NeuLog API
Doc version 10, API version 6

This Application Programming Interface (API) specifies how any software should interact with NeuLog sensors.

The API is based on HTTP protocol and can be accessed from any software like C, C++, Python, JS, Java and many more. Also, Microsoft Word™ and Acrobat™ can be used to control the sensors.

Minimum requirements:

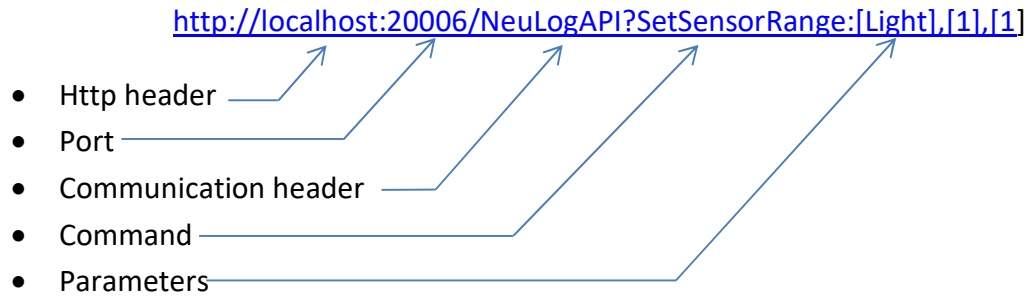
- NeuLog sensor(s)
- NeuLog USB module
- NeuLog API software

Getting started:

- Install the NeuLog API software on your PC/MAC/Linux
- Run the NeuLog API
- The API port number will appear on the software screen
- Run your software and contact the NeuLog API via the http “GET” command

The NeuLog API - general:

- The API is based on HTTP protocol and uses the standard “GET” type communication
- Each communication must be initiated by the user
- Each communication looks like this:



- The received command and the result will be shown on the API screen.
- Make sure you are using the right port number.
- The answer is a JSON string such as: {"SetSensorRange":"True"}
- This user manual appears when the “help” button ⓘ on the NeuLog API software is clicked.
- Do not use any spaces in the command string.

The API commands:

Command	Description	Parameters	Return
GetServerVersion	Get the server’s software version		{"GetServerVersion":"4.4.4"}
GetSeverStatus	The response could be: Ready, USB missing, Recording		{"GetServerStatus":"Ready"}
GetSensorValue:[sen1.type],[sen1.ID], ...,[seni.type],[seni.ID] Example: GetSensorValue:[Sound],[1],[Light],[2]	Reads one value from the sensor	1. Type & 2. ID of the sensor In the example it is: Sound sensor ID=1 and Light ID=2	{"GetSensorValue":[67.3,345] }
ResetSensor:[sen.type],[sen.ID] Example: ResetSensor:[PH],[2]	Reset the sensor value. Use in force, oxygen and a few other sensors	1. type & 2. ID of the sensor In the example it is: pH sensor ID=2	{"CalibSensor":"True"}
SetPositiveDirection:[sen.type],[sen.ID],[sen.dir] Example: SetPositiveDirection:[Force],[1],[1]	Set the force sensor positive values	1. type & 2. ID of the sensor 3. Direction (1=push positive, 2=pull positive) In the example it is Force sensor ID=1 and the direction is push positive	{"SetPositiveDirection":"True "}

StartExperiment:[sen1.type],[sen1.ID],[sen2.type],[sen2.ID],…[seni.type],[seni.ID],[rate],[samples] Example: StartExperiment:[Temperature],[1],[Temperature],[2],[Temperature],[3],[8],[101] <u>*NOTE:</u> The Photo gate sensor experiments works in a different way. See explanation below.	Start experiment with the specific parameters.	1. Sensors 2. Rate index: 1 = 10000 per second 2 = 3000 per second 3 = 2000 per second 4 = 1000 per second 5 = 100 per second 6 = 50 per second 7 = 20 per second 8 = 10 per second 9 = 5 per second 10 = 2 per second 11 = 1 per second 12 = 30 per minute 13 = 15 per minute 14 = 6 per minute 15 = 2 per minute 16 = 1 per minute 17 = 30 per hour 18 = 15 per hour 19 = 6 per hour 20 = 2 per hour 21 = 1 per hour 3. Samples (number) In the example there are 3 temperature sensors; the rate is 10 samples per second, the duration is 101 samples (10 sec)	{"StartExperiment":"True"}
StopExperiment			{"StopExperiment":"True"}
GetExperimentSamples:[sen1.typpr],[sen1.ID],…[seni.type,[seni.ID] Example: GetExperimentSamples:[Light],[1],[Sound],[1]	Get all the samples from specific sensor/s of the current experiment (while running or after running)	1. Sensors	{"GetExperimentSamples":[["Light",1,20,21,22,21,20,…],["Sound",1,68.3,88.8,…]]}
SetSensorRange:[sen.type],[sen.ID],[range] Example: SetSensorRange:[Light],[1],[2]		1. Type 2. Id 3. New range	{"SetSensorRange":"True"}
SetRFID:[New ID] Example: SetRFID:[3]		1. New ID	{"SetRFID":"True"}
SetSensorsID:[NewID] Example: SetSensorsID:[3]	Set all connected sensors to the new ID	1. New ID In the example the new ID is 3	{"SetSensorsID":"True"}

Photo Gate experiments:

The photogate sensor experiments are different from all other sensors. The experiment results are not a list of values taking in fix time, but a single value that measured when something interrupt the gate.

Photo gate has 6 different kinds of experiments, and for each one you need to send the correct information to the API:

1. Velocity with single gate
    a. StartGateExp:[1],[ID],[X]
    b. Where ID is the sensors ID (1-9), and X is the interrupter width in mm.
2. Acceleration with single gate
    a. StartGateExp:[2],[ID],[X],[Y]
    b. Where ID is the sensors ID (1-9), X is the first "U" shape interrupter width in mm, and Y is the second "U" interrupter width in mm.
3. Acceleration with two gates
    a. StartGateExp:[3],[ID1],[ID2],[X]
    b. Where ID1 is the first sensor ID (1-9), ID2 is the second sensor ID (1-9), X is the interrupter width in mm.
4. Velocity with two gates
    a. StartGateExp:[4],[ID1],[ID2],[X1],[X2],[M1],[M2]
    b. Where ID1 is the first sensors ID (1-9), ID2 is the second sensor ID (1-9), X1 is the first interrupter width in mm, X2 is the second interrupter width in mm, M1 is the first cart mass in grams, and M2 is the second cart mass in grams.
5. Delta between two gates
    a. StartGateExp:[5],[ID1],[ID2],[X]
    b. Where ID1 is the first sensors ID (1-9), ID2 is the second sensor ID (1-9), X is the interrupter width in mm.
6. Time with timing card
    a. StartGateExp:[6],[ID],[DURATION]
    b. Where ID is the sensors ID (1-9), and the DURATION indicate the experiment duration in miliseconds.
    c. DURATION options are: 25,50,150,300,1000,2000,5000

To see the result, you nead to send: "ReadGateSamples"

And the answer will be according to the experiment type:

1. {"ReadGateSamples":"0.1143~0.87489063867"}
    a. Values are: time, velocity
2. {"ReadGateSamples":"0.511969545672"}
    a. Value is: acceleration
3. {"ReadGateSamples":"8.32146826343"}
    a. Value is: acceleration

4. {"ReadGateSamples":"2.00400801603~1.19047619048~0.100200400802~0.059523809523 8"}
    a. Values are: velocity 1, velocity 2, momentum 1, momentum 2
5. {"ReadGateSamples":"0.1019"}
    a. Value is: time
6. {"ReadGateSamples":"0.0115~0.01~0.009~0.01"}
    a. Values are: times of each state in the gate.

Few examples working with Sound and Light sensor:

http://localhost:22006/NeuLogAPI?GetServerVersion

http://localhost:22006/NeuLogAPI?GetSeverStatus

http://localhost:22006/NeuLogAPI?GetSensorValue:[Sound],[1],[Light],[1]

http://localhost:22006/NeuLogAPI?StartExperiment:[Sound],[1],[Light],[1],[8],[101]

http://localhost:22006/NeuLogAPI?StopExperiment

http://localhost:22006/NeuLogAPI?GetExperimentSamples

http://localhost:22006/NeuLogAPI?SetSensorRange:[Light],[1],[2]

Sensor list:

'Temperature', 'Light', 'Voltage', 'Current', 'PH', 'Oxygen', 'PhotoGate', 'Pulse', 'Force', 'Sound', 'Humidity', 'Pressure', 'Motion', 'Magtnetic', 'Conductivity', 'GSR', 'CO2', 'Barometer', 'Rotary', 'Acceleration', 'Spirometer', 'SoilMoisture', 'Turbidity', 'UVB', 'EKG', 'Colorimeter', 'DropCounter', 'FlowRate', 'ForcePlate', 'BloodPressure', 'Salinity', 'UVA', 'SurfaceTemp', 'WideRangeTemp', 'InfraredThermometer', 'Respiration', 'HandDynamometer', 'Calcium', 'Chloride', 'Ammonium', 'Nitrate', 'Anemometer', 'GPS', 'Gyroscope', 'DewPoint', 'Charge'