TP2: Interfaz Gráfica y Simulación

Taller de Programación I - Curso Ing. Diego Azcurra - 2do cuatrimestre 2016

Objetivo

Desarrollar un programa que muestre una pantalla similar a la del juego Metal Slug (https://youtu.be/f00mTuq3cA0). Esta demo permitirá la participación de múltiples jugadores que podrán mover un soldado del escuadrón de Halcones Peregrinos.

Desarrollo

Se deberá realizar un cliente y un servidor en C/C++.

1. **Servidor**: Lee desde un archivo la definición de un escenario. El servidor recibe un único argumento opcional por línea de comando que indica el path del archivo de configuración de escenario. Si se omite este parámetro se carga un escenario por defecto.

2. **Cliente**: Se conecta al servidor y muestra dicho escenario gráficamente en una pantalla. El escenario estará compuesto por:

    a. Un fondo compuesto por dos o más imágenes.
    b. Un protagonista animado que sea controlado por cada jugador.

Podrán participar múltiples jugadores simultáneamente en red. Todos pueden interactuar simultáneamente moviendo sus soldados por la pantalla. Asimismo, todos los participantes pueden ver en tiempo real los movimientos de los soldados de los otros jugadores.

Red

El sistema consistirá de un servidor y varios clientes que se conectan a él. Queda a criterio de cada grupo si el servidor y los clientes son programas separados, o un mismo ejecutable que puede asumir los distintos roles. Cabe aclarar que de optar por esto último, un cliente no puede ser simultáneamente cliente y servidor en la misma ejecución.

El servidor define el escenario: al iniciar el servidor se especifica el archivo XML con la configuración del escenario y, por lo menos en este trabajo, el mismo no cambiará en ningún momento. Si el archivo no existe, se utilizará un escenario por defecto.

Una vez cargado el escenario se ejecuta el nivel. Por ser este un demo, el nivel no termina nunca y los participantes no pueden morir o perder.

El servidor no tendrá una interfaz gráfica. Cada escenario tendrá un máximo de jugadores que está especificado en el archivo de configuración. Alcanzado ese límite, el escenario no admite más jugadores. Cada jugador al conectarse debe especificar un nombre de usuario y servidor al que desea conectarse (ip y puerto). Si ese servidor ya tenía un usuario con ese nombre y el mismo no se encuentra conectado, entonces se acepta el nuevo jugador asignándole el estado guardado para dicho nombre. Si el servidor no contenía ese nombre de usuario y aún queda espacio disponible para un nuevo jugador, se crea el personaje y el jugador entra al escenario.

Cuando un jugador se desconecta del escenario su personaje se mostrará grisado y se indicará en un mensaje a los demás que el jugadores se ha desconectado (se puede usar la consola). Es indistinto si el jugador se desconecto por una falla de red o porque voluntariamente salió del programa. En todo momento cada jugador puede ver los movimientos de otros jugadores en tiempo real.

Handshake

Las dimensiones de ventana en el cliente se expresa en píxeles y puede ser fija.

El primer mensaje intercambiado entre el cliente y el servidor es un handshake en el cual el servidor le indica el ancho y alto de su mundo. Posteriormente el servidor envía la especificación de la biblioteca. En este momento el cliente puede detectar que le falta algún archivo de imagen.

Finalizado este handshake comienza "el juego" en el cual el servidor envía updates de los objetos a dibujar y el cliente envía eventos de teclado al servidor.

Protocolo de ejemplo

Un protocolo de ejemplo que el alumno puede considerar para empezar a diseñar uno propio puede ser:

Mensajes Servidor->Cliente

- **NewObject {**

  object_id

  Attributes {

  Spriteid

  X

  Y}

  }

  }

- **UpdateObject{**

```
            object_id
            Attributes {
                  State=[STILL|RUNNING]
                  X
                  Y}
            }
      }
```

Mensajes Cliente->Servidor

- **KeyEvent{**

```
            Scancode
            Type=[DOWN|UP]
      }
```

Biblioteca de Sprites

El archivo de configuración del servidor tiene una sección de sprites. Cada sprite está identificado por un id tipo string. Asimismo, cada cliente tendrá definido un directorio en disco con los archivos de imagen que va a utilizar. El nombre de archivo será igual al id del sprite (más la extensión .png, .jpg, etc.). De esa manera el cliente cuenta con un repositorio local de imágenes que el servidor va a indicar mediante dicho id.

Los sprites son animados y consisten de una secuencia de fotogramas (un sprite no animado es el caso particular de un sprite con un solo fotograma). Los fondos del escenario pueden ser sprites no animados (o incluso animados!)

Cada sprite se almacena como un único archivo de tipo imagen que contiene todos los fotogramas uno al lado del otro en una tira. Todos los fotogramas de un sprite tienen el mismo ancho (al que llamaremos ancho del sprite). El ancho del sprite se especifica en píxeles.

Restricción: El ancho de la imagen tiene que ser igual al ancho del fotograma multiplicado por la cantidad de fotogramas en la animación. Por ejemplo, si la animación tiene 10 fotogramas de 130 píxeles, el ancho de la imagen en píxeles será 1300 .

Por lo tanto para cada sprite de la biblioteca en el servidor se especifica:

- id <string>

- cantidad de fotogramas <entero>
- ancho <real>
- alto <real>

El ancho y alto indica cómo se debe escalar la imagen.

Escenario

Todos los objetos que integran el escenario que está definido en pixeles. Tanto el personaje como el escenario tienen un alto y un ancho. El personaje está confinado al rectángulo del escenario, es decir, si x_izquierdo y x_derecho son las coordenadas x de los bordes izquierdo y derecho del personaje, las siguientes son invariantes durante la ejecución

Vista multicapa

El escenario está formado por un array de “capas”. Las mismas serán imágenes que tendrán transparencias. El orden en el array define cuales capas están delante de otras. Consideramos que las primeras capas del array se encuentran detrás de las subsiguientes. Definimos como z-index de la capa al índice dentro del array. Cuando el personaje camina y sobrepasa un margen de la ventana se produce un desplazamiento horizontal (scroll) de la cámara. Todas las capas se mueven a una velocidad proporcional al ancho de la misma. Por lo tanto el ancho de cada capa define su velocidad de desplazamiento.

Controles

El protagonista podrá caminar y saltar. Al hacerlo se debe mostrar una animación que represente dicho movimiento.

Flecha Izquierda/Derecha: el protagonista camina hacia la derecha o izquierda.

Flecha arriba: el protagonista salta.

Tecla R: se vuelve a cargar el archivo de escenario (el cual puede haber sido modificado por un editor externo) y se reinicia la simulación.

Aclaraciones

- Las imágenes del fondo pueden estar en cualquier formato (por ejemplo: bmp, png, jpg) y puede ser cargada con las bibliotecas de extensión de SDL.

- Los archivos de escenarios deben respetar la sintaxis XML. El diseño de la estructura del mismo queda a criterio de los alumnos. Se recomienda pensar en un formato extensible que permita su reutilización en los siguientes trabajos prácticos.

- Los alumnos deben considerar todas las situaciones de error posibles, ante las cuales el programa deberá reaccionar de la manera que sea más apropiada (por ejemplo: mostrar un mensaje de error, no hacer nada, tomar una acción por defecto, etc).

Restricciones

- Para la representación gráfica se deberá utilizar la biblioteca SDL 2.0 (http://libsdl.org).
- Para la lectura y escritura de archivos XML debe utilizarse una biblioteca. **No se permite la utilización de un parser propio.**
- Todo el código debe ser desarrollado íntegramente por cada grupo. No se permite la reutilización de código de cuatrimestres anteriores o de otras materias. Ante cualquier duda se deberá consultar con los docentes. La reutilización de código sin consulta previa será condición suficiente para la desaprobación de la materia.

Este enunciado no es definitivo. Si se realizan cambios en clase se respetarán y evaluarán los mismos.

Fechas

Semana #	Fecha	Tema
1	17 de agosto	Presentación de la materia
2	24 de agosto	Presentación enunciado TP 1
3	31 de agosto	Consultas
4	7 de septiembre	Consultas
5	14 de septiembre	**Entrega TP1. Presentación enunciado TP 2**
6	21 de septiembre	Primer recuperatorio TP1. Consultas TP 2
7	28 de septiembre	Segundo recuperatorio TP1. Consultas TP 2
8	5 de octubre	Consultas
9	12 de octubre	**Entrega TP2. Presentación enunciado TP 3**
10	19 de octubre	Primer recuperatorio TP2. Consultas TP 3
11	26 de octubre	Segundo recuperatorio TP2. Consultas TP 3
12	2 de noviembre	Consultas
13	9 de noviembre	Consultas
14	16 de noviembre	**Entrega TP 3**
15	23 de noviembre	Primer recuperatorio TP 3
16	30 de noviembre	Segundo recuperatorio TP 3

Anexo: Archivo de escenario de ejemplo

```
<ventana>
    <ancho>800</ancho>
     <alto>600</alto>
</ventana>
 <sprites>
     <sprite>
          <id>personaje</id>
          <path>path/a/avion.png</path>
           <ancho>10</ancho>
            <alto>100</alto>
           <z-index>1</z-index>
      </sprite>
     <sprite>
         <id>disparo</id>
         <path>path/a/disparo.png</path>
         <ancho>5</ancho>
          <alto>5</alto>
         <z-index>1</z-index>
    </sprite>
</sprites>
<capas>
    <capa>
          <imagen_fondo>fondo1.png</imagen_fondo>
          <ancho> 500</ancho>
          <z-index>1</z-index>
    </capa>
   <capa>
          <imagen_fondo>fondo2.png</imagen_fondo>
          <ancho> 500</ancho>
          <z-index>2</z-index>
    </capa>
</capas>
```