FIB Facultat d'Informàtica de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

CONCEPTES AVANÇATS DE SISTEMES OPERATIUS
Departament d'Arquitectura de Computadors

SDL

La alternativa "Open Source" para Linux

(Seminaris de CASO)

Autors

Samuel Casanova Romero
Marc Sabat Olivé

Qué es?

- **Simple DirectMedia Layer**

⇒SDL es una API de desarrollo libre (open source) multimedia y multiplataforma en C/C++

⇒Se puede ejecutar en las siguientes plataformas:

- UNIX/Linux
- Win32
- MacOS
- BeOS
- PS2/DreamCast

Código Abierto
Cualquiera puede acceder al código fuente y modificarlo.

Qué me permite hacer?

- **Simple DirectMedia Layer**

⇒Se utiliza para:

- Juegos
- Demos
- Emuladores
- Aplicaciones Multimedia

Qué me permite hacer? (2)

- **Simple DirectMedia Layer**

⇒Con SDL se puede controlar:

- Eventos
- Audio y Video
- CD-ROM
- Hilos de ejecución
- Temporizadores

Qué me permite hacer? (3)

- **Gestión de Eventos**

→Control del teclado, ratón y demás dispositivos de E/S

→Los eventos se habilitan por separado gracias a la función SDL_EventState()

→Los eventos se filtran antes de mandarse a la cola de eventos

→La cola de eventos se gestiona mediante hilos de ejecución

Qué me permite hacer? (4)

- **Gestión de Audio**

→Reproducción de Audio en 8 y 16 bits, mono y stereo, con conversión de formato si el hardware no soporta el formato

→El sonido se ejecuta en un hilo independiente

→El sonido se mezcla utilizando una librería aparte disponible en los ejemplos de utilización de SDL

→SDL dispone de la API completa de control de audio del CD-ROM

Qué me permite hacer? (5)

- **Gestión de Video**

→Al igual que el Audio, podemos establecer el formato de Video. SDL se encarga de convertirlo en caso de que el hardware no soporte dicho formato

→Podemos escribir directamente sobre el búfer gráfico

→Es posible utilizar aceleración hardware si el equipo lo hace posible

→En la arquitectura x86, es posible optimizar los gráficos mediante MMX

Qué me permite hacer? (6)

- **Gestión de Hilos**

→SDL dispone de un API completa de creación y gestión de hilos simples de ejecución

→La sincronización de los hilos se puede controlar mediante semáforos binarios simples

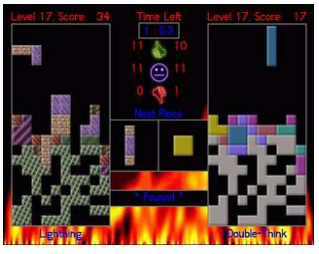
Qué me permite hacer? (7)

- **Gestión de Temporizadores**

→Permite obtener el número de milisegundos transcurridos desde el inicio de la ejecución

→También permite la programación de alarmas y funciones de wait() para los hilos de ejecución con precisión de milisegundos

Ejemplos de utilización:



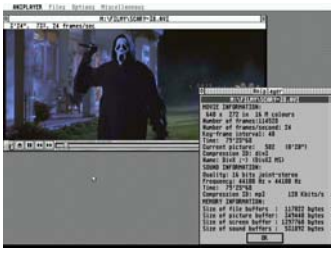
Juegos:

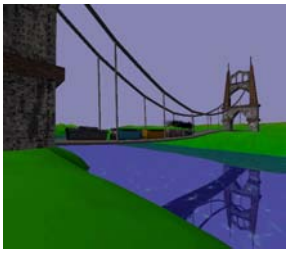
-Abe's Amazing Adventure

-Andromeda 9

-Alizarin Tetris

Ejemplos de utilización: (2)





Aplicaciones, emuladores y demos:

-Ani Player

-Final Burn (SN Emulator)

-Pontifex 2

Otras librerías gráficas

- **OpenGL**

⇒Creada por Silicon Graphics en 1992, pensada para desarrollar aplicaciones multiplataforma sin cambiar el código

⇒La primera solución y la más extendida, los primeros juegos 3D (Doom y Wolfestein) se hicieron utilizando esta librería

⇒También pensada para evolucionar con facilidad junto al desarrollo constante de hardware.

Otras librerías gráficas (2)

- **Allegro**

⇒Enfocada exclusivamente al mundo de los videojuegos

⇒La más fácil de manejar de todas, tiene efectos preprogramados. Por contra, la más limitada en cuanto a funciones avanzadas de desarrollo

⇒Su uso no está muy extendido por los programadores profesionales

Otras librerías gráficas (3)

- **DirectX**

⇒Creada por Microsoft en 1995 para desarrollar aplicaciones multimedia para plataformas Windows

⇒Muy extendida gracias a la potencia de sus funciones que acceden de forma directa a todo el hardware multimedia de la computadora, y a la eficiencia de la parte de Direct3D

⇒DirectX no es portable a otras plataformas

Otras librerías gráficas (4)

- **WineX**

⇒Se trata de un emulador para correr aplicaciones Win32 con DirectX en sistemas operativos bajo Linux

⇒Basado en el emulador Wine

⇒Todavía en fase de desarrollo, pretende incorporar todas las APIs DirectX, incluida Direct3D, a la plataforma Wine

PRÀCTICA

PRÁCTICA

Ejemplo de aplicación de la librería SDL

PRÁCTICA: OBJETIVOS

- Objetivos:
  - Comprobar el grado de portabilidad
  - Rendimiento en distintas plataformas (Win32 / Linux)
  - Complejidad del código (SDL vs. DirectX)

PRÁCTICA: PORTABILIDAD

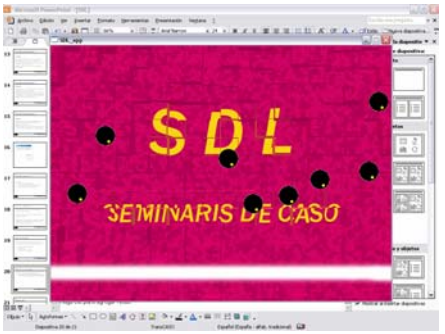
- Se inició el desarrollo en Linux (usando KDevelop como IDE y G++ com compilador).
- Una vez finalizado el desarrollo se intentó compilar bajo windows…y funcionó (usando Visual C++ 6.0).
- Resultado: No hubo que modificar ninguna línea de código. Bastó configurar correctamente el compilador para que apuntara a las librerías pertinentes.

PRÁCTICA: PRUEBAS

- Se sometió la aplicación a pruebas en Win32 y Linux en una máquina con las siguientes características básicas:
  - Procesador Intel Pentium IV 2'2Ghz
  - 512 Mb 333 Mhz
  - Tarjeta gráfica GForce 4 MX440
  - Sistemas Operativos: Linux (RedHat 8.0) / WinXP Professional

PRÁCTICA: La aplicación

- La aplicación es un sencillo programa que permite mover un número variable de "sprites" sobre un fondo.
  - Para comparar el rendimiento se han hecho pruebas en Win32 y Linux variando el número de "sprites" que se muestran a la vez.

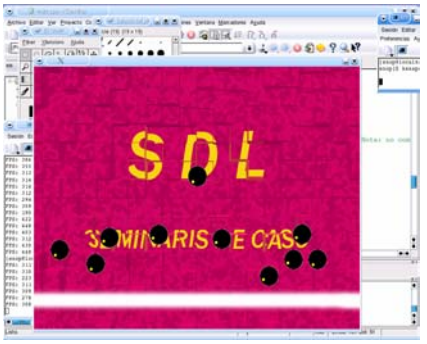


PRÁCTICA: La aplicación (2)

- Qué partes se han probado de SDL:
  - Gráficos en 2D: se ha usado una librería propia para la gestión de los sprites (adaptada de un tutorial).
  - Temporizadores: el uso de los temporizadores proporcionados por SDL ha sido muy sencillo.

PRÁCTICA: Compartiva Linux / Win32

- El movimiento con pocos sprites es fluido tanto en windows como en Linux. Ahora bien, fijémonos en los fps (frames per second):
  - Windows, 10 sprites: 84 fps
  - Linux, 10 sprites: 333 fps



PRÁCTICA: Comparativa Linux / Win32

- Obtenemos muchos mas fps en Linux que en windows. Windows fija los fps en función de la frecuencia de refresco del monitor.

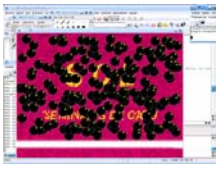
- En principio no tiene sentido superar la frecuencia de refresco, sin embargo, cuando aumentamos el número de sprites…

PRÁCTICA: Comparativa Linux / Win32

Prueba con 200 sprites:

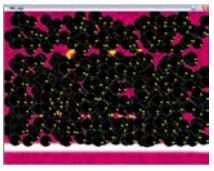
-Windows: 65 fps

- Linux: 85 fps

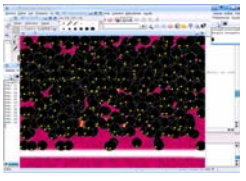


Prueba con 700 sprites:

- Windows: 30 fps



- Linux: 30 fps



PRÁCTICA: Comparativa Linux / Win32

- **En nuestra aplicación las diferencias se hacen notables a partir de unos 100 sprites:**

  - A pesar de hacer ciertas indagaciones no hemos podido corregir el problema.

  - Puede que no suponga un problema para la mayoría de aplicaciones pues, como vemos, el número de sprites mostrados es suficiente para las aplicaciones habituales (incluso juegos).

  - A partir de unos 500 sprites se iguala el rendimiento llegando a unos 40 fps.

  - En general, viendo la ejecución de la aplicación, en linux se aprecia mas fluidez en todos los casos.

PRÁCTICA: Complejidad de código

- Como indica su nombre (SIMPLE Directmedia Layer) el código resulta corto y fácil de entender.
- Tal como su programador (Sam Lantigna) admite: "No todo lo que se puede hacer en DirectX se puede hacer en SDL…aún".
- Sin embargo (sin siquiera contar la portabilidad) la sencillez, elegancia y claridad del código en SDL son sus puntos a favor cuando no necesitamos usar características avanzadas del hardware (como por ejemplo multimonitor).

PRÁCTICA: Complejidad de código (2)

- Ejemplo. Veamos los pasos básicos en SDL para cargar el fondo de pantalla y veremos su sencillez:

  -Inicializamos la pantalla:

  ```
  screen = SDL_SetVideoMode(640, 480, 16, SDL_SWSURFACE);
  ```

  - Cargamos la imagen:

  ```
  image = SDL_LoadBMP(file);
  ```

  -La mostramos:

  ```
  SDL_BlitSurface(image, NULL, screen, &dest);  //Vuelca las superf. en pant.
  SDL_UpdateRects(screen, 1, &dest); //Actualiza la porción de pantalla
  ```


PRÁCTICA: Conclusiones (DirectX vs. SDL)

	SDL	DirectX
Portabilidad	*Excelente*	*Nula*
Rendimiento	*Bueno / Aceptable*	*Muy bueno*
Aprovechamiento del hardaware	*Aceptable*	*Excelente*
Complejidad de código	*Excelente*	*Bueno*

Bibliografía

- http://www.libsdl.org
- http://sdl.euskal-linux.org/guia/
- http://www-106.ibm.com/developerworks/library/l-making-linux-fun/
- http://vjuegos.cem.itesm.mx/de/donde_empezar04.html
- http://www.lacompu.com/soporte/boletines/directx/index.php3
- http://alleg.sourceforge.net/
- http://linux.bankhacker.com/software/WineX/