Inputs & Outputs: datastructures

BufferClient output

The BufferClient.java file has a function getDoubleData, which is called by the buffer.m file in the get_dat function. This function returns data in the form **double [ ] [ ]**, with a column for each electrode (4) and a row for each sample (63). It needs a starting and an ending point, so the current position in the data should be monitored.
Note: though this one is used in the current application, getFloatData is an option as well. Since the chart library wants floats, we could use this one. However, the preprocessing file below wants doubles, so unless we want to display raw data, double might be more practical.

All the data for the current point in time is sent at once. The place and size of each part of information (header, data, event, success/checking value, error value, version) are hardcoded in BufferClient.java. Each datum is of a predetermined length and position, so the BufferClient can just carve the raw data. Data is sent as barebones binary data, with no specific data types for each datum (header,data,etc.). After receiving the binary data the BufferClient assigns the elements their proper variables. BufferClient also has a wait and clock sync method, which are used to keep the transfer of data in sync.

Regarding the header: The BufferClient.java has a function getHeader, which returns a Header object (also java). Similar functions exist for the other elements. The checking element is special in that it isn't used on its own, but is used as a check when trying to get the header/event or raw data. There is also a method to retrieve raw data, but this not used directly, instead being called by the other getters.

MPAndroidchart input

Short version:
Different kind of plots use different kinds of input. In order to use the plots, the data has to converted into special classes used in the library of the superclass **ChartData**, for example LineData for a line plot. This class needs, in turn, a **DataSet** type input that is specialized for each plot type as well (like LineDataSet).
This DataSet needs as input a List of **Entry** objects. An Entry is made simply with an X and a Y coordinate; these can be made in a loop over the dataset received from the client. Both coordinates need to be a **float** type. The DataSet also needs a String for the legend in the plot; Therefore it's practical to make one DataSet per electrode, since they'll be neatly labelled later if we put them in the same plot.
Note: the X-coordinates should be ordered in ascending order in order to avoid problems.
More detail:
https://github.com/PhilJay/MPAndroidChart/wiki/Setting-Data

for dynamic updating:
https://github.com/PhilJay/MPAndroidChart/wiki/Dynamic-&-Realtime-Data

PreProcClassifier

The preprocclassifier.java file manipulates Matrix objects: This is a class specifically made for this library, and is an extension of the Array2DRowRealMatrix class. Basically it has extra functions to process the data like linear algebra functions and more advanced options like fast fourier transform and detrending. It's a 2D matrix with **double** types; it can be instantiated with a double [ ] [ ], which is convenient. PreProcClassifier is not used anywhere in the chain we are running now (our buffer, client and server implementations), and is mainly used by the nascent java sigViewer implementation. It's likely that we can borrow parts of the code however.