UNIVERSITAT POLITÈCNICA DE VALÈNCIA

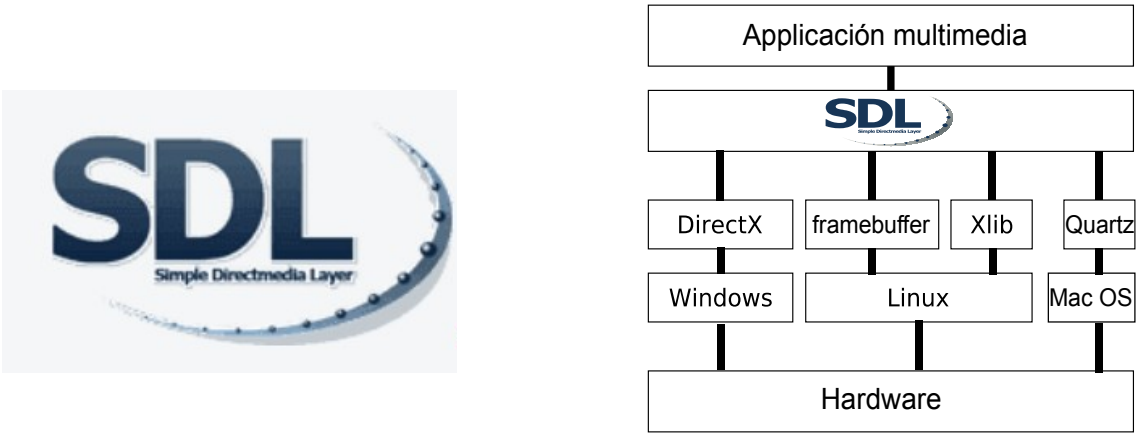
Ejemplos básicos de portabilidad de aplicaciones basadas en SDL: comparativa entre plataforma Switch, escritorio y 3DS

Apellidos, nombre	**Agustí i Melchor, Manuel** (magusti@disca.upv.es)
Departamento	**Departamento de Informática de Sistemas y Computadores (DISCA)**
Centro	Universitat Politècnica de València

1 Resumen de las ideas clave

A la hora de realizar una aplicación de características multimedia como un videojuego, hay que plantearse su **portabilidad**, esto es, si es posible llevarlo a cabo en una plataforma u otra. Se puede hablar de tres grupos de plataformas bastante diferentes: el computador de escritorio sobre los sistemas operativos (SO) Linux, macOS o Windows (y su correspondiente subsistema gráfico X11, Quartz o DirectX, respectivamente), las videoconsolas (con un SO generalmente propietario y cerrado) o los *smartphone (bajo Android, iOS, etc.)*. El uso de la biblioteca ***Simple DirectMedia Layer*** (SDL) **(**Figura 1a) permite realizar una versión portable, con **cambios** mínimos, en aplicaciones con características multimedia al ofrecer un nivel de abstracción (Figura 1b) común por encima de estas plataformas.

SDL [**1**] es una biblioteca gratuita de funciones, escrita en C, que ofrece un nivel común para diferentes SO de bajo consumo de recursos, con versiones para la mayoría de lenguajes de programación existentes y distribuida bajo licencia *zlib*. Fue creada por Sam Lantinga en 1998, mientras trabajaba en *Loki Software*, para realizar operaciones de carácter multimedia y multiplataforma. Fue diseñada para proporcionar acceso de bajo nivel a audio, entrada/salida (gestión de eventos producidos por teclado, ratón, *joystick*, etc. y acceso a ficheros), imágenes en 2D y con soporte hardware para 3D vía OpenGL, uso de temporizadores y sincronización basada en hilos. Conjuga operaciones de bajo nivel, con un alto nivel de portabilidad; lo que facilita el desarrollo multiplataforma, delegando en las interfaces nativas la implementación final de esas operaciones.



a) *b)*

Figura 1: SDL: (a) Logotipo de SDL y (b) Niveles de capa de abstracción del SO sobre el que se ejecuta una aplicación multiplataforma. Imágenes de <https://gbatemp.net/threads/release-sdl-3ds-1-2-15-simple-directmedia-layer-for-3ds.459291/> y <http://easy-learn-computer.blogspot.com/2012/01/learning-sdl-simple-directmedia-layer.html>.

Entre los SO soportados[1], en la versión oficial, se encuentran **Linux, macOS y Windows**. Y también de manera no oficial, esto es, realizadas por terceras partes, existen versiones para Nintendo **3DS** (N3DS o 3DS para abreviar) y Nintendo ***Switch*** (o, simplemente, *Switch*) entre otras. SDL, actual-

[1] La lista actual completa se puede ver en "SDL Wiki | Installing SDL" en la URL <https://wiki.libsdl.org/SDL2/Installation>.

mente, tiene dos grandes ramas y una tercera en camino[2]: SDL 1.2 (que está presente en el SDK no oficial para la 3DS) y SDL 2.0 (que lo está en el de la *Switch*).

La biblioteca de funciones SDL se compone de diferentes módulos, siendo SDL (la "standard library") el módulo básico (también llamado *core* o *kernel*) que se encarga de la gestión del modo de vídeo y los eventos de teclado y ratón. Sobre este pueden utilizarse otros como *SDL_image* (para soporte de formatos gráficos), *SDL_mixer* (formatos de audio, reproducción y mezcla), *SDL_net* (operaciones de transmisión en red) o *SDL_ttf* (uso de tipos de letra *TrueType*).

2 Objetivos

Partiremos de dos ejemplos de desarrollo *[2]* ya existentes en la plataforma *Switch* y veremos que es posible hablar de portabilidad y con qué modificaciones recompilarlos y ejecutarlos tanto en la plataforma 3DS, como en la del PC (el computador de escritorio).

Una vez que el lector haya revisado este artículo con detenimiento y explore el código que se adjunta, dispondrá de una referencia para plantear la portabilidad de una aplicación de videojuego que quiera llevar al computador de escritorio, a la 3DS o a la *Switch*. En particular, será capaz de:

- Analizar las diferencias de uso entre las tres plataformas con SDL.
- Plantear las modificaciones necesarias para desarrollar sobre SDL.
- Compilar y ejecutar las aplicaciones mostradas para el computador de escritorio sobre Linux, aunque sin pérdida de generalidad para otras plataformas de escritorio.

Para no extender la longitud del artículo se ha creado un repositorio en GitHub [3] donde se alojarán las versiones de código para las tres plataformas que se referencian en este artículo. El resto de este documento se centrará en exponer las diferencias propias de cada plataforma: primero exponiendo qué hacen los ejemplos de partida y, después, se pasará a ver qué cambios hay que introducir en el código original, así como la forma de compilación para obtener el ejecutable en cada una de las otras dos plataformas.

3 Introducción

Los dos ejemplos que exploramos los encontramos implementados para la plataforma *Switch* [*2*], son la referencia de partida. Ambos utilizan un modo de dibujado en pantalla, como se ha indicado en la Figura 1b, basado el acceso a través de un API, en este caso independiente del SO, sobre los diferentes SO actuales.

3.1 Ejemplo *sdl2-simple* en *Switch*

Este primer ejemplo (véase *Switch_SDL_2_0/sdl2-simple/main.cpp* en [3]), trabaja sobre la idea del doble *buffer*, modificando directamente cada píxel para dibujar lo que debe aparecer en pantalla. Como se muestra en la Figura 2a, la salida es sencilla: un fondo de color gris sobre el que se mueven

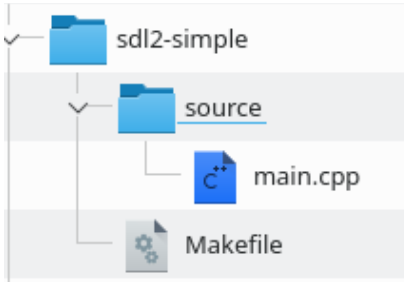
[2] Puede leer sobre esta futura versión en el sitio de Github: *Simple DirectMedia Layer (SDL) Version 3.0* <https://github.com/libsdl-org/SDL>.

en horizontal tres cuadrados de color rojo, verde y azul que, al llegar al extremo derecho de la ventana, vuelven a reaparecer en el lado izquierdo de la misma fila. Además, el código tiene en cuenta si el usuario:

- Presiona el botón A del primer *joy-con*[3] para cambiar la resolución de la pantalla de 1920x1080 a 1280x720 y viceversa.
- Presiona el botón '+' del primer *joy-con* para terminar la aplicación.

Como se ha adelantado, la aplicación utiliza la técnica del doble *buffer*, *esto es:* dibuja la escena completa en un *buffer*, mientras se utiliza otro para presentar la información en pantalla. Cuando se ha terminado de pintar todo el *buffer*, se intercambian los *buffers* y se vuelve a pintar en el otro *buffer* y, así, sucesivamente.

a)



b)

Figura 2: sdl-simple en la Switch: (a) captura de un instante de su ejecución y (b) contenido de directorio del proyecto.

El ejemplo original tiene la estructura que se muestra en la Figura 2b:

- Un solo fichero de código (de extensión ".cpp", pero podría ser perfectamente ".c", puesto que no se utiliza ninguna construcción propia de C++.
- Y un fichero *Makefile* que dirige la construcción del fichero a ejecutar. Cabe fijar la atención en el contenido de algunas variables que contiene este fichero:
  - ARCH, que en indica el juego de instrucciones que acepta la CPU para la que hay que generar el código; un ARM CORTEX A57 para la plataforma *Switch*.
  - CFLAGS, que indica dónde encontrar las cabeceras, especialmente las de SDL, que no son las básicas de la plataforma *Switch*.
  - LIBS, que indica la lista de bibliotecas de funciones a incorporar en el fichero ejecutable (la ROM en este caso) final.
  - LIBDIRS, que indica en qué directorios buscar las bibliotecas de funciones.

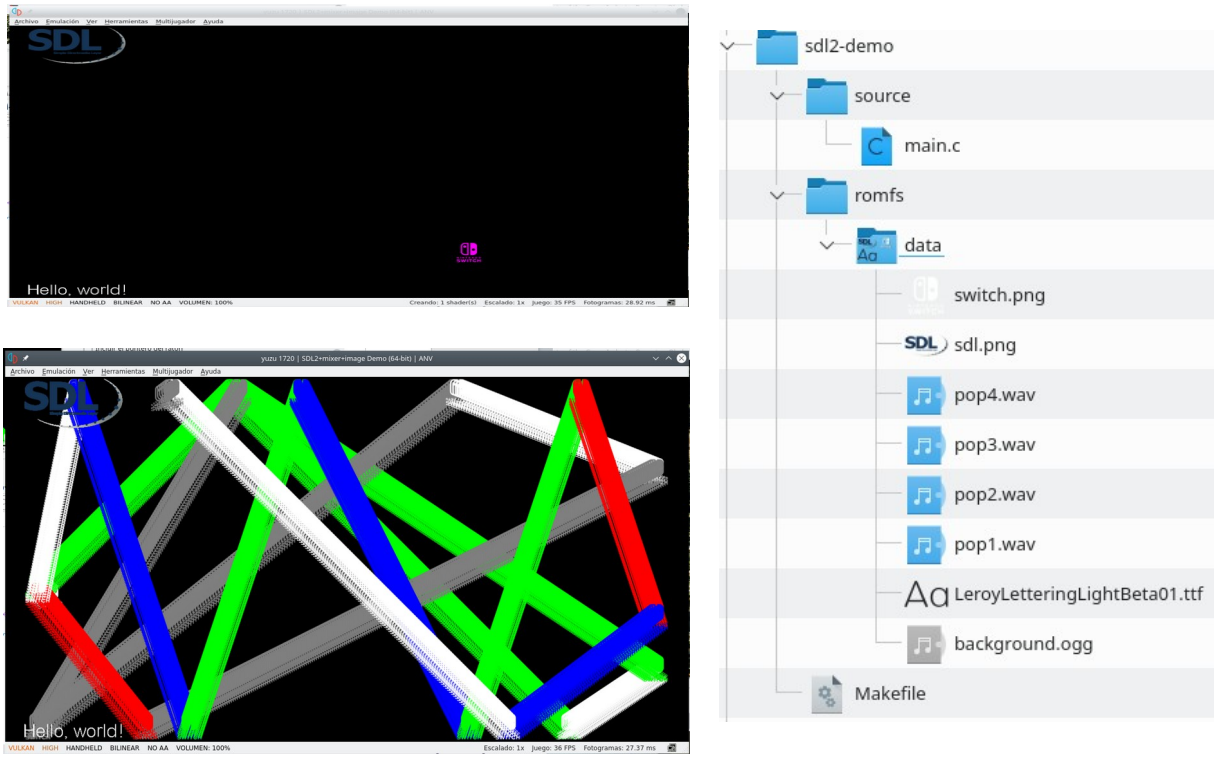
En los ejemplos de portabilidad al PC sobre Linux y a la 3DS se hablará de las variaciones en estas variables que se acaban de destacar.

3.2 Ejemplo *sdl2-demo* en *Switch*

Este ejemplo combina diferentes módulos de SDL: el soporte de audio (*SDL2_mixer*), de imagen (*SDL2_image*) y de tipos de letra (*SDL2_ttf*) por parte de SDL. Este segundo ejemplo (véase

[3] Los mandos de la *Switch* son vistos como *joysticks* en SDL.

Switch_SDL_2_0/sdl2-demo/main.c en [3]) trabaja sobre la idea del uso de la carga de *sprites* en tiempo de ejecución y cómo se consigue su movimiento por la pantalla. Además, acompañándolo de sonidos, tanto como efectos especiales (FX o pequeños fragmentos que se utilizan para remarcar eventos como colisiones), como de banda sonora (en el sentido de la música de fondo que acompaña en todo momento). Y, también, la utilización de tipos de letra *TrueType* para escribir texto en pantalla.



a) *b)*

Figura 3: sdl2-demo en la Switch: (a) dos instantes del ejemplo en funcionamiento y (b) contenido del directorio del proyecto.

Como se muestra en la Figura 3a, la salida gráfica está compuesta por un *sprite* (el del logo de SDL, con fondo transparente) que permanecerá en las mismas coordenadas durante toda la aplicación; otro (el del logo de Switch, también con fondo transparente) que va a ir cambiando de posición siguiendo una trayectoria rectilínea que, en alcanzar uno de los bordes de la pantalla se le hace cambiar de sentido para que siga "botando" dentro de la ventana. Y se acompaña de un cambio de color aleatorio (entre uno de los siete definidos en el código) y un sonido que es diferente en función de si el "rebote" se produce contra uno u otro de los límites de la pantalla. Además, el código tiene en cuenta los siguientes eventos de la interacción con el usuario:

- Si presiona el botón de flecha hacia arriba del primer *joy-con* se decrementará la variable *wait*, y si lo hace con el de flecha hacia abajo la incrementará. Este valor indica cuánto se esperará para generar el siguiente cuadro, con lo que a mayor valor de esta variable más lento es el proceso de avance del *sprite*.
- Si presiona el botón '+' del primer *joy-con*, terminará la aplicación.
- Si presiona el botón 'B' del primer *joy-con* se invierte el valor de *trail*. Este indicará si se limpia por dónde se va moviendo el *sprite* o no, dejando ver la estela de su paso.

En cuanto a la estructura de este proyecto, se descompone como se muestra en la Figura 3b en varios archivos organizados en diferentes subdirectorios:

- Un directorio *source* con el único fichero de código (de extensión ".c").
- Un directorio *romfs* que contiene un subdirectorio *data* con los ficheros de medios que se van a utilizar. Por un lado los dos PNG (que se utilizarán de texturas para los *sprites)*, también contiene los cuatro WAVE (para el sonido de los rebotes) y el fichero de música OGG. Además incluye el fichero TTF[4] para el tipo de letra que se utiliza.
- Y un fichero *Makefile*, para el que son válidos los comentarios que se han hecho en el primer ejemplo.

4 Desarrollo

En este punto se abordan los detalles de las modificaciones que es necesario llevar a cabo para portar los ejemplos mencionados tanto al PC sobre Linux, como a la 3DS.

4.1 Desarrollo en plataforma 3DS

Para portar las aplicaciones a esta plataforma hay que realizar cambios respecto a la versión de la plataforma *Switch*, tanto por la versión de SDL disponible como por las diferencias de hardware entre ambas plataformas:

- En primer lugar el hardware es muy diferente, empezando por que la 3DS tiene dos pantallas, por lo que la inicialización del modo de vídeo es diferente. Por simplicidad del código se ha optado por utilizar solo la pantalla superior. Además, la 3DS no tiene los mismos mandos que la *Switch*, por lo que la gestión de eventos habrá de ser remapeada a los controles que están aquí disponibles.
- En segundo lugar, la versión de la SDL disponible en la 3DS es la 1.2.15 [4]. En esta versión, la diferencia más importante es que solo se permite utilizar aplicaciones a pantalla completa, no en modo ventana.
- En tercer lugar, el *Makefile*. Hay que indicar al compilador el juego de instrucciones (dependiente de la familia de procesador utilizado) con el que ha de generar el código binario final y los *flags* para utilizar la versión de SDL disponible.

4.1.1. Ejemplo sdl-simple en 3DS

El Listado 1 resume los cambios que ha sido necesario realizar para que se puedan localizar en el código y que son debidos a:

1. Cambio en inicialización de módulos necesarios, no hay *joystick* como en *Switch*.
2. Resolución de la "pantalla". Hay que adaptarla al caso de 3DS.
3. Se han comentado las líneas de gestión de los eventos relativos a los *joystick* y el botón '+' del que no dispone la 3DS.

[4] Puede consultar en la Wikipedia sobre los tipos de letra *TrueType* <https://en.wikipedia.org/wiki/TrueType>.

4. Se ha cambiado el modo de pintado del fondo de pantalla en color gris, mediante la mecánica que ofre SDL 1.2 con un rectángulo de las dimensiones de la pantalla superior (*SDL_FillRect*).

5. Dibujado de los cuadrados de color. Ídem que el cambio 4.

6. Refresco de la pantalla con las funciones de SDL 1.2 (*SDL_UpdateRect*).

7. Liberar los recursos asociados a la pantalla con las instrucciones de SDL 1.2 (*SDL_FreeSurface*).

```
// (1) Cambio en inicialización de módulos necesarios
    if (SDL_Init(SDL_INIT_VIDEO ) < 0) {
//         SDL_Log("SDL_Init: %s\n", SDL_GetError());
        return -1;
    }

// (2) Cambio en resolución de "pantalla
    w=400;
    h=240;
    // Initialize the display, requesting a software surface
    pantallaSuperior = SDL_SetVideoMode(w, h, 8,
                       SDL_TOPSCR | SDL_CONSOLEBOTTOM | SDL_SWSURFACE);
    if ( pantallaSuperior == NULL ) { exit(1); }

// (3) Comentar/Quitar las líneas de gestión eventos del HW Switch
...
// (4) Pintar el fondo
    SDL_Rect f = {0, 0, w, h};
    SDL_FillRect(pantallaSuperior, &f,
                 SDL_MapRGB(pantallaSuperior->format, 111, 111, 111) );
...
// (5) Dibujado de los cuadrados de color
   SDL_FillRect(renderer, &r, SDL_MapRGB(renderer->format, 255, 0, 0));
   SDL_FillRect(renderer, &g, SDL_MapRGB(renderer->format, 0, 255, 0));
   SDL_FillRect(renderer, &b, SDL_MapRGB(renderer->format, 0, 0, 255));
...
// (6) Refresco de la pantalla
    SDL_UpdateRect(pantallaSuperior, 0, 0, w, h);
...
// (7) Liberar recursos de la pantalla
    SDL_FreeSurface( pantallaSuperior );
```

Listado 1: Cambios en el código del ejemplo sdl-simple (main.cpp) en 3DS.

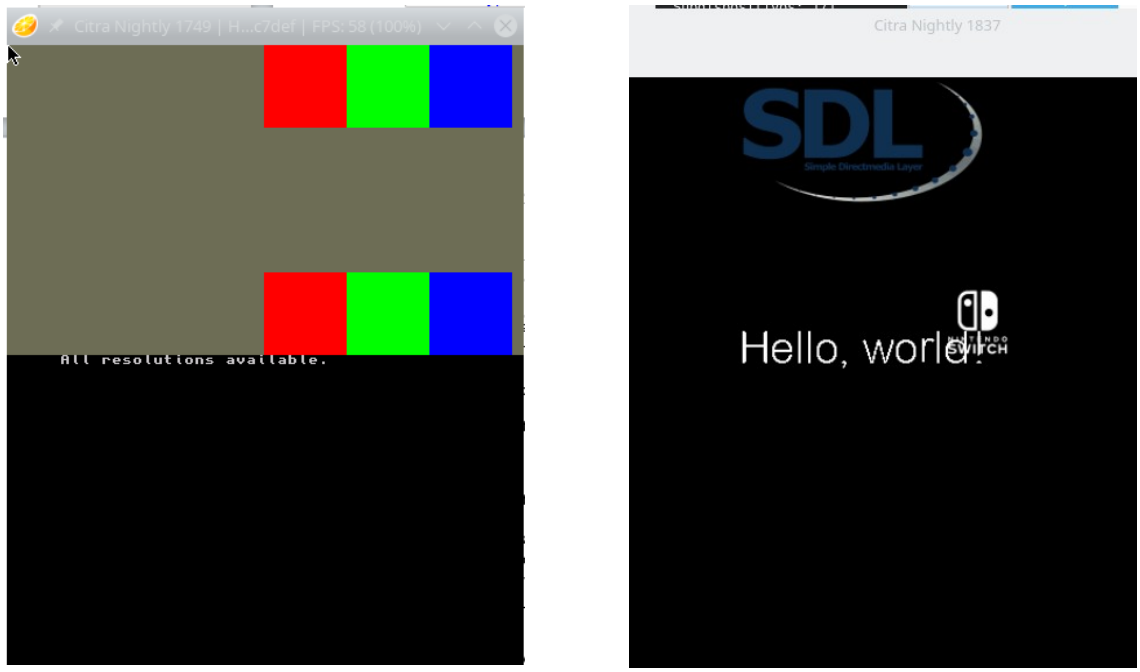
La Figura 4a muestra el resultado obtenido con este ejemplo. Aunque la pantalla inferior no se ha utilizado para replicar la funcionalidad del ejemplo, se ha utilizado para mostrar texto describiendo el avance de la aplicación y que se vea cómo se puede utilizar una de las pantallas, en este caso, en modo texto.

Y, en cuanto al Makefile, hay que modificar dos líneas que deben quedar de esta forma:

```
LIBS := `sdl-config --libs` -lctru -lm
```

y

```
LIBDIRS := $(PORTLIBS) $(CTRULIB)
```



a) *b)*

Figura 4: Resultado de la ejecución de los ejemplos portados a la 3DS (a) sdl-simple en N3DS con SDL 1.2 y (b) sdl-demo en N3DS con SDL 1.2.

4.1.2. Ejemplo sdl-demo en 3DS

La Figura 3b muestra el resultado obtenido en este ejemplo, donde también se ha utilizado la pantalla inferior para mostrar texto durante las pruebas de la implementación.

EL Listado 2 resume los cambios que ha sido necesario realizar debidos a:

1. Cambio en las cabeceras de la plataforma, porque utiliza la función *appletMainLoop* (Switch) vs *aptMainLoop* (3DS), que podría haber prescindido el ejemplo original. Así como las declaraciones de eventos propios de los *joy-con* que no están en la 3DS, la resolución de la "pantalla" y la frecuencia de uso de audio por limitación de la 3DS.
2. Cambiar el uso de *SDL_Renderer* y *SDL_Texture* por SDL_Surface, debido a que la versión de SDL para 3DS todavía no hace diferencias entre esos dos tipos.
3. Cambiar el uso del módulo de soporte para los *joy-con* (*joystick*), así como la gestión de los eventos relativos a los botones de la 3DS y, en particular, el botón '+' del que no dispone la 3DS.
4. Se ha cambiado el modo de pintado del fondo de pantalla en color gris, mediante la mecánica que ofre SDL 1.2 con un rectángulo de las dimensiones de la pantalla superior (*SDL_FillRect*) y el dibujado de los sprites (*SDL_BlitSurface)*.
5. Refresco de la pantalla (SDL_Flip) y liberar los recursos asociados a la pantalla (*SDL_FreeSurface*) con la mecánica de SD 1.2.

```
// (1) Cabeceras, resolución y gestión de eventos de la plataforma
// y propiedades del audio
#include <3ds.h>
…
// #define SCREEN_W 1280
#define SCREEN_W 320
// #define SCREEN_H 720
#define SCREEN_H 240
…
 while (!exit_requested && aptMainLoop() )
…
    Mix_OpenAudio(22050/2, AUDIO_S16, 2, 2048);
,,,
 // (2) SDL_Renderer/SDL_Texture → SDL_Surface
    SDL_Surface *screen;
    screen = SDL_SetVideoMode(SCREEN_W, SCREEN_H, 16,
                    SDL_TOPSCR | SDL_CONSOLEBOTTOM | SDL_HWSURFACE);

    SDL_Surface * render_text(SDL_Surface *renderer, ...
..
    SDL_BlitSurface(sdllogo, NULL, screen, NULL);

// (3) Cambio en inicialización de módulos necesarios
//     SDL_InitSubSystem(SDL_INIT_JOYSTICK);
//     SDL_JoystickEventState(SDL_ENABLE);
//     SDL_JoystickOpen(0);
…
//     if (event.type == SDL_JOYBUTTONDOWN) {
...
// (4) Pintar el fondo
    SDL_Rect f = {0, 0, w, h};
    SDL_FillRect(pantallaSuperior, &f,
                SDL_MapRGB(pantallaSuperior->format, 111, 111, 111) );
...
// (5) Refresco de la pantalla y Liberar recursos
    SDL_UpdateRect(pantallaSuperior, 0, 0, w, h);
...
    SDL_FreeSurface( screen );
```

Listado 2: Cambios en el código del ejemplo sdl-demo (main.c) en 3DS.

Y en el Makefile, hay que añadir estas referencias (todo en una misma línea):

```
LIBS := -lSDL_image -lpng -lz -ljpeg -lSDL_mixer -lSDL_ttf
-lfreetype -lbz2 -lSDL_mixer -lmikmod  -lvorbisidec -logg -lmad `sdl-
config --libs` -lctru -lm
```

4.2 Desarrollo en el computador de escritorio

En esta plataforma, en este caso el PC sobre Linux, respecto a la inicial de *Switch*, hay que tener en cuenta que hay diferencias importantes en cuanto al hardware, especialmente porque no suelen haber dos *joysticks* conectados. Así que se remapearán los eventos de la botonera a teclas del teclado. Además, en el computador pueden coexistir la versión 1.2 y la 2.0 de SDL, por lo que se comentará las diferencias respecto a la versión para 3DS (SDL 1.2.15) y la de *Switch* (SDL 2.0.14).

Una consideración aparte: solo se mostrarán capturas sobre la versión 2.0. Esto es así porque la versión 1.2 asume que hay una sola pantalla, que las aplicaciones trabajan a pantalla completa (no en ventanas) y no es posible hacer capturas de pantalla con ese modo de vídeo. Así que, para poder probar los ejemplos se ha de instalar al menos una de las dos versiones, según el interés del lector. El lector puede comprobar si tiene instalada alguna versión, en Linux, desde un terminal con la orden:

```
$ apt search libsdl
```

Lo que mostrará qué paquetes con ese prefijo aparecen en la lista de disponibles y si están instalados o no. A partir de ahí se puede proceder a instalar una o ambas versiones de desarrollo, la propia instalación incorporará todo lo necesario, incluyendo posibles dependencias que sean necesarias. Esto permitirá construir un ejecutable referenciando y enlazando con la versión de SDL que se desee. En función de la versión que se quiera instalar, se utilizará una de las dos órdenes siguientes y, se comprobará la instalación con la orden *sdl-config*. *Veamos las* secuencias de órdenes para un caso u otro, incluyendo la comprobación del número de versión instalado:

```
$ sudo apt-get install libsdl1.2-dev
$ sdl-config --version

1.2.15
```

o

```
$ sudo apt-get install libsdl2-dev
$ sdl2-config  --version
2.0.10
```

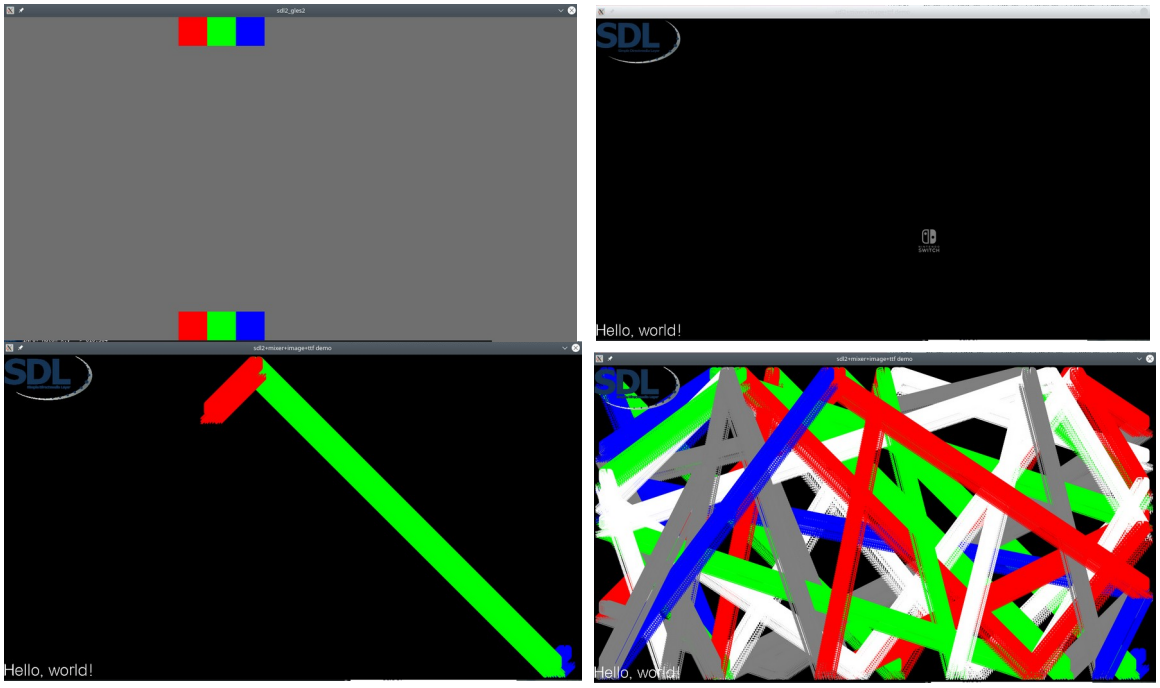


Figura 5: sdl-simple (arriba a la izquierda) y sdl-demo (arriba a la derecha y abajo)ejecutándose sobre el PC bajo Linux y con SDL 2.0.

En el caso de los dos ejemplos examinados, los cambios en el código se reducen más que en la 3DS, por lo que se sugiere comprobarlo directamente, examinando el código. En el caso de *sdl-simple*, solo es necesario remapear eventos de la botonera a acciones del teclado (porque no tenemos los *joy-con* emparejados al computador, en cuyo caso no hubiera sido necesario). Para ejecutar el ejemplo *sdl-simple* (Figura 5, arriba a la izquierda), se habrá de compilar con la orden:

```
$ gcc sdl2-simple_linux.cpp -o sdl2-simple_linux `pkg-config sdl2 --cflags
--libs`
```

Y en el caso de *sdl-demo (Figura 5, arriba a la derecha y abajo izquierda y derecha)*, también mínimos cambios como quitar la referencia a la cabecera *switch.h*, las instrucciones para acceder al sistema de archivos en la tarjeta de memoria (*romfsInit/romfsExit* y chdir), remapear acciones al teclado y el *appletMainLoop* del bucle principal. Para compilarlo se ejecutará:

```
$ gcc sdl2-demo_linux.cpp -o sdl2-demo_linux `pkg-config sdl2  --cflags --libs` `pkg-config SDL2_mixer SDL2_image SDL2_ttf --cflags --libs` && sdl2-demo_linux
```

5 Conclusión y cierre

A lo largo de este objeto de aprendizaje hemos visto las diferencias que hacen posible hablar de portabilidad de aplicaciones basadas en SDL entre plataformas como el PC, la 3DS y la *Switch*. Como ha podido comprobar el lector, las diferencias entre en las tres plataformas comparadas en este estudio son obligadas por la diferencia que hay entre el hardware de las tres. En el caso de plataformas más similares en hardware, los cambios se reducen y, si además se utiliza la misma versión de SDL en ambas, el número de cambios puede llegar a ser muy próximo a cero.

Ahora, con estas reflexiones en mente, es cuestión de proponerse un ejemplo propio. Aprovechando lo que se muestra aquí (y el código de los ejemplos [3]) se puede comenzar un Pong[5] o un juego de damas[6]. Ambos se pueden plantear en local o en red (el módulo *SDL_net*, da soporte a operaciones de transmisión en red). ¿Te animas, estimado lector?

Bibliografía y referencias

[1] SDL. Sitio web. <https://www.libsdl.org/>.

[2] devkitPro / Switch examples. <https://github.com/switchbrew/switch-examples>.

[3] Repositorio de los ejemplos de este artículo <https://github.com/magusti/3DS/portabilidad-SDL_PC_3DS_Switch

[4] Wenting Zhang (zephray). Repositorio de SDL-1.2-N3DS, para <https://github.com/zephray/SDL-1.2-N3DS>.

[5] Puede ver los detalles del juego en <https://es.wikipedia.org/wiki/Pong>.

[6] Este juego lo puede encontrar descrito en <https://es.wikipedia.org/wiki/Damas>.