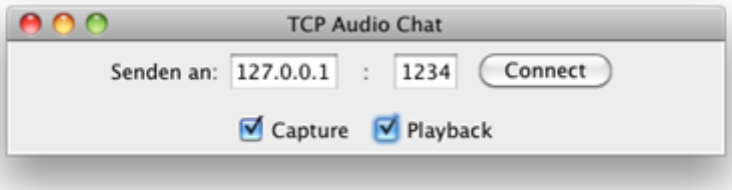
Übung zur Vorlesung

Multimedia im Netz

Doris Hausen
Ludwig-Maximilians-Universität München
Wintersemester 2009/2010

TCP Audio Chat

- GUI Beispiel
  - Eingabe für IP und Port
  - Auswahl von
    Aufnahme & Abspielen



- GUI darf natürlich gern erweitert und optimiert werden!

Warum Audio mit TCP

- Für das menschliche Gehör ist (mäßiger) Paketverlust kein Problem und UDP verursacht weniger Overhead bei der Übertragung und ist dadurch schneller.
  → UDP wird meist für Audio- und Videostreaming verwendet

- ABER: Wenn man UDP für Audioübertragung verwendet muss über Timestamps oder Paketnummern die Reihenfolge selbst kontrolliert werden, verlorene oder duplizierte Pakete gefunden werden usw.
  →TCP übernimmt das für uns

Audio Qualität vs. Datenrate

	SampleRate (Hz)	Bits per Sample	Mono/Stereo
Telefon	8000	8	Mono
Radio	22050	16	Stereo
CD	44100	16	Stereo

→ Für unsere Zwecke Orientierung am Telefon

Audio aufnehmen (1)

- `Line`
  Interface; Kann mit mono und mehrkanal Audioströmen umgehen; Sowohl für Audio Eingang wie auch Ausgang.

- `DataLine.Info`
  Fügt der `Line` verschiedene Funktionalitäten wie z.B. `start`, `stop`, `drain` oder `flush` hinzu.

- `TargetDataLine`
  Ist eine spezielle `DataLine` von der Audio gelesen werden kann. Daten kommen hier z.B. von einem Mikrophon.
  - `open(AudioFormat format):` Öffnet die Line und reserviert alle nötigen Resourcen
  - `start():` Startet die tatsächliche Aufnahme
    (Hinweis: Methode von DataLine nicht TargetDataLine)

Audio aufnehmen (2)

- In den `OutputStream` schreiben (innerhalb eines Threads):
  - `read(byte[] b, int off, int len)`: Liest Audiodaten in den InputBuffer der Line
  - `write(byte[] b, int off, int len)`: Schreibt die Daten auf den Outputstream

Audio abspielen (1)

- `InputStream` zu `AudioInputStream` aufwerten

- Analog zur Aufnahme: `DataLine.Info` und statt `TargetDataLine` eine `SourceDataLine`

- `SourceDataLine`
  Ist eine spezielle `DataLine` auf die Audio geschrieben werden kann. Sie buffert die Datenbytes wenn nötig
  - `open(AudioFormat format):` Öffnet die Line und reserviert alle nötigen Resourcen
  - `start():` Startet das Abspielen
    (Hinweis: Methode von DataLine nicht SourceDataLine)

Audio abspielen (2)

- Aus dem `AudioInputStream` lesen (innerhalb eines Threads):
  - `read(byte[] b, int off, int len):` Liest Audiodaten in den InputBuffer der Line
  - `write(byte[] b, int off, int len):` Schreibt die Daten aus dem Buffer auf die line und gibt sie dadurch als Ton aus
  - `drain():` Sorgt dafür, dass der Buffer bis zum Ende ausgelesen wird.