Usage

Running the Demo

Once the setup procedure has been completed Start the applicaiton by performing the follwoing steps:

1. On PC Host 1 and PC Host 2 from bdk\test\demo2 execute the following command "`lib\stringsloader.exe host-win32.init`"
2. On PC Host 1 run `"bdk\radkit\samples\audioplayer\Release\AudioPlayer.exe"`

User Interface
Once running, there are two large, empty boxes, with three control buttons in the middle. The box on the left will display MP3s. The box on the right displays speaker devices on the network where MP3s can be played (figure 1).

Controls

- The "Refresh View" button uses the network browser to discover MP3s and speakers. When it constructs a "query" for MP3s, it also optionally queries based on "genre" and "artist". The query can be arbitrarily restrictive, so the application does not have to sort a large response looking for what they want. After the user presses "Refresh View", the response is populated into a list-like structure in the left hand box.
- "Play" Starts a "data flow" between the mp3 file and the speaker
- "Stop" halts the "data flow" and destroys the music session.

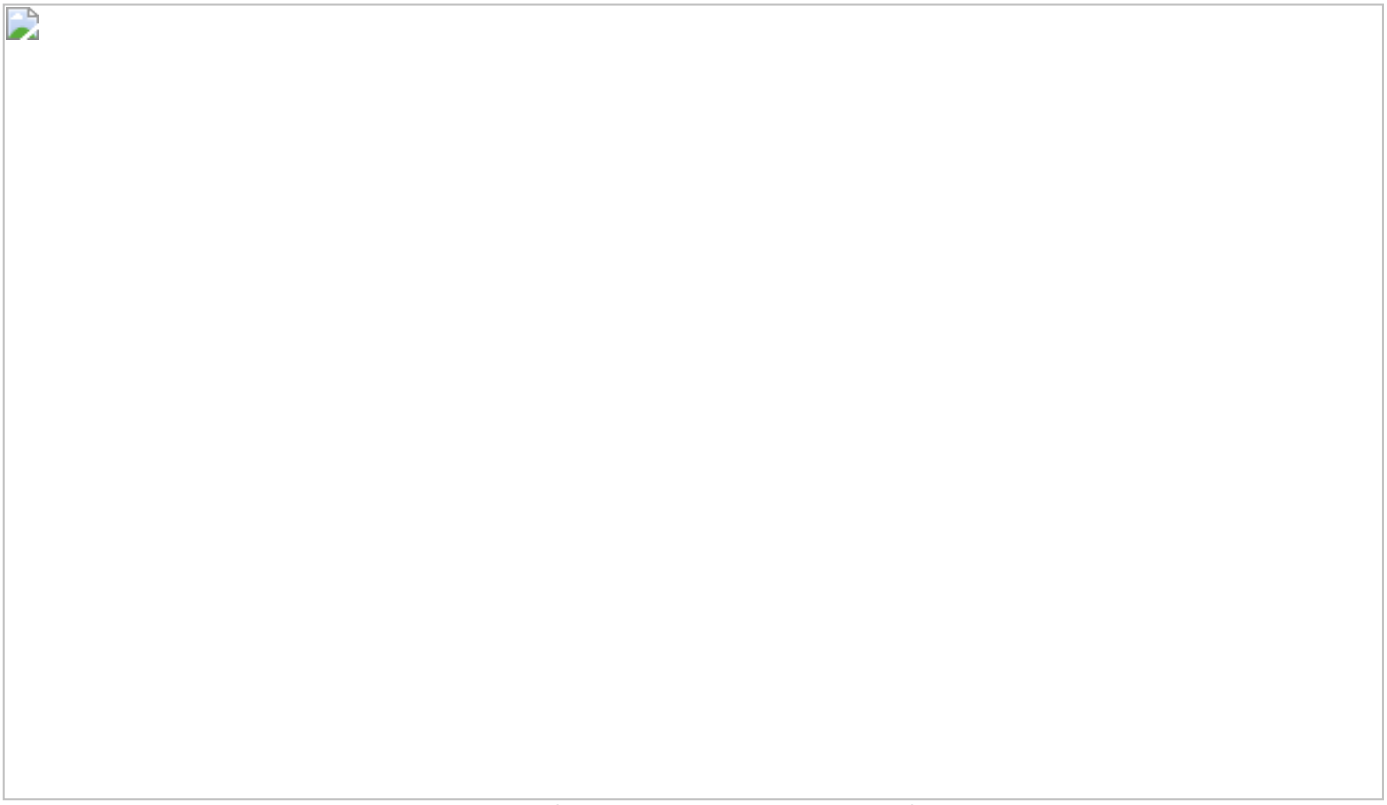

Figure 1 - Strings Audio Player User Interface

1. Select a single source from the "music file" box
2. Select one or more speakers from the "speakers" box
3. Press "Play".

What's happening:

- The Strings framework creates a session at the file and the speaker objects, then initiates a "data flow" from the music file to the speakers.
- When the "data flow" is started, Strings loads the MP3, then determines that it must convert the MP3 to PCM data before playing it on the speakers.
- The UI displays a "Session #" entry under the music file that is serving as the source, so that the user can further manipulate the data flow.
- The application user now can select the Session # by clicking on it and press the "Stop" button to stop the audio. If the audio runs to completion, the Session # will remain until the user selects it and presses Stop.
- Note that, because the application is coupled with a Strings engine, the user can kill the application without stopping the audio. The application controls a Strings engine, but the Strings engine does not need the application to run.

Design Overview

The application design is a standard form-based MFC application. This dialog box has two tree controls, one for MP3s and one for speakers.
When "Refresh View" button is pressed, the application constructs two networks queries. First it submits a query for MP3s, optionally constrained by a user supplied "genre" and "artist". For example, if the user typed "Nirvana" into the "artist" field, the application would construct a query string that looks like "query:ext='mp3' AND artist='Nirvana'". Then the application uses a network browser to submit the query. The response is a list of network objects representing the MP3s, which the application uses to populate the "music files" tree control. Next, the application submits a query for all network speakers. It uses, "query:Rpc-Class=='Speaker'", which asks for all network objects that support the "Speaker" remote procedure call interface. It uses these values to populate the speakers tree control.
When the user presses the "Play" button, the application creates a "session" on the selected music file and a session on each checked speaker. It then create a "data flow" from the music file session to the speaker sessions. The "data flow" is started, which will start the audio.
When the user presses the "Stop" button, the application stops the data flow and destroys it.

The code for this application is included and is located at 'bdk\radkit\samples\audioplayer