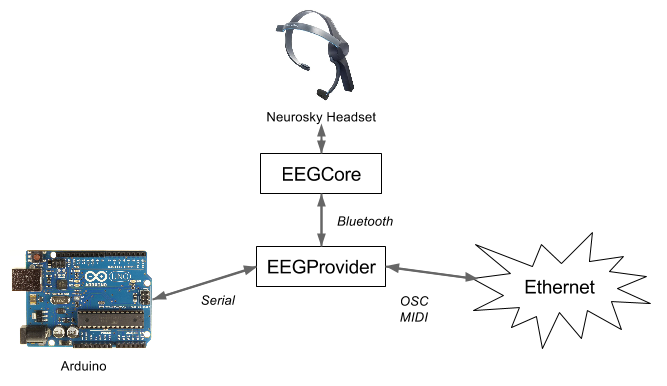
**General**

The premise of this app is simple: connect to an available EEG headset, record/monitor the incoming data, and make all this data available to networked OSC listeners. Listeners can pick and choose which data to use and how.

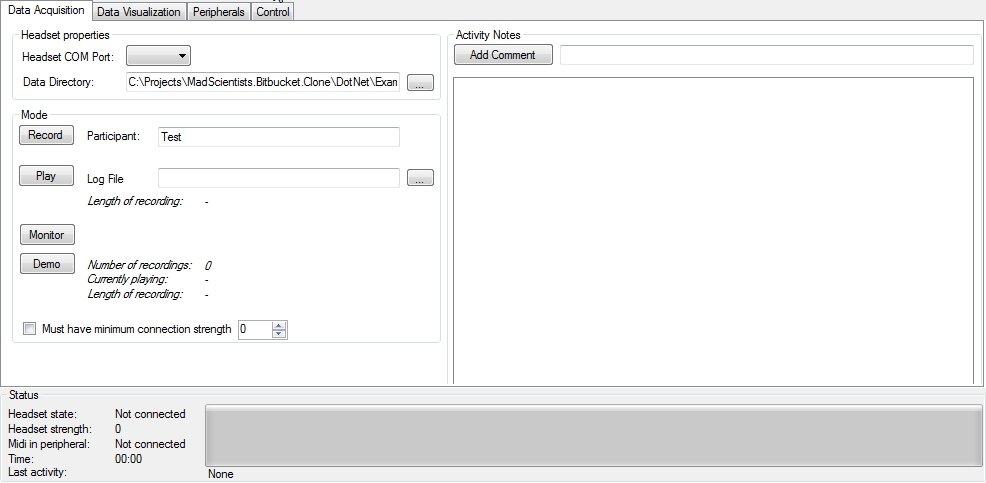
**EEGCore**

This is a c# library built around the neurosky SDK. Currently only the mindwave mobile is supported.

**EEGProvider**

This is a MIDI aware c# application that makes data from the connected headset available via OSC or serial. The main components of this application are data acquisition, data visualization, peripherals (for remote control of app parameters) , and direct control of some of the apps parameters.

**Data acquisition**



Headset properties

Currently only the Mindwave Mobile headset is supported

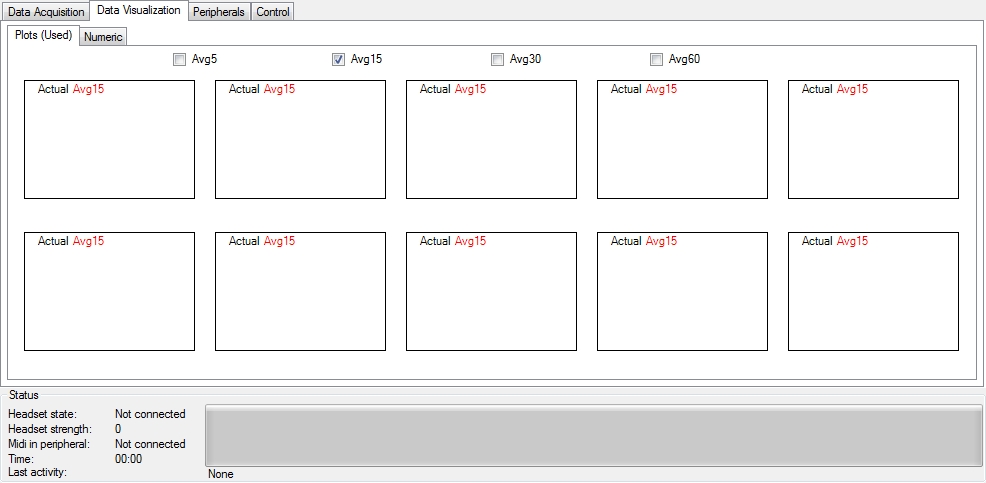
*Mode*

* Record: records and saves a session. Comments can be logged at any time.
* Play: a previously recorded session, along with comments.
* Monitor: monitor the headset without recording.
* Demo: play a random previously recorded session from the data directory.

**Data visualization**

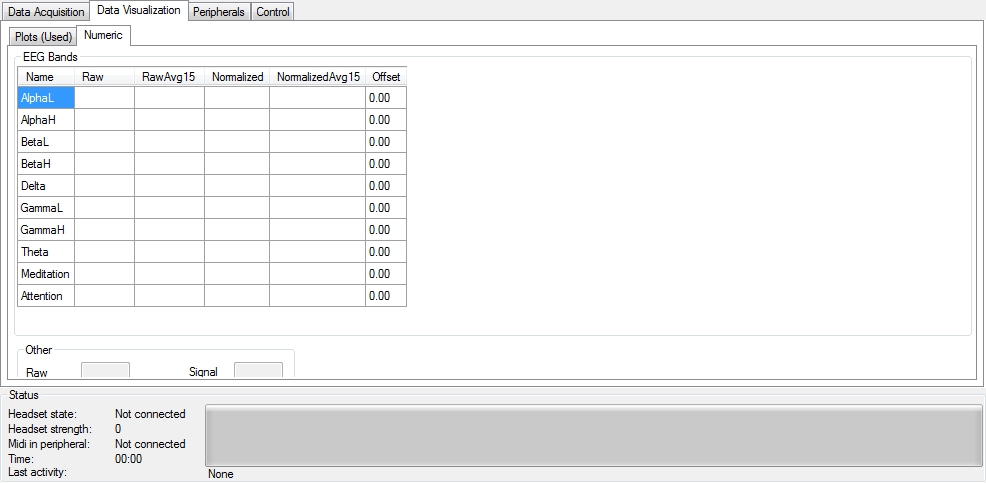
Two perspectives on the EEG data: plots and numeric

*Plots*



Data from each of the channels provided by the headset is normalized and represented in a plot. Each plot shows a graph for current value, and each average of the last 5, 15, 30, 60 seconds.

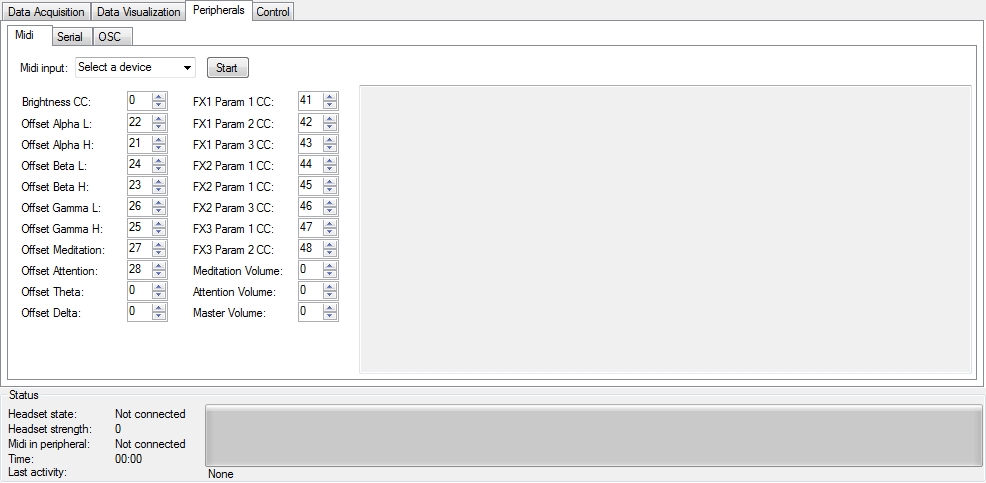
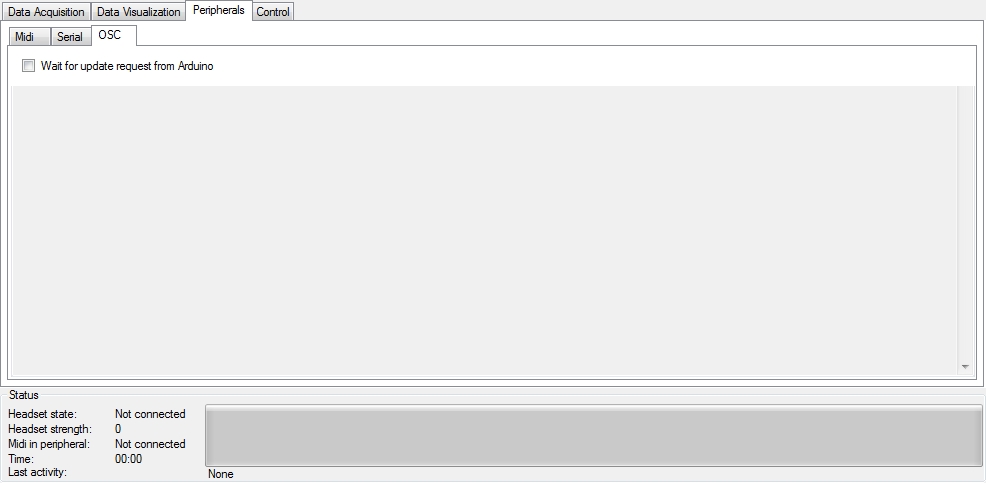
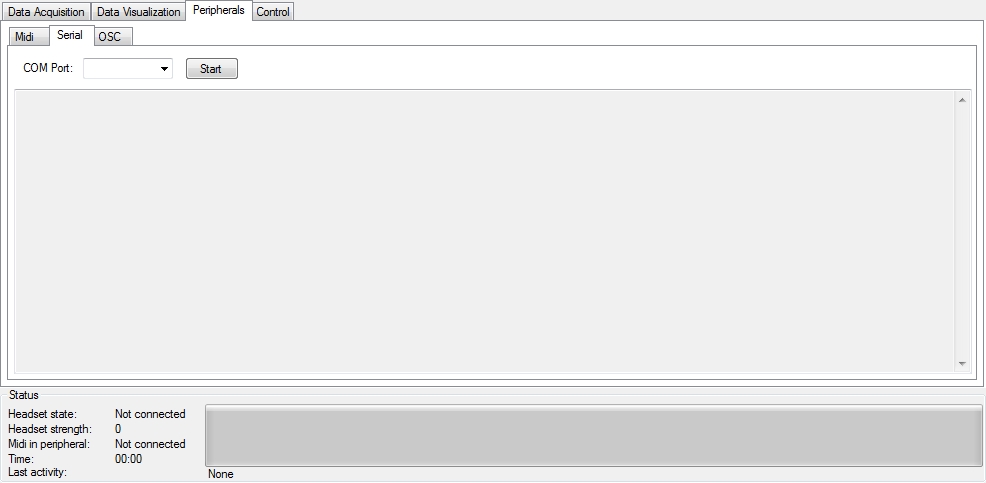
*Numeric*



Each channel can be offset by an amount. This can be controlled via midi or osc

**Peripherals**

Supported peripherals are:

* Midi: use a midi controller to change channel offsets
* OSC: raw, normalized, and averaged data from each available channel is sent out via OSC
* Serial: A connected embedded processor (e.g. Arduino) without ethernet can request data via a 115.2 kb/s serial connection

**OSC Messages**

OSC Messages are sent out about once per second. Data can be raw (as obtained from the headset), consisting of chanals with data in bound and unbound ranges. Data can also be normalized, with all data in the range of 0..100 (making it more useful for some OSC clients)

Raw channel OSC message:

* Address: EEGHeadset/<ChannelName>/Raw
* Raw, non-normalized data. Only channels MEDITATION and ATTENTION are in a known range of 0-100, all other channels are in unknown range
* Available parameters + indices:
  + 0 - Current time
  + 1 - Current value
  + 2 - Average over last 5 sec
  + 3 - Average over last 15 sec
  + 4 - Average over last 30 sec
  + 5 - Average over last 60 sec
  + 6 - Min over last 60 sec
  + 7 - Max over last 60 sec

Normalized channel OSC message:

* Address: EEGHeadset/<ChannelName>/Normalized
* All data in range [0-100]
* Available parameters:
  + 0 - Current time
  + 1 - Current value
  + 2 - Average over last 5 sec
  + 3 - Average over last 15 sec
  + 4 - Average over last 30 sec
  + 5 - Average over last 60 sec
  + 6 - Min over last 60 sec
  + 7 - Max over last 60 sec

Note for OSC clients: It seems best to use NORMALIZED data only; since the data is always in the range [0 100]

Note on OSC data: Only the ‘Meditation’ and ‘Attention’ channels have raw values in the 0..100 range. The raw data in the other channels (Alpha L+H, Beta L+H, Gamma L+H, Theta, Delta) varies a lot per person, so the data can be in any range. To make the data more compatible with use in OSC clients, for these channels EEGProvider calculates derived ‘normalized relative’ values:

* A value of 50 means that the current channel value is *equal to* the average over the last 60 seconds. In other words: channel activity is *unchanged*
* A value of 100 means that the current channel value is *much larger than* the average over the last 60 seconds. In other words: channel activity is *increasing*
* A value of 0 means that the current channel value is *much smaller than* the average over the last 60 seconds. In other words: channel activity is *decreasing*

From the OSC message, CurrentValue and Average5 have relatively a lot of variability; so they change a lot. Average 15, 30, and 60 change much more gradually