Delsys API 2.x

AeroPy Programmer's Guide

MAN-045-1-0

Contents

1 Important Information

1.1 Intended Use

The Delsys API and associated AeroPy software is a development tool to be used in conjunction with the Trigno Wireless Biofeedback System. The API is not intended to perform assessment or diagnostic procedures. It is intended to be used as a software component of a third-party software application. The function of the API is to manage the transfer of data from the Trigno System to third-party software applications, and is designed to work exclusively with the Trigno System. It is designed to facilitate communication with the Trigno System from a third-party software application.

1.2 Technical Service and Support

For information and assistance, please visit:

www.delsys.com

Contact us:

E-mail: support@delsys.com
Telephone: (508) 545 8200

1.3 Device Information

Please see the Trigno Wireless Biofeedback System User Guide for information on the device.

1.4 System Requirements

1.4.1 Windows RF Developer System Requirements:

- Windows 7 and above
- Microsoft Visual Studio 2015 or later

1.5 Documentation Reference

This document is meant only to provide an overview of the functions available in the AeroPy layer of the Delsys API. It is recommended you follow this guide and refer to the Delsys-Python-Demo example which can be found on the Delsys API website should you need more context. Additional information on the API can be found in the following references:

- MAN-032 API Quick Start Guide
- MAN-033 API User Guide

2 AeroPy Functions

The following functions demonstrate all the ways in which to interact with the Delsys API through the Python layer. For more information on the full Delsys API, please see the API User's Guide.

2.1 Connecting to the Trigno Base:

```
public void ValidateBase(string key, string license,string dataType):
```

Initial call to the Trigno Base. Sets up a connection to the base using the user's key and license strings. Initializes the Trigno Base in the desired data source configuration given by the dataType variable (e.g "RF","sim")

2.2 Connection and Searching for Sensor Functions (RF connection):

```
public System.Threading.Tasks.Task ScanSensors():
```

Main method for beginning sensor scan. Will scan for unpaired sensors via RF

```
public void PairSensors():
```

This sets the base into pairing mode, allowing for a user to pair a new sensor to the base to be found in the scan. Call scan after pairing to ensure that pipeline is set up properly for streaming

```
public bool ConnectSensors():
```

Connects and initializes all the sensors that have been currently found in the scan
If you only want to connect a specific sensor, use SelectSensor method

```
public void SelectSensors(int sensorNum):
```

Selects and initializes the sensor at index sensorNum

2.3 Sensor Streaming(RF connection):

```
public void ClearSensorList():
```

Clears the internal list of found sensors, call this first to refresh internal state before data streaming

```
public void
AddSensortoList(DelsysAPI.Components.TrignoRf.SensorTrignoRf
SelectedSensor):
```

Adds a sensor the internal list to give to stream data. Use the GetSensorObject() function call to get the internal TrignoRf sensor object of the desired sensor. See example code for more information.

```
public DelsysAPI.Components.TrignoRf.SensorTrignoRf GetSensorObject(int
sensorNo):
```

Returns the sensor object of the sensor at the index sensorNo

```
public DelsysAPI.Transforms.Transform CreateTransform(string type):
```

Method to create a basic transform to be passed into streamData. This returns a DelsysApi Transform object to be passed into the stream data function call.

`public void StreamData(List<int> index, DelsysAPI.Transforms.Transform newTransform, int frameThroughput):`
This calls the stream data function. It will stream data from the sensor indices passed in index at a frame throughput specified by the user, default is set to 2 frames. This tells the Trigno base to begin streaming data to the data buffers. Use CheckDataQueue() function to determine if new data is available and PollData() function to retrieve the data.

`public void StopData():`
Tells the Trigno base to stop the data stream and clean up the data pipeline after data streaming has completed.

2.4 Data Management (RF connection):

`public bool CheckDataQueue():`
This returns a flag that denotes whether there is new data in the internal data buffer that is ready to be extracted. Use the PollData() function to return the data structure.

`public List<List<double>>[] PollData():`
This retrieves the data from the data buffer fed by the StreamData data stream. Returns data the form of packets, length denoted by the currently set sample mode of the sensor. Each row in the data packet contains the returned data for a different channel in the sensor (eg. EMG, IMU, ACC, GYRO, etc.).

2.5 Helper Functions (RF connection):

`public int GetTotalPackets():`
Returns the total number of data packets collected from the current streaming session

`public List<string> GetSampleMode():`
Getter for the current samplemodes of the sensors

`public void SetSampleMode(int componentNum, string sampleMode)`
Sets the sample mode for the sensors. Will set the sensor at index componentNum to the mode given by sampleMode

`public string[] ListSensorNames():`
This will return a string array of the current sensors found

`public string[] ListSensorModes(int sensorSelected):`
This will list out the list of different sensor modes for the sensor at index sensorSelected