Arcan – Getting Started

This document provides you with a short route to getting Arcan up and running -- making sure that it is compatible with your computer and setup, along with a brief primer of terminology and structure useful when developing your own theme.

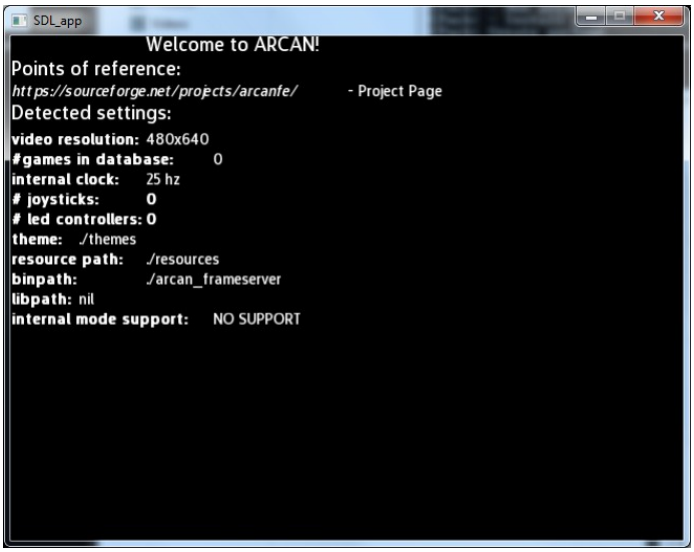
1. Test-Run

.1 – Welcome theme

Assuming you have already installed the program package as such, you should be able to just launch the program to be presented with the 'Welcome' theme.

Note: For OSX / Linux, this is preferably done from a console since it will be helpful later on. For Windows users, the various shortcuts added to the start menu should suffice for now.

This theme provides you with some basic information about your specific configuration, and will be shown if you launch the program without specifying a theme, or specifying a theme that is missing or incomplete and is thus also to provide some basic troubleshooting support.



The welcome theme looks something like the screen-shot above. Worth to note here is the "*#games in database*", "*theme*", "*resource path*" and "*libpath*" values, particularly for OSX/Linux users as the path where the program binaries, resources and themes are placed is determined dynamically and may depend on distribution, configuration and environmental variables. These search-paths can also be overridden with command-line switches.

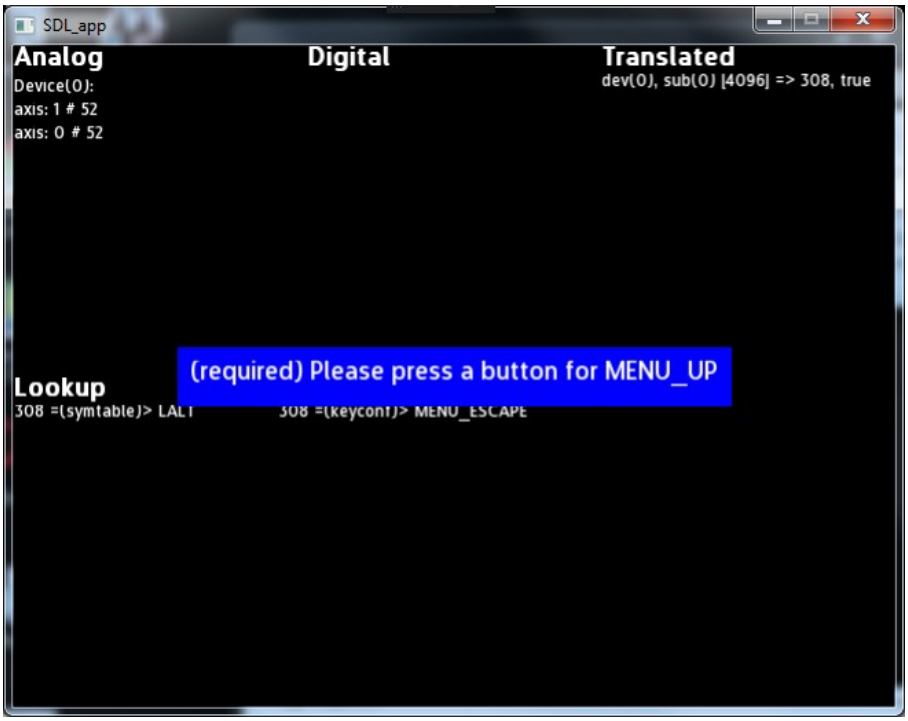
Troubleshooting:
If this window does not appear, there are a few options (ordered likely to unlikely)

- OpenGL drivers (make sure you have the latest ones, particularly for those running on Win/Lin)
- Damaged theme resources (<resourcepath>\fonts\default.ttf could not be found)
- Damaged theme script, redownload and reinstall, check <themepath>\welcome\welcome.lua

There is usually some troubleshooting trace data to be found on ”stderr”. For some windows versions, this is silenced by some builds of external dependencies. To force tracing error data to a file, use the -2 filename command-line argument.

.2 – eventtest theme

This theme allows you to test all different kinds of keyboards, mice, joysticks etc. that might be hooked up to your system. When you first launch however, you'll likely see something like:



The blue box will reappear in most themes, and is part of a support script found in *<resources>/scripts/keyconf.lua* and is used to query a user for inputs matching different events

MENU_ESCAPE is always the first label this script will try to define, as that can be used to skip other options (not marked as 'required'). You can map many labels to the same button, their actual effect will depend on the theme it is used in.

Now you can make sure that all the devices you like to be able to use are actually detected and work.

Troubleshooting:

If, after you have defined a few labels, the dialog doesn't disappear, this means that the script could not save the configuration. This happens when you don't have write- access to your theme- folder (and subdirectories). Change permissions for both your resources and your themes subtrees.If you want to reset the keyconfig, delete the <themepath>/<current theme>/keysym.lua

.3 – soundtest theme

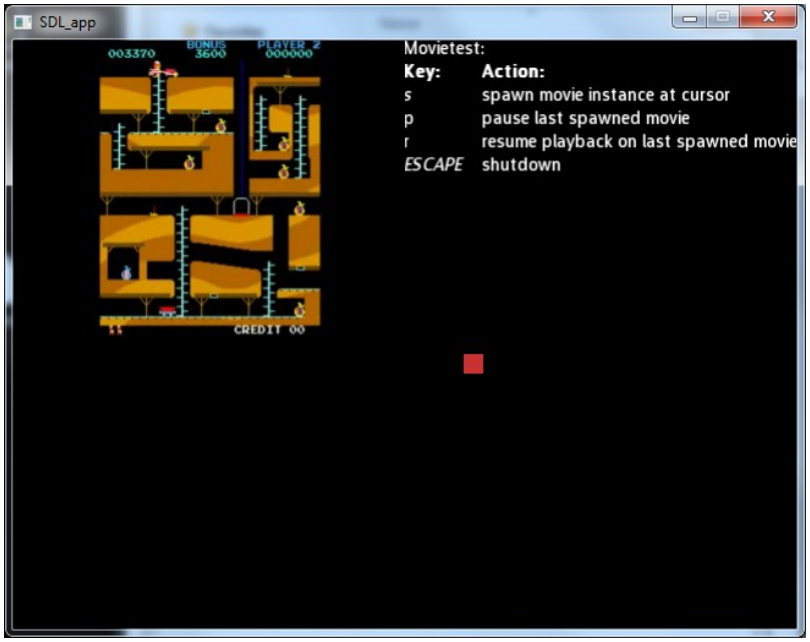
This theme is provided to make sure that your system can handle both music playback and sound samples. Launch the theme, press 'c' on your keyboard and listen for a familiar sound ;-)

If that works, proceed to press 'd' and make sure that you hear music playing.

Troubleshooting:

Make sure that sound works in other parts of your system (so we can exclude wiring, speakers, drivers, volume, ...) check the themefolder so that both the samplefile and the musicfile is included. If all that checks out OK, remaining culprits would be sound-card drivers + OpenAL combination (check for new drivers) along with permissions and possible conflicts with various sound daemons (important for the Linux- crowd).

.4 – movietest theme



Immediately upon launch, you should see a video-clip playing in the topleft corner of the screen and a red box acting as mouse cursor. Let the clip loop once and make sure it restarts properly. Spawn a few more movie playback sessions by pressing S (these will not loop).

Note that the movie playback is not as robust in terms of supported formats (huge topic) and is indended for small video snaps, not being a full-on movie player. *There is a known issue with frameserver and audio buffering on some windows configurations when audio starts repeatedly playing the last second of audio on high load and certain Window Manager events.*

Troubleshooting:

Make sure that the frameserver was found properly, this can be seen in the display from the Welcome theme. Check that the permissions on the arcan_frameserver binary are correct (albeit it won't launch properly outside the confines of the main program).

2. Configuration

For remaining tests, and getting any use out of the program, we now need to set up an emulator to launch, some games and some media to present alongside. It is assumed that you have some experience as to how this works for mame- dedicated frontends (regarding roms, video-snaps and screenshots).

Locate the *resource* folder- for your installation.

Note the folders marked '*targets*', '*games*', '*movies*', '*screenshots'*.

There is a strong connection between file-system layout, database and how Arcan behaves. A '*target*' is simply a launchable binary residing in <*resources\targets\targetname*>. For each target, there should be a corresponding folder in <*resources\games*>. These then have to be entered into the database <*resources\arcandb.sqlite*> for which there is a helper script (”*romman*”).

.1 – mame

First, copy or link your precompiled mame binary into <resources\targets> as 'mame'.

Second, create a subfolder in <resources\games> also called 'mame'.

Copy some romsets into the <resources\games\mame> folder.

Copy matching screenshots into <resources\screenshots> and movieclips (avi) into <resources\movies>.

Now we want to use the *romman* tool to scan, verify, import etc. these roms into the database.

Windows

For the prebuilt Windows installation package, there's a Build DB shortcut that will do this for you, click it, and if your folders, mame etc. all were layed out correctly, you'll see them being added in the terminal window.

Linux / Mac OSX

For Linux/OSX, make sure that you have ruby (preferably 1.9) installed, and that the two gems 'nokogiri' and 'sqlite3' are also installed (running *gem install nokogiri sqlite3* as administrator should do the trick). This is the point where aforementioned terminal is useful.

cd to your resources/scripts and run:

ruby romman.rb builddb --dbname ../arcandb.sqlite --rompath ../games --targetpath ../targets --mamegood --mameskipclone --mameshorttitle

Suffice it to say that there is a large number of arguments that this tool accepts, and while most of them are fairly obvious, note that [gamepath] and [romset] as part of an argument will expand into <resourcepath\games> and the 'setname' field for the game in arcan_db.c when launching a target. To expand upon this tool, look into <resourcepath\scripts\importers> and look at generic.rb, mame.rb and scummvm.rb for the general patterns on how, as the romman_base.rb is quite messy.

Command-line arguments

With any luck, you should now be able to relaunch the Welcome theme, and the # of games detected should correspond to what you placed in games\mame.

Now you have two sample themes are your disposal, *dishwater* and *space* (note: these will not launch unless there are games found in the database). Linux and Mac users also have a more esoteric 'internaltest' that demonstrates the basics of the internal launch mode.

When things work the way you want them to, there is a set of command-line arguments that can be

used to tweak general behavior regarding video, paths etc. for tighter integration with your project, for troubleshooting and testing different video resolutions etc. These are currently:

[Video Options]

-w, --width	*force specific width*
-h, --height	*force specific height*
-f, --fullscreen	*toggle full-screen mode, this might not work properly on all platforms versions and driver- combinations as it will also switch resolution to -w <width> -h <height>*
-s, --windowed	*set width/height to your desktop resolution (faked fullscreen)*
-m, --conservative	*reduce memory usage in favor of more disk I/O*
-r, --scalemode	*change the default (themes can override as well) for how textures are managed when they do not have a power-of-two dimension; 0 means that your graphics card supports non-power-of-two textures 1 means that images are forcibly scaled (become slightly less sharp) 2 means that we waste some memory and tweak texture coordinates*

[Input / Output options]

-p, --rpath	*set a specific resource- path*
-1, --stdout *filename*	*redirect stdout to filename (might be needed on windows)*
-2, --stderr *filename*	*redirect stderr to filename (might be needed on windows)*

Currently, the LUA API is sparsly documented, there is a small cheat-sheet of functions added in <resourcepath>/doc/cheatsheet.pdf