MindSetUWP

MindWave EEG Headset library for Universal Windows Platform

Ondrej Rehacek*

April 26, 2016

*reh0063@vsb.cz

A NeuroSky MindWave and MindWave Mobile EEG headset library for Windows 10 universal applications ecosystem. A part of my bachelor thesis at VŠB - Technical University of Ostrava. Library is designed for use with MindWave Mobile Bluetooth headset paired with Windows 10 device. You can easily receive EEG readings to any UWP application and use them.

Basics

Library can be used for any .NET Windows 10 Universal application running on every Windows 10 device that have a Bluetooth adapter installed with appropriate system drivers. The EEG headset must be paired in the system Bluetooth menu prior to the use of the library itself.

1 Creating the instance of MindSetConnection

After you successfully imported the library into your application references, you can proceed by declaring the instance of MindSetConnection - a class that represents a connection to a EEG headset.

```
// Create a new instance of MindSetConnection class.
private MindSetConnection MyHeadset = new MindSetConnection();
```

2 Establishing Bluetooth connection

Now we can use a method "ConnectBluetooth" followed by supplying the Bluetooth name of the headset (In most cases the name is "MindWave Mobile") to initiate the packet connection to the EEG headset - that is all, easy as that - now we are receiving the formatted EEG readings from the headset.

```
private void EstConnBtn_Click(object sender, RoutedEventArgs e)
        {
            //Initiate bluetooth connection to a headset named
                "MindWave Mobile"
            MyHeadset.ConnectBluetooth("MindWave Mobile");
        }
```

3 Working with EEG data (MindsetDataStruct)

MindSet headset transmits ThinkGear Data Values, encoded within ThinkGear Packets, as a serial stream of bytes over Bluetooth via a standard Bluetooth

Serial Port Profile (SPP). MindSetUWP library process all of this packets and manages Bluetooth connection with headset for you, so you are no longer required to do any extra work. Before we start working with the data , lets talk about all measurements that headset send to your application and this library supports.

Table 1: EEG Data

Variable Name	Value Range	Type	Required
Delta	0.5 - 2.75Hz	Integer	X
Theta	3.5 - 6.75Hz	Integer	X
AlphaLow	7.5 - 9.25Hz	Integer	X
AlphaHigh	10 - 11.75Hz	Integer	X
BetaLow	13 - 16.75Hz	Integer	X
BetaHigh	18 - 29.75Hz	Integer	X
GammaLow	31 - 39.75Hz	Integer	X
GammaMid	41 - 49.75Hz	Integer	X
esense.attention	0 - 100	Integer	
esense.meditation	0 - 100	Integer	
TimeStamp		DateTime	X
Quality	0-255	Integer	X

All of the variables in the table represent particular EEG data received from the MindWave headset, they are publicly accessible variables in the instance you created and connected to a headset using "RealtimeData" class.

```
private void ShowDataBtn_Click(object sender, RoutedEventArgs e)
    {
        Debug.Write(MyHeadset.RealtimeData.AlphaHigh); //Writes
            AlpaHigh data into debug console.
        Debug.Write(MyHeadset.RealtimeData.AlphaLow); //Writes
            AlpaLow data into debug console.
    }
```

This is of course the most basic example of how you can work with the EEG data from the headset, you can use and work with them like with any public variable in your application. They are updated constantly while the Bluetooth headset is connected.

3.1 Alternative methods of working with EEG data.

MindSetUWP library also contains alternative three methods how you can work with received EEG data. These three methods are called **AllToIntArray** witch returns an integer array with all of the EEG readings. Alternatively you can use **AllToArray** wich returns array of all EEG readings

formatted to a string values.The position of EEG data in this arrays are the same as in the Table 1.1 in this document. At last you can also use **AllToString** that returns a long nicely formatted text string of all EEG values.

```
    private void WrtDataBtn_Click(object sender, RoutedEventArgs e)
       {
           Debug.Write(MyHeadset.RealtimeData.AllToString());
               //Writes all of the data into debug console.

           int[] TableofEEG =
               MyHeadset.RealtimeData.AllToIntArray(); //Writes all
               of the data into integer array.
       }
```

4 Handling Events

Library offers multiple events that you can use. We need to work StatusUpdateHandler that is included in library to handle these events and project them into your application.

```
//An OnConnected void will be called when the OnConnected event is
    fired
MyHeadset.Connected += new
    MindSetConnection.StatusUpdateHandler(OnConnected);


//An NoHeadsetFound function will be called when the NoHeadset
    event is fired
MyHeadset.NoHeadset += new
    MindSetConnection.StatusUpdateHandler(NoHeadsetFound);
```

4.1 Available events

OnConnected	Connection to a headset is established.
Connecting	The connection to a headset is in progress, but not yet established.
PacketRecieved	A EEG data packet is received from headset.
Disconnected	Connection to a headset is lost.
NoHeadset	No MindWave Mobile Bluetooth headset is in range or not paired with the device.
QualityChange	The quality of the data reading has changed in the last packet.
ParseFail	Parsing of the data has failed (a corrupted packet has been received)
Recording	A recording of the incoming packets has started.
RecordingStopped	A recording of the incoming packets has been paused or stopped.

5 Recording EEG data

Library also includes internal recording of the packets. Using this we can work with EEG data from longer time period and analyse them further in your application. The recording is also deeply configurable.

5.1 Recording Density

We can set the RecordDensity - a integer number that identifies how many received packets to record (For example if the RecordingDensity is set to "1" every received packet is recorded, if the RecordingDensity is set to "10" an every tenth packet is recorded).

5.2 Recording Filtering

To obtain only reliable data to your recording, we can use RecordingFiltering. This filter sets the recording to record only data where Quality of received EEG data is reliable and eSense data is received. Please note that if the recording filtering is turned off , every received data is recorded, even if the headset is misplaced on user head, or packet data are not reliable.

5.3 Usage

To start recording we need to use only one line of code, please note the arguments explained in the commented line:

```
//MyHeadset.StartRecording(RecordingDensity, RecordingFiltering);
MyHeadset.StartRecording(1, true);
```

Same as when we want to pause the recording:

```
MyHeadset.StopRecording();
```

Please note that this command only pauses the recording and if we start the recording again, the new data will be added to new one.

If we want to clear already recorded data on the other hand, use:

```
MyHeadset.ClearRecordingData();
```

5.4 Working with the recorded data

We can work with recorded data same as with the real time data. Only exception is that the recorded data are stored in the array. To obtain the recorded data saved in the memory use the function called "RecordingToArray".

A great example is this FOR loop, that prints every received packet in string format into debug window.

```
MindsetDataStruct[] MyRecordedData = MyHeadset.RecordingToArray();

          for (int i = 0; i < MyRecordedData.Length; i++)
          {
              Debug.WriteLine(MyRecordedData[i].AllToString);
          }
```