you could find Allegro5 source code and addons converted to U++ packages and documentation (in pdf format).

To note:
Originally, Allegro5 uses CMake build system, which generates special configuration header files for supported platforms and available dependencies/features. The packages in the attachments are configured to support Windows and Unix/Linux operating systems only with specific dependencies, but other (e.g. *.m for MacOS X and IPhone, etc.) files still there, if you want to include them to build and configure.
Also this is slightly modified version of Allegro5 to support U++ package structure and features.

Requirements:
1. Third party dependency libraries (as U++ packages) from below message with following configured assembly package nests:
Allegro;Libraries;upp\uppsrc
where Allegro - the path of extracted Allegro5 libraries; Libraries - the path with third party dependency libraries; upp\uppsrc - the path to U++ uppsrc directory with *.upt templates to create packages.

2. a) On Windows you need to install DirectX SDK headers and libraries and configure them for appropriate TheIDE builder(s), e.g.
for MSC compiler: download and install June 2010 SDK and add following paths to TheIDE builder:
Include directories:
C:\Program Files\Microsoft DirectX SDK\Include
LIB directories (for 32 bit operating system):
C:\Program Files\Microsoft DirectX SDK\Lib\x86
LIB directories (for 64 bit operating system):
C:\Program Files\Microsoft DirectX SDK\Lib\x64

for MinGW GCC compiler: Download and extract dx9mgw.zip to preferred directory. Then add include and lib directories to TheIDE builder respectively.

2. b) On (Ubuntu based distributive of) Linux you need to install:
- Development libraries for X11: xorg-dev
- Development libraries for OpenGL: libgl1-mesa-dev, libglu-dev
- Development libraries for GTK (native dialog support): libgtk2.0-dev

Edit: Updated to 5.1.9 (WIP) version.
Fixed link for dx9mgw.zip.

File Attachments

1) Allegro_v5.1.9.7z, downloaded 390 times

Subject: Allegro5 - Examples and demos
Posted by Sender Ghost on Wed, 01 Aug 2012 01:26:33 GMT
View Forum Message <> Reply to Message

In the attachments you could find U++ packages for Allegro5 with examples and demos.
They require data files to run, which you could download from here (about 4.86 Mb) or by

extracting following directories from release archive (about 5.95 Mb) to data directory:
allegro/examples/data -> data
allegro/demos/cosmic_protector/data -> data
allegro/demos/skater/data -> data
Place them to the same directory, where you extracted Allegro5 library or add additional assembly.
After building of applications, move them near data directory to run. The place of data directory might be different on Unix/Linux operating systems.

To note:
This is modified version of Allegro5 examples and demos to support U++ package structure and features.
Also I fixed possible memory leaks on application(s) exit and changed some data paths.
Checksums for data zip archiveCRC32:
77209a15
MD5:
d68e47aff3321e0270ec48b33047268a
SHA256:
ce5473122a626df5102111749d7fb31b49b49e3a4ff81c4f282154d21d5b6a9e

Edit: Updated to 5.1.9 (WIP) version.

File Attachments

1) Allegro_v5.1.9_extra.7z, downloaded 364 times

Subject: Allegro5 - Third party libraries
Posted by Sender Ghost on Wed, 01 Aug 2012 01:33:39 GMT
View Forum Message <> Reply to Message

In the attachments you may find U++ packages with following Allegro5 third party libraries:

AL - OpenAL cross platform audio library (v1.13)
bzlib - bzip2/libbzip2 - a program and library for lossless, block-sorting data compression (v1.0.6)
freetype - FreeType 2 font engine (v2.4.11)
jpeg - JPEG image file format (v9a)
ogg - Ogg - Media container (v1.3.2)
physfs - PhysicsFS - a portable, flexible file i/o abstraction (v2.0.3)
png - PNG image file format (v1.6.16)
theora - Theora - Video codec (v1.2.0alpha)
vorbis - Vorbis - Audio codec (v1.3.4)
zlib - zlib compression library (v1.2.8)

Edit: Removed png/CHANGES file to reduce file size of attached archive, because of file size requirements.

File Attachments

1) Allegro_v5_dependencies_20150202.7z, downloaded 366 times

Subject: Re: Allegro5 - Third party libraries
Posted by koldo on Wed, 01 Aug 2012 07:12:45 GMT
View Forum Message <> Reply to Message

A huge job!

Subject: Re: Allegro5 - Third party libraries
Posted by bushman on Fri, 03 Aug 2012 01:38:14 GMT
View Forum Message <> Reply to Message

I had a hard time compiling the 'skater' and 'a5steroids' exemples in Windows because MSC compiler was complaining about multiple copy of _main function in 'barkmel.obj', previously defined in skater.c and a5teroids.cpp, respectively. Just commented out main definition in vorbis/lib/barkmel.c and it solved the problem. I'm just thinking that others might have the same issue, so here is the solution.

tks.

Subject: Re: Allegro5 - Third party libraries
Posted by Sender Ghost on Fri, 03 Aug 2012 02:17:19 GMT
View Forum Message <> Reply to Message

Hello, Edson.
kropniczki wrote on Fri, 03 August 2012 03:38I had a hard time compiling the 'skater' and 'a5steroids' examples in Windows because MSC compiler was complaining about multiple copy of _main function in 'barkmel.obj', previously defined in skater.c and a5teroids.cpp, respectively. Just commented out main definition in vorbis/lib/barkmel.c and it solved the problem. I'm just thinking that others might have the same issue, so here is the solution.
Thank you, for pointing this out. Also, just enough to remove such "vorbis/lib/barkmel.c" application from vorbis U++ package. I uploaded fixed archive to the "third party libraries" message.
(Un)fortunately, I didn't have such problems with my versions of compilers, therefore I didn't see such kind of error.

Subject: Allegro5 - Scope addon
Posted by Sender Ghost on Wed, 22 Aug 2012 19:11:52 GMT
View Forum Message <> Reply to Message

There are some ALLEGRO_* types, which require resource management, such as ALLEGRO_BITMAP, ALLEGRO_DISPLAY, ALLEGRO_EVENT_QUEUE with al_destroy_bitmap, al_destroy_display, al_destroy_event_queue function respectively, etc.

For such cases I created allegro5/scope C++ addon. It consist of:
- al_destroy template function, which invokes appropriate al_destroy_* function, based on argument type:
Toggle example#include <allegro5/allegro.h>
```
#include <allegro5/allegro_scope.h>

int main(int argc, const char *argv[]) {
 if (!al_init()) return 1;
 ALLEGRO_EVENT_QUEUE *queue = al_create_event_queue();
 ALLEGRO_DISPLAY *display = al_create_display(640, 480);
 ALLEGRO_BITMAP *bitmap = al_create_bitmap(32, 32);

 al_register_event_source(queue, al_get_display_event_source(display));

 al_destroy(queue);
 al_destroy(bitmap);
 al_destroy(display);

 return 0;
}
```

- Scope template class, which uses al_destroy template function on scope completion:
Toggle example
```
#include <allegro5/allegro.h>
#include <allegro5/allegro_scope.h>

int main(int argc, const char *argv[]) {
 if (!al_init()) return 1;
 Scope<ALLEGRO_EVENT_QUEUE> queue = al_create_event_queue();
 Scope<ALLEGRO_DISPLAY> display = al_create_display(640, 480);
 Scope<ALLEGRO_BITMAP> bitmap(al_create_bitmap(32, 32));

 al_register_event_source(~queue, al_get_display_event_source(~display));

 return 0;
}
```

- ScopeList class, which has container to store ALLEGRO_* types (actually, as void *) and invoke approriate al_destroy_* function (in some predefined order, e.g. al_destroy_display function used last) on scope completion:
Toggle example
```
#include <allegro5/allegro.h>
#include <allegro5/allegro_scope.h>

int main(int argc, const char *argv[]) {
 if (!al_init()) return 1;
 ScopeList sl;
 ALLEGRO_EVENT_QUEUE *queue = al_create_event_queue(); sl.Add(queue);
 ALLEGRO_DISPLAY *display = sl << al_create_display(640, 480);
 ALLEGRO_BITMAP *bitmap = sl << al_create_bitmap(32, 32);

 al_register_event_source(queue, al_get_display_event_source(display));

 return 0;
}
```

Some notes:
The allegro5/allegro_scope.h header file is context sensitive, which means, that you need to include it after other allegro* header files (including addons). This is because of preprocessor, which checks used header files (by already defined header guards, e.g. __al_included_allegro5_allegro_h) to associate their ALLEGRO_* types and al_destroy_* functions (where associations created manually).

How to install:
The allegro5/scope is header-only addon, at the moment, with some defined directory structure for Allegro5 addons. I created allegro5/scope/scope.cpp file as a stub for compatibility with TheIDE, but not required to compile. Therefore, either:
- include the path to base folder with allegro5/scope addon
or
- in terms of TheIDE, create new assembly with following package nests:
AllegroDev;Allegro;Libraries;upp\uppsrc
where AllegroDev is path to base folder with allegro/scope addon; other paths explained above.
And add allegro5/scope package to your applcation package.

In the attachments archive you could find allegro5/scope addon and some example.

File Attachments

1) Allegro_v5_scope_addon.zip, downloaded 431 times