Delsys API

AeroPy Programmer's Guide

MAN-045-1-1

Contents

1 Important Information 3
1.1 Intended Use 3
1.2 Technical Service and Support 3
1.3 Device Information 3
1.4 System Requirements 3
1.5 Documentation Reference 3
2 AeroPy Functions 4
2.1 Connecting to the Trigno Base/Lite 4
2.2 Sensor Management 4
2.3 Sensor Configuration 5
2.4 Data Collection Management 5
2.5 Helper Functions 6

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 medical assessments 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. Any claims regarding the intended use of the 3rd party software are the sole responsibility of the 3rd party software manufacturer.

1.2 Technical Service and Support

For information and assistance, please visit ***www.delsys.com*** or contact us at

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

1.3 Device Information

Please see the Trigno Wireless Biofeedback System User Guide for additional information pertaining to the device.

1.4 System Requirements

- Windows 10 and above
- Microsoft Visual Studio 2017 or later

1.5 Documentation Reference

This document is only meant to provide the user with an overview of the functions available in the AeroPy layer of the Delsys API. It is recommended that the user follow this guide and refer to the Delsys-Python-Demo example which can be found on the Delsys API website should further context be required. 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/Lite

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

Initial call to the Trigno Base. Sets up a connection to the base using the user's key and license strings.

2.2 Sensor Management

```
public Task ScanSensors()
```

Scan for previously paired sensors (RF). Pipeline must be in the Off or Connected State to run this command.

```
public void PairSensors()
```

This sets the base into pairing mode, allowing for a user to pair a new sensor to the base. Pipeline must be in the Off or Connected State to run this command.

```
public bool SelectAllSensors()
```

Selects all the sensors that have been found in the scan. If you only want to select a specific sensor, use SelectSensor method.

```
public void SelectSensor(int sensorNum)
```

Selects the sensor for streaming at index sensorNum. Use SelectAllSensors() method to select all scanned sensors.

```
public SensorTrignoRf GetSensorObject(int sensorNum)
```

Get the sensor object of the sensor at the index sensorNo.

```
public List<string> GetAllSampleModes()
```

Get all of the sample modes that the sensors are currently set to.

```
public void SetSampleMode(int componentNum, string sampleMode)
```

Sets the sample mode for the given sensor. Will set the sensor at index componentNum to the mode given by sampleMode

```
public string[] GetSensorNames()
```

Get all of the sample modes that the sensors are currently set to.

```
public string[] AvailibleSensorModes(int sensorSelected)
```

Get all of the sample modes that the sensors are currently set to.

2.3 Sensor Configuration

```
public void Configure()
```

Default Configure method - Will configure pipeline for raw data output on all scanned sensors. Pipeline must be in the Off or Connected State. Pipeline will transition to Armed.

```
public void AddSensortoList(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 SensorTrignoRf GetSensorObject(int sensorNo)
```

Returns the sensor object of the sensor at the index sensorNo

```
public Transform CreateTransform(string type)
```

Create a basic transform to be passed into streamData method.

2.4 Data Collection Management

```
public void Start()
```

Starts Data Stream - Pipeline must be in the Armed state. Pipeline will transition to Running.

```
public bool CheckDataQueue()
```

Called while in the Running state (live data collection) Returns true if there is new data in the internal data buffer that is ready to be extracted. If true, use `PollData()` to return the data structure.

```
public bool Dictionary<Guid, List<double>> PollData()
```

This retrieves the data from the data buffer after the Start() method is called. Each time this method is called it will return the data, then clear the internal data queue. The return type is a dictionary output where channel GUID is the key and the channel data is the value.

```
public void Stop()
```

Tells the Trigno base to stop the data stream and clean up the data pipeline after data streaming has completed.

```
public void ResetPipeline()
```

Resets (disarms) Pipeline - Pipeline must be in the Armed state. Pipeline will transition to Connected (Allows for users to call Scan/Pair after a collection is stopped).

2.5 Helper Functions

```
public string GetPipelineState()
```
Returns the current state of the RF pipeline.

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