EScribe

www.evolvapor.com

Table of Contents

Device Monitor

The EScribe Device Monitor can be used to monitor all aspects of an Evolv DNA vape. Device Monitor features data logging, scripting, statistics, and can graph power, temperature, current, voltage, and more in real time.

Python Scripting

Device Monitor features built-in IronPython scripting. This can be used for scientific study, automation, and testing.

To run a script, click the Diagnostics button in Device Monitor, and select Advanced from the popup menu. Then click Run Script.

If you run scripts often, go to the Options menu in EScribe, click User Interface, and select either Researcher or Manufacturer mode. In either of these modes, Run Script will be on the Diagnostics menu, and will not have a warning message. This will save you time.

Commands

As a reference, here are the functions available through the Python scripting:

`ECig.Puff(time)`
Time is in seconds.

`ECig.ClearTracking()`
Removes all variables from the graph.

`ECig.Track(state)`
Adds a variable to the graph. Tracked variables update more quickly.

`ECig.IsTracked(state)`
Returns True if the variable is on the graph.

`ECig.StopTracking(state)`
Removes a variable from the graph.

`Recorder.Record(filename)`
Starts recording a CSV file. You can't change which variables are on the graph while recording.

`Recorder.StopRecording()`
Stops recording a CSV file.

`Recorder.IsRecording`
Returns True if the Device Monitor is recording.

`Serial.Open(portName)`
`Serial.Open(portName, baudRate)`
Opens a serial port. If you do not specify a baud rate, the default is 9600 baud.
You can use the returned object to communicate with a serial device.

`UI.Message(text)`
Pops up a message box.

Variables

The `ECig` object also has an indexer.

```
x = ECig[state]

ECig[state] = y
```

States are strings representing the variables in the device monitor.
For example, `ECig['Power Set']`, `ECig['Current']`, `ECig['Battery Cell 1']`.

Here is a list of available variables:

```
Power
Power Set
Temperature
Temperature Set
Current
Voltage
Cold Ohms
Live Ohms
Puffs
Last Puff Energy
Last Puff Power
Last Puff Temperature
Last Puff Peak Temperature
Last Puff Time
Battery Charge
Battery Pack
Battery Cell 1
Battery Cell 2
Battery Cell 3
USB Power
USB Current
USB Voltage
```

Example 1

This sample script records current, voltage, and power while doing 3-second puffs ranging from 5 to 30 watts.

```
import time

ECig.ClearTracking()
ECig.Track('Current')
ECig.Track('Voltage')
ECig.Track('Power')
Recorder.Record(r'C:\Users\James\Test.csv')

time.sleep(1)
for p in xrange(5, 35, 5):
  ECig['Power Set'] = p
  ECig.Puff(3)
  time.sleep(6)

Recorder.StopRecording()
```

Example 2

This sample script communicates with a Dimension Engineering Sabertooth 2x32 motor driver on COM10, turning on power output P1. (This can be useful for controlling air cylinders, relays, etc.)

Then, it does a 5-second puff at 20 watts. Finally, it turns off power output P1, and reads back the puff energy and displays it in a message box.

```
import time

ECig.Track('Last Puff Energy')
motor = Serial.Open('COM10', 9600)
motor.WriteLine('P1: 2047')
time.sleep(1)

ECig['Power Set'] = 20;
ECig.Puff(5)
time.sleep(5)

time.sleep(1)
motor.WriteLine('P1: -2047')

energy = ECig['Last Puff Energy']
ECig.StopTracking('Last Puff Energy')

UI.Message('The last puff energy was {0} milliwatt-hours.'.format(energy))
```