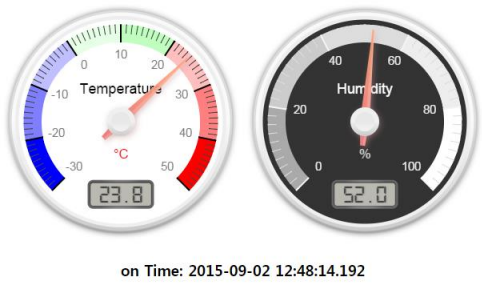
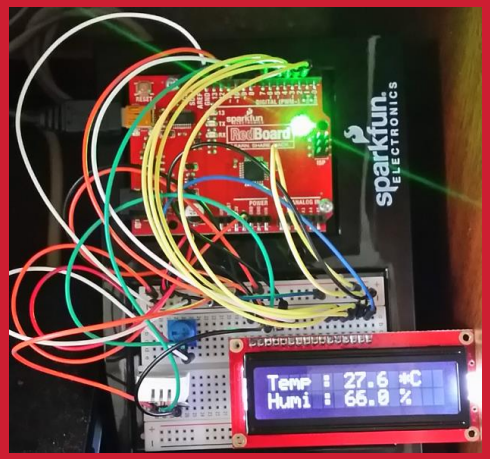
# Healthcare-IOT
[wk12]



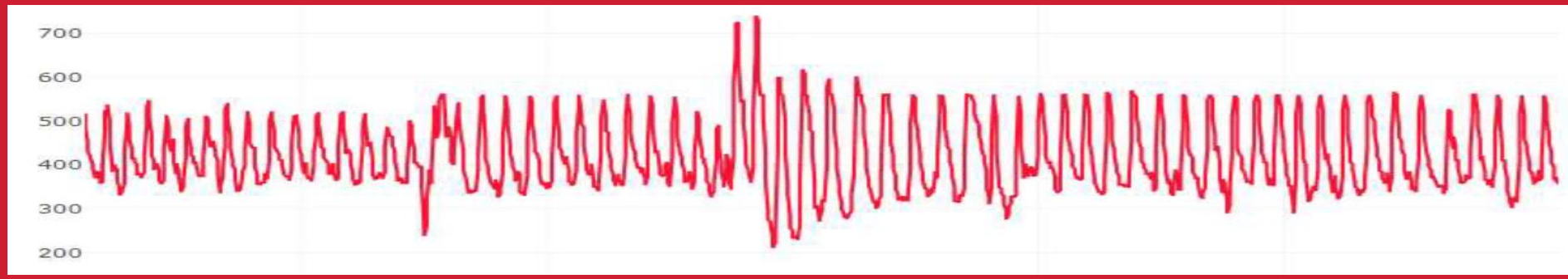

# Data storaging using MongoDB I.

Visualization of Healthcare Signals using Arduino & Node.js

HCit, INJE University

1st semester, 2018

Email : chaos21c@gmail.com



# My ID

오전

| 성명 | ID |
|---|---|
| 김민선 | HS01 |
| 김영걸 | HS02 |
| 김주란 | HS03 |
| 김주현 | HS04 |
| 김태민 | HS05 |
| 여준하 | HS06 |
| 이수민 | HS07 |
| 정민지 | HS08 |
| 정유현 | HS09 |
| 정재은 | HS10 |
| 주하영 | HS11 |
| 한준영 | HS12 |

오후

| 성명 | ID |
|---|---|
| 신영주 | HS21 |
| 오가영 | HS22 |
| 윤민수 | HS23 |
| 윤진아 | HS24 |
| 이진영 | HS25 |
| 임상은 | HS26 |
| 임재형 | HS27 |
| 최민영 | HS28 |
| 황유빈 | HS29 |

# 주간계획서

| 주간계획서 | | | |
|---|---|---|---|
| 주차 | 수업방법 | 수업내용 | 과제물 |
| 1 | 강의/실습 | 수업 및 실습 안내<br>-포터블 소프트웨어 설치 | |
| 2 | 강의/실습 | Node.js I<br>- Node.js 코드의 기본 구조<br>- 기초 Node 서버 및 클라이언트 | 실습확인 |
| 3 | 강의/실습 | Node.js II<br>- Node.js Express 서버 | 실습확인 |
| 4 | 강의/실습/발표 | Arduino I<br>- 아날로그 신호 회로<br>- LCD를 이용한 센서 신호 모니터링 | 실습확인 |
| 5 | 강의/실습 | Arduino II<br>- 단일 센서 회로와 Node.js 연결<br>- 다중 센서 회로와 Node.js 연결 | 실습확인 |
| 6 | 강의/실습 | 프로젝트1<br>- 생체 센서 회로와 Node.js 연결<br>- 생체 신호 소개 | 프로젝트1 |
| 7 | 강의/실습/발표 | IOT 데이터 시각화 I (Plotly.js)<br>- 데이터 및 시계열 차트<br>- 데이터 스트리밍 | 실습확인 |
| 8 | 시험 | 중간고사 | |
| 9 | 강의/실습 | IOT 데이터 시각화 II (Plotly.js)<br>- 다중 센서 데이터 시각화<br>- 다중 센서 데이터 스트리밍 | 실습확인 |
| 10 | 강의/실습/발표 | 프로젝트II<br>- 생체 센서 데이터 시각화<br>- 생체 센서 데이터 스트리밍 | 프로젝트II |
| 11 | 강의/실습 | IOT 데이터 저장과 처리<br>- MongoDB 설치 및 Mongo shell<br>- MongoDB와 Node.js 연결 및 데이터 저장 | 실습확인 |
| 12 | 강의/실습 | 프로젝트III<br>- MongoDB에 IOT 데이터 저장 및 모니터링<br>- 생체 센서 데이터 저장 및 시각화 | 프로젝트III |
| 13 | 강의/실습 | IOT 데이터 마이닝<br>- 아두이노에서 발생된 데이터 관리<br>- 데이터마이닝 소개 | 실습확인 |
| 14 | 강의/실습/발표 | 프로젝트IV<br>- 생체 센서 데이터 관리<br>- 생체 센서 데이터 마이닝 | 프로젝트IV |
| 15 | 시험 | 기말고사 | |

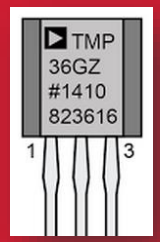
# Purpose of HS

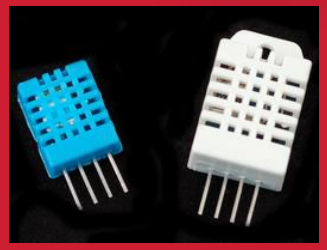
주요 수업 목표는 다음과 같다.

1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리
4. 생체 센서 발생 신호 처리, 시각화 및 저장
5. 생체 센서 발생 신호 저장 및 분석
6. 생체 신호 장비 활용 능력



# [Review]



- [wk11]
  - RT Data Visualization with node.js
  - Multi-sensor circuits
  - Complete your real-time WEB clients
  - Upload file name : HSnn_Rpt09.zip

# [wk11] Practice-09 HSnn_Rpt09.zip

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

제출파일명 : **HSnn_Rpt09.zip**

- 압축할 파일들

① **HSnn_DS_cds_tmp36.png**

② **HSnn_cds_dht22_data.png**

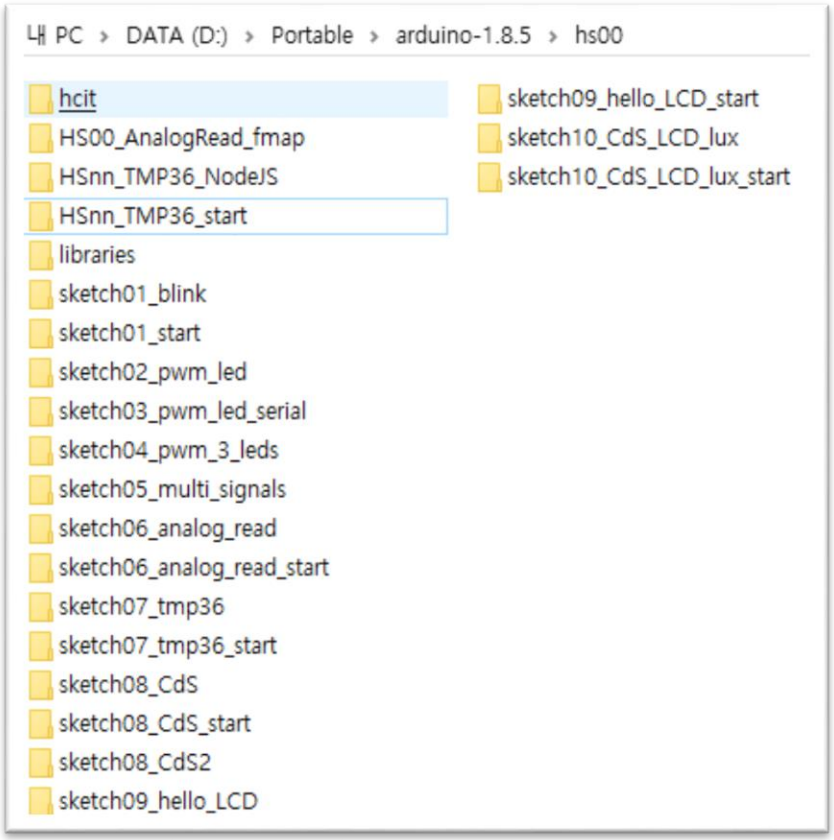
③ **HSnn_cds_dht22.html**

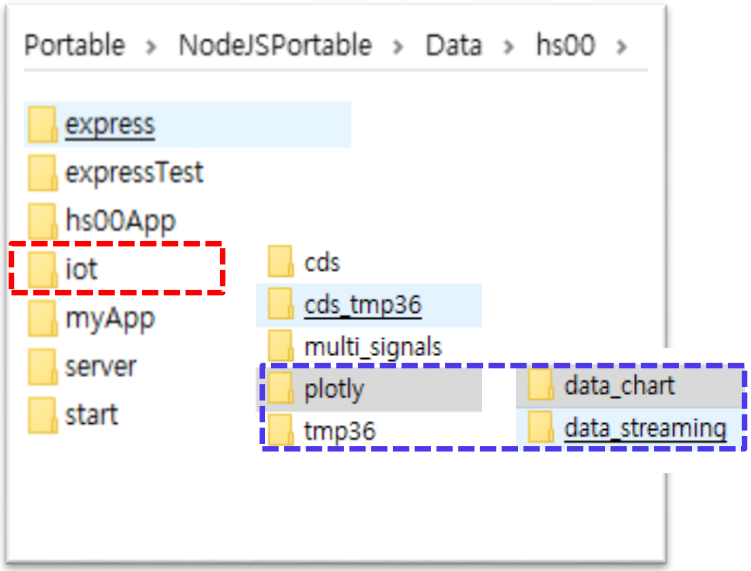
④ **HSnn_cds_dht22.png**

**Email : chaos21c@gmail.com**

**[** 제목 : **id,** 이름 (수정) **]**

# [My working folder – wk11]

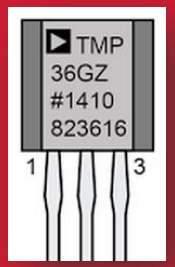


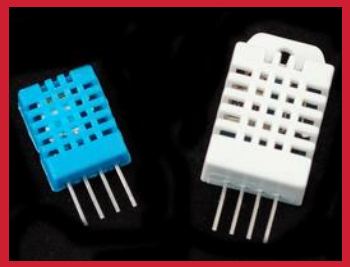



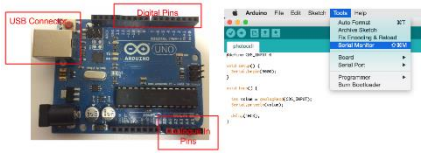
# Arduino

# + Node.js

# + plotly.js




# IOT: HSC

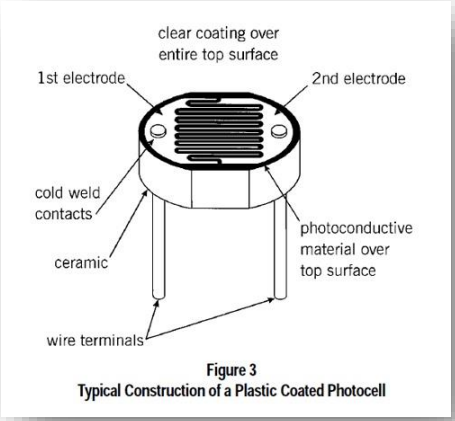
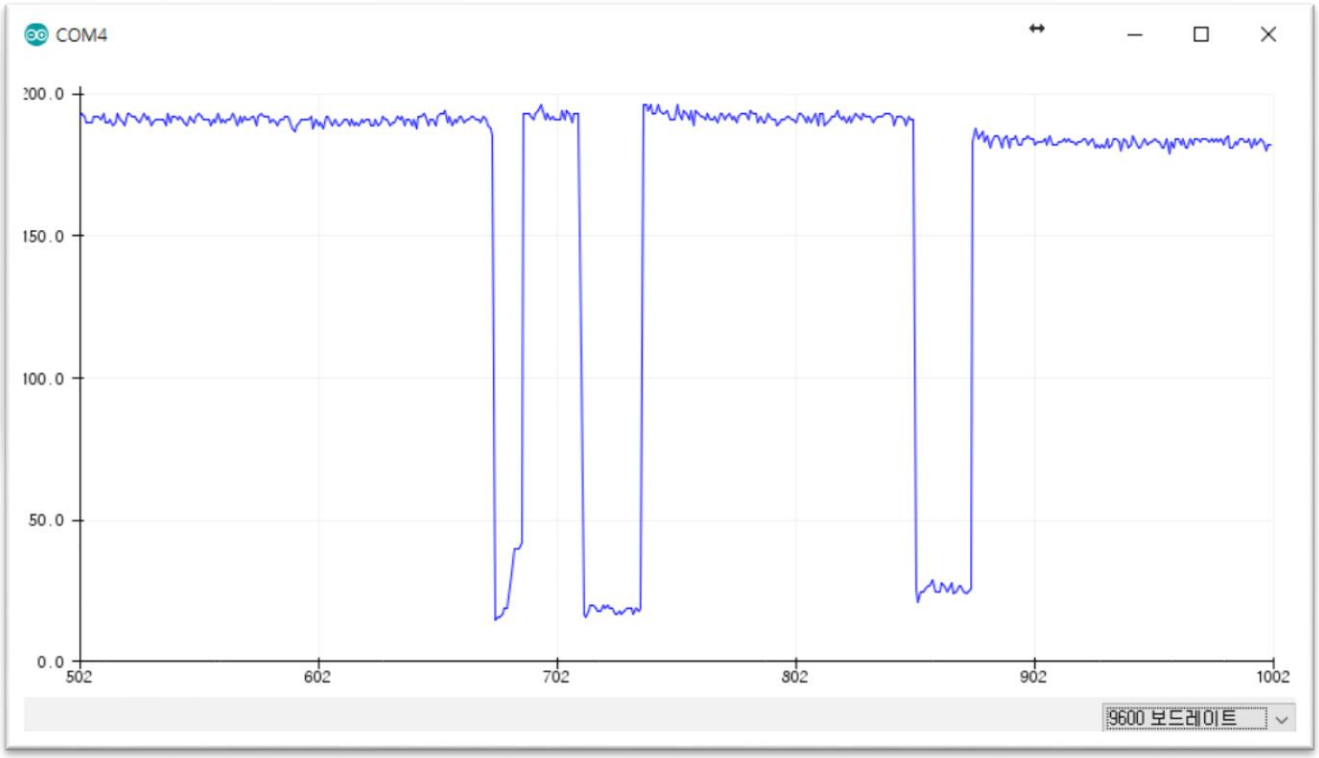
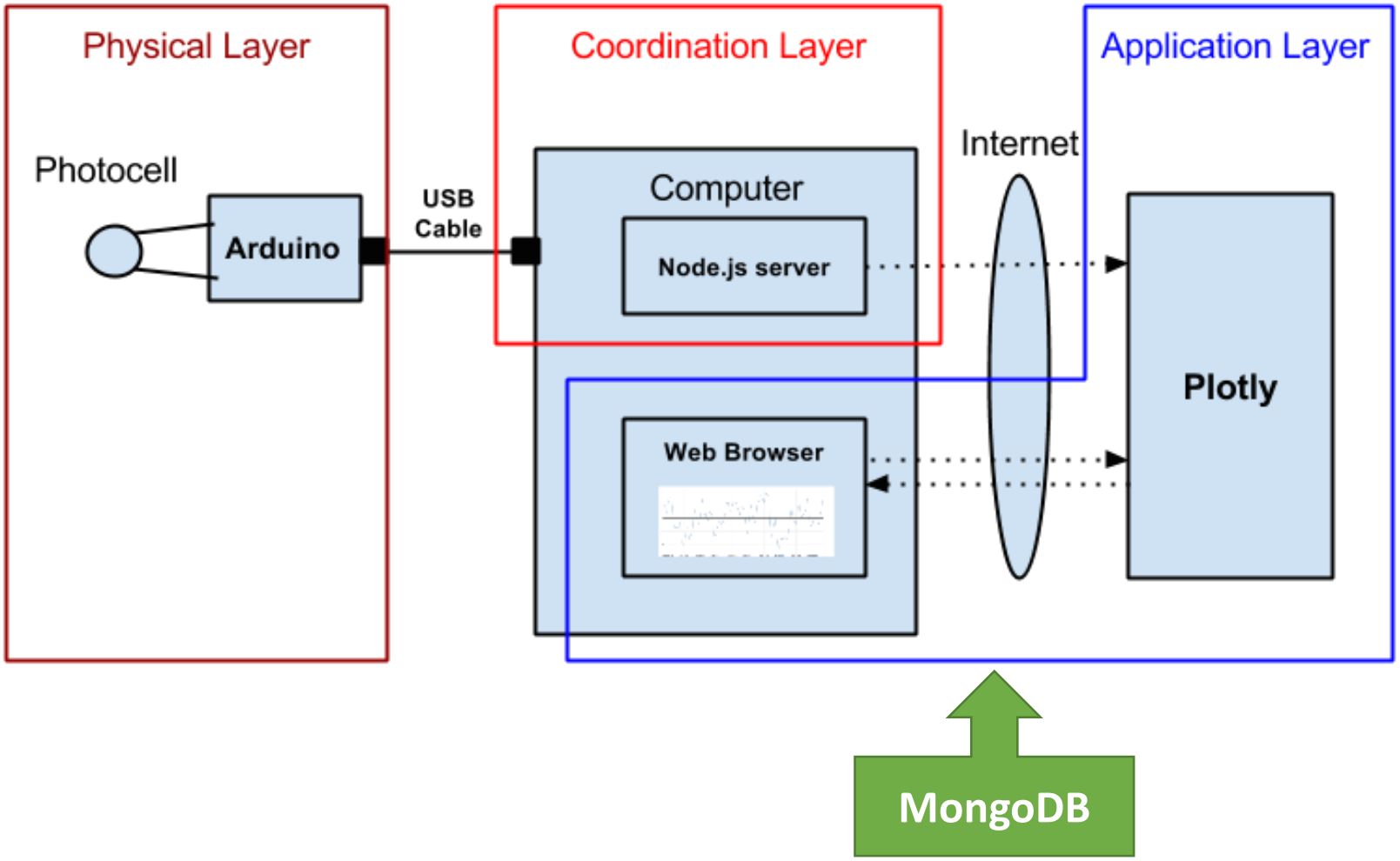


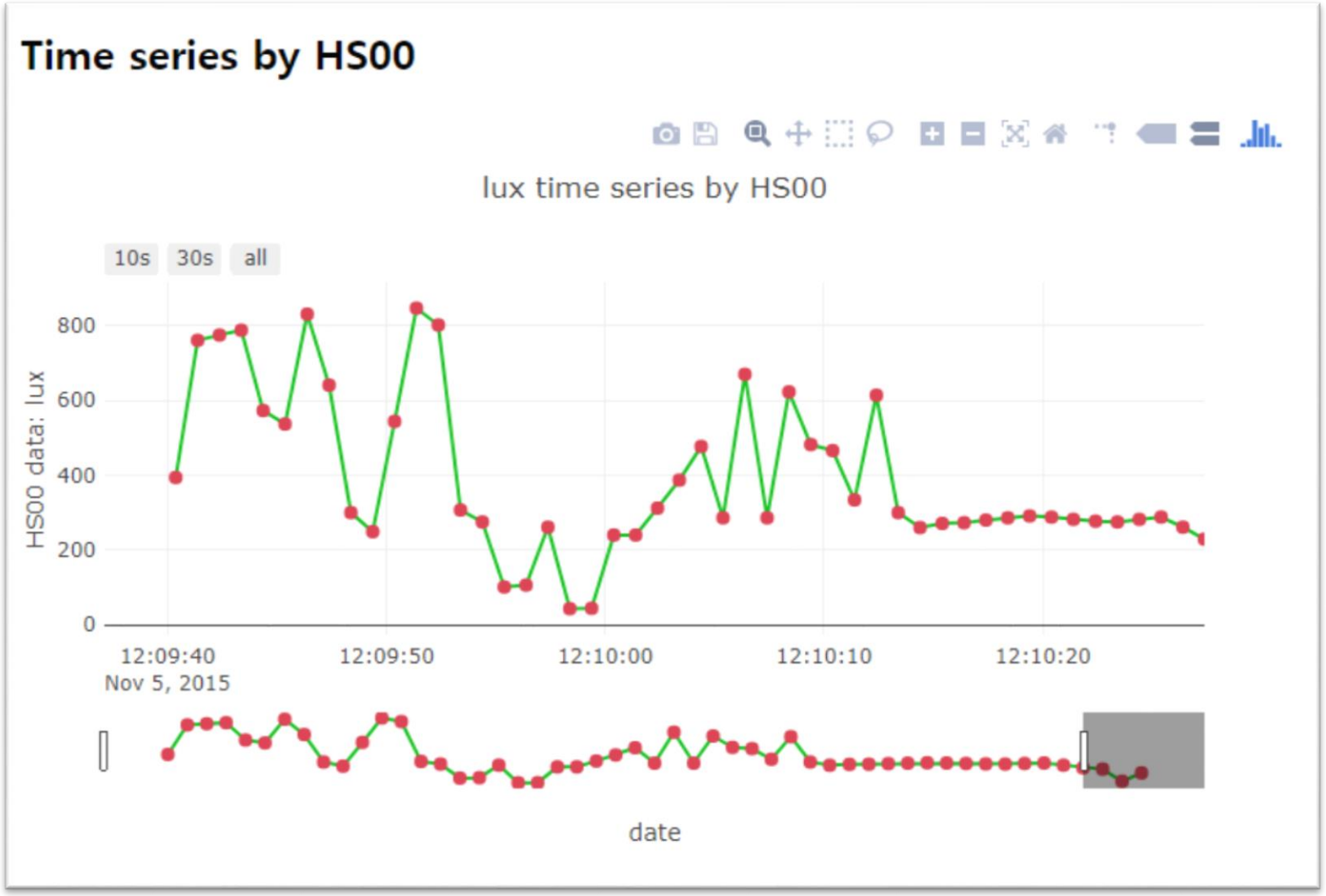
Figure 3
Typical Construction of a Plastic Coated Photocell



# Layout [H S C]

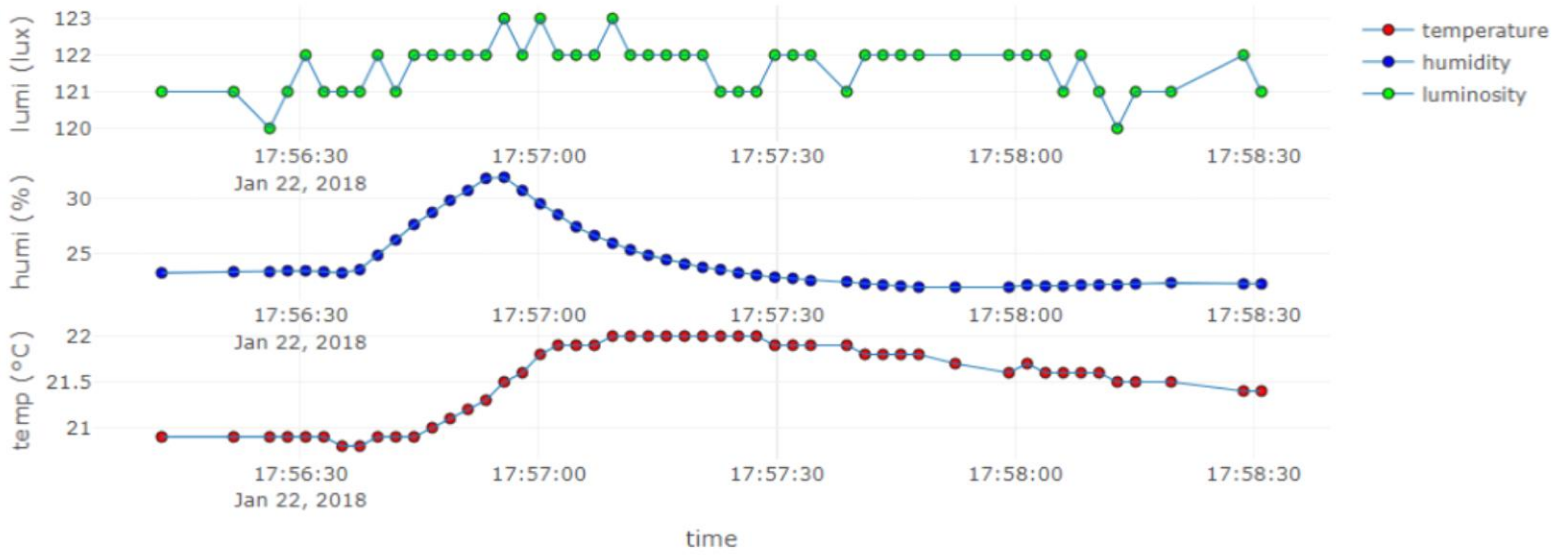

Physical Layer
Photocell
Arduino
USB Cable
Coordination Layer
Computer
Node.js server
Web Browser
Internet
Application Layer
Plotly
MongoDB

# Arduino data + plotly

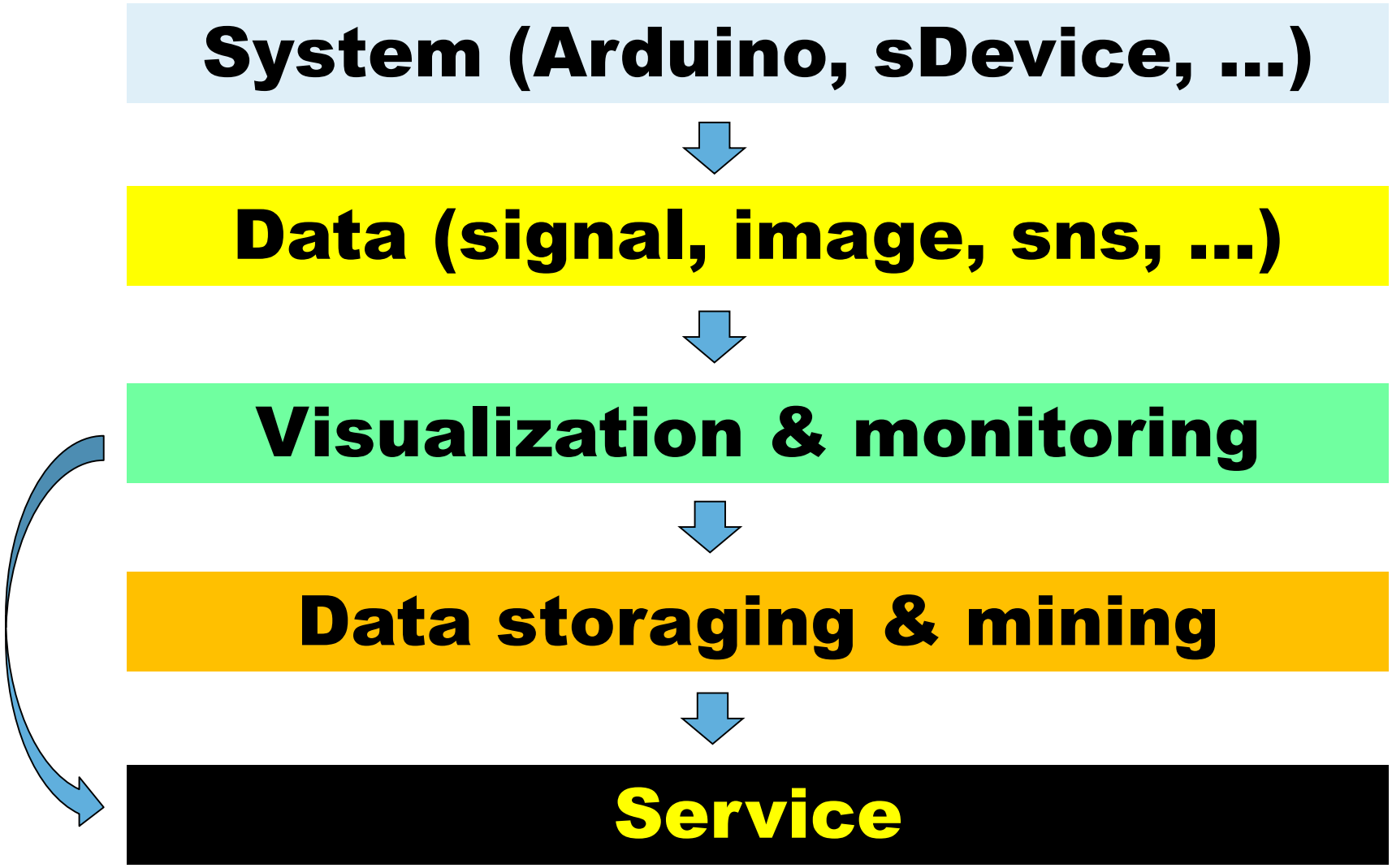


# Real-time Weather Station from sensors


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



# A5. Introduction to visualization



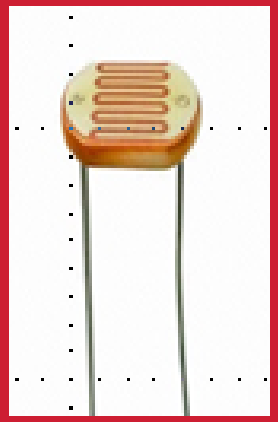
# CdS + DHT22

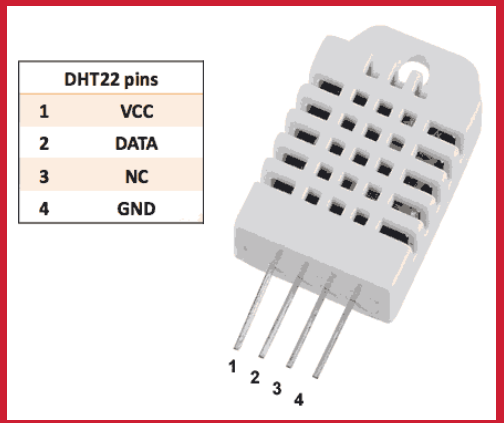
# + plotly.js

# Node project

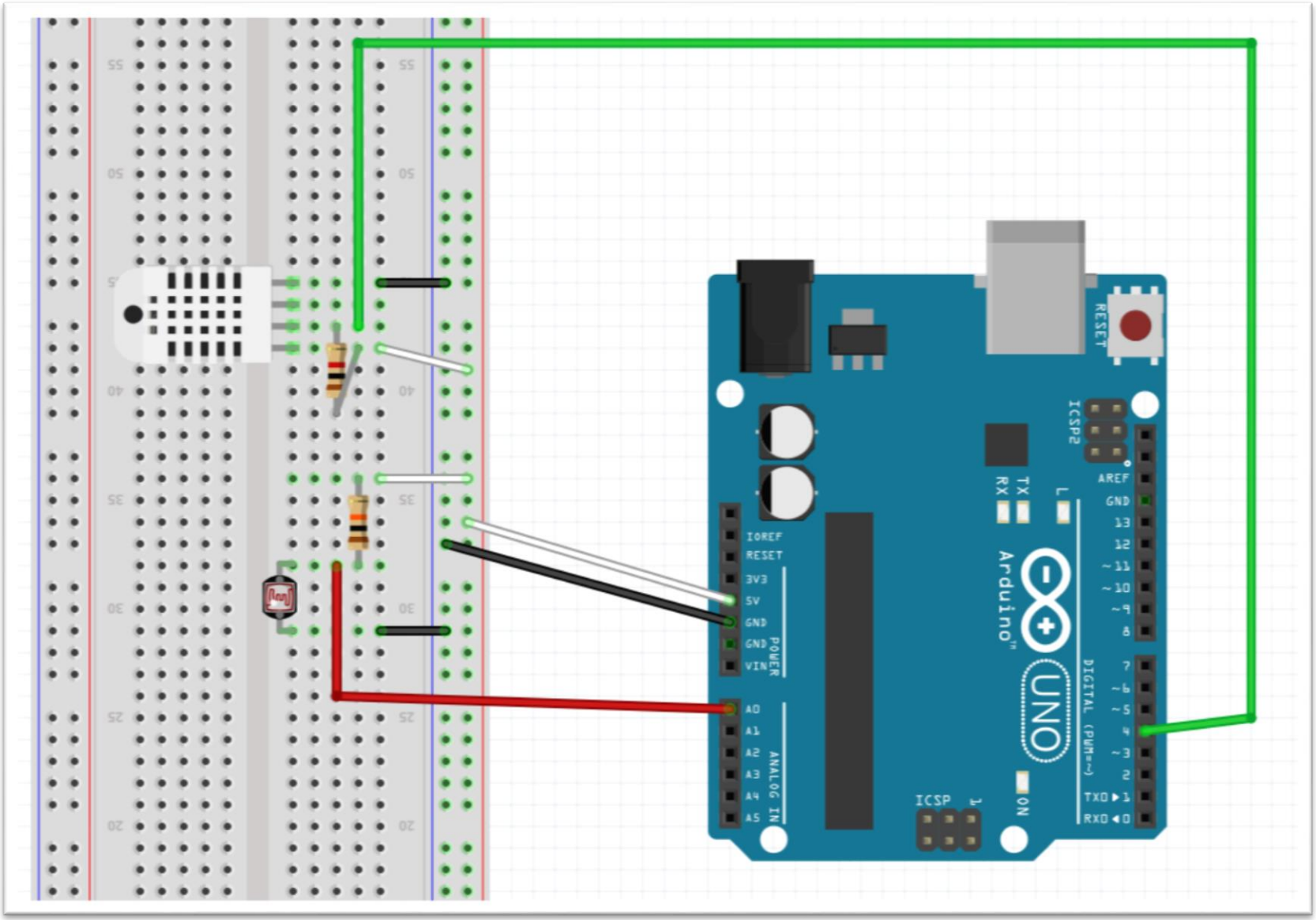

## Multi-sensors

## DHT22 + CdS



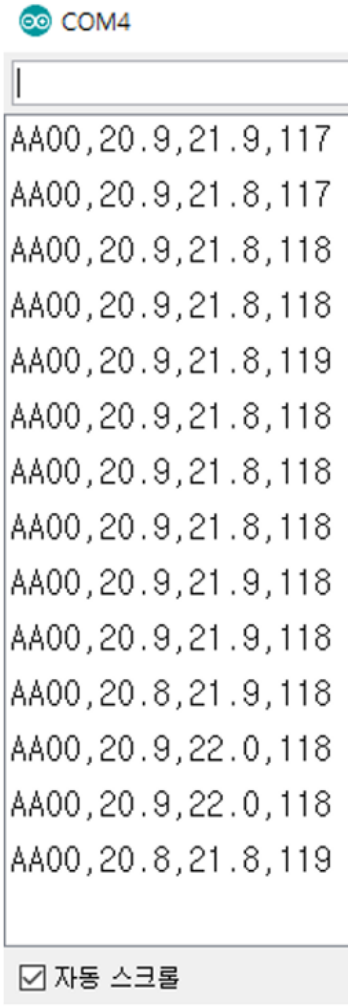


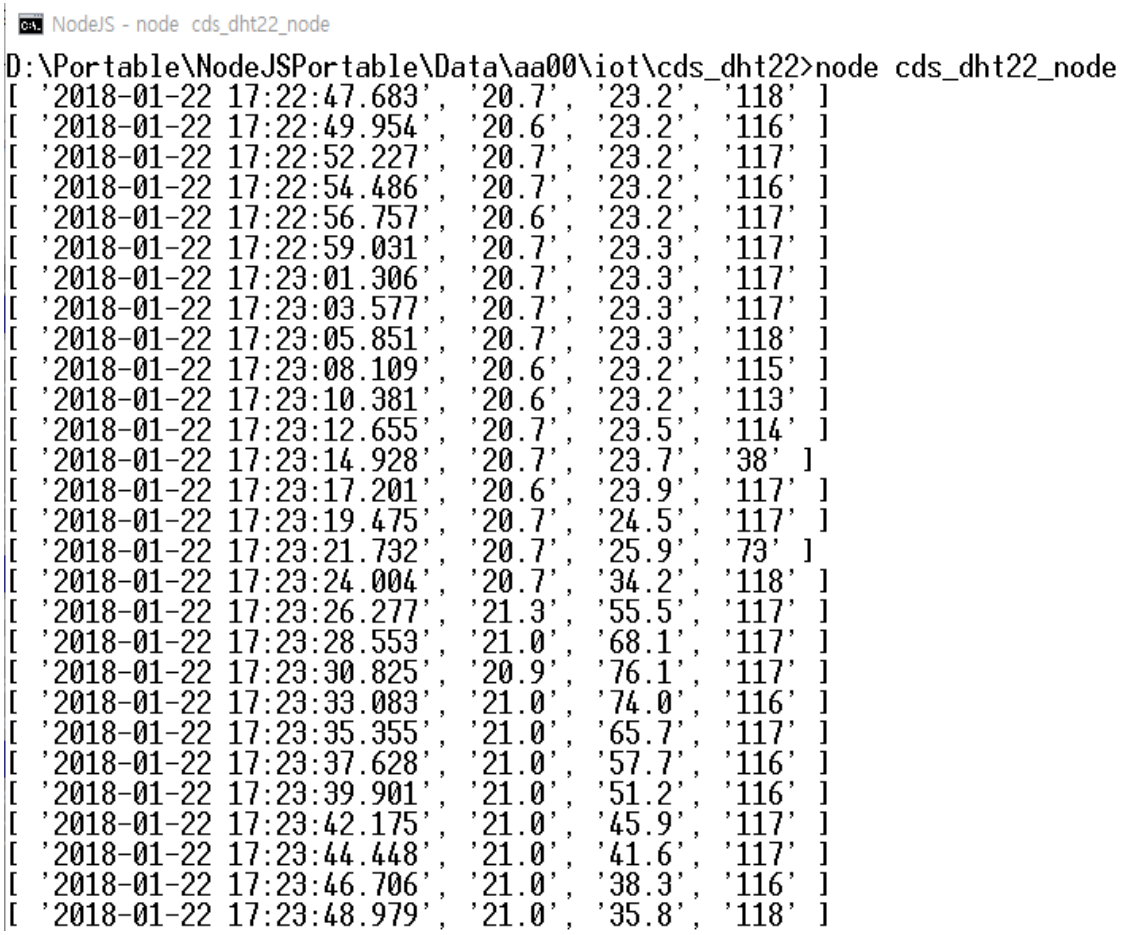
# A5.7.1 DHT22 + CdS circuit



# A5.7.10 DHT22 + CdS + Node.js

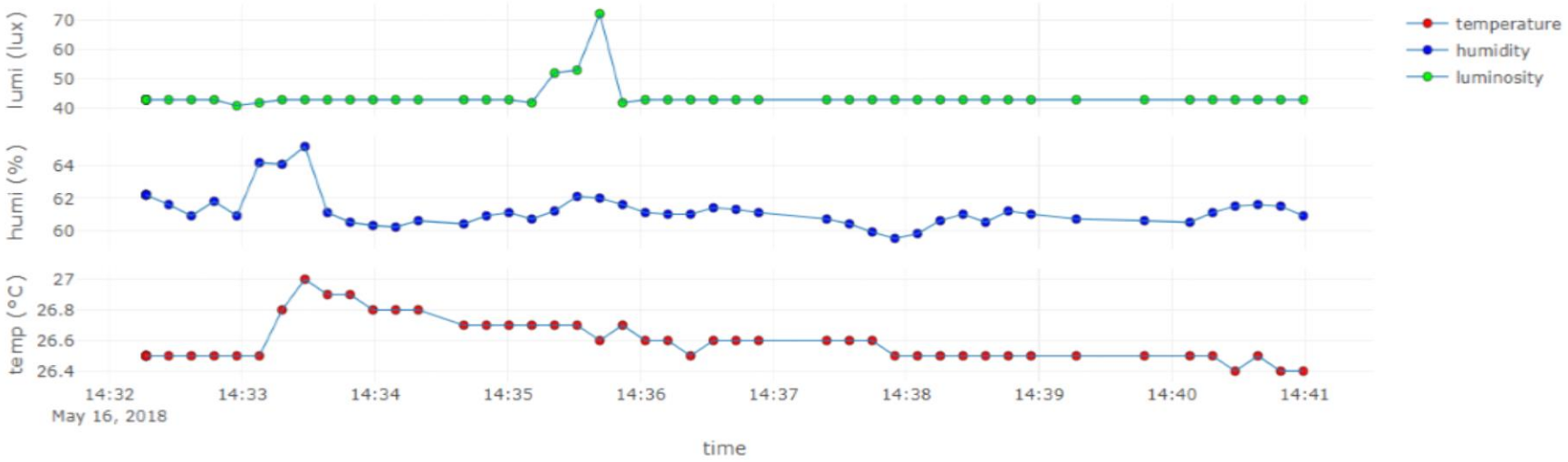
## [3] Result: Parsed streaming data from dht22 & CdS (Run in Node cmd)



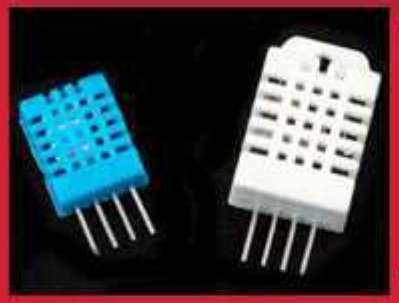



# Real-time Weather Station from sensors


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








# [Goal]

## Arduino + Node.js

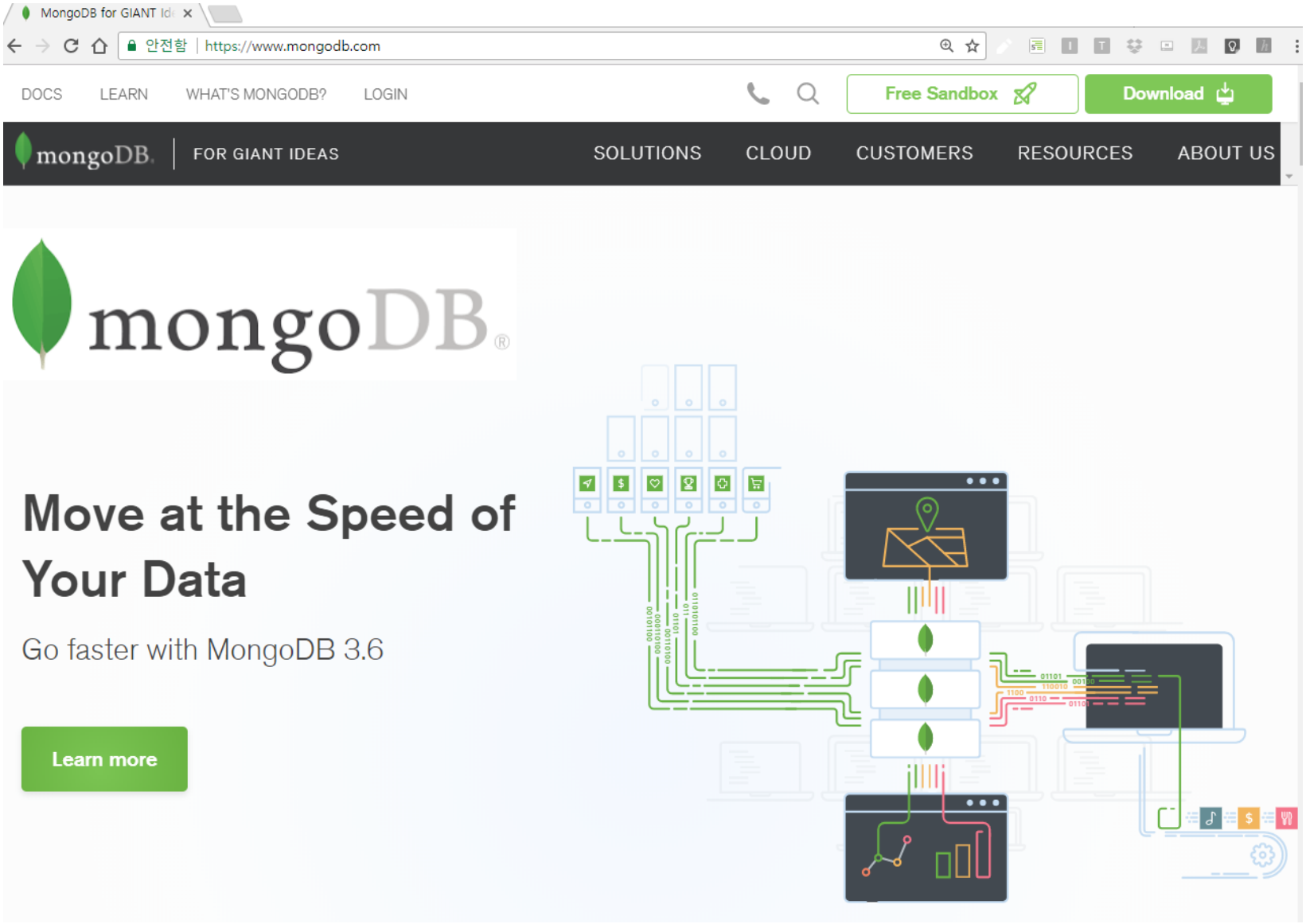
## + plotly.js

## + MongoDB

## → Data storaging

## & visualization

# A5.9 MongoDB


MongoDB for GIANT Id
안전함 | 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 데이터저장소 입니다. 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" } }
```

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

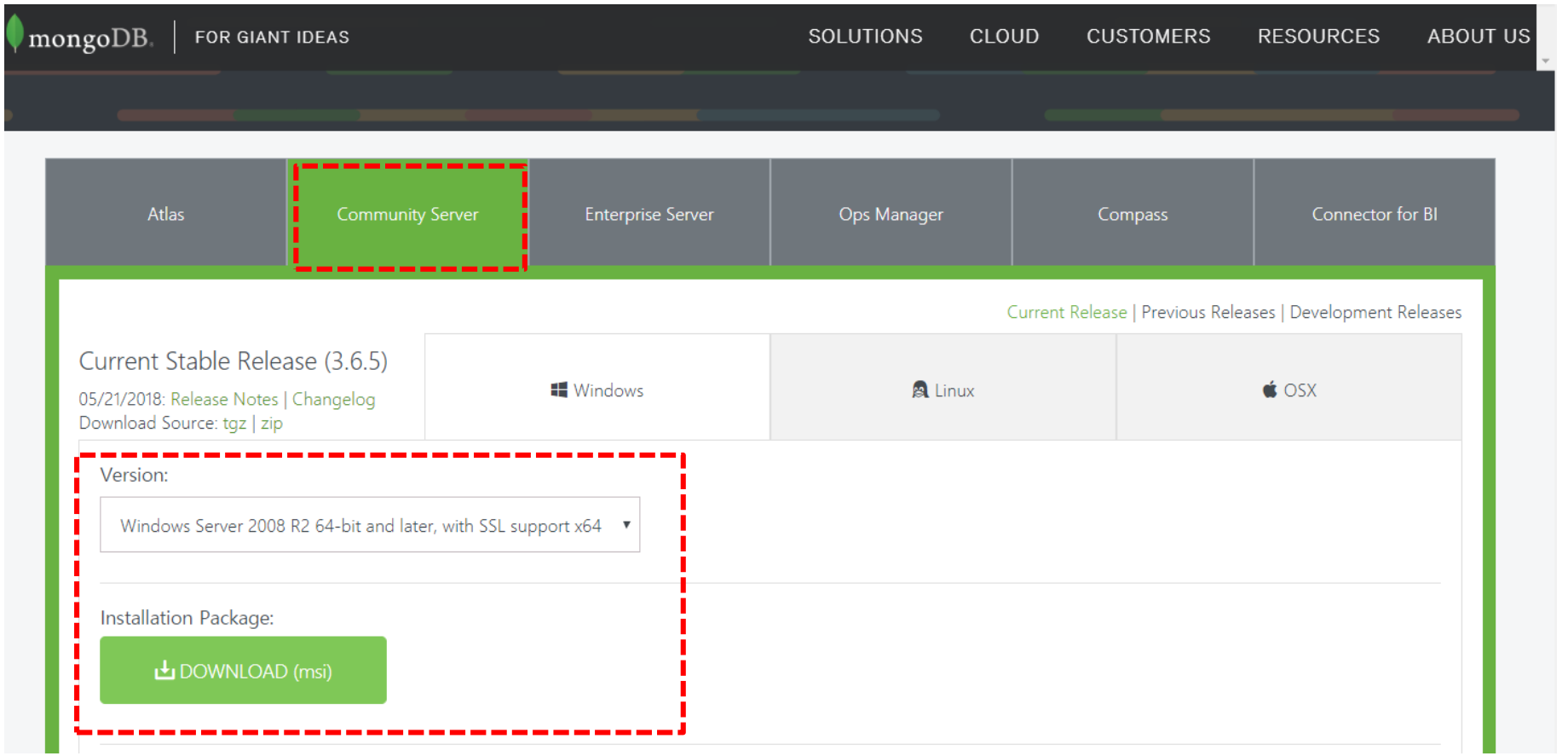
## Database?

Database는 Collection들의 물리적인 컨테이너입니다. 각 Database는 파일시스템에 여러파일들로 저장됩니다.

# A5.9 MongoDB

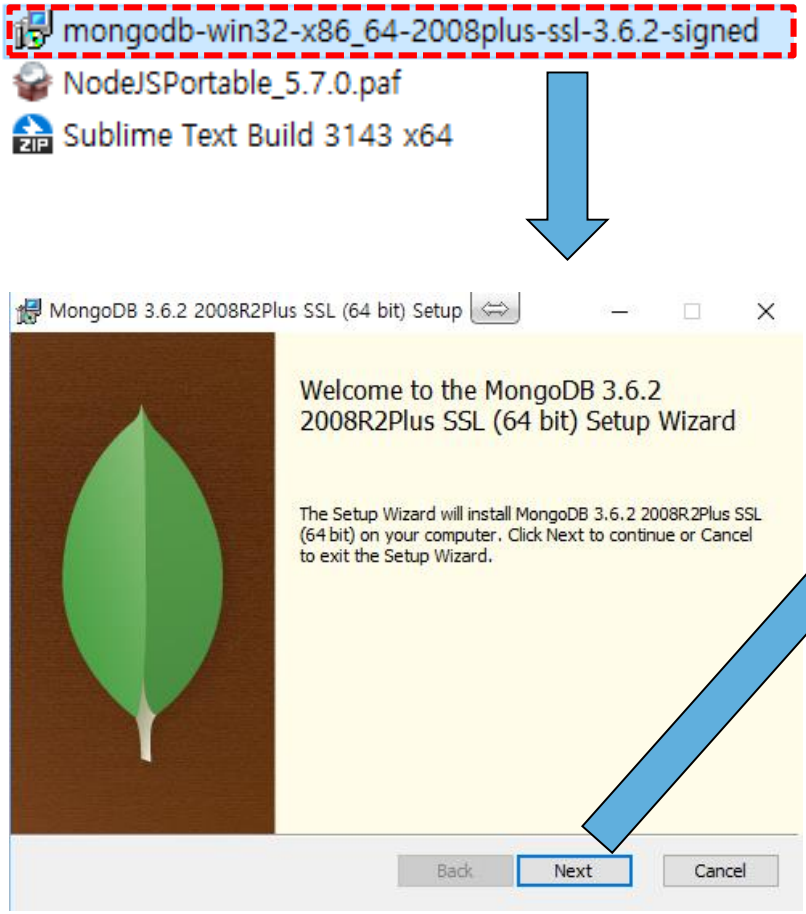

https://www.mongodb.com/download-center#community

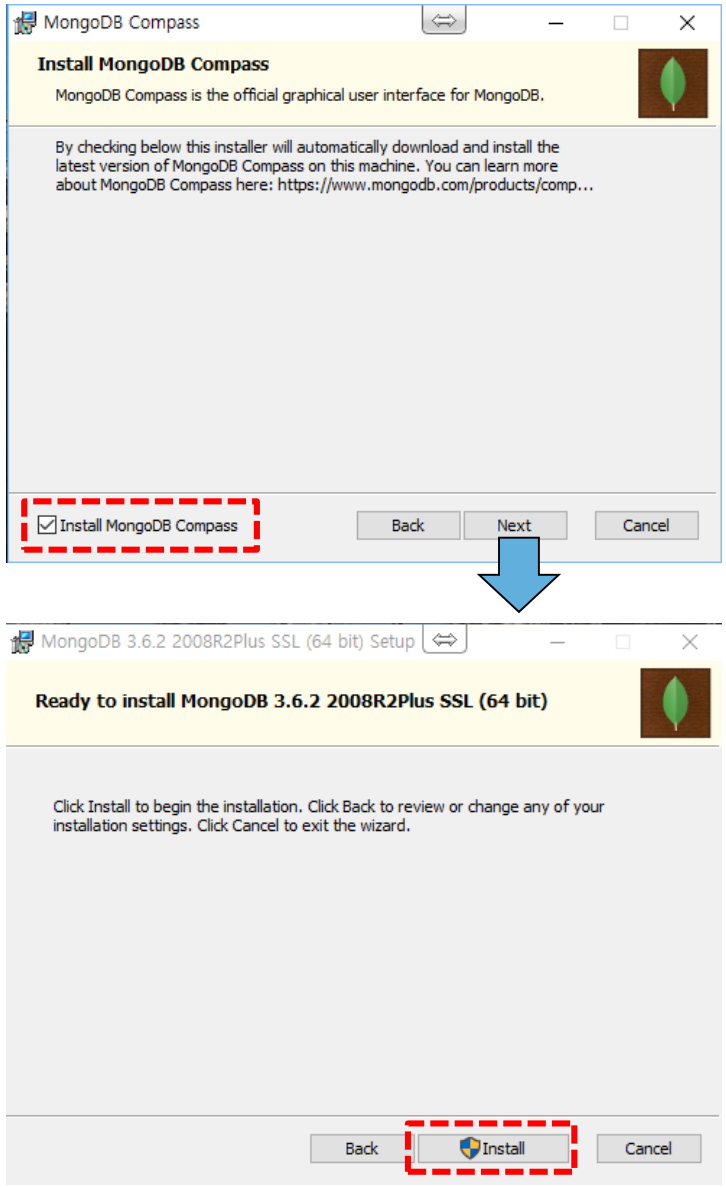
# A5.9 MongoDB



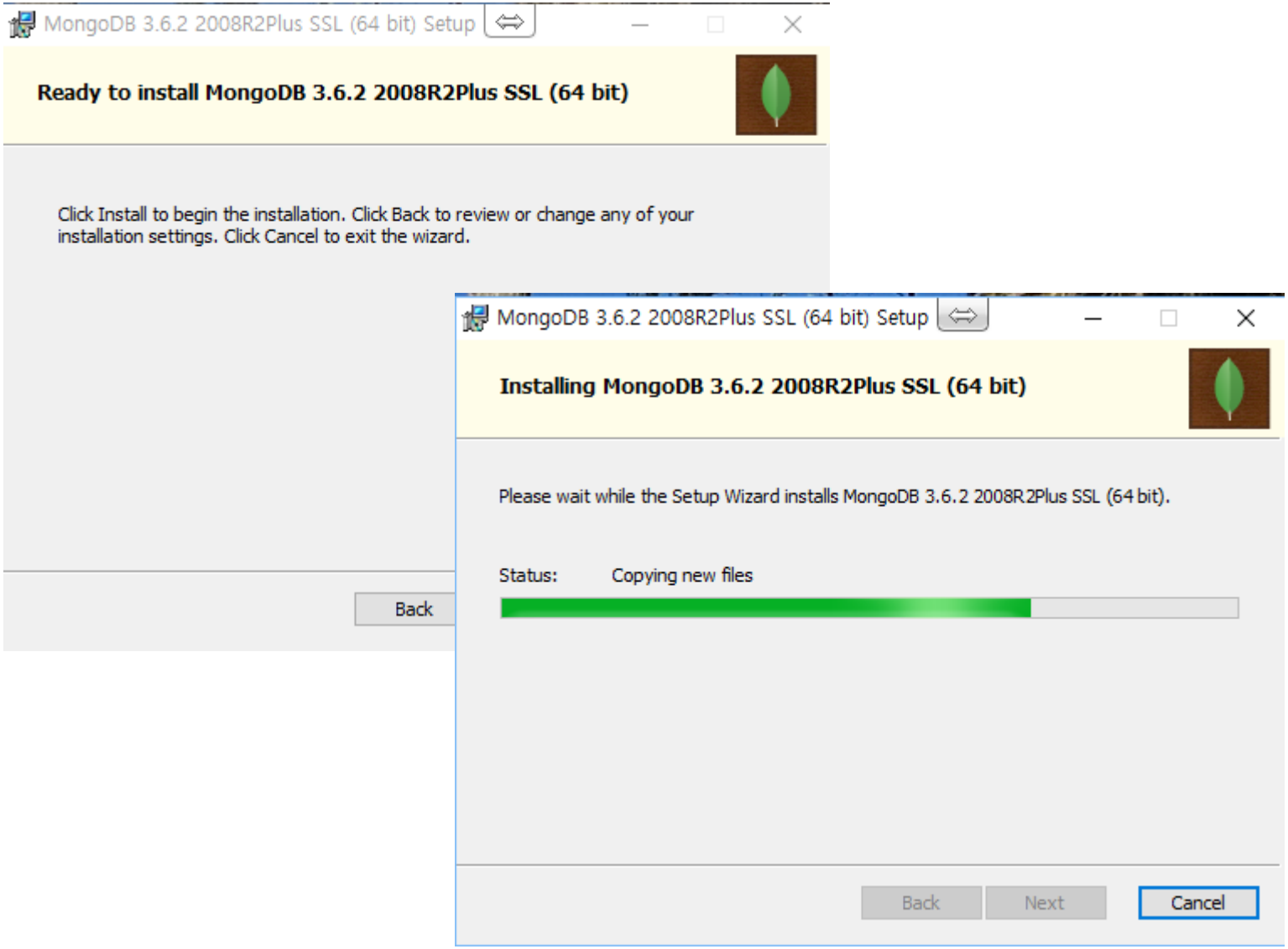

https://www.mongodb.com/download-center#community

# A5.9.1 MongoDB install - 1



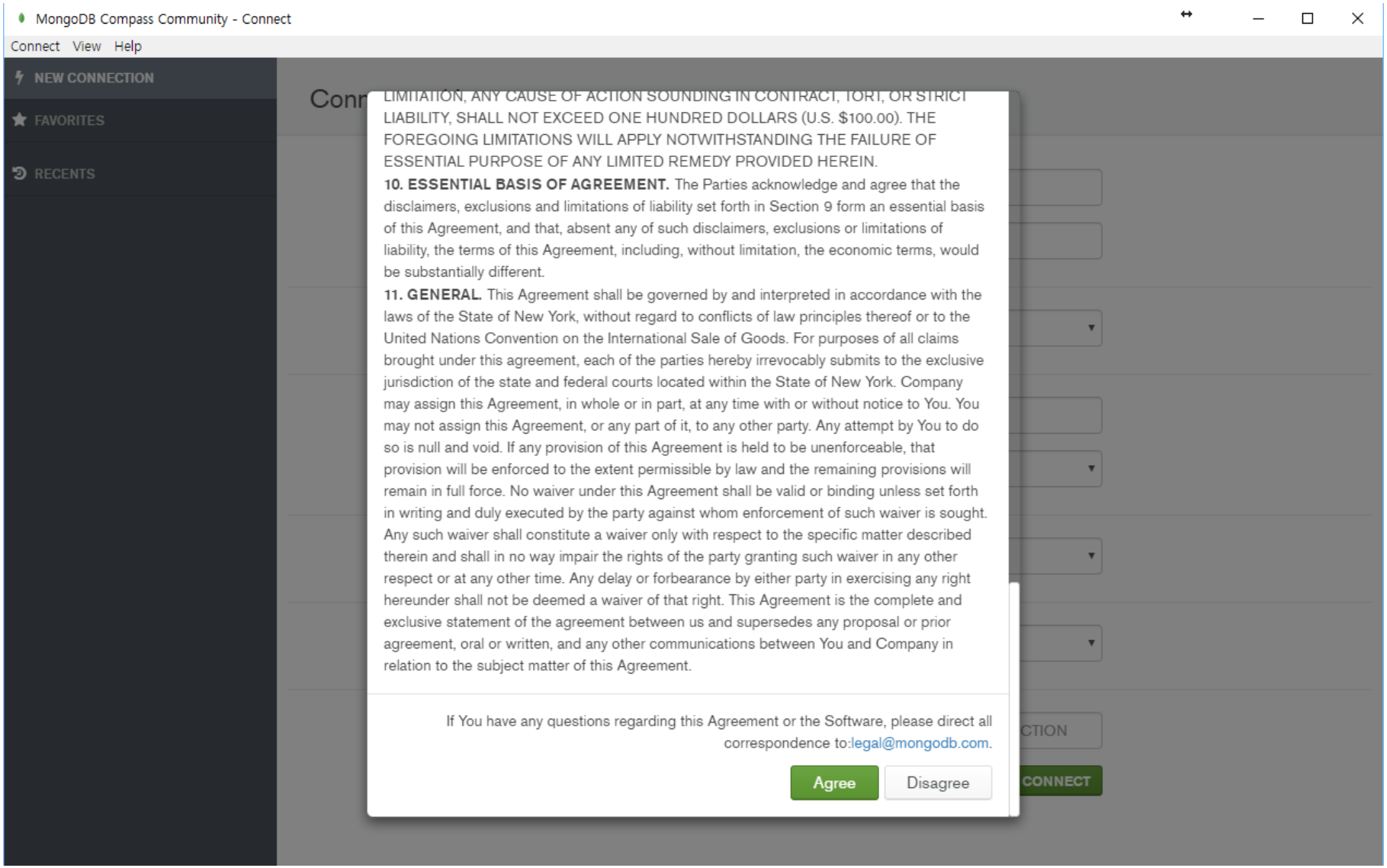



# A5.9.1 MongoDB install - 2



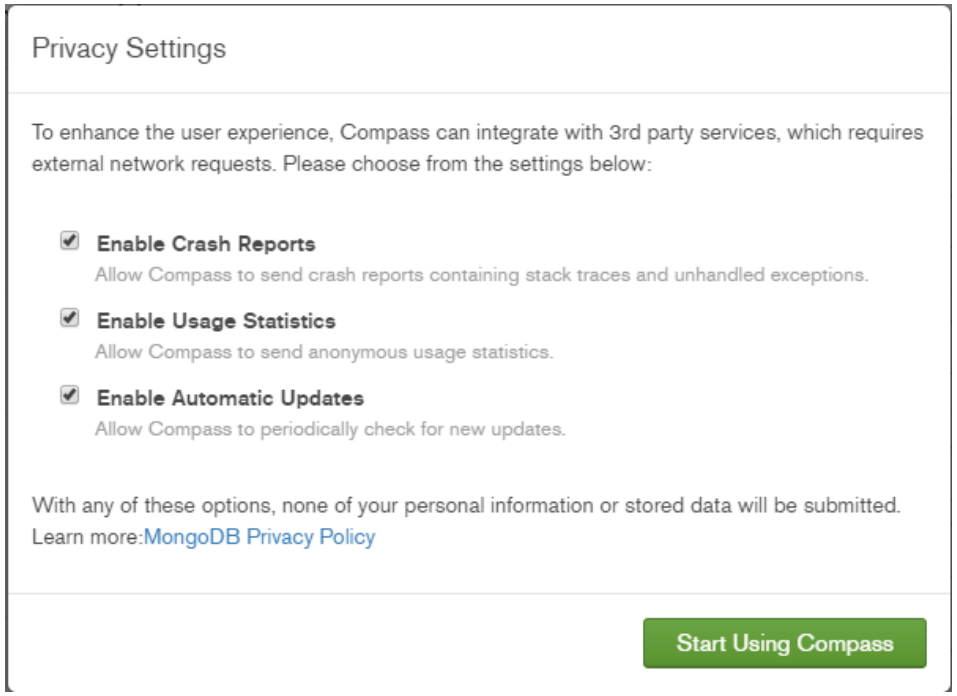

https://www.mongodb.com/download-center#community

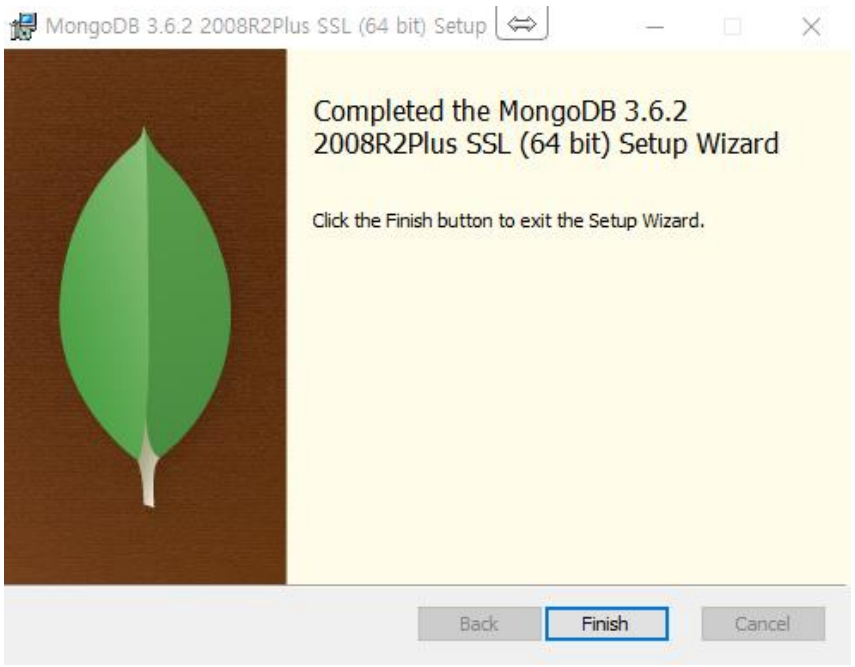
# A5.9.1 MongoDB install – 3




https://www.mongodb.com/download-center#community

# A5.9.1 MongoDB install – 4




https://www.mongodb.com/download-center#community

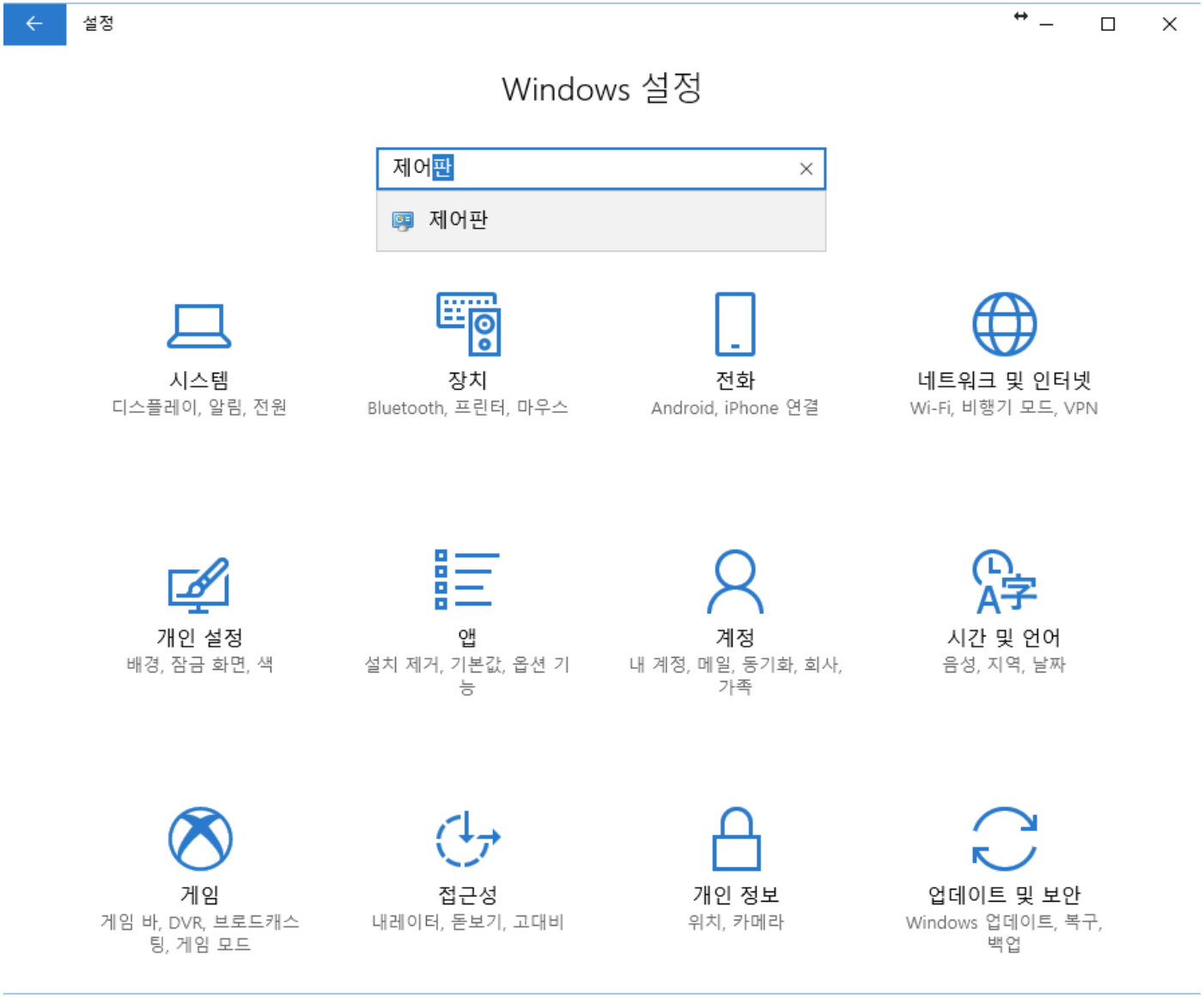
# A5.9.1 MongoDB install – 5.


윈도우10: 설정 > 시스템 > 정보

[중요] 시스템 환경변수 : **PATH** 에 경로 추가

**C:\Program Files\MongoDB\Server\3.6\bin**

# A5.9.1 MongoDB install – 6.



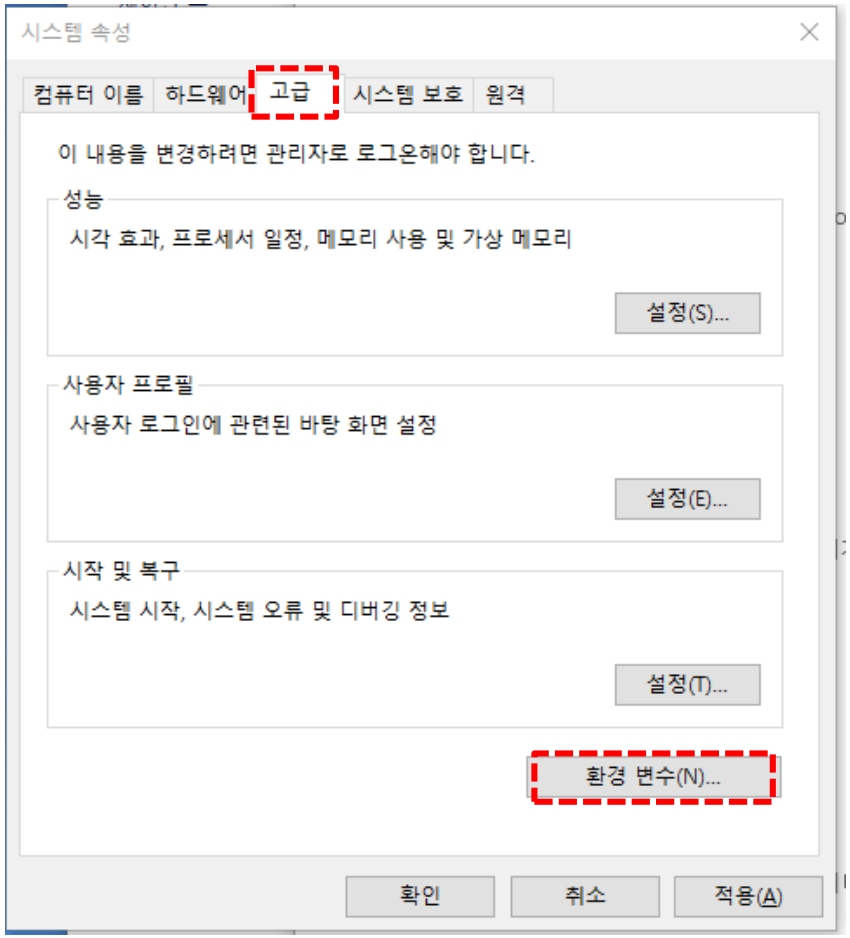
# A5.9.1 MongoDB install – 7.

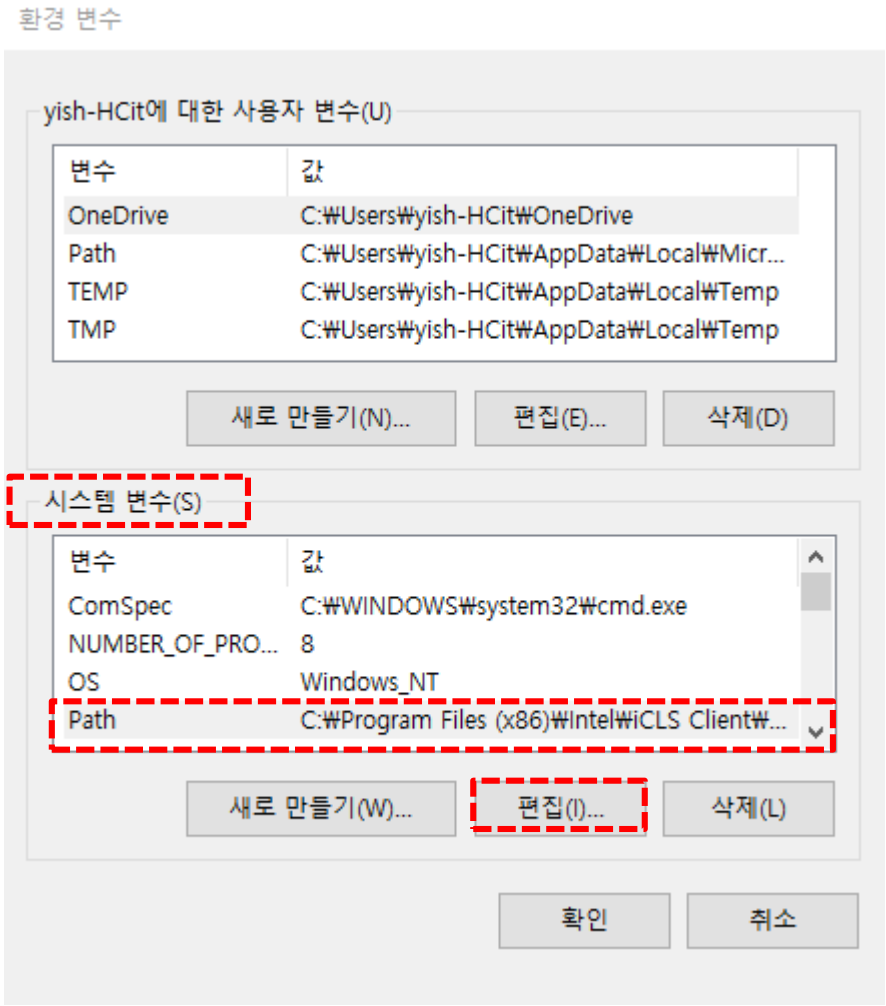
윈도우10: 설정 > '제어판' 검색 > 모든 제어판 항목에서 '시스템' 선택

> 고급 시스템 설정

# A5.9.1 MongoDB install – 8.

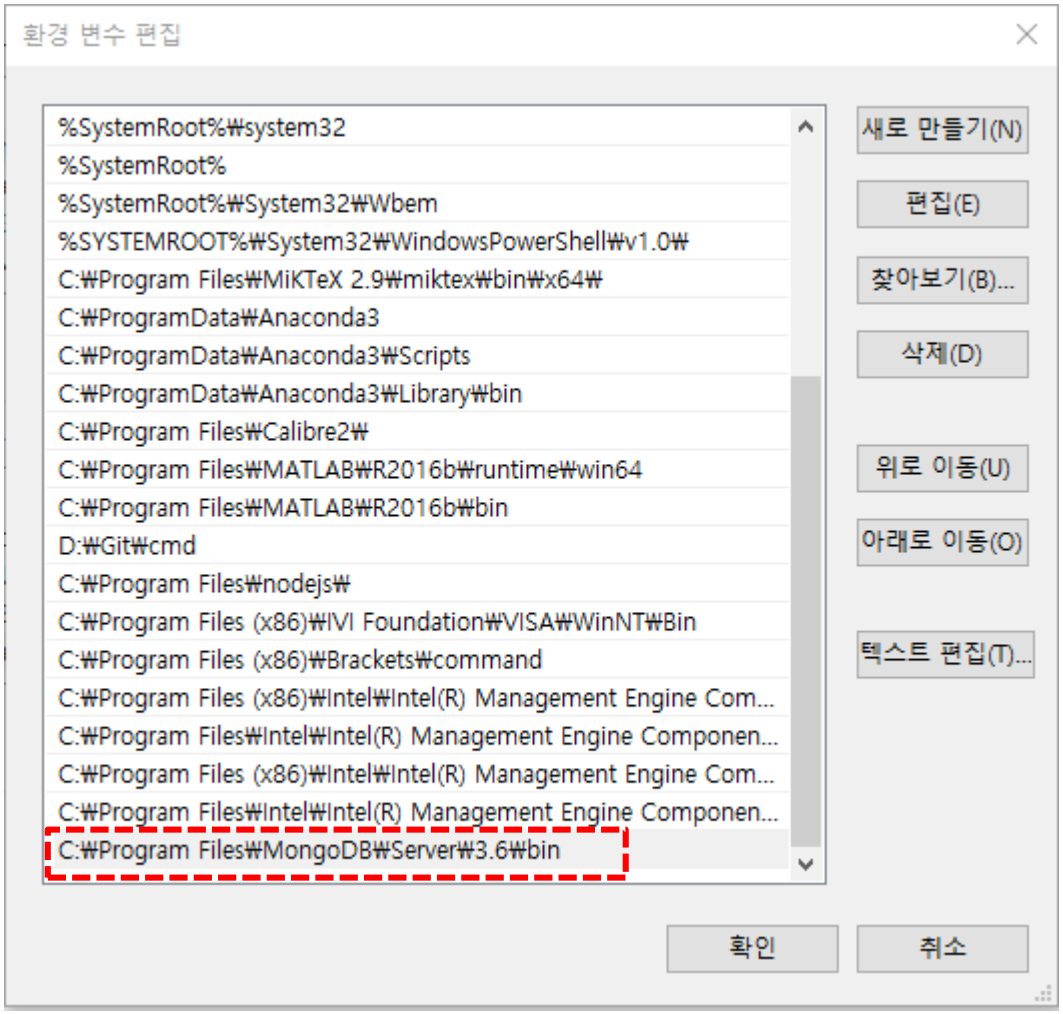
## 환경 변수 설정



# A5.9.1 MongoDB install – 9.

## 환경 변수 추가



# A5.9.2 MongoDB shell - 1

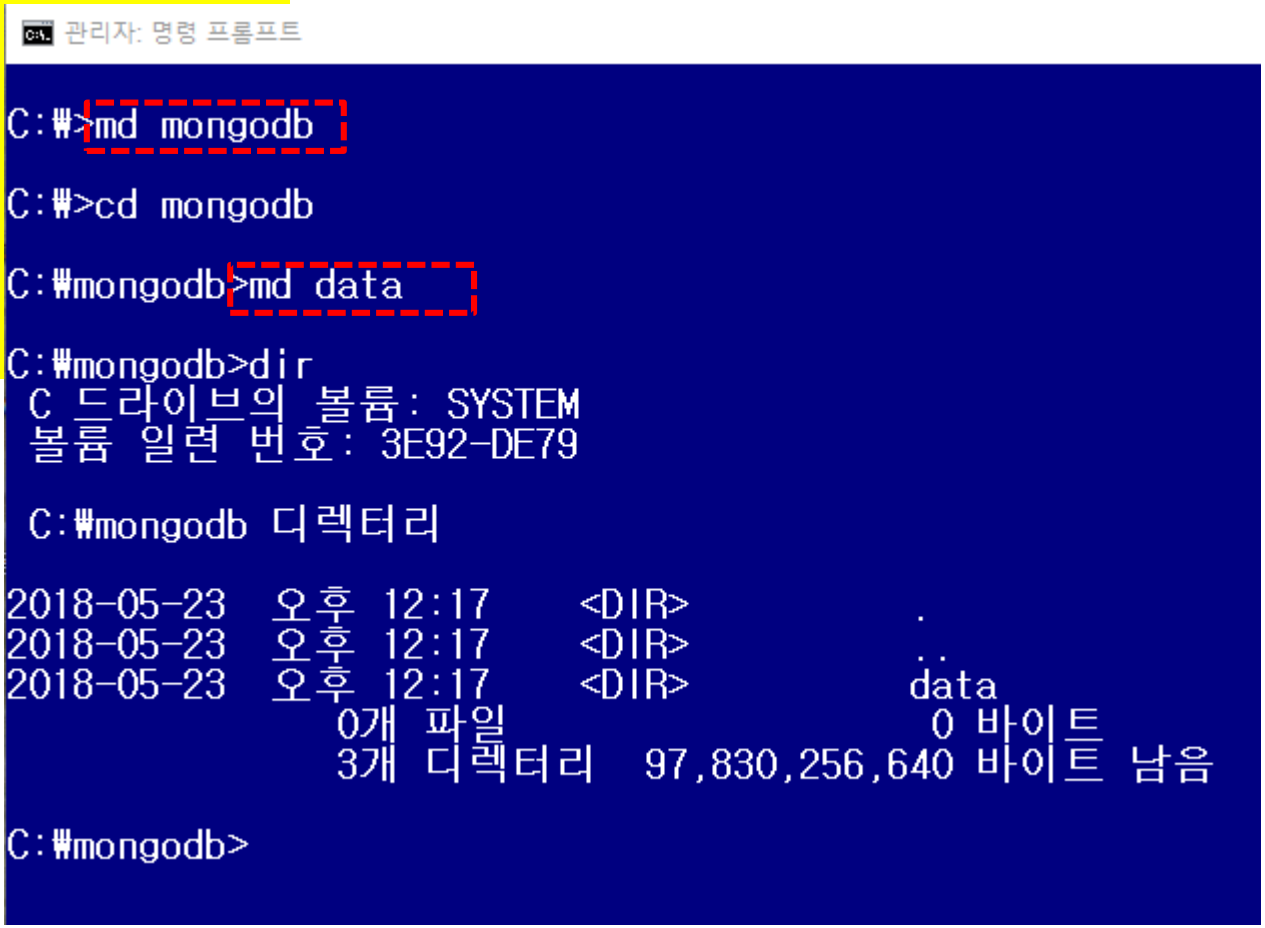
## 1. Mongo shell 실행

## > mongo

# A5.9.2 MongoDB shell - 2

## 2. MongoDB 저장소 만들기

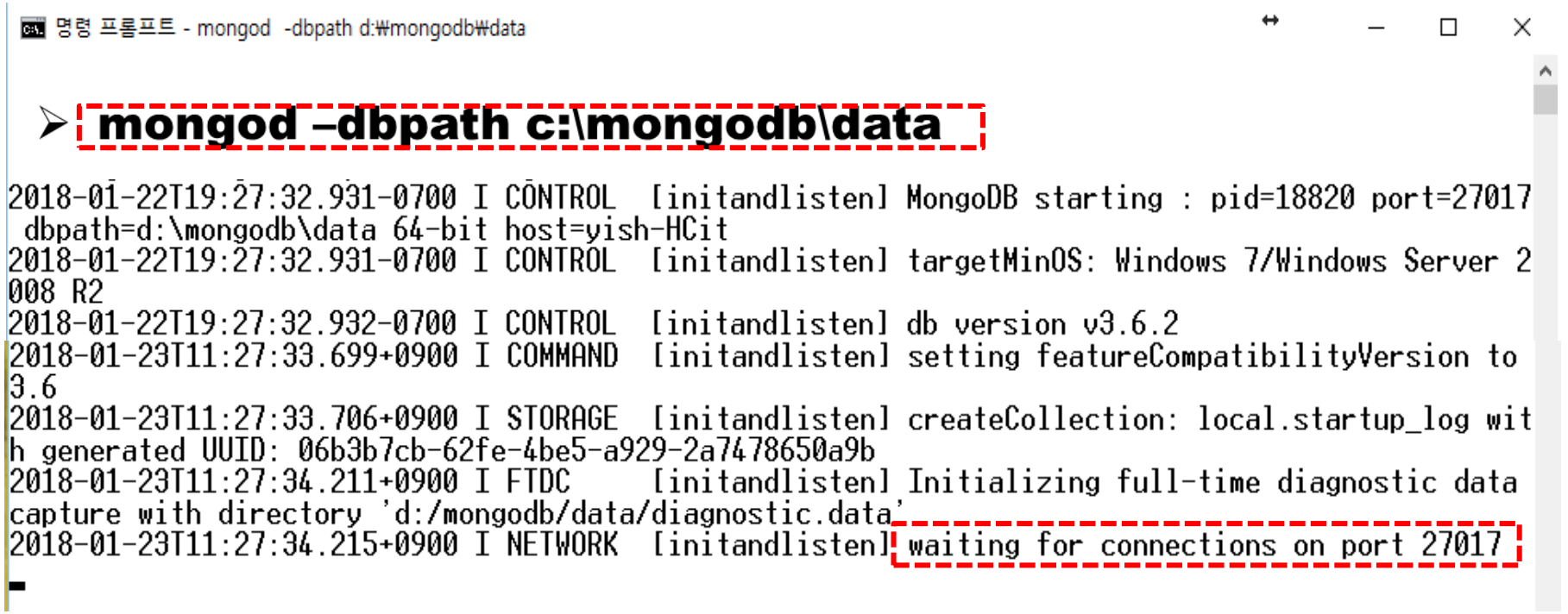
- md mongodb
- cd mongodb
- dir
- md data
- dir



# A5.9.2 MongoDB shell - 3

## 3. Run MongoDB by using mongod.exe

- **mongod –dbpath c:\mongodb\data**




사용 PC 환경에 맞게 실행 (특히, 경로 지정)

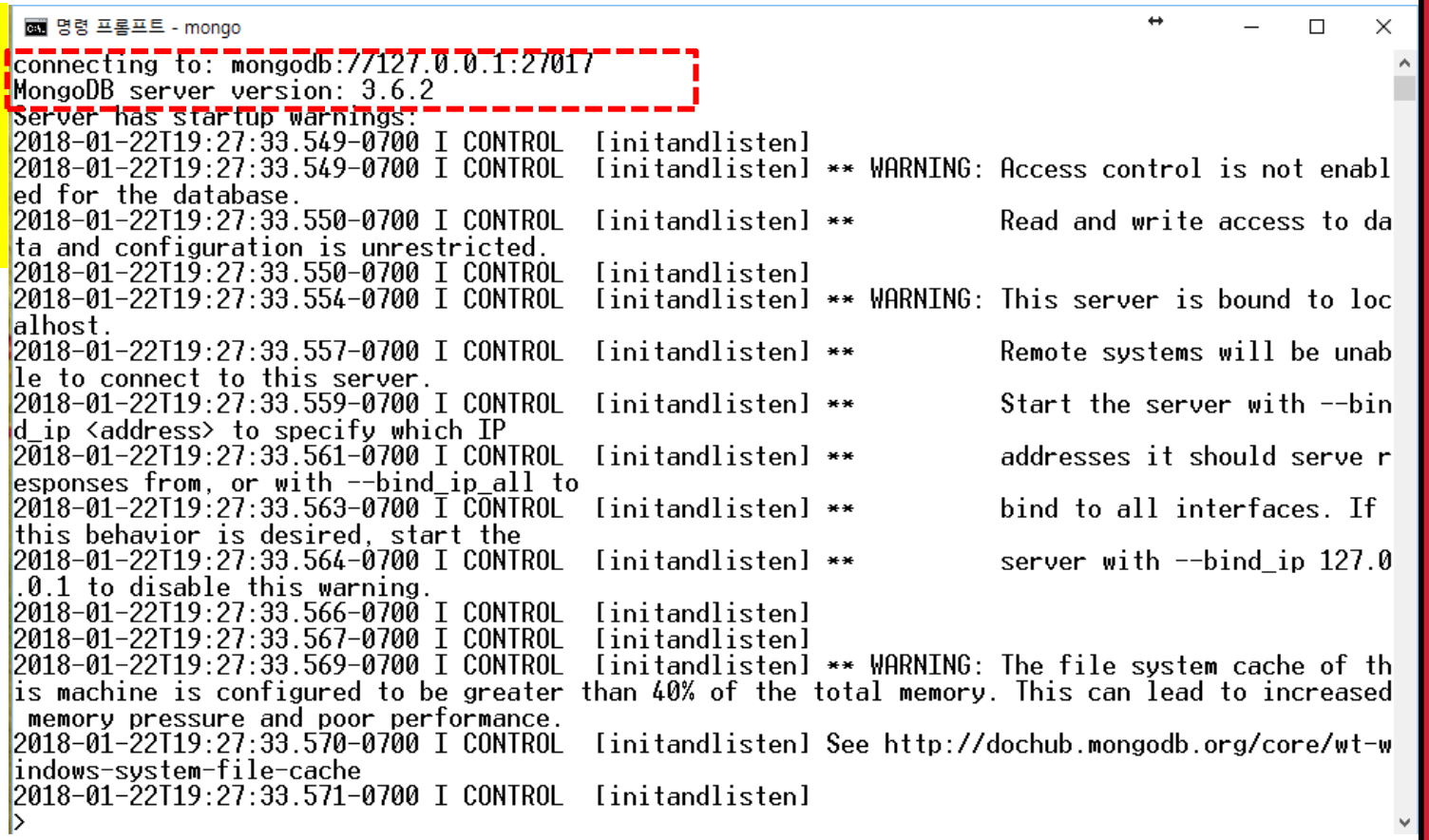
# A5.9.2 MongoDB shell – 4

## 4. Run mongo shell : mongo.exe [use new cmd]

- **mongo**

**Run new cmd**

**mongo**



# A5.9.2 MongoDB shell - 5

## 5. mongo shell :

Run new cmd

mongo

**show dbs**

**use local**

**show collections**

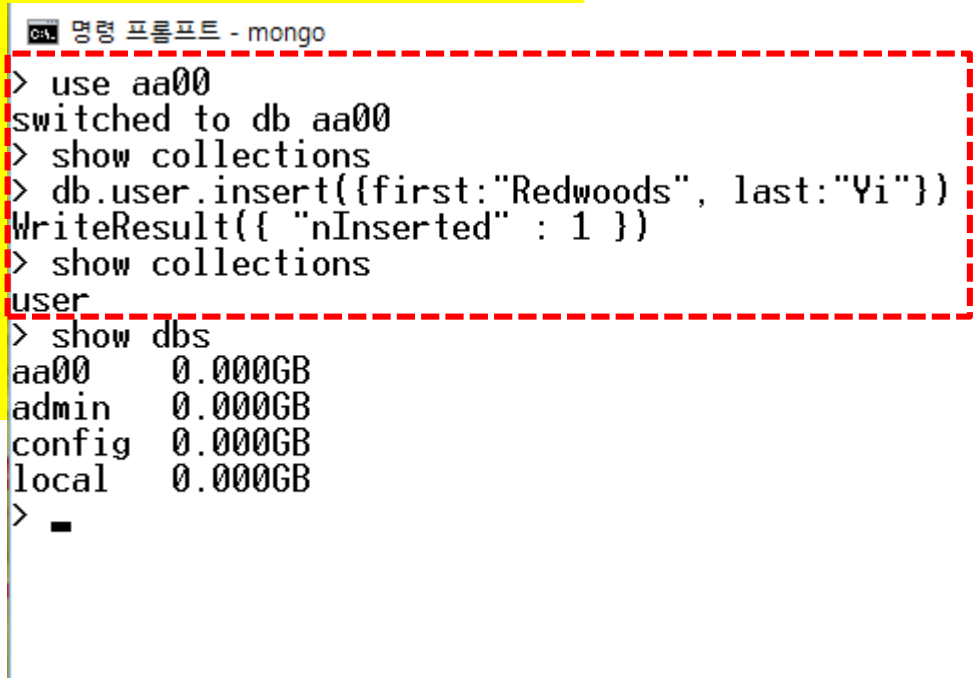
**help**

```
명령 프롬프트 - mongo
> show dbs
admin  0.000GB
local  0.000GB
> use local
switched to db local
> show collections
startup_log
> help
        db.help()                    help on db methods
        db.mycoll.help()             help on collection methods
        sh.help()                    sharding helpers
        rs.help()                    replica set helpers
        help admin                   administrative help
        help connect                 connecting to a db help
        help keys                    key shortcuts
        help misc                    misc things to know
        help mr                      mapreduce

        show dbs                     show database names
        show collections             show collections in current database
        show users                   show users in current database
        show profile                 show most recent system.profile entries with time >= 1ms
        show logs                    show the accessible logger names
        show log [name]              prints out the last segment of log in memory, 'global' is
default
        use <db_name>                set current database
        db.foo.find()                list objects in collection foo
        db.foo.find( { a : 1 } )     list objects in foo where a == 1
        it                           result of the last line evaluated; use to further iterate
        DBQuery.shellBatchSize = x   set default number of items to display on shell
        exit                         quit the mongo shell
>
```

# A5.9.3 MongoDB shell coding

## 1. make my own db (hsnn) & insert one record (document)

```
use hs00
show collections

insert record
db.user.insert({first:”Redwoods”, last:”Yi”})

show collections

show dbs

db.user.find()
```



# A5.9.3 MongoDB shell coding

## 2. insert more records with different schema & show records

insert record2
insert record3

show collections

db.user.find()

db.user.find().pretty()

```
명령 프롬프트 - mongo
> db.user.insert({first:"Chaos", last:"Kim"})
WriteResult({ "nInserted" : 1 })
> db.user.insert({first:"Gildong", last:"Hong"})
WriteResult({ "nInserted" : 1 })
> show collections
user
> db.user.find()
{ "_id" : ObjectId("5a66b44b9f0d55608f5f7582"), "first" : "Redwoods", "last" : "Yi" }
{ "_id" : ObjectId("5a66b5759f0d55608f5f7583"), "first" : "Chaos", "last" : "Kim" }
{ "_id" : ObjectId("5a66b5869f0d55608f5f7584"), "first" : "Gildong", "last" : "Hong" }
> 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 는 12bytes의 hexadecimal 값으로서, 각 document의 유일함(uniqueness)을 제공합니다.
이 값의 첫 4bytes는 현재 timestamp, 다음 3bytes는 machine id, 다음 2bytes는 MongoDB 서버의 프로세스id, 마지막 3bytes는 순차번호입니다.

# 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
HSnn_mongo_schemas.png

# A5.9.3 MongoDB shell coding

## 4. remove one of  records (or documents)

remove record3

db.user.find().pretty()

명령 프롬프트 - mongo

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

# 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

HSnn_mongo_update.png

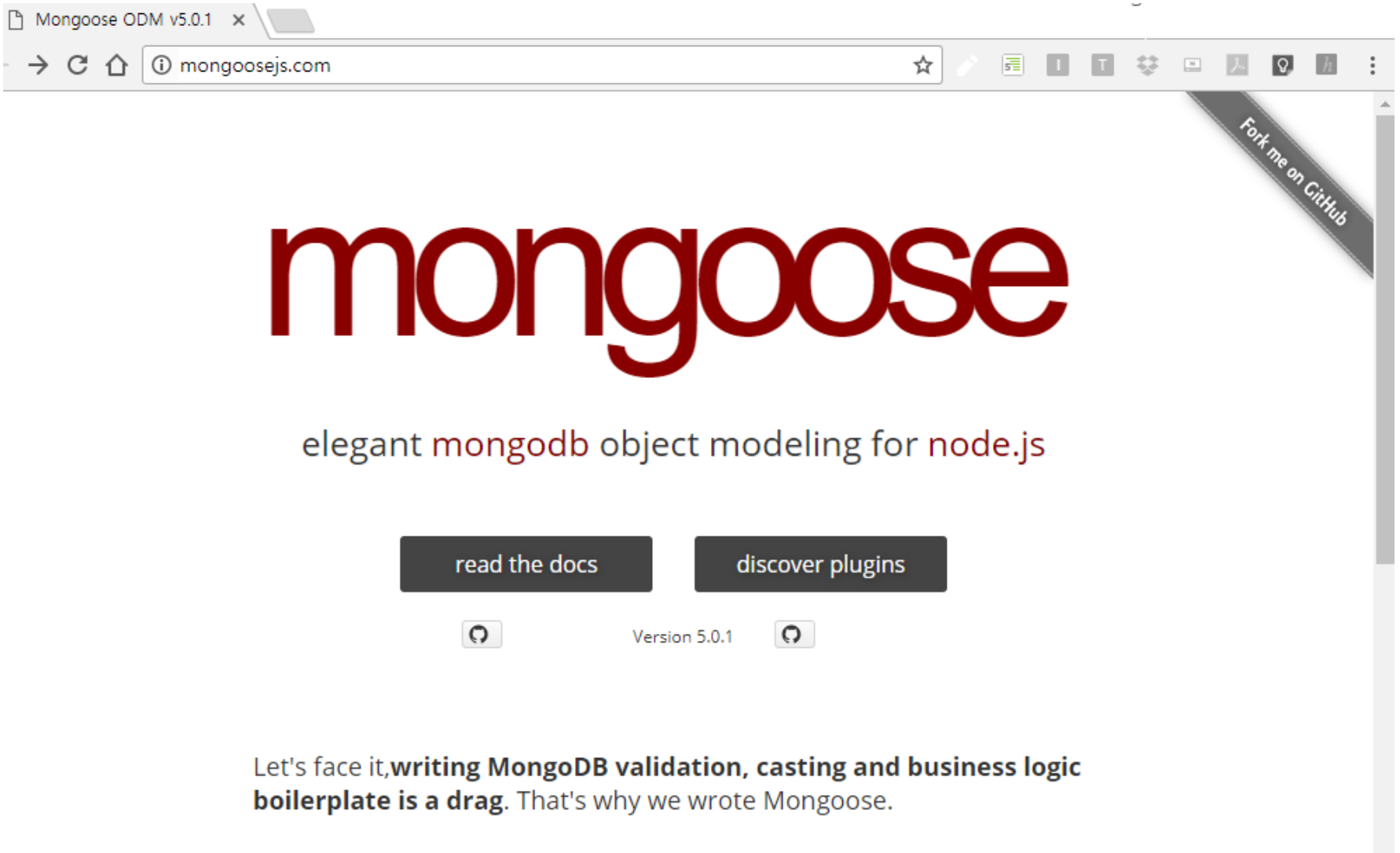


# Node.js

# +

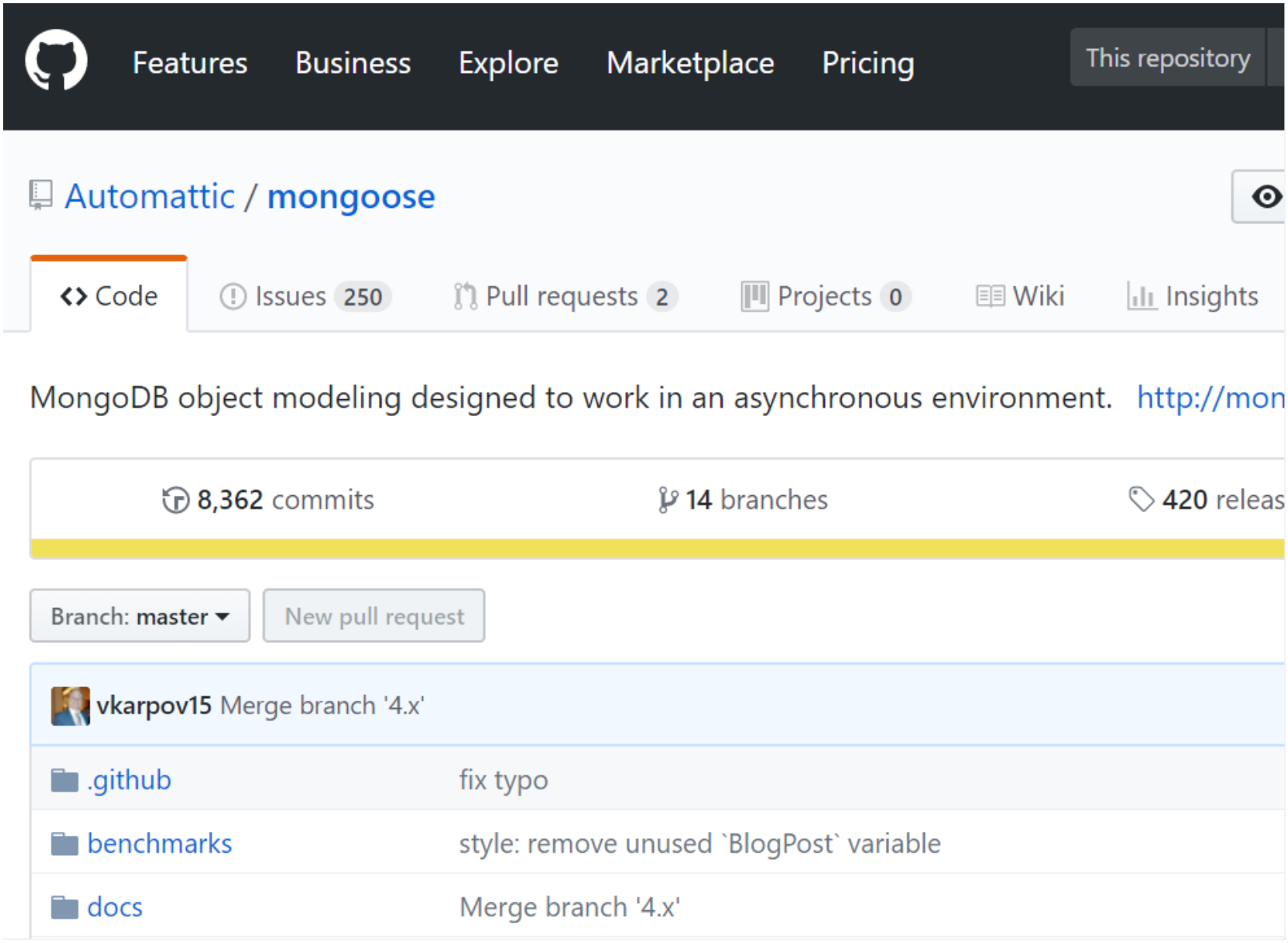
# MongoDB

# A5.9.4 MongoDB + Node.js：mongoose




http://mongoosejs.com/

# A5.9.4 MongoDB + Node.js : mongoose




https://github.com/Automattic/mongoose

# A5.9.4 MongoDB + Node.js : mongooseJS

## 1. Install mongoose in node.js project http://mongoosejs.com/

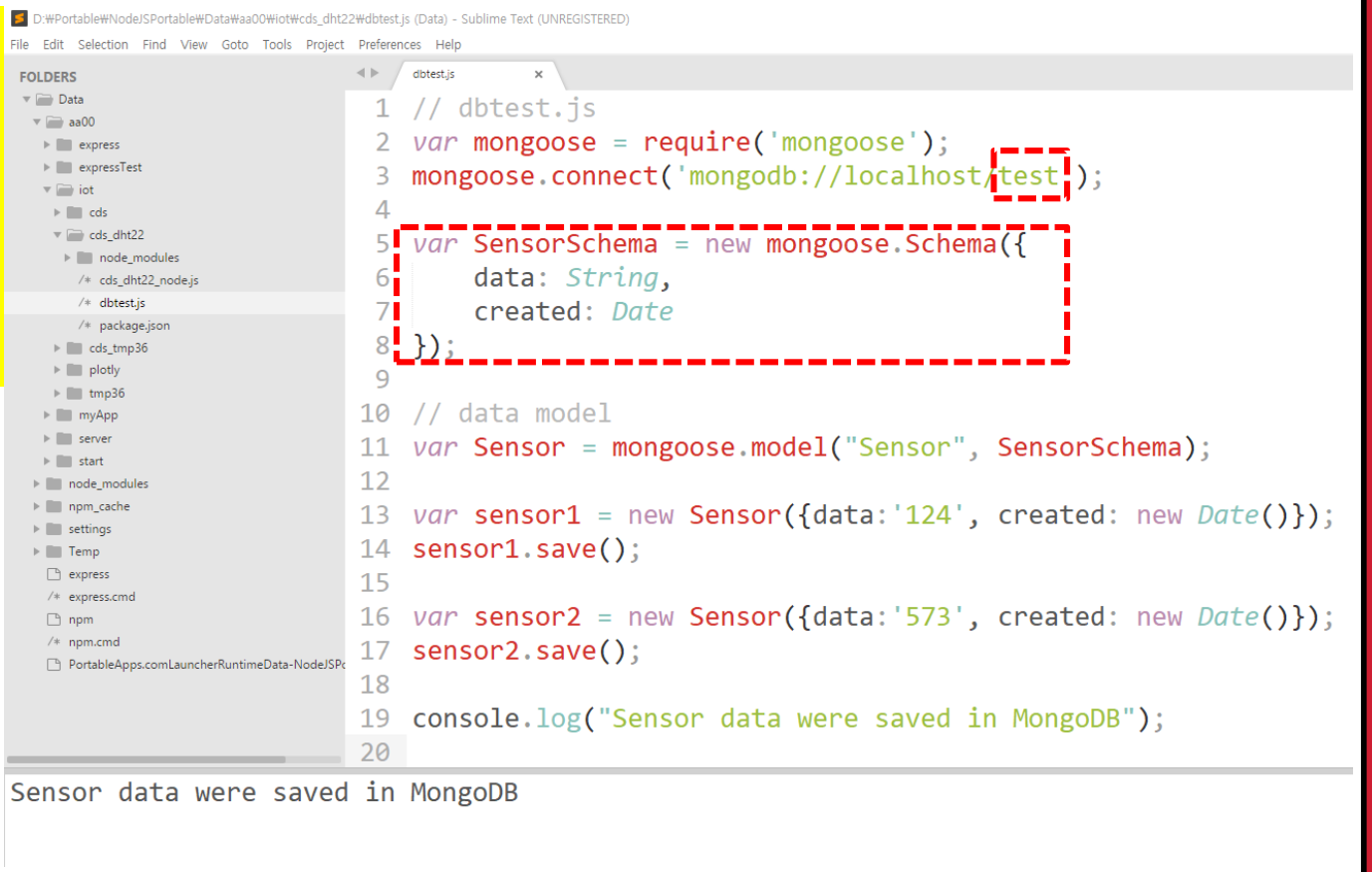
- Go to cds_dht22 project
- npm install --save mongoose

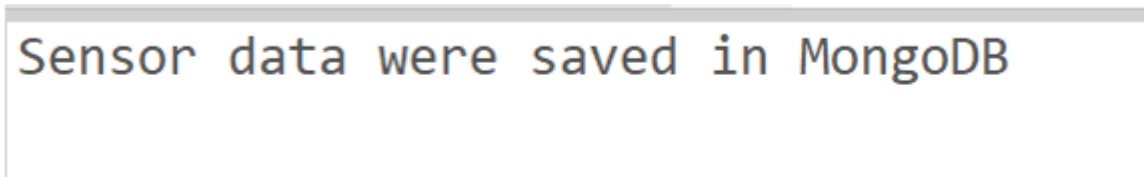
```
NodeJS
D:\Portable\NodeJSPortable\Data\aa00\iot\cds_dht22>npm install -s mongoose
loadRequestedDeps → fetch ? ??????????????????????????????????????????????????????????
loadRequestedDeps → fetch ? ??????????????????????????????????????????????????????????
loadRequestedDeps → netwo ? ??????????????????????????????????????????????????????????
loadRequestedDeps → fetch ? ??????????????????????????????????????????????????????????
loadDep:sliced → request  ? ??????????????????????????????????????????????????????????
loadDep:sliced → 200      ? ??????????????????????????????????????????????????????????
loadDep:sliced → fetch    ? ??????????????????????????????????????????????????????????
loadDep:sliced → headers  ? ??????????????????????????????????????????????????????????
loadDep:sliced → fetch    ? ??????????????????????????????????????????????????????????
loadDep:sliced → fetch    ? ??????????????????????????????????????????????????????????
loadDep:sliced → get      ? ??????????????????????????????????????????????????????????
loadDep:sliced → afterAdd ? ??????????????????????????????????????????????????????????
extract:mongoose → gunzla ? ??????????????????????????????????????????????????????????
extract:mongoose → gently ? ??????????????????????????????????????????????????????????
finalize:sliced → finaliz ? ??????????????????????????????????????????????????????????
build:resolve-from → link ? ??????????????????????????????????????????????????????????
cds_dht22@1.0.0 D:\Portable\NodeJSPortable\Data\aa00\iot\cds_dht22
└──mongoose@5.0.1  extraneous


D:\Portable\NodeJSPortable\Data\aa00\iot\cds_dht22>
```

# A5.9.4 MongoDB + Node.js : mongoose

## 2. node.js project using mongoose (use Sublime Text 3)

- cds_dht22 project in SBT3
- New file: dbtest.js
- ^B (run dbtest.js)



# A5.9.4 MongoDB + Node.js : mongoose

## 3. node.js project using mongoose (mongo shell)

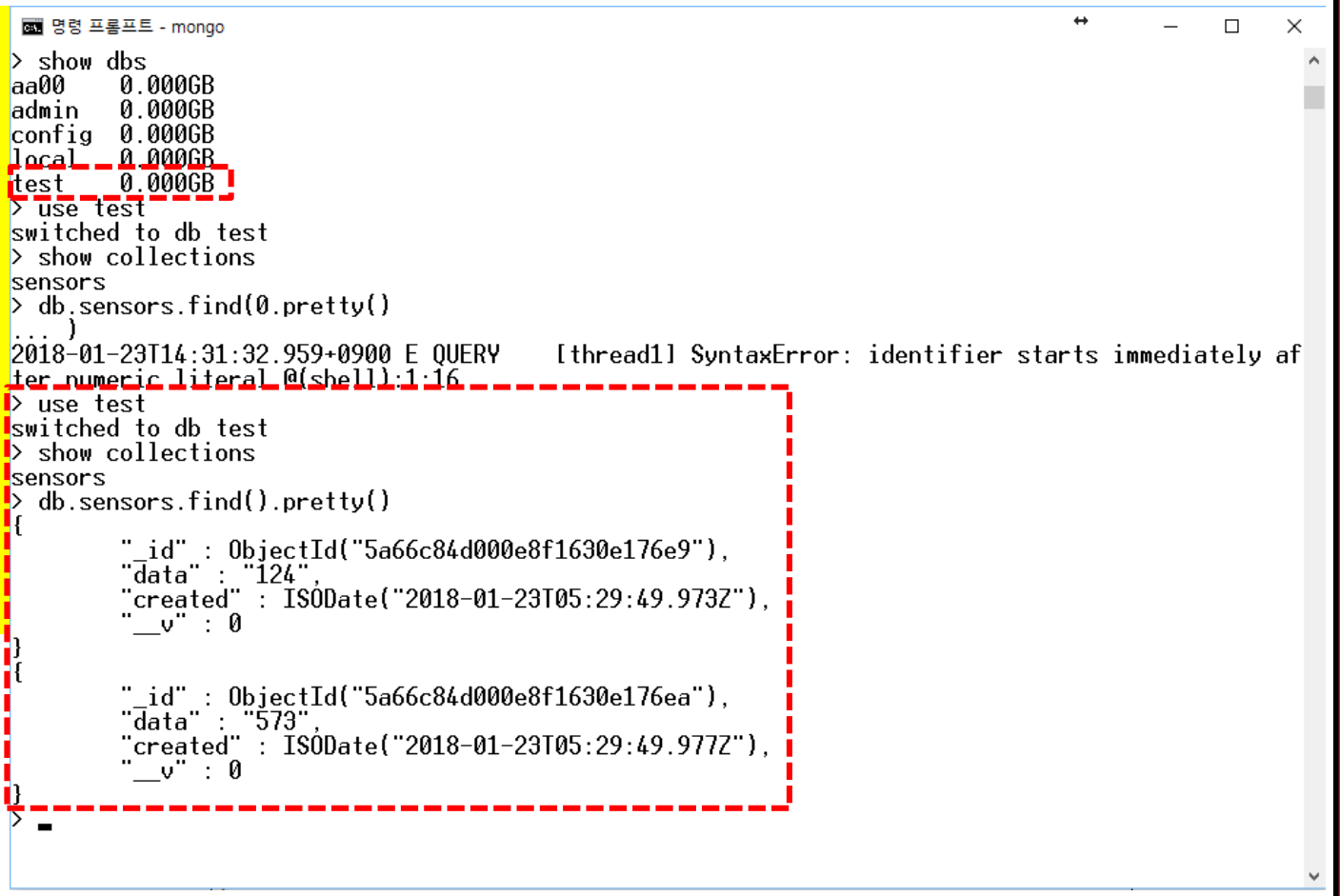
**Mongo shell**

**> show dbs**

**> use test**

**> show collections**

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



# 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 | cds_node.js

```
1  // dbtest2.js
2  var mongoose = require('mongoose');
3  mongoose.connect('mongodb://localhost/test2');
4
5  var SensorSchema = new mongoose.Schema({
6      data: String,
7      created: String
8  });
9
10 // data model
11 var Sensor = mongoose.model("Sensor", SensorSchema);
12
13 var sensor1 = new Sensor({data:'124', created: getDateString()});
14 sensor1.save();
15
16 var sensor2 = new Sensor({data:'573', created: getDateString()});
17 sensor2.save();
18
19 console.log("[dbtest2.js]: Sensor data were saved in MongoDB");
20
21 // helper function to get a nicely formatted date string
22 function getDateString() {
23     var time = new Date().getTime();
24     // 32400000 is (GMT+9 Korea, GimHae)
25     // for your timezone just multiply +/-GMT by 3600000
26     var datestr = new Date(time +32400000).
27     toISOString().replace(/T/, ' ').replace(/Z/, '');
28     return datestr;
29 }
```

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

# A5.9.4 MongoDB + Node.js : mongoose

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

**Mongo shell**

**> show dbs**

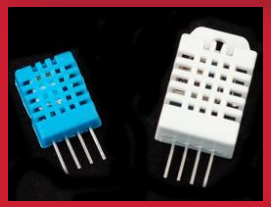
**> use test2**

**> 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.
FOR GIANT IDEAS



mongoose
elegant mongodb object modeling for node.js

# 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
>
```

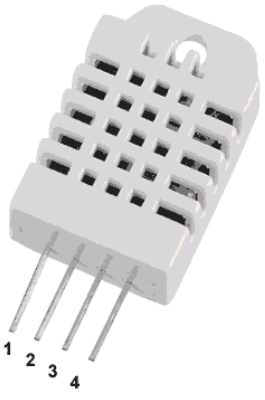
```
mongo db connection OK.
info() - Current date is 2015-11-26 12:04:21.411, Lumi: 67
info() - Current date is 2015-11-26 12:04:26.415, Lumi: 67
info() - Current date is 2015-11-26 12:04:31.416, Lumi: 67
info() - Current date is 2015-11-26 12:04:36.422, Lumi: 104
info() - Current date is 2015-11-26 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
```

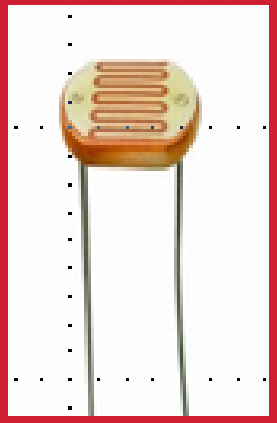
# Arduino & Node.js & MongoDB


## Multi-sensors

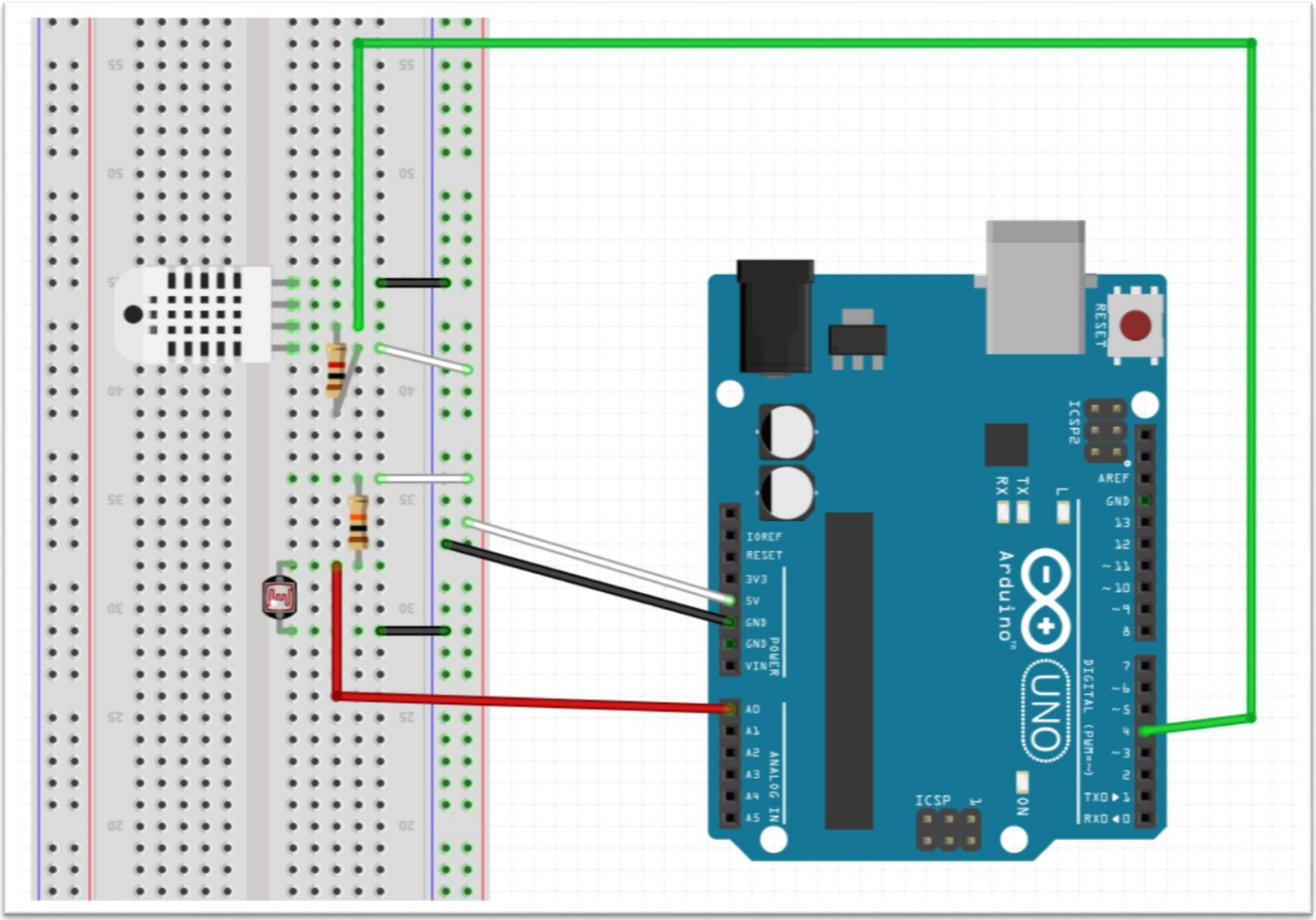
## DHT22 + CdS

| DHT22 pins | |
|---|---|
| 1 | VCC |
| 2 | DATA |
| 3 | NC |
| 4 | GND |



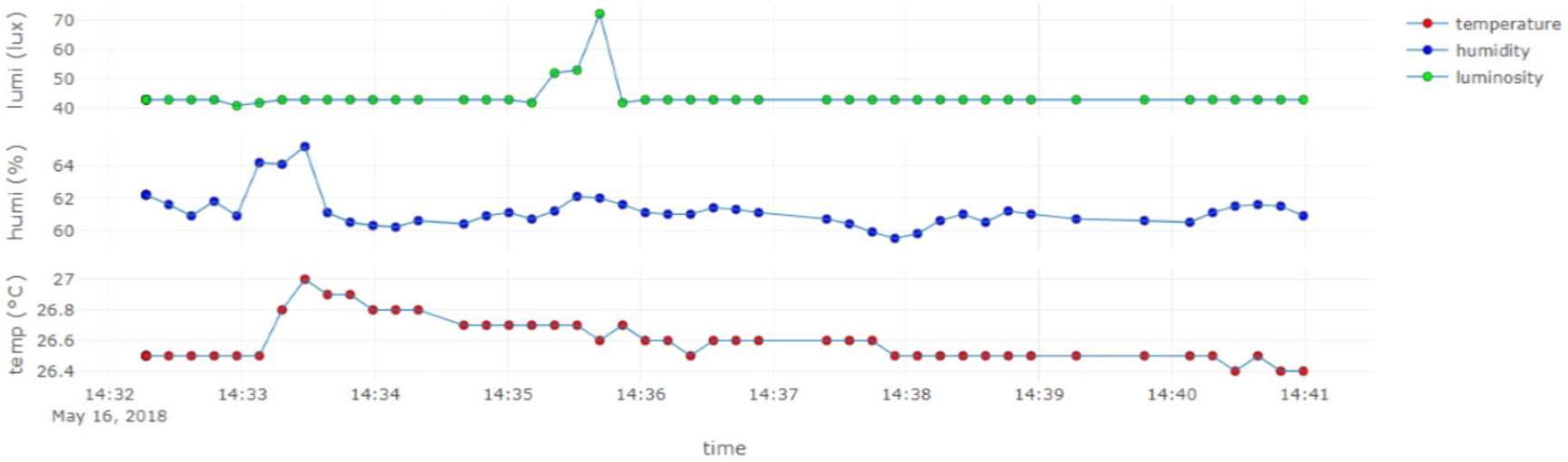



# DHT22 + CdS : circuit



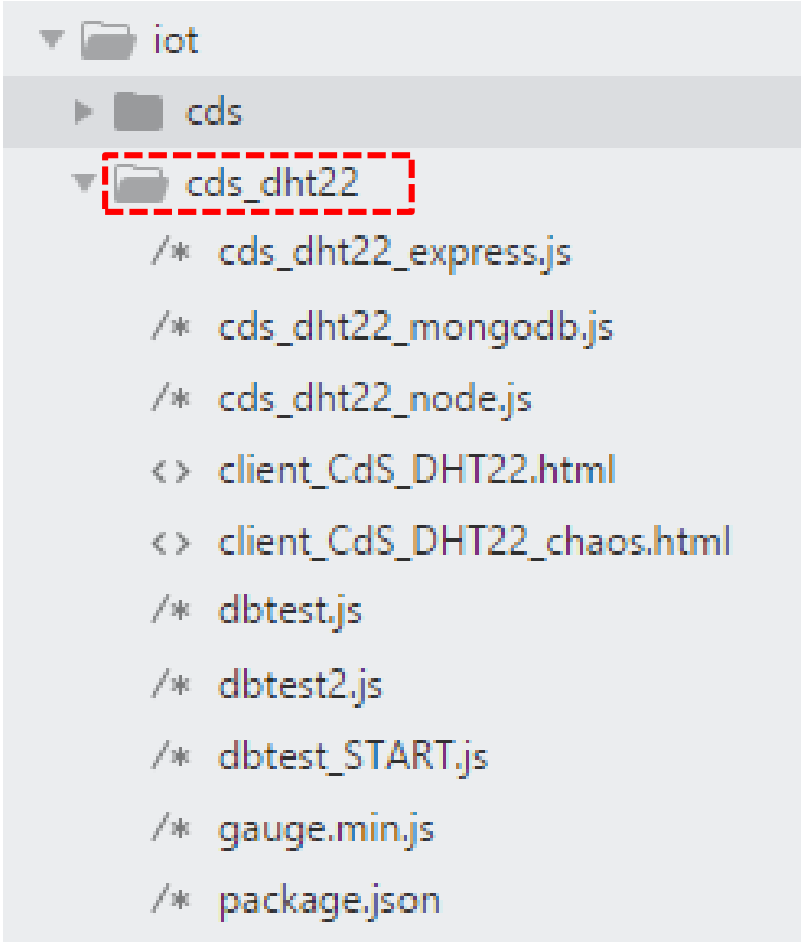
# Real-time Weather Station from sensors


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



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

## 1. 작업 폴더 구조 [2018]



# 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
```

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

## 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**
**HSnn_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!');
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");
```

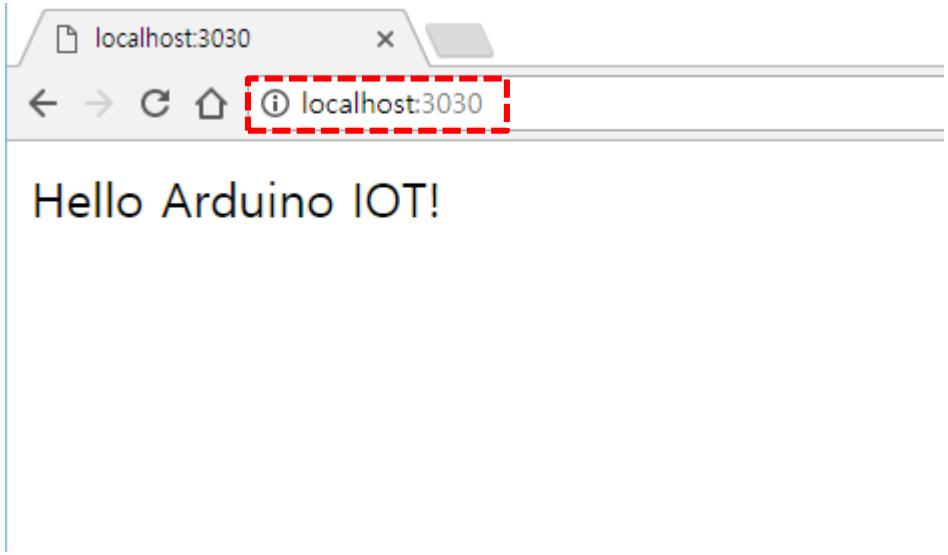
# A5.9.6 DHT22 + CdS + Node.js + MongoDB

## 2.3 cds_dht22_express.js → Run

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

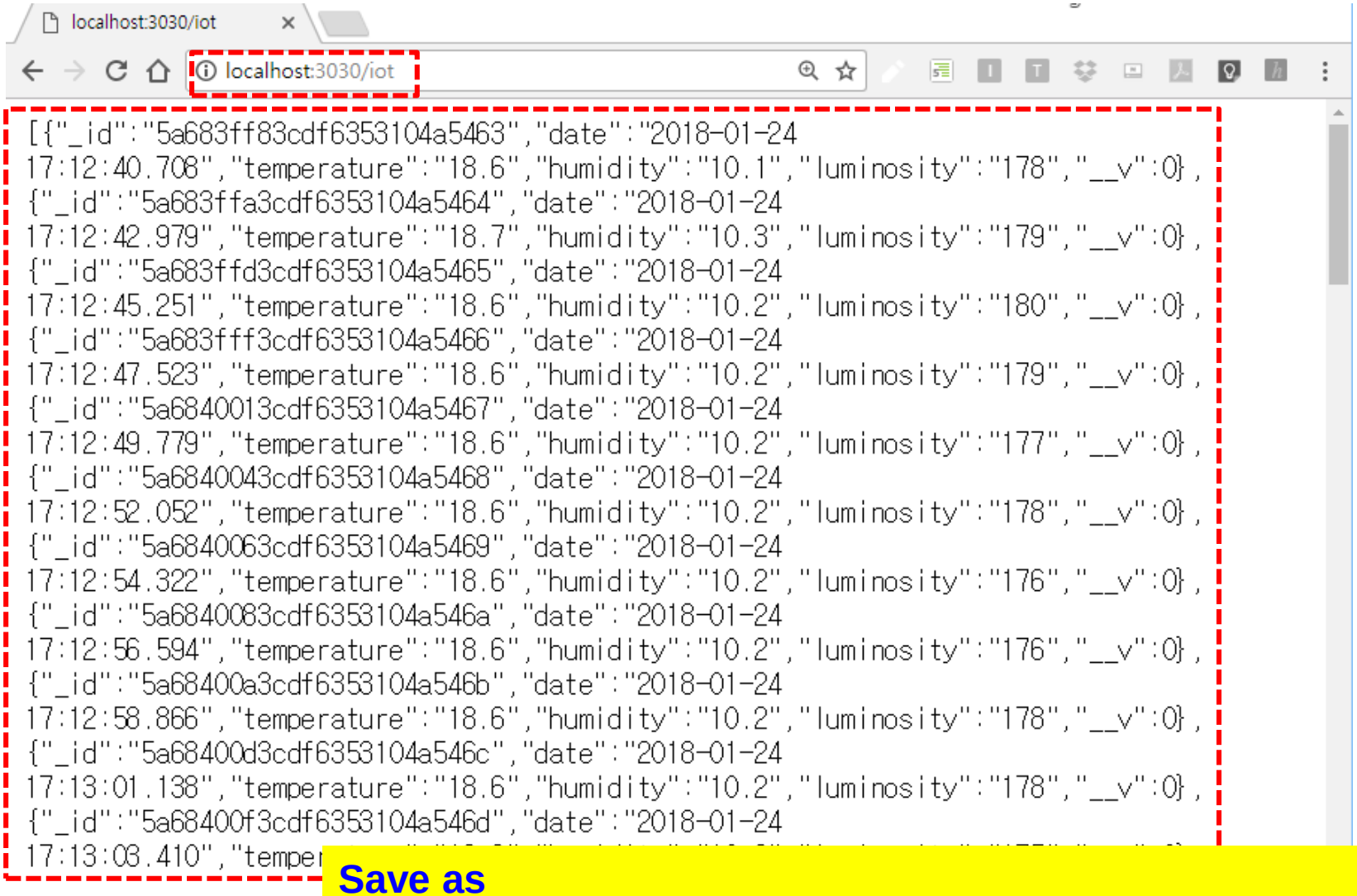
# A5.9.6 DHT22 + CdS + Node.js + MongoDB

## 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

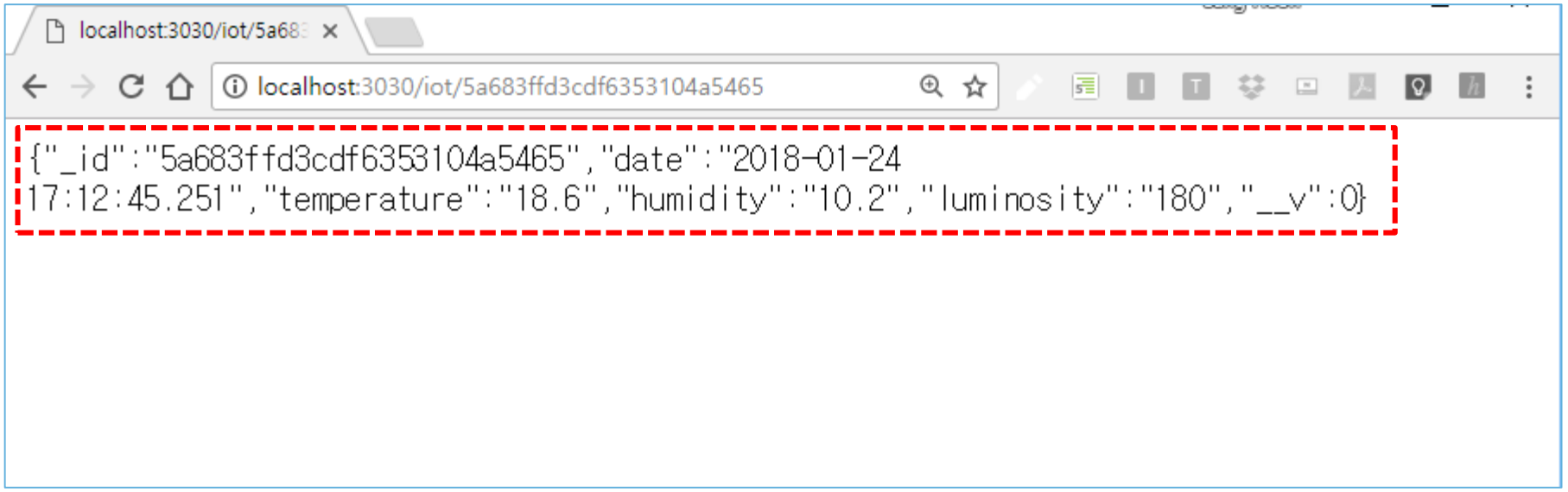



**Save as**

**HSnn_iot_mongodb_web.png**

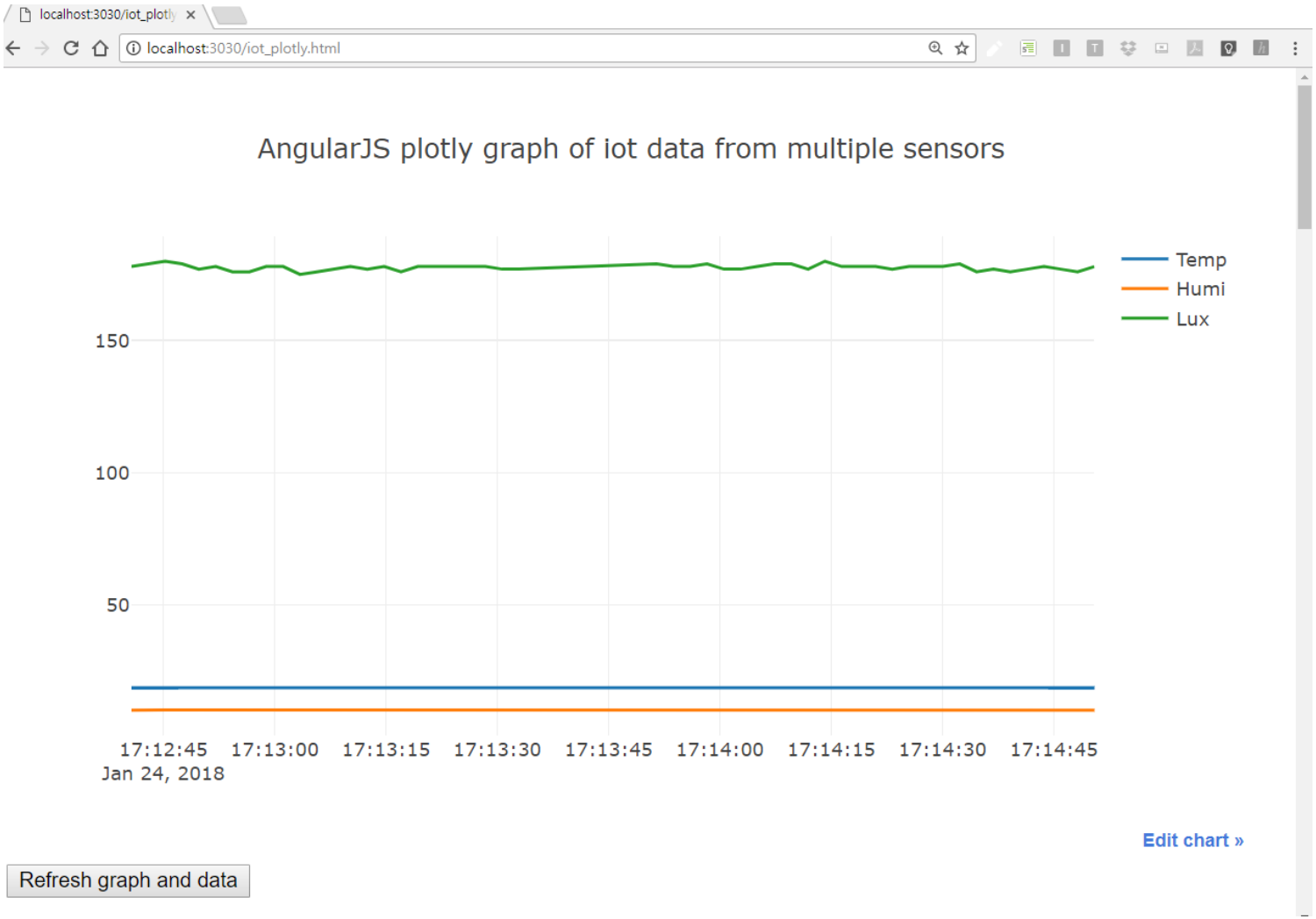
# A5.9.6 DHT22 + CdS + Node.js + MongoDB

## 2.6 cds_dht22_express.js → routing2 http://localhost:3030/iot:id



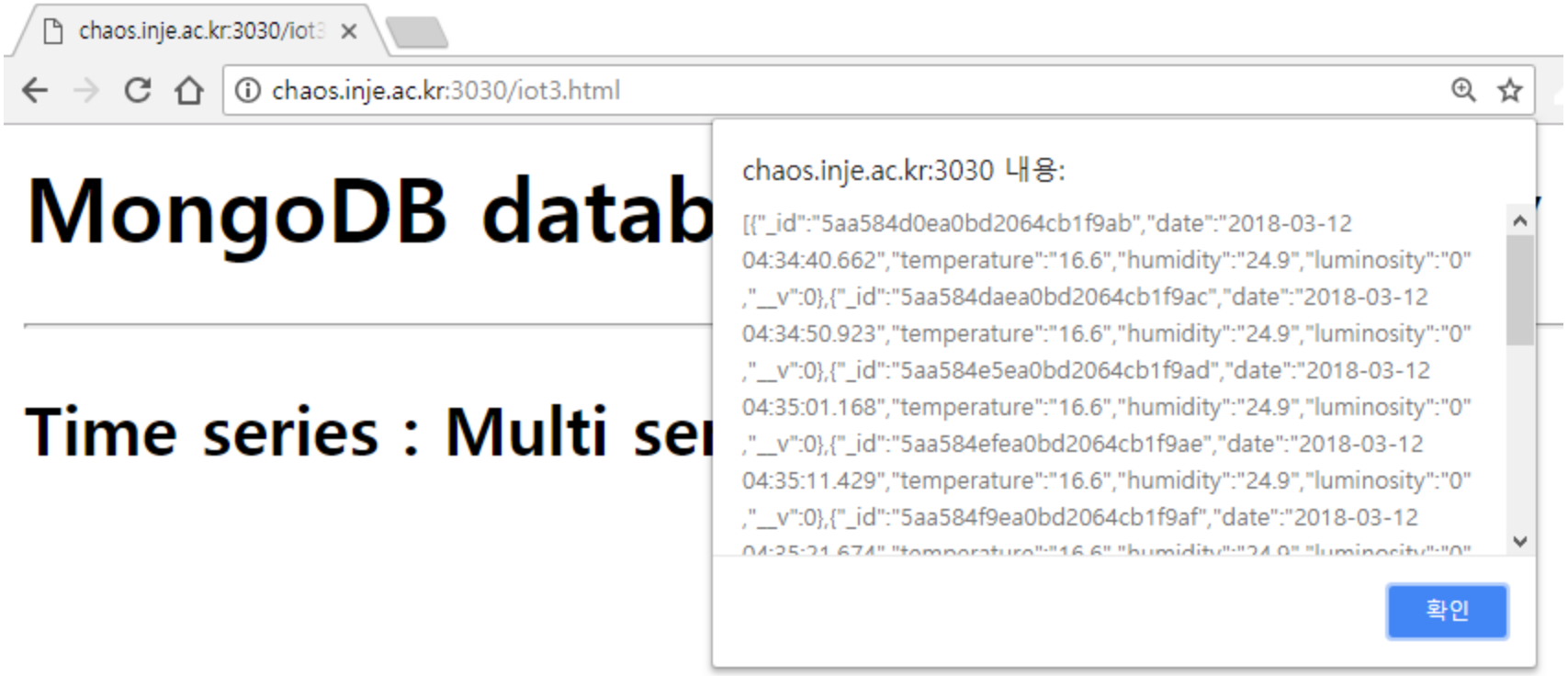
# DHT22 + CdS + Node.js + MongoDB

## [Next week] Web monitoring



# DHT22 + CdS + Node.js + MongoDB

**[Next week] Web monitoring**

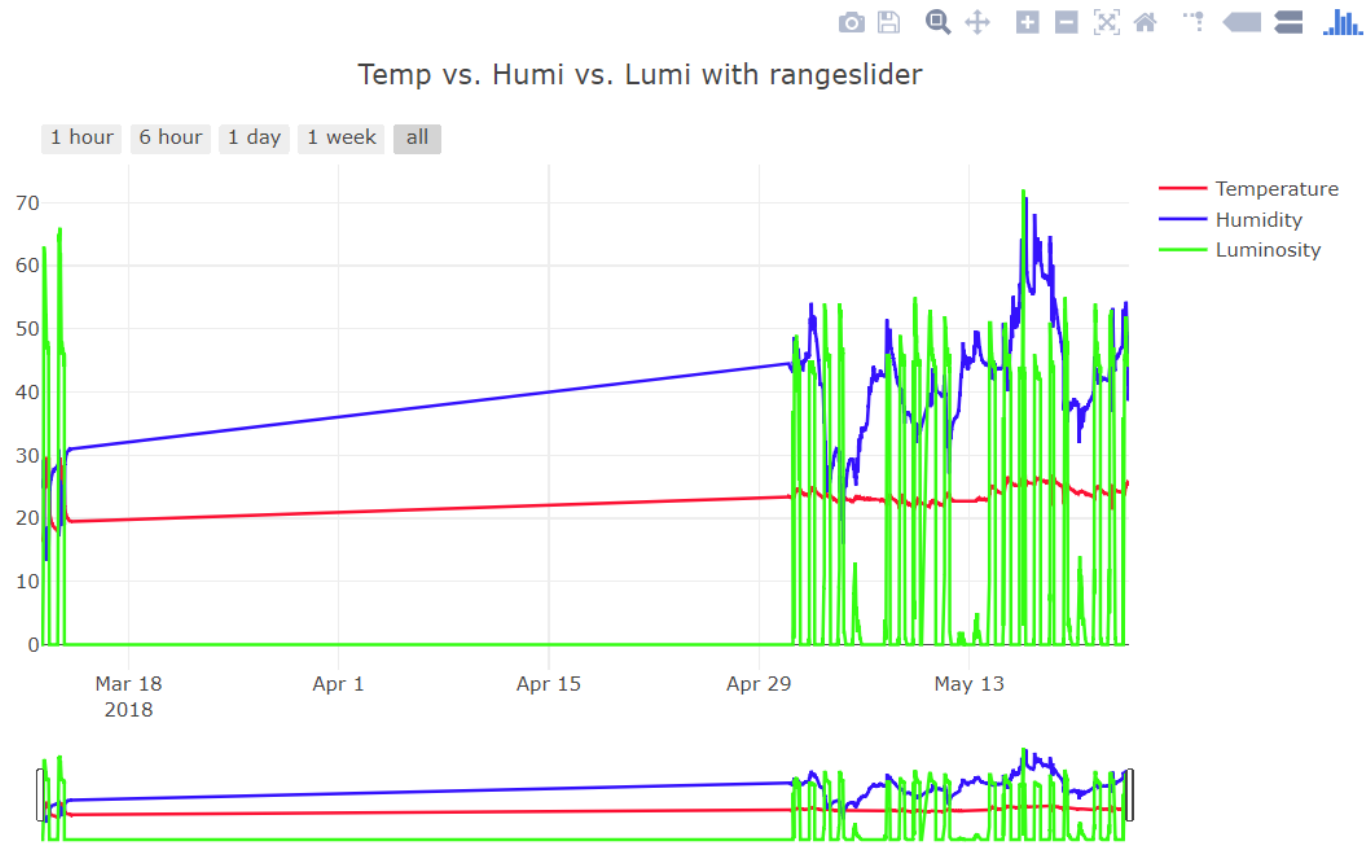


# DHT22 + CdS + Node.js + MongoDB

## [Next week] Web monitoring

### MongoDB database visualization by AA00

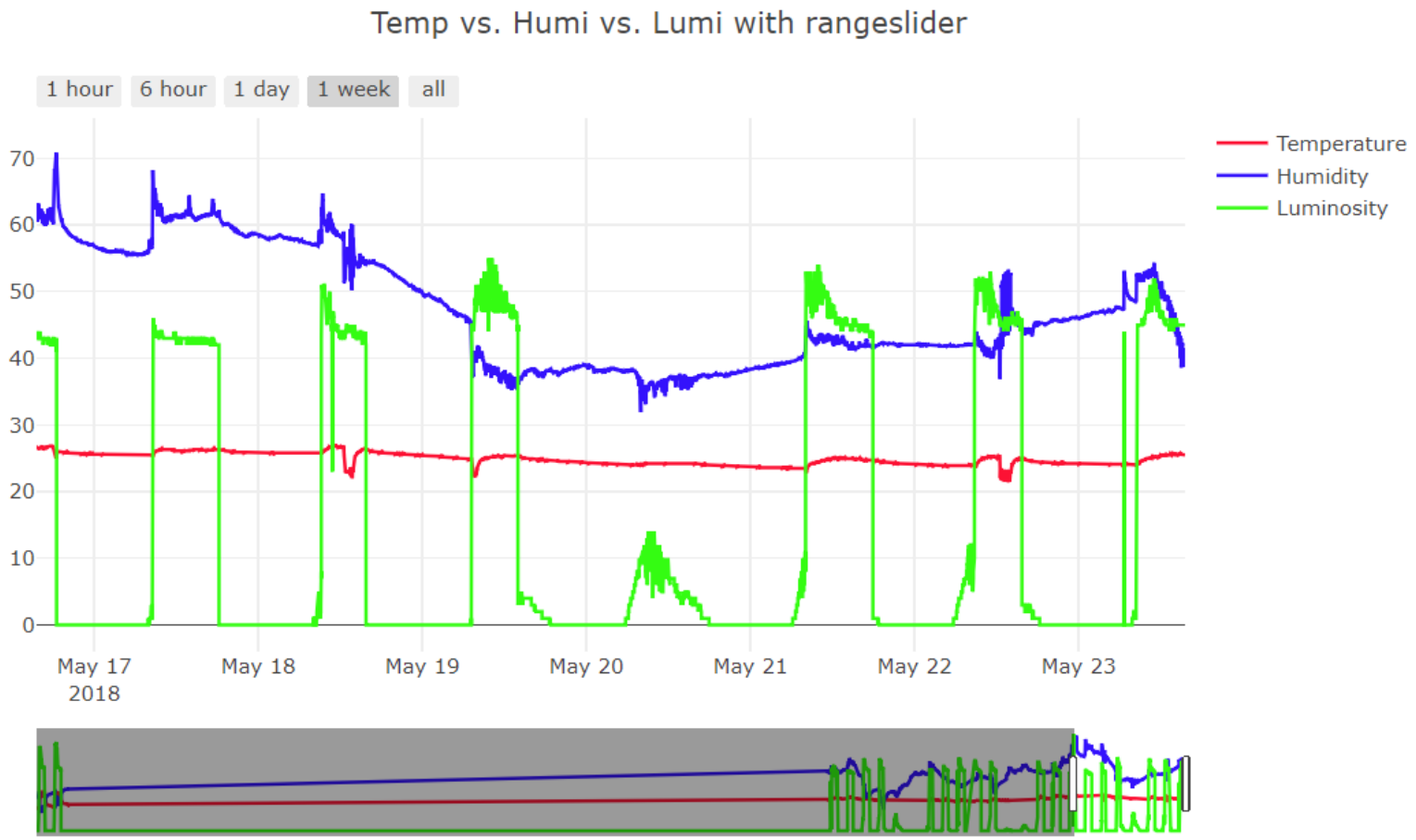
#### Time series : Multi sensor data



# DHT22 + CdS + Node.js + MongoDB

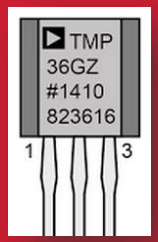
## [Next week] Web monitoring

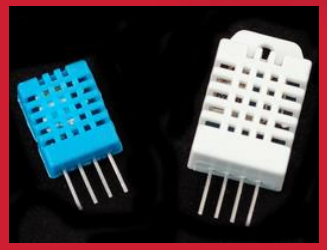
### Time series : Multi sensor data





# [Practice]



- [wk12]
  - RT Data storsging with MongoDB
  - Multi-sensor circuits
  - Complete your project
  - Upload file name : HSnn_Rpt10.zip

# [wk12] Practice-10 HSnn_Rpt10.zip

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

제출파일명 : **HSnn_Rpt10.zip**

- 압축할 파일들

① **HSnn_mongo_schemas.png**

② **HSnn_mongo_update.png**

③ **HSnn_iot_mongodb.png**

④ **HSnn_iot_mongodb_web.png**

**Email : chaos21c@gmail.com**

**[** 제목 : **id,** 이름 **(**수정**) ]**

# Lecture materials

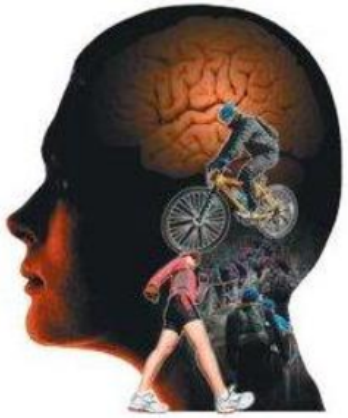
## References & good sites

- ✓ http://www.nodejs.org/ko Node.js
- ✓ http://www.arduino.cc Arduino Homepage
- ✓ http://www.w3schools.com By w3schools
- ✓ http://www.github.com GitHub
- ✓ http://www.google.com Googling

# Github.com/Redwoods

GitHub, Inc. [US] | https://github.com Redwoods (Redwoods Yi) · GitHub

Features Business Explore Marketplace Pricing Search GitHub Sign in or Sign up



## Redwoods Yi

Redwoods

Block or report user

GimHae, Republic of Korea

Overview Repositories 5 Stars 2 Followers 0 Following 0

Pinned repositories

**dht22-iot-project**

Iot project to monitor data streaming from DHT22 wired at Arduino.

HTML

**Lec**

All lectures by Redwoods in Inje University

**arduino-nodejs-plotly-streaming**

This repo introduces a simple and efficient way to plot the streaming data from Arduino with Easy Pulse ppg sensor or DHT11 sensor.

HTML

**hw-coding**

Resource for lecture of Hardware Programming (2017, Inje university)

Arduino

# Github.com/Redwoods/healthcare-signal-iot


Redwoods / Lec

Unwatch 1

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

All lectures by Redwoods in Inje University

Add topics

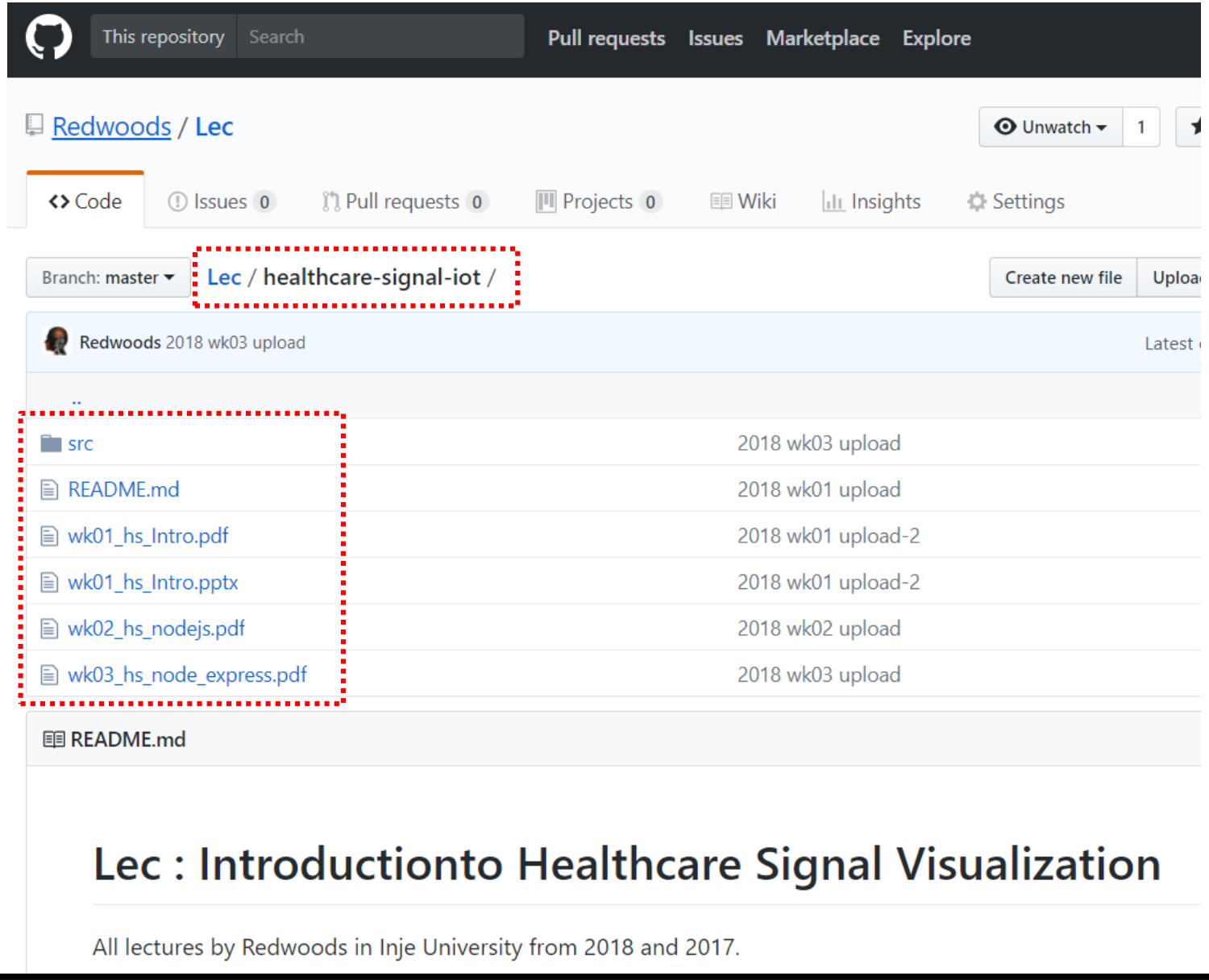
81 commits | 1 branch | 0 releases

Branch: master | New pull request | Create new file | Upload files

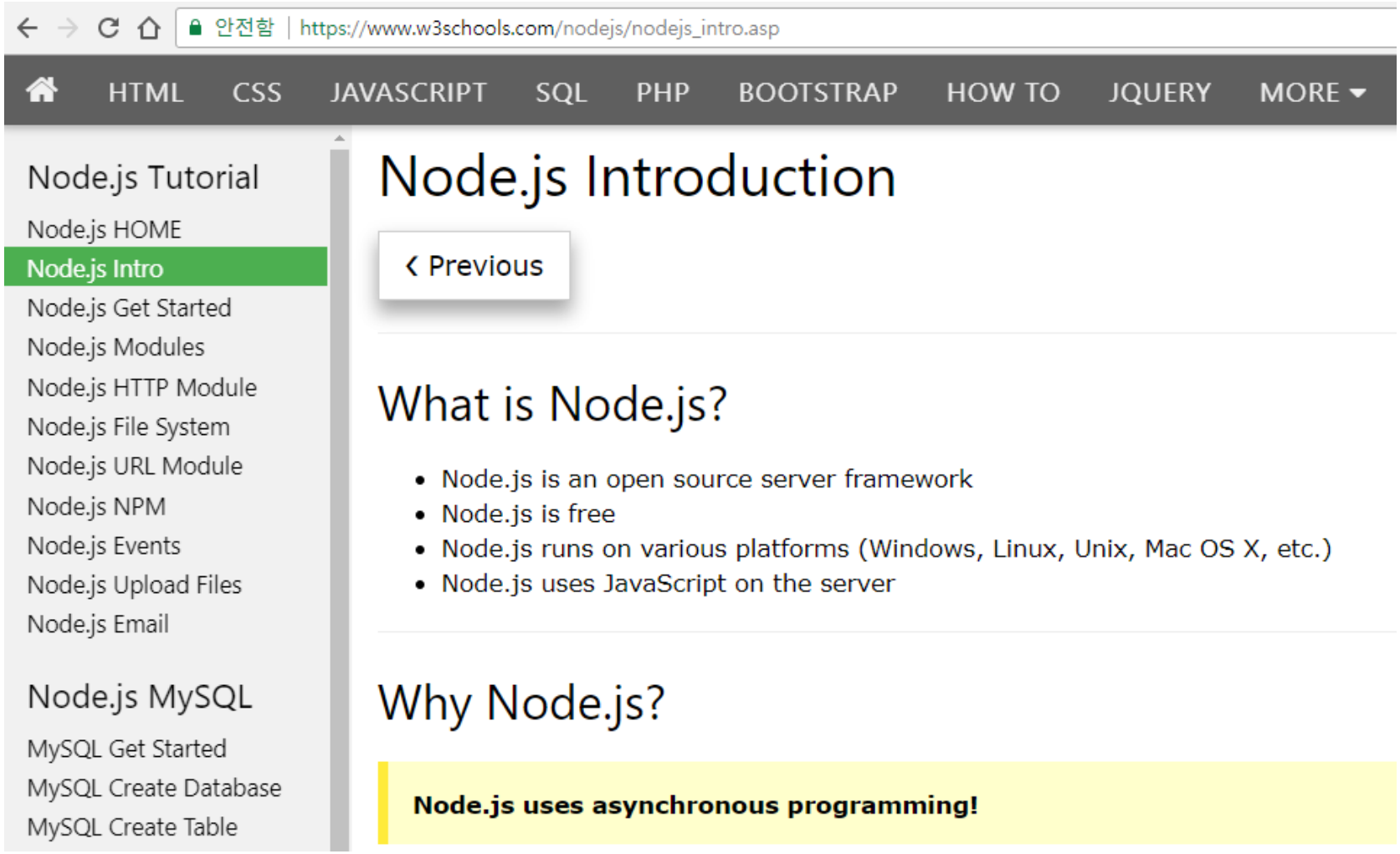
Redwoods 2018 wk01 upload

| | |
|---|---|
| advanced-Arduino-iot | wk16 exam upload |
| ev3 | wk16 final exam. answers |
| healthcare-signal-iot | 2018 wk01 upload |
| html5-basic | 2018 wk01 upload |
| html5-mobile-simulation | wk15 lec upload |
| Lec.Rproj | 2018 wk01 upload |
| README.md | wk03 upload and fix links |

# Github.com/Redwoods/healthcare-signal-iot

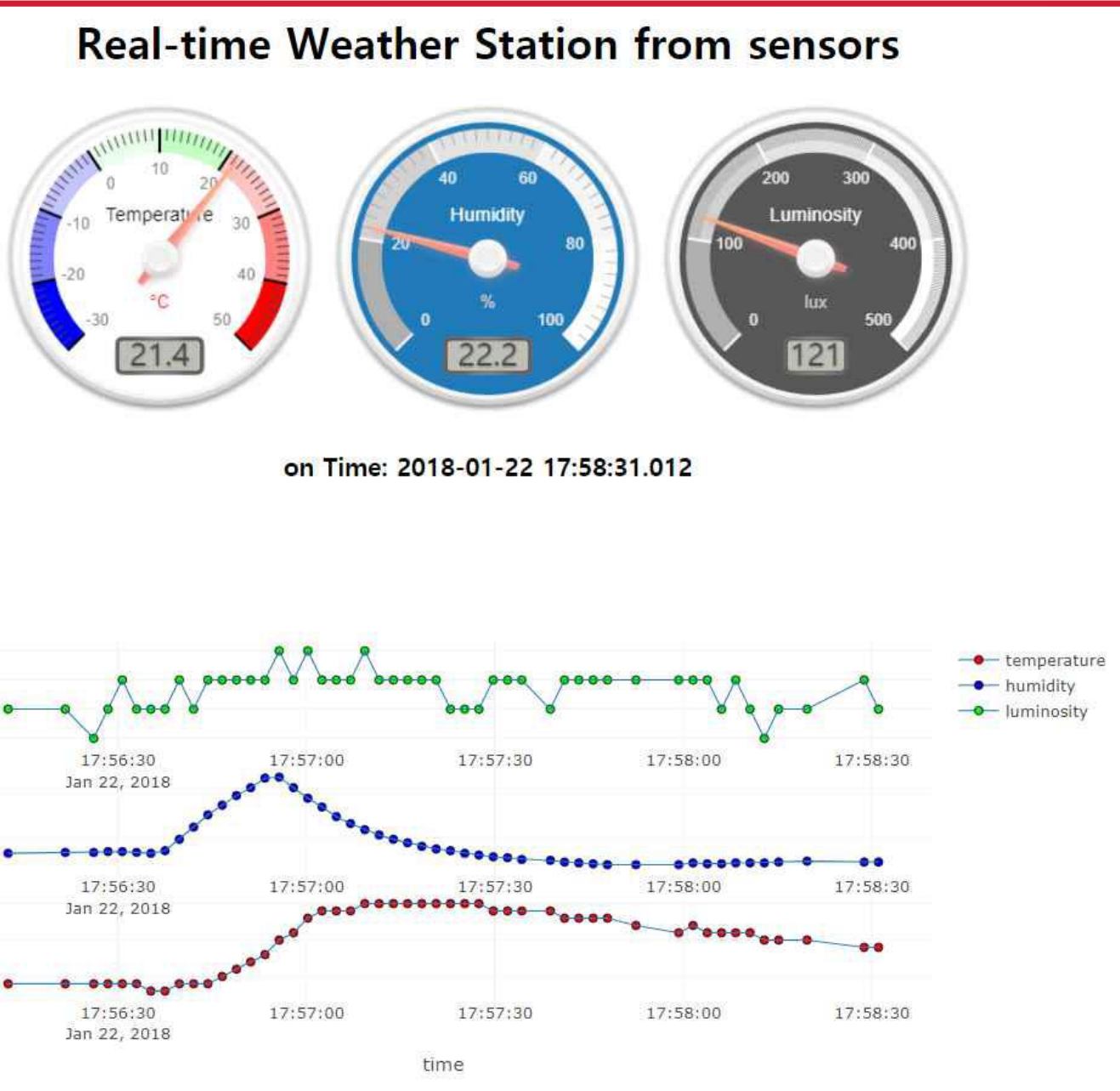


# 1.0 What is node.js?

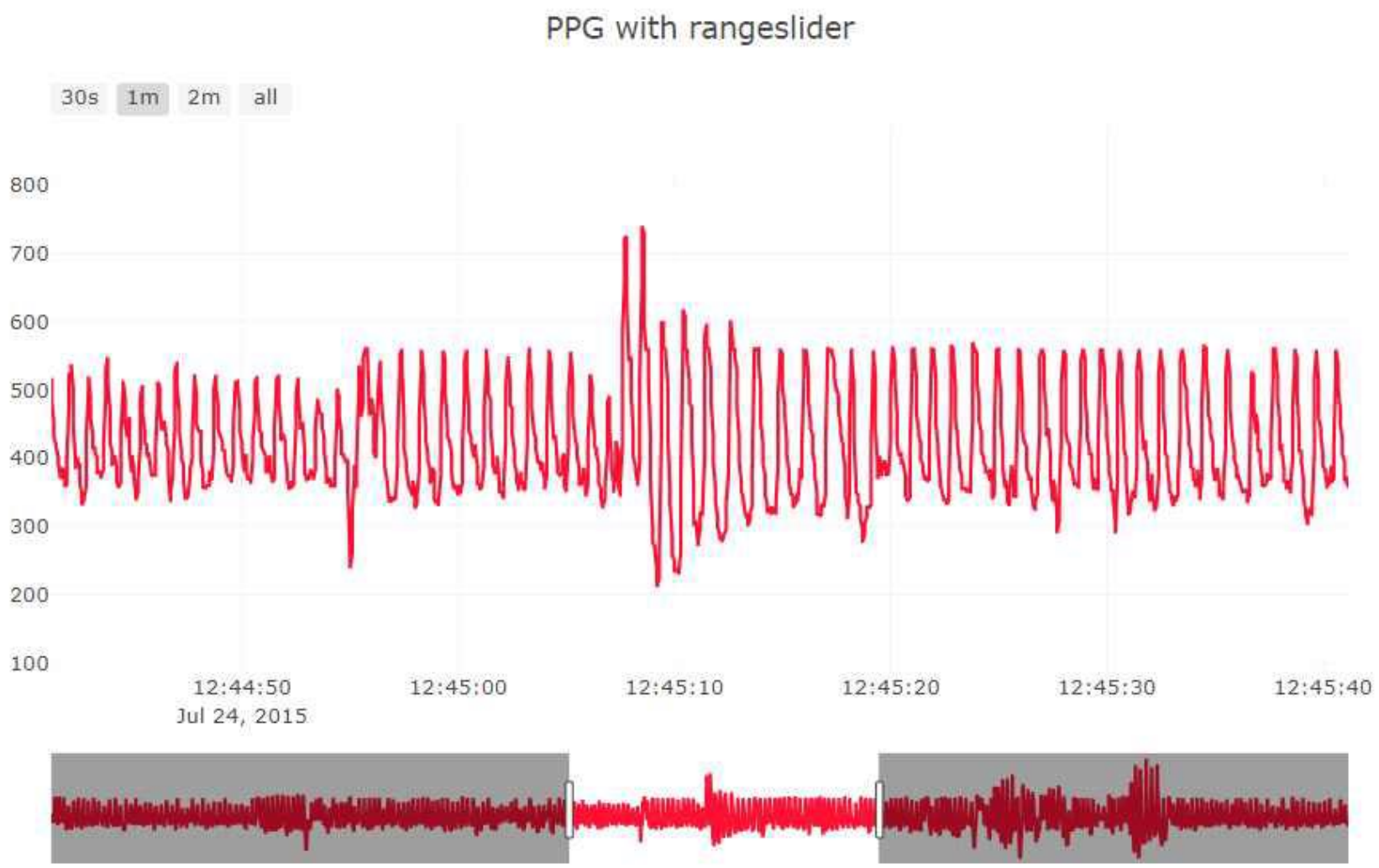



https://www.w3schools.com/nodejs/nodejs_intro.asp

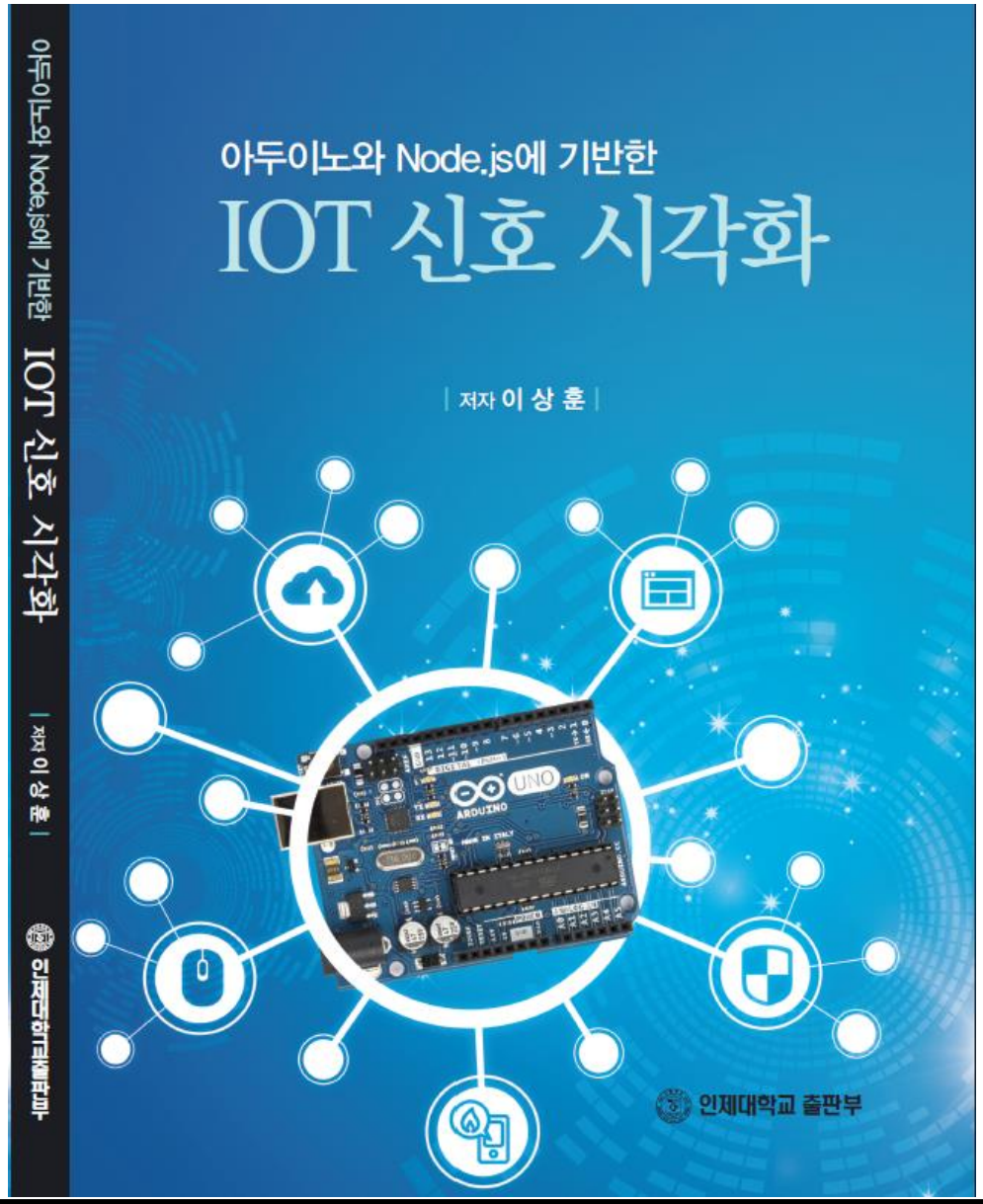
# Target of this class



# Project of this class



# 주교재


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