How to play multiple sound fx in Java ME

This article needs to be updated: If you found this article useful, please fix the problems below then delete the {{ArticleNeedsUpdate}} template from the article to remove this warning.

Reasons: hamishwillee (28 Sep 2012)
An article with relatively complicated code like this should deliver the sound manager in an assembly, along with test code. Ideally it should also state what platforms it has been tested on, and also provide links to other Sound managers, and explanation of why it is better.

After playing with many sound managers I decided to write my own SoundManager. This is what i'm using in my games. It proved to be stable (at least on Nokia 5310) and good performance. If you think you can improve it, please do it, and don't forget to send me (or publish!) your version.

SoundManager manage the sounds via buffer. Only the buffer is manipulated; this is to improve performance. Each SoundFX holds one player.

SoundFX class is responsible for:

- `load()`: load sound file;
- `anula()`: destroy player;
- `started()`: inform if it is playing;
- `closed()`: inform if it played till the end;
- `play()`: allocate and play;

Warning: On NOKIA 5310 the maximum SOUND length is 6, no matter file size. When I tried 7 (very small file, 5Kb), it didn't work! (in other words: no sound thru speaker!) Memory? If it doesn't work try less SOUNDS.

How to use the class

```
* habilita(boolean enable): enable/disable. for example, called from options menu
(toggle on/off)
* playFX(int index): play SOUNDS[index]. for example, called from game logic
* Put into destroyApp(boolean b): if (SoundManager.taLigado())
SoundManager.desligaSounds();    (or Menu -> exit)
* Put into Midlet constructor:  new SoundManager();
```

```
/*****************************************************************************************/
/*
 * Esta classe tem como principal objetivo administrar efeitos sonoros tocados
 * no aparelho, ja que a maioria so' toca um som de cada vez.
 * Se o buffer estiver ainda tocando e tentarmos chamar playFX com outro dado
 * index (diferente do atual) o som atual termina de executar ate' o fim e este
 * ultimo som passado por index, logicamente, nao sera tocado.
 * Por este motivo, tente ter o menor tamanho em milisegundos de som (tipico max = 500)
 * contido no arquivo, se quiser um efeito tocado bem proximo do outro.
 * Somente o buffer e' manipulado, os players utilizados sempre estao alocados.
 * OBS: no NOKIA 5310 consegui somente 6 itens em SOUNDS. na tentativa de um setimo
 * efeito (o arquivo era bem pequeno 5Kbytes) nao funcionou mais (uma suposicao de
estouro
 * de memoria provavelmente), em outras palavras, nao saiu mais som pelo alto-falante.
 * Como usar (TECHNICAL INFORMATION):
 * habilita(boolean enable) chamado por exemplo de um menu opcoes (toggle on/off)
 * playFX(int index) toca o som atual passado por exemplo na logica do jogo
 * nao se esqueca de inserir: if (SoundManager.taLigado()) SoundManager.desligaSounds();
 * onde?   R.: na logica destroyApp(boolean b)     (ou Menu -> exit)
 */
```


(C) Copyright Nokia 2014. All rights reserved.

```
import javax.microedition.media.MediaException;

/**
 * @author George Roberto Peres
 * @version 1.1
 */

public final class SoundManager {

  private static SoundFX buffer = new SoundFX(null);
  private static final SoundFX[] EFFECTS;
  private static final String[] SOUNDS;
  // put your sounds here (with full path)
  static
  {
    EFFECTS = new SoundFX[(SOUNDS = (new String[] {
        "/jump.wav", "/ovni.wav", "/xplod1.wav", "/xplod2.wav", "/yeah.wav"
    })).length];
    for (byte i=0; i<SOUNDS.length; i++) EFFECTS[i] = new SoundFX(SOUNDS[i]);
  }
  //
  private static boolean ligado;
  // enable
  protected static void habilita(boolean enable) {
    if (enable) ligaSounds(); else desligaSounds();
  }

  // turn on
  private static void ligaSounds() {
    if (ligado) return;
    ligado = true;
    for (byte i=0; i<EFFECTS.length; i++) EFFECTS[i].load();
  }

  // turn off
  protected static void desligaSounds() {
    if (!ligado) return;
    ligado = false;
    for (byte i=0; i<EFFECTS.length; i++) EFFECTS[i].anula();
    buffer.anula();
  }

  // play sound according to SOUNDS[index]
  protected static void playFX(int index) {
    if (!ligado || buffer.started()) return;  // playing?
    buffer = EFFECTS[index];   // allocate buffer
    buffer.play();   // play
  }
}



/*****************************************************************************************/


/* This class (arg fileName) is responsible for load the sound into memory. Methods:
 * load(): load sound file
 * anula(): finish player object
 * started(): still playing?
 * play(): aloccate and play
```


(C) Copyright Nokia 2014. All rights reserved.

```
 *
 */

import java.io.IOException;
import javax.microedition.media.Manager;
import javax.microedition.media.MediaException;
import javax.microedition.media.Player;
//import javax.microedition.media.control.VolumeControl;

/**
 * @author George Roberto Peres
 * @version 1.1
 */

public final class SoundFX {

  private final String file;
  private Player player;
  //private VolumeControl vc;

  protected SoundFX(String fileName) {
    file = fileName;
  }

  protected void load() {
    if (player != null) return;
    try {
      player = Manager.createPlayer(getClass().getResourceAsStream(file), "audio/wav");
      player.prefetch(); //change to REALIZED state
      // volume control
      //VolumeControl vc = (VolumeControl) player.getControl("VolumeControl");
      //if (vc != null) vc.setLevel(100);
    }
    catch (IOException e) { return; }
    catch (MediaException e) { return; }
  }

  protected void anula() {
    if (player == null) return;
    player.close();
    player = null;
  }

  protected boolean started() {
    if (player == null) return false;
    return (player.getState()==Player.STARTED);
  }

  protected void play() {
    if (player != null) {
      try {
        player.prefetch();
        player.stop();
        player.start();
      }
      catch (MediaException me) {  }
    }
  }

}
```


(C) Copyright Nokia 2014. All rights reserved.

Note from the author

Please don't forget to check my game projects (need feedback): Queops, LunarPatrol & X-Rally.

Since last year Nokia Developer have grown a lot, and the site was filled with many source code examples; many of them didn't worked as expected in the real thing....and many worked.....but they serve as an inspiration on how to start anything in any platform.... Thanks very much for all the contributors! You all deserve it. (you know who you are!). I hope you understand my English, 'cause my native language is Brazilian Portuguese. "Todo artista quer ver sua obra publicada."

(C) Copyright Nokia 2014. All rights reserved.