Java Sound

Chris Orona
CS491B
May 17, 2005

Playing Sound Clips

- AudioClip getAudioClip(URL codebase, String filename)
  - Used in Applets to download an audio clip.
- play()
  - Plays an audio clip.
- stop()
  - Stops the clip.
- loop()
  - Plays the clip in a loop.

(Demo Applet)

AudioClip Limitations

- Can only be used by Applets
- Can't pause/replay clips
- Can't play portions of clips
- Can't record data
- No event system

Java Sound Library

- javax.sound.sampled.* package
- Can only play simple types, like WAV or AU, but can be extended to include other types.

Java Sound Classes

- AudioSystem
  - Object that we use to obtain our audio objects
- AudioInputStream
  - Stream which contains the audio data
- DataLine
  - A buffer which is used for reading or writing
- Mixer
  - Sound mixers installed on the system
- Clip
  - Object which contains the high level functions, similar to AudioClip.

Playing Sound Clips (2)

- AudioInputStream stream = AudioSystem.getAudioInputStream(source)
  - source = (InputStream, File, URL)
- Clip clip = AudioSystem.getClip();
  - Creates a new clip object using the default mixer
- clip.open(stream);
- clip.start();
  - Plays the clip
- clip.close();
  - When we're done using the clip

(Sound Demo 1)

Sound Events

- LineListener interface
- Can listen when a line is opened/closed, or stopped/started

Sound Event Code

```
clip.addLineListener(new LineListener() {
   public void update(LineEvent e) {
        LineEvent.Type t = e.getType();
        if (e == LineEvent.Type.START) {
                // line started
        }
   }
}
```

LineEvent object
    getLine()
        audio source
    getType()
        type of event
    CLOSE, OPEN, START
    or STOP

(Sound Demo 2)

MIDI in Java

- javax.sound.midi.* package
- Plays MIDI, RMF formats

Music Playback

- MidiSystem
  - Used like AudioSystem, master class that we obtain objects from
- Sequencer
  - A device that plays MIDI files.
- Sequence
  - The actual music file. Like Clip is for sounds.

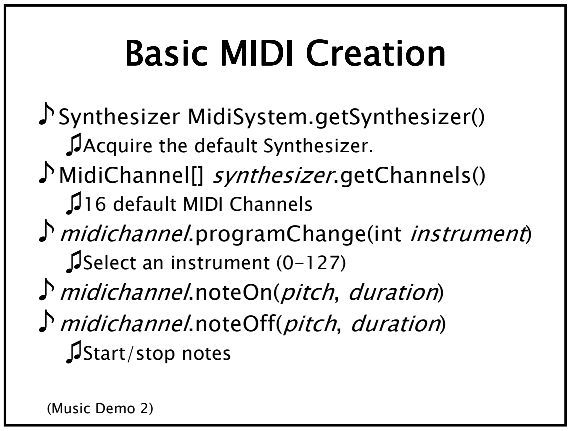
Music Playback (2)

- MidiSystem.getSequencer()
  - Obtains the default Sequencer
- MidiSystem.getSequence(source)
  - Loads a sequence from a source
- Sequencer.setSequence(Sequence)
  - Loads a sequence into the sequencer
- Sequencer.start()
- Sequencer.stop()
  - Start or stop a sequence
- Sequencer.setLoopCount()
  - Set number of times to loop

(Music Demo 1)

MIDI Synthesizing

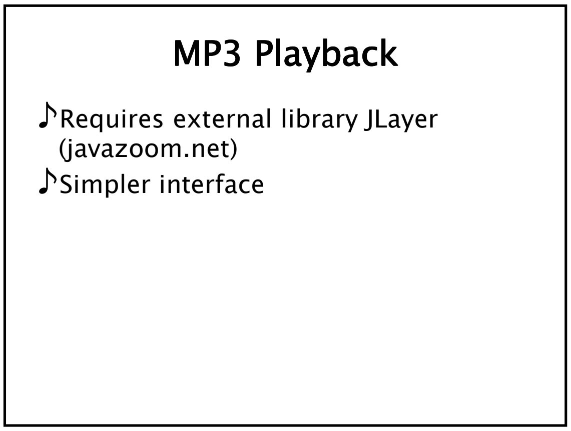
- Create and record your own MIDI files.
- Ways to handle MIDI events:
  - Synthesizer high level functions
  - Using MidiMessages
    - (not covered)



Basic MIDI Creation

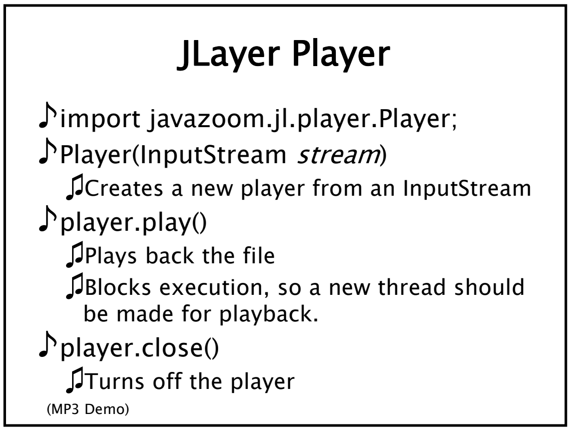
- Synthesizer MidiSystem.getSynthesizer()
  - Acquire the default Synthesizer.
- MidiChannel[] *synthesizer*.getChannels()
  - 16 default MIDI Channels
- *midichannel*.programChange(int *instrument*)
  - Select an instrument (0-127)
- *midichannel*.noteOn(*pitch*, *duration*)
- *midichannel*.noteOff(*pitch*, *duration*)
  - Start/stop notes

(Music Demo 2)



MP3 Playback

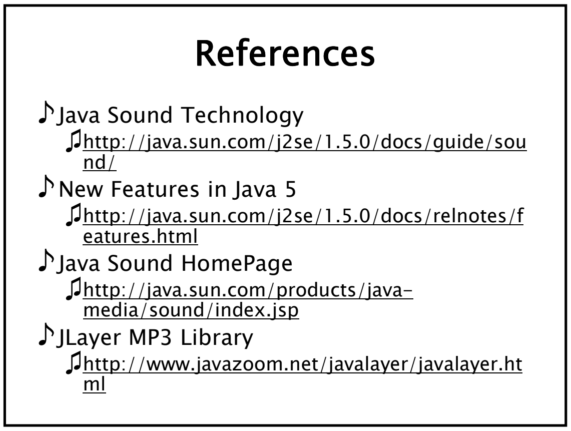
- Requires external library JLayer (javazoom.net)
- Simpler interface



JLayer Player

- import javazoom.jl.player.Player;
- Player(InputStream *stream*)
  - Creates a new player from an InputStream
- player.play()
  - Plays back the file
  - Blocks execution, so a new thread should be made for playback.
- player.close()
  - Turns off the player

(MP3 Demo)



References

- Java Sound Technology
  - http://java.sun.com/j2se/1.5.0/docs/guide/sound/
- New Features in Java 5
  - http://java.sun.com/j2se/1.5.0/docs/relnotes/features.html
- Java Sound HomePage
  - http://java.sun.com/products/java-media/sound/index.jsp
- JLayer MP3 Library
  - http://www.javazoom.net/javalayer/javalayer.html