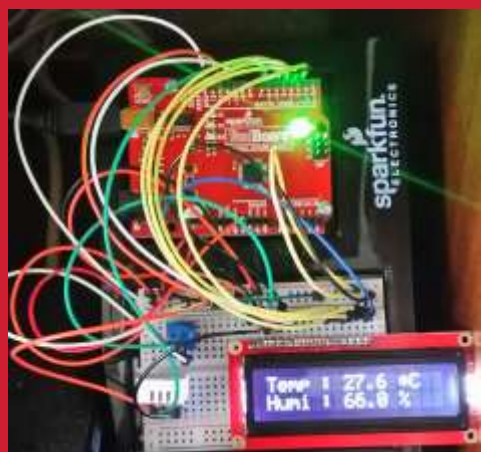




# Arduino-IoT

[wk09]

## cds\_dht22 + node MongoDB-I



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



Drone-IoT-Comsi, INJE University

2<sup>nd</sup> semester, 2023

Email : chaos21c@gmail.com



# My ID

## ID를 확인하고 github에 repo 만들기

ID	성명
AA01	강동하
AA02	고서진
AA03	김민재
AA04	김예원
AA05	김주호
AA06	김창욱
AA07	김현서
AA08	박종혁
AA09	서명진
AA10	유동기
AA11	
AA12	이근보
AA13	정호기

위의 id를 이용해서 github에 repo를 만드시오.

**Option:** 아두이노응용 실습 과제 - **AAnn**

**Public,**     **README.md** **check**



# [Practice]

## ◆ [wk10: project]

- RT Data Visualization with node.js
- Multiple data and Usage of gauge.js
- Complete your real-time WEB charts
- Upload folder: aann-rpt10
- Use repo “aann” in github

# wk10 : Project : aann-rpt10

## ◆ [Target of this week]

- Complete your works : **project - cds\_dht22**
- Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt10**

- 제출할 파일들

- ① **AAnn\_cds\_dht22\_data.png**
- ② **AAnn\_signals\_cds\_dht22.html**
- ③ **AAnn\_cds\_dht22.html**
- ④ **AAnn\_cds\_dht22.png**
- ⑤ **All \*.ino**
- ⑥ **All \*.js**
- ⑦ **All \*.html**

# Purpose of AA

주요 수업 목표는 다음과 같다.

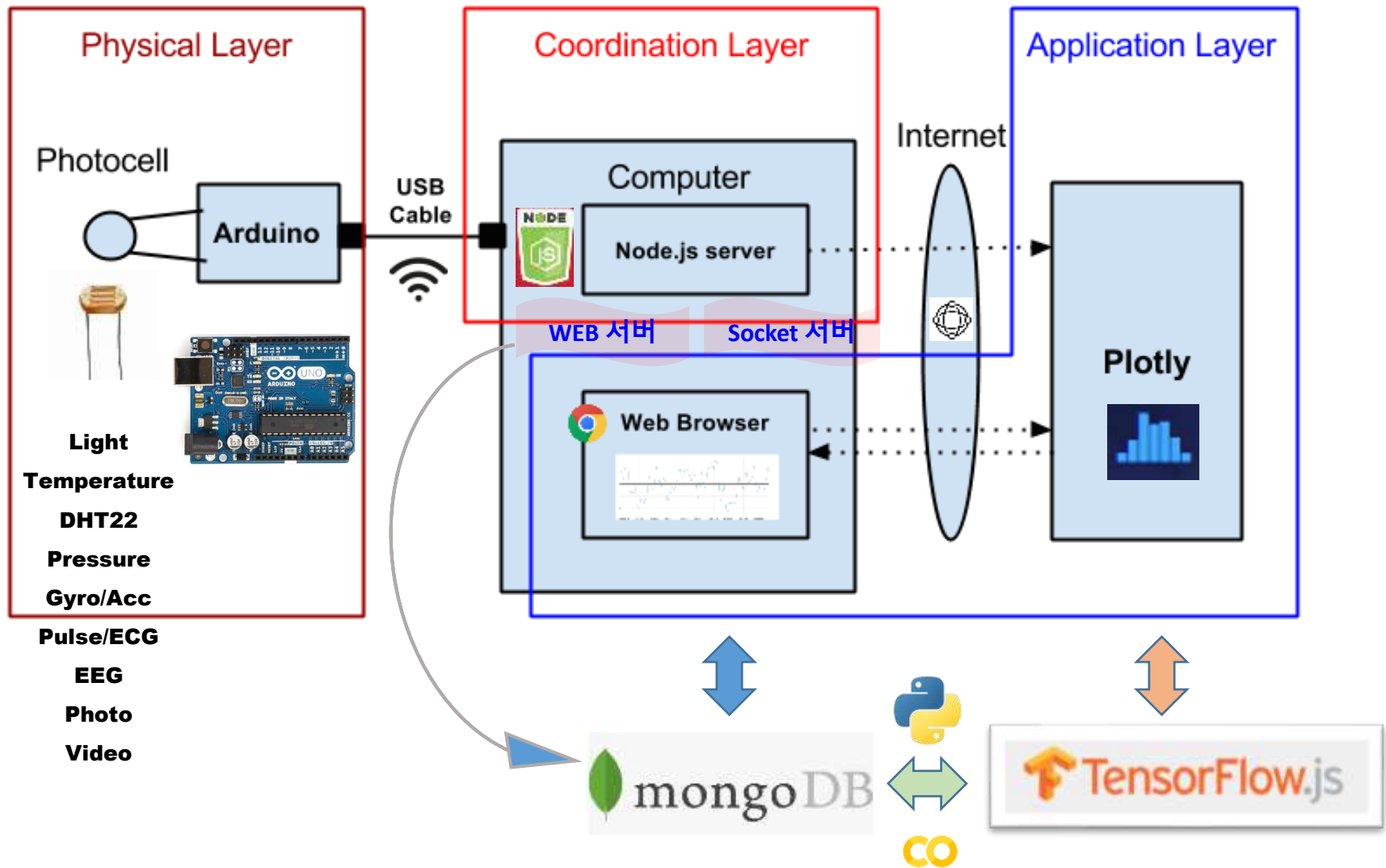
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리



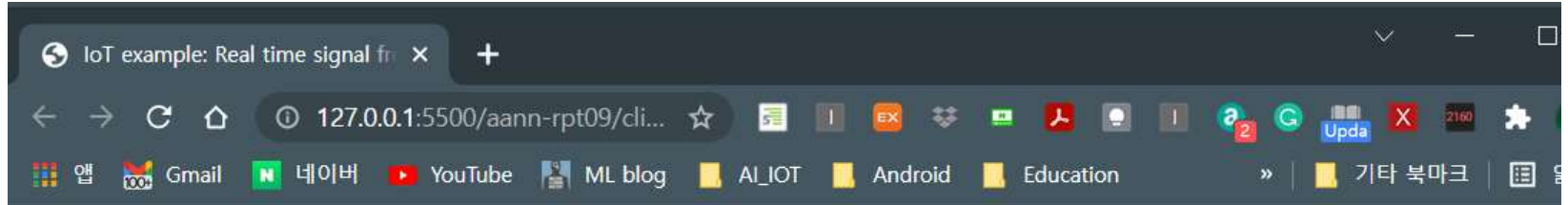
## 4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)



# Layout [H S C]



# on WEB monitoring Arduino data



## IoT Signal from Arduino Weather Station

### Real-time Signals

---

on Time: 2021-10-27 11:54:48.997

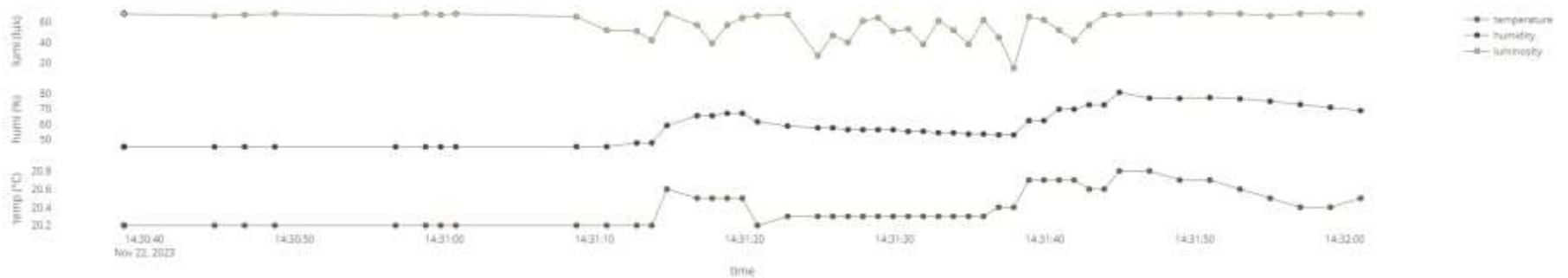
Signals (온도, 습도, 조도) : 23.4, 42.6, 286

---

## Real-time Weather Station from sensors



on Time: 2023-11-22 14:32:01.051







## A5. Introduction to IoT service

**System (Arduino, sDevice, ...)**



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



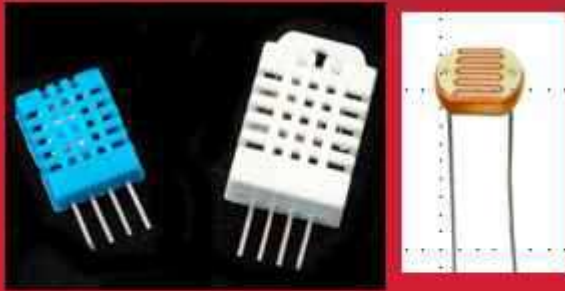
**Visualization & monitoring**



**Data storing & mining**



**Service**



[Goal]

Arduino + Node.js

+ plotly.js

+ MongoDB

→ Data storaging

& visualization

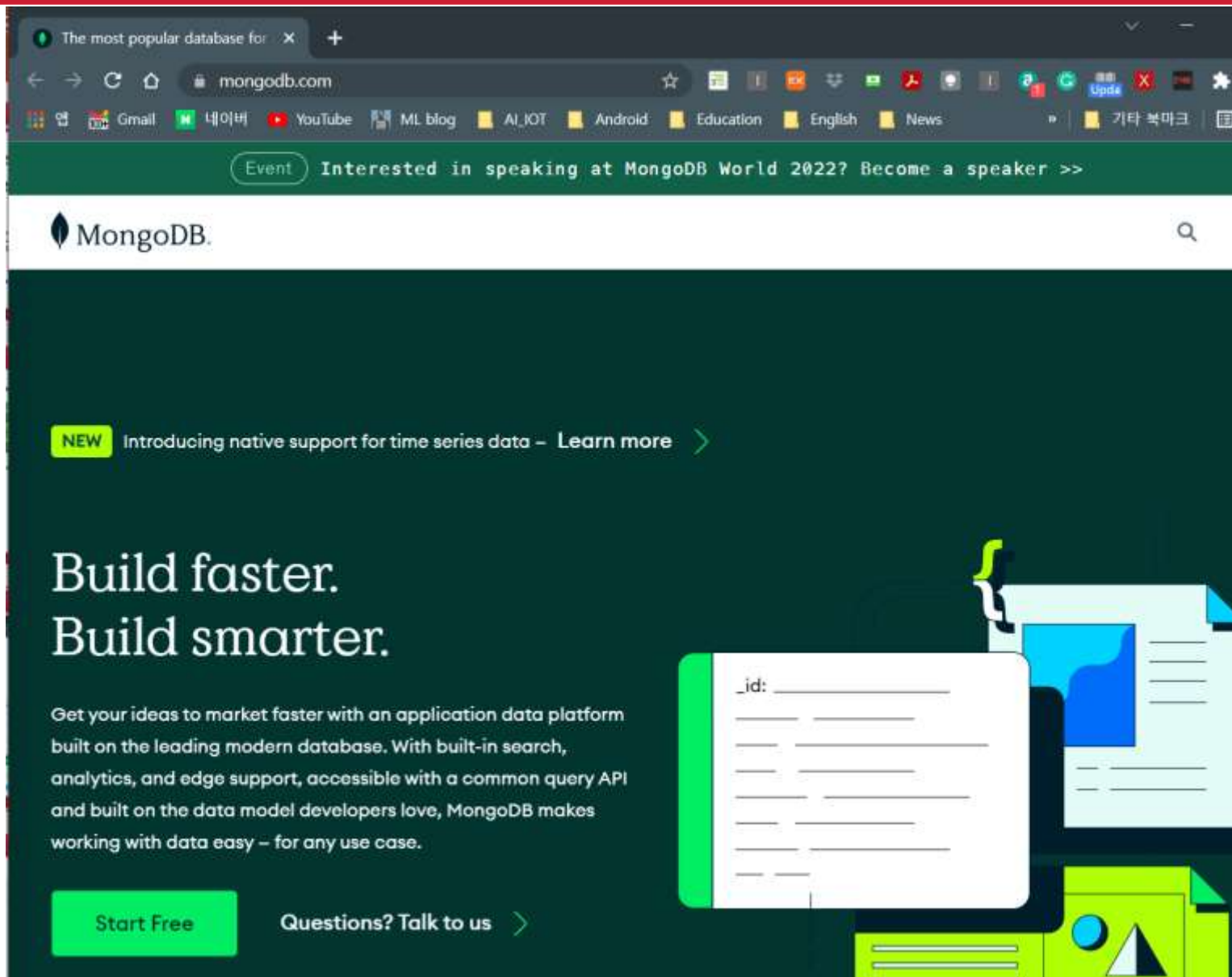


## A5.9 MongoDB





# A5.9 MongoDB



The screenshot shows the MongoDB website homepage in a web browser. The browser's address bar displays 'mongodb.com'. A green banner at the top of the page reads 'Event Interested in speaking at MongoDB World 2022? Become a speaker >>'. Below this is the MongoDB logo and a search icon. The main content area has a dark green background. A yellow 'NEW' badge is next to the text 'Introducing native support for time series data – Learn more >'. The headline 'Build faster. Build smarter.' is prominently displayed. Below the headline, a paragraph states: 'Get your ideas to market faster with an application data platform built on the leading modern database. With built-in search, analytics, and edge support, accessible with a common query API and built on the data model developers love, MongoDB makes working with data easy – for any use case.' At the bottom left, there is a yellow 'Start Free' button. To its right, the text 'Questions? Talk to us >' is shown. On the right side of the page, there is a stylized illustration of a document with a blue line graph and a yellow box containing the text '\_id: \_\_\_\_\_'.



# A5.9 MongoDB



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

## NoSQL?

흔히 **NoSQL**이라고 해서 아, **SQL**이 없는 데이터베이스구나! 라고 생각 할 수도 있겠지만, 진짜 의미는 **Not Only SQL** 입니다. 기존의 **RDBMS**의 한계를 극복하기 위해 만들어진 새로운 형태의 데이터저장소 입니다. **관계형 DB**가 아니므로, **RDMS**처럼 고정된 스키마 및 **JOIN**이 존재하지 않습니다.

## Document?

**Document Oriented** 데이터베이스라는데.. 여기서 말하는 **Document**가 뭘까요? 문서? 이게 그냥 '문서'로 번역해버리면 조금은 애매합니다. 문서라고 하면 보통 워드/엑셀에 사용되는 그런 문서가 떠오르는데요, 그것과는 다릅니다. **Document**는 **RDMS**의 **record**와 비슷한 개념인데요, 이의 데이터 구조는 한개이상의 **key-value pair**으로 이루어져 있습니다. **MongoDB** 샘플 **Document**를 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 는 현재 timestamp, 다음 3bytes 는 machine id, 다음 2bytes 는 MongoDB 서버의 프로세스id, 마지막 3bytes 는 순차번호입니다 추가될때마다 값이 높아진다는거지요.

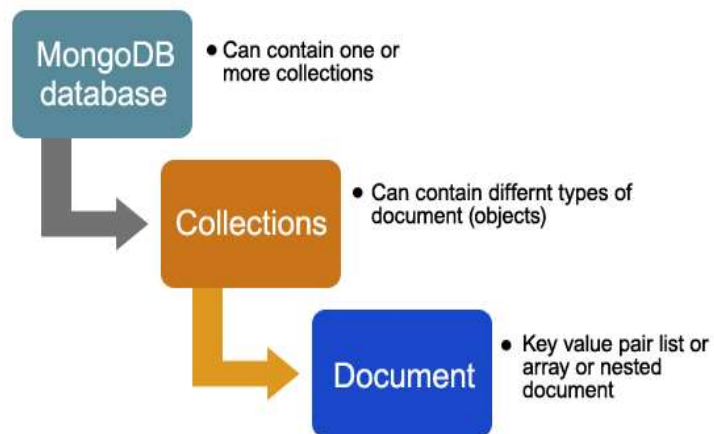
Document는 동적(dynamic)의 schema 를 갖고있습니다. 같은 Collection 안에 있는 Document 끼리 다른 schema 를 갖고 있을 수 있는데요, 쉽게 말하면 서로 다른 데이터 (즉 다른 key) 들을 가지고 있을 수 있습니다.

## Collection?

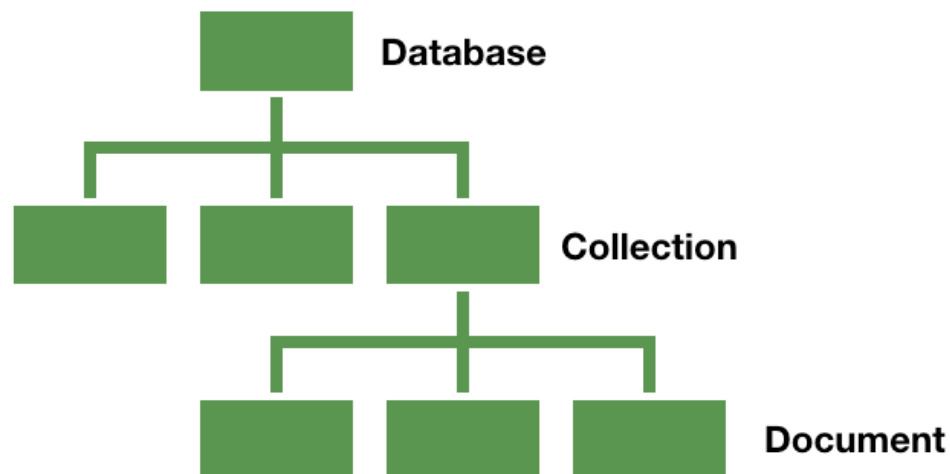
Collection은 MongoDB Document의 그룹입니다. Document들이 Collection 내부에 위치하고 있습니다. RDMS의 table 과 비슷한 개념이지만 RDMS와 달리 schema 를 따로 가지고 있지않습니다. Document 부분설명에 나와있듯이 각 Document들이 동적인 schema 를 가지고 있으니까요

## Database?

Database는 Collection들의 물리적인 컨테이너입니다. 각 Database는 파일시스템에 여러파일들로 저장됩니다.



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



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

MongoDB Community Download

mongodb.com/try/download/community

MongoDB

Atlas  
MongoDB as a service

On-premises  
MongoDB locally

Tools  
Boost productivity

Mobile & Edge  
Realm Datastore

MongoDB Enterprise Server

**MongoDB Community Server**

The Community version of our distributed database offers a flexible document data model along with support for ad-hoc queries, secondary indexing, and real-time aggregations to provide powerful ways to access and analyze your data.

The database is also offered as a fully-managed service with MongoDB Atlas. Get access to advanced functionality such as auto-scaling, serverless instances (in preview), full-text search, and data distribution across regions

Available Downloads

Version: 4.2.17

Platform: Windows

Package: msi

**Download** Copy Link

Current releases & packages

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



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:

Account Password:

Service Name:

Data Directory:

Log Directory:

< Back    Next >    Cancel

## A5.9.1 MongoDB install - 1



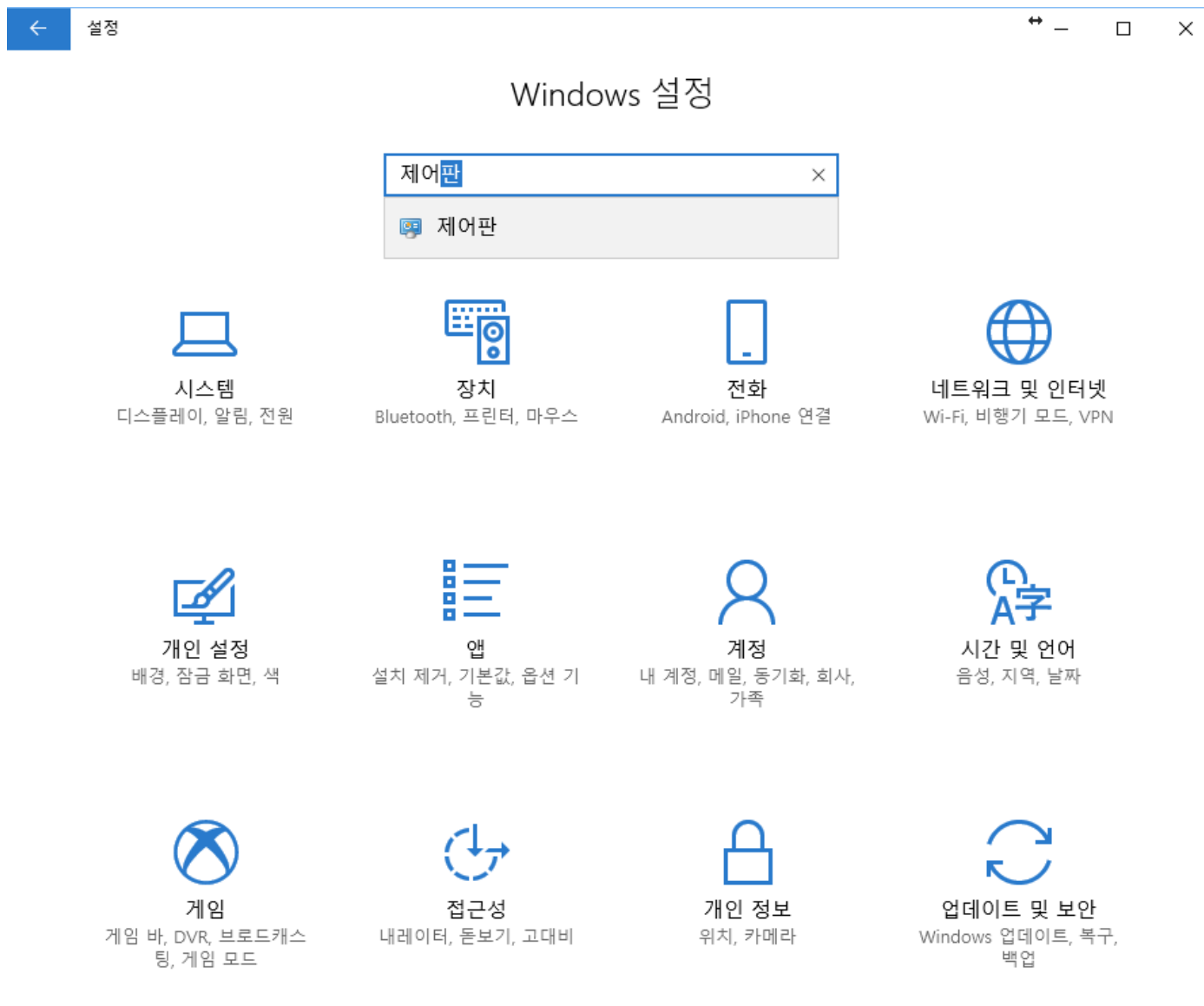
윈도우10: 설정 > 시스템 > 정보

[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

← → ▾ ▴ [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.

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

### Windows 정품 인증

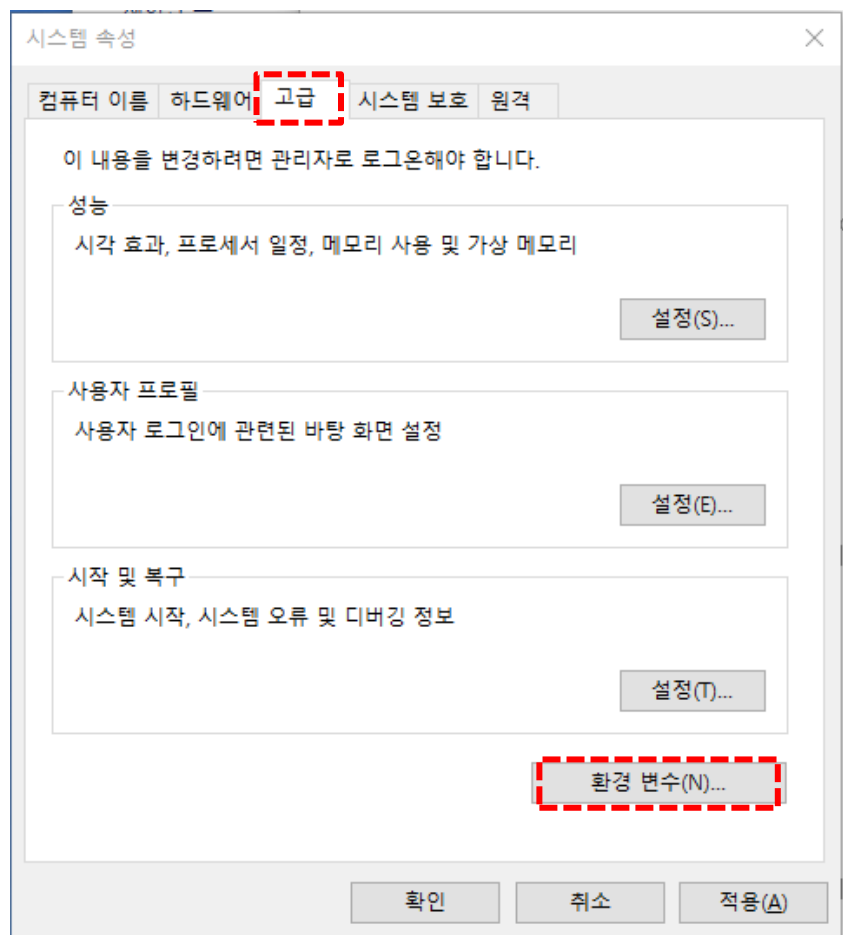
Windows 정품 인증을 받았습니다. [Microsoft 소프트웨어 사용 조건 읽기](#)

제품 ID: 00325-96080-39821-AAOEM

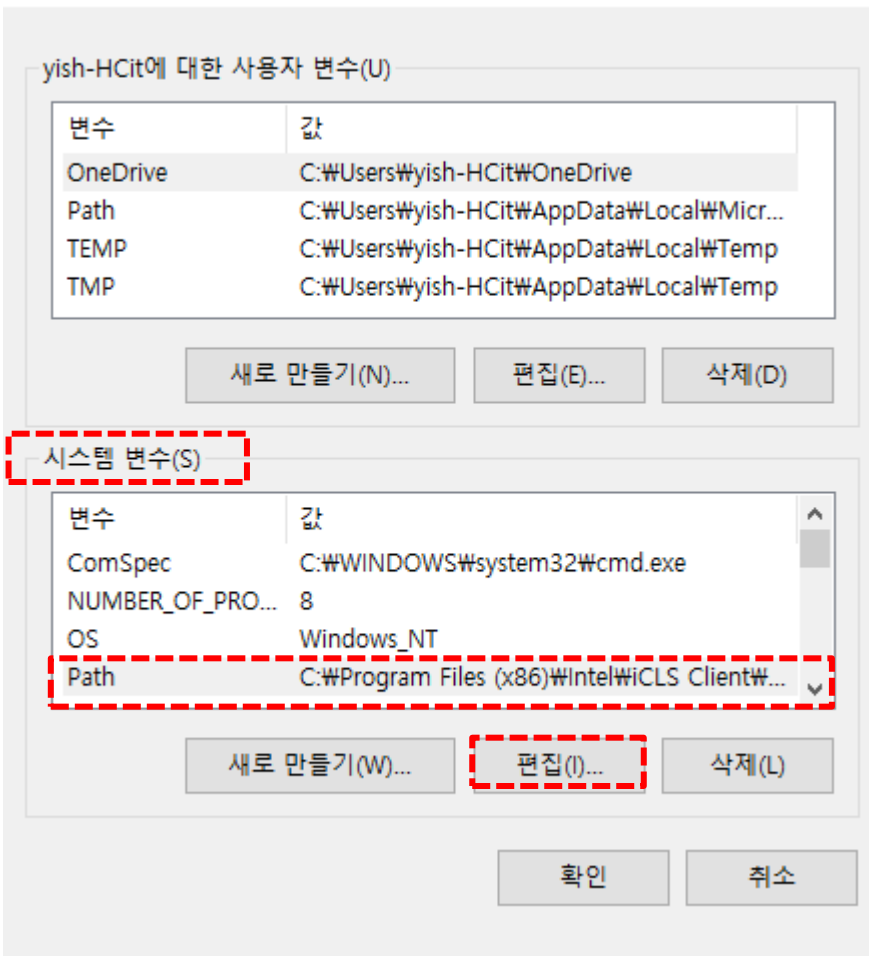
참고 항목

보안 및 유지 관리

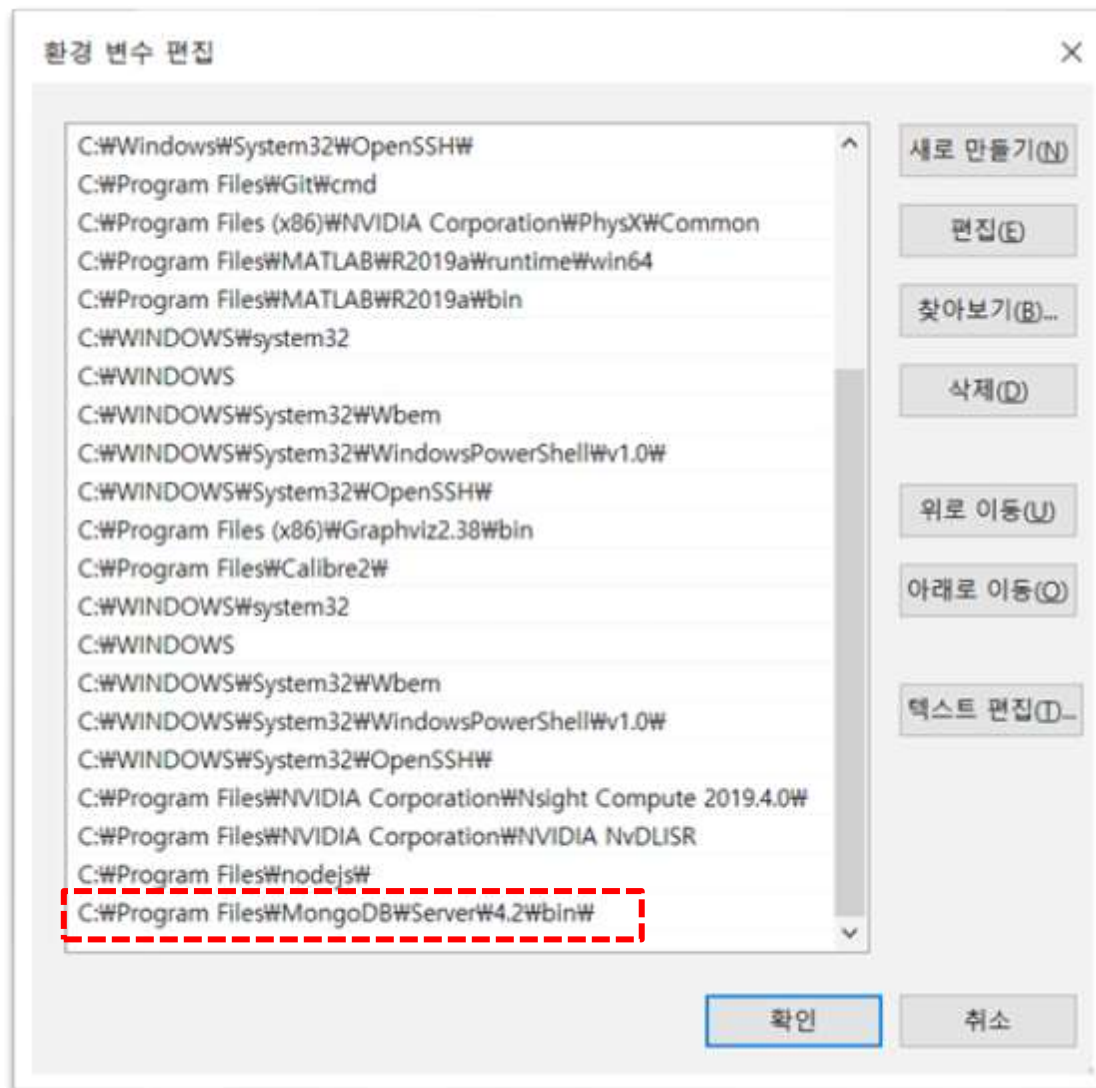
## 환경 변수 설정



### 환경 변수



## 환경 변수 추가





## A5.9.2 MongoDB shell - 1

### 1. Mongo shell 실행

> **mongo (or mongosh)**

명령 프롬프트 - mongo

```
D:\Wmongodb>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
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**

## 2. MongoDB 저장소 만들기 → D drive

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

```

명령 프롬프트
D:\>cd mongodb
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 맞추어서 **D:**에 **mongodb data** 폴더 지정



## 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 or Terminal]

#### ➤ `mongo`

Run in Terminal

`mongo`

```
✓ 제목 없음(작업 영역)
  v aann-rpt11
    > Arduino
    > cds_dht22
    > client_html
    > wk11_src
  v aann-rpt11-Complete
    > arduino
    v cds_dht22
      > node_modules
      > public
      > cds_dht22_express.js
      > cds_dht22_mongodb.js
      > cds_dht22_node.js
      > node_modules.zip
      > package-lock.json
      > package.json
    > aann-rpt12
  v aann-rpt13
    > arduino
  v data_mining
    > iot_data_201005_clean.csv
    > iot_data_201005_raw.csv

D:\aann\aann-rpt11>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("fcfbb1d2-4911-4d98-ba15-bb726ab5cdcd") }
MongoDB server version: 4.2.11-rc1
Server has startup warnings:
2023-11-20T09:30:59.316+0900 I CONTROL [initandlisten]
2023-11-20T09:30:59.316+0900 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2023-11-20T09:30:59.316+0900 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2023-11-20T09:30:59.317+0900 I CONTROL [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible 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.disableFreeMonitoring()
---
```



## A5.9.2 MongoDB shell - 5

### 5. mongo shell :

Run new cmd

mongo

show dbs

use local

show  
collections

help

exit

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()
db.mycoll.help()
sh.help()
rs.help()
help admin
help connect
help keys
help misc
help mr
```

```
help on db methods
help on collection methods
sharding helpers
replica set helpers
administrative help
connecting to a db help
key shortcuts
misc things to know
mapreduce
```

```
show dbs
show collections
show users
show profile
```

```
show database names
show collections in current database
show users in current database
show most recent system.profile entries with time >=
```

lms

```
show logs
show log [name]
```

```
show the accessible logger names
prints out the last segment of log in memory, 'global'
```

is default

```
use <db_name>
db.foo.find()
db.foo.find( { a : 1 } )
it
```

```
set current database
list objects in collection foo
list objects in foo where a == 1
result of the last line evaluated; use to further ite
```

rate

```
DBQuery.shellBatchSize = x
exit
```

```
set default number of items to display on shell
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](#)

## 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"
}
>
```



## 5. update a record with change of schema

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

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"
}
> _
```

Note that it is possible to change schema.  
Save as

[AAnn\\_mongo\\_update.png](#)







# Node.js



+

# MongoDB





## A5.9.4 MongoDB + Node.js : mongoose

Fork me on GitHub

# mongoose

elegant **mongodb** object modeling for **node.js**

read the docs

discover plugins

 Star

Version 6.0.12

 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');


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

<http://mongoosejs.com/>



# A5.9.4 MongoDB + Node.js : mongoose

 [Features](#) [Business](#) [Explore](#) [Marketplace](#) [Pricing](#) [This repository](#)


Automatic / 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 releases](#)

Branch: master ▾ [New pull request](#)

 vkarpov15 Merge branch '4.x'

 <a href="#">.github</a>	fix typo
 <a href="#">benchmarks</a>	style: remove unused 'BlogPost' variable
 <a href="#">docs</a>	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@6.7.1` (버전 : 6.7.1)

```
문제   출력   디버그 콘솔   터미널
cmd + v  [icon] [trash] [up] [down] [close]

D:\aann\aann-rpt10\cds_dht22>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@6.0.12
added 24 packages from 60 contributors, removed 10 packages and audited 68
packages in 2.941s

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

found 0 vulnerabilities
```



## A5.9.4 MongoDB + Node.js : mongoose

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

➤ cds\_dht22 project

➤ New file: dbtest.js

➤ Run:  
node dbtest

localhost ==>  
127.0.0.1

```
aann > aann-rpt10 > cds_dht22 > JS dbtest.js > ...  
1 // dbtest.js  
2 var mongoose = require("mongoose");  
3 mongoose.connect("mongodb://localhost/test", {  
4   useNewUrlParser: true,  
5   useUnifiedTopology: true,  
6 });  
7  
8 var SensorSchema = new mongoose.Schema({  
9   data: String,  
10  created: Date,  
11 });  
12  
13 // data model  
14 var Sensor = mongoose.model("Sensor", SensorSchema);  
15  
16 var sensor1 = new Sensor({ data: "124", created: new Date() });  
17 sensor1.save();  
18  
19 var sensor2 = new Sensor({ data: "573", created: new Date() });  
20 sensor2.save();  
21  
22 console.log("Sensor data were saved in MongoDB");
```

Sensor data were saved in MongoDB



## 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()

```
mongo  cmd  JS dbtest.js
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
test2    0.000GB
> use test
switched to db test
> show collections
sensors
> db.sensor.find()
> db.sensors.find()
{ "_id" : ObjectId("6180e44fbc986ac8a5297195"), "data" : "124", "created" : ISODate("2021-11-02T07:10:07.689Z"), "__v" : 0 }
{ "_id" : ObjectId("6180e44fbc986ac8a5297196"), "data" : "573", "created" : ISODate("2021-11-02T07:10:07.692Z"), "__v" : 0 }
>
```





## A5.9.4 MongoDB + Node.js : mongoose

### 4. dbtest2.js

./ aann > aann-rpt10 > cds\_dht22 > JS dbtest2.js > getDateString

```
1 // dbtest2.js
2 var mongoose = require("mongoose");
3 mongoose.connect("mongodb://localhost/test2", {
4   useNewUrlParser: true,
5   useUnifiedTopology: true,
6 });
7 var SensorSchema = new mongoose.Schema({
8   data: String,
9   created: String,
10 });
11 // data model
12 var Sensor = mongoose.model("Sensor", SensorSchema);
13
14 var sensor1 = new Sensor({ data: "124", created: getDateString() });
15 sensor1.save();
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 // helper function to get a nicely formatted date string
21 function getDateString() {
22   var time = new Date().getTime();
23   // 32400000 is (GMT+9 Korea, GimHae)
24   // for your timezone just multiply +/-GMT by 3600000
25   var datestr = new Date(time + 32400000)
26     .toISOString()
27     .replace(/T/, " ")
28     .replace(/Z/, "");
29   return datestr;
30 }
```





## A5.9.4 MongoDB + Node.js : mongoose

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

#### Mongo shell

> show dbs

> use test2

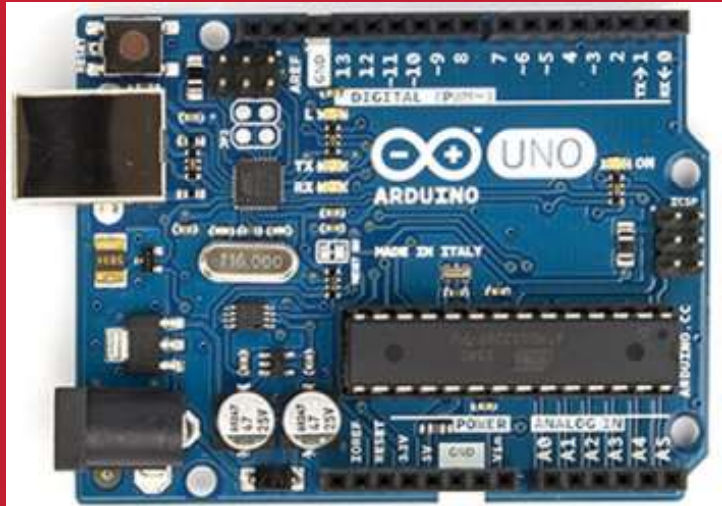
> show collections

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

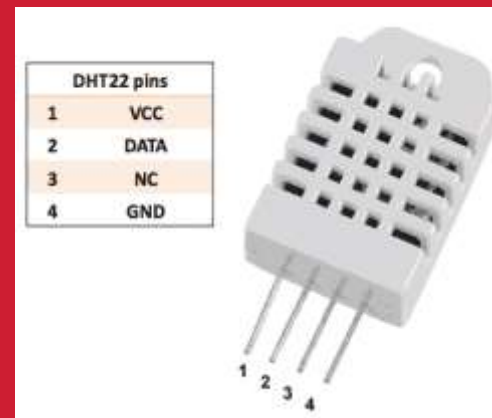
```
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
test2    0.000GB
> use test2
switched to db test2
> db.sensors.find().pretty()
{
  "_id" : ObjectId("6181d4c4501d6bc94cfa6903"),
  "data" : "124",
  "created" : "2021-11-03 09:16:04.448",
  "__v" : 0
}
{
  "_id" : ObjectId("6181d4c4501d6bc94cfa6904"),
  "data" : "573",
  "created" : "2021-11-03 09:16:04.451",
  "__v" : 0
}
>
```



# Arduino & Node.js & MongoDB

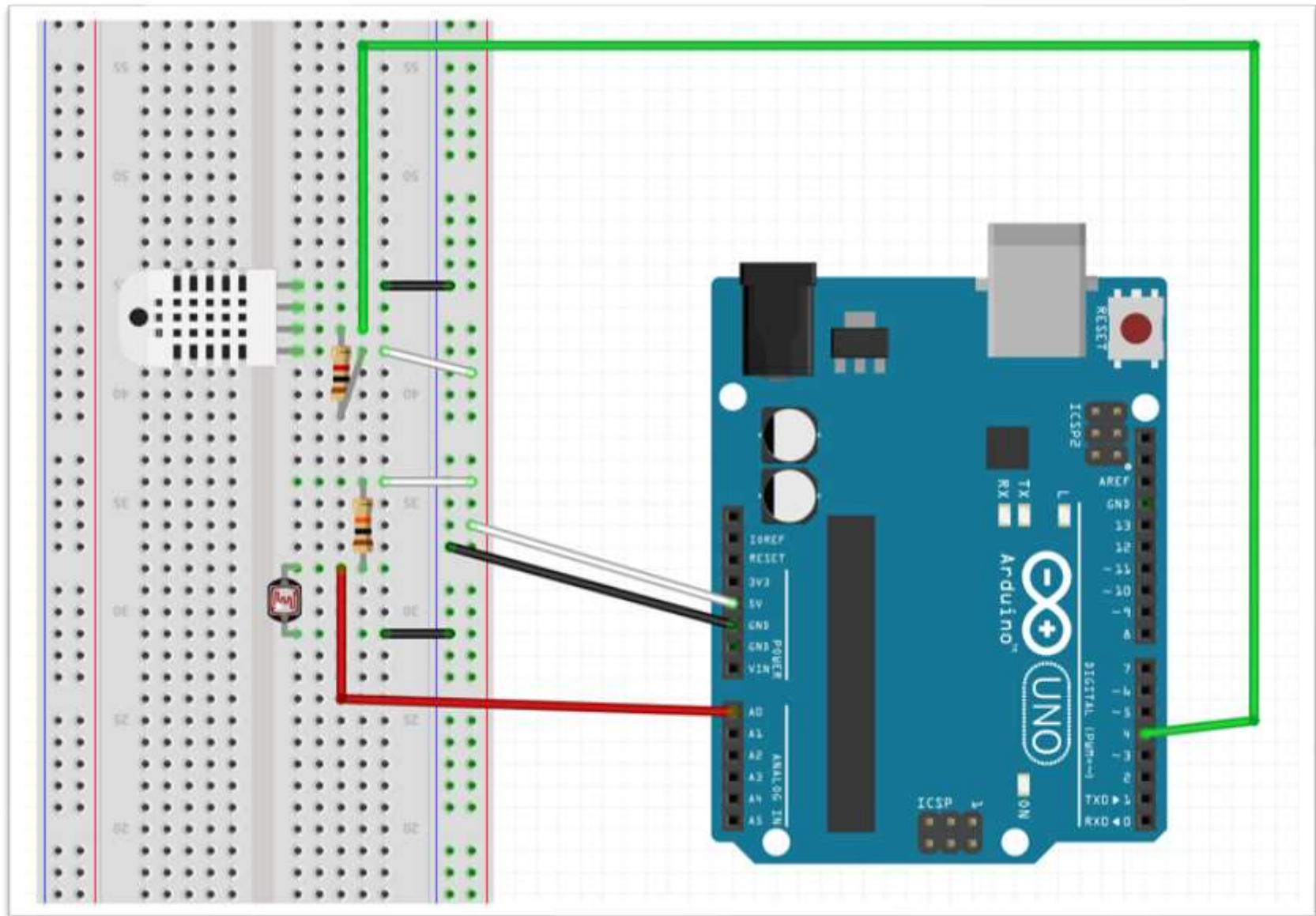


**Multi-sensors**  
**DHT22 + CdS**





# DHT22 + CdS circuit



DHT22[D4] + 1 k $\Omega$ , CdS[A0] + 10 k $\Omega$



# DHT22 + CdS + Node.js

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

node cds\_dht22\_node.js

COM3

21.0,24.7,205  
21.0,24.7,207  
21.0,24.7,205  
21.0,24.7,152  
21.0,24.7,167  
20.9,24.6,166  
20.9,24.6,204  
21.0,24.8,204  
21.0,24.8,152  
21.0,24.8,173  
21.0,24.8,191  
21.0,24.8,203  
21.0,24.8,207  
21.0,24.9,204  
21.0,24.9,204

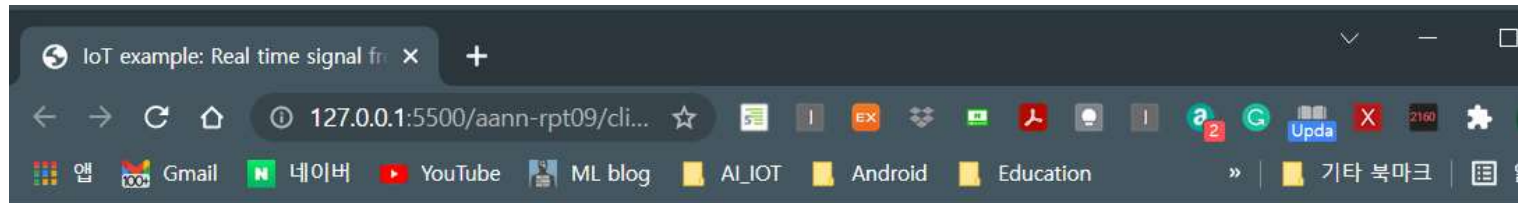
☒ 자동 스크롤 ☐ 타임스탬프



문제 출력 디버그 콘솔 터미널 JUPYTER node

AAnn,2021-10-27 11:53:01.872,23.4,42.6,286  
AAnn,2021-10-27 11:53:02.872,23.4,42.6,286  
AAnn,2021-10-27 11:53:04.150,23.4,42.6,286  
AAnn,2021-10-27 11:53:05.154,23.4,42.6,286  
AAnn,2021-10-27 11:53:06.428,23.4,42.6,286  
AAnn,2021-10-27 11:53:07.431,23.4,42.6,286  
AAnn,2021-10-27 11:53:08.709,23.4,42.6,286  
AAnn,2021-10-27 11:53:09.713,23.4,42.6,286  
AAnn,2021-10-27 11:53:10.987,23.4,42.6,286  
AAnn,2021-10-27 11:53:11.990,23.4,42.6,286  
AAnn,2021-10-27 11:53:13.269,23.4,42.6,284  
AAnn,2021-10-27 11:53:14.268,23.4,42.6,284  
AAnn,2021-10-27 11:53:15.546,23.4,42.6,286  
AAnn,2021-10-27 11:53:16.550,23.4,42.6,284  
AAnn,2021-10-27 11:53:17.824,23.4,42.6,286  
AAnn,2021-10-27 11:53:18.827,23.4,42.6,286

# Arduino data on network socket



## IoT Signal from Arduino Weather Station

### Real-time Signals

---

on Time: 2021-10-27 11:54:48.997

Signals (온도, 습도, 조도) : 23.4, 42.6, 286

---

**Real-time monitoring of signals from Arduino  
CdS + DHT22 circuit**

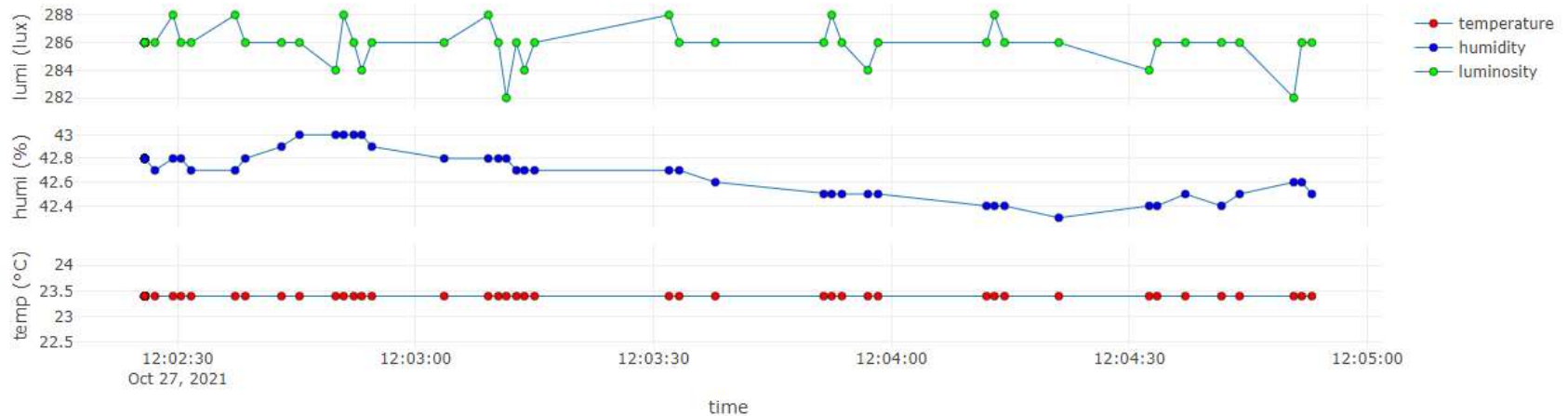


# WEB client : client\_cds\_dht22.html

## Real-time Weather Station from sensors



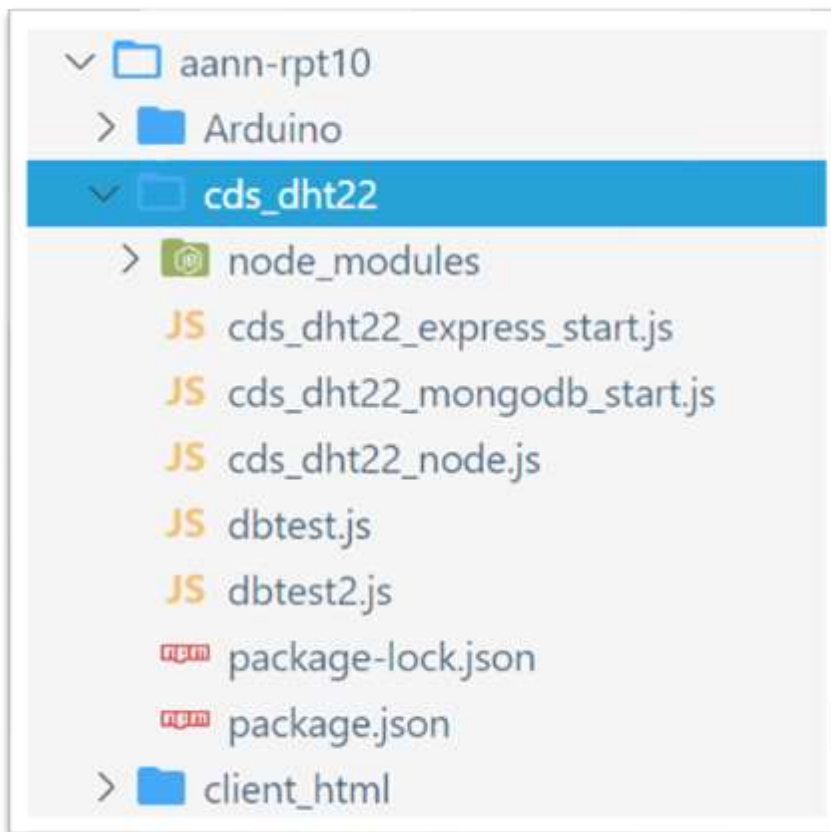
on Time: 2021-10-27 12:04:53.016





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

## 1. 작업 폴더 구조





# 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

```
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

## 2.3 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;
69
70 var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
```



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

## 2.4 cds\_dht22\_mongodb.js – parsing data & save data in MongoDB

```
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   // parsing data into signals
77   if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
78     temp = readData.substring(0, firstcommaidx);
79     humi = readData.substring(
80       firstcommaidx + 1,
81       readData.indexOf(",", firstcommaidx + 1)
82     );
83     lux = readData.substring(readData.lastIndexOf(",") + 1);
84   }
85   readData = "";
86   dStr = getDateString();
87   mdata[0] = dStr; // Date
88   mdata[1] = temp; // temperature data
89   mdata[2] = humi; // humidity data
90   mdata[3] = lux; // luminosity data
91   var iot = new Sensor({
92     date: dStr,
93     temperature: temp,
94     humidity: humi,
95     luminosity: lux,
96   });
97   // save iot data to MongoDB
98   iot.save(function (err, iot) {
99     if (err) return handleEvent(err);
100     iot.info(); // Display the information of iot data on console.
101   });
102   io.sockets.emit("message", mdata); // send data to all clients
103 } else {
104   // error
105   console.log(readData);
106 }
107 }
108 }
109 });
```

// Sensor document 객체



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

## 2.5 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 is called.
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

### 2.6 [Run] node cds\_dht22\_mongodb.js

```
D:\aann\aann-rpt11-Complete\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
```

```
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.083, Temp: .2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.090, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.090, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.088, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.092, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.092, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:20.672, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:21.672, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:22.683, Temp: 20.3, Humi: 45.7, Lux: 63
```



# 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()

```
mongo node JS cds_dht22_mongodb.js
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
> use iot
switched to db iot
> show collections
sensors
> db.sensors.find().pretty()
{
  "_id" : ObjectId("6181eb5338cdbc755b232170"),
  "date" : "2021-11-03 10:52:19.797",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "84",
  "__v" : 0
}
{
  "_id" : ObjectId("6181eb5638cdbc755b232172"),
  "date" : "2021-11-03 10:52:22.074",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "51",
  "__v" : 0
}
{
  "_id" : ObjectId("6181eb5838cdbc755b232174"),
  "date" : "2021-11-03 10:52:24.352",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "81",
  "__v" : 0
}
```

Save as

AAnn\_iot\_mongodb.png



# [Practice]

## ◆ [wk11: mongoDB test]

- Insert documents to test mongodb
- Upload folder: aann-rpt11
- Use repo “aann” in github

# wk11 : Practice : aann-rpt11

## ◆ [Target of this week]

- Complete your works : **mongoDB test.**
- Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt11** 에 아래 파일을 추가

### - 제출할 파일들

- ① **AAnn\_mongo\_schemas.png**
- ② **AAnn\_mongo\_update.png**
- ③ **dbtest.js** (cds\_dht22 folder)
- ④ **dbtest2.js** (cds\_dht22 folder)
- ⑤ **AAnn\_iot\_mongodb.png**



## ● 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

# Target of this class

## Real-time Weather Station from nano 33 BLE sensors



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

