Έλεγχος με Η/Υ – Ενεργοποιητές

Μεταπτυχιακό Τμήματος Μηχανικών Πληροφορικής – Κατεύθυνσης Μηχανικών Η/Υ «Μηχανικών Η/Υ και Συστημάτων – Ευφυή Συστήματα και IoT»

Διδάσκων: Παπαϊωάννου Αθανάσιος

Επιμέλεια Απαντήσεων : Λόκας Δημήτριος

1η Σειρά Ασκήσεων

**ΑΣΚΗΣΗ 2**

H πλατφόρμα του sitewhere έχει γραφικό περιβάλλον χρήσης.

Στην παρούσα άσκηση θα χρειαστεί να εκτελέσουμε **3 εφαρμογές** από CLI (Command Line των Windows) για να δημιουργήσουμε το Smart Home με εγκατάσταση 3 αισθητήρων Arduino prototype.

1. **Εκτέλεση MONGODB**

To MongoDB είναι μια πλατφόρμα τοπικής βάσης δεδομένων, η οποία επιτελεί το ρόλο της offline database. Εκεί γίνονται store τα δεδομένα του γραφικού περιβάλλοντος του SiteWhere όπως και backup ολων των Events που καταγράφηκαν και αφορούν το Smart Deployment μας. Χρησιμοποιεί την θύρα επικοινωνίας **27017**.Στο συγκεκριμένο παράδειγμα κάνουμε χρήση της 3.6 έκδοσης.

Εκτέλεση

|  |
| --- |
|  |

Η τοπική βάση δεδομένων θα αποθηκεύει πληροφορίες στον υποκατάλογο

**C:\Mongodbdata .**

1. **Εκτέλεση HiveMQ**

Το HiveMQ είναι η εφαρμογή η οποία εκτελεί χρέη MQTT Broker.Δηλαδή προωθεί στο Cloud σε σμήνος υπολογιστών (hive) πολλαπλά μηνύματα από αισθητήρες σε WSNs.Επικοινωνεί στη θύρα **8000** όπως ορίσαμε ως TCP LISTENER.Χρήση της έκδοσης 3.3.3 της εφαρμογής.

|  |
| --- |
|  |

1. **Εκτέλεση SiteWhere**

To sitewhere είναι η εφαρμογή που χρησιμοποιούμε για να δημιουργήσουμε το smart Home της παρούσας άσκησης.

Δημιουργεί virtual οποιοδήποτε Smart Deployment με δυνατότητες **αντιστοίχησης** σε αυτό αισθητήρων και τοπικών σταθμών WSNs που βρίσκονται σε ακίνητες (stationary) ή κινητές (mobile) πλατφόρμες.

Στο SiteWhere μπορούμε να λαμβάνουμε Real Time Events (Alerts,Measurements και locations) από οποιαδήποτε Asset και να έχουμε εποπτεία σε πραγματικό χρόνο.

Το γραφικό περιβάλλον τρέχει από προεπιλογή στην θύρα **8080.**

Κάναμε χρήση της έκδοσης server 1.13.0 της εφαρμογής

|  |
| --- |
|  |

Στη συνέχεια επισκεπτόμαστε στο διακομιστή την αρχική σελίδα της εφαρμογής

<http://localhost:8080/sitewhere/admin> συμπληρώνουμε Admin,password

και βρισκόμαστε στην αρχική σελίδα.

Εκεί μπορεί να μην υπάρχει κανένα default Site η κάποιο τυπικό ανάλογα με την έκδοση της εφαρμογής.

* **Δημιουργία Smart Home**

Στην αρχική <http://localhost:8080/sitewhere/admin/default/sites/list.html>

Επιλέγουμε +Add New Site

Στο παράθυρο που εμφανίζεται

|  |
| --- |
|  |

Συμπληρώνω Site Name,Description και διεύθυνση εικόνας URL για να έχουμε εικονίδιο

Στη καρτέλα Map information συμπληρώνω συντεταγμένες που θα βρίσκεται το smart home μου Center Latitude, Center Longitude η με choose location το χωροθετώ πάνω στο χάρτη.

|  |
| --- |
|  |

|  |
| --- |
|  |

Στην συνέχεια επιλέγω assets.

Στην καρτέλα <http://localhost:8080/sitewhere/admin/default/assets/categories.html>

|  |
| --- |
| +add new asset category |

Συμπληρώνω όλα τα στοιχεία.hw id,category type και name

Επιλέγω το νέο asset category που δημιουργήθηκε στην ίδια καρτέλα πατώντας view asset category.

|  |
| --- |
|  |

Eδω θα βρίσκονται όλα τα asset της συγκεκριμένης κατηγορίας. Στην παρούσα άσκηση δημιούργησα asset category Arduino 1

+add new

|  |
| --- |
|  |

Συμπληρώνω hardware id των 3 arduino μου τα οποία και σημειώνω.

A SKU (Stock Keeping Unit) κωδικοί γρήγορης καταχώρησης

Image URL φωτογραφία online για εύκολη εποπτία

Name SmartSensorsKitchen

hardware id cf5dd611-9092-448a-ac99-6f02b5c4de2d

sku ARD-3BL

SmartSensorsLivingRoom

hardware id cf5dd611-9092-448a-ac99-6f02b5c4de2b

sku ARD-3

SmartSensorsRoom

hardware id cf5dd611-9092-448a-ac99-6f02b5c4de2e

sku ARD-2BK

Μετά πηγαίνω σε <http://localhost:8080/sitewhere/admin/default/specifications/list.html>

Θα ορίσω τα Specifications των asset μου.

+Αdd New Specification

|  |
| --- |
|  |

Επιλέγω device type, type, asset provider και βάζω Νame

‘Ορισα τα 3 specs

|  |
| --- |
|  |

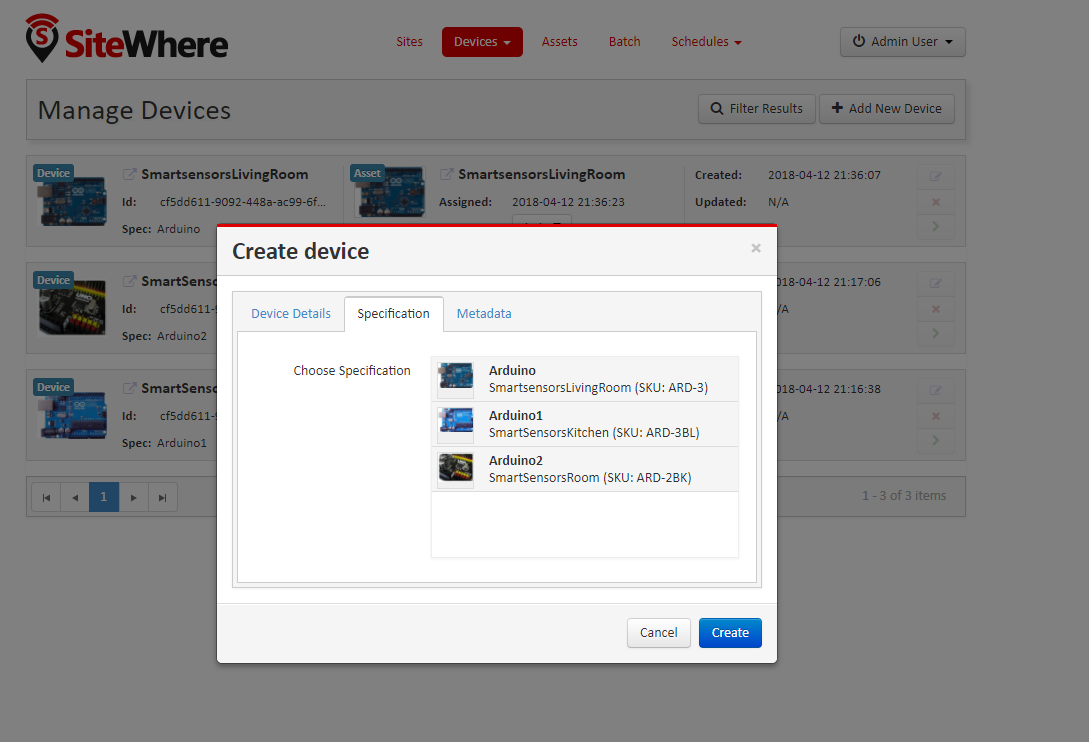
Στη συνέχεια επιλέγω Devices

<http://localhost:8080/sitewhere/admin/default/devices/list.html>

+add New Device

**Καρτελα Specification**

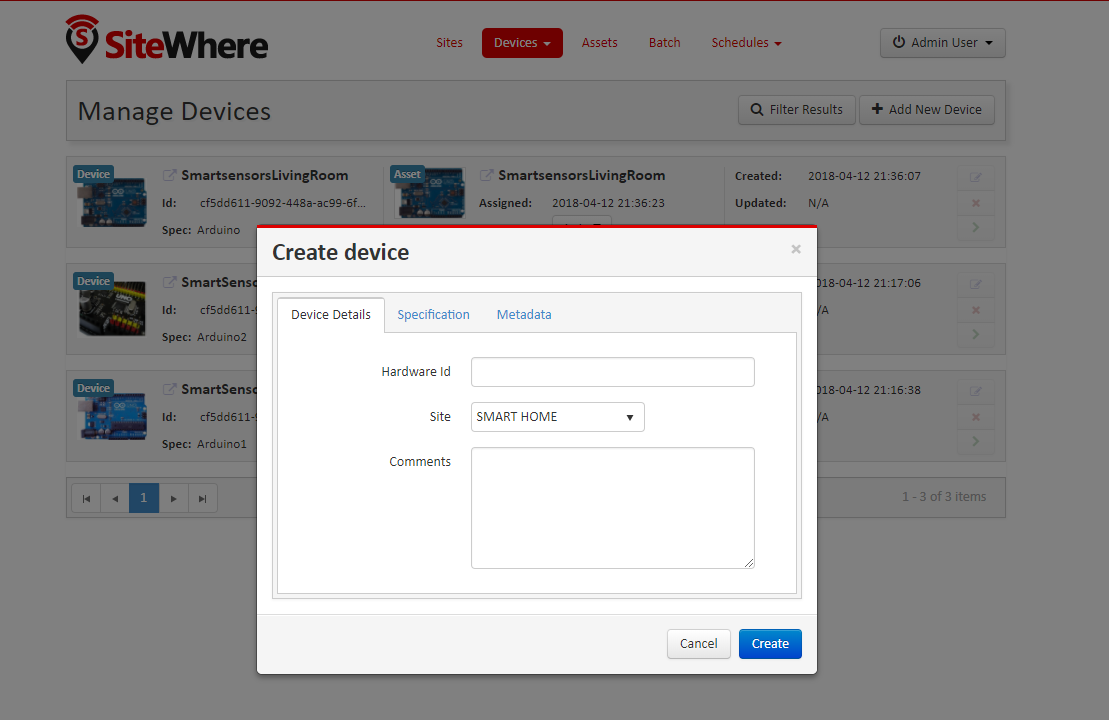
Πάμε και επιλέγουμε το spec



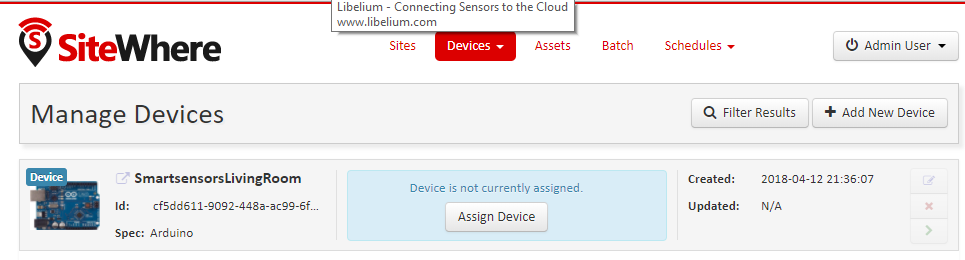
**Καρτέλα Device Details**

Επιλέγουμε το Site που θα κάνουμε χρήση στο παράδειγμά μας το SMART HOME.

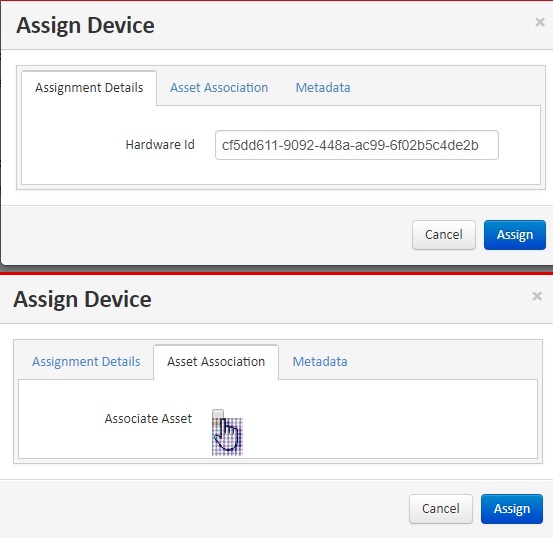
Συμπληρώνουμε το αντίστοιχο HW ID που σημειώσαμε πριν στα asset για το συγκεκριμένο specification και πατάμε CREATE



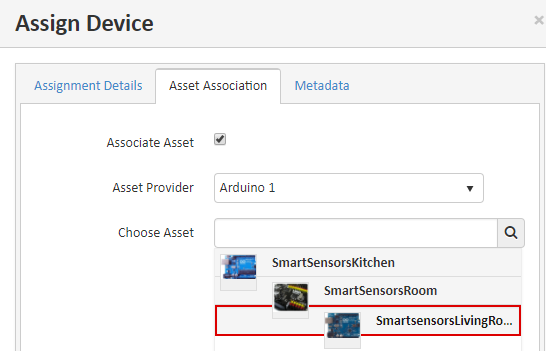
DEVICE ASSIGNMENTS



Για να κάνουμε assign επιλέγουμε Assign Device



Και επιλέγουμε το αντιστοιχο Asset Provider και Hardware Id που αντιστοιχει στο asset που κάνουμε assign στο device.

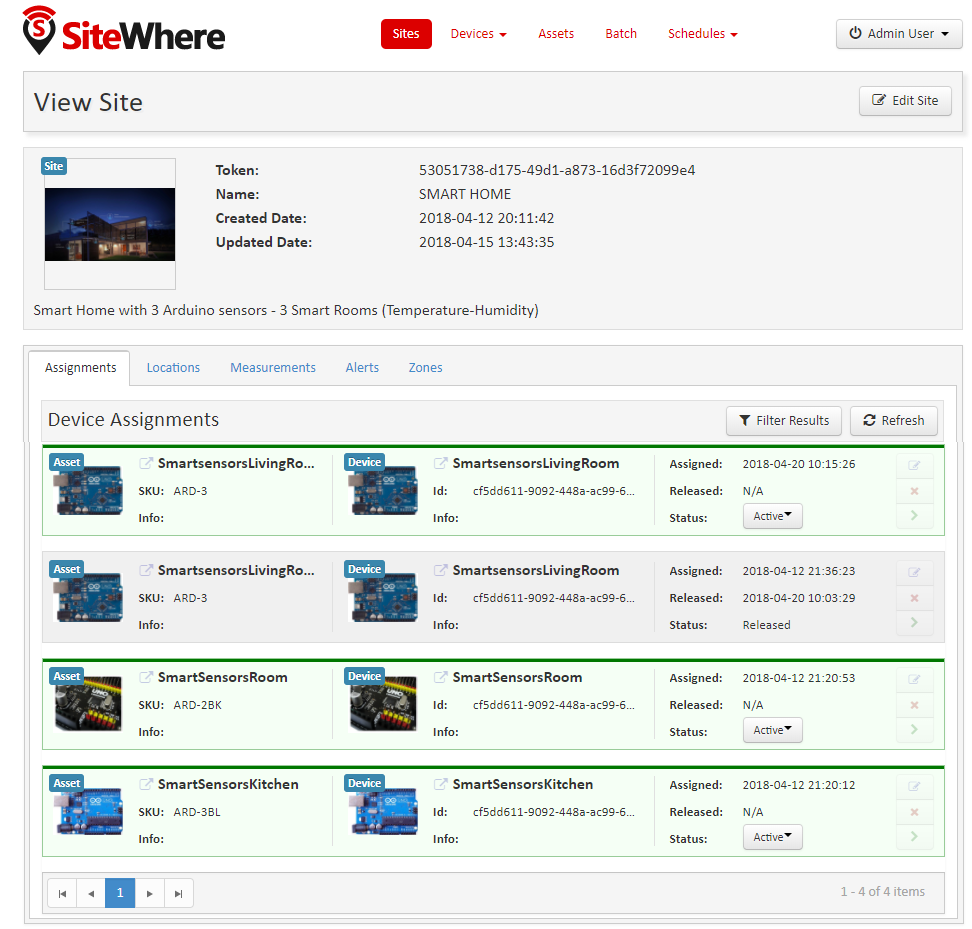


Επίσης από το status στο <http://localhost:8080/sitewhere/admin/default/devices/list.html>

Μπορούμε να κάνουμε και release assignments.

To SMART HOME είναι ετοιμο.

Στην καρτέλα μας .

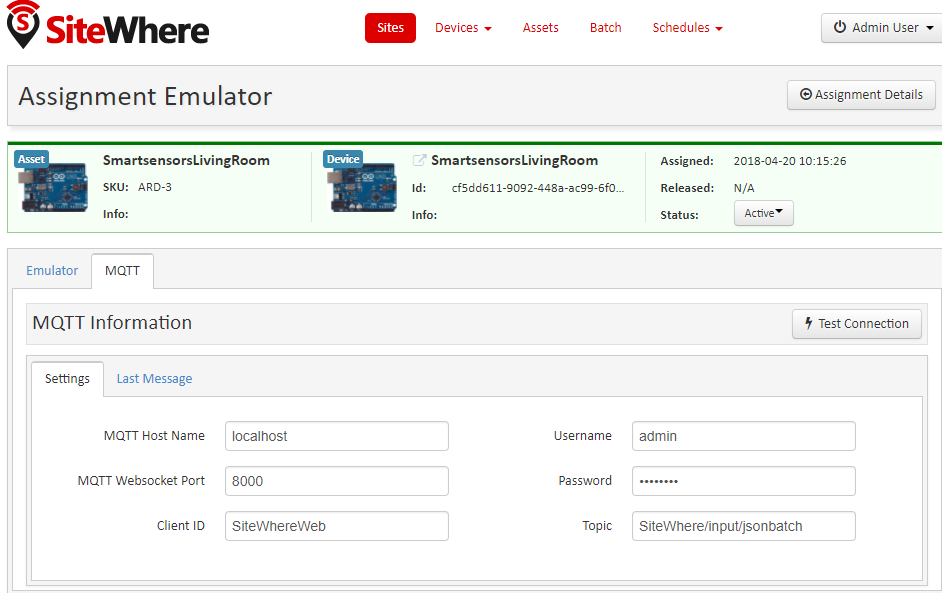


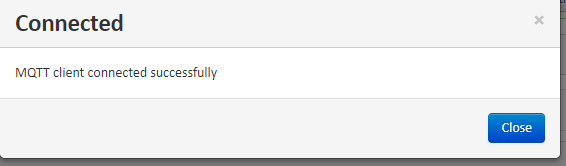
http://localhost:8080/sitewhere/admin/default/sites/53051738-d175-49d1-a873-16d3f72099e4.html?

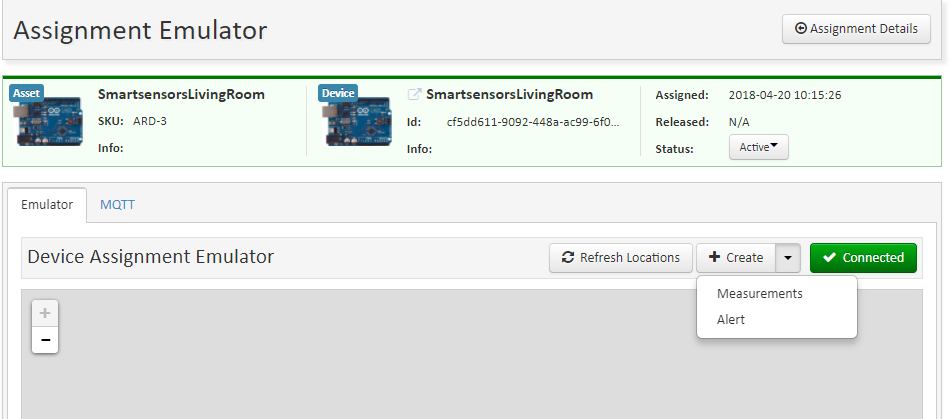
επιλέγοντας κάποιο active Assignment μπορούμε να κάνουμε emulate assignment, δηλαδή να στείλουμε σε όποια θύρα επικοινωνίας κάνει Listen ο server του HiveMQ RealTimeEvents δηλαδη Alerts, Measurements ή Location data.

O HiveMQ στη συνέχεια τα επιστρέφει πίσω στο sitewhere ως καταχωρήσεις όπως βλέπουμε στις αντίστοιχες καρτέλες επιτήρησης.

.







Κώδικας MQTT broker se JSON

JSON Object για αποστολή RTE στο SiteWhere

JSON Packet Format

The following sections show examples of JSON data structured to interact with SiteWhere. All JSON packets share some common information as shown below:

{  
 "hardwareId": "(unique hardware id)",  
 "type": "(indicator for request type)",  
 "request": {  
 ...  
 }  
}

Send a Measurements Event

{  
"hardwareId": "123-TEST-4567890",  
"type": "DeviceMeasurements",  
"request": {  
"measurements": {   
"fuel.level": 87.1,  
"engine.temp": 170  
},  
"updateState": **true**,  
"eventDate": "2016-02-10T19:40:03.391Z"  
}  
}

Send an Alert Event

SiteWhere supports storing alerts for exceptional conditions as event data. The JSON format for sending a device alert is shown below:

{  
 "hardwareId": "123-TEST-4567890",  
 "type":"DeviceAlert",  
 "request": {  
 "type": "engine.overheat",  
 "level": "Warning",  
 "message": "The engine is about to overheat! Turn the machine off!",  
 "updateState": **false**,  
 "eventDate": "2016-02-10T19:40:03.391Z",  
 "metadata": {   
 "name1": "value1",  
 "name2": "value2"  
 }  
 }  
}

Send a Location Event

SiteWhere supports storing device location information as event data. The JSON format for sending a device location is shown below:

{  
 "hardwareId": "123-TEST-4567890",  
 "type":"DeviceLocation",  
 "request": {  
 "latitude": "33.75",  
 "longitude": "-84.39",  
 "elevation": "0",  
 "updateState": **true**,  
 "eventDate": "2016-02-10T19:40:03.390Z"  
 }  
}

Για να στείλουμε δεδομένα στο sitewhere εκτελούμε Python 3.6 σε CLI mode

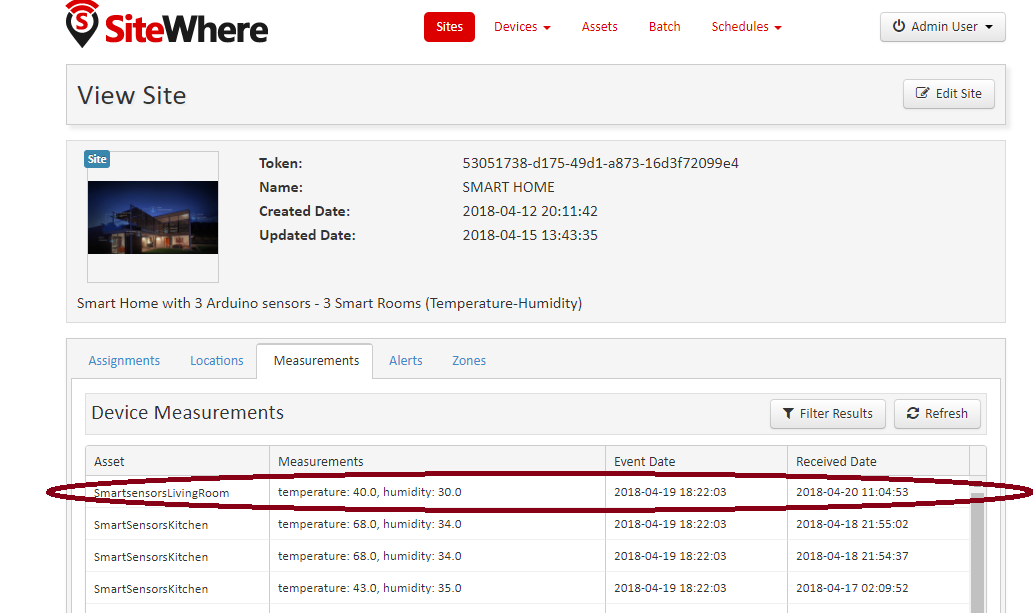
Xρησιμοποιούμε το paho mqtt client ως library import στο Python.

Και αντιγραφή επικόλληση του αντίστοιχου κώδικα.

Π.Χ

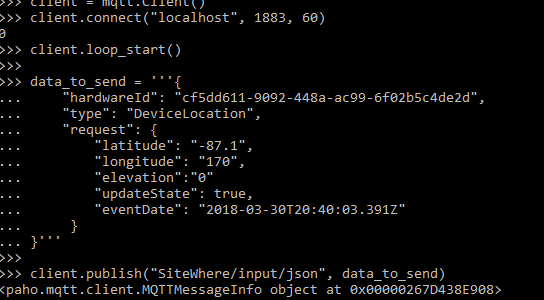
Αποστολή Measurements στον SmartLivingRoom sensor του Smart Home στο sitewhere με εκτέλεση λαμβάνουμε ένα μήνυμα από τον paho client ότι παραδόθηκε object .





Όμοιως για Locations και Alerts στο SmartSensorsKitchen





**ΑΣΚΗΣΗ 1**

Σχήμα βάσης ή σχεσιακό μοντέλο

θα είναι ένας πίνακας της μορφής

CREATE TABLE sensor\_station (

station\_id text,

greenhouse\_owner\_id text,

greenhouse\_id text,

greenhouse\_location\_id text,

station\_position\_id text,

temperature text,

humidity text,

luminosity text,

event\_time timestamp,

PRIMARY KEY (station\_id, event\_time)

);

Όπου σύμφωνα με την εκφώνηση της άσκησης έχουμε :

**διαφορετικούς παραγωγούς** **( greenhouse\_owner\_id )**

**για πολλαπλά θερμοκήπια** **( greenhouse\_id )**

Οι σταθμοί αισθητήρων μπορούν να χρησιμοποιηθούν

σε **διαφορετικά θερμοκήπια** **( station\_id )**

σε **διαφορετικές τοποθεσίες** **( greenhouse\_location\_id )**

και σε **διαφορετικές θέσεις**

**μέσα στο θερμοκήπιο** **(** **station\_position\_id )**

τιμές,

**θερμοκρασίας temperature**

**υγρασίας humidity**

**φωτεινότητας luminosity**

**timestamp**

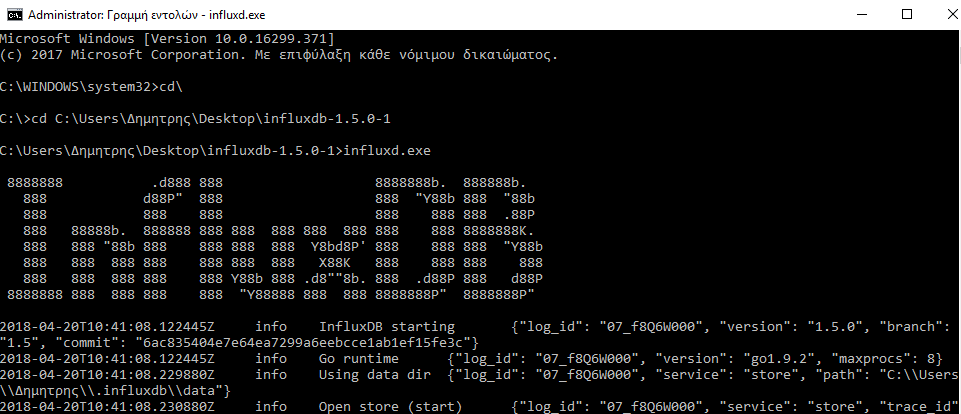
**Primary key**

1. Υλοποίηση του Σχήματος Βάσης στην InfluxDB

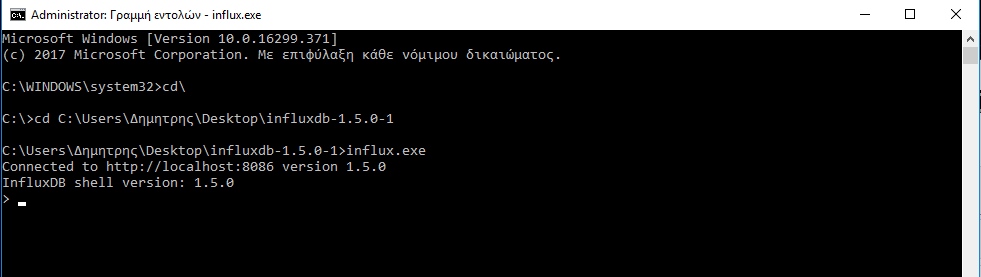
H InfluxDB είναι μια TSD (Time Series Database) ανοιχτής ανάπτυξης λογισμικού (open source)

Για να υλοποιήσουμε το σχήμα θα εκτελέσουμε την InfluxDB σε CLI από τα windows.Η InfluxDB είναι δυνατον να εκτελεστεί και να εμφανίσει δεδομένα βάσης τόσο μέσω command line η σε Python με το κατάλληλο κώδικα καθώς και να εμφανίσει αποτελέσματα ανάκτησης βάσης τόσο σε CLI σε file ή σε HTTP.

1. Εκκινούμε τον server από command Line με την εντολή Influxd.exe



1. Και μετά το command line της InfluxDB με την εντολή influx.exe



1. Εντολές InfluxDB σε CLI

Για να λειτουργήσει η InfluxDB θα πρέπει να δημιουργήσουμε μια καινούρια Βάση.Η εντολή που χρησιμοποιούμαι είναι η

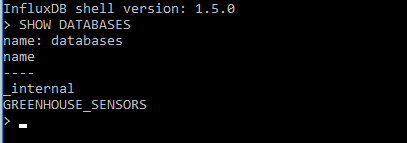
CREATE DATABASE GREENHOUSE\_SENSORS

Δημιουργήσαμε μια βάση δεδομένων με το ονομα Greenhouse\_Sensors

Με την εντολη

SHOW DATABASES

Βλέπουμε την εσωτερική βάση που δημιουργήσαμε.



An Θέλουμε να σβήσουμε ολοκληρωτικά μια βάση δεδομένων μαζί με τα δεδομένα που περιέχει φυσικά χρησιμοποιούμε την εντολη

DROP DATABASE (ονομα\_βασης)

Για να κάνουμε χρήση μιας βάσης εκτελούμε την εντολή

USE (database)

Στο παράδειγμα μας

USE GREENHOUSE\_SENSORS

Για να εισάγουμε δεδομένα κάνουμε χρήση της εντολής INSERT

Η σύνταξη της εντολής είναι της μορφής

INSERT <measurement>[,<tag-key>=<tag-value>..] <field-key>=<field-value>[,fields..] [unix-nano-timestamp]

**Παράδειγμα υλοποίησης της βάσης μας**

INSERT ARDUINO\_SENSOR,station\_id=15,greenhouse\_owner\_id=PAPAIOANOU,greenhouse\_id=GIANNOULI,greenhouse\_location\_id=LARISA,station\_place\_id=MAIN TEMPERATURE=37,HUMIDITY=65,LUMINOSITY=960

INSERT ARDUINO\_SENSOR,station\_id=16,greenhouse\_owner\_id=LOKAS,greenhouse\_id=TEI,greenhouse\_location\_id=LARISA,station\_place\_id=MAIN TEMPERATURE=34,HUMIDITY=60,LUMINOSITY=945

.

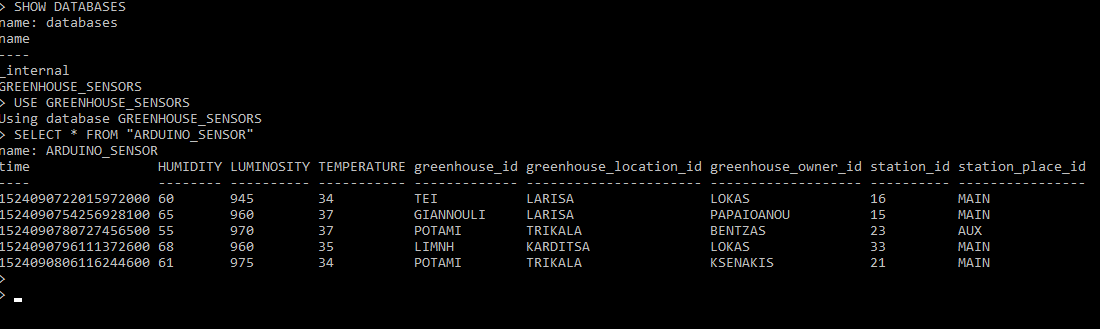
.

.

Όπου measurement ARDUINO\_SENSOR

Τα αποτελέσματα του πίνακα της βάσης δεδομένων που σχηματίζεται δίνονται με την εντολή

SELECT \* FROM "measurement"



1. Υλοποίηση του Σχήματος Βάσης στην Apache Cassandra

H Cassandra είναι μια Masterless peer-to-peer TSD βάση δεδομένων η οποία βασίζεται στην Dynamo της Amazon.

Εκτέλεση της Cassandra σε γραφικό περιβάλλον Windows.Ανοίγουμε τον υπο κατάλογο C:\Program Files\DataStax Community\apache-cassandra\bin και εκτελούμε το αρχείο Cassandra.bat.Η θύρα που χρησιμοποιεί από προεπιλογή είναι η **7199**.



Έχοντας σηκώσει το server εκτελούμε την εντολή cqlsh.bat η οποία είναι η command line της Cassandra.

Η Cassandra λειτουργει με keyspaces.

Με την εντολη

**CREATE** **KEYSPACE** [**IF** **NOT** **EXISTS**] ***keyspace\_name***

**WITH** **REPLICATION** = {

'class' : 'SimpleStrategy', 'replication\_factor' : ***N*** }

| 'class' : 'NetworkTopologyStrategy',

'***dc1\_name***' : ***N*** [, ...]

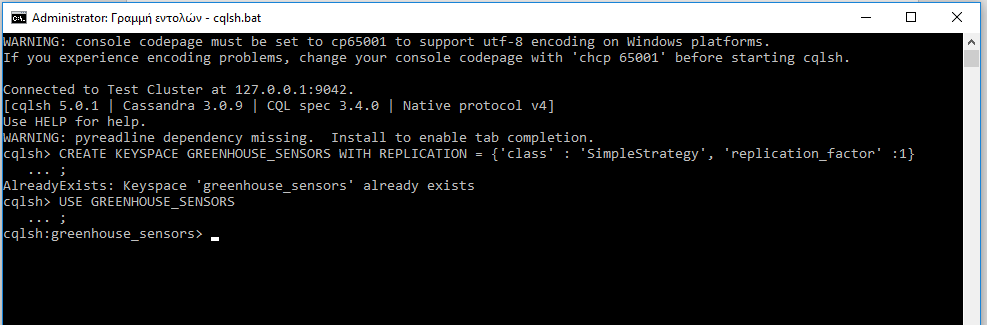
}

[**AND** **DURABLE\_WRITES** = true|false] ;

Κάνουμε δημιουργία keyspace με το όνομα GREENHOUSE\_SENSORS

Με Replication Map Ν=1 δηλαδη κρατάει **ένα αντίγραφο για κάθε κλειδί** σύμφωνα με την εκφώνηση της άσκησης

Με την εντολή USE <keyspase> κάνουμε χρήση της βάσης δεδομένωμ που φτιάξαμε.



Mε την εντολή CREATE TABLE δημιούργουμε τον πίνακα της βάσης δεδομένων μας.

**CREATE** **TABLE** [**IF** **NOT** **EXISTS**] [***keyspace\_name.***]***table\_name*** (

***column\_definition*** [, ...]

**PRIMARY** **KEY** (***column\_name*** [, ***column\_name*** ...])

[**WITH** ***table\_options***

| **CLUSTERING** **ORDER** **BY** (***clustering\_column\_name*** ***order***])

| ID = '***table\_hash\_tag***'

| **COMPACT** **STORAGE**]

Στην περίπτωση μας βάζω τα εξής statements

CREATE TABLE ARDUINO\_SENSORS (

station\_id text,

greenhouse\_owner\_id text,

greenhouse\_id text,

greenhouse\_location\_id text,

station\_place\_id text,

temperature text,

humidity text,

luminosity text,

event\_time timestamp,

PRIMARY KEY (station\_id, event\_time)

);

Αν θέλουμε να διαγράψουμε ένα πίνακα μαζί με όλα τα περιεχόμενα του

εκτελούμε την εντολή

**DROP** **TABLE** [**IF** **EXISTS**] ***keyspace\_name.table\_name***

Για να εισάγουμε δεδομένα στον πίνακα μας χρησιμοποιούμε την εντολή

**INSERT** **INTO** [***keyspace\_name***.] ***table\_name*** (***column\_list***)

**VALUES** (***column\_values***)

[**IF** **NOT** **EXISTS**]

[**USING** **TTL** ***seconds*** | **TIMESTAMP** ***epoch\_in\_microseconds***]

Π.Χ

INSERT INTO GREENHOUSE\_SENSORS.ARDUINO\_SENSORS(station\_id,event\_time,greenhouse\_id,greenhouse\_location\_id,greenhouse\_owner\_id,station\_place\_id,temperature,humidity,luminosity) VALUES('21','0','POTAMI','TRIKALA','LOKAS','MAIN','35','68','940')USING TTL 86400 AND TIMESTAMP 123456789

Τέλος με την εντολή

**SELECT FROM \*** [**IF** **EXISTS**] ***keyspace\_name.table\_name***

SELECT \* FROM GREENHOUSE\_SENSORS.ARDUINO\_SENSORS

Βλέπουμε τον πίνακα μας

