Subject: : AmigaOS4
Topic: : SDL1 open issues
Re: SDL1 open issues
Author: : kas1e
Date: : 2019/1/20 11:38:40
URL:

@Capehill
Thanks !

Found today some other issue (which can be again issues with game's code, but maybe not): i build some game via SDL1 / MiniGL which, when switch to/from fullscreen, or resize the gl window when in window mode, give a black window. Game logic works, but just black window.

If i start originally game in lets say fullscreen mode : it works.
If i start originalluy game in window mode: it works.

But once i trying to swith to/from window/fullscreen, then black window. Same for resizing when in window mode.

The functions in question are this ones:

```
/**************************************************************************
*      Initialize SDL and make a SDL-Window / Fullscreen                  *
**************************************************************************/

void sys_create_display(int width,int height)
{
  /* Information about the current video settings. */
  const SDL_VideoInfo* info = NULL;
  int vidmode_flags=0, samplingerror = 0;

  /* First, initialize SDL's video subsystem. */
  if( SDL_Init( SDL_INIT_VIDEO | SDL_INIT_TIMER | SDL_INIT_AUDIO) < 0 ) {
    fprintf( stderr, "Video or Audio initialization failed: %sn",
    SDL_GetError( ) );
    sys_exit(1);
  }

  sdl_on = 1 ;

  /* Let's get some video information. */
  info = SDL_GetVideoInfo( );
```

```
if( !info ) {
  /* This should probably never happen. */
  fprintf( stderr, "Video query failed: %sn",
  SDL_GetError( ) );
  sys_exit(1);
}

vidmode_bpp = info->vfmt->BitsPerPixel;

/*
 * Now, we want to setup our requested
 * window attributes for our OpenGL window.
 * We want *at least* 5 bits of red, green
 * and blue. We also want at least a 16-bit
 * depth buffer.
 *
 * The last thing we do is request a VMfloat
 * buffered window. '1' turns on VMfloat
 * buffering, '0' turns it off.
 *
 * Note that we do not use SDL_DOUBLEBUF in
 * the flags to SDL_SetVideoMode. That does
 * not affect the GL attribute state, only
 * the standard 2D blitting setup.
 */

 SDL_GL_SetAttribute( SDL_GL_RED_SIZE, 5 );
 SDL_GL_SetAttribute( SDL_GL_GREEN_SIZE, 5 );
 SDL_GL_SetAttribute( SDL_GL_BLUE_SIZE, 5 );
 SDL_GL_SetAttribute( SDL_GL_DEPTH_SIZE, 16 );
 if (SDL_GL_SetAttribute( SDL_GL_DOUBLEBUFFER, 1 ) <0) {
     fprintf(stderr, "SDL_GL_DOUBLEBUFFER error: %sn", SDL_GetError());
     options_vsync = 0;
 } else {
//compile without errors, if SDL is < Version 1.2.10 at compile time
#if SDL_MAJOR_VERSION == 1 && SDL_MINOR_VERSION == 2 && SDL_PATCHLEVEL > 9
// The next works only with fsaa options off!!!!
 if(!options_fsaa_value) {
     if(SDL_GL_SetAttribute( SDL_GL_ACCELERATED_VISUAL, 1 ) < 0 ) {
             fprintf( stderr, "Unable to guarantee accelerated visual with libSDL < 1.2.10n");
     }
 }
 if(vsync_supported()) {
     if(options_vsync) {
    if (SDL_GL_SetAttribute(SDL_GL_SWAP_CONTROL, 1) < 0) { // since SDL v1.2.10
      fprintf(stderr, "SDL_GL_SWAP_CONTROL error: %sn", SDL_GetError());
      options_vsync = 0;
    }
     }
 } else {
```

```
  fprintf(stderr,"SDL-System without control of vsync. Scrolling may stuttern");
 }
#endif
 }


 if(options_fsaa_value) {
  samplingerror = SDL_GL_SetAttribute( SDL_GL_MULTISAMPLEBUFFERS, 1);
  if(!samplingerror) {
    samplingerror = SDL_GL_SetAttribute( SDL_GL_MULTISAMPLESAMPLES,options_fsaa_value);
  }
 }
 if (samplingerror == -1) {

  options_fsaa_value = 0;

 }

 SDL_EnableKeyRepeat( SDL_DEFAULT_REPEAT_DELAY, SDL_DEFAULT_REPEAT_INTERVAL );
 /* key repeat caused problem when toggling fullscreen !!! */

 vidmode_flags = SDL_OPENGL;

 if ( info->hw_available ) {
  vidmode_flags |= SDL_HWSURFACE;
  vidmode_flags |= SDL_HWPALETTE; /* Store the palette in hardware */
 } else {
  vidmode_flags |= SDL_SWSURFACE;
 }

 if ( info->blit_hw ) { /* checks if hardware blits can be done */
  vidmode_flags |= SDL_HWACCEL;
 }
  if (fullscreen) {
    vidmode_flags |= SDL_FULLSCREEN;
  }else{
    vidmode_flags |= SDL_RESIZABLE;
 }

 //Set the window icon
 SDL_WM_SetIcon(SDL_LoadBMP("icon.bmp"),NULL);

 if(options_fsaa_value > options_maxfsaa) {
    options_fsaa_value = options_maxfsaa;
 }
 while (vid_surface == NULL) {

  if((vid_surface=SDL_SetVideoMode( width, height, vidmode_bpp, vidmode_flags )) == NULL) {
   if(!options_fsaa_value) {
    fprintf( stderr, "Video mode set failed: %sn", SDL_GetError());
    sys_exit(1);
```

```
  }

  fprintf( stderr, "Video mode set failed: %snSwitch to other moden", SDL_GetError());
  if(options_fsaa_value) {
    options_fsaa_value >>= 1;
    fprintf(stderr,"FSAA %in",options_fsaa_value);
    SDL_GL_SetAttribute(SDL_GL_MULTISAMPLESAMPLES, options_fsaa_value);
  } else {
    SDL_GL_SetAttribute(SDL_GL_MULTISAMPLEBUFFERS,0);
  }

 }

}
// Check new settings for better output the next line is a must for multisample
if(!samplingerror && options_fsaa_value){
  glEnable(GL_MULTISAMPLE);

//glHint(GL_MULTISAMPLE_FILTER_HINT_NV,GL_NICEST); //be careful (Nvidia-specific), set over an option
?
}

SDL_WM_SetCaption("game","game");

glPolygonMode(GL_FRONT,GL_FILL);  // fill the front of the polygons
glPolygonMode(GL_BACK,GL_LINE);   // only lines for back (better seeing on zooming)
glCullFace(GL_BACK);              // Standards for rendering only front of textures
glEnable(GL_CULL_FACE);
}

/*****************************************************************
 *                    Fullscreen active ?                        *
 *****************************************************************/

int sys_get_fullscreen(void)
{
  return fullscreen;
}

/*****************************************************************
 *          Set a fullscreen(1) or window(0) window              *
 * SDL_WM_ToggleFullScreen(screen) works only on X11 and there not stable *
 *****************************************************************/

void sys_fullscreen( int fullscr )
{

  SDL_Surface * screen;
  Uint32 flags;

  screen = SDL_GetVideoSurface();
```

```
    flags = screen->flags; /* Save the current flags in case toggling fails */
    SDL_EnableKeyRepeat( 0, 0 );
    if ( fullscr!=0 && (screen->flags & SDL_FULLSCREEN)==0 ){
        screen = SDL_SetVideoMode( 0, 0, 0, screen->flags | SDL_FULLSCREEN );
    } else if( fullscr==0 && (screen->flags & SDL_FULLSCREEN)!=0 ){
        screen = SDL_SetVideoMode( 0, 0, 0, screen->flags & ~SDL_FULLSCREEN);
    }
    if(screen == NULL) {
        screen = SDL_SetVideoMode(0, 0, 0, flags); /* If toggle FullScreen failed, then switch back */
    } else {
        fullscreen = fullscr;
    }
    SDL_EnableKeyRepeat( SDL_DEFAULT_REPEAT_DELAY, SDL_DEFAULT_REPEAT_INTERVAL );
    if(screen == NULL) {
        fprintf(stderr,"Video-Error on set full-screen/windowed mode. Terminatingn");
        sys_exit(1); /* If you can't switch back for some reason, then epic fail */
    }
}

/*****************************************************************************
 *         Toggle between Fullscreen and windowed mode                       *
 *****************************************************************************/

void sys_toggle_fullscreen( void )
{
    if (fullscreen){
        sys_fullscreen( 0 );
    } else {
        sys_fullscreen( 1 );
    }
}



/*****************************************************************************
 *               Resize the SDL Surface handle                               *
 *****************************************************************************/

void sys_resize( int width, int height, int callfrom )
{

    SDL_Surface * screen;
    Uint32 flags;

    if(width < 958) width = 958;      // don't resize below this
    if(height < 750) height = 750;
    ignore = callfrom;
    screen = SDL_GetVideoSurface();
    flags = screen->flags; /* Save the current flags in case toggling fails */
    SDL_EnableKeyRepeat( 0, 0 );
    screen = SDL_SetVideoMode( width, height, screen->format->BitsPerPixel, screen->flags);
```

```
    SDL_Delay(300);
    //fprintf(stderr,"Called x: %i y: %in",width,height);
    SDL_EnableKeyRepeat( SDL_DEFAULT_REPEAT_DELAY, SDL_DEFAULT_REPEAT_INTERVAL );
    if(screen == NULL) {
        screen = SDL_SetVideoMode(0, 0, 0, flags); /* If failed, then switch back */
    }
    SDL_EnableKeyRepeat( SDL_DEFAULT_REPEAT_DELAY, SDL_DEFAULT_REPEAT_INTERVAL );
    if(screen == NULL) {
        fprintf(stderr,"Video-Error on window resize. Terminatingn");
        sys_exit(1); /* If you can't switch back for some reason, then epic fail */
    }
    ResizeWindow(width,height);
}



/*****************************************************************
 *          get all resolution modes for SDL/OpenGL              *
 *****************************************************************/

sysResolution *sys_list_modes( void ) {
    sysResolution * sysmodes;
    SDL_Rect ** modes;
    int i, modenr;

    modes = SDL_ListModes(NULL, SDL_FULLSCREEN|SDL_HWSURFACE);
    for(i=0;modes[i];i++);
    modenr=i;
    sysmodes = (sysResolution *) malloc((modenr+1)*sizeof(sysResolution));
    for(i=0;modes[i];i++){
        sysmodes[i].w = modes[i]->w;
        sysmodes[i].h = modes[i]->h;
    }
    sysmodes[i].w=0;  /* terminator */
    sysmodes[i].h=0;  /* terminator */

    return( sysmodes );
}
```

(scroll till end for all of them).

Maybe its something with SDL_GetVideoSurface() ? Is looks like something with sys_resize() done wrong for us (or didn't done what we need to do). Maybe something with HWSURFACE there ?

On win32 same code works of course.

I use latest sld1 sources from repo.