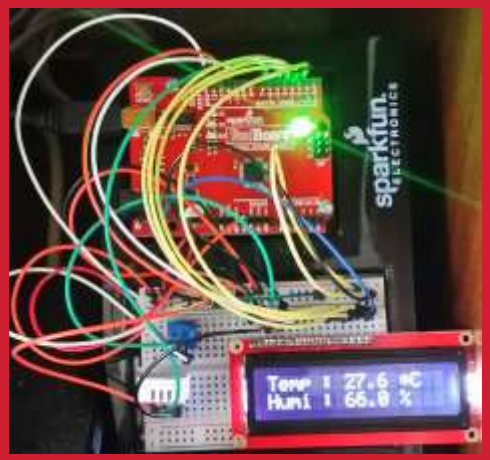
Arduino-IOT

[wk13]

Arduino + Node Data storaging II



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

Drone-IoT-Comsi, INJE University

2nd semester, 2020

Email : chaos21c@gmail.com

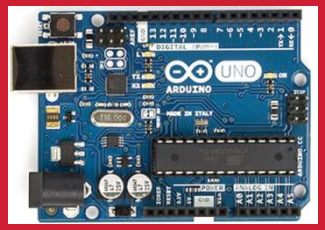
My ID

1분반-목요일 (2학년)

- AA1-01: 강서현
- AA1-02: 강태민
- AA1-03: 김세은
- AA1-04: 여수민
- AA1-05: 정영훈
- AA1-06: 차혁준
- AA1-07: 하태헌
- AA1-08: 김경욱
- AA1-09: 김민욱
- AA1-10: 김민성
- AA1-11: 김민준
- AA1-12: 김인수
- AA1-13: 김현식
- AA1-14: 장성운
- AA1-15: 전승진
- AA1-16: 정희철
- AA1-17: 조동현
- AA1-18: 전동빈
- AA1-19: 신종원

2분반-수요일 (3학년)

- AA2-01: 강민수
- AA2-02: 구병준
- AA2-03: 김종민
- AA2-04: 박성철
- AA2-05: 이승현
- AA2-06: 이창호
- AA2-07: 손성빈
- AA2-08: 안예찬
- AA2-09: 유종인
- AA2-10: 이석민
- AA2-11: 이정문
- AA2-12: 이주원
- AA2-13: 정재영
- AA2-14: 하태성
- AA2-15: 김경미
- AA2-16: 김규년
- AA2-17: 김유빈
- AA2-18: 송다은
- AA2-19: 정주은
- AA2-20: 권준표



[Review]

- [wk11-12]
  - RT Data Visualization with node.js
  - Multiple data and Usage of gauge.js
  - Complete your real-time WEB charts
  - Upload folder: aax-nn-rpt09
  - Use repo “aax-nn” in github

wk11 : Practice : aax-nn-rpt09

- [Target of this week]
  - Complete your works
  - Save your outcomes and upload outputs in github

제출폴더명 : **aax-nn-rpt09**

- 제출할 파일들

① **AAnn_cds_dht22_data.png**

② **AAnn_cds_dht22.html**

③ **AAnn_cds_dht22.png**

④ **All *.ino**

⑤ **All *.js**

⑥ **All *.html**

IOT: HSC

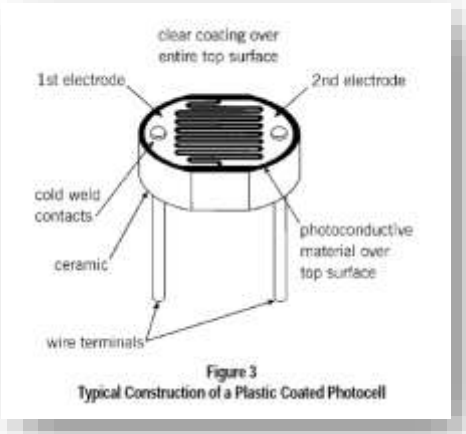
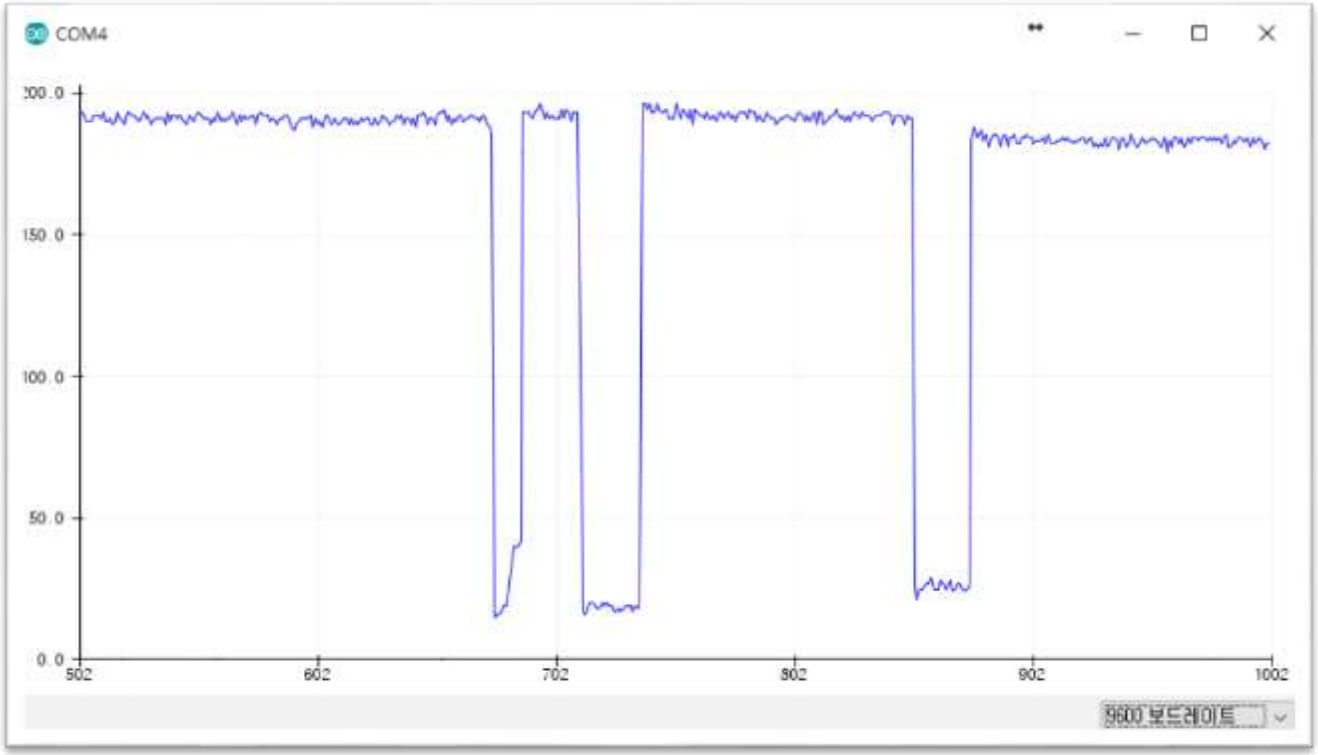
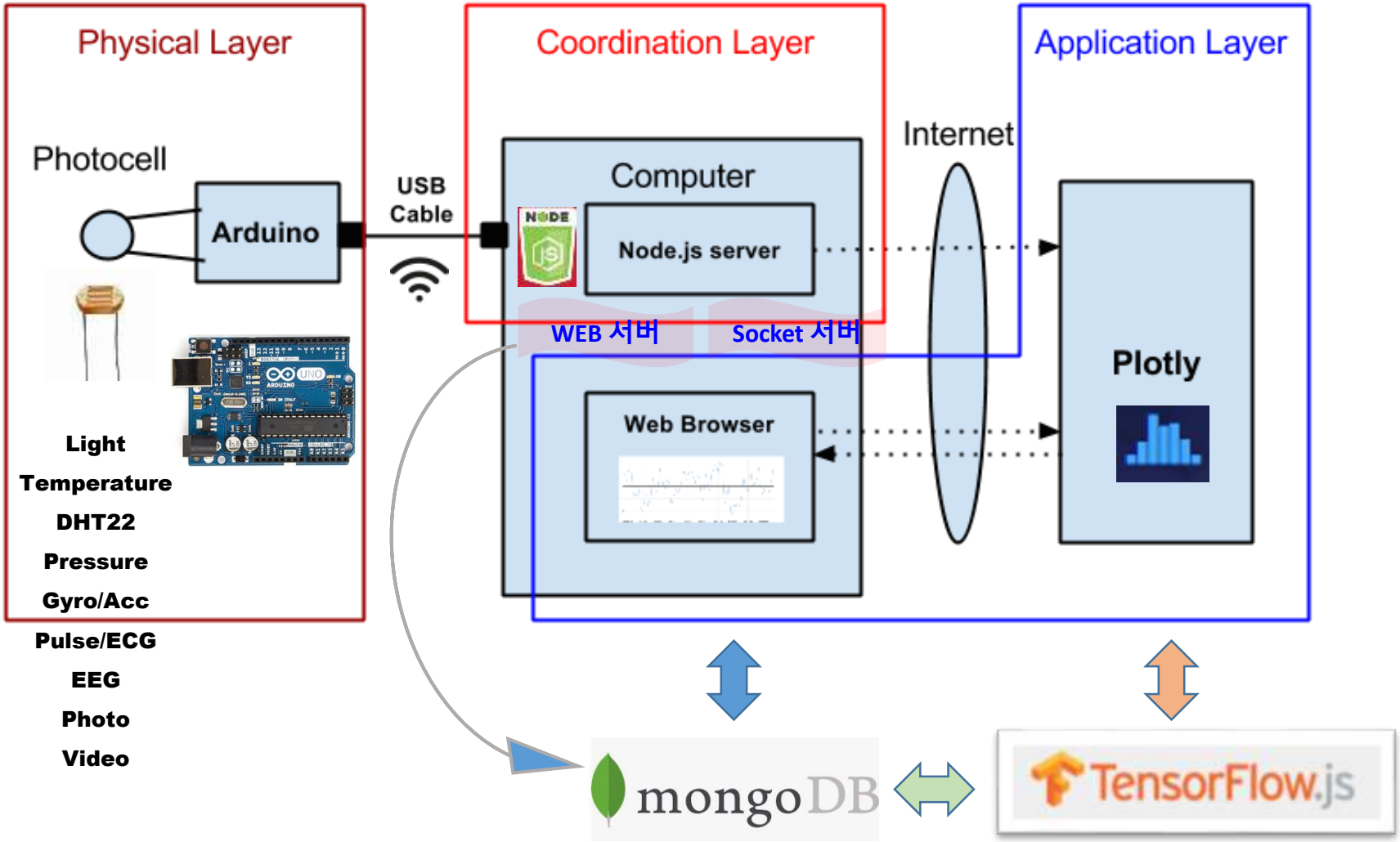


Figure 3
Typical Construction of a Plastic Coated Photocell

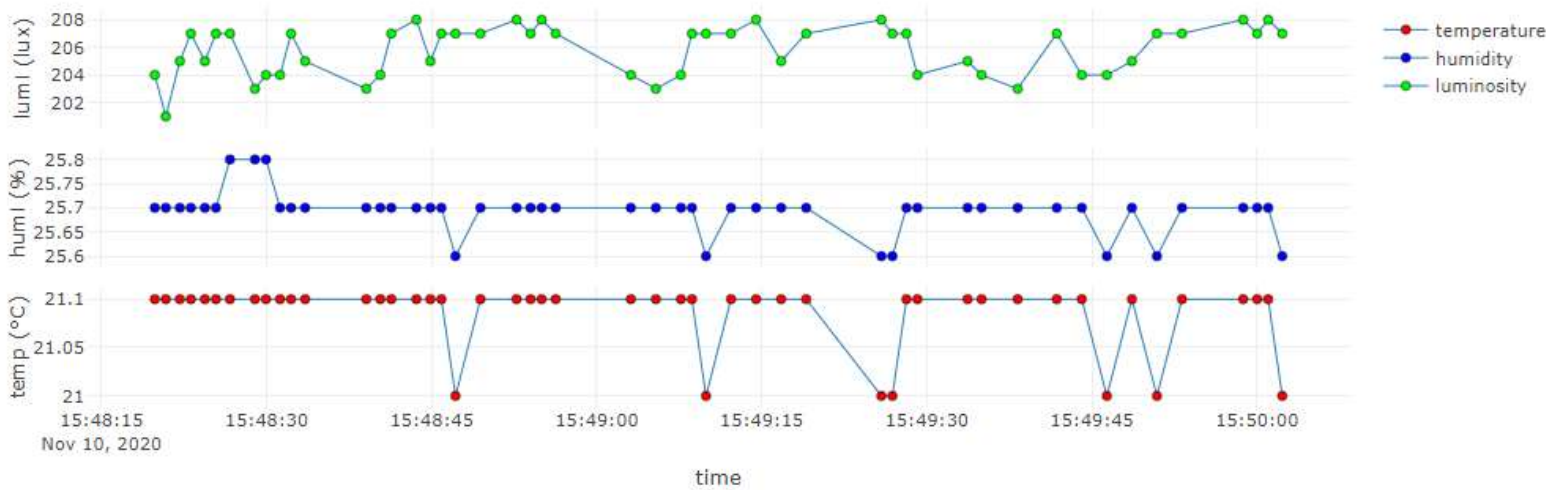


Layout [H S C]


Physical Layer
Photocell
Arduino
Light
Temperature
DHT22
Pressure
Gyro/Acc
Pulse/ECG
EEG
Photo
Video
USB
Cable
Coordination Layer
Computer
Node.js server
WEB 서버
Socket 서버
Web Browser
Internet
Application Layer
Plotly
mongoDB
TensorFlow.js

Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300



A5. Introduction to IoT service

System (Arduino, sDevice, ...)

↓

Data (signal, image, sns, ...)

↓

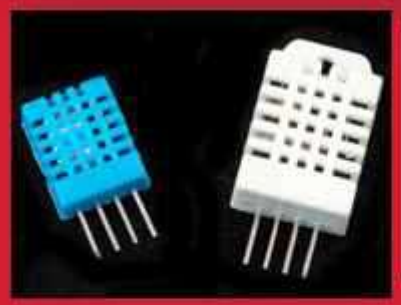
Visualization & monitoring

↓

Data storaging & mining

↓

Service



[Goal]

Arduino + Node.js

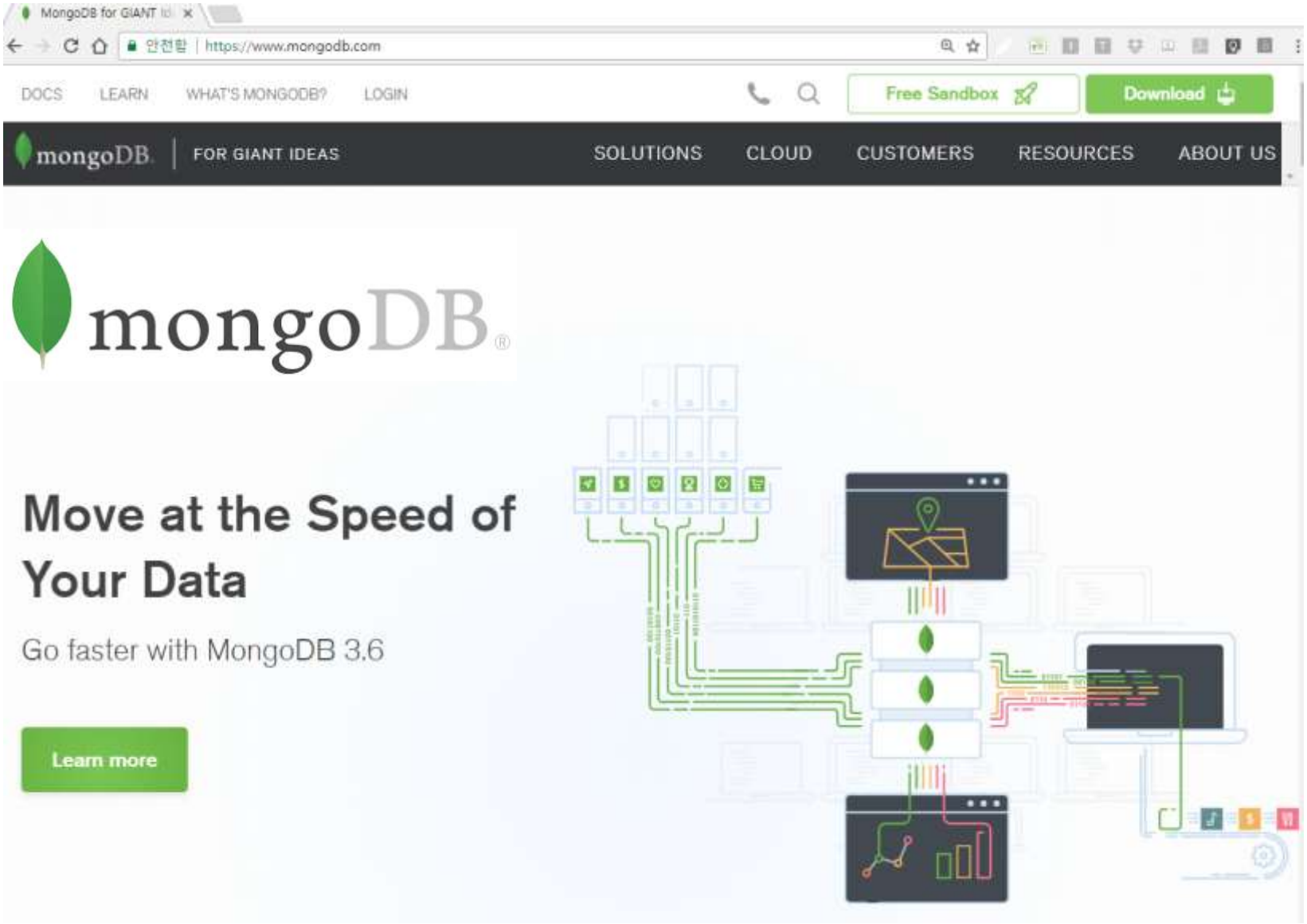
+ plotly.js

+ 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" } }
```

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.

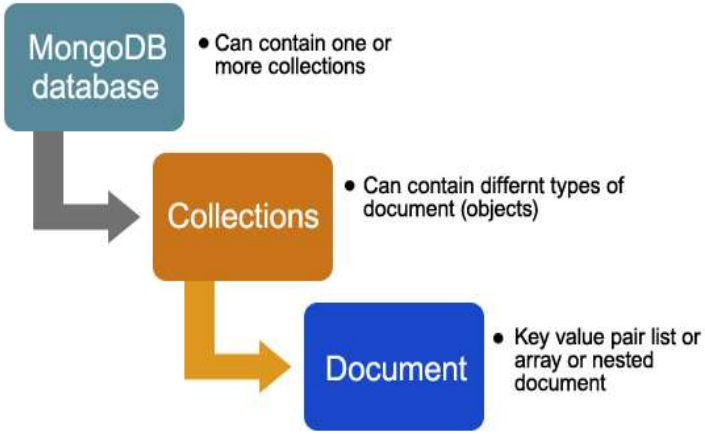
Collection?

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

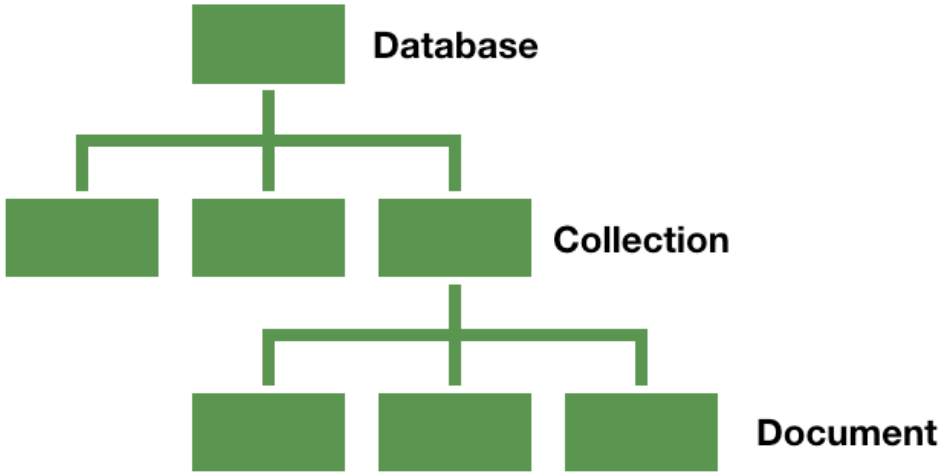
Database?

Database는 Collection들의 물리적인 컨테이너입니다. 각 Database는 파일시스템에 여러파일들로 저장됩니다.

A5.9 MongoDB



https://cdn.educba.com/academy/wp-content/uploads/2019/04/MongoDB-chart2.jpg



https://i.imgur.com/Att4uVC.png

A5.9 MongoDB: download

mongoDB

MongoDB Enterprise Server

MongoDB Community Server

MongoDB offers both an Enterprise and Community version of its powerful distributed document database. The community version offers the flexible document model along with ad hoc queries, indexing, and real time aggregation to provide powerful ways to access and analyze your data. As a distributed system you get high availability through built-in replication and failover along with horizontal scalability with native sharding.

The MongoDB Enterprise Server gives you all of this and more. Review the Enterprise Server tab to learn what else is available.

Available Downloads

Version
4.2.11-rc1 (release candidate)

Platform
Windows

Package
msi

Download

Copy Link

https://www.mongodb.com/try/download/community

A5.9 MongoDB: folder

MongoDB 4.2.11 2008R2Plus SSL (64 bit) Service Customi...

Service Configuration

Specify optional settings to configure MongoDB as a service.

☑ Install MongoD as a Service

◉ Run service as Network Service user

○ Run service as a local or domain user:

Account Domain: .

Account Name: MongoDB

Account Password:

Service Name: MongoDB

Data Directory: D:₩mongodb₩data₩

Log Directory: D:₩mongodb₩log₩

< Back | Next > | Cancel

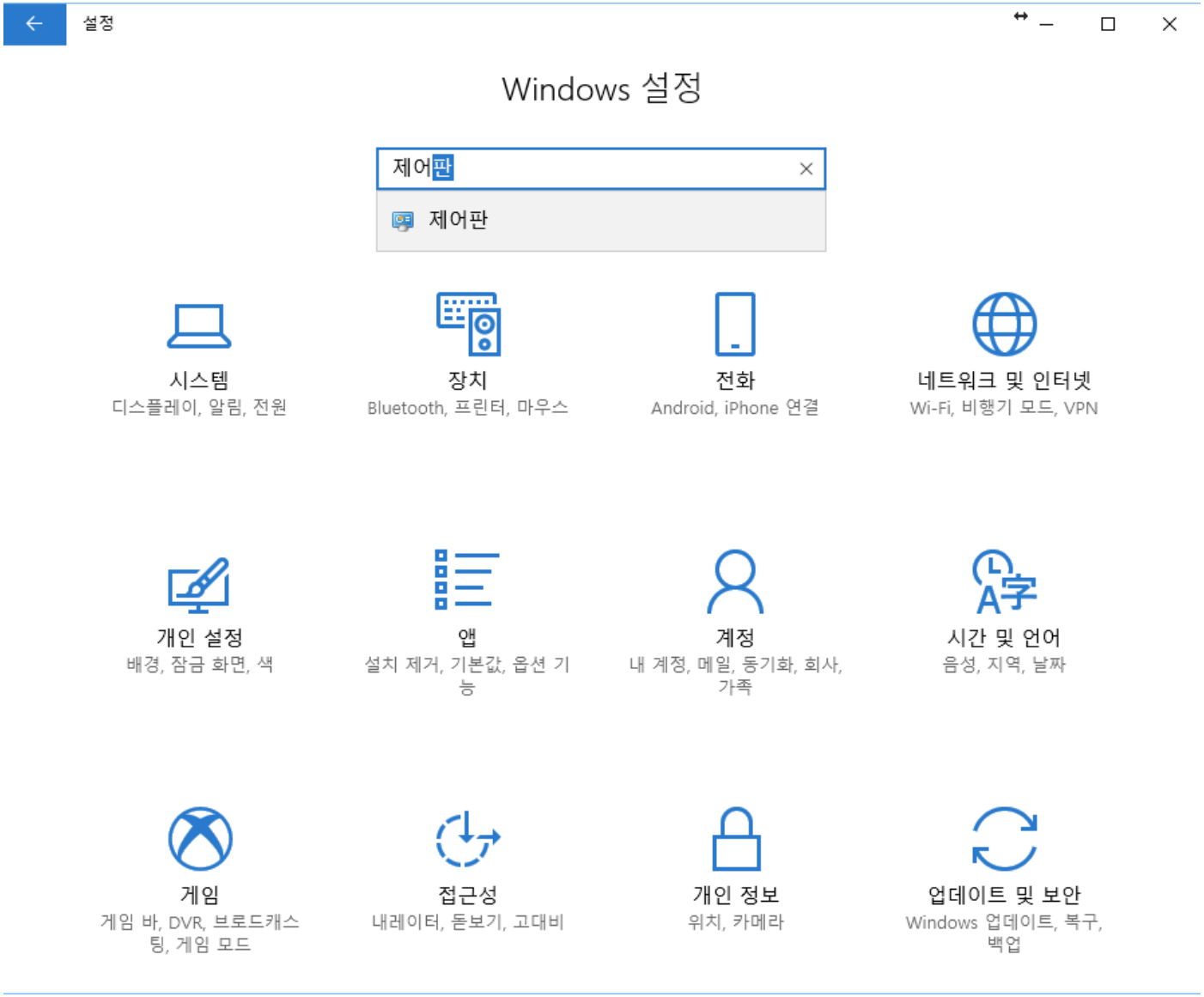
A5.9.1 MongoDB install – 1

윈도우10: 설정 > 시스템 > 정보

[중요] 시스템 환경변수 : **PATH** 에 경로 추가

C:\Program Files\MongoDB\Server\4.2\bin

A5.9.1 MongoDB install – 2



A5.9.1 MongoDB install – 3

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

> 고급 시스템 설정

시스템

› 제어판 › 모든 제어판 항목 › 시스템

제어판 검색

제어판 홈

장치 관리자

원격 설정

시스템 보호

고급 시스템 설정

컴퓨터에 대한 기본 정보 보기

Windows 버전

Windows 10 Home

© 2017 Microsoft Corporation. All rights reserved.

Windows 10

시스템

프로세서:	Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz 3.40 GHz
설치된 메모리(RAM):	32.0GB
시스템 종류:	64비트 운영 체제, x64 기반 프로세서
펜 및 터치:	이 디스플레이에 사용할 수 있는 펜 또는 터치식 입력이 없습니다.

컴퓨터 이름, 도메인 및 작업 그룹 설정

컴퓨터 이름:	yish-HCit
전체 컴퓨터 이름:	yish-HCit
컴퓨터 설명:	
작업 그룹:	WORKGROUP

설정 변경

Windows 정품 인증

Windows 정품 인증을 받았습니다. Microsoft 소프트웨어 사용 조건 읽기

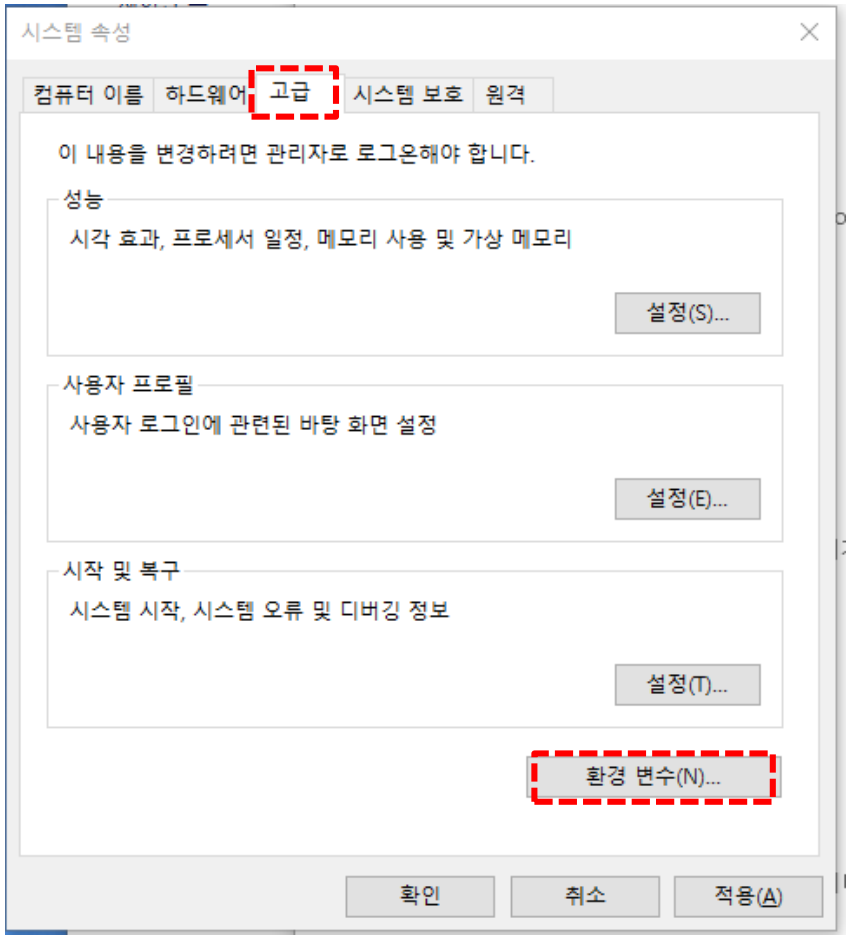
제품 ID: 00325-96080-39821-AAOEM

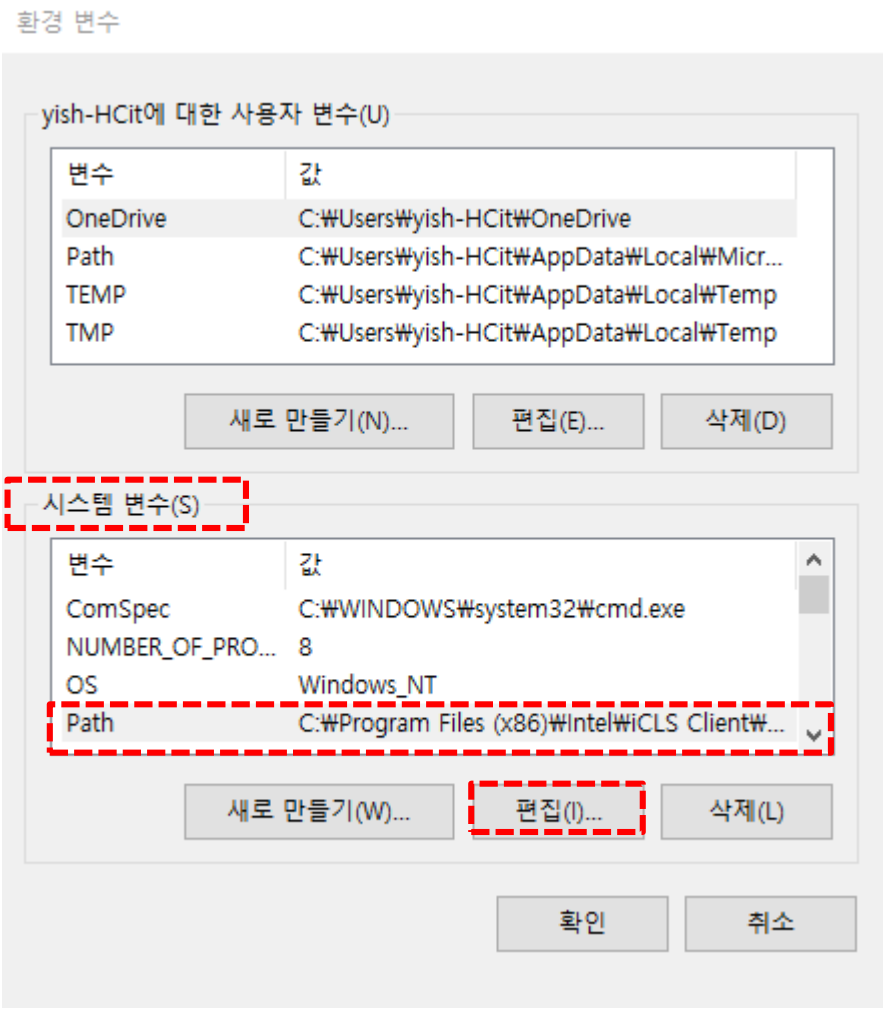
제품 키 변경

참고 항목

보안 및 유지 관리

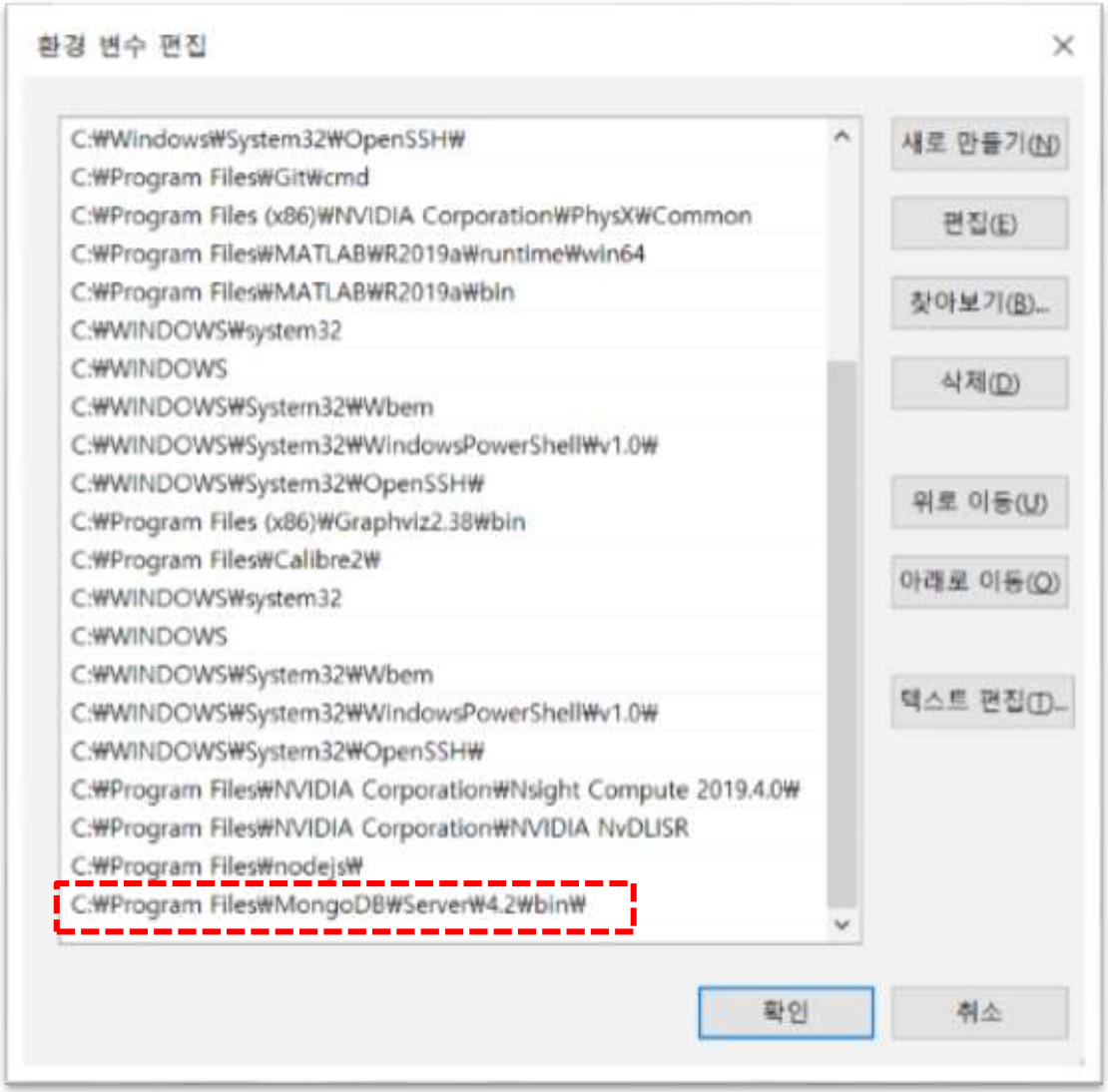
환경 변수 설정


시스템 속성
컴퓨터 이름
하드웨어
고급
시스템 보호
원격
이 내용을 변경하려면 관리자로 로그온해야 합니다.
성능
시각 효과, 프로세서 일정, 메모리 사용 및 가상 메모리
설정(S)...
사용자 프로필
사용자 로그인에 관련된 바탕 화면 설정
설정(E)...
시작 및 복구
시스템 시작, 시스템 오류 및 디버깅 정보
설정(T)...
환경 변수(N)...
확인
취소
적용(A)


환경 변수
yish-HCit에 대한 사용자 변수(U)
변수
값
OneDrive
C:₩Users₩yish-HCit₩OneDrive
Path
C:₩Users₩yish-HCit₩AppData₩Local₩Micr...
TEMP
C:₩Users₩yish-HCit₩AppData₩Local₩Temp
TMP
C:₩Users₩yish-HCit₩AppData₩Local₩Temp
새로 만들기(N)...
편집(E)...
삭제(D)
시스템 변수(S)
변수
값
ComSpec
C:₩WINDOWS₩system32₩cmd.exe
NUMBER_OF_PRO...
8
OS
Windows_NT
Path
C:₩Program Files (x86)₩Intel₩iCLS Client₩...
새로 만들기(W)...
편집(I)...
삭제(L)
확인
취소

A5.9.1 MongoDB install – 5

환경 변수 추가



A5.9.2 MongoDB shell – 1

1. Mongo shell 실행

> mongo

명령 프롬프트 - mongo

```
D:₩mongodb>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mon
godb
Implicit session: session { "id" : UUID("862dc45b-18a7-4f26-8006-b28d58799e64") }
MongoDB server version: 4.2.11-rc1
Server has startup warnings:
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten]
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten] ** WARNING: Access control
is not enabled for the database.
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten] **          Read and write
access to data and configuration is unrestricted.
2020-11-18T09:43:57.885+0900 I  CONTROL  [initandlisten]
---
```

If, Connect failed… → DB 데몬 실행

A5.9.2 MongoDB shell – 2

2. MongoDB 저장소 만들기 → D drive

- **md mongodb**
- **cd mongodb**
- **dir**
- **md data**
- **md log**
- **dir**

명령 프롬프트

```
D:₩mongodb>dir
 D 드라이브의 볼륨: DATA
 볼륨 일련 번호: 82D1-4852

 D:₩mongodb 디렉터리

2020-11-18  오전 09:39    <DIR>          .
2020-11-18  오전 09:39    <DIR>          ..
2020-11-18  오전 10:11    <DIR>          data
2020-11-18  오전 09:39    <DIR>          log
               0개 파일                   0 바이트
               4개 디렉터리  2,332,408,369,152 바이트 남음

D:₩mongodb>
```

사용 PC 환경에 맞게 실행 (특히, 경로 지정)

실습실 환경에 맞춰서 D:에 mongoDB data 폴더 지정

A5.9.2 MongoDB shell – 3

3. Run MongoDB by using mongod.exe

➢ **mongod –dbpath d:\mongodb\data**

명령 프롬프트 - mongod -dbpath d:₩mongodb₩data

```
D:\mongodb>md data

D:\mongodb>mongod -dbpath d:\mongodb\data
2018-01-22T19:27:32.931-0700 I CONTROL  [initandlisten] MongoDB starting : pid=18820 port=27017
 dbpath=d:\mongodb\data 64-bit host=yish-HCit
2018-01-22T19:27:32.931-0700 I CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2
008 R2
2018-01-22T19:27:32.932-0700 I CONTROL  [initandlisten] db version v3.6.2
2018-01-23T11:27:33.699+0900 I COMMAND  [initandlisten] setting featureCompatibilityVersion to
3.6
2018-01-23T11:27:33.706+0900 I STORAGE  [initandlisten] createCollection: local.startup_log wit
h generated UUID: 06b3b7cb-62fe-4be5-a929-2a7478650a9b
2018-01-23T11:27:34.211+0900 I FTDC     [initandlisten] Initializing full-time diagnostic data
capture with directory 'd:/mongodb/data/diagnostic.data'
2018-01-23T11:27:34.215+0900 I NETWORK  [initandlisten] waiting for connections on port 27017
```

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

명령 프롬프트 - mongo

```
D:₩mongodb>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mon
godb
Implicit session: session { "id" : UUID("438383b5-267a-466a-b22e-1bb71bf78985") }
MongoDB server version: 4.2.11-rc1
Server has startup warnings:
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten]
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten] ** WARNING: Access control
is not enabled for the database.
2020-11-18T09:43:57.884+0900 I  CONTROL  [initandlisten] **          Read and write
access to data and configuration is unrestricted.
2020-11-18T09:43:57.885+0900 I  CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and di
splay
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessi
ble to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonit
oring()
---

>
```

5. mongo shell :

Run new cmd

mongo

show dbs

use local

show collections

help

명령 프롬프트 - mongo

```
> show dbs
admin   0.000GB
config  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 ite
rate
        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 (aann) & insert one record (document)

use aa00
show collections

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

show collections
→ “user”
show dbs

db.user.find()

명령 프롬프트 - mongo

```
> use aa00
switched to db aa00
> show collections
> db.user.insert({first:"Redwoods", last:"Yi"})
WriteResult({ "nInserted" : 1 })
> show collections
user
> show dbs
aa00    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

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

동적스키마

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

AAnn_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

AAnn_mongo_update.png

A5.9.3 MongoDB shell coding

6. Delete(or remove) DB

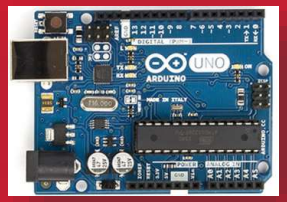
use dbName

db.dropDatabase()

명령 프롬프트 - mongo

```
> use aa00
switched to db aa00
> show collections
user
> db.user.find()
{ "_id" : ObjectId("5a66b44b9f0d55608f5f7582"), "first" : "Redwoods", "last" : "Yi" }
{ "_id" : ObjectId("5a66b5869f0d55608f5f7584"), "first" : "GilDong", "last" : "Hong", "age
" : 21 }
{ "_id" : ObjectId("5a66b6439f0d55608f5f7585"), "firstName" : "Fractal", "last" : "Park" }

>
>
>
>
>
>
>
>
> db.dropDatabase()
{ "dropped" : "aa00", "ok" : 1 }
> show dbs
```



Node.js

+

MongoDB

A5.9.4 MongoDB + Node.js：mongoose

mongoose

elegant mongodb object modeling for node.js

read the docs　discover plugins

Star　Version 5.10.15　Fork

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

```
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/test', {useNe

const Cat = mongoose.model('Cat', { name: String });
```

http://mongoosejs.com/

A5.9.4 MongoDB + Node.js：mongoose

Features Business Explore Marketplace Pricing This repository

Automattic / mongoose

Code Issues 250 Pull requests 2 Projects 0 Wiki Insights

MongoDB object modeling designed to work in an asynchronous environment. http://mon

8,362 commits 14 branches 420 releas

Branch: master New pull request

vkarpov15 Merge branch '4.x'

.github	fix typo
benchmarks	style: remove unused `BlogPost` variable
docs	Merge branch '4.x'

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 (버전 : 5.10.15)

```
D:\Portable\vscode-portable\data\aa2-00\aa2-99-rpt09\wk11_src\Node>npm install --save mongoose
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN cds_dht22@1.0.0 No repository field.

+ mongoose@5.10.15
added 21 packages from 16 contributors and audited 149 packages in 3.331s

4 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities


D:\Portable\vscode-portable\data\aa2-00\aa2-99-rpt09\wk11_src\Node>
```

A5.9.4 MongoDB + Node.js : mongoose

2. node.js project using mongoose (use VSCode)

- cds_dht22 project in vscode
- New file: dbtest.js
- node dbtest.js

```
// dbtest.js
var mongoose = require("mongoose");
mongoose.connect("mongodb://localhost/test", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

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

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

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

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

console.log("Sensor data were saved in MongoDB");
```

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>node dbtest
Sensor data were saved in MongoDB
^C
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>
```

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()

```
> show dbs
aa99    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
> use test
switched to db test
> show collections
sensors
> db.sensors.find()
{ "_id" : ObjectId("5fbcb10c7f7b5c2a7084834f"), "data" : "124", "created" : I
SODate("2020-11-24T07:06:52.096Z"), "__v" : 0 }
{ "_id" : ObjectId("5fbcb10c7f7b5c2a70848350"), "data" : "573", "created" : I
SODate("2020-11-24T07:06:52.100Z"), "__v" : 0 }
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5fbcb10c7f7b5c2a7084834f"),
        "data" : "124",
        "created" : ISODate("2020-11-24T07:06:52.096Z"),
        "__v" : 0
}
{
        "_id" : ObjectId("5fbcb10c7f7b5c2a70848350"),
        "data" : "573",
        "created" : ISODate("2020-11-24T07:06:52.100Z"),
        "__v" : 0
}
```

A5.9.4 MongoDB + Node.js : mongoose

4. dbtest2.js

```
// dbtest2.js
var mongoose = require("mongoose");
mongoose.connect("mongodb://localhost/test2", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});
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;
}
```

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

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

Mongo shell

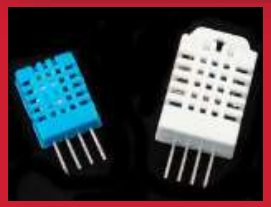
> show dbs

> use test2

> show collections

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

```
> show dbs
aa99    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
test2   0.000GB
> use test2
switched to db test2
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5fbcb31261c2ce07c4bb3401"),
        "data" : "124",
        "created" : "2020-11-24 16:15:30.214",
        "__v" : 0
}
{
        "_id" : ObjectId("5fbcb31261c2ce07c4bb3402"),
        "data" : "573",
        "created" : "2020-11-24 16:15:30.217",
        "__v" : 0
}
>
```



MongoDB from Arduino with node.js & mongoose

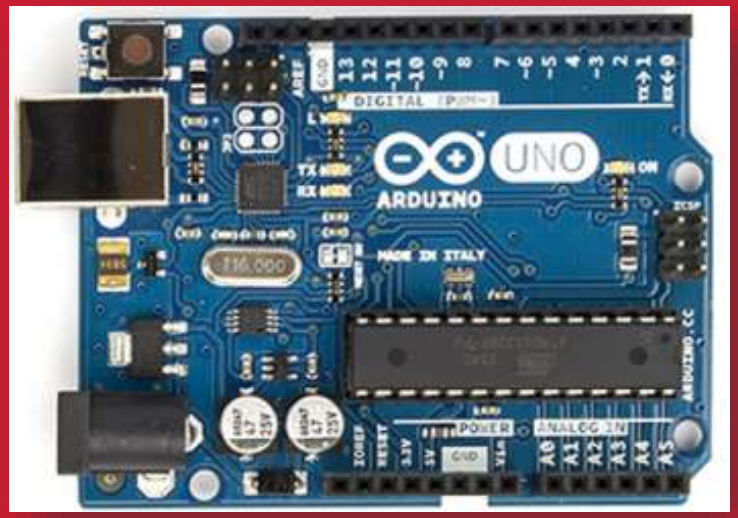
```
> show dbs
aa00     0.000GB
admin    0.000GB
config   0.000GB
iot      0.000GB
iot2     0.000GB
iot3     0.001GB
local    0.000GB
test     0.000GB
test2    0.000GB
>
```

```
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

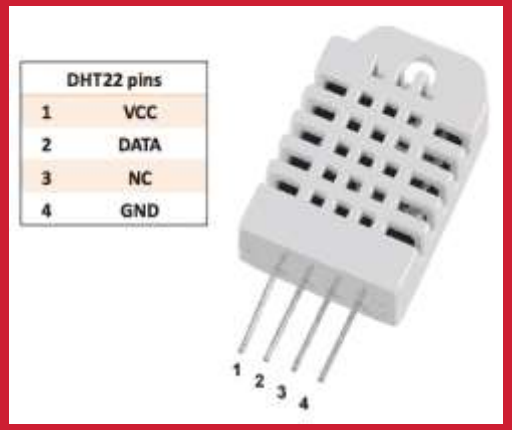
& Node.js

& MongoDB

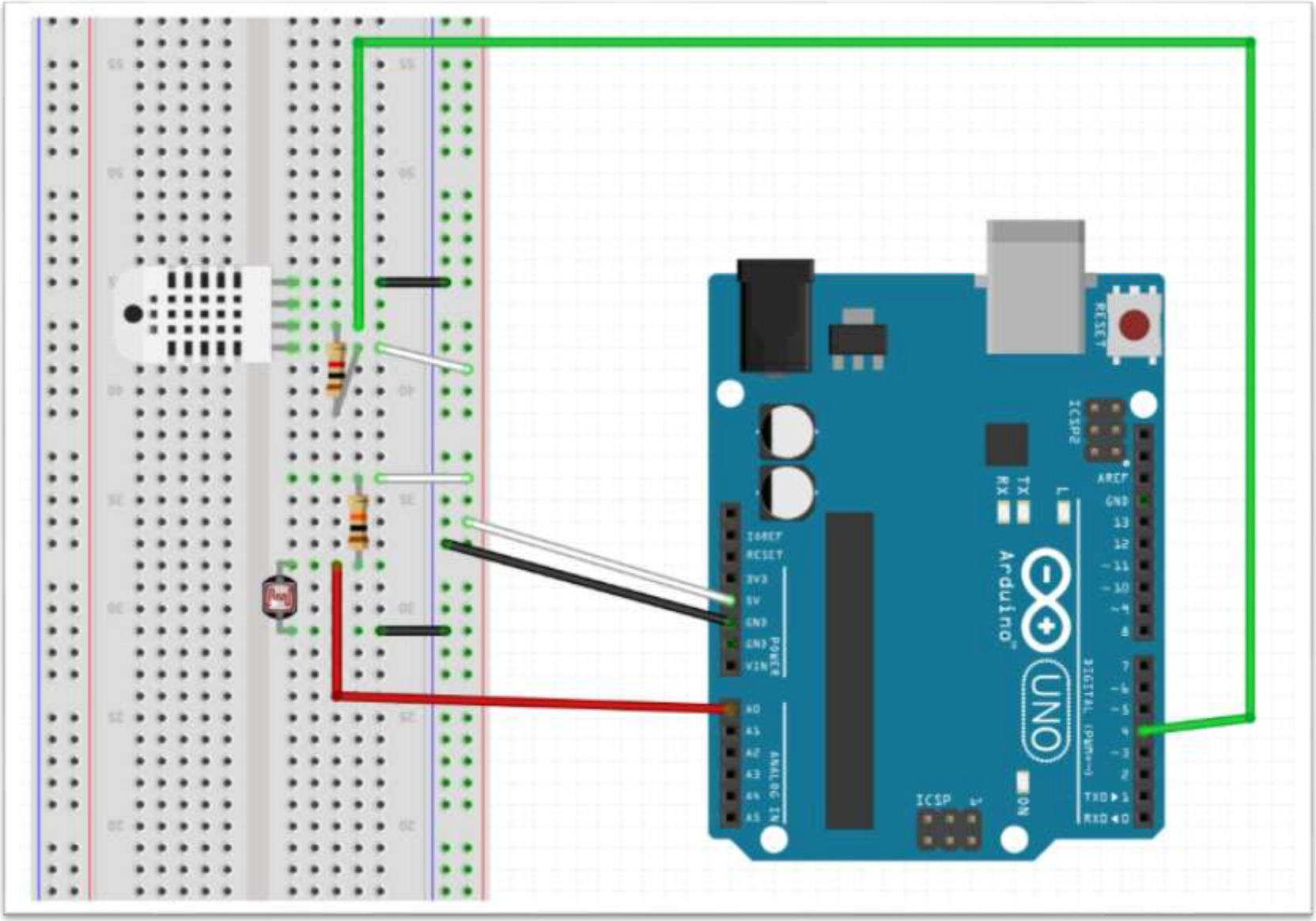


Multi-sensors

DHT22 + CdS

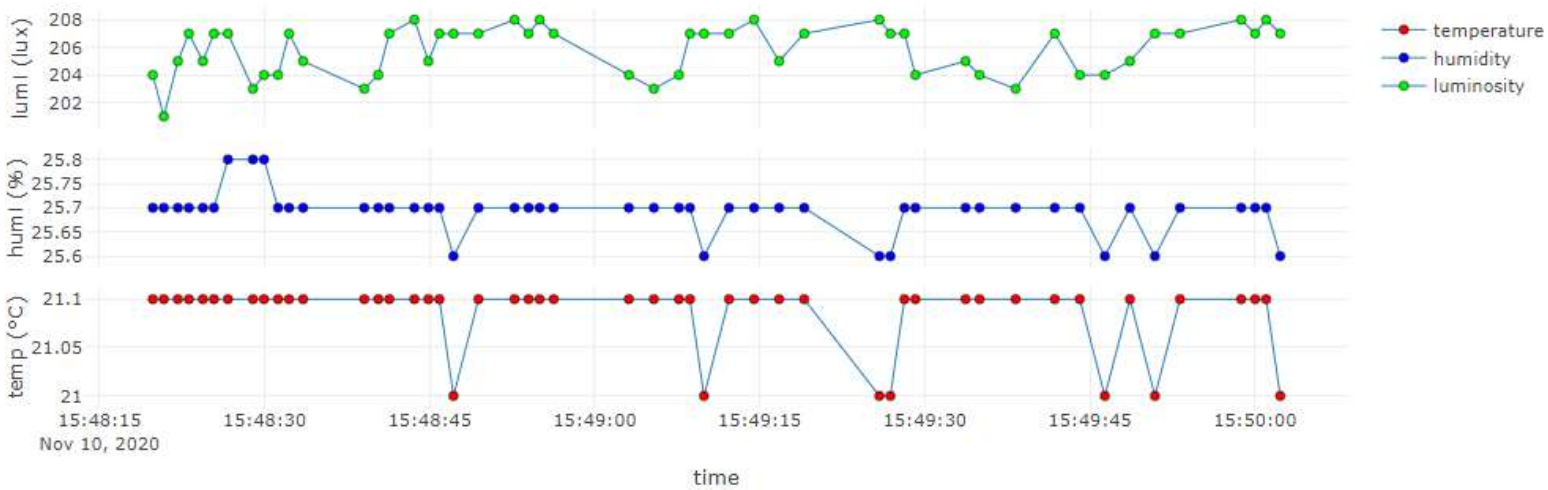


DHT22 + CdS : circuit



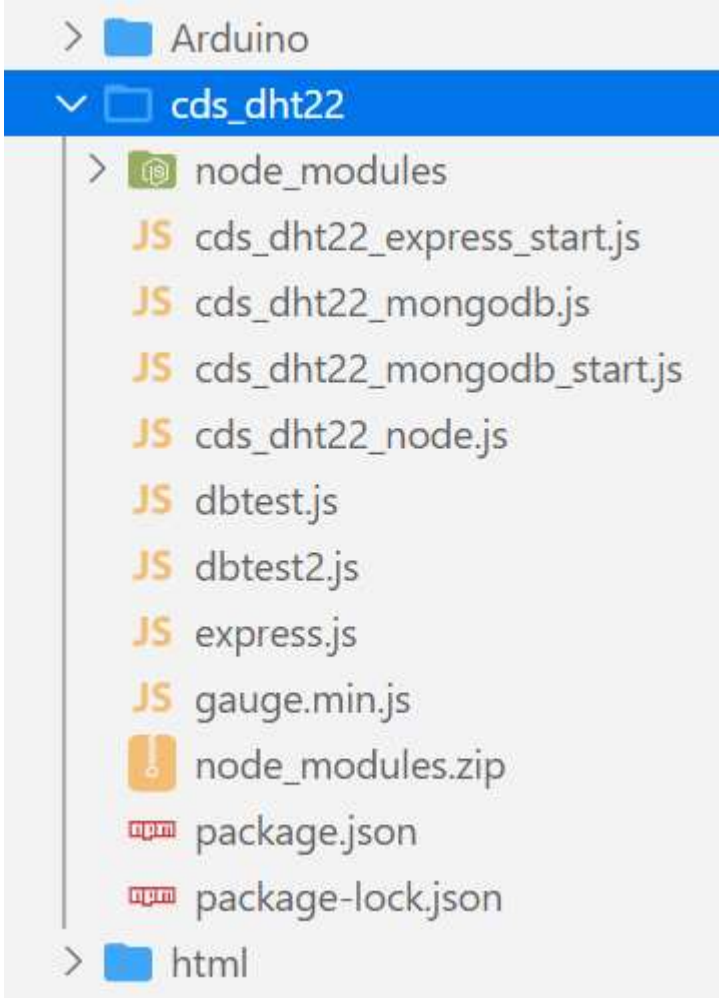
Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300



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

1. 작업 폴더 구조 [2020]



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

2.1 cds_dht22_mongodb.js

```
// cds_dht22_mongodb.js

var serialport = require("serialport");
var portName = "COM3"; // check your COM port!!
var port = process.env.PORT || 3000;

var io = require("socket.io").listen(port);

// MongoDB
var mongoose = require("mongoose");
var Schema = mongoose.Schema;
// MongoDB connection
mongoose.connect("mongodb://localhost:27017/iot", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

```

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

2.1 cds_dht22_mongodb.js

```
18  var db = mongoose.connection;
19  db.on("error", console.error.bind(console, "connection error:"));
20  db.once("open", function callback() {
21    console.log("mongo db connection OK.");
22  });
23  // Schema
24  var iotSchema = new Schema({
25    date: String,
26    temperature: String,
27    humidity: String,
28    luminosity: String,
29  });
30  // Display data on console in the case of saving data.
31  iotSchema.methods.info = function () {
32    var iotInfo = this.date
33      ? "Current date: " +
34        this.date +
35        ", Temp: " +
36        this.temperature +
37        ", Humi: " +
38        this.humidity +
39        ", Lux: " +
40        this.luminosity
41      : "I don't have a date";
42    console.log("iotInfo: " + iotInfo);
43  };
```

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

2.2 cds_dht22_mongodb.js

```
45  const Readline = require("@serialport/parser-readline");
46  // serial port object
47  var sp = new serialport(portName, {
48    baudRate: 9600, // 9600  38400
49    dataBits: 8,
50    parity: "none",
51    stopBits: 1,
52    flowControl: false,
53    parser: new Readline("\r\n"),
54  });
55
56  const parser = sp.pipe(new Readline({ delimiter: "\r\n" }));
57
58  // Read the port data
59  sp.on("open", () => {
60    console.log("serial port open");
61  });
62
63  var readData = ""; // this stores the buffer
64  var temp = "";
65  var humi = "";
66  var lux = "";
67  var mdata = []; // this array stores date and data from multiple sensors
68  var firstcommaidx = 0;
```

2.2 cds_dht22_mongodb.js

```
70  var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
71
72  parser.on("data", function (data) {
73    // call back when data is received
74    readData = data.toString(); // append data to buffer
75    firstcommaidx = readData.indexOf(",");
76
77    // parsing data into signals
78    if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
79      temp = readData.substring(
80        firstcommaidx + 1,
81        readData.indexOf(",", firstcommaidx + 1)
82      );
83      humi = readData.substring(
84        readData.indexOf(",", firstcommaidx + 1) + 1,
85        readData.lastIndexOf(",")
86      );
87      lux = readData.substring(readData.lastIndexOf(",") + 1);
88      readData = "";
```

2.2 cds_dht22_mongodb.js

```
90      dStr = getDateString();
91      mdata[0] = dStr; // Date
92      mdata[1] = temp; // temperature data
93      mdata[2] = humi; // humidity data
94      mdata[3] = lux; // luminosity data
95      var iot = new Sensor({
96        date: dStr,
97        temperature: temp,
98        humidity: humi,
99        luminosity: lux,
100     });
101     // save iot data to MongoDB
102     iot.save(function (err, iot) {
103       if (err) return handleEvent(err);
104       iot.info(); // Display the information of iot data  on console.
105     });
106     io.sockets.emit("message", mdata); // send data to all clients
107   } else {
108     // error
109     console.log(readData);
110   }
111 });
```

2.4 cds_dht22_mongodb.js

```
113  io.sockets.on("connection", function (socket) {
114    // If socket.io receives message from the client browser then
115    // this call back will be executed.
116    socket.on("message", function (msg) {
117      console.log(msg);
118    });
119    // If a web browser disconnects from Socket.IO then this callback
120    socket.on("disconnect", function () {
121      console.log("disconnected");
122    });
123  });
124
125  // helper function to get a nicely formatted date string
126  function getDateString() {
127    var time = new Date().getTime();
128    // 32400000 is (GMT+9 Korea, GimHae)
129    // for your timezone just multiply +/-GMT by 3600000
130    var datestr = new Date(time + 32400000)
131      .toISOString()
132      .replace(/T/, " ")
133      .replace(/Z/, "");
134    return datestr;
135  }
```

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

2.5 node cds_dht22_mongodb.js [vscode 터미널에서 실행]

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2020-11-25 09:27:08.915, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:09.915, Temp: 15.9, Humi: 23.7, Lux: 315
iotInfo: Current date: 2020-11-25 09:27:11.193, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:12.192, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:13.470, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:14.470, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:15.747, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:16.747, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:18.024, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:19.024, Temp: 15.9, Humi: 23.7, Lux: 315
iotInfo: Current date: 2020-11-25 09:27:20.302, Temp: 15.9, Humi: 23.7, Lux: 312
```

3. cds_dht22_mongodb.js → Check documents in Mongo shell

Mongo shell

> show dbs

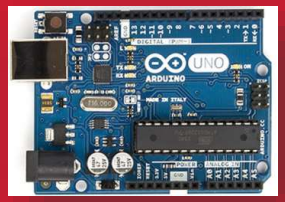
> use iot

> show collections

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

```
> show dbs
aa99    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("5fbda4354fe24d3218cf1400"),
        "date" : "2020-11-25 09:24:21.094",
        "temperature" : "16.1",
        "humidity" : "23.6",
        "luminosity" : "315",
        "__v" : 0
}
{
        "_id" : ObjectId("5fbda4354fe24d3218cf1401"),
        "date" : "2020-11-25 09:24:21.098",
        "temperature" : "16.2",
        "humidity" : "23.6",
        "luminosity" : "312",
        "__v" : 0
}
```

Save as AAnn_iot_mongdb.png



Arduino

& Node.js

& MongoDB

& Express server

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

1.1 Install express server

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

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>npm install --save express

npm WARN cds_dht22@1.0.0 No repository field.

+ express@4.17.1
added 51 packages from 33 contributors and audited 200 packages in 3.078s

4 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

1.2 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",
▷ Debug
"scripts": {
  "test": "echo \"Error: no test specifi
},
"author": "aa00",
"license": "MIT",
"dependencies": {
  "express": "^4.17.1",
  "mongoose": "^5.10.15",
  "serialport": "^9.0.1",
  "socket.io": "^2.3.0"
}
```

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");
```

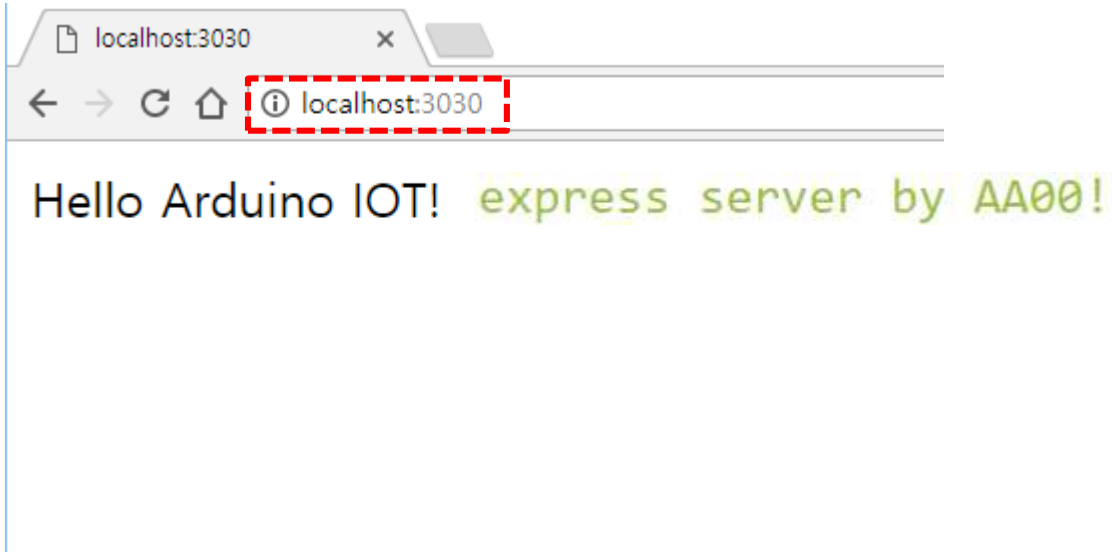
A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.3 cds_dht22_express.js → Run
(cds_dht22__mongodb.js 는 현재 running!)

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_sta
rt\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2020-11-25 10:16:06.207, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:08.484, Temp: 19
.4, Humi: 24.3, Lux: 213
iotInfo: Current date: 2020-11-25 10:16:10.757, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:13.034, Temp: 19
.4, Humi: 24.3, Lux: 213
iotInfo: Current date: 2020-11-25 10:16:15.312, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:17.585, Temp: 19
.4, Humi: 24.3, Lux: 212
```

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds
_dht22>node cds_dht22_express
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

http://localhost:3030/iot


```
[{"_id":"5fbdab6ad02de805786af439","date":"2020-11-25
09:55:06.241","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab6cd02de805786af43a","date":"2020-11-25
09:55:08.514","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab6ed02de805786af43b","date":"2020-11-25
09:55:10.791","temperature":"18.9","humidity":"24.6","luminosity":"207","__v":0},
{"_id":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab73d02de805786af43d","date":"2020-11-25
09:55:15.341","temperature":"18.9","humidity":"24.7","luminosity":"208","__v":0},
{"_id":"5fbdab75d02de805786af43e","date":"2020-11-25
09:55:17.619","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab77d02de805786af43f","date":"2020-11-25
09:55:19.896","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7ad02de805786af440","date":"2020-11-25
09:55:22.169","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7cd02de805786af441","date":"2020-11-25
09:55:24.446","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7ed02de805786af442","date":"2020-11-25
09:55:26.723","temperature":"18.9","h
```


Save as
AAnn_iot_mongodb_web.png

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

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

localhost:3030/iot/5fbdab71d02

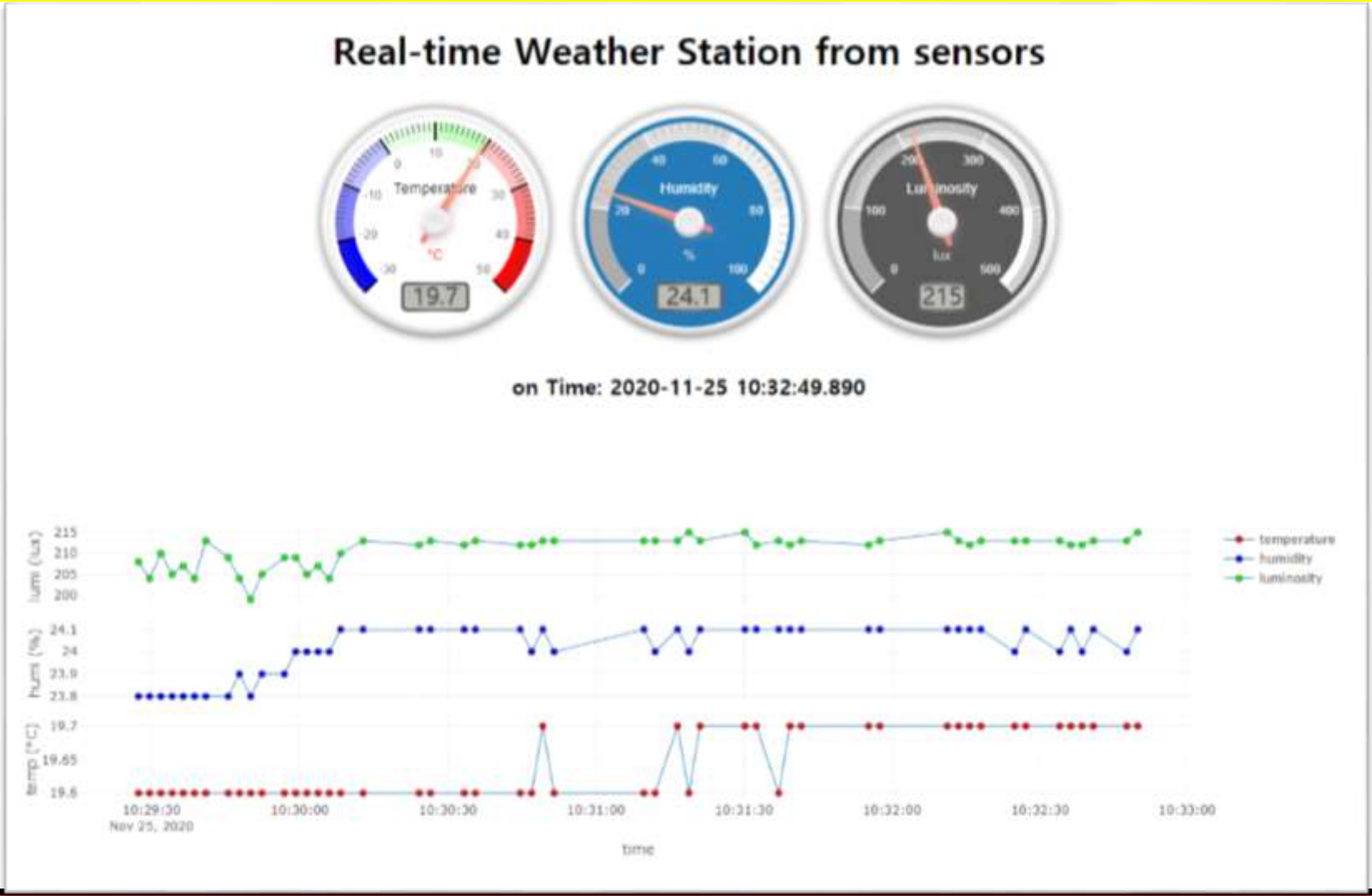
http://localhost:3030/iot/5fbdab71d02de805786af43c


```
{"_id":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0}
```

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

2.7 copy cds_dht22_client.html & gauge.min.js → ./public/ subfolder

http://localhost:3030/client_cds_dht22.html (web root folder)



DHT22 + CdS + Node.js + MongoDB

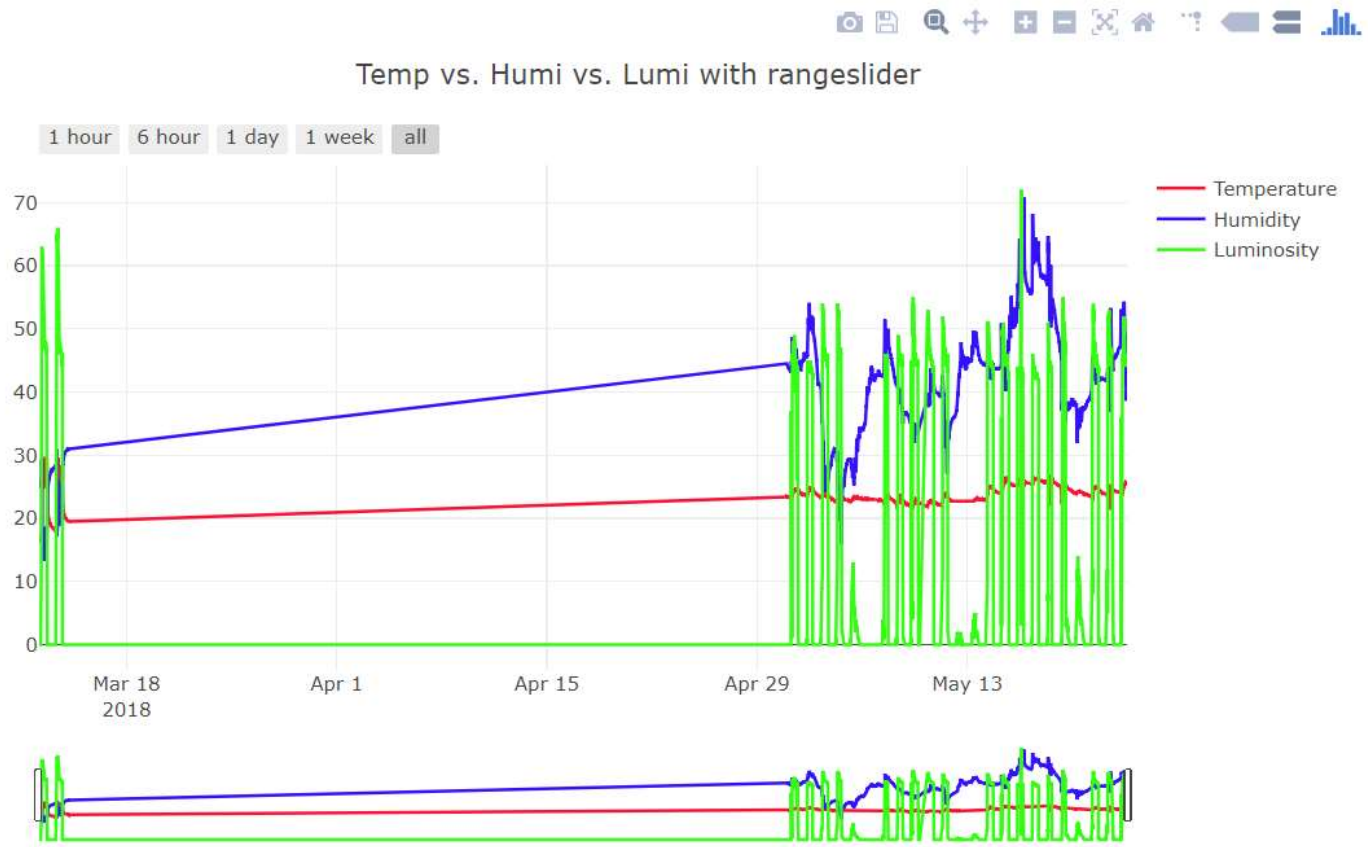
[Next week] Web monitoring

DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring

MongoDB database visualization by AA00

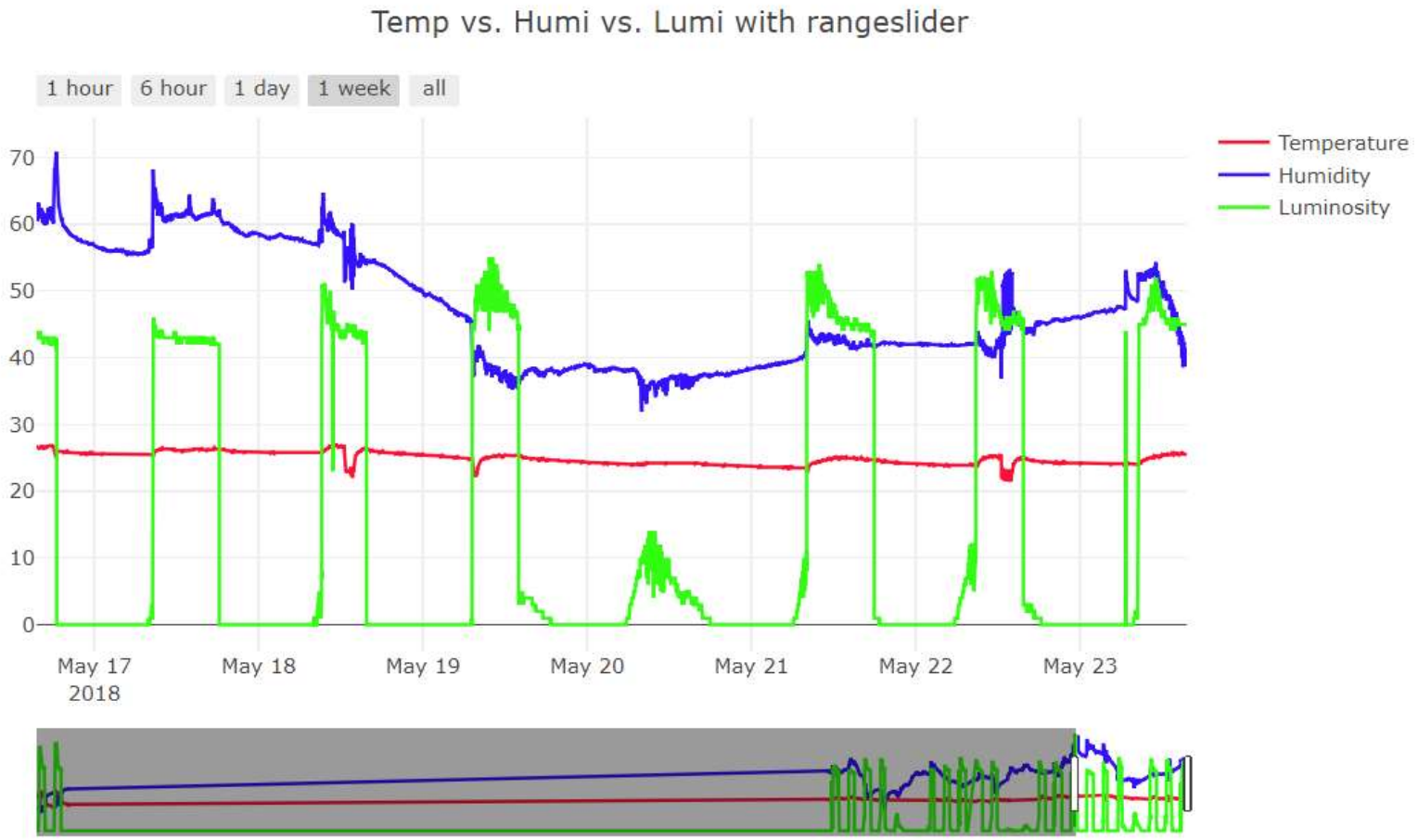
Time series : Multi sensor data

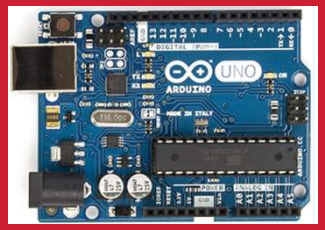


DHT22 + CdS + Node.js + MongoDB

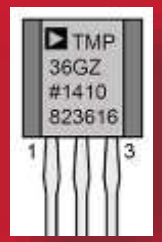
[Next week] Web monitoring

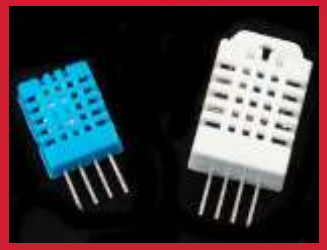
Time series : Multi sensor data





[Practice]





- [wk13]
  - RT Data storaging with MongoDB
  - Multi-sensor circuits (cds-dht22)
  - Complete your project
  - Upload folder: aax-nn-rpt10
  - Use repo “aax-nn” in github

wk13 : Practice : aax-nn-rpt10

- [Target of this week]
  - Complete your works
  - Save your outcomes and upload outputs in github

제출폴더명 : **aax-nn-rpt10**

- 압축할 파일들

① AAnn_mongo_schemas.png

② AAnn_mongo_update.png

③ AAnn_iot_mongodb.png

④ AAnn_iot_mongodb_web.png

⑤ All *.ino

⑥ All *.js

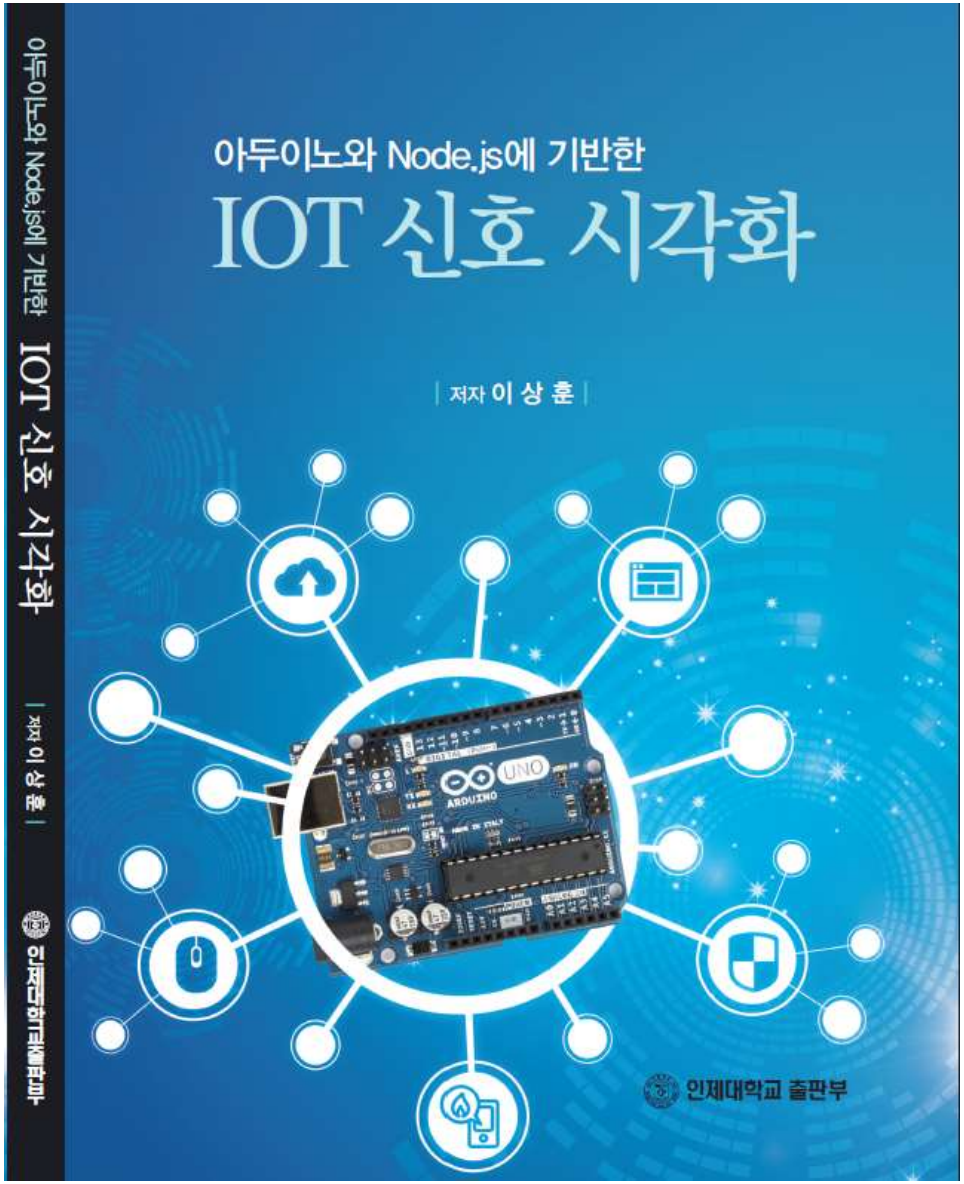
⑦ All *.html

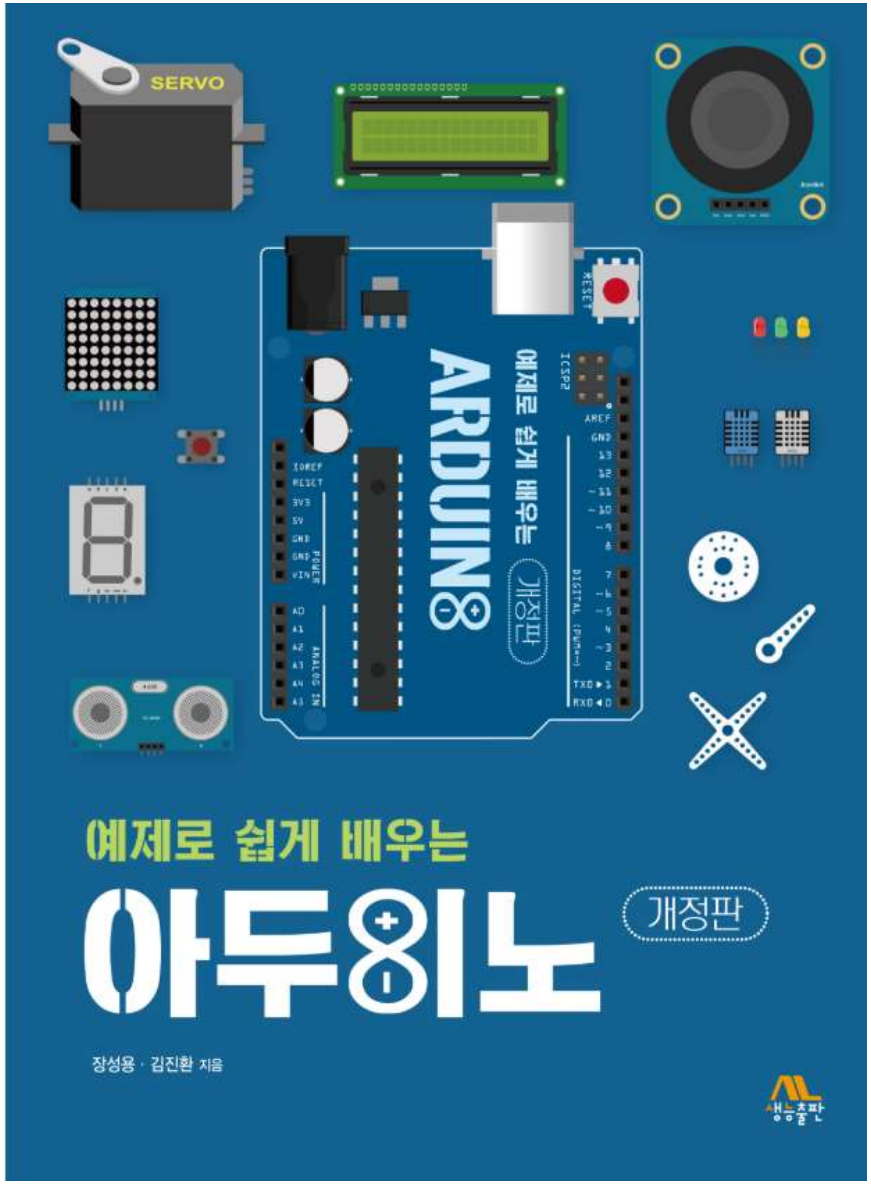
Lecture materials

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

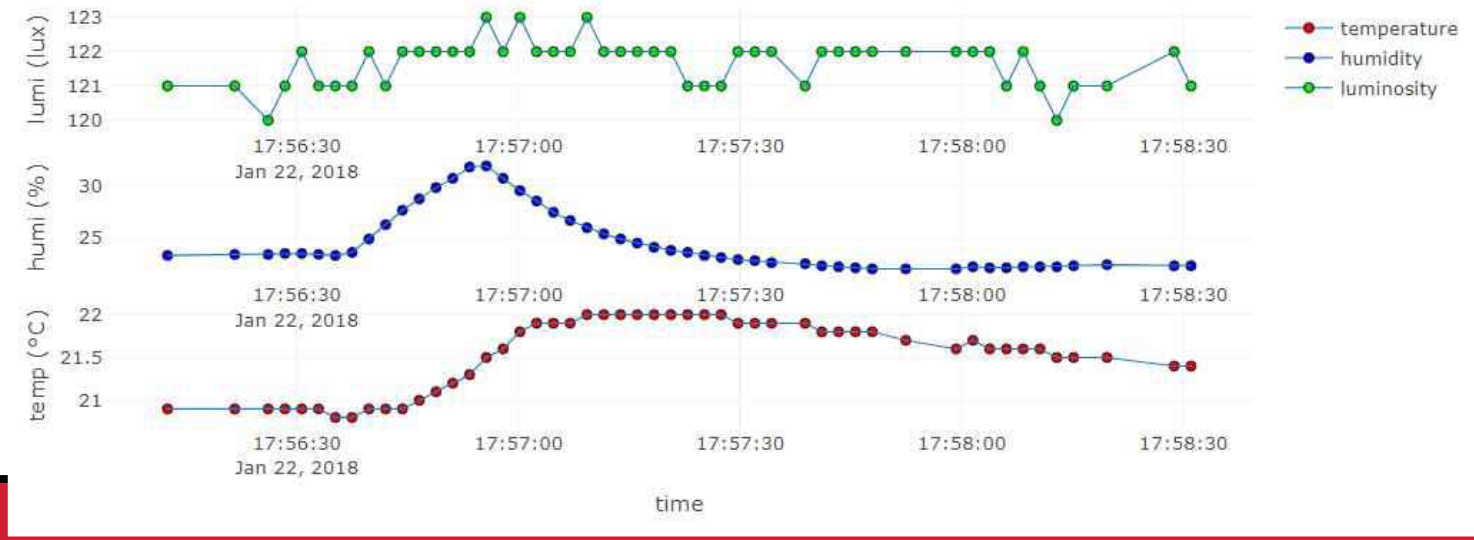

아두이노와 Node.js에 기반한
IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교 출판부
아두이노와 Node.js에 기반한 IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교출판부


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

Target of this class

Real-time Weather Station from sensors

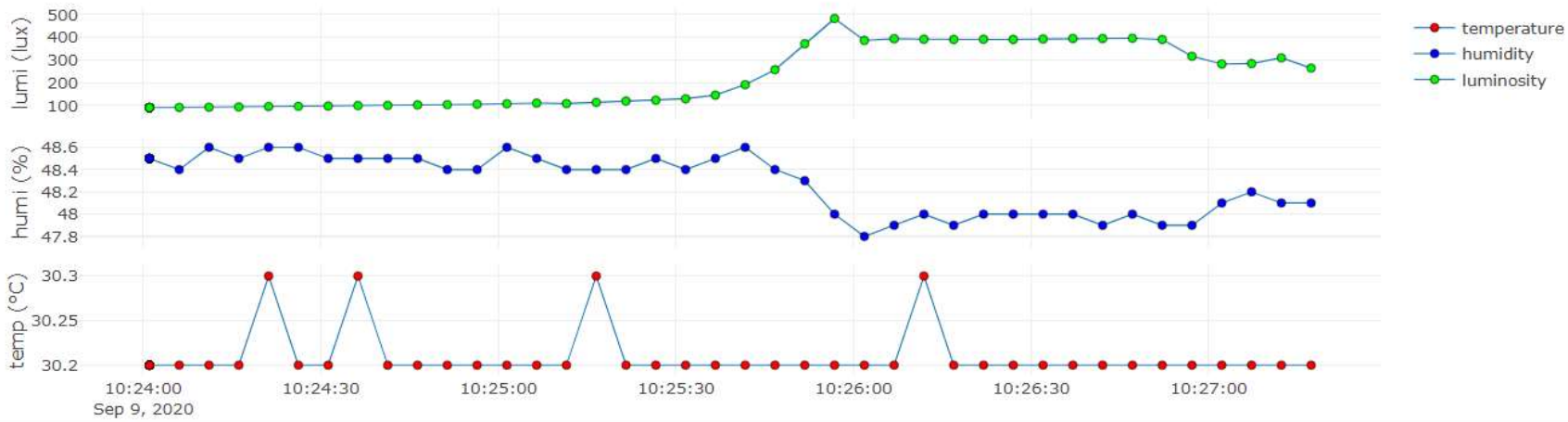
on Time: 2018-01-22 17:58:31.012



Target of this class

Real-time Weather Station from nano 33 BLE sensors

on Time: 2020-09-09 10:27:17.321



Another target of this class