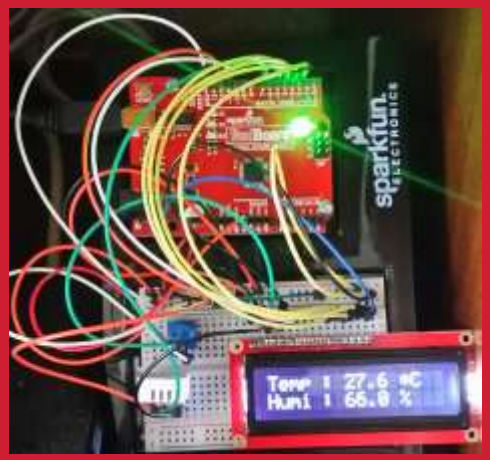
Arduino-IOT

[wk15]

Arduino + Node Data mining



Visualization of Signals using Arduino, Node.js & storing signals in MongoDB

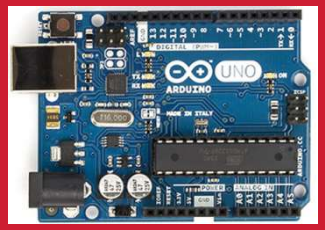
Comsi, INJE University

2nd semester, 2018

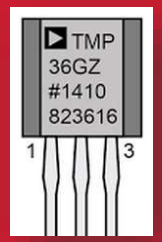
Email : chaos21c@gmail.com

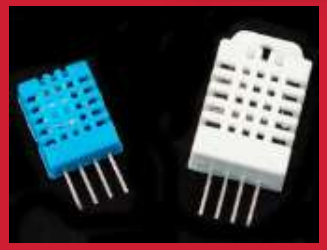
My ID

진영빈	AA01
김태은	AA02
도한솔	AA03
박지수	AA04
신성	AA05
박현승	AA06
이석주	AA07
전규은	AA08
정영관	AA09
정의석	AA10
이근재	**AA11**



[Review]





- [wk14]
  - RT Data storaging with MongoDB
  - Multi-sensor circuits(cds-dht22)
  - Complete your project
  - Upload file name : AAnn_Rpt10.zip

[wk14] Practice-10 AAnn_Rpt10.zip

- [Target of this week]
  - Complete your charts
  - Save your outcomes and compress them.

제출파일명 : **AAnn_Rpt10.zip**

- 압축할 파일들

① AAnn_mongo_schemas.png

② AAnn_mongo_update.png

③ AAnn_iot_mongodb.png

④ AAnn_iot_mongodb_web.png

⑤ AAnn_iot_json.png

⑥ AAnn_iot_client.png

⑦ AAnn_s1000.csv (mongoexport file)

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

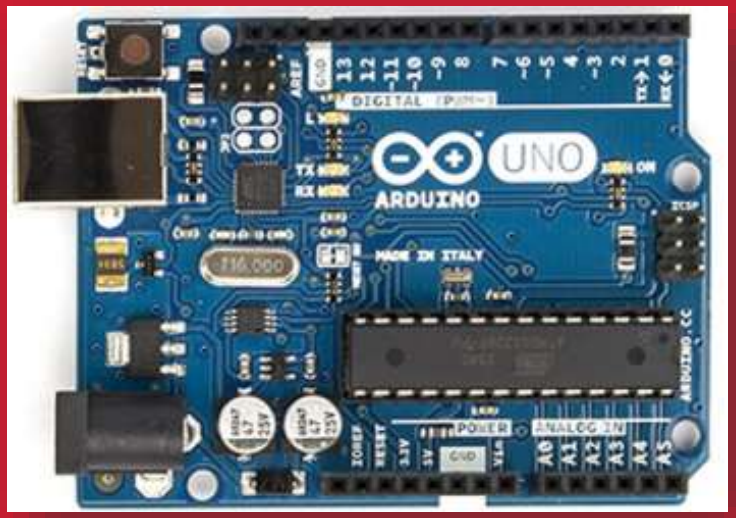
wk15 : 기말고사 안내

[1] 실기

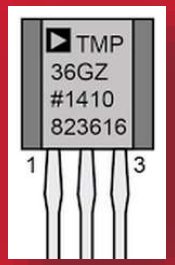
- 시간: 12월 4일 오전 9 시~
- 장소: E323 실습실
- 배점: 10점
- 실습 내용을 github에 업로드.

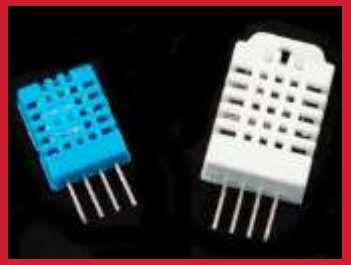
[2] 필기

- 시간: 12월 11일 오전 11 시~12시
- 장소: E323 실습실
- 배점: 20 점
- 범위: wk09 ~ wk15



Arduino & Node.js





IOT: HSC

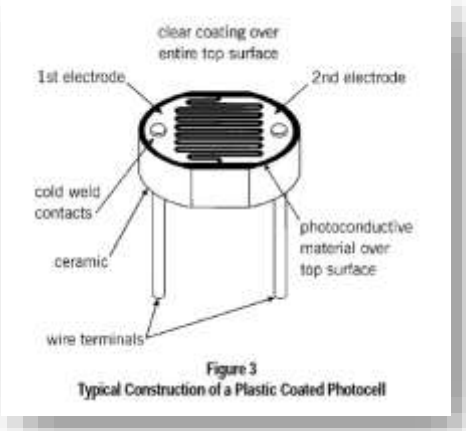
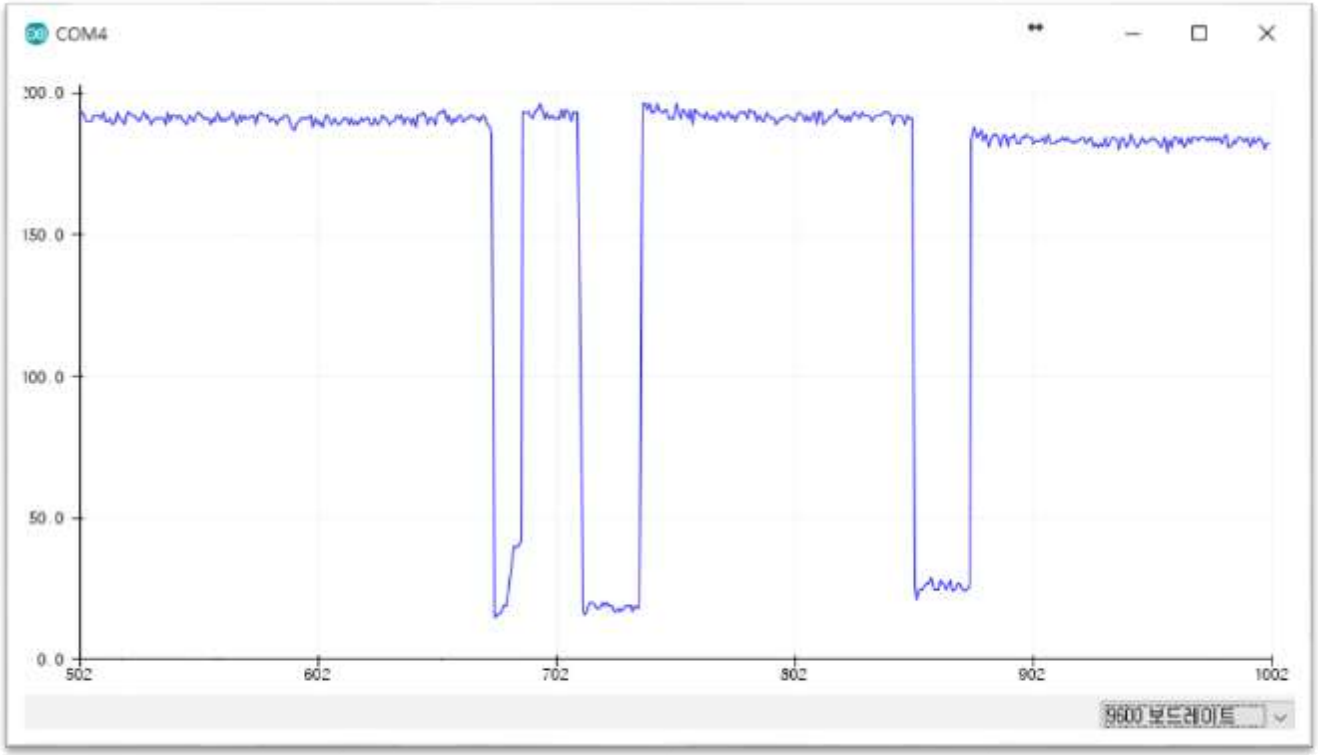
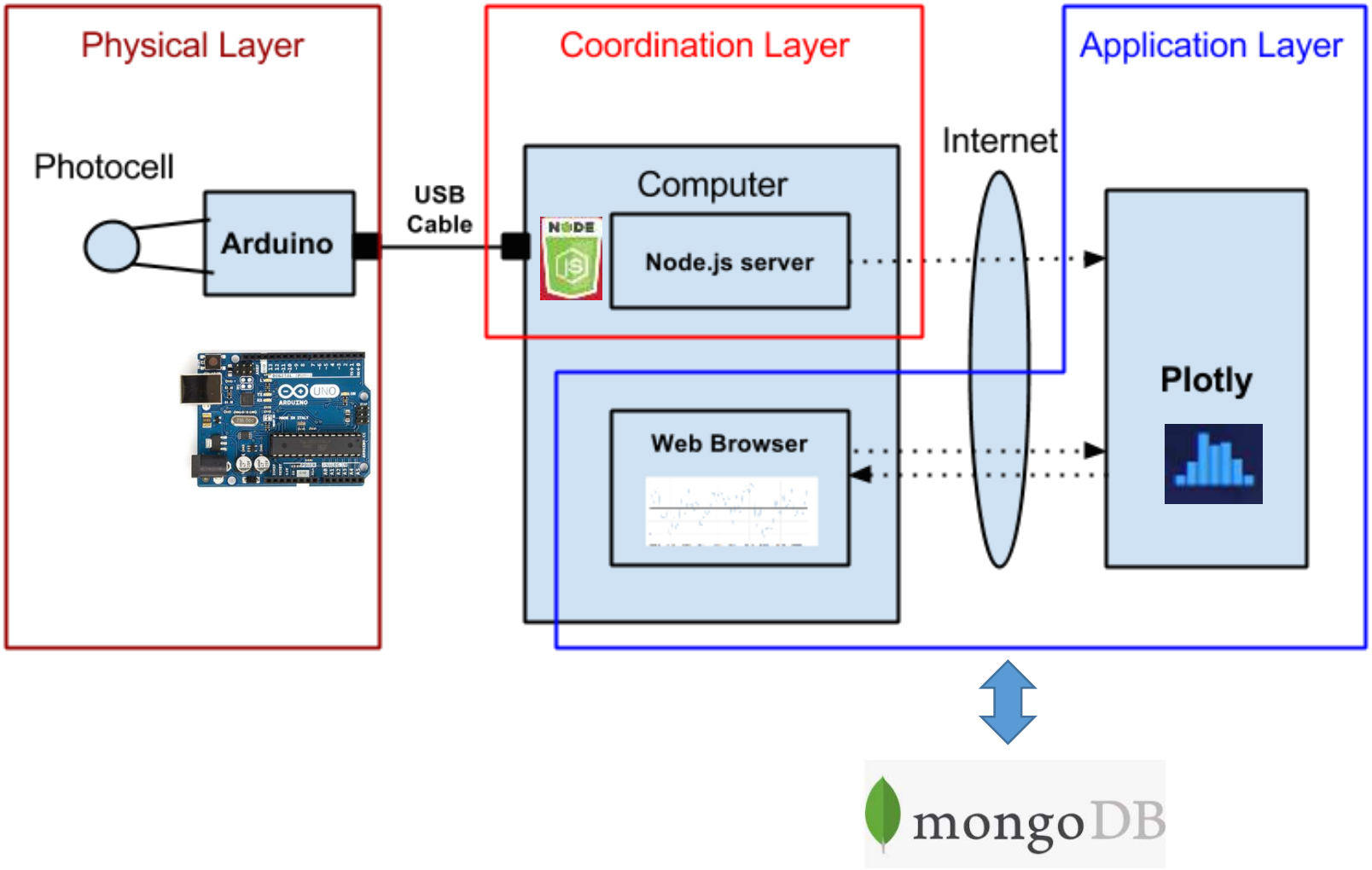


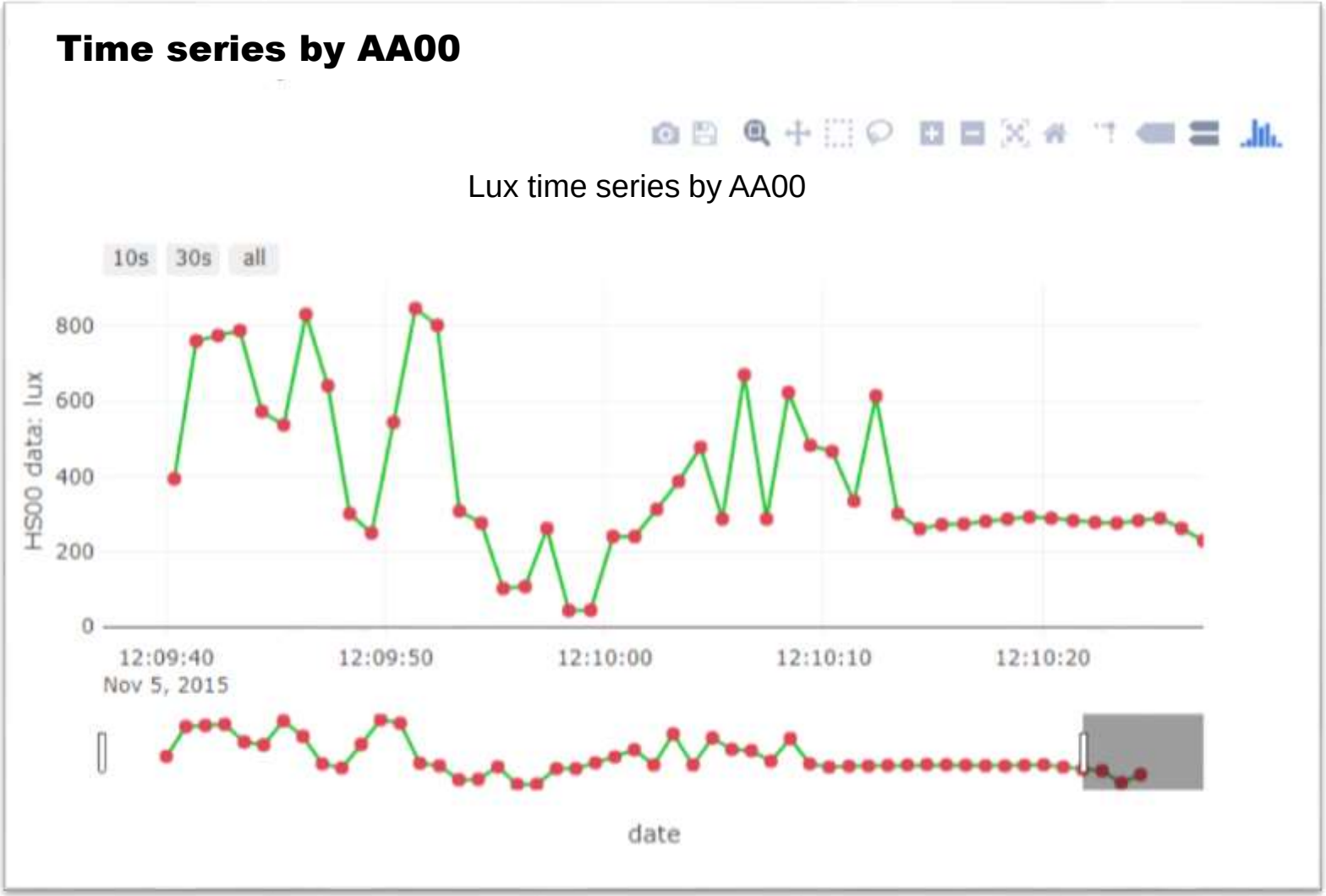
Figure 3
Typical Construction of a Plastic Coated Photocell



Layout [H S C]

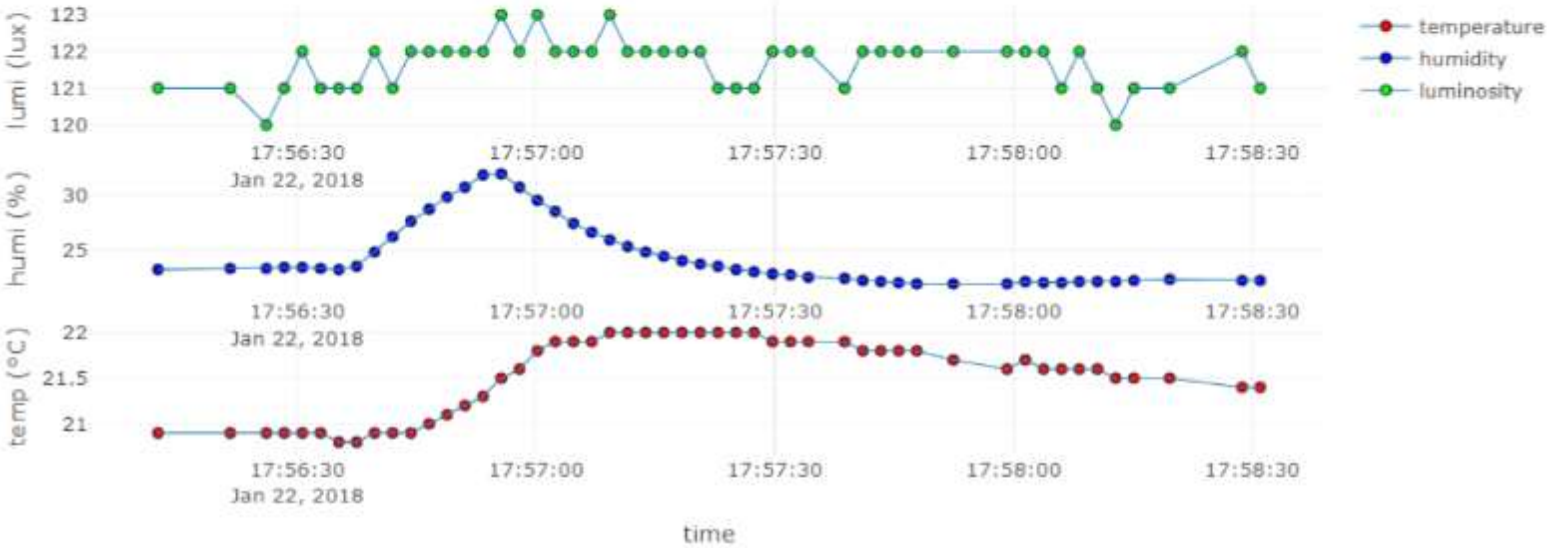

Physical Layer
Photocell
Arduino
USB Cable
Coordination Layer
Computer
Node.js server
Web Browser
Internet
Application Layer
Plotly
mongoDB

Arduino data + plotly



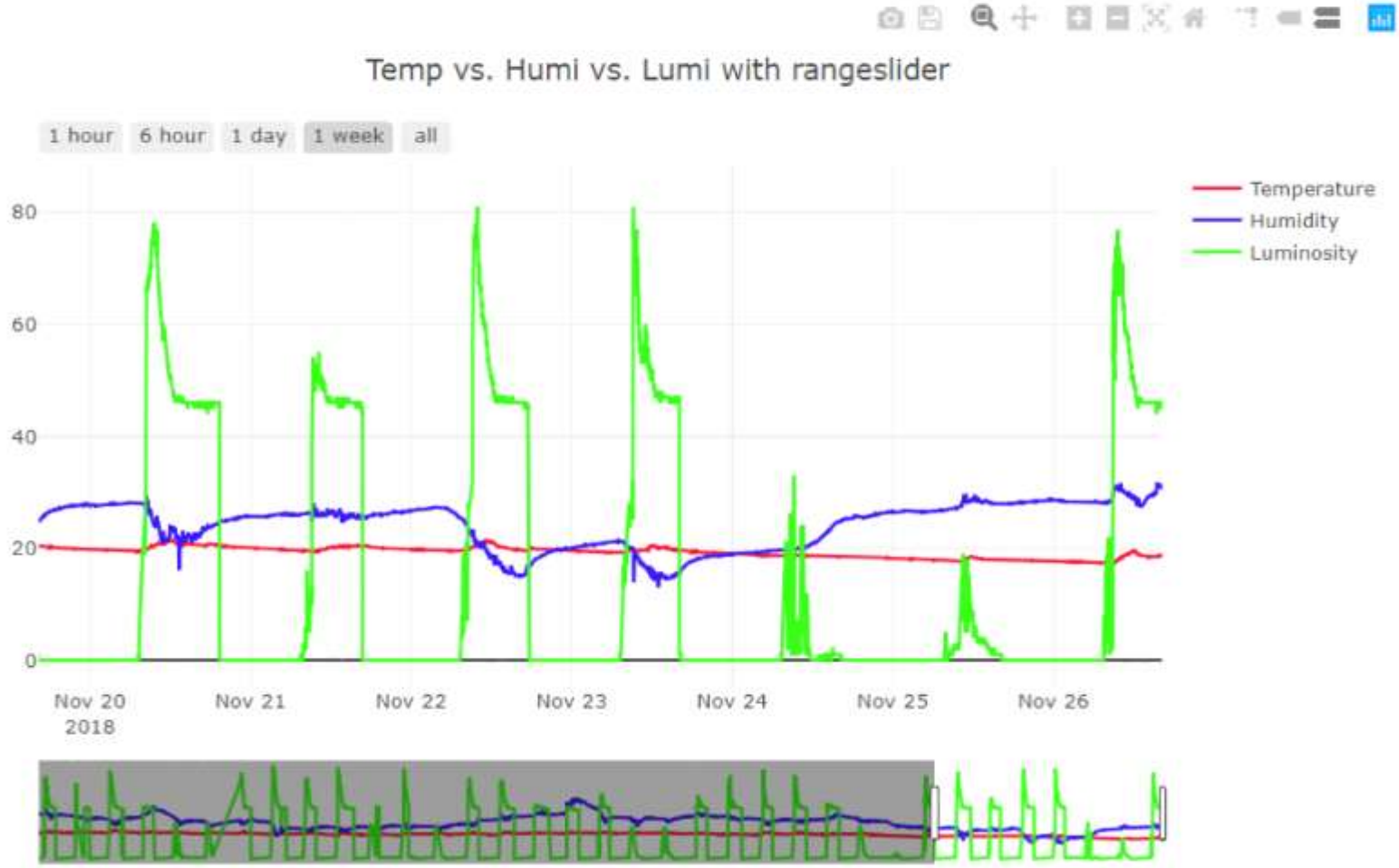
Real-time Weather Station from sensors

on Time: 2018-01-22 17:58:31.012



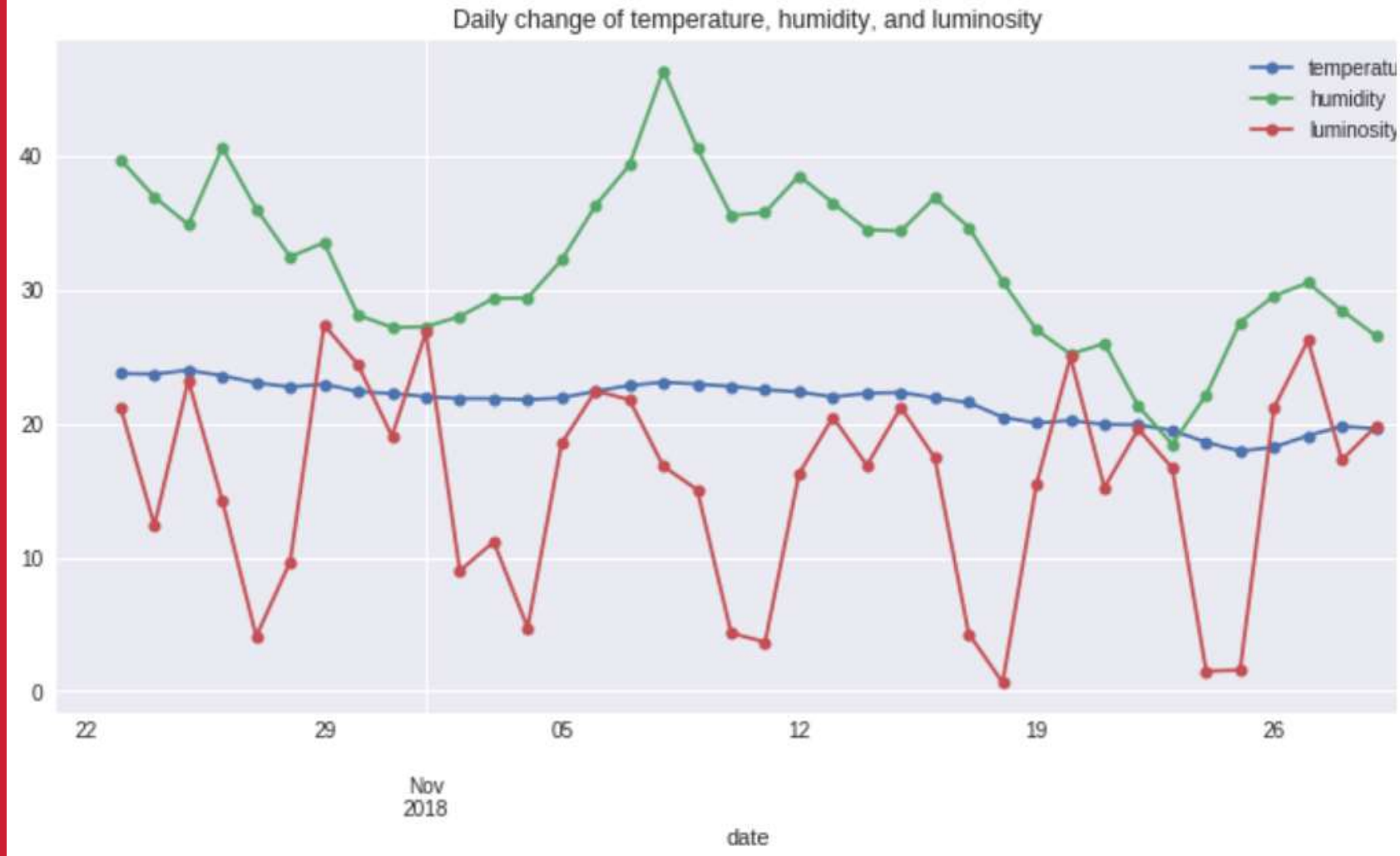
MongoDB database visualization by AA00

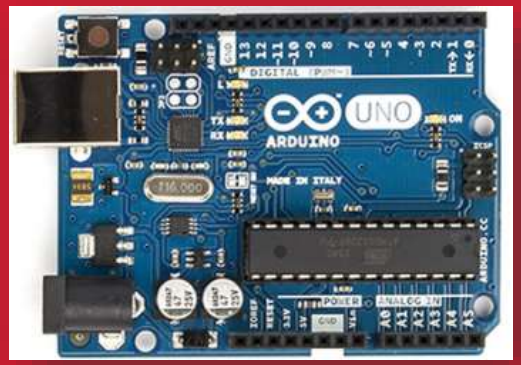
Time series : Multi sensor data

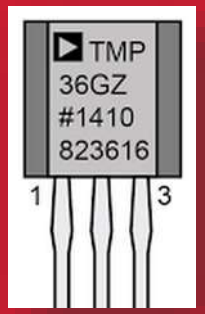


```
# Plot mean of the iot data per every day
iot_data.resample('D').mean().plot(kind='line', marker='o', ms=6, figsize=(12,6),
                                   title='Daily change of temperature, humidity, and luminosit
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2c7fb7f0>
```

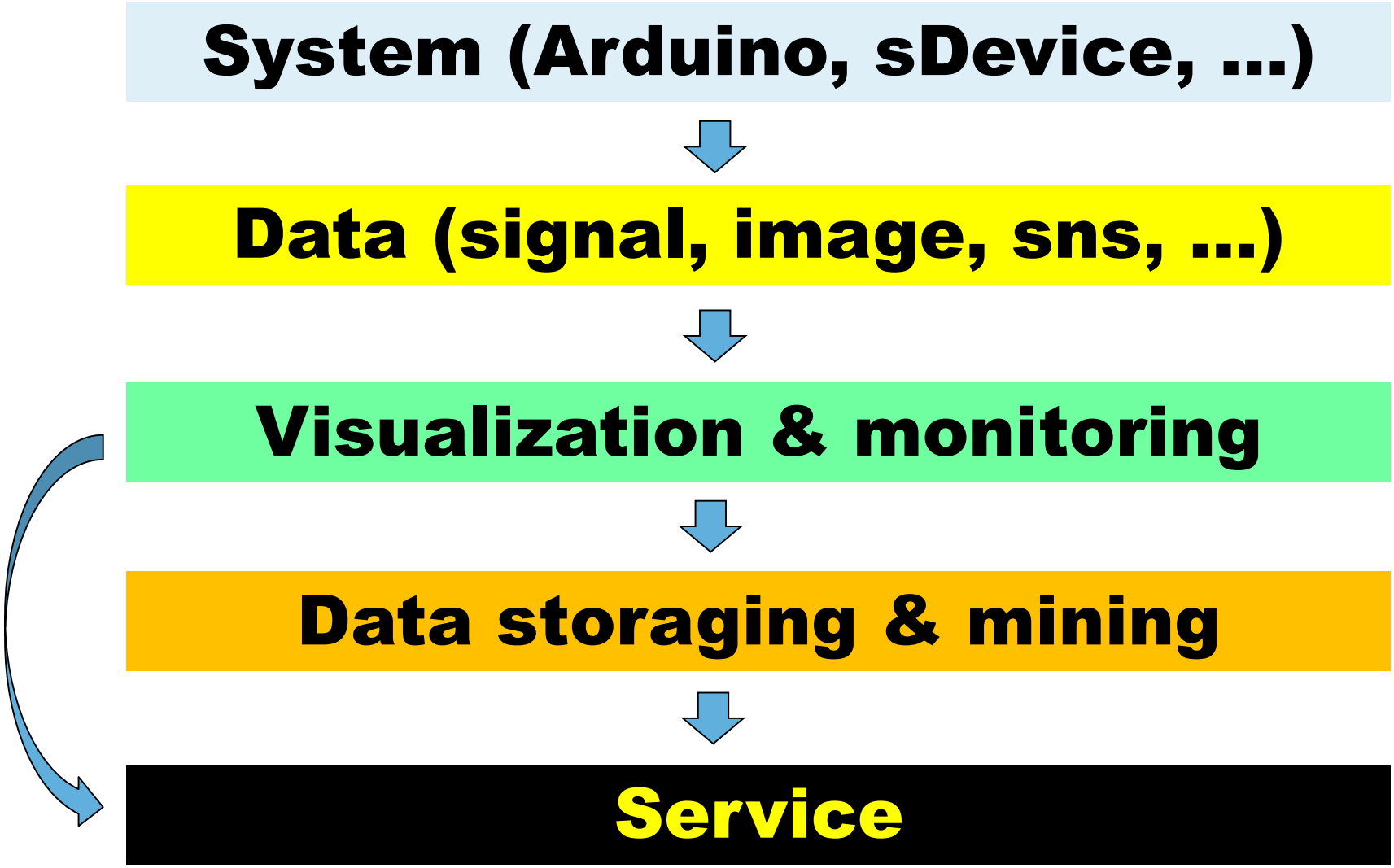


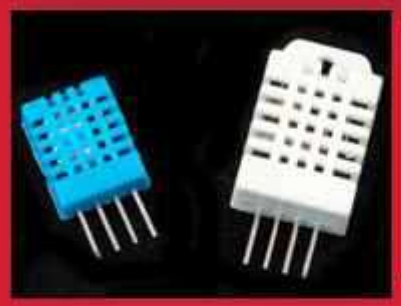




Data visualization using ploy.ly

A5. Introduction to visualization





[Goal]

Arduino + Node.js

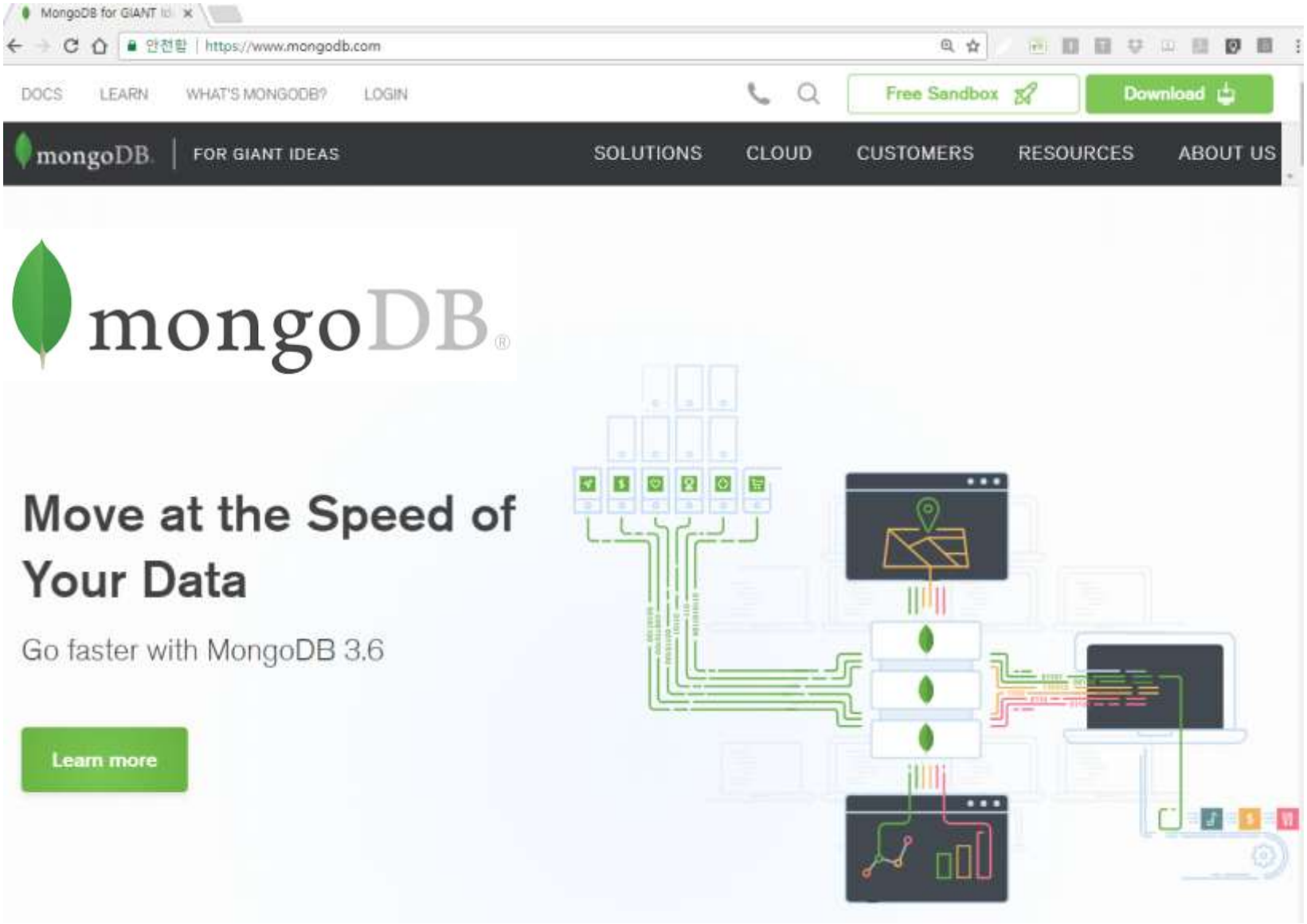
+ plotly.js

+ MongoDB

→ Data storaging

& visualization

A5.9 MongoDB


MongoDB for GIANT
https://www.mongodb.com
DOCS
LEARN
WHAT'S MONGODB?
LOGIN
Free Sandbox
Download
mongoDB
FOR GIANT IDEAS
SOLUTIONS
CLOUD
CUSTOMERS
RESOURCES
ABOUT US
mongoDB
Move at the Speed of Your Data
Go faster with MongoDB 3.6
Learn more

A5.9 MongoDB

MongoDB는 C++로 작성된 오픈소스 문서지향(Document-Oriented) 적 Cross-platform 데이터베이스이며, 뛰어난 확장성과 성능을 자랑합니다. 또한, 현존하는 NoSQL 데이터베이스 중 인지도 1위를 유지하고있습니다.

NoSQL?

흔히 NoSQL이라고 해서 아, SQL이 없는 데이터베이스구나! 라고 생각 할 수도 있겠지만, 진짜 의미는 Not Only SQL 입니다. 기존의 RDBMS의 한계를 극복하기 위해 만들어진 새로운 형태의 데이터저장소입니다. 관계형 DB가 아니므로, RDMS처럼 고정된 스키마 및 JOIN 이 존재하지 않습니다.

Document?

Document Oriented 데이터베이스라는데.. 여기서 말하는 Document가 뭘까요? 문서? 이게 그냥 '문서' 로 번역해버리면 조금은 애매합니다. 문서라고 하면 보통 워드/엑셀에 사용되는 그런 문서가 떠오르는데요, 그것과는 다릅니다. Document는 RDMS의 record 와 비슷한 개념인데요, 이의 데이터 구조는 한개이상의 key-value pair 으로 이뤄져있습니다. MongoDB 샘플 Document를 확인 해 볼까요?

```
{ "_id": ObjectId("5099803df3f4948bd2f98391"),
"username": "velopert",
"name": { first: "M.J.", last: "Kim" } }
```

A5.9 MongoDB

여기서 _id, username, name 은 key 이고 그 오른쪽에 있는 값들은 value 입니다.

_id 는 12bytes의 hexadecimal 값으로서, 각 document의 유일함(uniqueness)을 제공합니다.
이 값의 첫 4bytes 는현재 timestamp, 다음 3bytes는 machine id, 다음 2bytes는 MongoDB 서버의 프로세스id, 마지막 3bytes는 순차번호입니다 추가될때마다 값이 높아진다는거지요.

Document는 동적(dynamic)의 schema 를 갖고있습니다. 같은 Collection 안에 있는 Document 끼리 다른 schema 를 갖고 있을 수 있는데요, 쉽게 말하면 서로 다른 데이터 (즉 다른 key) 들을 가지고 있을 수 있습니다.

Collection?

Collection은 MongoDB Document의 그룹입니다. Document들이 Collection내부에 위치하고 있습니다. RDMS의 table과 비슷한 개념입니다만 RDMS와 달리 schema를 따로 가지고 있지않습니다. Document 부분설명에 나와있듯이 각 Document들이 동적인 schema를 가지고 있으니까요

Database?

Database는 Collection들의 물리적인 컨테이너입니다. 각 Database는 파일시스템에 여러파일들로 저장됩니다.

A5.9.3 MongoDB shell coding

3. insert more records with different schema & show records

insert record4
with firstName key

db.user.find()

db.user.find().pretty()

명령 프롬프트 - mongo

```
> db.user.insert({firstName:"Fractal", last:"Park"})
WriteResult({ "nInserted" : 1 })
> db.user.find().pretty()
{
        "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
        "first" : "Redwoods",
        "last" : "Yi"
}
{
        "_id" : ObjectId("5a66b5759f0d55608f5f7583"),
        "first" : "Chaos",
        "last" : "Kim"
}
{
        "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
        "first" : "Gildong",
        "last" : "Hong"
}
{
        "_id" : ObjectId("5a66b6439f0d55608f5f7585"),
        "firstName" : "Fractal",
        "last" : "Park"
}
>
```

Dynamic schema

Note that there are two kinds of schemas in JSON. Save as

AAnn_mongo_schemas.png

A5.9.3 MongoDB shell coding

5. update a record

update record2

db.user.find().pretty()

명령 프롬프트 - mongo

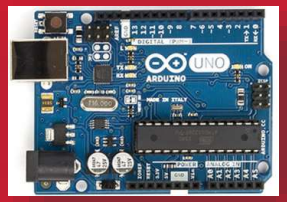
```
> db.user.update({last:"Hong"},{$set:{first:"GilDong", age:21}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.user.find().pretty()
{
        "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
        "first" : "Redwoods",
        "last" : "Yi"
}
{
        "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
        "first" : "GilDong",
        "last" : "Hong",
        "age" : 21
}
{
        "_id" : ObjectId("5a66b6439f0d55608f5f7585"),
        "firstName" : "Fractal",
        "last" : "Park"
}
>
```

db.user.update({last:"Hong"},{$set:{first:"GilDong", age:21}})

Note that it is possible to change schema.

Save as

AAnn_mongo_update.png



Node.js

+

MongoDB

A5.9.4 MongoDB + Node.js：mongoose

Mongoose ODM v5.0.1

mongoosejs.com

Fork me on GitHub

mongoose

elegant mongodb object modeling for node.js

read the docs | discover plugins

Version 5.0.1

Let's face it,**writing MongoDB validation, casting and business logic boilerplate is a drag**. That's why we wrote Mongoose.

http://mongoosejs.com/

A5.9.4 MongoDB + Node.js : mongoose

4. dbtest2.js (use Sublime Text 3)

```
var SensorSchema = new mongoose.Schema({
        data: String,
        created: String
});
```

D:\#Portable\#NodeJSPortable\#Data\#aa00\#iot\#cds_dht22\#dbtest2.js (Data) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
- Data
  - aa00
    - express
    - expressTest
    - iot
      - cds
        - node_modules
        - cds_node.js
        - package.json
      - cds_dht22
        - node_modules
        - cds_dht22_node.js
        - dbtest.js
        - dbtest2.js
        - package.json
      - cds_tmp36
      - plotly
      - tmp36
    - myApp
    - server
    - start
  - node_modules
  - npm_cache
  - settings
  - Temp
  - express
  - express.cmd
  - npm
  - npm.cmd
  - PortableApps.comLauncherRuntimeData-NodeJSPo

dbtest2.js

```
// dbtest2.js
var mongoose = require('mongoose');
mongoose.connect('mongodb://localhost/test2');

var SensorSchema = new mongoose.Schema({
    data: String,
    created: String
});

// data model
var Sensor = mongoose.model("Sensor", SensorSchema);

var sensor1 = new Sensor({data:'124', created: getDateString()});
sensor1.save();

var sensor2 = new Sensor({data:'573', created: getDateString()});
sensor2.save();

console.log("[dbtest2.js]: Sensor data were saved in MongoDB");

// helper function to get a nicely formatted date string
function getDateString() {
    var time = new Date().getTime();
    // 32400000 is (GMT+9 Korea, GimHae)
    // for your timezone just multiply +/-GMT by 3600000
    var datestr = new Date(time +32400000).
    toISOString().replace(/T/, ' ').replace(/Z/, '');
    return datestr;
}
```

```
[dbtest2.js]: Sensor data were saved in MongoDB
```

5. dbtest2.js (change Schema & check using mongo shell)

Mongo shell

> show dbs

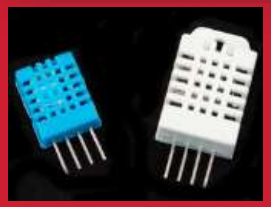
> use test2

> show collections

> db.sensors.find().pretty()

명령 프롬프트 - mongo

```
> show dbs
aa00    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
test2   0.000GB
> use test2
switched to db test2
> show collections
sensors
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5a66cc2f56c1ac4e4051ae35"),
        "data" : "124",
        "created" : "2018-01-23 14:46:23.231",
        "__v" : 0
}
{
        "_id" : ObjectId("5a66cc2f56c1ac4e4051ae36"),
        "data" : "573",
        "created" : "2018-01-23 14:46:23.235",
        "__v" : 0
}
>
```

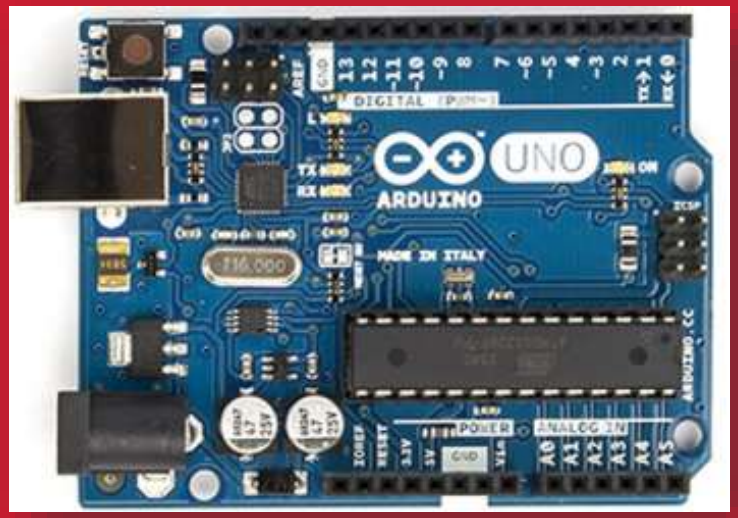


MongoDB from Arduino with node.js & mongoose

```
> show dbs
aa00     0.000GB
admin    0.000GB
config   0.000GB
iot      0.000GB
iot2     0.000GB
iot3     0.001GB
local    0.000GB
test     0.000GB
test2    0.000GB
>
```

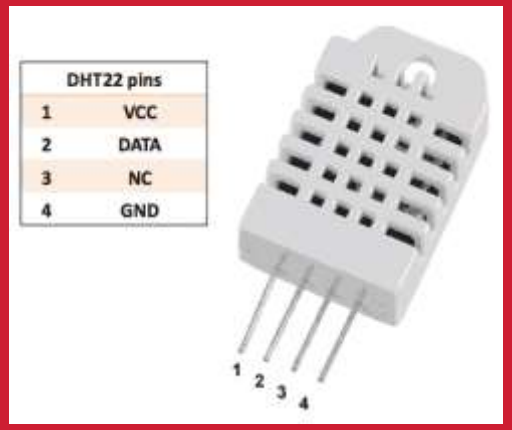
```
mongo db connection OK.
info() - Current date is 2015-11-26 12:04:21.411, Lumi: 67
info() - Current date is 2015-11-26 12:04:26.415, Lumi: 67
info() - Current date is 2015-11-26 12:04:31.416, Lumi: 67
info() - Current date is 2015-11-26 12:04:36.422, Lumi: 104
info() - Current date is 2015-11-26 12:04:41.427, Lumi: 92
info() - Current date is 2015-11-26 12:04:46.432, Lumi: 410
info() - Current date is 2015-11-26 12:04:51.432, Lumi: 67
info() - Current date is 2015-11-26 12:04:56.438, Lumi: 66
```

Arduino & Node.js & MongoDB

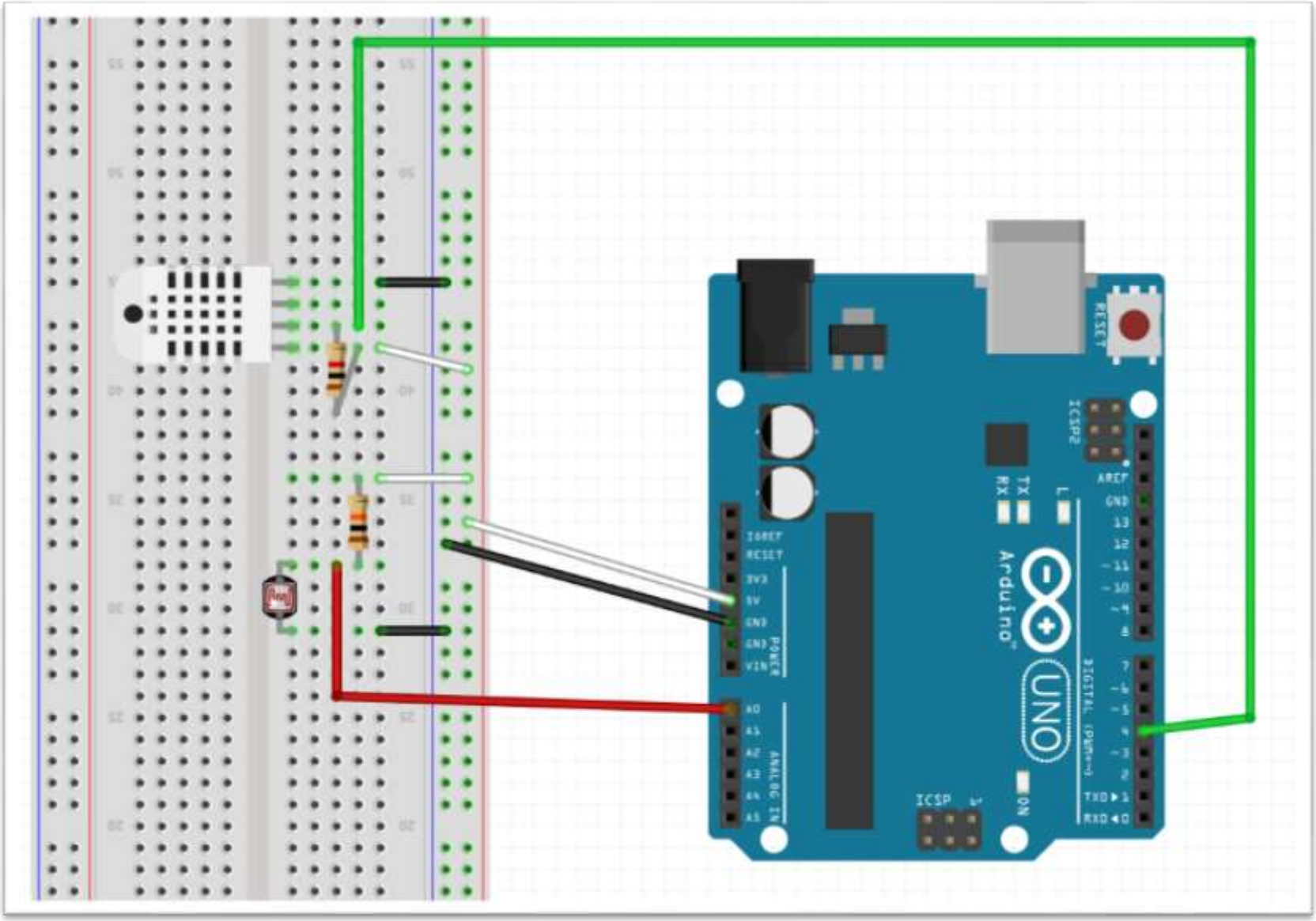


Multi-sensors

DHT22 + CdS

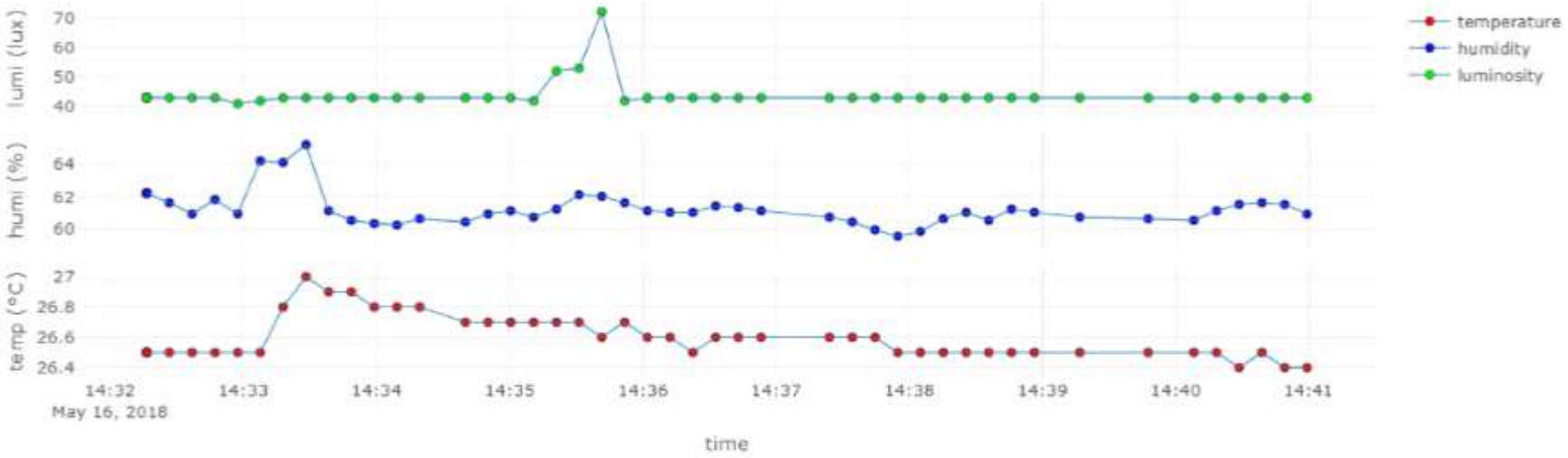


DHT22 + CdS : circuit



Real-time Weather Station from sensors

on Time: 2018-05-16 14:40:59.402



A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.1 cds_dht22_mongodb.js

```
1  // cds_dht22_mongodb.js
2
3  var serialport = require('serialport');
4  var portName = 'COM4';  // check your COM port!!
5  var port     =   process.env.PORT || 3000;
6
7  var io = require('socket.io').listen(port);
8
9  // MongoDB
10 var mongoose = require('mongoose');
11 var Schema = mongoose.Schema;
12 // MongoDB connection
13 mongoose.connect('mongodb://localhost:27017/iot'); // DB name
14 var db = mongoose.connection;
15 db.on('error', console.error.bind(console, 'connection error:'));
16 db.once('open', function callback () {
17     console.log("mongo db connection OK.");
18 });
19 // Schema
20 var iotSchema = new Schema({
21     date : String,
22     temperature : String,
23     humidity : String,
24     luminosity: String
25 });
```

A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.2 cds_dht22_mongodb.js

```
27 iotSchema.methods.info = function () {
28     var iotInfo = this.date
29     ? "Current date: " + this.date +", Temp: " + this.temperature
30     + ", Humi: " + this.humidity + ", Lux: " + this.luminosity
31     : "I don't have a date"
32     console.log("iotInfo: " + iotInfo);
33 }
34
35 // serial port object
36 var sp = new serialport(portName,{
37     baudRate: 9600,   // 9600  38400
38     dataBits: 8,
39     parity: 'none',
40     stopBits: 1,
41     flowControl: false,
42     parser: serialport.parsers.readline('\r\n')  // new serialport.parsers
43 });
44
45 var readData = '';  // this stores the buffer
46 var temp ='';
47 var humi ='';
48 var lux ='';
49 var mdata =[]; // this array stores date and data from multiple sensors
50 var firstcommaidx = 0;
51
52 var Sensor = mongoose.model("Sensor", iotSchema);  // sensor data model
```

A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.3 cds_dht22_mongodb.js

```
54 sp.on('data', function (data) { // call back when data is received
55     readData = data.toString(); // append data to buffer
56     firstcommaidx = readData.indexOf(',');
57
58     // parsing data into signals
59     if (readData.lastIndexOf(',') > firstcommaidx && firstcommaidx > 0) {
60         temp = readData.substring(firstcommaidx + 1, readData.indexOf(',',firstcommaidx+1));
61         humi = readData.substring(readData.indexOf(',',firstcommaidx+1) + 1, readData.lastIndexOf(','));
62         lux = readData.substring(readData.lastIndexOf(',')+1);
63
64         readData = '';
65
66         dStr = getDateString();
67         mdata[0]=dStr;  // Date
68         mdata[1]=temp;  // temperature data
69         mdata[2]=humi;  // humidity data
70         mdata[3]=lux;   // luminosity data
71         //console.log(mdata);
72         var iot = new Sensor({date:dStr, temperature:temp, humidity:humi, luminosity:lux});
73         // save iot data to MongoDB
74         iot.save(function(err, iot) {
75             if(err) return handleEvent(err);
76             iot.info();  // Display the information of iot data  on console.
77         })
78         io.sockets.emit('message', mdata);  // send data to all clients
79     } else {  // error
80         console.log(readData);
81     }
82 });
```

A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.4 cds_dht22_mongodb.js

```
85  io.sockets.on('connection', function (socket) {
86      // If socket.io receives message from the client browser then
87      // this call back will be executed.
88      socket.on('message', function (msg) {
89          console.log(msg);
90      });
91      // If a web browser disconnects from Socket.IO then this callback
92      socket.on('disconnect', function () {
93          console.log('disconnected');
94      });
95  });
96
97  // helper function to get a nicely formatted date string
98  function getDateString() {
99      var time = new Date().getTime();
100     // 32400000 is (GMT+9 Korea, GimHae)
101     // for your timezone just multiply +/-GMT by 3600000
102     var datestr = new Date(time +32400000).
103     toISOString().replace(/T/, ' ').replace(/Z/, '');
104     return datestr;
105 }
```

A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.5 cds_dht22_mongodb.js → result (^B)

```
mongo db connection OK.
iotInfo: Current date: 2018-01-24 17:13:51.449, Temp: 18.6, Humi: 10.1, Lux: 179
iotInfo: Current date: 2018-01-24 17:13:53.720, Temp: 18.6, Humi: 10.1, Lux: 178
iotInfo: Current date: 2018-01-24 17:13:55.992, Temp: 18.6, Humi: 10.1, Lux: 178
iotInfo: Current date: 2018-01-24 17:13:58.264, Temp: 18.6, Humi: 10.1, Lux: 179
iotInfo: Current date: 2018-01-24 17:14:00.536, Temp: 18.6, Humi: 10.1, Lux: 177
iotInfo: Current date: 2018-01-24 17:14:02.792, Temp: 18.6, Humi: 10.0, Lux: 177
iotInfo: Current date: 2018-01-24 17:14:05.065, Temp: 18.6, Humi: 10.0, Lux: 178
iotInfo: Current date: 2018-01-24 17:14:07.336, Temp: 18.6, Humi: 10.0, Lux: 179
iotInfo: Current date: 2018-01-24 17:14:09.608, Temp: 18.6, Humi: 10.0, Lux: 179
iotInfo: Current date: 2018-01-24 17:14:11.880, Temp: 18.6, Humi: 10.0, Lux: 177
iotInfo: Current date: 2018-01-24 17:14:14.152, Temp: 18.6, Humi: 10.0, Lux: 180
```

3. cds_dht22_mongodb.js → Check documents in Mongo shell

Mongo shell

> show dbs

> use iot

> show collections

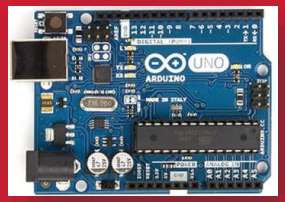
> db.sensors.find() .pretty()

명령 프롬프트 - mongo

```
> show dbs
aa00    0.000GB
admin   0.000GB
config  0.000GB
iot     0.000GB
local   0.000GB
test    0.000GB
test2   0.000GB
> use iot
switched to db iot
> show collections
sensors
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5a683ff83cdf6353104a5463"),
        "date" : "2018-01-24 17:12:40.708",
        "temperature" : "18.6",
        "humidity" : "10.1",
        "luminosity" : "178",
        "__v" : 0
}
{
        "_id" : ObjectId("5a683ffa3cdf6353104a5464"),
        "date" : "2018-01-24 17:12:42.979",
        "temperature" : "18.7",
        "humidity" : "10.3",
        "luminosity" : "179",
        "__v" : 0
}
{
        "_id" : ObjectId("5a683ffd3cdf6353104a5465"),
        "date" : "2018-01-24 17:12:45.251",
        "temperature" : "18.6",
        "humidity" : "10.2",
        "luminosity" : "180",
        "__v" : 0
}
```

Save as

AAnn_iot_mongdb.png



Arduino

& Node.js

& MongoDB

& Express server

A5.9.6 DHT22 + CdS + Node.js + MongoDB

1. Install express server

- Go to cds_dht22 project
- npm install --save express
- package.json

```
{
  "name": "cds_dht22",
  "version": "1.0.0",
  "description": "cds-dht22-node project",
  "main": "cds_dht22_node.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "aa00",
  "license": "MIT",
  "dependencies": {
    "express": "^4.16.2",
    "mongoose": "^5.0.1",
    "serialport": "^4.0.7",
    "socket.io": "^1.7.3"
  }
}
```

A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.1 cds_dht22_express.js

```
1  // cds_dht22_express.js
2
3  // Express
4  var express = require('express');
5  var app = express();
6  var web_port = 3030;  // express port
7
8  // MongoDB
9  var mongoose = require('mongoose');
10 var Schema = mongoose.Schema;  // Schema object
11 // MongoDB connection
12 mongoose.connect('mongodb://localhost:27017/iot'); // DB name
13 var db = mongoose.connection;
14 db.on('error', console.error.bind(console, 'connection error:'));
15 db.once('open', function callback () {
16         console.log("mongo db connection OK.");
17 });
18 // Schema
19 var iotSchema = new Schema({
20     date : String,
21     temperature : String,
22     humidity : String,
23     luminosity: String
24 });
25 var Sensor = mongoose.model("Sensor", iotSchema);  // sensor data model
```

A5.9.6 DHT22 + CdS + Node.js + MongoDB

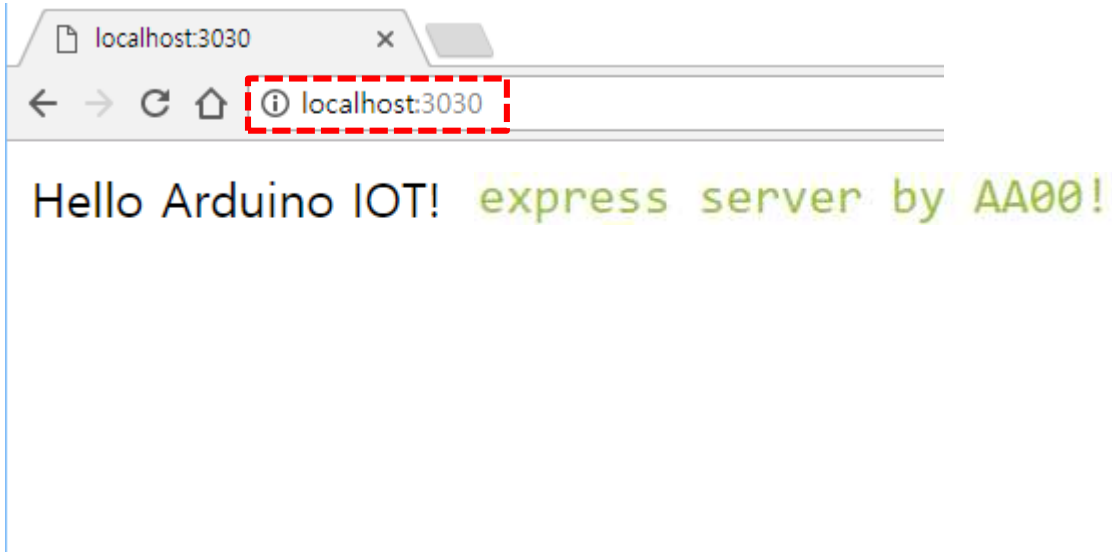
2.2 cds_dht22_express.js

```
27 // Web routing addrebss
28 app.get('/', function (req, res) {  // localhost:3030/
29   res.send('Hello Arduino IOT: express server by AA00!');
30 });
31 // find all data & return them
32 app.get('/iot', function (req, res) {
33     Sensor.find(function(err, data) {
34         res.json(data);
35     });
36 });
37 // find data by id
38 app.get('/iot/:id', function (req, res) {
39     Sensor.findById(req.params.id, function(err, data) {
40         res.json(data);
41     });
42 });
43
44 // Express WEB
45 app.use(express.static(__dirname + '/public'));  // WEB root folder
46 app.listen(web_port);  // port 3030
47 console.log("Express_IOT is running at port:3030");
```

2.3 cds_dht22_express.js → Run

```
Express_IOT is running at port:3030
mongo db connection OK.
```

2.4 cds_dht22_express.js → routing1, http://localhost:3030/



A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.5 cds_dht22_express.js → routing2 http://localhost:3030/iot

localhost:3030/iot

localhost:3030/iot

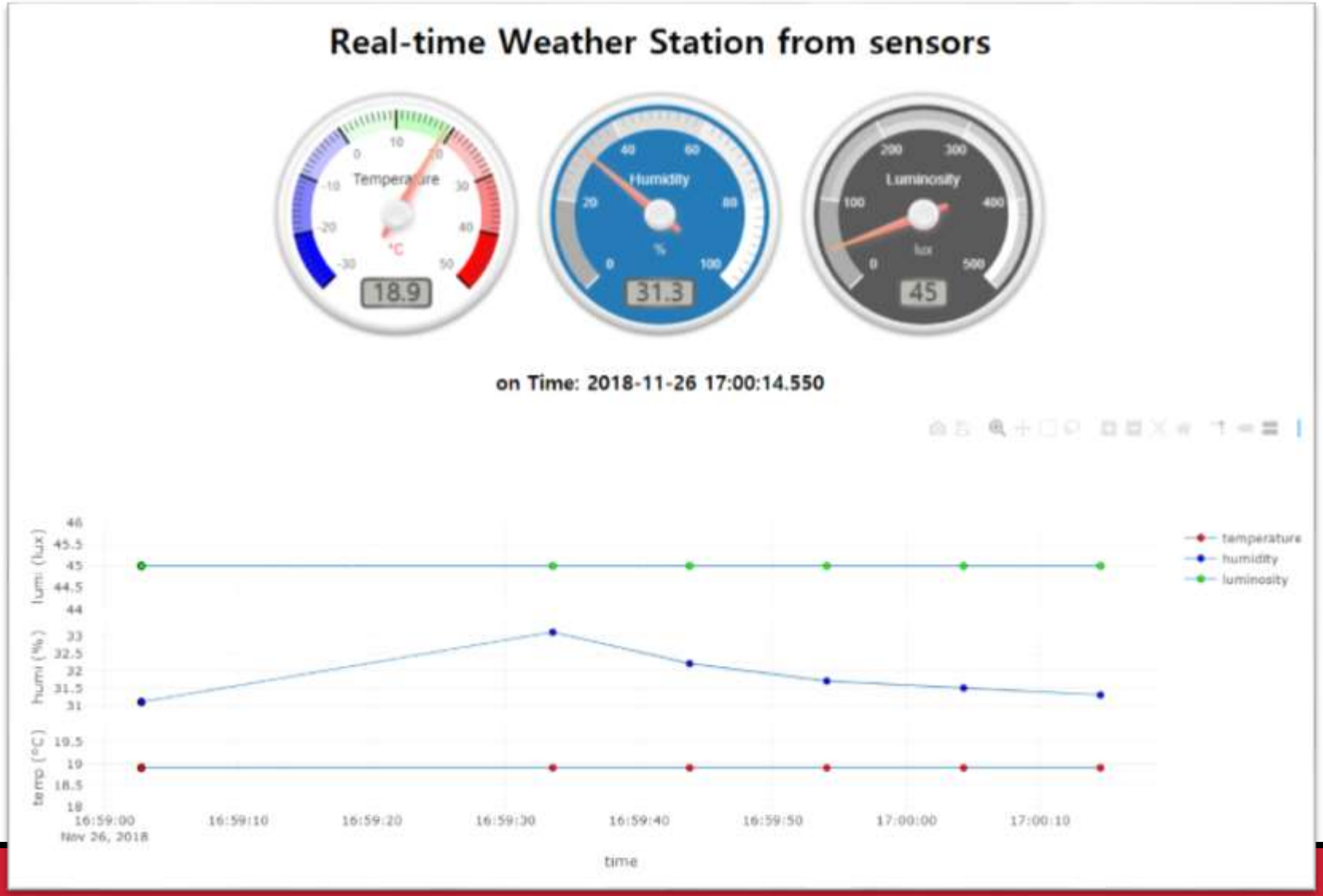

```
[{"_id":"5a683ff83cdf6353104a5463","date":"2018-01-24
17:12:40.708","temperature":"18.6","humidity":"10.1","luminosity":"178","__v":0},
{"_id":"5a683ffa3cdf6353104a5464","date":"2018-01-24
17:12:42.979","temperature":"18.7","humidity":"10.3","luminosity":"179","__v":0},
{"_id":"5a683ffd3cdf6353104a5465","date":"2018-01-24
17:12:45.251","temperature":"18.6","humidity":"10.2","luminosity":"180","__v":0},
{"_id":"5a683fff3cdf6353104a5466","date":"2018-01-24
17:12:47.523","temperature":"18.6","humidity":"10.2","luminosity":"179","__v":0},
{"_id":"5a6840013cdf6353104a5467","date":"2018-01-24
17:12:49.779","temperature":"18.6","humidity":"10.2","luminosity":"177","__v":0},
{"_id":"5a6840043cdf6353104a5468","date":"2018-01-24
17:12:52.052","temperature":"18.6","humidity":"10.2","luminosity":"178","__v":0},
{"_id":"5a6840063cdf6353104a5469","date":"2018-01-24
17:12:54.322","temperature":"18.6","humidity":"10.2","luminosity":"176","__v":0},
{"_id":"5a6840083cdf6353104a546a","date":"2018-01-24
17:12:56.594","temperature":"18.6","humidity":"10.2","luminosity":"176","__v":0},
{"_id":"5a68400a3cdf6353104a546b","date":"2018-01-24
17:12:58.866","temperature":"18.6","humidity":"10.2","luminosity":"178","__v":0},
{"_id":"5a68400d3cdf6353104a546c","date":"2018-01-24
17:13:01.138","temperature":"18.6","humidity":"10.2","luminosity":"178","__v":0},
{"_id":"5a68400f3cdf6353104a546d","date":"2018-01-24
17:13:03.410","tempe
```


Save as

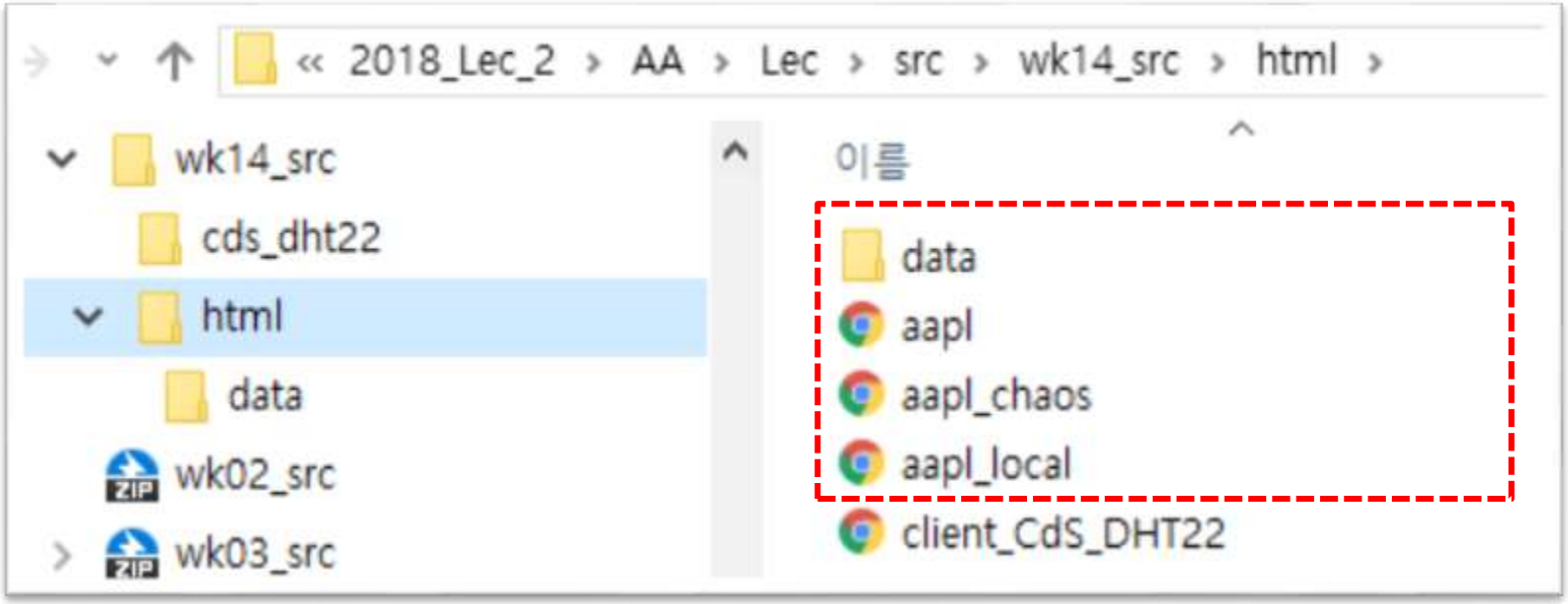
AAnn_iot_mongodb_web.png

A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.7 copy cds_dht22_client.html & gauge.min.js → ./public/ subfolder
http://localhost:3030/cds_dht22_client.html



2.8 CORS bug



Apple 사의 주가그래프를 그리는 html client 3개를 실행하고 결과를 비교.
→ Local file에 접근 불허
→ CORS problem

→ public 폴더로 html,data를 복사한 후에 비교.

A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.9 CORS patch on the express server → cds_dht22_express.js
Node cmd에서 'cors' module 설치 (version 2.84 이상)
npm install –save cors

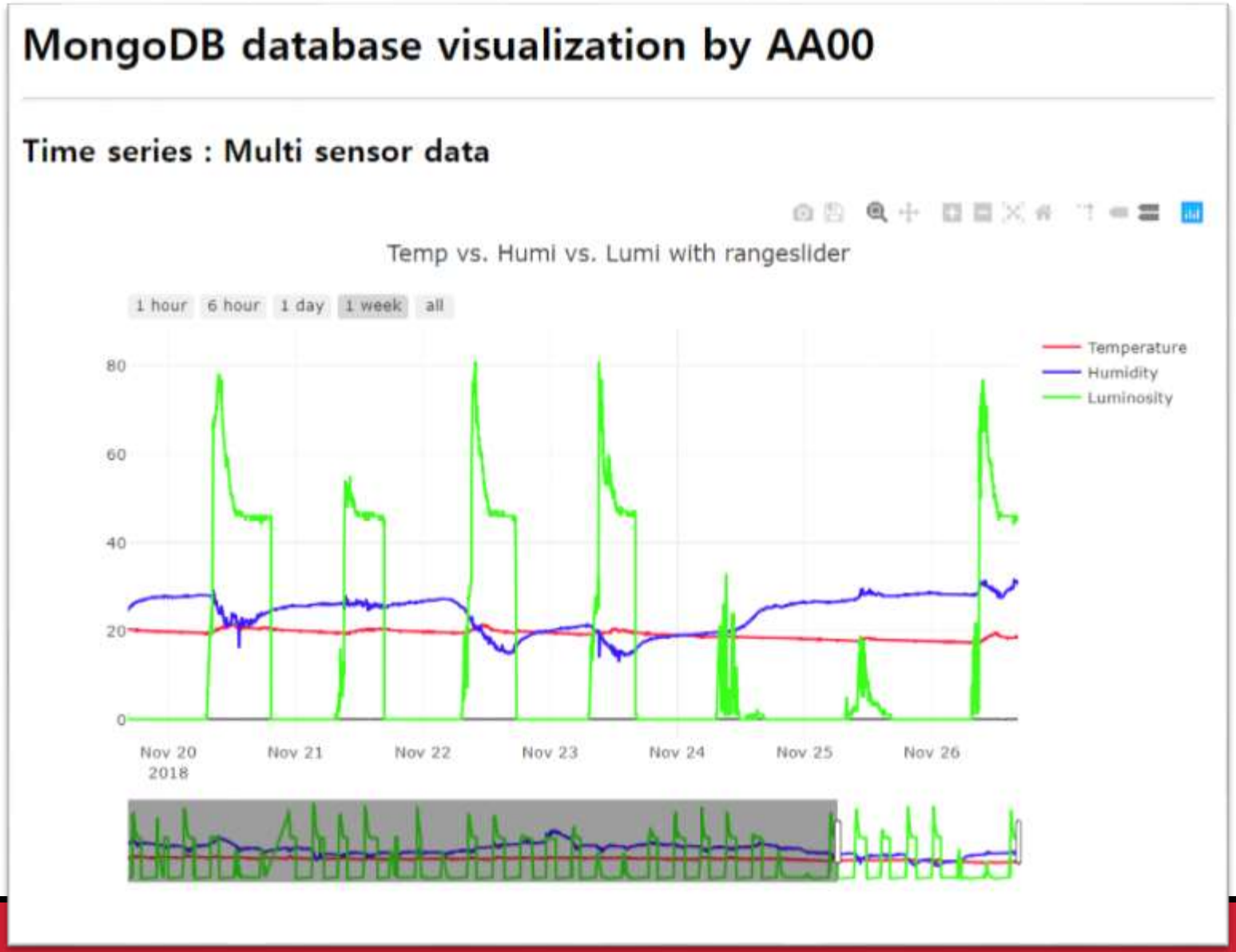
```
// cds_dht22_express.js
// Express with CORS
var express = require('express');
var cors = require('cors');  // CORS: Cross Origin Resource Sharing
var app = express();
// CORS
app.use(cors());

var web_port = 3030;  // express port
// MongoDB
var mongoose = require('mongoose');
var Schema = mongoose.Schema;  // Schema object
// MongoDB connection
mongoose.connect('mongodb://localhost:27017/iot11'); // DB name
var db = mongoose.connection;
db.on('error', console.error.bind(console, 'connection error:'));
db.once('open', function callback () {
        console.log("mongo db connection OK.");
});
```

Web monitoring

A5.9.7 DHT22 + CdS + Node.js + MongoDB

Web monitoring-2: week



A5.9.8 DHT22 + CdS + Node.js + MongoDB

3.1 Web client: client_iotDB.html

client_iotDB.html

```html
<!DOCTYPE html>
<head>
    <meta charset="utf-8">
    <!-- Plotly.js -->
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
</head>
<body>
    <h1>MongoDB database visualization by AA00</h1>
    <hr>
    <h2>Time series : Multi sensor data</h2>

    <!-- Plotly chart will be drawn inside this DIV -->
    <div id="myDiv" style="width: 900px;height: 600px"></div>

```

A5.9.7 DHT22 + CdS + Node.js + MongoDB

3.2 Web client: client_iotDB.html

```
<script>
    <!-- JAVASCRIPT CODE GOES HERE -->

    Plotly.d3.json("http://localhost:3030/iot", function(err, json){
        //alert(json);
         alert(JSON.stringify(json));  // It works!!!
        //alert(JSON.parse(eval(json)));
        if(err) throw err;

        var date = [];
        var temp = [];
        var humi = [];
        var lumi = [];
        var jsonData = eval(JSON.stringify(json));
        //alert(jsonData.length);
        //alert(jsonData[2].luminosity);

        for (var i = 0; i < jsonData.length; i++) {
            date[i] = jsonData[i].date;
            temp[i] = jsonData[i].temperature ;
            humi[i] = jsonData[i].humidity;
            lumi[i] = jsonData[i].luminosity;

        }
```

3.3 Web client: client_iotDB.html – data & layout

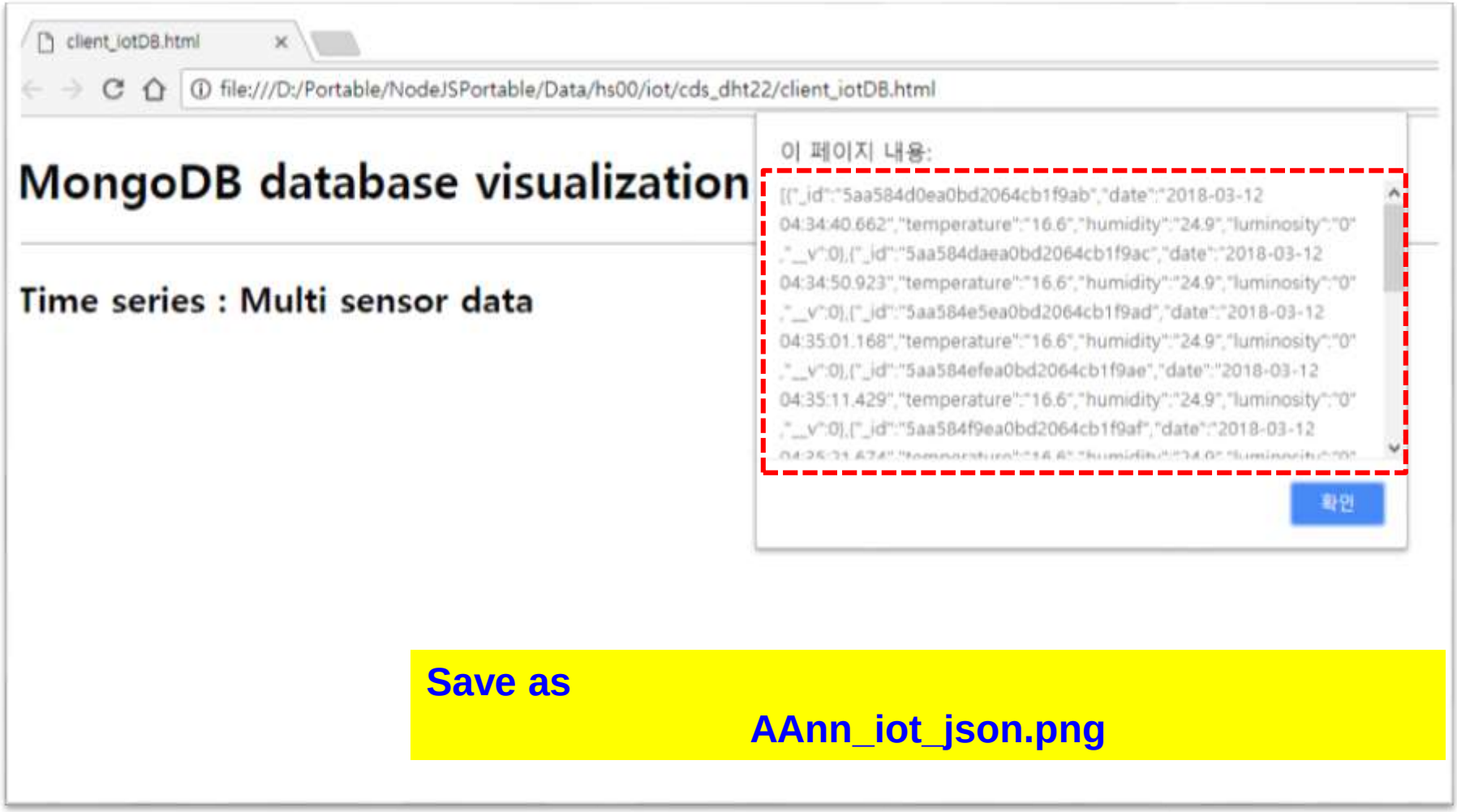
```
// time series of sensor data
var trace1 = {
    type: "scatter",
    mode: "lines",
    name: 'Temperature',
    x: date,
    y: temp,
    line: {color: '#fc1234'}
}

var trace2 = {
    type: "scatter",
    mode: "lines",
    name: 'Humidity',
    x: date,
    y: humi,
    line: {color: '#3412fc'}
}
var trace3 = {
    type: "scatter",
    mode: "lines",
    name: 'Luminosity',
    x: date,
    y: lumi,
    line: {color: '#34fc12'}
}
var data = [trace1, trace2, trace3];
```

```
// Layout with builtin rangeslider
var layout = {
    title: 'Temp vs. Humi vs. Lumi with rangeslider',
    xaxis: {
        autorange: true,
        range: [date[0], date[date.length-1]],
        rangeselector: {buttons: [
            {
                count: 1,
                label: '1 hour',
                step: 'hour',
                stepmode: 'backward'
            },
            {
                count: 6,
                label: '6 hour',
                step: 'hour',
                stepmode: 'backward'
            },
            {
                count: 24,
                label: '1 day',
                step: 'hour',
                stepmode: 'backward'
            },
            {
                count: 7,
                label: '1 week',
                step: 'day',
                stepmode: 'backward'
            },
            {step: 'all'}
            ]},
            rangeslider: {range: [date[0], date[date.length-1]]},
            type: 'date'
        },
        yaxis: {
            autorange: true,
            range: [0, 300],
            type: 'linear'
        }
    };

    Plotly.newPlot('myDiv', data, layout);
})
```

3.4 Web client: client_iotDB.html – load iot data in json file

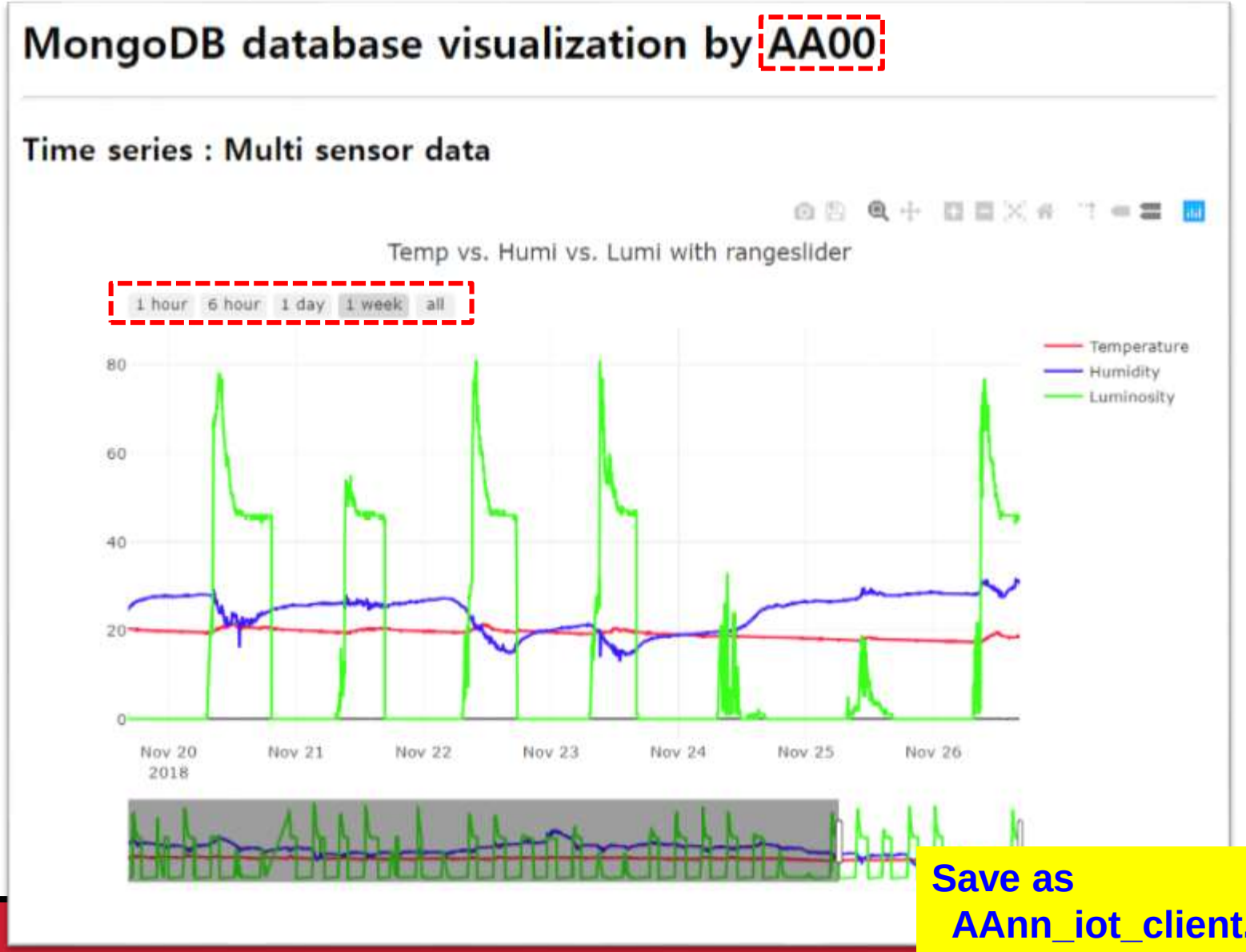


Save as

AAnn_iot_json.png

A5.9.7 DHT22 + CdS + Node.js + MongoDB

3.5 Web client: client_iotDB.html – iot DB monitoring



Save as AAnn_iot_client.png

MongoDB data management

- Query in mongo shell
- Export & import MongoDB
- Using and understanding iot data with Python (or R)

A5.9.8 MongoDB management

1. Query in Mongo shell

db.sensors.count() → sensors collection에 있는 도큐먼트 (문서)의 수

db.sensors.find().sort({_id: 1}).limit(10) → 오래된 document 10개 추출

db.sensors.find().sort({_id: -1}).limit(10) → 최근 document 10개 추출

db.sensors.find({date: {$gt: “2018-11-26 22:26:05”}}) → 특정 시간 이후 document 추출

db.sensors.find({temperature: {$gt: 29}}) → 온도가 29도를 넘는 document 추출

https://docs.mongodb.com/manual/tutorial/query-documents/

A5.9.8 MongoDB management

2. Import or export MongoDB (windows cmd 창에서 실행)

- mongoimport -d dbName -c collectionName --type csv --headerline --file fileName.csv

- mongoexport -d dbName -c collectionName --fields <field1,field2,…> --limit=nn --type csv --out fileName.csv

json 또는 csv 파일로 import/export

https://docs.mongodb.com/manual/reference/program/mongoimport/

https://docs.mongodb.com/manual/reference/program/mongoexport/

A5.9.8 MongoDB management

[Tip] iot db의 최근 데이터 500개를 csv 파일 (s500.csv)로 저장할 때,

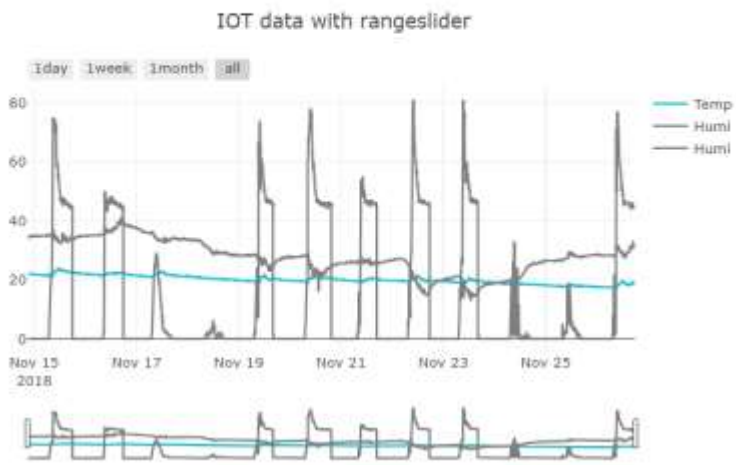
- **mongoexport –d iot -c sensors --sort “{_id: -1}” --limit=500 --fields date,temperature,humidity,luminosity --type=csv --out s500.csv**

```
C:₩Users₩biochaos>mongoexport -d iot11 -c sensors --sort "{_id:-1}" --limit=100000 --type=csv --fields date,temperature,
humidity,luminosity --out iot_chaos.csv
2018-11-26T17:50:23.577+0900    connected to: localhost
2018-11-26T17:50:24.576+0900    [################.........]  iot11.sensors  64000/100000  (64.0%)
2018-11-26T17:50:24.797+0900    [########################]  iot11.sensors  100000/100000  (100.0%)
2018-11-26T17:50:24.798+0900    exported 100000 records
```

	A	B	C	D
1	date	temperatu	humidity	luminosity
2	50:18.6	18.9	31.6	45
3	50:08.4	18.9	31.6	45
4	49:58.1	18.9	31.6	45
5	49:47.8	19	31.7	45
6	49:37.6	19	31.7	45
7	49:27.3	18.9	31.7	45
8	49:17.1	18.9	31.6	45

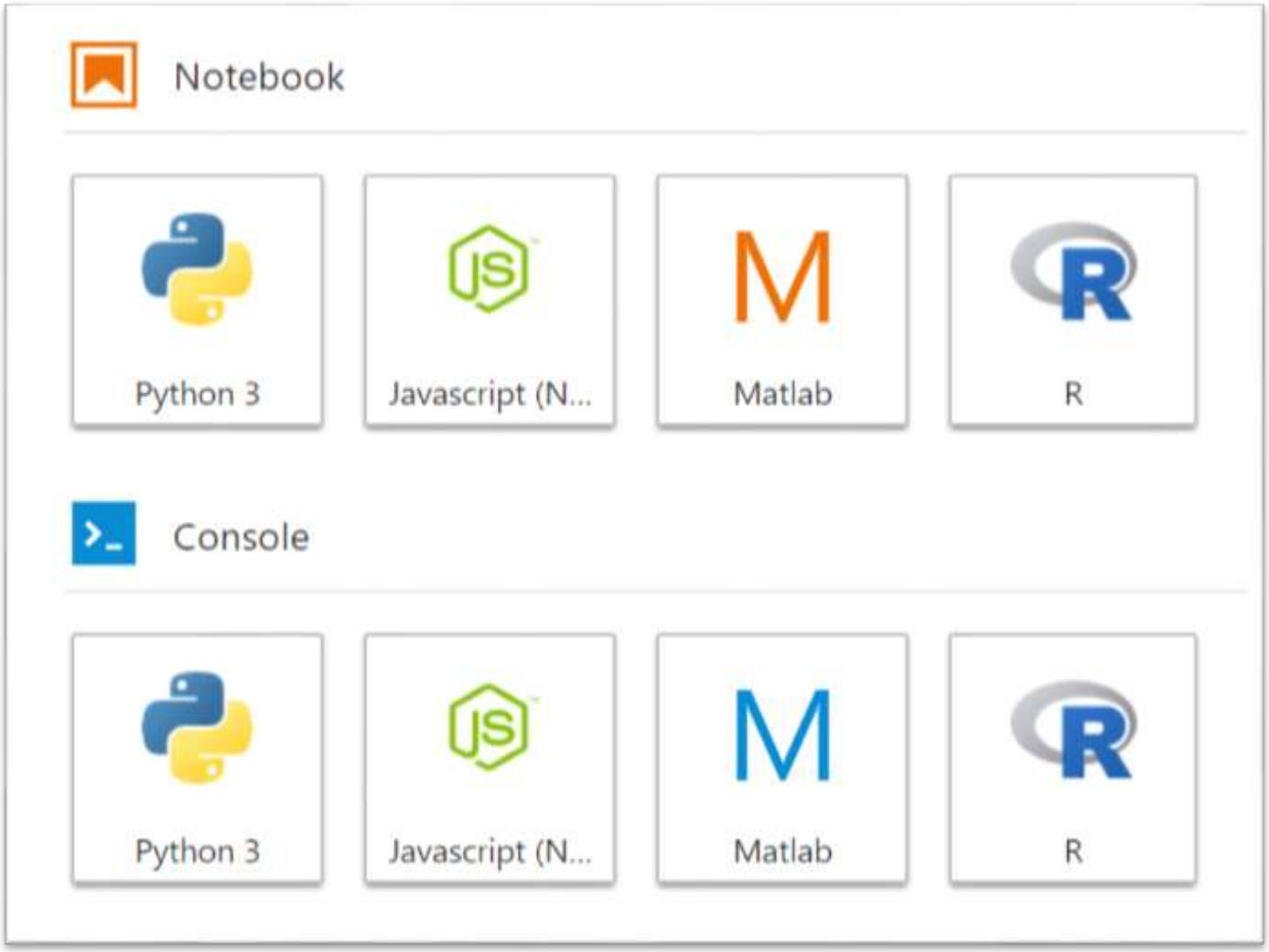
Data visualization by AAnn

Time series by AAnn



A5.9.8 IOT data mining

3. How to use and understand iot data? → Python(or R) in Jupyter lab



A5.9.8 IOT data mining

3. How to use and understand iot data? → Python in Google Colab

iot_json의 사본

파일 수정 보기 삽입 런타임 도구 도움말

Pandas: access to the remote json from MongoDB

- The json file is generated on the fly from the express server of Node.js.
- The data stored in MongoDB are saved in the json file.
- The data are composed of three time series; temperature, humidity, and luminosity.

```
import pandas as pd
```

```
# loading json file from MongoDB via web (CORS, port=3030)
url='http://chaos.inje.ac.kr:3030/iot'
j1=pd.read_json(url)
```

```
j1.head()
```

	__v	_id	date	humidity	luminosity	temperature
0	0	5bce24218d1ec32774d781a9	2018-10-23 04:25:21.349	39.7	0	23.2
1	0	5bce242b8d1ec32774d781aa	2018-10-23 04:25:31.594	39.7	0	23.2

A5.9.8 IOT data mining

3.1 How to use and understand iot data? → iot_json.ipynb

Redwoods / Arduino

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki

Branch: master | Arduino / ar-iot / py-pandas /

Redwoods Add files via upload

..

data	Add files via upload
iot_csv.ipynb	Colaboratory를 통해 생성됨
iot_json.ipynb	Colaboratory를 통해 생성됨

A5.9.8 MongoDB management

3.2 Loading data … → iot_json.ipynb

[1]
```
import pandas as pd
```

[2]
```
# loading json file from MongoDB via web (CORS, port=3030)
url="http://chaos.inje.ac.kr:3030/iot"
j1=pd.read_json(url)
```

1. Express 서버에서 MongoDB에 접속한다.
2. 아두이노에서 만들어져 전송되어 MongoDB에 저장되고 있는 센서 데이터를 json 파일로 가져온다.

[3]
```
j1.head()
```

	__v	_id	date	humidity	luminosity	temperature
0	0	5bce24218d1ec32774d781a9	2018-10-23 04:25:21.349	39.7	0	23.2
1	0	5bce242b8d1ec32774d781aa	2018-10-23 04:25:31.594	39.7	0	23.2
2	0	5bce24358d1ec32774d781ab	2018-10-23 04:25:41.855	39.7	0	23.2
3	0	5bce24408d1ec32774d781ac	2018-10-23 04:25:52.100	39.7	0	23.2
4	0	5bce244a8d1ec32774d781ad	2018-10-23 04:26:02.360	39.7	0	23.2

3.3 Make dataframe from json data

Dataframe with date and three sensor values(temperature, humidity, luminosity)

```
[ ]  iot_data = j1[['date', 'temperature', 'humidity', 'luminosity']]
```

Json 객체에서 필요한 항목을 선택해서 pandas의 dataframe을 구성한다.

```
[ ]  iot_data.shape
```

```
(340230, 4)
```

```
[ ]  iot_data.head()
```

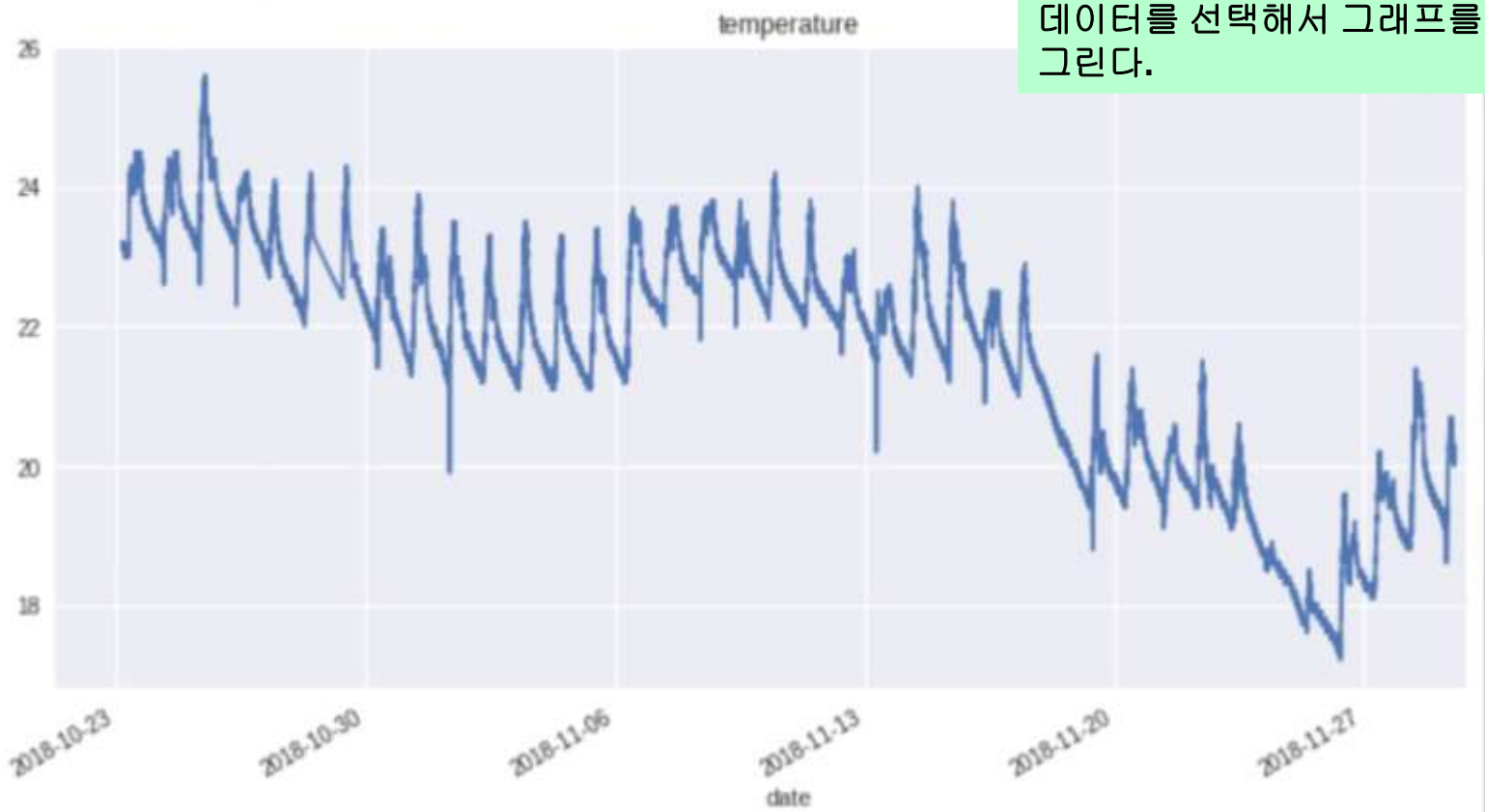
	date	temperature	humidity	luminosity
0	2018-10-23 04:25:21.349	23.2	39.7	0
1	2018-10-23 04:25:31.594	23.2	39.7	0
2	2018-10-23 04:25:41.855	23.2	39.7	0
3	2018-10-23 04:25:52.100	23.2	39.7	0
4	2018-10-23 04:26:02.360	23.2	39.7	0

3.4.1 Plot iot data (time series)

Plot time series of sensor data

```
[ ] iot_data.plot(x='date', y='temperature', figsize=(12,6), title='temperature')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2596e438>
```



Dataframe에서 시간과 온도 데이터를 선택해서 그래프를 그린다.

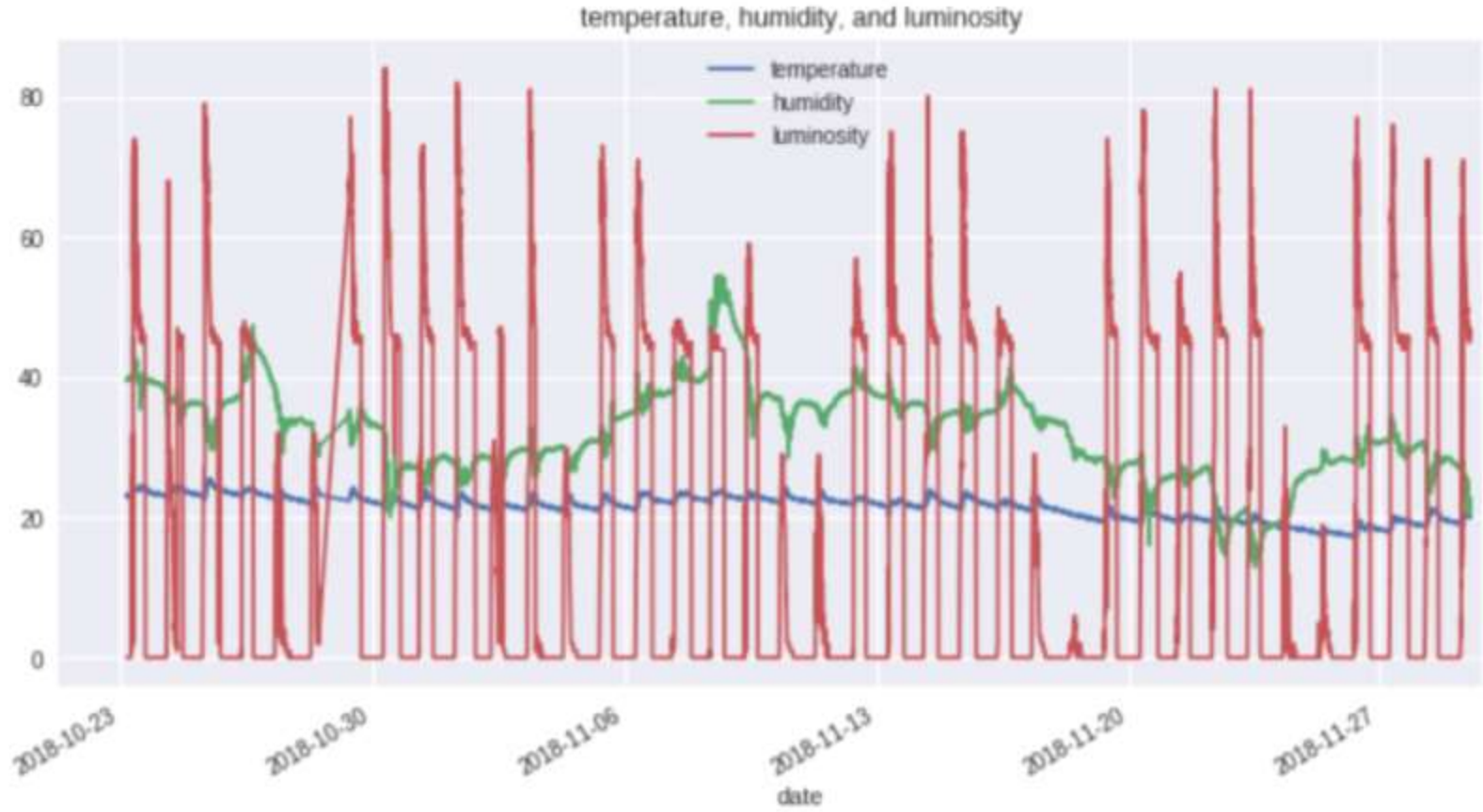
A5.9.8 IOT data mining

3.4.2 Plot iot data (time series)

```
# Plot of ['temperature','humidity','luminosity']
iot_data.plot(x='date', y=['temperature','humidity','luminosity'], figsize=(12,6),
              title='temperature, humidity, and luminosity')
```

```
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_core.py:1716:
  series.name = label
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b28813128>
```

Dataframe에서 시간과 세 개의 센서 데이터를 전부 선택해서 그래프를 그린다.

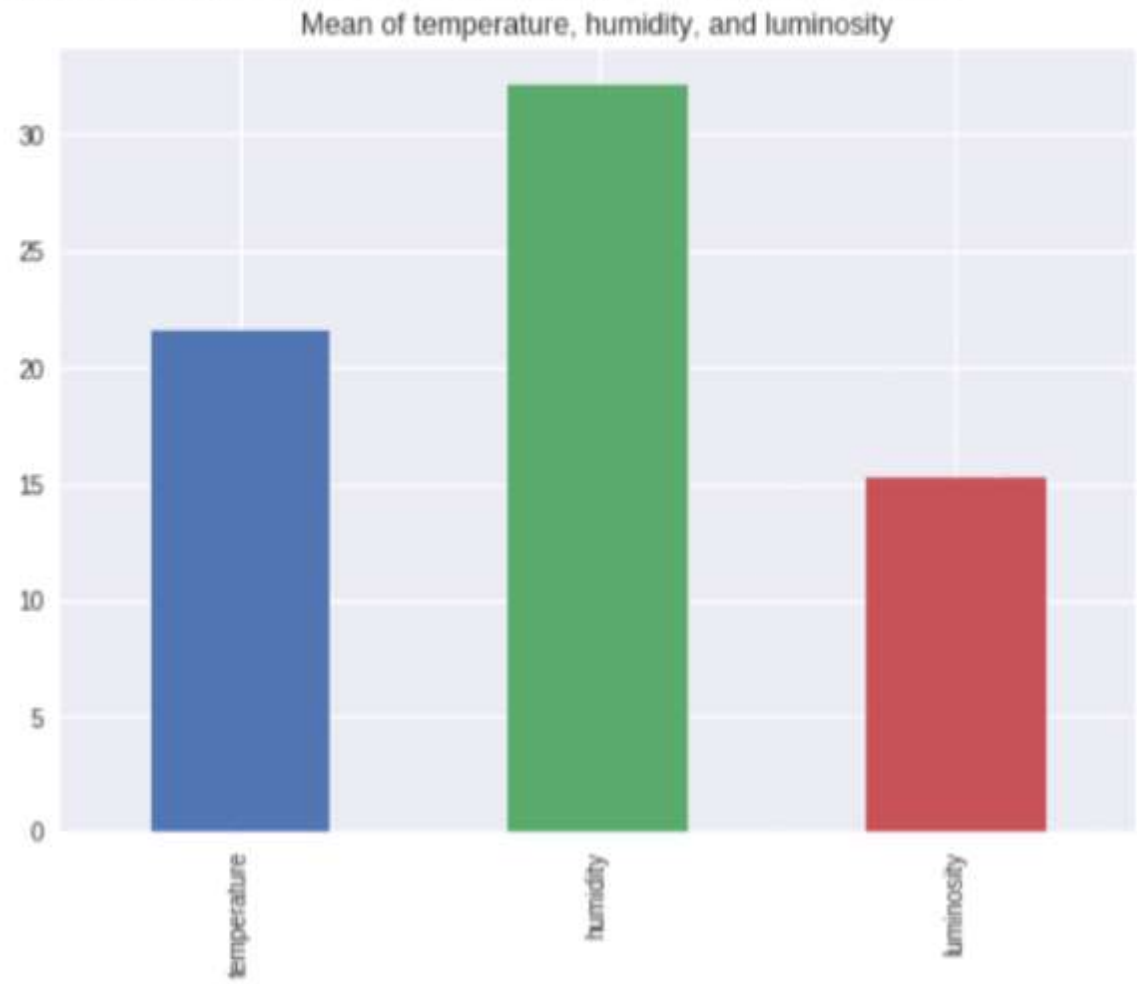


3.5 Plot mean of sensor data

```
iot_data[['temperature','humidity','luminosity']].mean().plot.bar(figsize=(8,6),
                                                                  title="Mean of temperature,
```

Dataframe에서 세 개의 센서 데이터의 평균을 구해서 그래프를 그린다.

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b297d9470>
```



A5.9.8 IOT data mining

3.6.1 Plot the change of sensor data over various time spans.

Set date as index of timestamp

```
[ ]  iot_data.set_index('date',inplace=True)
```

```
[ ]  iot_data.info() # timestamp index
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 307849 entries, 2018-10-23
Data columns (total 3 columns):
temperature    307849 non-null float64
humidity       307849 non-null float64
luminosity     307849 non-null int64
dtypes: float64(2), int64(1)
memory usage: 9.4 MB
```

```
iot_data.head()
```

	temperature	humidity	luminosity
date			
2018-10-23 04:25:21.349	23.2	39.7	0
2018-10-23 04:25:31.594	23.2	39.7	0
2018-10-23 04:25:41.855	23.2	39.7	0
2018-10-23 04:25:52.100	23.2	39.7	0
2018-10-23 04:26:02.360	23.2	39.7	0

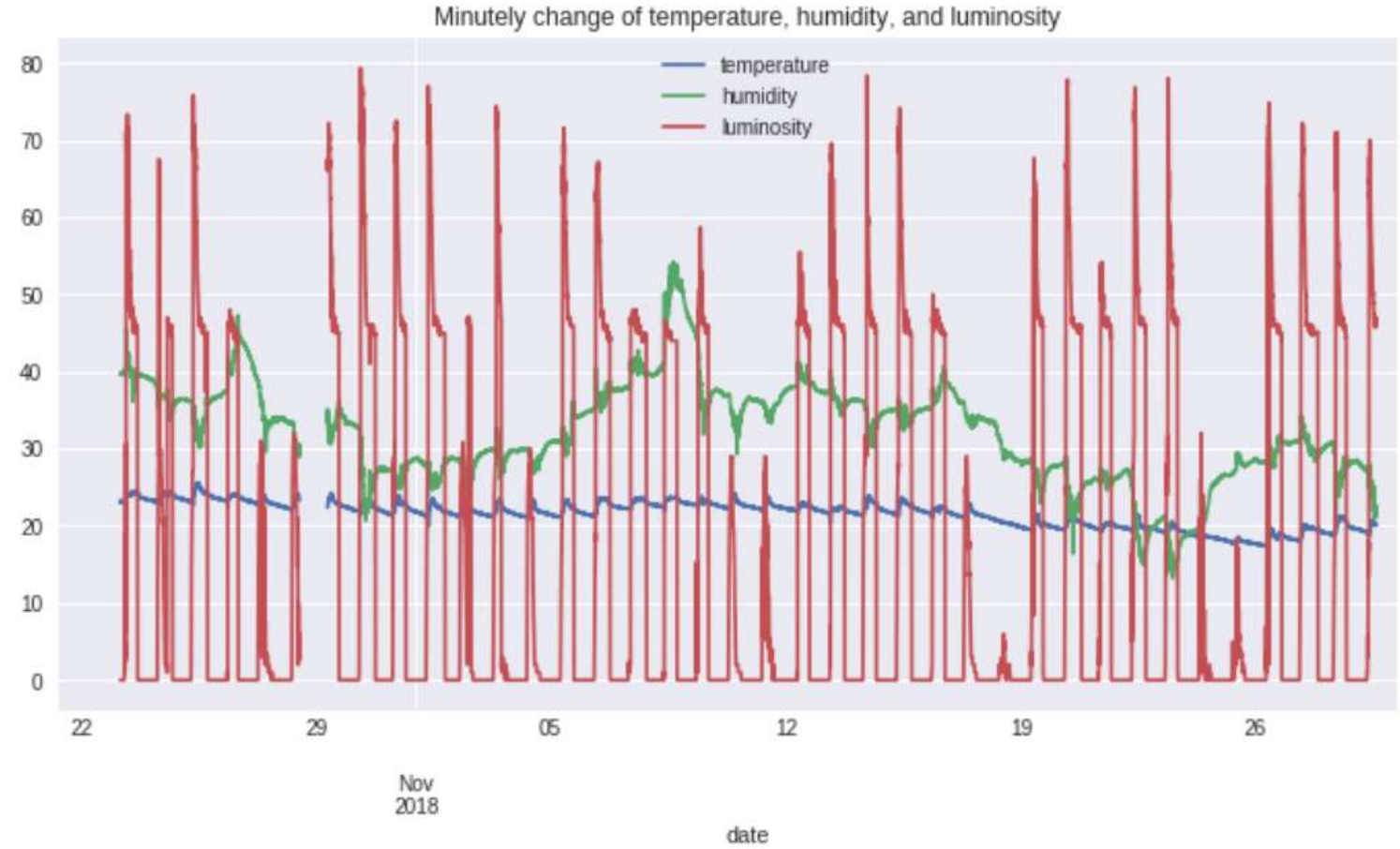
시간**(date)**을 **timestamp** 형으로
변경해서 데이터를 재구성한다**.**

A5.9.8 IOT data mining

3.6.2 Plot the change of sensor data over various time spans.
1 분당 평균 그래프

```
# Plot mean of the iot data per every minute
iot_data.resample('60S').mean().plot(figsize=(12,6),
                                     title='Minutely change of temperature, humidity, and lumi
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2b57c630>
```

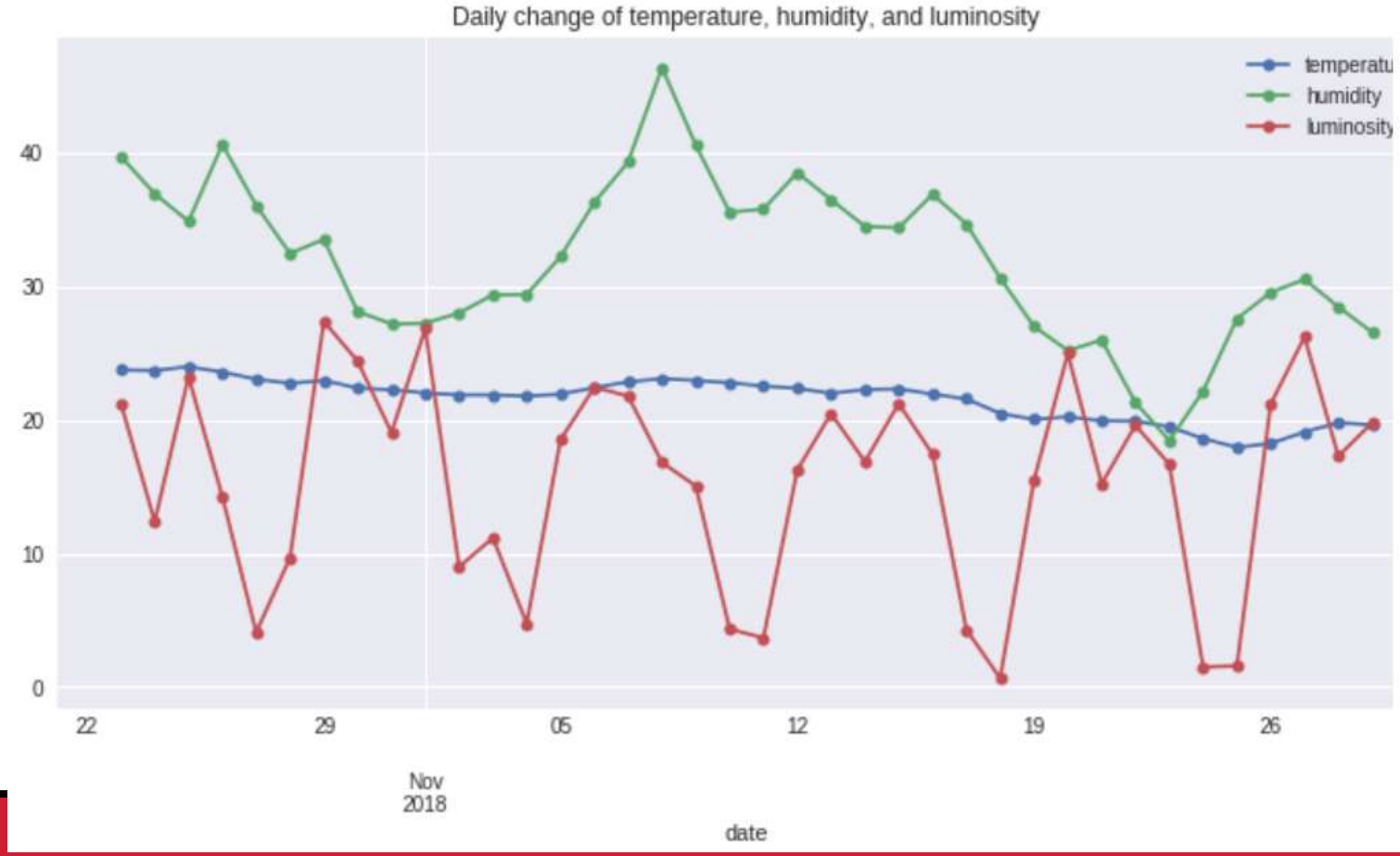


A5.9.8 IOT data mining

3.6.3 Plot the change of sensor data over various time spans.
1 일당 평균 그래프

```
# Plot mean of the iot data per every day
iot_data.resample('D').mean().plot(kind='line', marker='o', ms=6, figsize=(12,6),
                                   title='Daily change of temperature, humidity, and luminosit
```

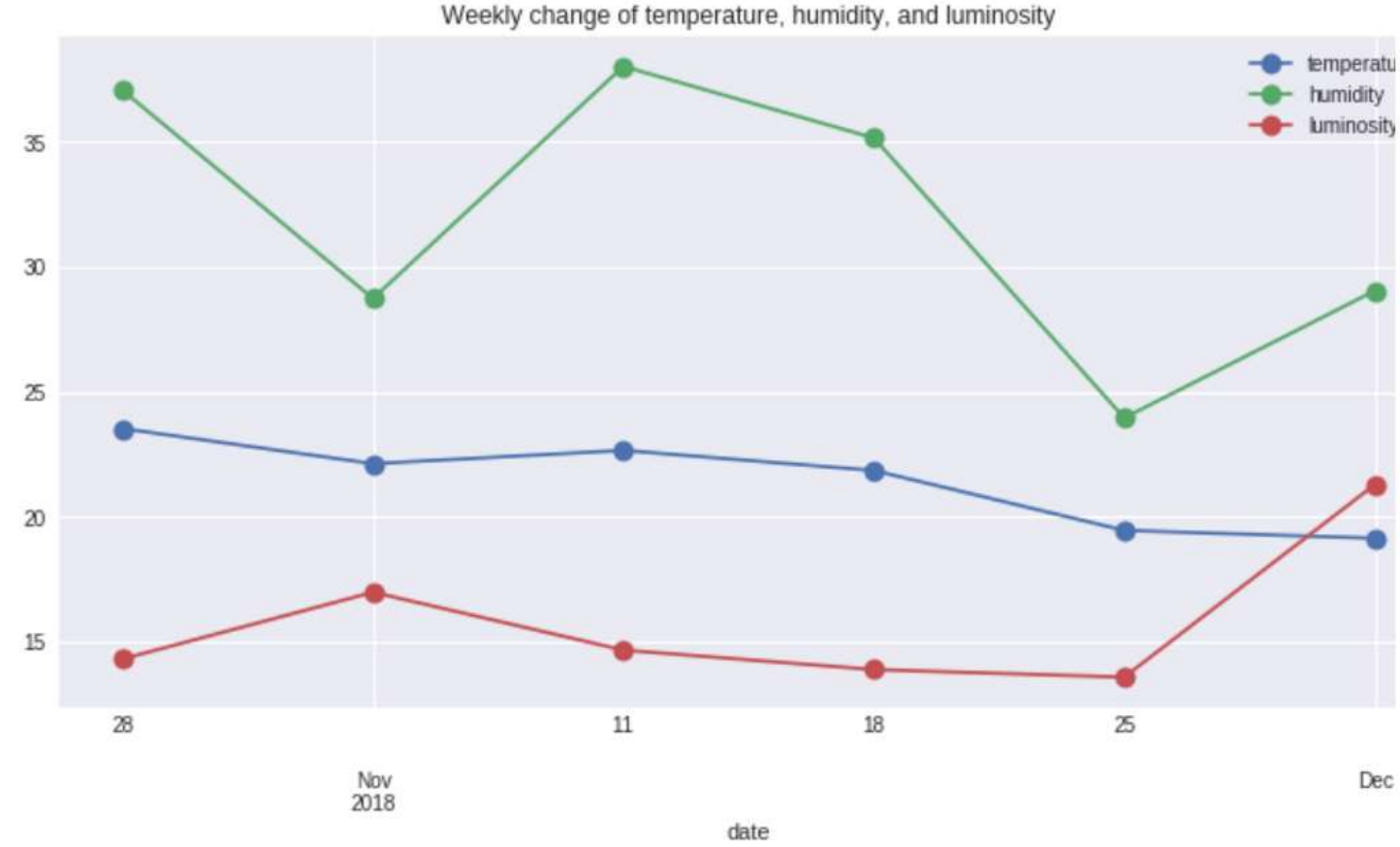
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2c7fb7f0>
```



3.6.3 Plot the change of sensor data over various time spans.
1 주당 평균 그래프

```
# Plot mean of the iot data per every week
iot_data.resample('W').mean().plot(kind='line', marker='o', ms=10,
                                   figsize=(12,6),
                                   title='Weekly change of temperature, humidity, and luminosi
```

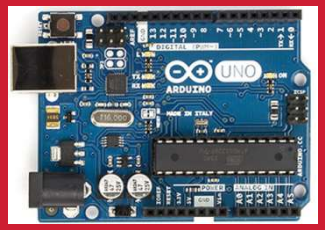
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2c8f8748>
```



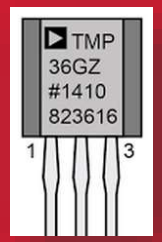
A5.9.8 IOT data mining - DIY

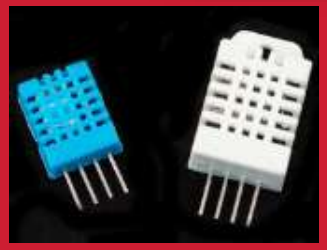
[DIY] wk14에 저장한 "AAnn_s1000.csv" 데이터를 colab으로 로딩한다.

- **iot_csv.ipynb 파일를 iot_csv_aann.ipynb로 저장한다.**
- **AAnn_s1000.csv" 데이터를 이용해서 1분, 5분, 10분 평균 그래프를 그린다.**
- **Colab에서 만든 iot_csv_aann.ipynb 파일을 github에 "arnn-iot" 저장장소(git repogitory)를 만들어서 올린다.**
- **사용한 AAnn_s1000.csv 파일은 "arnn-iot" 안에 'data' 폴더를 만들어서 올린다.**



[Practice]





- [wk15]
  - RT Data mining with Colab
  - Multi-sensor data(cds-dht22)
  - Complete your project
  - Upload file name : AAnn_Rpt11.zip

[wk15] Practice-11 AAnn_Rpt11.zip

- [Target of this week]
  - Complete your work in Colab.
  - Save your outcomes in your Github.

- Github에 파일 올리기

① Colab에서 파일 저장.

② 저장된 파일을 github에 "arnn-iot" 저장장소(git repogitory)를 만들어서 올린다.

③ 저장할 파일명

1. iot_csv.ipynb
2. iot_json.ipynb
3. iot_csv_arnn.ipynb

→ 이메일로 github 주소를 보내세요.

Email :
chaos21c@gmail.com

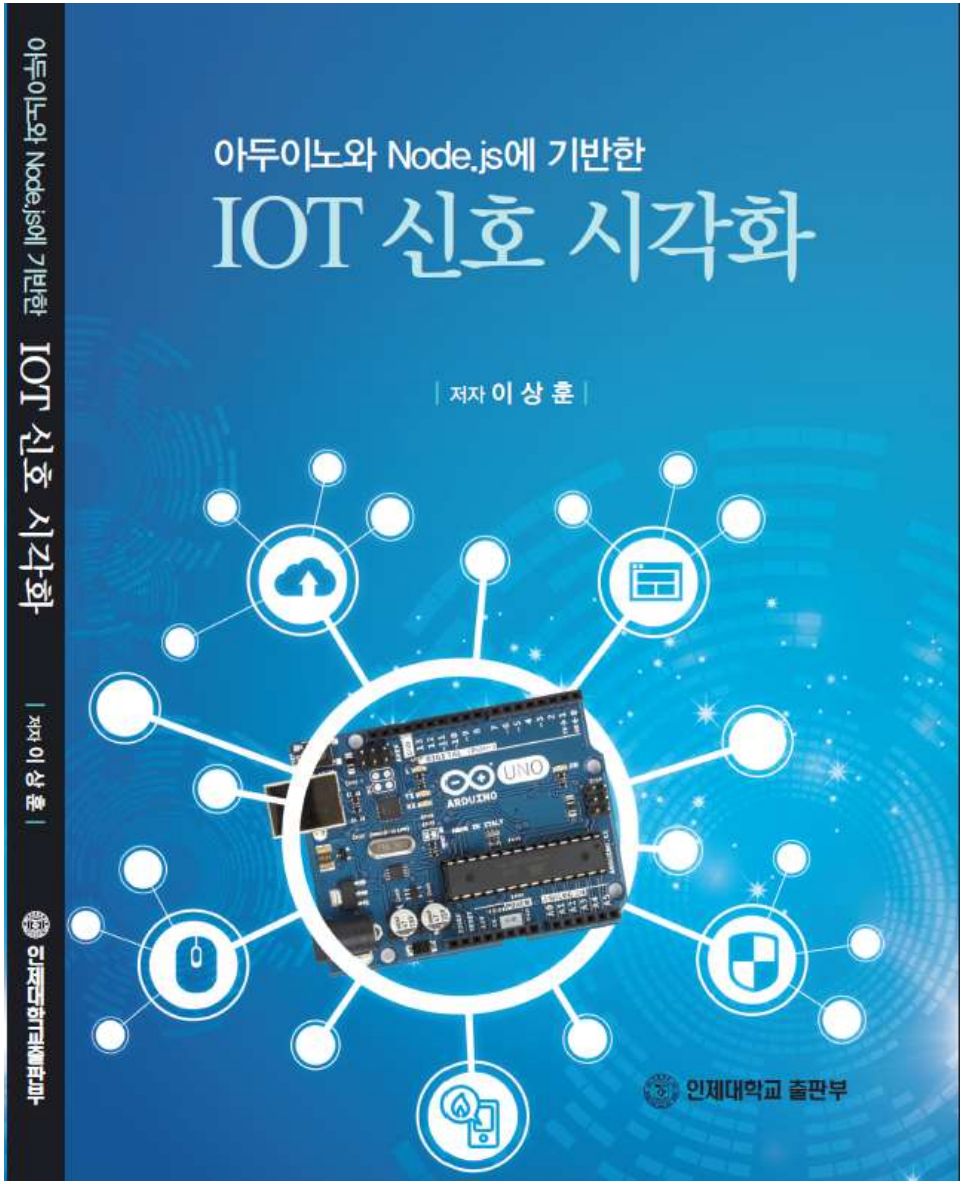
[제목 : id, 이름 (수정)]

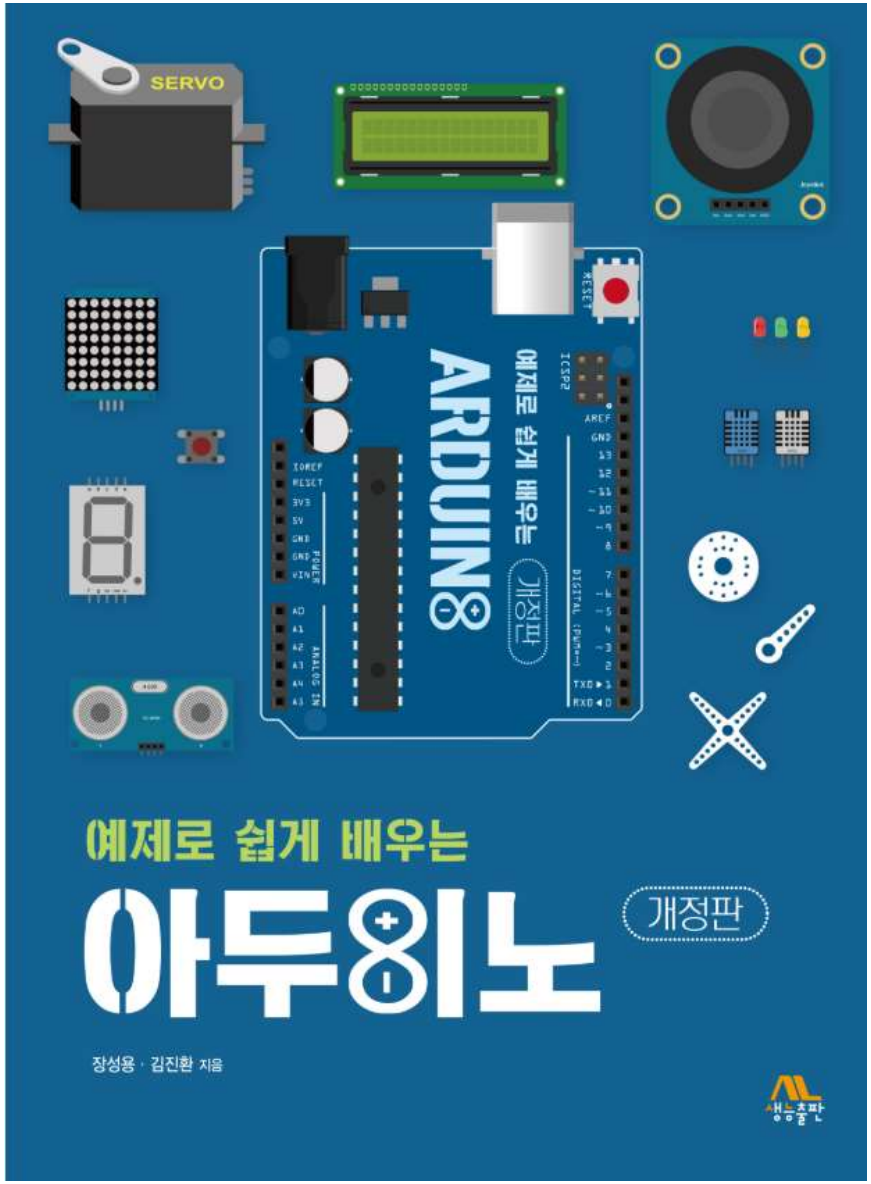
Lecture materials

References & good sites

- ✓ http://www.arduino.cc Arduino Homepage
- ✓ http://www.nodejs.org/ko Node.js
- ✓ https://plot.ly/ plotly
- ✓ https://www.mongodb.com/ MongoDB
- ✓ http://www.w3schools.com By w3schools
- ✓ http://www.github.com GitHub

주교재 및 참고도서


아두이노와 Node.js에 기반한
IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교 출판부
아두이노와 Node.js에 기반한 IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교출판부


SERVO
ARDUINO
예제로 쉽게 배우는
개정판
예제로 쉽게 배우는
아두이노
개정판
장성용 · 김진환 지음

Target of this class

Real-time Weather Station from sensors

on Time: 2018-01-22 17:58:31.012

Another target of this class