

SUMMARY

1. Server starts running

- a. Searches `Instruments` directory, attempts to load files (ignores directories)
- b. Constructs list of instruments from files
 - i. `MidiInstrument` class sets up MIDI output and translation from OSC
 - ii. Base class `Instrument` sets up OSC listeners according to what is defined in file.
 - iii. All instruments have two default messages they expect if nothing is specified –
`/<name>/note, ii` and `/<name>/control, ii`
- c. Starts listening for new clients

2. Client starts

- a. Sends `/system/addme, si` to Server (with own hostname and port)
- b. Server responds with a series of `/system/instruments/add, s` messages which list the instruments constructed earlier by name.
- c. Any instruments with possible messages beyond the basic two send `/system/instruments/extend, ssi` to the client which contain the name of the instrument, the pattern for the message and the MIDI status byte to transform.
- d. All instruments send any information they know about themselves in `/system/instruments/note, ss` messages where the first string is the name of the instrument and the second is a note about the instrument, probably defined in the data file.¹
- e. Client uses this data to construct a table of MIDI input to OSC output and prints details about the instruments connected to the server to the console.
- f. Client listens for MIDI input on specified port and translates appropriately.

¹Confusion between `/<name>/note, ii` and `/system/instrument/note, ss` should be avoidable given the different typetags and the `/system` prefix, although it is an unfortunate homonym.