

# Healthcare-IOT

## [wk12]

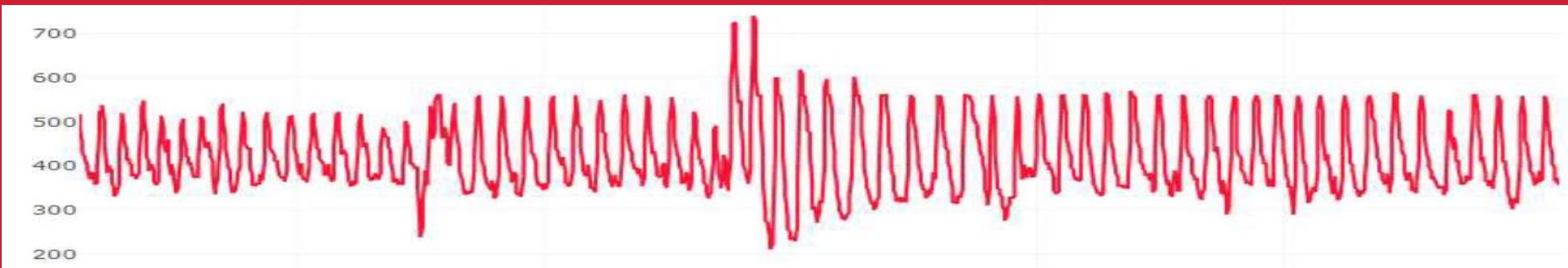
# Data storing using MongoDB I.

Visualization of Healthcare Signals using  
Arduino & Node.js

HCit, INJE University

1<sup>st</sup> semester, 2018

Email : [chaos21c@gmail.com](mailto: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 - 포터블 소프트웨어 설치	
2	강의/실습	Node.js I - Node.js 코드의 기본 구조 - 기초 Node 서버 및 클라이언트	실습확인
3	강의/실습	Node.js II - Node.js Express 서버	실습확인
4	강의/실습/발표	Arduino I - 아날로그 신호 회로 - LCD를 이용한 센서 신호 모니터링	실습확인
5	강의/실습	Arduino II - 단일 센서 회로와 Node.js 연결 - 다중 센서 회로와 Node.js 연결	실습확인
6	강의/실습	프로젝트1 - 생체 센서 회로와 Node.js 연결 - 생체 신호 소개	프로젝트1
7	강의/실습/발표	IOT 데이터 시각화 I (Plotly.js) - 데이터 및 시계열 차트 - 데이터 스트리밍	실습확인
8	시험	중간고사	
9	강의/실습	IOT 데이터 시각화 II (Plotly.js) - 다중 센서 데이터 시각화 - 다중 센서 데이터 스트리밍	실습확인
10	강의/실습/발표	프로젝트II - 생체 센서 데이터 시각화 - 생체 센서 데이터 스트리밍	프로젝트II
11	강의/실습	IOT 데이터 저장과 처리 - MongoDB 설치 및 Mongo shell - MongoDB와 Node.js 연결 및 데이터 저장	실습확인
12	강의/실습	프로젝트III - MongoDB에 IOT 데이터 저장 및 모니터링 - 생체 센서 데이터 저장 및 시각화	프로젝트III
13	강의/실습	IOT 데이터 마이닝 - 아두이노에서 발생된 데이터 관리 - 데이터마이닝 소개	실습확인
14	강의/실습/발표	프로젝트IV - 생체 센서 데이터 관리 - 생체 센서 데이터 마이닝	프로젝트IV
15	시험	기말고사	

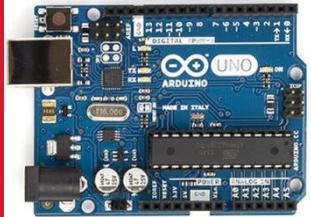


# Purpose of HS

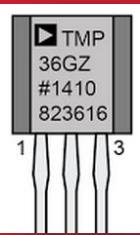
주요 수업 목표는 다음과 같다.

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



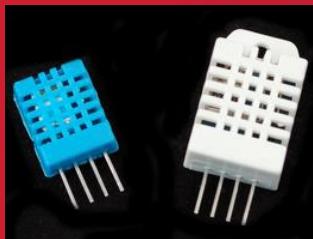


# [Review]



## ◆ [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](mailto:chaos21c@gmail.com)

[ 제목 : **id**, 이름 (수정) ]



# [My working folder – wk11]

The image displays two separate file explorer windows side-by-side, illustrating the structure of a working folder for a combined Arduino and Node.js project.

**Left File Explorer (Arduino Sketches):**

- Path: PC > DATA (D:) > Portable > arduino-1.8.5 > hs00
- Content:
  - hcit
  - HS00\_AnalogRead\_fmap
  - HSnn\_TMP36\_NodeJS
  - HSnn\_TMP36\_start** (highlighted)
  - libraries
  - sketch01\_blink
  - sketch01\_start
  - sketch02\_pwm\_led
  - sketch03\_pwm\_led\_serial
  - sketch04\_pwm\_3\_leds
  - sketch05\_multi\_signals
  - sketch06\_analog\_read
  - sketch06\_analog\_read\_start
  - sketch07\_tmp36
  - sketch07\_tmp36\_start
  - sketch08\_CdS
  - sketch08\_CdS\_start
  - sketch08\_CdS2
  - sketch09\_hello\_LCD

**Right File Explorer (Node.js Applications):**

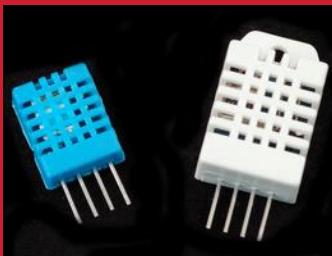
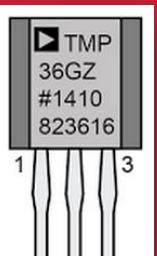
- Path: Portable > NodeJSPortable > Data > hs00 >
- Content:
  - express
  - expressTest
  - hs00App
  - iot** (highlighted with a red dashed box)
  - myApp
  - server
  - start
  - cds
  - cds\_tmp36** (highlighted with a blue dashed box)
  - multi\_signals
  - plotly
  - tmp36
  - data\_chart
  - data\_streaming

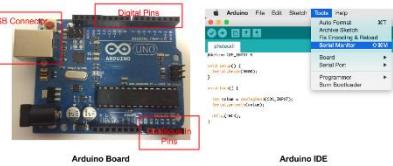


# Arduino

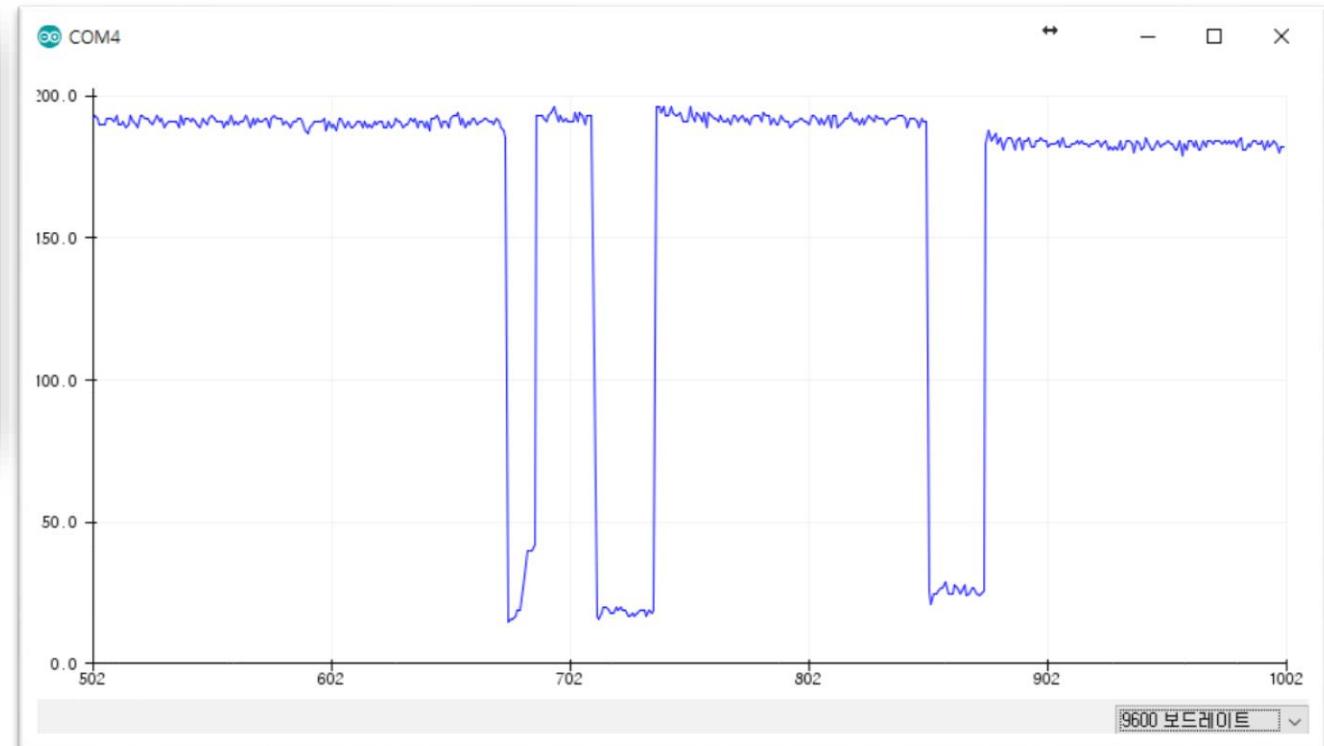
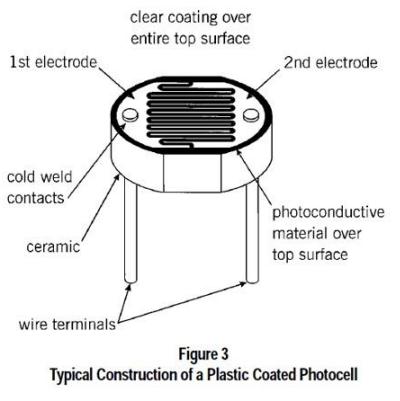
+ Node.js

+ plotly.js

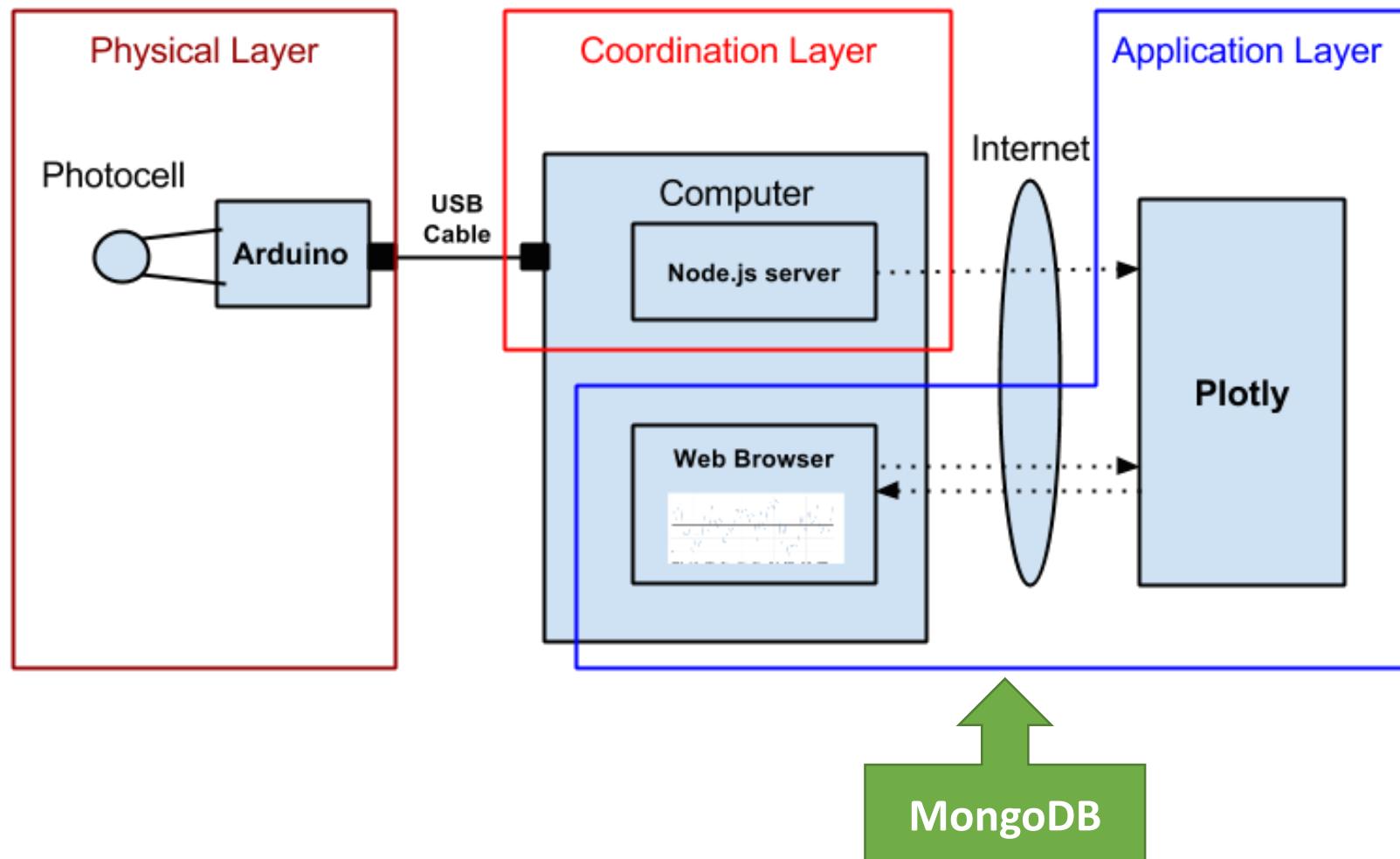




# IOT: HSC

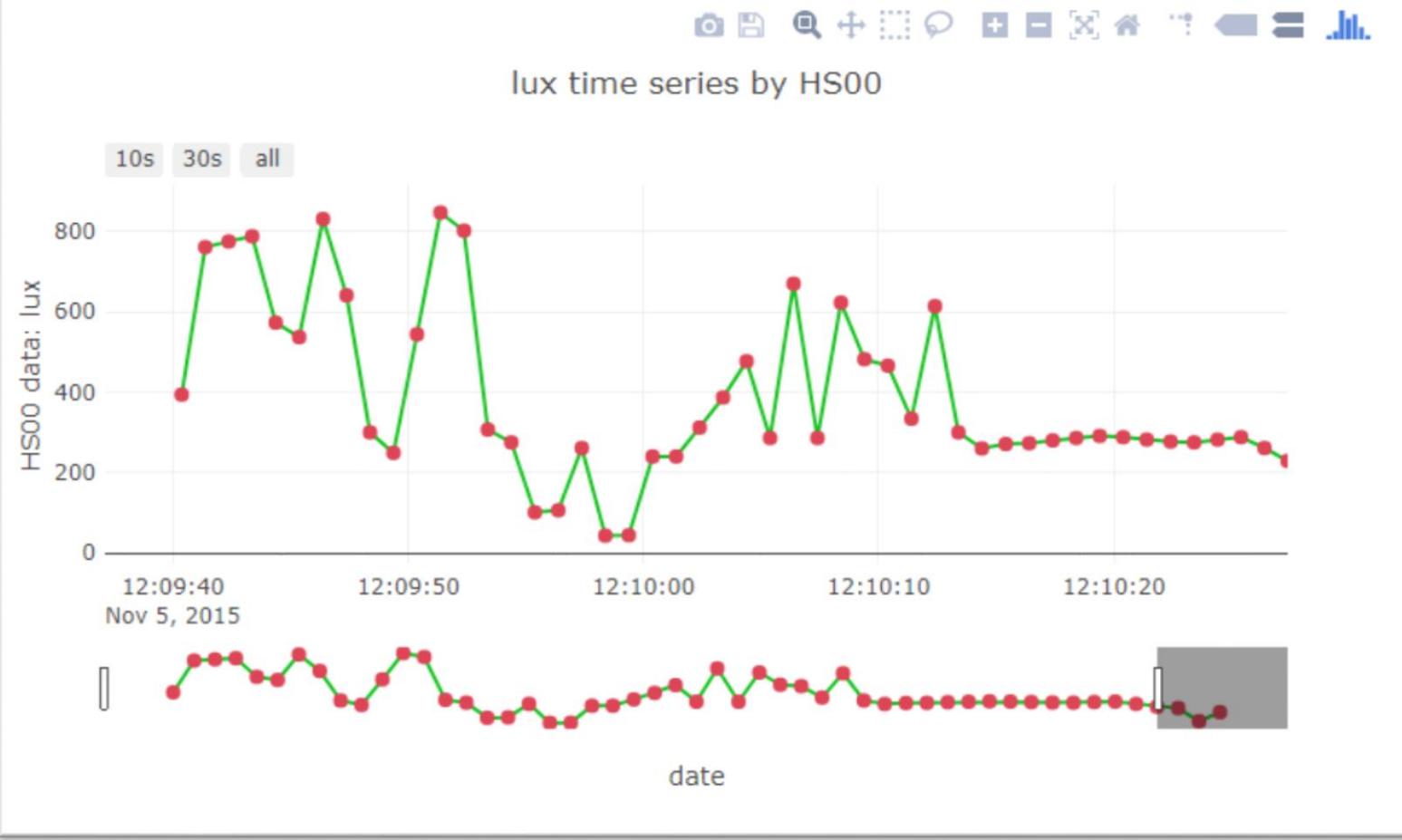


# Layout [H S C]



# Arduino data + plotly

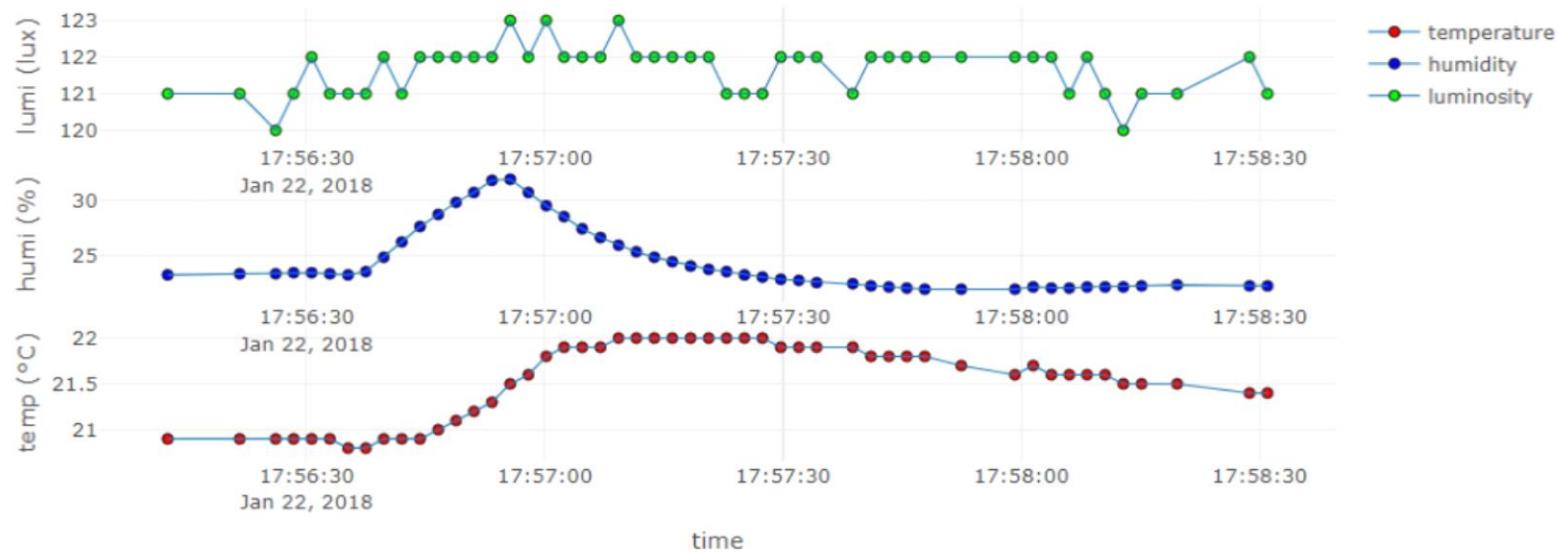
## Time series by HS00



# Real-time Weather Station from sensors



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



## A5. Introduction to visualization

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



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



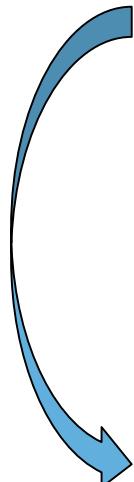
**Visualization & monitoring**



**Data storing & mining**

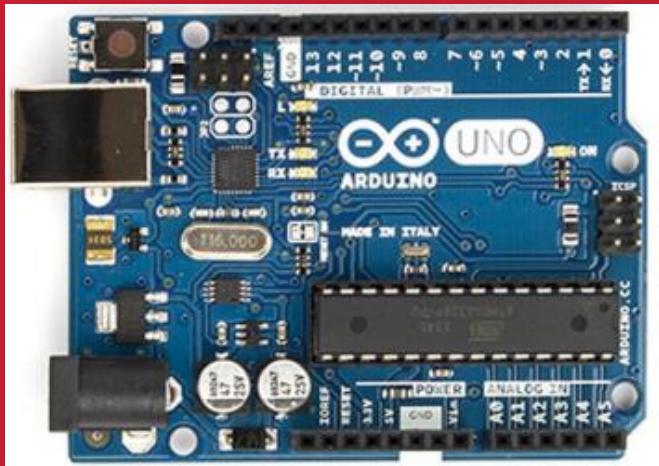


**Service**





# CdS + DHT22

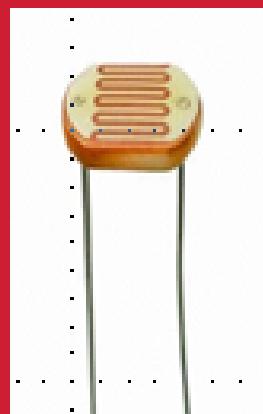


## + plotly.js

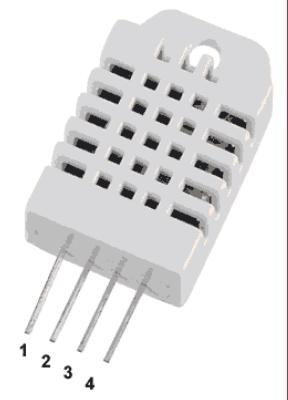
## Node project

**Multi-sensors**

**DHT22 + CdS**

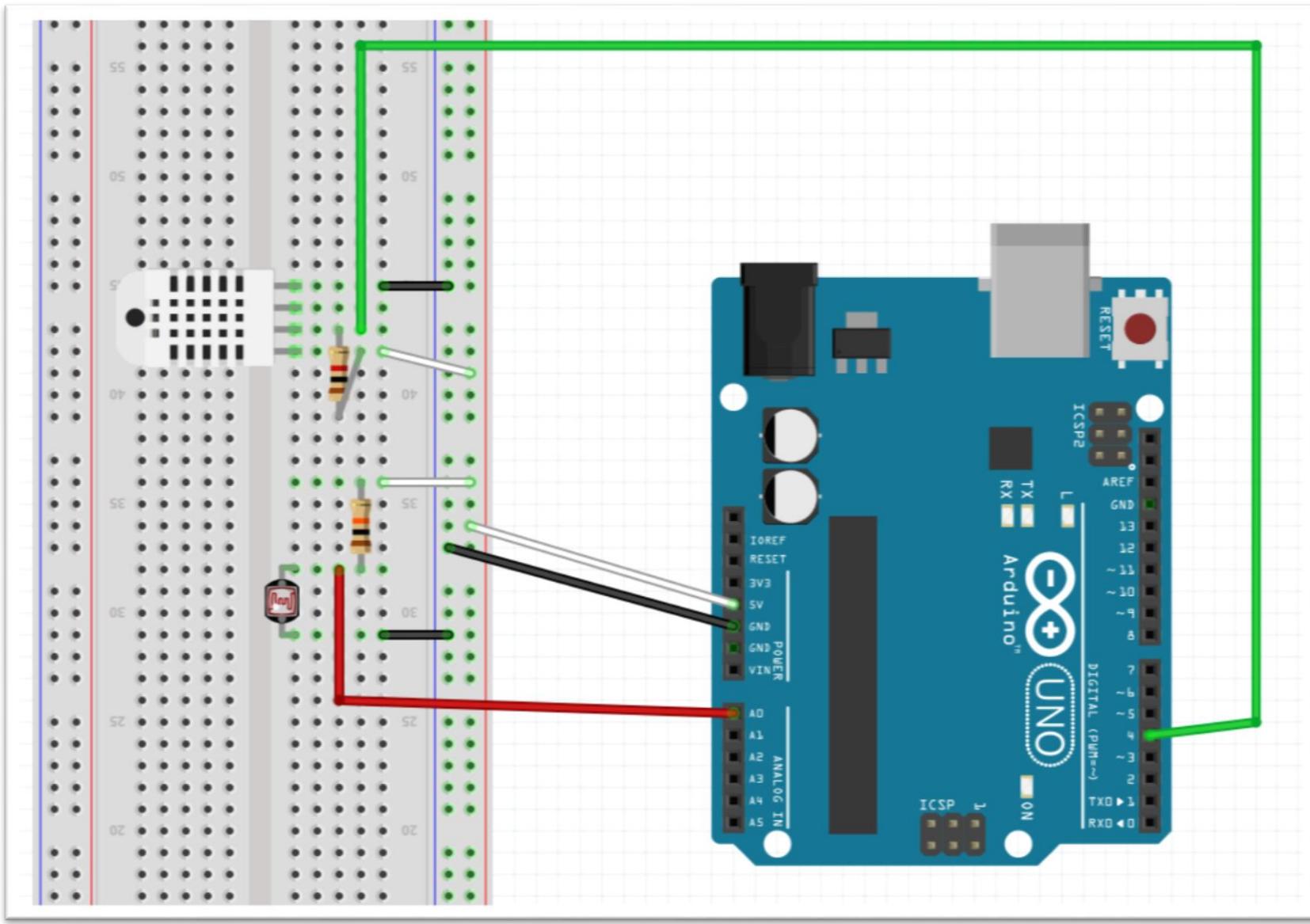


DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND





# 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)

COM4

```
|  
AA00,20.9,21.9,117  
AA00,20.9,21.8,117  
AA00,20.9,21.8,118  
AA00,20.9,21.8,118  
AA00,20.9,21.8,119  
AA00,20.9,21.8,118  
AA00,20.9,21.8,118  
AA00,20.9,21.8,118  
AA00,20.9,21.9,118  
AA00,20.9,21.9,118  
AA00,20.8,21.9,118  
AA00,20.9,22.0,118  
AA00,20.9,22.0,118  
AA00,20.8,21.8,119
```

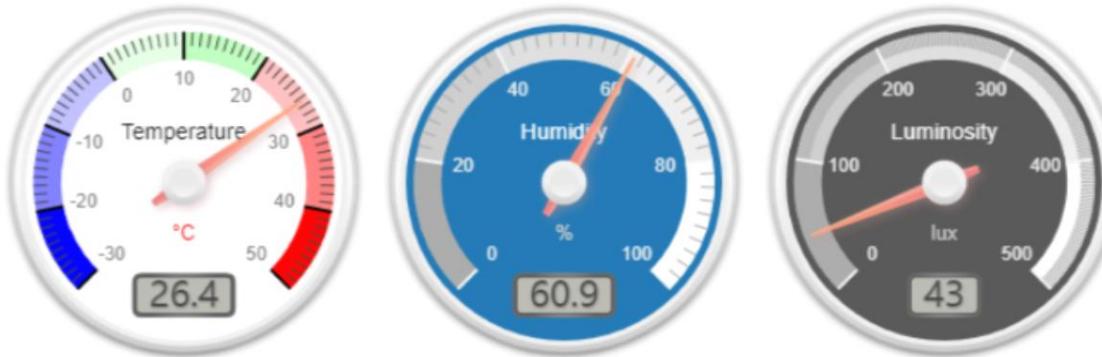


NodeJS - node cds\_dht22\_node

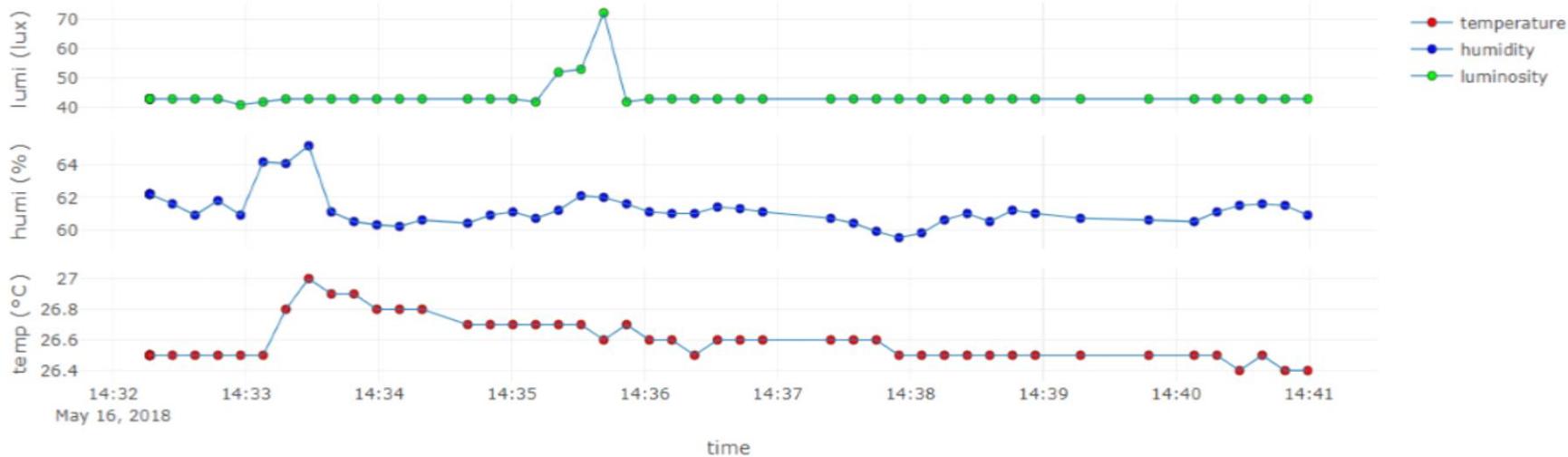
```
D:\Portable\NodeJSPortable\Data\aa00\iot\cds_dht22>node cds_dht22_node  
[ '2018-01-22 17:22:47.683', '20.7', '23.2', '118' ]  
[ '2018-01-22 17:22:49.954', '20.6', '23.2', '116' ]  
[ '2018-01-22 17:22:52.227', '20.7', '23.2', '117' ]  
[ '2018-01-22 17:22:54.486', '20.7', '23.2', '116' ]  
[ '2018-01-22 17:22:56.757', '20.6', '23.2', '117' ]  
[ '2018-01-22 17:22:59.031', '20.7', '23.3', '117' ]  
[ '2018-01-22 17:23:01.306', '20.7', '23.3', '117' ]  
[ '2018-01-22 17:23:03.577', '20.7', '23.3', '117' ]  
[ '2018-01-22 17:23:05.851', '20.7', '23.3', '118' ]  
[ '2018-01-22 17:23:08.109', '20.6', '23.2', '115' ]  
[ '2018-01-22 17:23:10.381', '20.6', '23.2', '113' ]  
[ '2018-01-22 17:23:12.655', '20.7', '23.5', '114' ]  
[ '2018-01-22 17:23:14.928', '20.7', '23.7', '38' ]  
[ '2018-01-22 17:23:17.201', '20.6', '23.9', '117' ]  
[ '2018-01-22 17:23:19.475', '20.7', '24.5', '117' ]  
[ '2018-01-22 17:23:21.732', '20.7', '25.9', '73' ]  
[ '2018-01-22 17:23:24.004', '20.7', '34.2', '118' ]  
[ '2018-01-22 17:23:26.277', '21.3', '55.5', '117' ]  
[ '2018-01-22 17:23:28.553', '21.0', '68.1', '117' ]  
[ '2018-01-22 17:23:30.825', '20.9', '76.1', '117' ]  
[ '2018-01-22 17:23:33.083', '21.0', '74.0', '116' ]  
[ '2018-01-22 17:23:35.355', '21.0', '65.7', '117' ]  
[ '2018-01-22 17:23:37.628', '21.0', '57.7', '116' ]  
[ '2018-01-22 17:23:39.901', '21.0', '51.2', '116' ]  
[ '2018-01-22 17:23:42.175', '21.0', '45.9', '117' ]  
[ '2018-01-22 17:23:44.448', '21.0', '41.6', '117' ]  
[ '2018-01-22 17:23:46.706', '21.0', '38.3', '116' ]  
[ '2018-01-22 17:23:48.979', '21.0', '35.8', '118' ]
```

자동 스크롤

# Real-time Weather Station from sensors



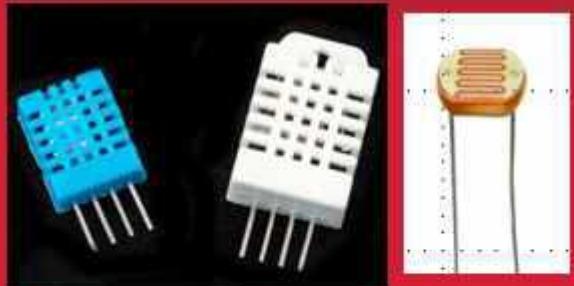
on Time: 2018-05-16 14:40:59.402





[Goal]

Arduino + Node.js



+ plotly.js

+ MongoDB

→ Data storaging

& visualization





# A5.9 MongoDB



MongoDB for GIANT Ideas | 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

The diagram illustrates the speed and integration of MongoDB. It features a central MongoDB database icon connected to multiple client applications (laptops and mobile devices) via green lines. One client shows a map with a location pin, another shows a bar chart, and others show icons for a plane, dollar sign, heart, lock, and gear. Binary code (01101000, 00110110, etc.) is shown flowing between the database and the clients, symbolizing fast data transfer. The background is white with faint, overlapping icons of various applications.



# 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 유지하고 있습니다.

## NoSQL?

흔히 NoSQL이라고 해서 아, SQL이 없는 데이터베이스구나!라고 생각 할 수도 있겠지만, 진짜 의미는 Not Only SQL입니다. 기존의 RDBMS의 한계를 극복하기 위해 만들어진 새로운 형태의 데이터저장소입니다. 관계형 DB가 아니므로, RDMS처럼 고정된 스키마 및 JOIN이 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 record와 비슷한 개념인데요, 이의 데이터 구조는 한개이상의 key-value pair으로 이루어져있습니다. MongoDB 샘플 Document를 확인해 볼까요?

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



# A5.9 MongoDB



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

**\_id** 는 12bytes의 hexadecimal 값으로서, 각 **document**의  
유일함(uniqueness)을 제공합니다.  
이 값의 첫 4bytes 는 현재 **timestamp**, 다음 3bytes는 **machine id**, 다음  
2bytes는 **MongoDB** 서버의 프로세스 **id**, 마지막 3bytes는 순차번호입니다 추가될때마다  
값이 높아진다는 거지요.

**Document**는 동적(dynamic)의 **schema**를 갖고 있습니다. 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?

**Collection**은 **MongoDB Document**의 그룹입니다. **Document**들이  
**Collection** 내부에 위치하고 있습니다. **RDMS**의 **table**과 비슷한 개념입니다만 **RDMS**와  
달리 **schema**를 따로 가지고 있지 않습니다. **Document** 부분설명에 나와있듯이 각  
**Document**들이 동적인 **schema**를 가지고 있으니까요

## Database?

**Database**는 **Collection**들의 물리적인 컨테이너입니다. 각 **Database**는 파일시스템에  
여러파일들로 저장됩니다.



# MongoDB 3.6

## Move at the Speed of your Data

MongoDB 3.6 introduces innovations that make you more productive with less code and operations, whether it's rapidly delivering cutting-edge applications to market, ensuring an exceptional experience on a global scale, or unlocking the intelligence you need for your next move.

[Try it now](#)[Download the Guide to MongoDB 3.6](#)

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



# A5.9 MongoDB



The screenshot shows the MongoDB download center page. At the top, there's a navigation bar with links for SOLUTIONS, CLOUD, CUSTOMERS, RESOURCES, and ABOUT US. Below this is a horizontal bar with colored segments corresponding to different MongoDB products: Atlas (grey), Community Server (green, highlighted with a red dashed box), Enterprise Server (grey), Ops Manager (grey), Compass (grey), and Connector for BI (grey). A green vertical bar is on the left side of the page. In the main content area, there's a section for the "Current Stable Release (3.6.5)" dated 05/21/2018. It includes links for "Release Notes" and "Changelog", and download options for "tgz" and "zip". Below this, there's a "Version:" dropdown menu set to "Windows Server 2008 R2 64-bit and later, with SSL support x64". Under "Installation Package:", there's a green button labeled "DOWNLOAD (msi)". At the top right of the main content area, there are links for "Current Release", "Previous Releases", and "Development Releases".

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

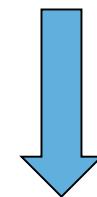


# A5.9.1 MongoDB install - 1

mongodb-win32-x86\_64-2008plus-ssl-3.6.2-signed

NodeJSPortable\_5.7.0.paf

Sublime Text Build 3143 x64



MongoDB 3.6.2 2008R2Plus SSL (64 bit) Setup



MongoDB Compass

## Install MongoDB Compass

MongoDB Compass is the official graphical user interface for MongoDB.

By checking below this installer will automatically download and install the latest version of MongoDB Compass on this machine. You can learn more about MongoDB Compass here: <https://www.mongodb.com/products/comp...>

Install MongoDB Compass

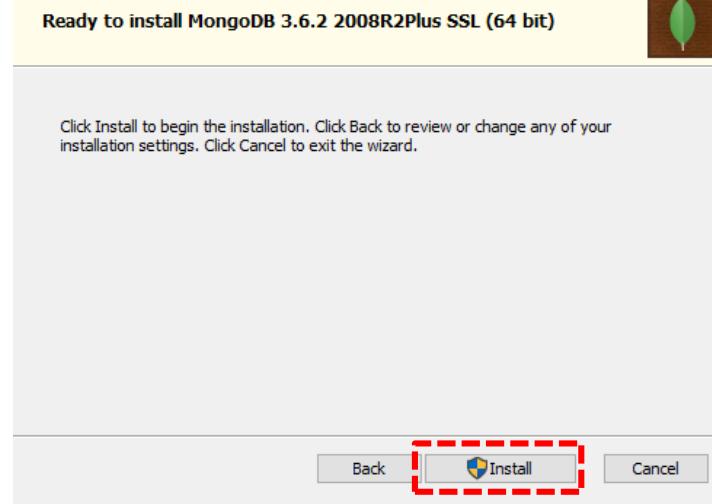
Back

Next

Cancel

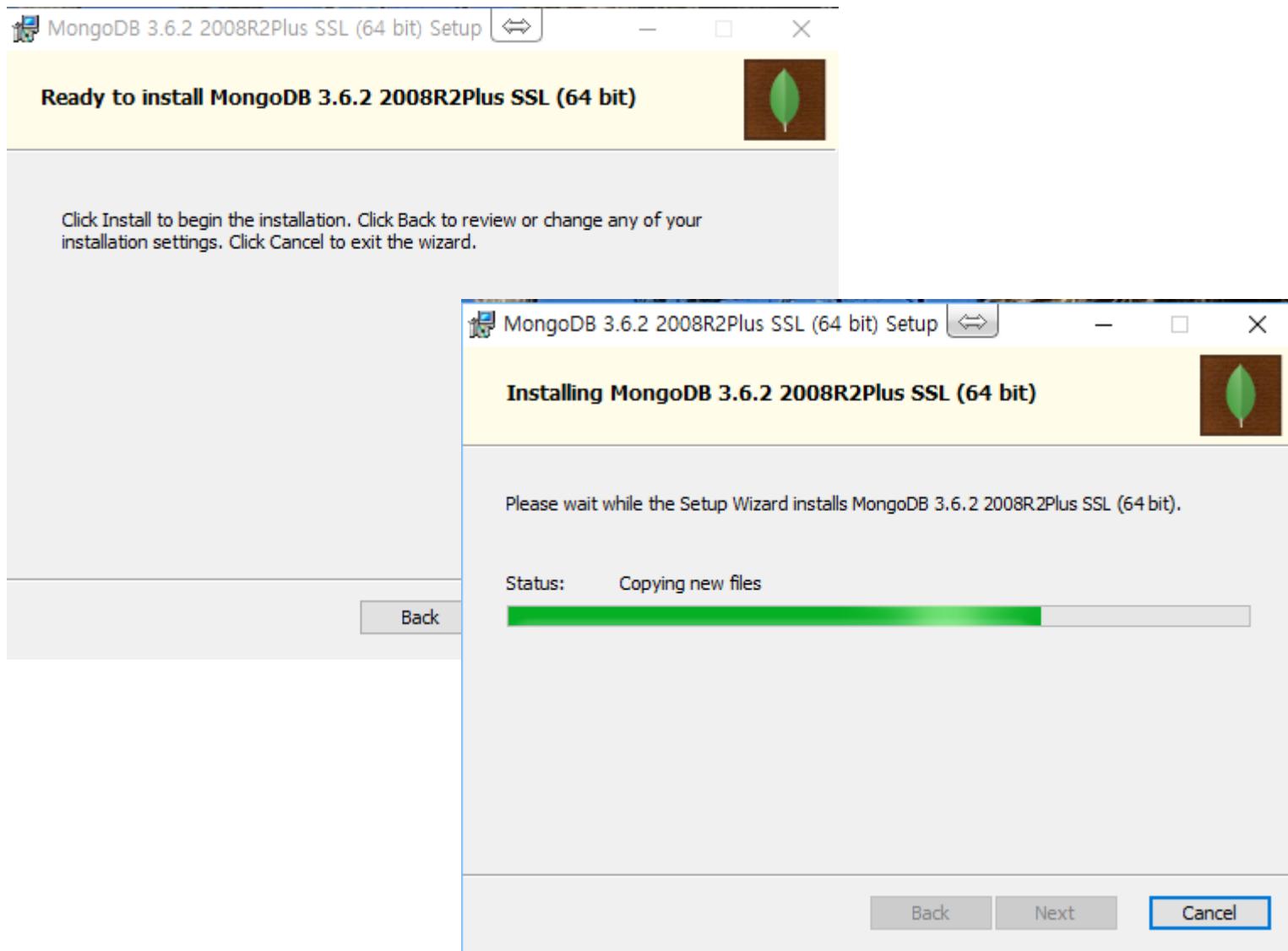


MongoDB 3.6.2 2008R2Plus SSL (64 bit) Setup





# A5.9.1 MongoDB install - 2





# A5.9.1 MongoDB install – 3

MongoDB Compass Community - Connect

Connect View Help

NEW CONNECTION

FAVORITES

RECENTS

Conn

LIMITATION, ANY CAUSE OF ACTION SOUNDING IN CONTRACT, TORT, OR STRICT LIABILITY, SHALL NOT EXCEED ONE HUNDRED DOLLARS (U.S. \$100.00). THE FOREGOING LIMITATIONS WILL APPLY NOTWITHSTANDING THE FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY PROVIDED HEREIN.

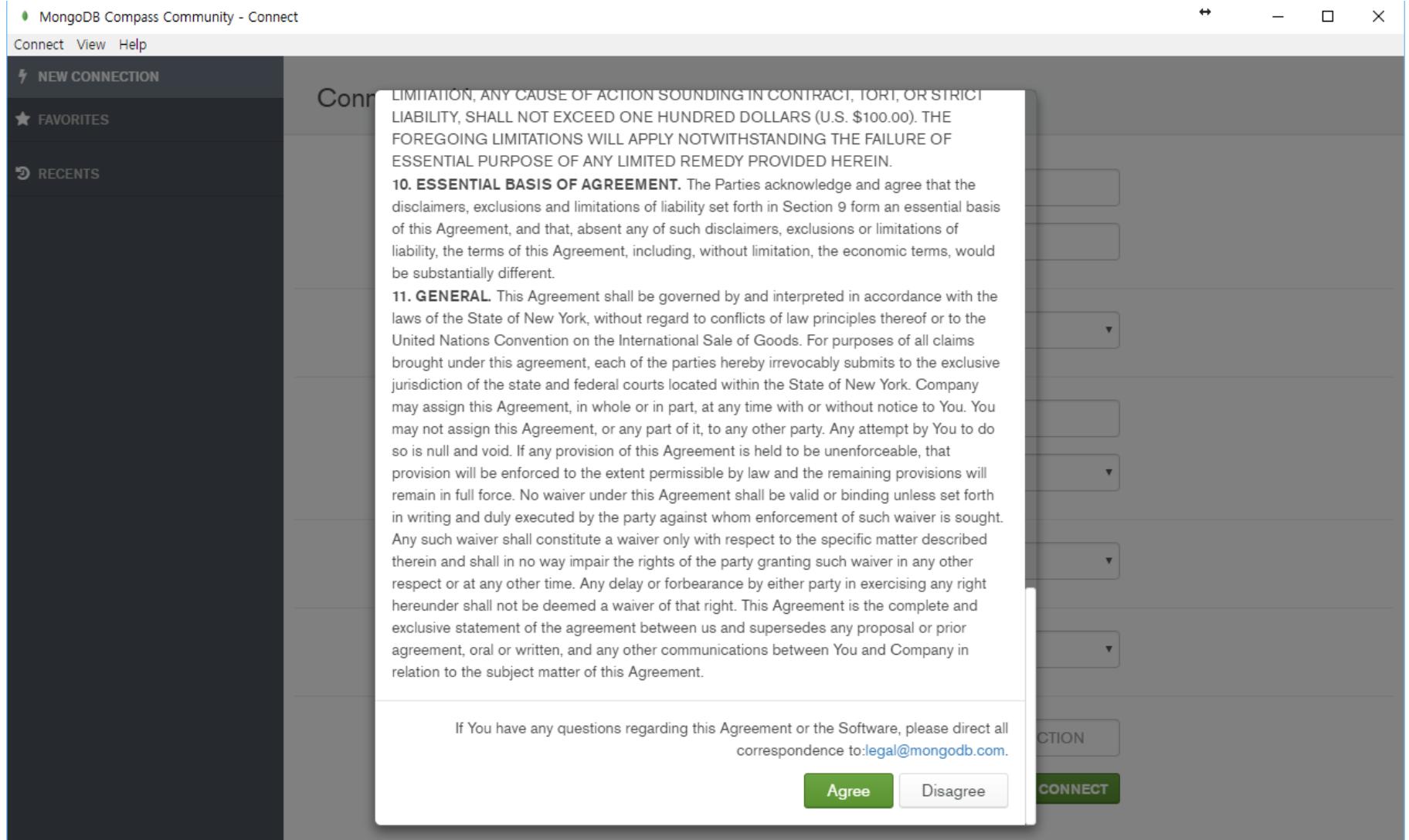
**10. ESSENTIAL BASIS OF AGREEMENT.** The Parties acknowledge and agree that the disclaimers, exclusions and limitations of liability set forth in Section 9 form an essential basis of this Agreement, and that, absent any of such disclaimers, exclusions or limitations of liability, the terms of this Agreement, including, without limitation, the economic terms, would be substantially different.

**11. GENERAL.** This Agreement shall be governed by and interpreted in accordance with the laws of the State of New York, without regard to conflicts of law principles thereof or to the United Nations Convention on the International Sale of Goods. For purposes of all claims brought under this agreement, each of the parties hereby irrevocably submits to the exclusive jurisdiction of the state and federal courts located within the State of New York. Company may assign this Agreement, in whole or in part, at any time with or without notice to You. You may not assign this Agreement, or any part of it, to any other party. Any attempt by You to do so is null and void. If any provision of this Agreement is held to be unenforceable, that provision will be enforced to the extent permissible by law and the remaining provisions will remain in full force. No waiver under this Agreement shall be valid or binding unless set forth in writing and duly executed by the party against whom enforcement of such waiver is sought. Any such waiver shall constitute a waiver only with respect to the specific matter described therein and shall in no way impair the rights of the party granting such waiver in any other respect or at any other time. Any delay or forbearance by either party in exercising any right hereunder shall not be deemed a waiver of that right. This Agreement is the complete and exclusive statement of the agreement between us and supersedes any proposal or prior agreement, oral or written, and any other communications between You and Company in relation to the subject matter of this Agreement.

If You have any questions regarding this Agreement or the Software, please direct all correspondence to: [legal@mongodb.com](mailto:legal@mongodb.com).

Agree Disagree

CONNECT





# A5.9.1 MongoDB install - 4

## Privacy Settings

To enhance the user experience, Compass can integrate with 3rd party services, which requires external network requests. Please choose from the settings below:

**Enable Crash Reports**

Allow Compass to send crash reports containing stack traces and unhandled exceptions.

**Enable Usage Statistics**

Allow Compass to send anonymous usage statistics.

**Enable Automatic Updates**

Allow Compass to periodically check for new updates.

With any of these options, none of your personal information or stored data will be submitted.

Learn more:[MongoDB Privacy Policy](#)

**Start Using Compass**





## A5.9.1 MongoDB install – 5.



원도우10: 설정 > 시스템 > 정보

[중요] 시스템 환경변수 : PATH 에 경로 추가

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



# A5.9.1 MongoDB install – 6.



설정



## Windows 설정

제어판

제어판



시스템  
디스플레이, 알림, 전원



장치  
Bluetooth, 프린터, 마우스



전화  
Android, iPhone 연결



네트워크 및 인터넷  
Wi-Fi, 비행기 모드, VPN



개인 설정  
배경, 잠금 화면, 색



앱  
설치 제거, 기본값, 옵션 기능



계정  
내 계정, 메일, 등기화, 회사, 가족



시간 및 언어  
음성, 지역, 날짜



게임  
게임 바, DVR, 브로드캐스팅, 게임 모드



접근성  
내레이터, 돋보기, 고대비



개인 정보  
위치, 카메라



업데이트 및 보안  
Windows 업데이트, 복구, 백업



# A5.9.1 MongoDB install – 7.

원도우10: 설정 > ‘제어판’ 검색 > 모든 제어판 항목에서 ‘시스템’ 선택  
➤ 고급 시스템 설정

The screenshot shows the Windows 10 Control Panel interface. The left sidebar has a red dashed box around the '제어판 흘' (Control Panel Home) icon. The main area displays basic computer information:

제어판 흘

제어판에 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	설정 변경
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

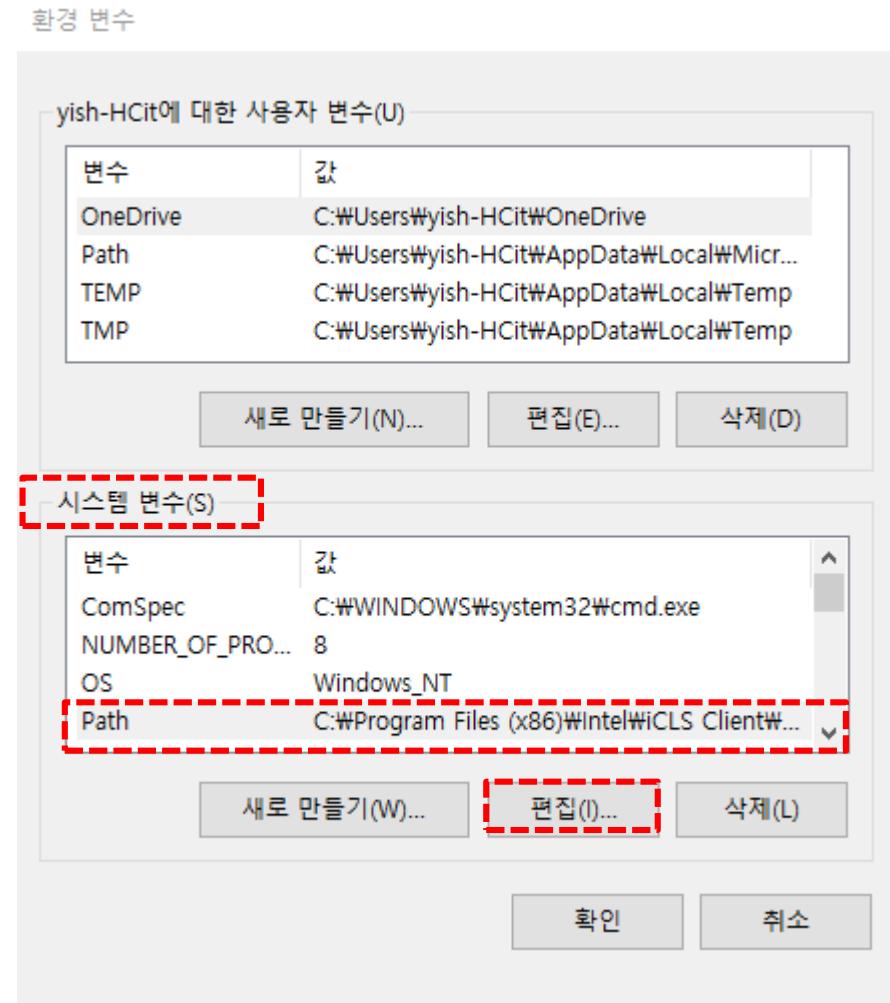
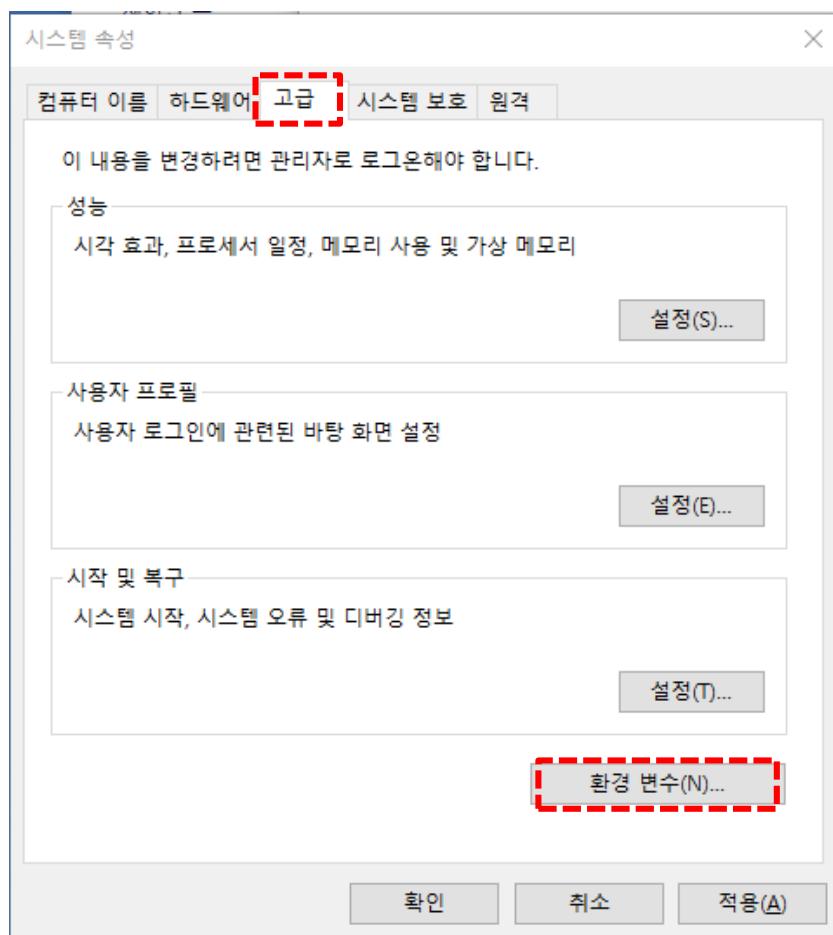
보안 및 유지 관리

Windows 10 logo



A5.9.1 MongoDB install – 8.

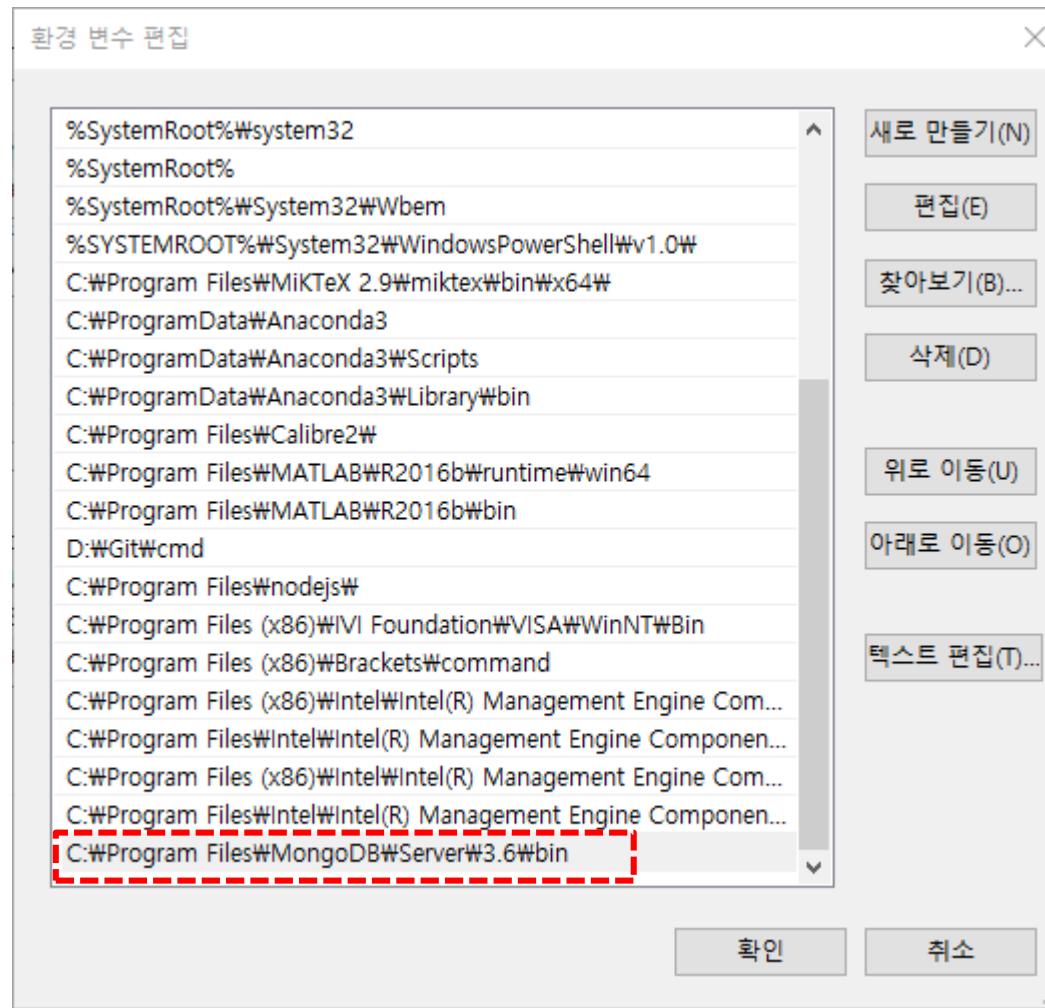
## 환경 변수 설정





# A5.9.1 MongoDB install – 9.

## 환경 변수 추가





# A5.9.2 MongoDB shell - 1

## 1. Mongo shell 실행

> mongo

```
명령 프롬프트
Microsoft Windows [Version 10.0.16299.192]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\yish-HCit>mongo
MongoDB shell version v3.6.2
connecting to: mongodb://127.0.0.1:27017/
2018-01-23T11:12:26.270+0900 W NETWORK  [thread1] Failed to connect to 127.0.0.1:27017 after 50
00ms milliseconds, giving up.
2018-01-23T11:12:26.271+0900 E QUERY    [thread1] Error: couldn't connect to server 127.0.0.1:2
7017, connection attempt failed :
connect@src/mongo/shell/mongo.js:251:13
@(connect):1:6
exception: connect failed

C:\Users\yish-HCit>
```



# A5.9.2 MongoDB shell - 2

## 2. MongoDB 저장소 만들기

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

```
c:\ 관리자: 명령 프롬프트
C:\>md mongodb
C:\>cd mongodb
C:\mongodb>md data
C:\mongodb>dir
C 드라이브의 볼륨: SYSTEM
볼륨 일련 번호: 3E92-DE79

C:\mongodb 디렉터리

2018-05-23 오후 12:17 <DIR> .
2018-05-23 오후 12:17 <DIR> ..
2018-05-23 오후 12:17 <DIR> data
                      0개 파일          0 바이트
                      3개 디렉터리 97,830,256,640 바이트 남음

C:\mongodb>
```



## A5.9.2 MongoDB shell - 3

### 3. Run MongoDB by using **mongod.exe**

➤ **mongod --dbpath c:\mongodb\data**

C:\ 명령 프롬프트 - mongod --dbpath d:\mongodb\data



➤ **mongod --dbpath c:\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
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.6.2
Server has startup warnings:
2018-01-22T19:27:33.549-0700 I CONTROL [initandlisten]
2018-01-22T19:27:33.549-0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-01-22T19:27:33.550-0700 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2018-01-22T19:27:33.550-0700 I CONTROL [initandlisten]
2018-01-22T19:27:33.554-0700 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2018-01-22T19:27:33.557-0700 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2018-01-22T19:27:33.559-0700 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2018-01-22T19:27:33.561-0700 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all to
2018-01-22T19:27:33.563-0700 I CONTROL [initandlisten] ** bind to all interfaces. If
this behavior is desired, start the
2018-01-22T19:27:33.564-0700 I CONTROL [initandlisten] ** server with --bind_ip 127.0
.1 to disable this warning.
2018-01-22T19:27:33.566-0700 I CONTROL [initandlisten]
2018-01-22T19:27:33.567-0700 I CONTROL [initandlisten]
2018-01-22T19:27:33.569-0700 I CONTROL [initandlisten] ** WARNING: The file system cache of this machine is configured to be greater than 40% of the total memory. This can lead to increased memory pressure and poor performance.
2018-01-22T19:27:33.570-0700 I CONTROL [initandlisten] See http://dochub.mongodb.org/core/wt-w
indows-system-file-cache
2018-01-22T19:27:33.571-0700 I CONTROL [initandlisten]
```



# A5.9.2 MongoDB shell - 5

## 5. mongo shell :

Run new cmd

mongo

show dbs

use local

show  
collections

help

```
C:\ 명령 프롬프트 - mongo
> show dbs
admin 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

    show dbs
    show collections
    show users
    show profile
    show logs
    show log [name]

default
    use <db_name>
    db.foo.find()
    db.foo.find( { a : 1 } )
    it
    DBQuery.shellBatchSize = x
    exit

>
```

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 database names  
show collections in current database  
show users in current database  
show most recent system.profile entries with time >= 1ms  
show the accessible logger names  
prints out the last segment of log in memory, 'global' is

set current database  
list objects in collection foo  
list objects in foo where a == 1  
result of the last line evaluated; use to further iterate  
set default number of items to display on shell  
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()

```
C:\ 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 프로세스 id, 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()

```
C:\ 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()**

```
C:\ 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",
    "lastName" : "Park"
}
>
```



# A5.9.3 MongoDB shell coding

## 5. update a record

update record2

db.user.find().pretty()

```
C:\ 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

The screenshot shows a web browser window with the title "Mongoose ODM v5.0.1". The address bar contains "mongoosejs.com". The page itself features a large, bold red "mongoose" logo. Below it, the text "elegant mongodb object modeling for node.js" is displayed. At the bottom of the page, there are two dark grey buttons with white text: "read the docs" and "discover plugins". Between these buttons, the text "Version 5.0.1" is centered. A "Fork me on GitHub" button is located in the top right corner of the page area.

mongoose

elegant mongodb object modeling for node.js

read the docs

discover plugins

Version 5.0.1

Fork me on GitHub

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



# 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 releases

Branch: master ▾ New pull request

vkarlov15 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

```
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)**

```
D:\Portable\NodeJS\Portable\Data\Waa00\iot\cds_dht22\dbtest.js (Data) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
  Data
    aa00
      express
      expressTest
    iot
      cds
        cds_dht22
          node_modules
          /* cds_dht22_node.js
          /* dbtest.js
          /* package.json
          /* cds_tmp36
          /* plotly
          /* tmp36
          myApp
          server
          start
          node_modules
          npm_cache
          settings
          Temp
            express
            /* express.cmd
            npm
            /* npm.cmd
            PortableApps.comLauncherRuntimeData-NodeJS
dbtest.js
1 // dbtest.js
2 var mongoose = require('mongoose');
3 mongoose.connect('mongodb://localhost/test');
4
5 var SensorSchema = new mongoose.Schema({
6   data: String,
7   created: Date
8 });
9
10 // data model
11 var Sensor = mongoose.model("Sensor", SensorSchema);
12
13 var sensor1 = new Sensor({data:'124', created: new Date()});
14 sensor1.save();
15
16 var sensor2 = new Sensor({data:'573', created: new Date()});
17 sensor2.save();
18
19 console.log("Sensor data were saved in MongoDB");
20
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()

```
명령 프롬프트 - mongo
> show dbs
aa00 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().pretty()
...
2018-01-23T14:31:32.959+0900 E QUERY [thread1] SyntaxError: identifier starts immediately after numeric literal @shell:1:16
> use test
switched to db test
> show collections
sensors
> db.sensors.find().pretty()
{
    "_id" : ObjectId("5a66c84d000e8f1630e176e9"),
    "data" : "124",
    "created" : ISODate("2018-01-23T05:29:49.973Z"),
    "__v" : 0
}
{
    "_id" : ObjectId("5a66c84d000e8f1630e176ea"),
    "data" : "573",
    "created" : ISODate("2018-01-23T05:29:49.977Z"),
    "__v" : 0
}
>
```



# A5.9.4 MongoDB + Node.js : mongoose

## 4. dbtest2.js (use Sublime Text 3)

D:\Portable\NodeJS\Portable\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

```
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

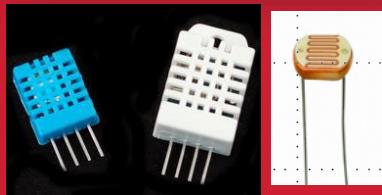
> show dbs

> 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
}
> -
```



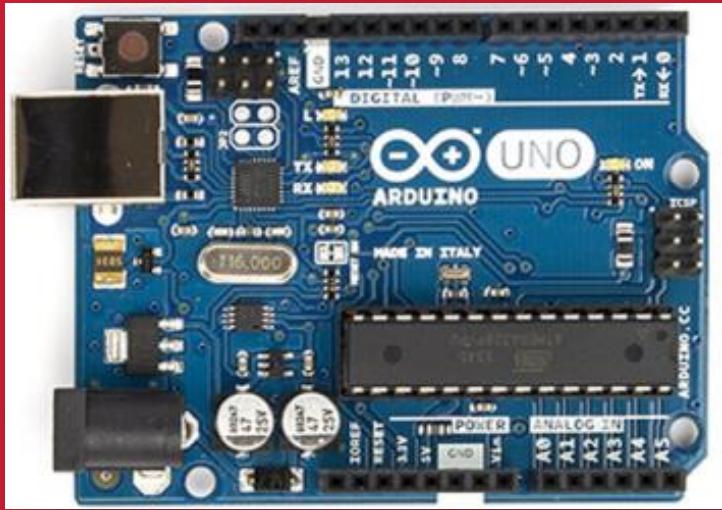
```
> show dbs
aa00    0.000GB
admin   0.000GB
config  0.000GB
iot     0.000GB
iot2    0.000GB
iot3    0.001GB
local   0.000GB
test    0.000GB
test2   0.000GB
>
```

# MongoDB from Arduino with node.js & mongoose

```
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
```



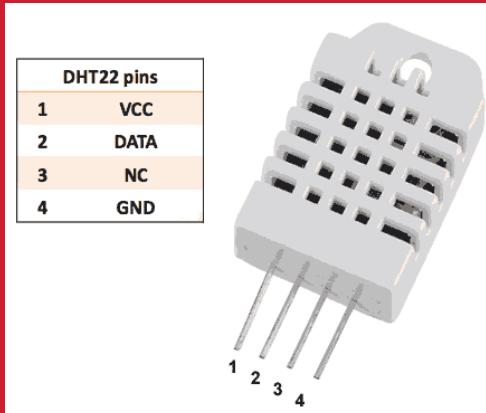
# Arduino & Node.js & MongoDB



**Multi-sensors**

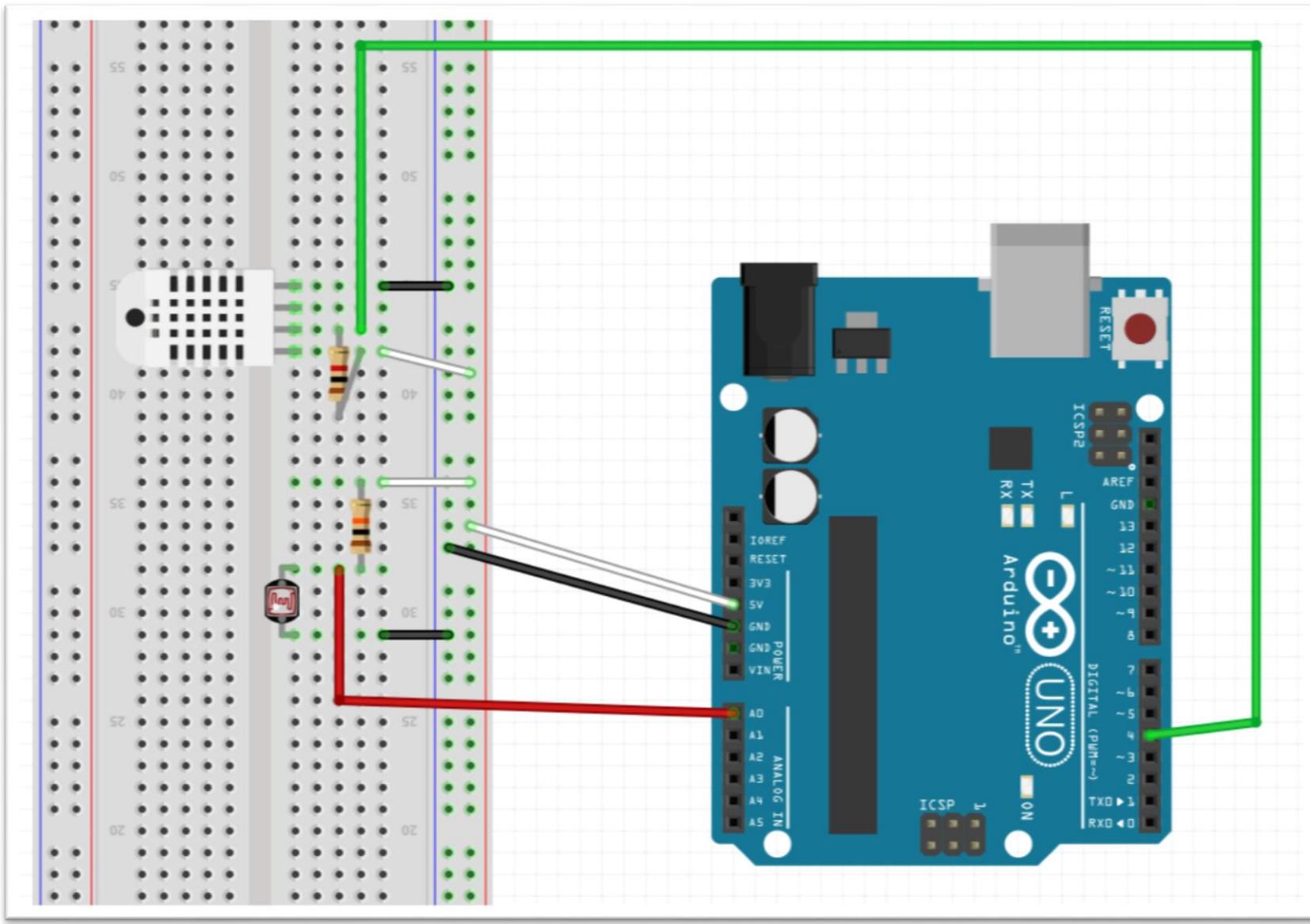
**DHT22 + CdS**

DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

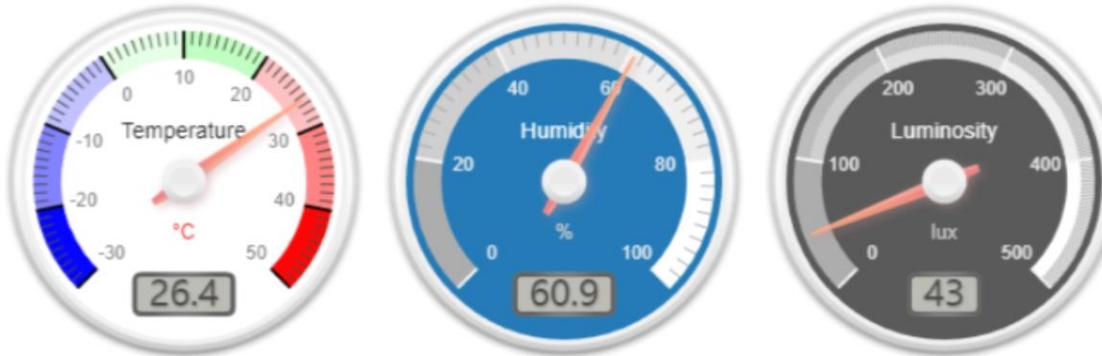




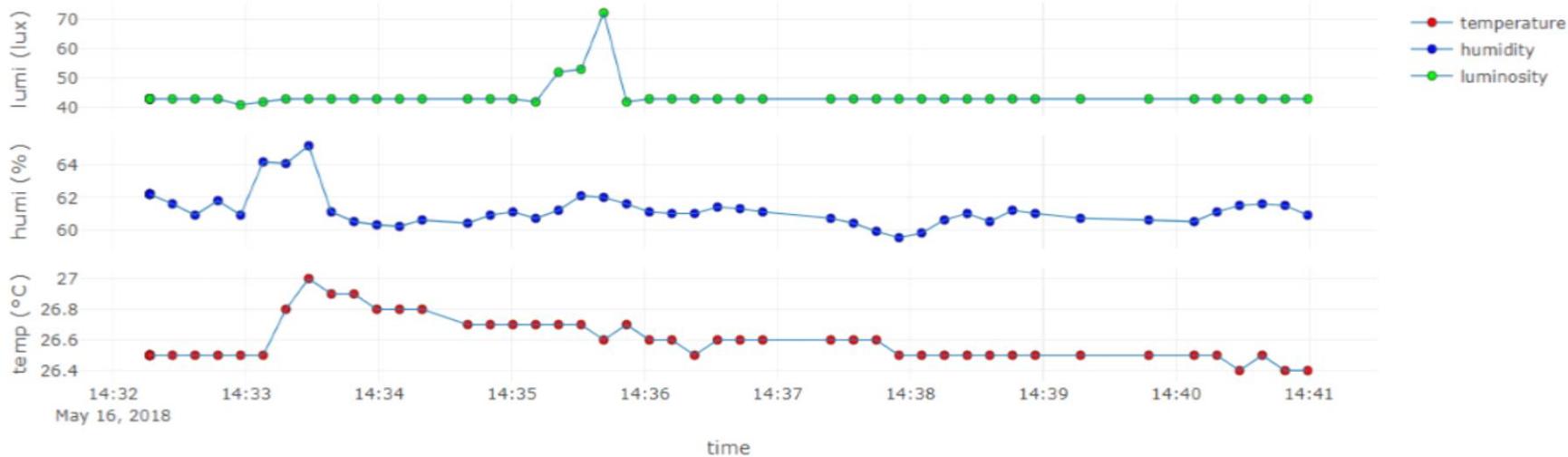
# DHT22 + CdS : circuit



# 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]

```
└─ iot
    └─ cds
        └─ cds_dht22
            /* cds_dht22_express.js
            /* cds_dht22_mongodb.js
            /* cds_dht22_node.js
            <> client_CdS_DHT22.html
            <> client_CdS_DHT22_chaos.html
            /* dbtest.js
            /* dbtest2.js
            /* dbtest_START.js
            /* gauge.min.js
            /* package.json
```



# 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\_mongodb.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

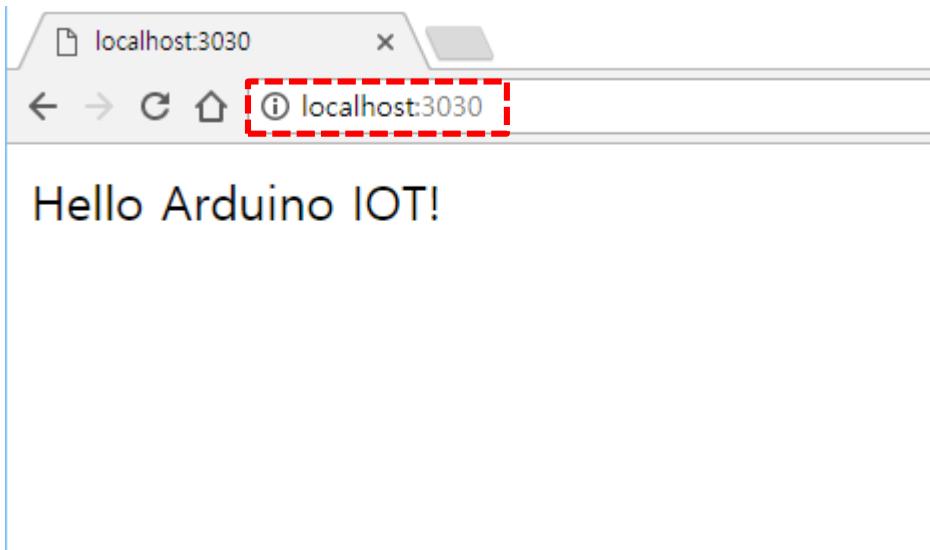
## 2.3 cds\_dht22\_express.js → Run

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



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

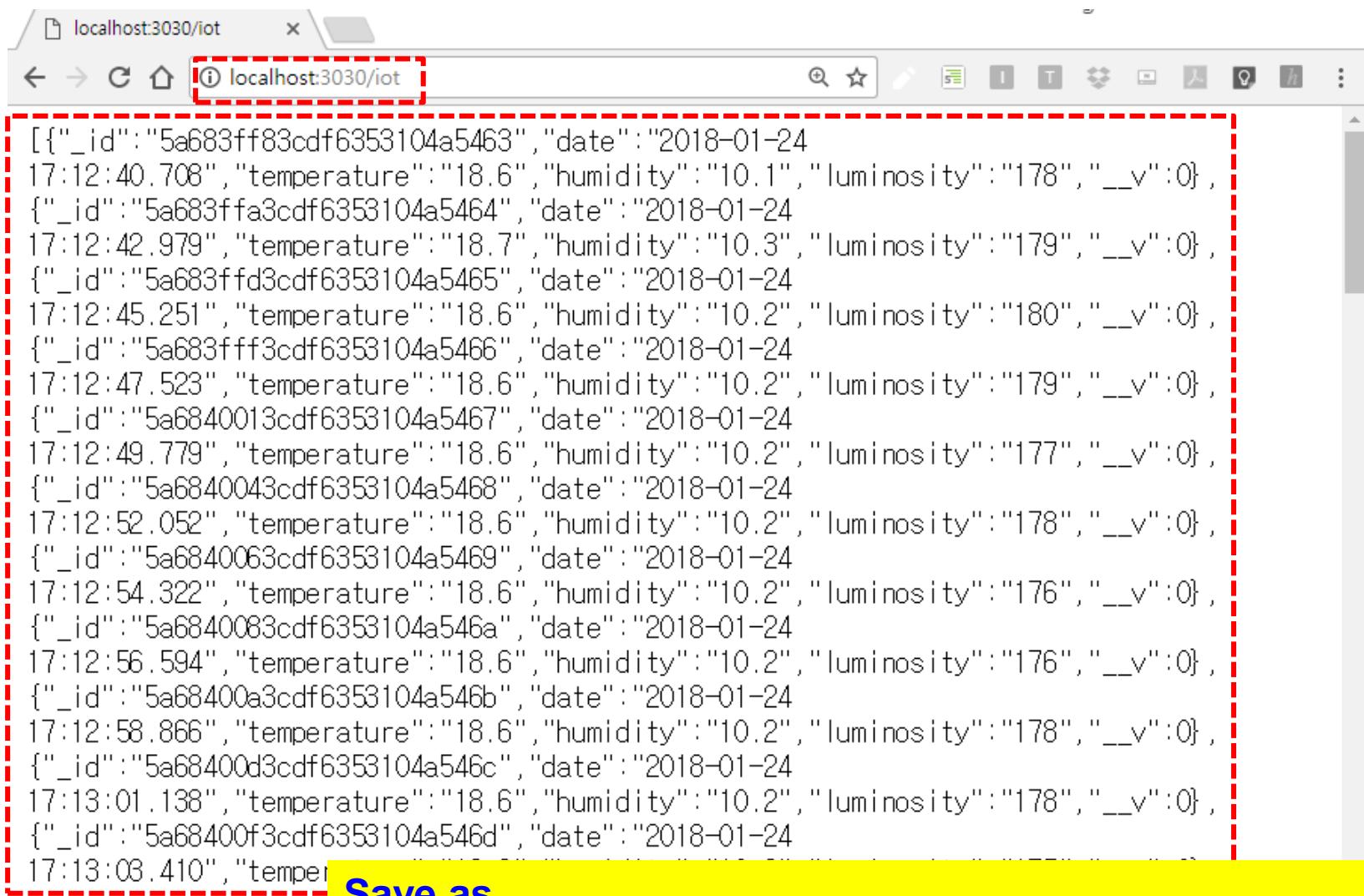
## 2.4 cds\_dht22\_express.js → routing1, <http://localhost:3030/>





## 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>

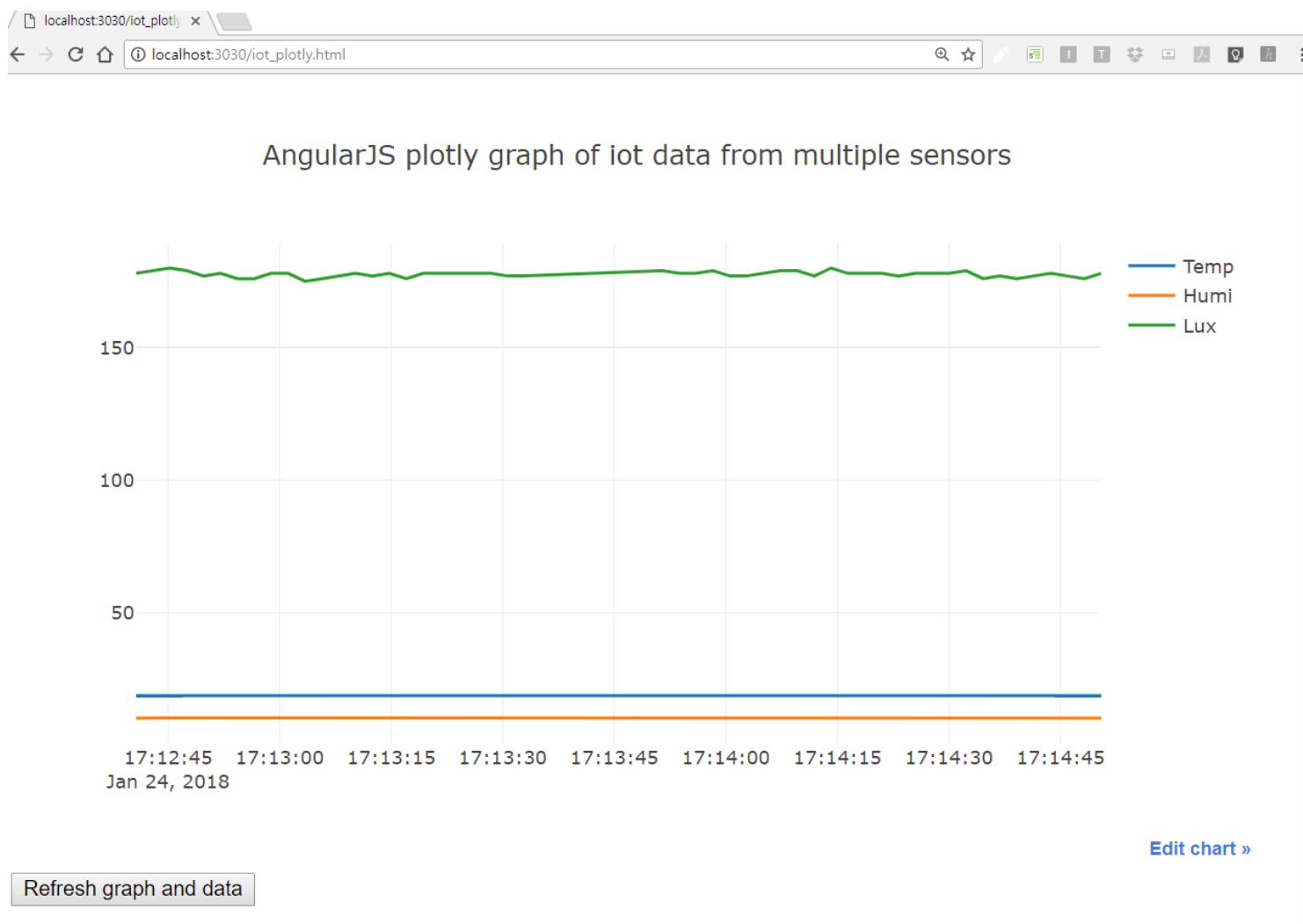
The screenshot shows a web browser window with the URL `localhost:3030/iot/5a683ffd3cdf6353104a5465`. The page content is a JSON object with a red dashed border around it:

```
{"_id": "5a683ffd3cdf6353104a5465", "date": "2018-01-24 17:12:45.251", "temperature": "18.6", "humidity": "10.2", "luminosity": "180", "__v": 0}
```



# DHT22 + CdS + Node.js + MongoDB

## [Next week] Web monitoring





# DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring

The screenshot shows a web browser window with the URL `chaos.inje.ac.kr:3030/iot3.html`. The page title is "MongoDB database". Below the title, there is a large heading "Time series : Multi sensor". To the right of the heading, a modal dialog box is displayed with the text "chaos.inje.ac.kr:3030 내용:" followed by a list of MongoDB documents. At the bottom right of the dialog box is a blue button labeled "확인".

```
[{"_id": "5aa584d0ea0bd2064cb1f9ab", "date": "2018-03-12 04:34:40.662", "temperature": "16.6", "humidity": "24.9", "luminosity": "0", "__v": 0}, {"_id": "5aa584daea0bd2064cb1f9ac", "date": "2018-03-12 04:34:50.923", "temperature": "16.6", "humidity": "24.9", "luminosity": "0", "__v": 0}, {"_id": "5aa584e5ea0bd2064cb1f9ad", "date": "2018-03-12 04:35:01.168", "temperature": "16.6", "humidity": "24.9", "luminosity": "0", "__v": 0}, {"_id": "5aa584efea0bd2064cb1f9ae", "date": "2018-03-12 04:35:11.429", "temperature": "16.6", "humidity": "24.9", "luminosity": "0", "__v": 0}, {"_id": "5aa584f9ea0bd2064cb1f9af", "date": "2018-03-12 04:35:21.674", "temperature": "16.6", "humidity": "24.9", "luminosity": "0", "__v": 0}]
```

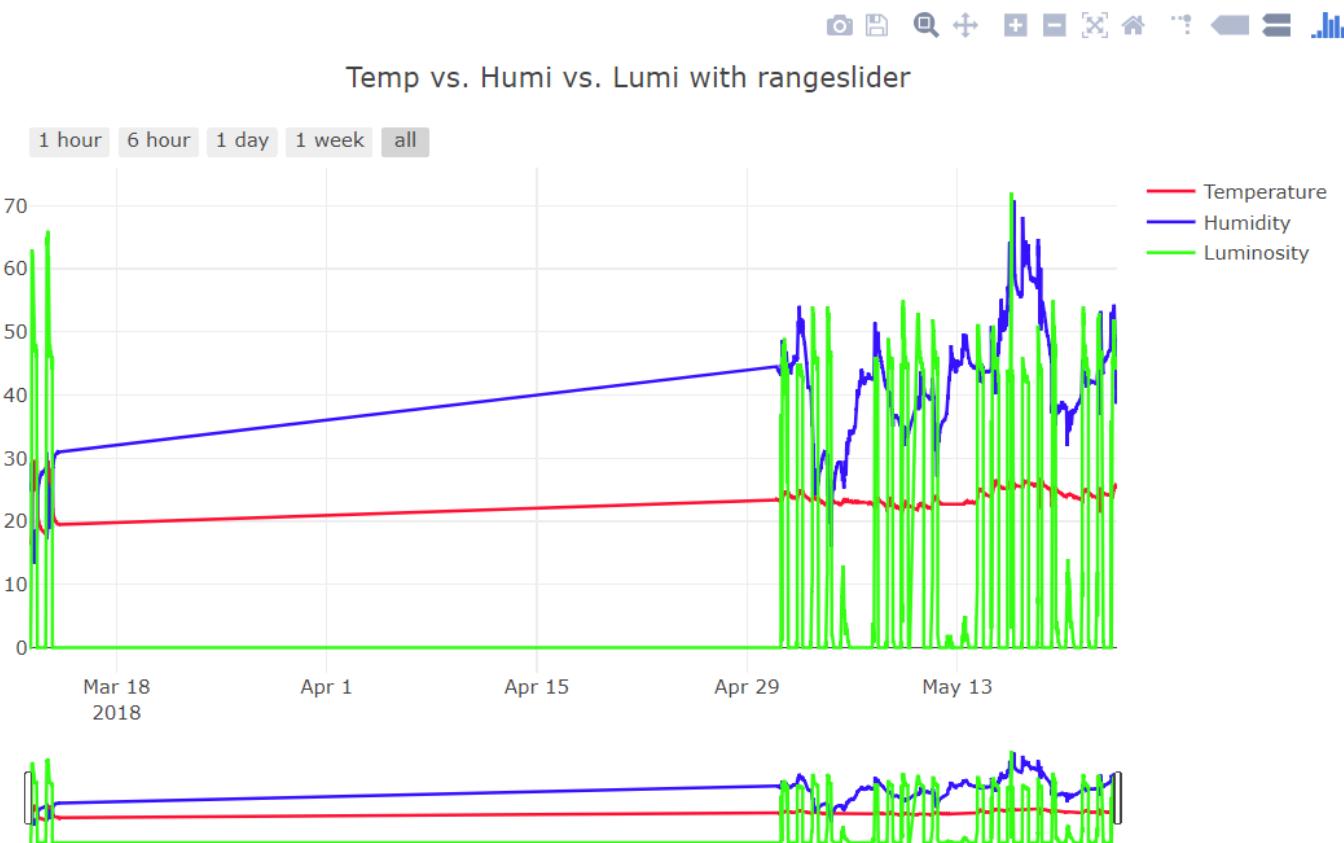


# DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring

MongoDB database visualization by AA00

Time series : Multi sensor data

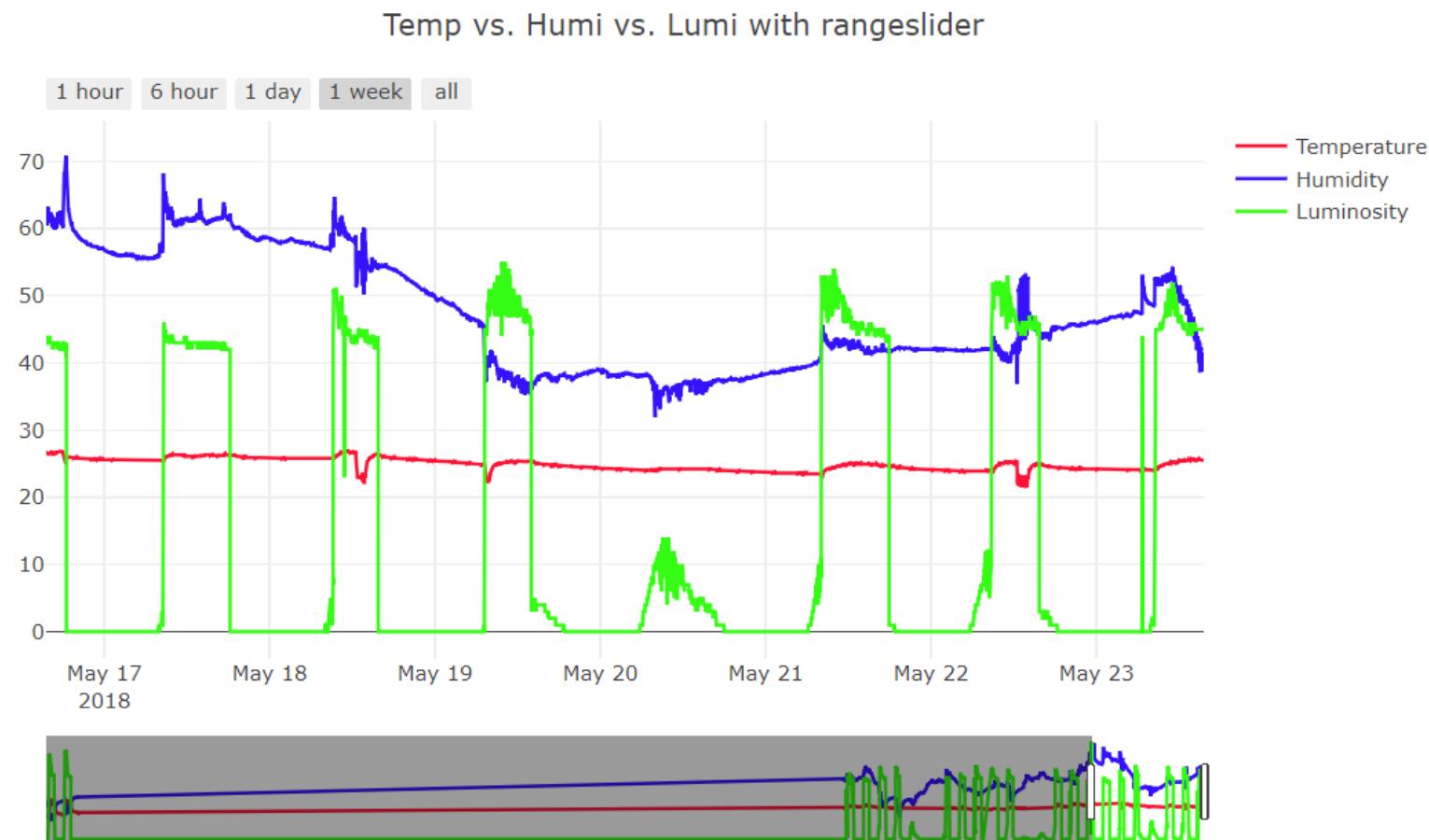


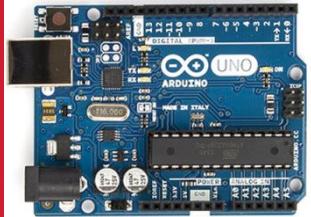


# DHT22 + CdS + Node.js + MongoDB

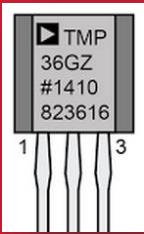
[Next week] Web monitoring

## Time series : Multi sensor data



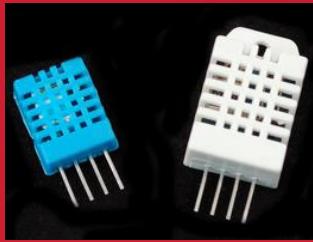


# [Practice]



## ◆ [wk12]

- RT Data storing 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](mailto:chaos21c@gmail.com)

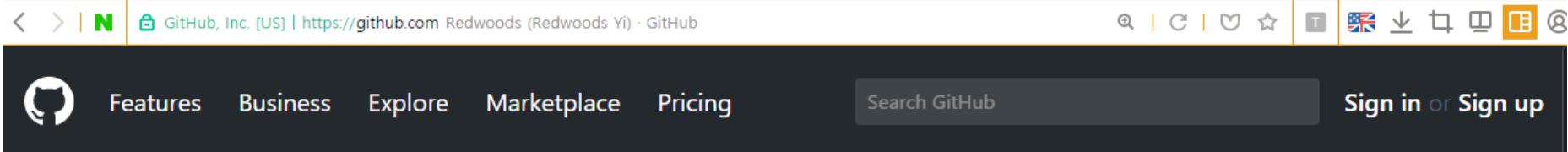
[ 제목 : **id**, 이름 (수정) ]

# Lecture materials

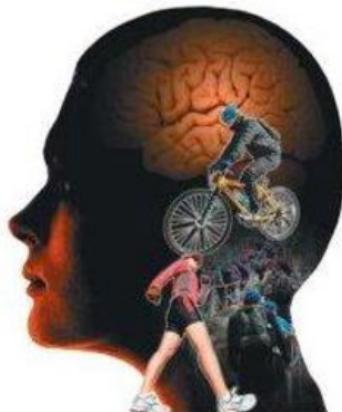


## ● References & good sites

- ✓ <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



A screenshot of a GitHub user profile page. At the top, there's a navigation bar with icons for back, forward, and search, followed by the URL "GitHub, Inc. [US] | https://github.com Redwoods (Redwoods Yi) - GitHub". To the right are icons for search, refresh, notifications, and account management. Below the bar, the GitHub logo is on the left, followed by links for "Features", "Business", "Explore", "Marketplace", and "Pricing". A search bar says "Search GitHub". On the far right, there are "Sign in or Sign up" buttons.



## 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

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

Last

 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

This repository Search Pull requests Issues Marketplace Explore

Unwatch 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Branch: master Lec / healthcare-signal-iot / Create new file Upload

Redwoods 2018 wk03 upload Latest

..

src	2018 wk03 upload
README.md	2018 wk01 upload
wk01_hs_Intro.pdf	2018 wk01 upload-2
wk01_hs_Intro.pptx	2018 wk01 upload-2
wk02_hs_nodejs.pdf	2018 wk02 upload
wk03_hs_node_express.pdf	2018 wk03 upload

README.md

## Lec : Introduction to Healthcare Signal Visualization

All lectures by Redwoods in Inje University from 2018 and 2017.



# 1.0 What is node.js?

← → ⌂ ⌂ 🔒 안전함 | [https://www.w3schools.com/nodejs/nodejs\\_intro.asp](https://www.w3schools.com/nodejs/nodejs_intro.asp)

HOME CSS JAVASCRIPT SQL PHP BOOTSTRAP HOW TO JQUERY MORE ▾

Node.js Tutorial  
Node.js HOME  
**Node.js Intro**  
Node.js Get Started  
Node.js Modules  
Node.js HTTP Module  
Node.js File System  
Node.js URL Module  
Node.js NPM  
Node.js Events  
Node.js Upload Files  
Node.js Email

Node.js MySQL  
MySQL Get Started  
MySQL Create Database  
MySQL Create Table

## Node.js Introduction

◀ Previous

### What is Node.js?

- Node.js is an open source server framework
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server

### Why Node.js?

**Node.js uses asynchronous programming!**

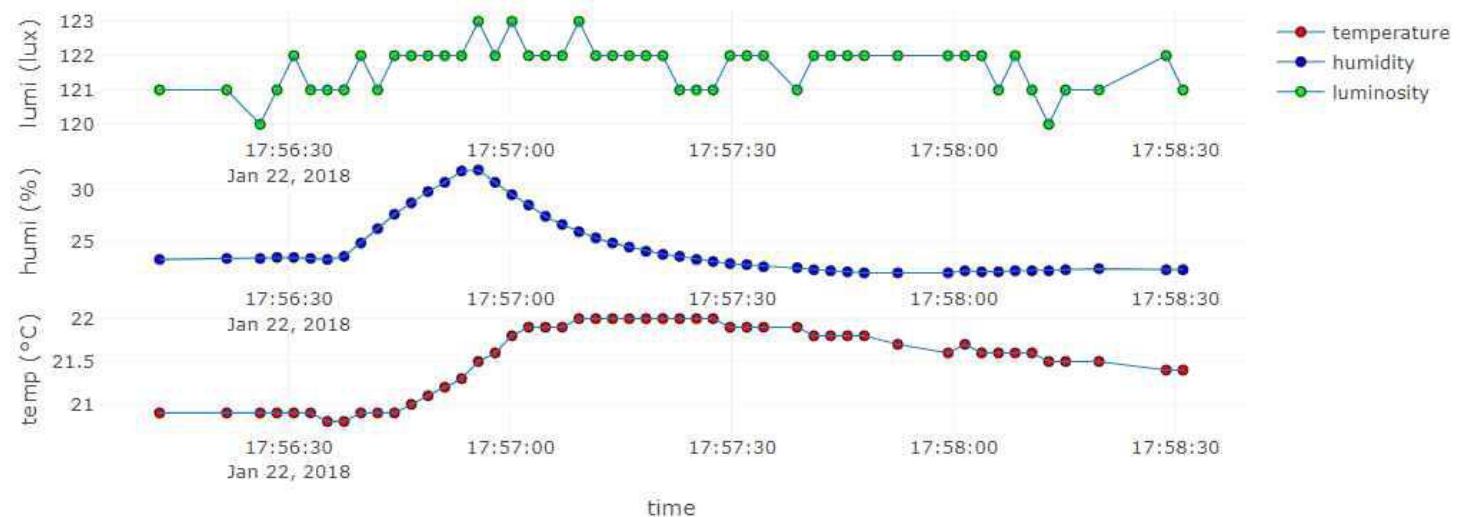
[https://www.w3schools.com/nodejs/nodejs\\_intro.asp](https://www.w3schools.com/nodejs/nodejs_intro.asp)

# Target of this class

## Real-time Weather Station from sensors

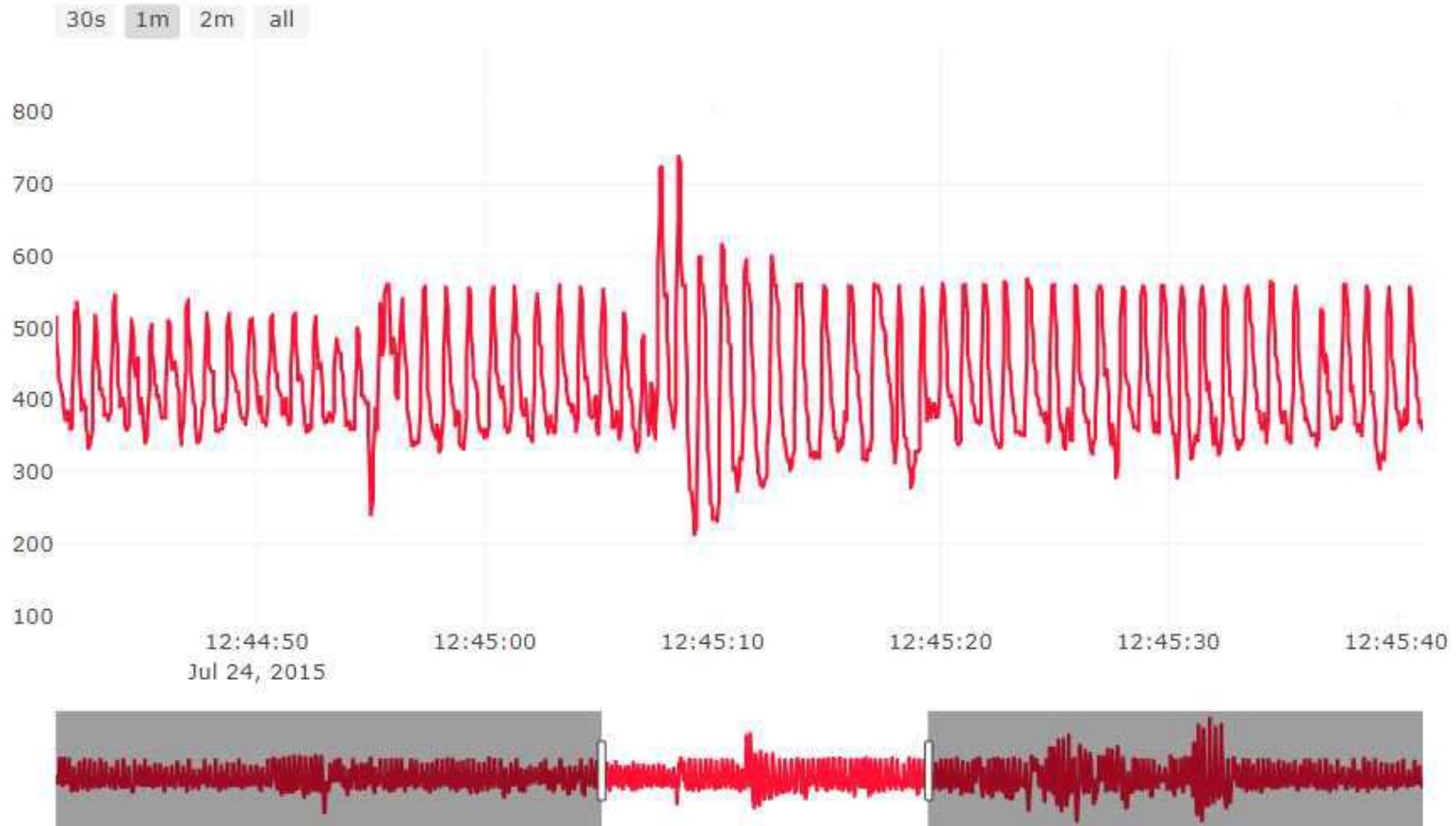


on Time: 2018-01-22 17:58:31.012



# Project of this class

PPG with rangeslider





# 주교재

아두이노와 Node.js에 기반한

IOT 신호 시각화

| 저자 이상훈 |

인제대학교 출판부

## 아두이노와 Node.js에 기반한 IOT 신호 시각화

| 저자 이상훈 |



인제대학교 출판부