Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2233 Programación Avanzada (II/2014)

Tarea 2

1. Objetivos

- Aplicar conceptos de I/O para crear, unir y leer archivos.
- Manejar archivos en formato binario.
- Manejar rutas y directorios.
- Familiarizarse con el concepto de serialización, por medio de la creación y lectura de archivos XML.

2. Introducción

Soldados del DCC,

El Dr. Fadić ha sido derrotado (mediante una larga y violenta guerra que costó la vida de millones de soldados simulados). ¡Larga vida a nuestro nuevo líder y supremo emperador, el General Zod-To!
Ahora, al fin podremos retomar la operación "Muere Tierra Muere". Eso sí, nuestro General, en su grandísma sabiduría, ha tenido una idea que facilitará de gran manera la conquista de este planeta. Nos ha pedido a nosotros que desarrollemos una maquinaría de guerra que permita generar ritmos que induzcan terror en los ejércitos que se nos opongan. Si ustedes llegasen a fallar en esta tarea, serán devorados.

Atte,
Comandante Anung Un Rama
Nueva División de Inteligencia
Ejercito planetario Adriano LXIX[1]

3. Problema

Tu misión es implementar una aplicación interactiva en consola[2] de un "drum machine". Éste deberá entregar la opciones de leer y crear partituras (5). Al leer una de estas partituras, tu programa deberá ser capaz de crear un archivo de audio en formato WAV, a partir de los *samples* de audio que te fueron entregados.

[1] La primera iniciativa de nuestro grandísimo General fue cambiarle el nombre al planeta.
[2] Mientras más amigable mejor.

4. Drum machine

Al iniciarse el Drum Machine, al usuario se le dará la posibilidad, mediante un menú en consola, de elegir entre dos modalidades: lector y escritor de partituras. Para esto además será necesaria la implementación de un explorador de archivos. Los detalles sobre estas modalidades y su funcionamiento se presentan a continuación.

4.1. Explorador de archivos

El programa debe ofrecer la opción de explorar los directorios del computador para buscar la partitura que se desea leer. A su vez, el usuario debe contar con la posibilidad de ingresar la ruta específica de la partitura para poder cargarla.
Una vez abierta la partitura, deberá pedirle al usuario dónde desea guardar el archivo WAV que se creará (puede ser explorando los directorios, o ingresando una ruta). De ahí, se creará el archivo WAV automáticamente según lo indicado por la partitura.

4.2. Lector de partituras

El lector debe ser capaz de leer e interpretar las partituras, cuya estructura está especificada en 5, generando como resultado un archivo WAV. Terminado esto, el programa regresará al menú inicial.

4.3. Creador de partituras

El creador de partituras debe entregar de una manera eficiente[3] un editor para confeccionar las partituras. La forma en que lo hagas queda a tu criterio, pero debe contar con la opción de elegir entre los distintos tipos de *samples* que se te fueron entregados.
Una vez creada la partitura, tu programa ofrecerá al usuario la opción de crear un archivo WAV a partir de ella, o volver al menú inicial.

5. Partituras

Las partituras de cada canción están constituídas por un archivo XML que contiene una lista de compases. Un compás corresponde a la estructura básica de cada canción, y tiene un cierto número de "beats" (unidades de tiempo) y velocidad, además de una lista de notas. La estructura de cada partitura se presenta a continuación, con los elementos que la conforman y sus atributos: [4]

- Lista de compases
  - Largo del compás - *length*: Corresponde al número de tiempos del compás. Dependiendo de la velocidad del compás, este atributo permite determinar la duración de cada compás, en segundos.
  - Velocidad del compás - *tempo*: Determina la velocidad del compás en bpm (beats per minute). Es decir, este atributo determina cuántos tiempos del compás habrían en un minuto.
  - Repeticiones - *loop*: Indica cuántas veces se repite el compás.
  - Lista de notas en el compás
    - Instrumento - *type*: Tipo de instrumento al que correspnde la nota. Ej: *kick*, *snare*, *ride*, etc.
    - Número de sample - *num*: Número del sample de audio correspondiente a la nota, del tipo descrito por el atributo anterior.
    - Posición - *pos*: La posición de la nota dentro del compás partiendo del primero. Puede ser fraccionaria, para indicar notas posicionadas entre tiempos.
    - Intensidad - *i*: Qué tan fuerte debe sonar la nota en el archivo final. Corresponde a un porcentaje.

[3] Por el bien de su nota, de usted, y de la humanidad.
[4] Revisar XML de ejemplo.

6. *Samples* de audio

Los samples consisten de archivos de audio en formato WAV que corresponden al sonido de una nota de alguna parte de la batería. Estos samples se encuentran separados en carpetas según el tipo de instrumento (bombo, caja, platillo). **No se deberán mover los samples de sus carpetas**. Además, para cada tipo de instrumento, se otorgan tres samples distintos. El formato del nombre de estos archivos es el siguiente: `(id) + (num) + ".wav"`, donde `id` corresponde al identificador del tipo de instrumento y `num` distingue el número de sample. Los identificadores para cada tipo de instrumento se presentan a continuación:

- *crash_high* - `ch`
- *crash_low* - `cl`
- *hihat_closed* - `hhc`
- *hihat_open* - `hho`
- *kick* - `k`
- *ride* - `r`
- *snare* - `s`
- *tom_floor* - `tf`
- *tom_high* - `th`
- *tom_low* - `tl`

A partir de una partitura, se deberá construir un audio final sumando las muestras de los samples dados en distintos instantes según lo descrito por ésta. Por último, cabe destacar que la frecuencia de muestreo de los samples de audio es de 44100Hz[5]. Por lo tanto, el audio final que el programa genere debe tener esta misma frecuencia.

7. Bonus

7.1. THIS IS MY JAM! (7 %)

El general Zod-To, muy complacido con tu resultante maquinaria de guerra, ha decidido que los enemigos del imperio deben enfrentar su perdición mientras escuchan su secuencia rítmica favorita.

Para obtener este bonus[6], debes entregar, junto con tu tarea, la partitura y archivo de audio correspondiente al solo de batería de la canción *La poderosa muerte* interpretada por *Los Jaivas*[7]. Esta partitura **debe** ser desarrollada por tí, y el archivo de audio **debe** haber sido generado por tu programa.

7.2. Needs more Cowbell (3 %)

El General Zod-To tiene una fiebre, y la única prescripción es **más cencerro**.

Para obtener este bonus debes agregar un nuevo tipo de sample de audio, que corresponda a un cencerro (cowbell). Estos samples deben ir en su carpeta respectiva, llamada *cowbell*, junto con el resto de los samples. El identificador para este instrumento será `cow`. Además, su programa deberá adaptarse para ser capaz de escribir y leer partituras que contengan este nuevo tipo de sample, junto con construir los archivos de audio respectivos.

[5] Curiosamente, los adrianos del planeta Adriano LXIX perciben el mismo rango de frecuencias que los habitantes del planeta Tierra...
[6] Y no solo un bonus de nota, pues el que cumpla con lo pedido se ganará un café invitado por el mismísimo General Zod-To.
[7] Corresponde al segmento del 5:00 al 5:27 aprox., del siguiente video.

8. Restricciones y alcances

- Tu programa debe ser desarrollado en el lenguaje C#, sobre .Net Framework 4.5.
- El entorno de desarrollo a utilizar debe ser Microsoft Visual Studio 2013.
- El ayudante puede bonificar o castigar tu puntaje[8] de la tarea, si le parece adecuado. Se recomienda ordenar el código.
- No subas las carpetas bin, obj ni el archivo .suo[9], hay un descuento de un punto en tu nota por hacerlo.
- Es obligatorio usar un archivo de texto llamado ReadMe para documentar su programa, explicar las cosas que asumió, describir en qué cosas falla y en qué no, etc. En caso de no agregar este archivo a su repositorio, su nota se verá reducida.
- Queda extrictamente prohibido el uso de cualquier librería externa que trabaje con audios. El trabajo con los archivos WAV deberá ser de forma directa con los bytes.

9. Entrega

- **Fecha/hora:** Jueves 25 de Septiembre de 2014 / 23:59.
- **Lugar:** En repositorios SVN personales, al interior del directorio `Tarea 2`.

Debe subir la solución de Visual Studio completa en código, sin ejecutables. Tareas que no compilen o no cumplan con las restricciones señaladas en este enunciado tendrán la calificación mínima (1.0).

[8] ±5 décimas.
[9] Es un archivo oculto.