

---

Subject: : AmigaOS4

Topic: : SDL1 open issues

Re: SDL1 open issues

Author: : kas1e

Date: : 2019/8/26 8:22:01

URL:

@Capehill

I minimize SDL2 version till veery small test case , so, there is:

```
#include "IrrCompileConfig.h"
```

```
#ifdef _IRR_COMPILE_WITH_SDL_DEVICE_
```

```
#include "CIrrDeviceSDL.h"
```

```
#include "IEventReceiver.h"
```

```
#include "irrList.h"
```

```
#include "os.h"
```

```
#include "CTimer.h"
```

```
#include "irrString.h"
```

```
#include "Keycodes.h"
```

```
#include "COSOperator.h"
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include "SIrrCreationParameters.h"
```

```
#include <SDL2/SDL_syswm.h>
```

```
#include <SDL2/SDL_video.h>
```

```
namespace irr
```

```
{
```

```
///  
//! constructor
```

```
CIrrDeviceSDL::CIrrDeviceSDL(const SIrrlichtCreationParameters& param)
```

```
    : CIrrDeviceStub(param),
```

```
      ScreenWindow(0), ScreenTexture(0),
```

```
      MouseX(0), MouseY(0), MouseXRel(0), MouseYRel(0), MouseButtonStates(0),
```

```
      Width(param.WindowSize.Width), Height(param.WindowSize.Height),
```

```
      Resizable(false), WindowHasFocus(true)
```

```
{
```

```
    printf("ebaniy constuctor!\n");
```

```
    fflush(stdout);
```

```

if (SDL_Init( SDL_INIT_TIMER|SDL_INIT_VIDEO| SDL_INIT_NOPARACHUTE ) <0)
{
    os::Printer::log( "Unable to initialize SDL!", SDL_GetError());
    Close = true;
}

createWindow();

// create cursor control
CursorControl = new CCursorControl(this);

// create driver
createDriver();

if (VideoDriver)
    createGUIAndScene();

printf("ebaniy constructor done!\n");
fflush(stdout);
}

//! destructor
ClrrDeviceSDL::~ClrrDeviceSDL()
{
    printf("ebaniy descrutctor!\n");
    fflush(stdout);

    if (ScreenTexture)
    {
        SDL_DestroyTexture(ScreenTexture);
        ScreenTexture = NULL;
    }
    SDL_DestroyWindow(ScreenWindow);
    SDL_DestroyRenderer(ScreenRenderer);

    SDL_Quit();

    os::Printer::log("Quit SDL", ELL_INFORMATION);
}

bool ClrrDeviceSDL::createWindow()
{
    if ( Close )
        return false;

    int flags = 0;
    ScreenWindow = SDL_CreateWindow("Untitled", SDL_WINDOWPOS_UNDEFINED,

```

```

SDL_WINDOWPOS_UNDEFINED, Width, Height, flags);
ScreenRenderer = SDL_CreateRenderer(ScreenWindow, -1, 0);
ScreenTexture = SDL_CreateTexture(ScreenRenderer, SDL_PIXELFORMAT_ARGB1555,
SDL_TEXTUREACCESS_STREAMING, Width, Height);

return true;
}

///! create the driver
void ClrrDeviceSDL::createDriver()
{
switch(CreationParams.DriverType)
{
case video::EDT_SOFTWARE:
#ifdef _IRR_COMPILE_WITH_SOFTWARE_
VideoDriver = video::createSoftwareDriver(CreationParams.WindowSize, CreationParams.Fullscreen,
FileSystem, this);
#else
os::Printer::log("No Software driver support compiled in.", ELL_ERROR);
#endif
break;

default:
os::Printer::log("Unable to create video driver of unknown type.", ELL_ERROR);
break;
}
}

///! runs the device. Returns false if device wants to be deleted
bool ClrrDeviceSDL::run()
{
os::Timer::tick();

SEvent irrevent;
SDL_Event SDL_event;

while ( !Close && SDL_PollEvent( &SDL_event ) )
{
switch ( SDL_event.type )
{
case SDL_MOUSEMOTION:
irrevent.EventType = irr::EET_MOUSE_INPUT_EVENT;
irrevent.MouseInput.Event = irr::EMIE_MOUSE_MOVED;
MouseX = irrevent.MouseInput.X = SDL_event.motion.x;
MouseY = irrevent.MouseInput.Y = SDL_event.motion.y;
irrevent.MouseInput.ButtonStates = MouseButtonStates;
}
}
}

```

```

postEventFromUser(irrevent);
break;

case SDL_QUIT:
    Close = true;
    break;

default:
    break;
} // end switch

} // end while

return !Close;
}

//! pause execution temporarily
void ClrrDeviceSDL::yield()
{
    SDL_Delay(0);
}

//! pause execution for a specified time
void ClrrDeviceSDL::sleep(u32 timeMs, bool pauseTimer)
{
    const bool wasStopped = Timer ? Timer->isStopped() : true;
    if (pauseTimer && !wasStopped)
        Timer->stop();

    SDL_Delay(timeMs);

    if (pauseTimer && !wasStopped)
        Timer->start();
}

//! sets the caption of the window
void ClrrDeviceSDL::setWindowCaption(const wchar_t* text)
{
    core::stringc textc = text;
    SDL_SetWindowTitle(ScreenWindow, textc.c_str());
}

//! presents a surface in the client area
bool ClrrDeviceSDL::present(video::Image* surface, void* windowId, core::rect<s32>* srcClip)
{

```

```
if (SDL_UpdateTexture(ScreenTexture, NULL /* update whole texture */, surface->lock(), surface->getPitch  
()) != 0) {  
    // SDL_GetError  
}  
  
if (SDL_RenderCopy(ScreenRenderer, ScreenTexture, NULL, NULL) != 0) {  
    // SDL_GetError  
}  
  
SDL_RenderPresent(ScreenRenderer);  
return true;  
}  
  
//! notifies the device that it should close itself  
void ClrrDeviceSDL::closeDevice()  
{  
    Close = true;  
}  
  
//! return Pointer to a list with all video modes supported  
video::IVideoModeList* ClrrDeviceSDL::getVideoModeList()  
{  
}  
  
//! Sets if the window should be resizable in windowed mode.  
void ClrrDeviceSDL::setResizable(bool resize)  
{  
}  
  
//! Minimizes window if possible  
void ClrrDeviceSDL::minimizeWindow()  
{  
    if (ScreenWindow) {  
        SDL_MinimizeWindow(ScreenWindow);  
    }  
}  
  
//! Maximize window  
void ClrrDeviceSDL::maximizeWindow()  
{  
    if (ScreenWindow) {  
        SDL_MaximizeWindow(ScreenWindow);  
    }  
}
```

```

//! Restore original window size
void ClrrDeviceSDL::restoreWindow()
{
    // do nothing
}

//! returns if window is active. if not, nothing need to be drawn
bool ClrrDeviceSDL::isWindowActive() const
{
    return (WindowHasFocus && !WindowMinimized);
}

//! returns if window has focus.
bool ClrrDeviceSDL::isWindowFocused() const
{
    return WindowHasFocus;
}

//! returns if window is minimized.
bool ClrrDeviceSDL::isWindowMinimized() const
{
    return WindowMinimized;
}

//! Set the current Gamma Value for the Display
bool ClrrDeviceSDL::setGammaRamp( f32 red, f32 green, f32 blue, f32 brightness, f32 contrast )
{
    return false;
}

//! Get the current Gamma Value for the Display
bool ClrrDeviceSDL::getGammaRamp( f32 &red, f32 &green, f32 &blue, f32 &brightness, f32 &contrast )
{
    return false;
}

//! returns color format of the window.
video::ECOLOR_FORMAT ClrrDeviceSDL::getColorFormat() const
{
    return ClrrDeviceStub::getColorFormat();
}

} // end namespace irr

#endif // _IRR_COMPILE_WITH_SDL1_DEVICE_

```

As you can see there is almost no code left, but it works as expected on win32 (same speed as for sdl1, etc). On amigaos4 the same 1:1 code copied and compiled in, bring some madness-move of camera, without stop. Like i move mouse fast fast everywhere. At least visually.

Then, once i add check/set on "focus", i.e. to make run() be like this:

```
/// runs the device. Returns false if device wants to be deleted
bool CIrrDeviceSDL::run()
{
    os::Timer::tick();

    SEvent irrevent;
    SDL_Event SDL_event;

    while ( !Close && SDL_PollEvent( &SDL_event ) )
    {
        switch ( SDL_event.type )
        {
            case SDL_MOUSEMOTION:
                irrevent.EventType = irr::EET_MOUSE_INPUT_EVENT;
                irrevent.MouseInput.Event = irr::EMIE_MOUSE_MOVED;
                MouseX = irrevent.MouseInput.X = SDL_event.motion.x;
                MouseY = irrevent.MouseInput.Y = SDL_event.motion.y;
                irrevent.MouseInput.ButtonStates = MouseButtonStates;

                postEventFromUser(irrevent);
                break;
            case SDL_WINDOWEVENT:
                switch (SDL_event.window.event)
                {

                    case SDL_WINDOWEVENT_ENTER:
                    case SDL_WINDOWEVENT_FOCUS_GAINED:
                        printf("we got events about focus=true\n");
                        fflush(stdout);
                        WindowHasFocus = true;
                        break;

                    case SDL_WINDOWEVENT_LEAVE:
                    case SDL_WINDOWEVENT_FOCUS_LOST:
                        printf("we got events about focus=false\n");
                        fflush(stdout);
                        WindowHasFocus = false;
                        break;
                }
                break;

            case SDL_QUIT:
                Close = true;
        }
    }
}
```

```
break;

default:
    break;
} // end switch

} // end while

return !Close;
}
```

Then everything stops, and i have 1-2 fps. On win32, all works as before, without visuall change of course. With or without focus check/set it have the same fps. But on amigaos4 without check everything moves fast-fast, but with check, everything stops.

It looks like something with mouse / focus code in our sdl2 cause this. Maybe the fact that without "focus" value being checked/set everything just start moves like a madman, and that when i put back focus checks and it all start to be slow like hell, are the roots of the same issue ?

EDIT: interesting, that when i run that code with set/check on focus, but didn't set it, only printf's , then i can see that for win32, when i run app, it just says:

```
we got events about focus=true
we got events about focus=false
```

And nothing more.

But if i run it under amigaos4, its flood me with those entries without stop until i move mouse. Something pretty different there between our and win32 sdl2 ports.

As i say its just i have now 2 files (sdl1 ones and sdl2 ones), which i just copy to win32 vesion and to amigaos4 version. And with win32 version it all works, speed same in both versions. On amigaos4, sdl1 one ok, and sdl2 : when set on windowfocus used : 2 fps. When no set, then like i move mouse fast-fast in all the directions.

EDIT2: Oh, and i go pretty close now : i found that if i just change creating of sdl\_window be SDL\_FULLSCREEN one, then both bugs gone ! I.e. in fullscreen everything works and when i check/set focus, and when i didn't. In boch cases all renders fine, speed the same, and no flood of those "true/false". In other words, issue is with window mode only, and only on amigaos4 side (with the same 1:1 sdl2 code)