Sonido en videojuegos

FMOD Studio 2

1. Utilizando las muestras de la carpeta tormenta vamos a crear un evento tormenta con varios parámetros para controlar la intensidad de la lluvia, la frecuencia de los truenos, el volumen del evento y la posible obstrucción/oclusión del sonido. A continuación se detalla el funcionamientos de los parámetros:

   - *Intensidad*: utilizando distintas muestras de lluvia (pueden añadirse otras disponibles en internet), este parámetro controla la intensidad del sonido de lluvia. Hacer una mezcla adecuada de las muestras para conseguir transiciones suaves.
   - *Truenos*: utilizaremos distintos sonidos de truenos que se dispararán con mayor o menor frecuencia en función de este parámetro. Distintas muestras de truenos pueden sonar simultáneamente.
   - *Volumen*: controla el volumen del master del evento.
   - *Obstrucción/Oclusión*: estos+ parámetros servirán para simular la percepción de oyente desde un habitáculo (parcialmente) cerrado, como una casa o un coche. El efecto se consigue incorporando un *filtro paso bajo* (puede utilizarse en ecualizador multibanda). El parámetro obstrucción controlará la frecuencia de corte del filtro y la oclusión controlará la ganancia del efecto.

2. El archivo *maquina.mp3* contiene una muestra de máquina de escribir con varias pulsaciones y retorno de carro. Trocear la pista para obtener distintas pulsaciones aisladas o muestras de varias pulsaciones y una muestra con el retorno de carro. Después generar un evento con un instrumento multitímbrico que incorpore dichas muestras. Pueden incluirse ligeras variaciones dinámicas de pitch y volumen para tener más variación. Para obtener un efecto realista deben asociarse probabilidades adecuadas a las pulsaciones y el retorno de carro.

   Puede hacerse más sofisticado utilizando un *Scatterer Instrument*, con *Polyphony=1* y controlando *Min & Max Spawn Interval* para incluir un intervalo de tiempo aleatorio entre pulsaciones.

   Aún puede hacerse más sofisticado teniendo en cuenta que una línea de texto en una máquina de escribir tiene un número limitado de caracteres (que obliga al retorno de carro), que al llegar al final de línea hay un pequeño timbre de aviso (una de las pulsaciones de la muestra incluye ese timbre), que puede haber líneas imcompletas, varios retornos de carro seguidos.... ¿Se podría incorporar todo esto en un evento FMOD? En caso negativo, pueden implementarse eventos independientes para las pulsaciones, retorno de carro y timbre, y luego controlarlos desde la API con un autómata con transiciones probabilísticas entre estados.

3. Música adaptativa. La carpeta *musica* contine 12 pistas correspondientes a un mismo tema musical y que pueden mezclarse dinámicamente para obtener mayor o menor riqueza tímbrica. Para ello se diseñará un evento *música* que incorpore dichas pistas, cuya ganancia se controlará con un parámetro *intensidad*. Puede utilizase la pista de bajo como pista de referencia e ir añadiendo percusiones y líneas melódicas (algunas de ellas aletoriametne) en función de la del parámetro de intensidad.

   Más difícil: las pistas proporcionadas son loops que pueden acortarse desde Audacity. Se pueden extraer esos loops y minimizar el tamaño de las muestras. Para garantizar la sincronización será necesario hacer un análisis cuidadoso del BPM de las muestras antes de incorporar los fragmentos en FMOD.

4. Utilizando la API *studio* de FMOD de puede implementar (en modo texto) un explorador de eventos y parámetros para cualquier bando de sonido de FMOD. Para el banco que viene como ejemplo con FMOD este navegador mostraría el contenido de los distintos bancos:

```
Master.bank
  0 [ ]: snapshot:/Slow-Motion
  1 [ ]: snapshot:/Health Low
  2 [ ]: snapshot:/Reverb/Sewer Reverb
  3 [ ]: snapshot:/Pause
  4 [ ]: snapshot:/Reverb/Cave Reverb

Master.strings.bank

Music.bank
  5 [ ]: event:/Music/Level 03
         [a/A]  Stinger [0,1]: 0
         [b/B]  Intensity [0,4]: 0
  6 [ ]: event:/Music/Level 01
         [c/C]  Stinger [0,1]: 0
         [d/D]  Progression [0,1]: 0
  7 [ ]: event:/Music/Radio Station
         [e/E]  Spatializer [0,2]: 0
         [f/F]  Freq [0,3]: 0
  8 [ ]: event:/Music/Level 02
         [g/G]  Area [0,80]: 0

SFX.bank
  9 [ ]: event:/Character/Player Footsteps
         [h/H]  Surface [0,2]: 0
 10 [ ]: event:/Character/Door Open
 11 [ ]: snapshot:/Health Low
 12 [ ]: event:/Weapons/Explosion
         [i/I]  Size [0,3]: 0
         [j/J]  Distance [0,100]: 0
 13 [ ]: event:/Character/Enemy Footsteps
 14 [ ]: event:/Interactables/Barrel Roll
         [k/K]  Speed [0,8]: 0
 15 [ ]: event:/Character/Door Close
```

Por ejemplo, `Music.Bank` contiene 4 eventos, numerados de 5 a 8, que se activan con dichos números. Así, 5 activaría el evento `Music/Level 03`. Después las letras permiten subir o bajar el valor de los parámetros correspondientes: `"B"` sube el parámetro `Intensity` y `"b"` lo baja.

Algunos métodos útiles para consultar: *loadBankFile, getEventCount, getEventList, getPath, getParameterDescriptionCount, getParameterDescriptionByIndex, getEvent, createInstance, getParameterDescriptionByName, setParameterByID*