

LIBGDX Musica son

UNIDADE 2: Música e efectos de son

Sumario

- 1 Introducción
- 2 Efectos de son
- 3 Música
- 4 Efectos de son aleatorios

Introdución

Aínda que non o pareza a música e efectos de son son moi importantes para un xogo. Agora mesmo como tedes o xogo non 'engancha' moito, pero xa veredes como cambia a cousa cando poñamos a música e os efectos de son.

Algúns dos recursos que temos en Internet son:

- Música: <http://dig.ccmixer.org>
- Efectos de son:
 - <http://www.soundboard.com/category/Sound-Effects>
 - <http://www.sonidosmp3gratis.com/download.php?sonido=claxon%20de%20coche>
 - http://www.drpetter.se/project_sfxr.html: Programa para xerar sons de tipo arcade.

Efectos de son

Información da wiki: <https://github.com/libgdx/libgdx/wiki/Sound-effects>

Os efectos de son a diferenza da música, son sons que duran uns segundos e que son tocados cando se produce algún evento no xogo.

A clase que manexa os sons é a [clase Sounds](#).

A forma que temos de cargar un arquivo será a seguinte:

```
Sound son = Gdx.audio.newSound(Gdx.files.internal("data/arquivo_son.mp3"));
```

Veremos no punto [Gardando datos](#) que daremos despois, explicamos que `Gdx.files.internal` fai referencia ó cartafol assets do proxecto Android.

Podedes ver todos os métodos dispoñibles no enlace anterior da clase `Sound`. Entre eles temos:

- `play()`: toca o son. Está sobrecargado e pode usarse a versión no que se lle envía o nivel de volume.

Dito método devolve un id (long) o cal pode ser usado para:

[Facer looping](#): que se repita o son.
[Modificar o volume](#): modifica o volume do son. Valores entre 0 e 1.

- [Método stop](#): para de tocar.
- [Método dispose](#): libera o son da memoria. Necesario facelo ó saír do xogo.

Por exemplo:

```
Sound son = Gdx.audio.newSound(Gdx.files.internal("data/arquivo_son.mp3"));
```

```
son.play();
son.dispose();
```

Música

Información da wiki: <https://github.com/libgdx/libgdx/wiki/Streaming-music>

A diferenza dos efectos de son que se gardan en memoria, a música é cargada do disco cando se necesita. Sempre deberemos utilizar esta opción cando teñamos unha música de fondo.

A música de fondo soe ocupar moito espazo en disco polo que é aconsellable editala e reducir a súa calidade (frecuencia de muestreo, canal mono,....)

Un programa para facer isto é o Audacity: <http://audacity.sourceforge.net/?lang=es>

Unha vez temos a música no cartafol assets da versión Android podemos utilizala facendo uso da clase **Music**.

```
Music musica = Gdx.audio.newMusic(Gdx.files.internal("data/arquivo_musica.mp3"));
```

Entre os métodos a utiliza temos:

- **Método play**: reproduce o arquivo de audio.
- **Método isPlaying**: devolve un boolean indicando se a música segue tocando.
- **Método isLooping**: devolve un boolean indicando se a música non vai parar.
- **Método getPosition**: devolve a posición en segundos.
- **Modificar o volume**: modifica o volume do son. Valores entre 0 e 1.
- **Método stop**: para de tocar.
- **Método dispose**: libera o son da memoria. Necesario facelo ó saír do xogo.

Por exemplo:

```
Music musica = Gdx.audio.newMusic(Gdx.files.internal("data/arquivo_musica.mp3"));
musica .setVolume(0.5f);
musica .setLooping(true);
musica .stop();
musica.dispose();
```

TAREFA 2.12 A FACER: Esta parte está asociada á realización dunha tarefa.

Efectos de son aleatorios

No noso xogo queremos que cada certo tempo se escoite un son de claxon.

Para facer isto temos dúas opcións:

- Opción a):

- ◊ Definir unha variable no método controlarXogo de tipo float: float tempo=0;
- ◊ Sumarlle delta: tempo+=delta dentro do método update e cada certo tempo facer tocar o claxon:

```
tempo+=delta;
if (tempo>=2f) {
    tempo=0f;
    Audio.claxon[0].play();
}
```

- Opción b):

Utilizar a **clase Timer**.

Esta clase permite cada certo tempo e optativamente, de forma continuada, realizar **unha tarefa (Task)**.

O proceso para o que nos ocupa sería:

Definimos na clase Audio e dentro do método inicializarMusica (feito na tarefa anterior) un array de efectos de son para os coches:

Código da clase Audio

Obxectivo: definir un array de son para os tipos de claxon.

```
public static Sound claxon[] = new Sound[3];
.....
public static void inicializarMusica() {
claxon[0] = Gdx.audio
.newSound(Gdx.files.internal("MUSICA/LIBGDX_claxon.mp3"));
claxon[1] = Gdx.audio
.newSound(Gdx.files.internal("MUSICA/LIBGDX_claxon2.mp3"));
claxon[2] = Gdx.audio
.newSound(Gdx.files.internal("MUSICA/LIBGDX_claxon3.mp3"));
.....
}
```

Nota: Os arquivos de audio os tedes subidos previamente na realización da tarefa 2.12.

Agora definimos na mesma clase unha tarefa para tocar o claxon. Necesitamos definila (poderíamos crear unha tarefa sen ter referencia a ela, de forma anónima) xa que cando vaíamos á pantalla de pausa, temos que parar os claxon´s e volvelos a poñer en marcha cando volvamos á pantalla do xogo.

Código da clase Audio

Obxectivo: definir a tarefa para que cada certo tempo toque un claxon.

```
public class Audio {
.....
private static Task claxonCoches;
public static void iniciarClaxon(){
claxonCoches = new Task(){

@Override
public void run() {
// TODO Auto-generated method stub
Audio.claxon[MathUtils.random(0, 2)].play();
}
};

Timer.schedule(claxonCoches, 0, 2f);

}

public static void paraClaxon(){
claxonCoches.cancel();
}
.....
}
```

Agora só temos que chamar a ditos métodos dende a pantalla do xogo.

Código da clase PantallaXogo

Obxectivo: iniciar e deter o claxon dos coches en función se estamos xogando ou en pausa/saír.

```
@Override
```

```

public void show() {
    // TODO Auto-generated method stub
    Gdx.input.setInputProcessor(this);
    if (pause) Audio.playMusica();
    pause=false;
    Audio.auga.play();
    Audio.iniciarClaxon();
}

@Override
public void hide() {
    // TODO Auto-generated method stub
    Audio.auga.stop();
    if ((finXogo) || (sair)) dispose();
    Audio.paraClaxon();
}

@Override
public void resume() {
    // TODO Auto-generated method stub
    Gdx.input.setInputProcessor(this);
    pause=false;
    Audio.playMusica();
    Audio.iniciarClaxon();
}

.....

```