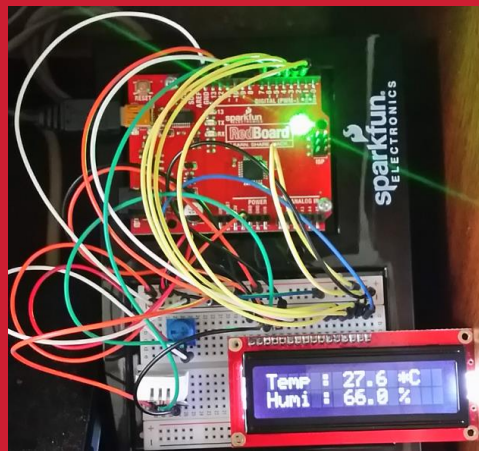
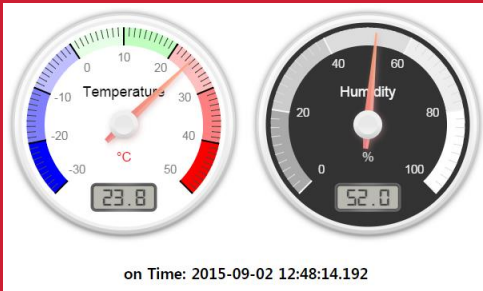




Healthcare-IoT

[wk03]

Node.js Server

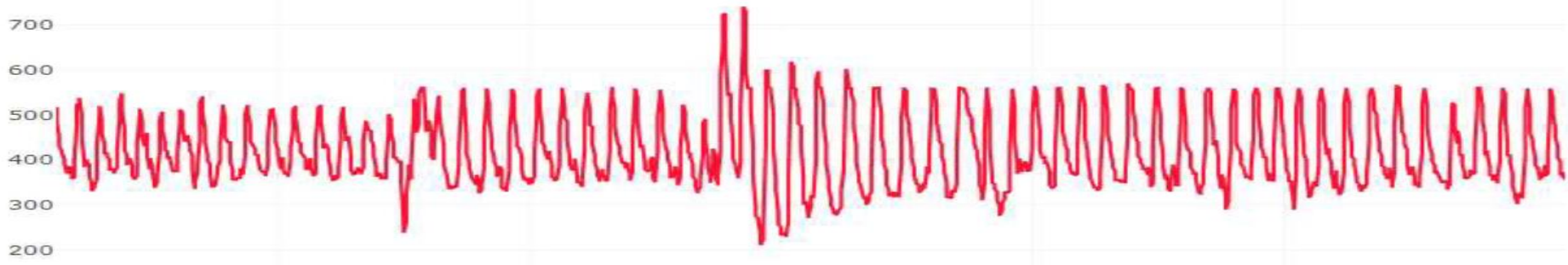


Visualization of Healthcare Signals using
Arduino & Node.js

HCit, INJE University

1st semester, 2018

Email : chaos21c@gmail.com





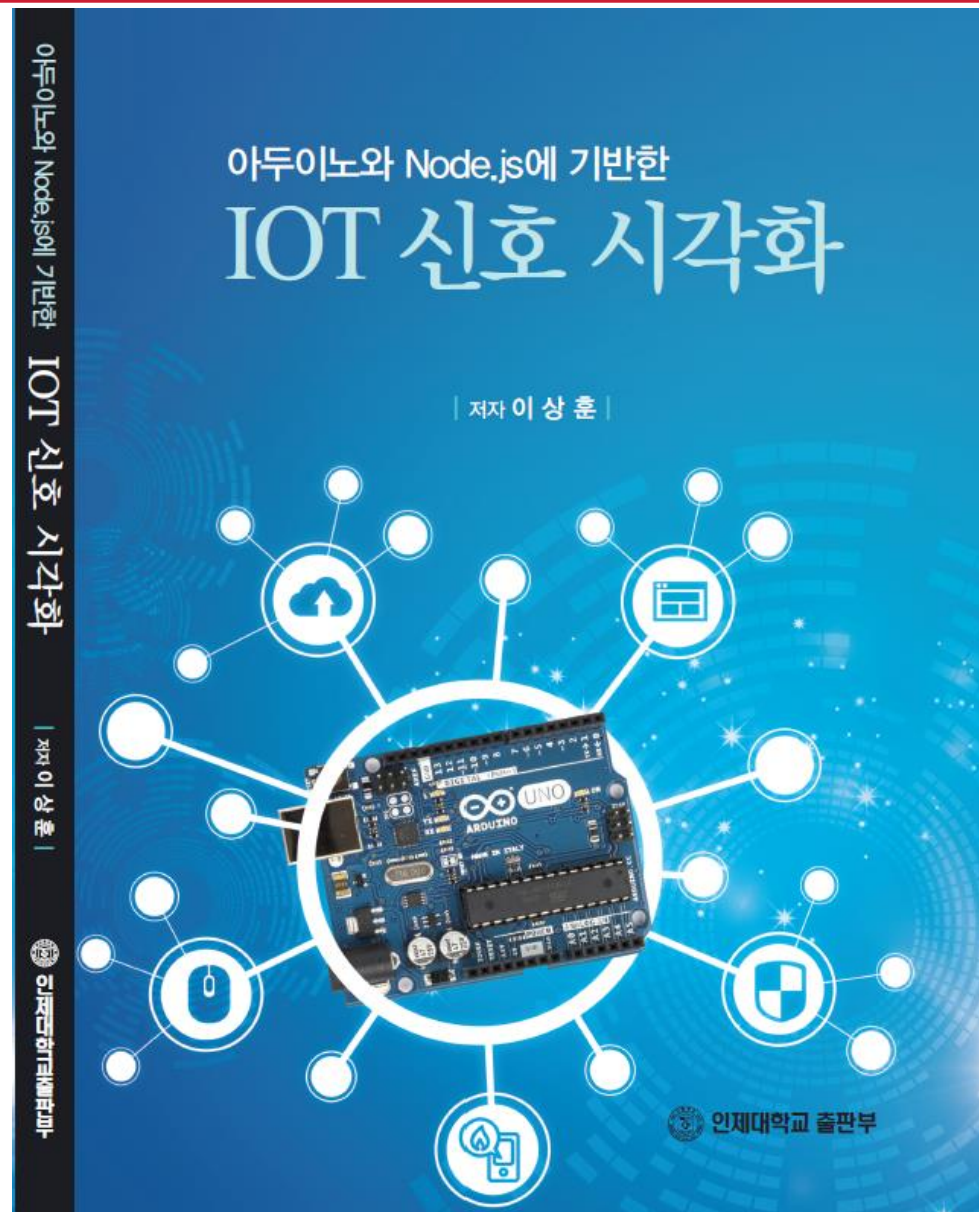
My ID

오전

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

오후

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



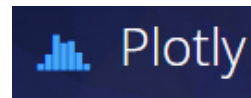
주간계획서

주간계획서			
주차	수업방법	수업내용	과제물
1	강의/실습	수업 및 실습 안내 - 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	프로젝트I - 생체 센서 회로와 Node.js 연결 - 생체 신호 소개	프로젝트I
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]

◆ [wk02]

- Node module : **hsnninfo.js**
- Upload file name : **HSnn_Rpt01.zip**



[practice] local module : **hsnninfo.js**

index_hsnn.js uses local module **hsnninfo.js** in start subfolder.

D:\Portable\NodeJS\Portable\Data\hs00\start\index_hs00.js (hs00, hs00) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS

- hs00
 - hs00
 - server
 - start
 - /* circle.js
 - /* circle_info.js
 - /* hello.js
 - /* hello_call_module.js
 - /* hello_function.js
 - /* hello_module.js
 - /* hello_mymodule.js
 - /* hs00info.js
 - /* index_hs00.js
 - /* package.json

```
1 // index_hsnn.js
2
3 var myinfo = require('./hs00info');
4
5 myinfo("hs00", "Redwoods", '010-1234-5678');
6
7 myinfo("hs77", "HCit", '010-5678-1234');
```

My Info
ID : hs00
Name : Redwoods
Phone : 010-1234-5678

My Info
ID : hs77
Name : HCit
Phone : 010-5678-1234

[Finished in 0.2s]

Save as
HSnn_info.png



[practice] local module : hsninfo.js

How to make hsninfo.js in start subfolder.

1. Make local module – hsninfo.js
2. Call hsninfo.js from index_hsn.js.
3. Capture your result.

```
hs00info.js  x  index_hs00.js  x
1 // hsninfo.js
2
3 module.exports = function(id, name, phone) {
4     console.log("My Info");
5     console.log("ID : " + id);
6     console.log("Name : " + name);
7     console.log("Phone : " + phone + "\n");
8 }
```

[\[참고\] Node local module 만들기](#)



1.0 What is node.js?

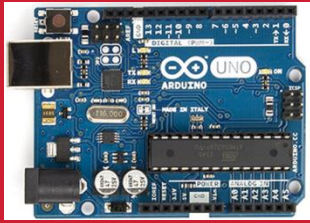




2.2 Install node.js portable



<https://gareth.flowers/nodejs-portable/>



Using node.js

in Sublime text 3

→ Node build system



Node.js Project

npm init



5. Node Apps

Node Apps



5.2.3 Using local module – hello_mymodule.js

D:\Portable\NodeJS\Portable\Data\hs00\start\hello_mymodule.js (hs00, hs00) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS

- hs00
 - hs00
 - start
 - hello.js
 - hello_function.js
 - hello_module.js
 - hello_mymodule.js
 - package.json

```
1 // hello_mymodule.js
2
3 module.exports = function(what) {
4     console.log("Hello " + what + "!");
5 }
6
```

사용자 정의 모듈

FOLDERS

- hs00
 - hs00
 - start
 - hello.js
 - hello_call_module.js
 - hello_function.js
 - hello_module.js
 - hello_mymodule.js
 - package.json

```
1 // hello_call_module.js
2
3 var olleh = require('./hello_mymodule');
4
5 olleh("HCit");
6 olleh("hs00");
```

Hello HCit!
Hello hs00!
[Finished in 0.2s]

Call local module 'hello_mymodule.js' from hello_call_module.js



Node.js Server

1. http, tcp, file

2. Express



6. Node Server

Node Server I.

- 1. HTTP server**
- 2. TCP server**
- 3. File upload**



6.1.1 http server

D:\Portable\Node\SPortable\Data\hs00\server\HTTP\index.js (hs00, hs00) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS

- ▶ hs00
- ▼ hs00
 - ▶ server
 - ▶ File
 - ▼ HTTP
 - /* index.js
 - ▶ TCP
 - ▶ start

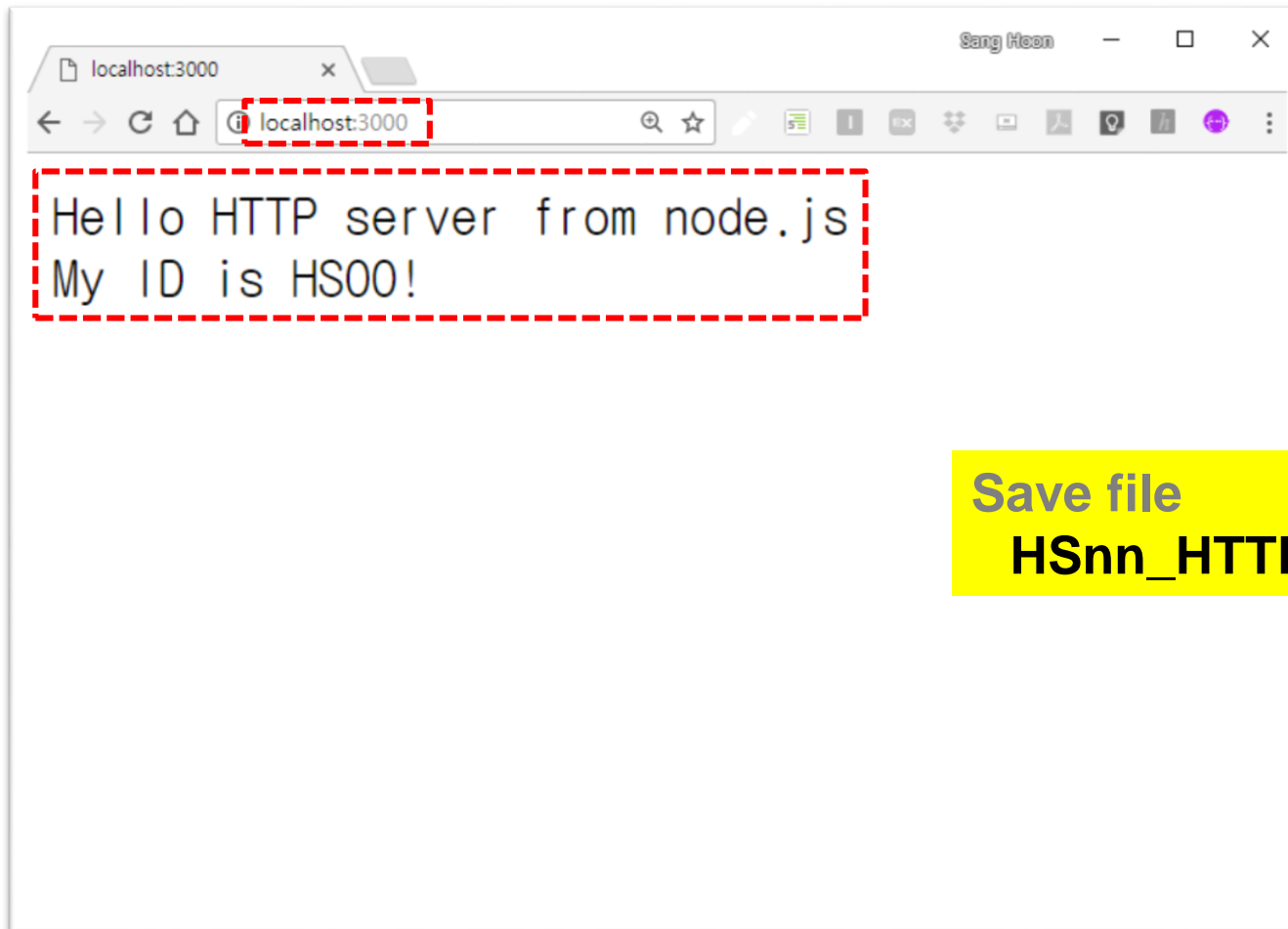
```
1 // http server : index.js
2
3 var http = require('http');
4 port = 3000;
5
6 var server = http.createServer(function(request, response) {
7     response.writeHead(200, {
8         "Content-Type": "text/plain"
9     });
10    response.write("Hello HTTP server from node.js"); // WEB response
11    response.write("\nMy ID is HS00!");
12    response.end();
13
14 });
15
16 server.listen(port);
17 console.log("Server Running on " + port +
18     ".\nLaunch http://localhost:" + port);
19
```

Server Running on 3000.
Launch http://localhost:3000



6.1.2 http server : result 1

Server Running on 3000.
Launch `http://localhost:3000`



Save file
HSnn_HTTP.png



7. Node Server

Node Server II.

- 1. Express server**
- 2. Full Express App**
- 3. My Express App**



7.1.1 Express server test

Step 1 : npm init

Step 2 : npm install --save express

Step 3 : Write Express code

Step 4 : Run app.js

Step 5 : http://localhost:3000



7.1.2 Express server test

1. Make folder expressTest

2. Go to the folder, "expressTest"

3. npm init

NodeJS

```
D:\Portable\NodeJSPortable\Data>cd hs00
```

```
D:\Portable\NodeJSPortable\Data\hs00>dir
```

```
D 드라이브의 볼륨: DATA  
볼륨 일련 번호: 7A01-106A
```

```
D:\Portable\NodeJSPortable\Data\hs00 디렉터리
```

```
2018-03-14 오후 02:57 <DIR> .  
2018-03-14 오후 02:57 <DIR> ..  
2018-03-14 오후 02:57 <DIR> server  
2018-03-14 오후 04:04 <DIR> start  
0개 파일 0 바이트  
4개 디렉터리 889,974,755,328 바이트 남음
```

```
D:\Portable\NodeJSPortable\Data\hs00>md expressTest
```

```
D:\Portable\NodeJSPortable\Data\hs00>cd expressTest
```

```
D:\Portable\NodeJSPortable\Data\hs00\expressTest>npm init
```



7.1.2 Express server test: npm init

FOLDERS

▶ hs00

▼ hs00

▶ expressTest

▼ server

▶ File

▶ HTTP

▶ TCP

▶ start

```
npm
D:\Portable\NodeJSPortable\Data\hs00\expressTest>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help json' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg> --save' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
name: (expressTest)
Sorry, name can no longer contain capital letters.
name: (expressTest) expresstest
version: (1.0.0)
description: Test express server
entry point: (index.js) app.js
test command:
git repository:
keywords:
author: hs00
license: (ISC) MIT
About to write to D:\Portable\NodeJSPortable\Data\hs00\expressTest\package.json:

{
  "name": "expresstest",
  "version": "1.0.0",
  "description": "Test express server",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "hs00",
  "license": "MIT"
}

Is this ok? (yes)
```




7.1.3 Express server test: package.json

package.json

```
D:\Portable\NodeJS\Portable\Data\hs00\expressTest\package.json (hs00, hs00) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
  hs00
  hs00
    expressTest
      /* package.json
      server
      start

1 {
2   "name": "expresstest",
3   "version": "1.0.0",
4   "description": "Test express server",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "hs00",
10  "license": "MIT"
11 }
12
```




7.1.5 Express server test: express install

package.json

```
1 {
2   "name": "expresstest",
3   "version": "1.0.0",
4   "description": "Test express server",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "hs00",
10  "license": "MIT",
11  "dependencies": {
12    "express": "^4.16.3"
13  }
14 }
```

expressTest 폴더 내의 **node_modules** subfolder에 **express server modules**들이 저장되어 서버와 연동된다. 그리고 **package.json**에 **express** 모듈 정보가 “**dependencies**” 속성에 저장된다.



7.1.6 Express server test: app.js

FOLDERS

- ▶ hs00
- ▼ hs00
 - ▼ expressTest
 - ▶ node_modules
 - /* app.js
 - /* package.json
 - ▼ server
 - ▶ File
 - ▶ HTTP
 - ▶ TCP
 - ▶ start

```
1 // app.js, express server
2 var express = require('express');
3 var app = express();
4 var port = 3000;
5
6 app.get('/', function(req, res) {
7   res.send('<a href="/hello">Hello Page</a>');
8 });
9
10 app.get('/hello', function(req, res) {
11   res.send('Hello hs00');
12 });
13
14 app.get('/hcit', function(req, res) {
15   res.send('Hello HCit!');
16 });
17
18 app.get('/hs00', function(req, res) {
19   res.send('Hello hs00, HCit!  My first express server!!!');
20 });
21
22 var server = app.listen(port, function() {
23   console.log('Listening on port %d', server.address().port);
24 });
```



7.1.7 Express server test: run app.js

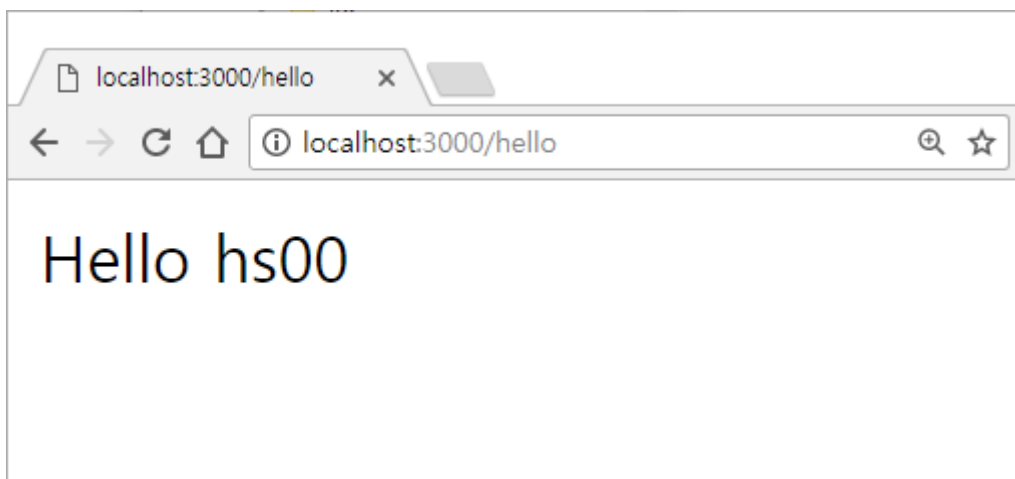
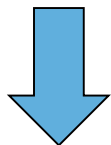
^B (실행 전 Tools > Cancel Build 해제 확인!)

```
1 // app.js, express server
2 var express = require('express');
3 var app = express();
4 var port = 3000;
5
6 app.get('/', function(req, res) {
7   res.send('<a href="/hello">Hello Page</a>');
8 });
9
10 app.get('/hello', function(req, res) {
11   res.send('Hello hs00');
12 });
13
14 app.get('/hcit', function(req, res) {
15   res.send('Hello HCit!');
16 });
17
18 app.get('/hs00', function(req, res) {
19   res.send('Hello hs00, HCit! My first express server!!!');
20 });
21
22 var server = app.listen(port, function() {
23   console.log('Listening on port %d', server.address().port);
24 });
25
```

Listening on port 3000



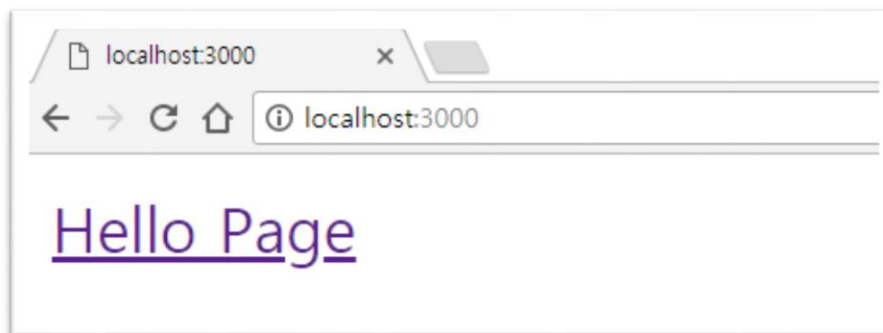
7.1.8 Express server test: run `app.js`



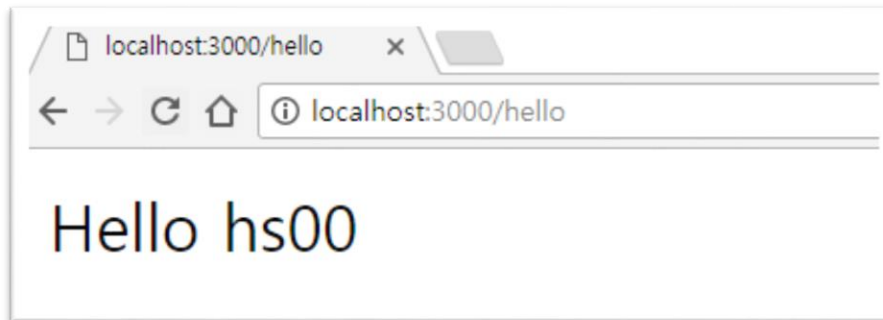


7.1.9 Express server test: test server **routing**

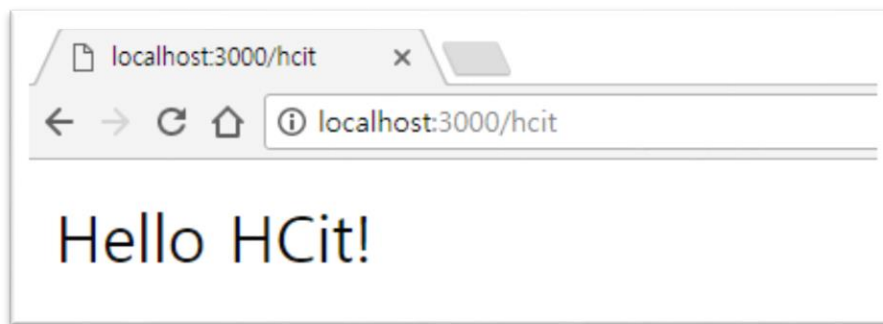
localhost:3000



localhost:3000/hello



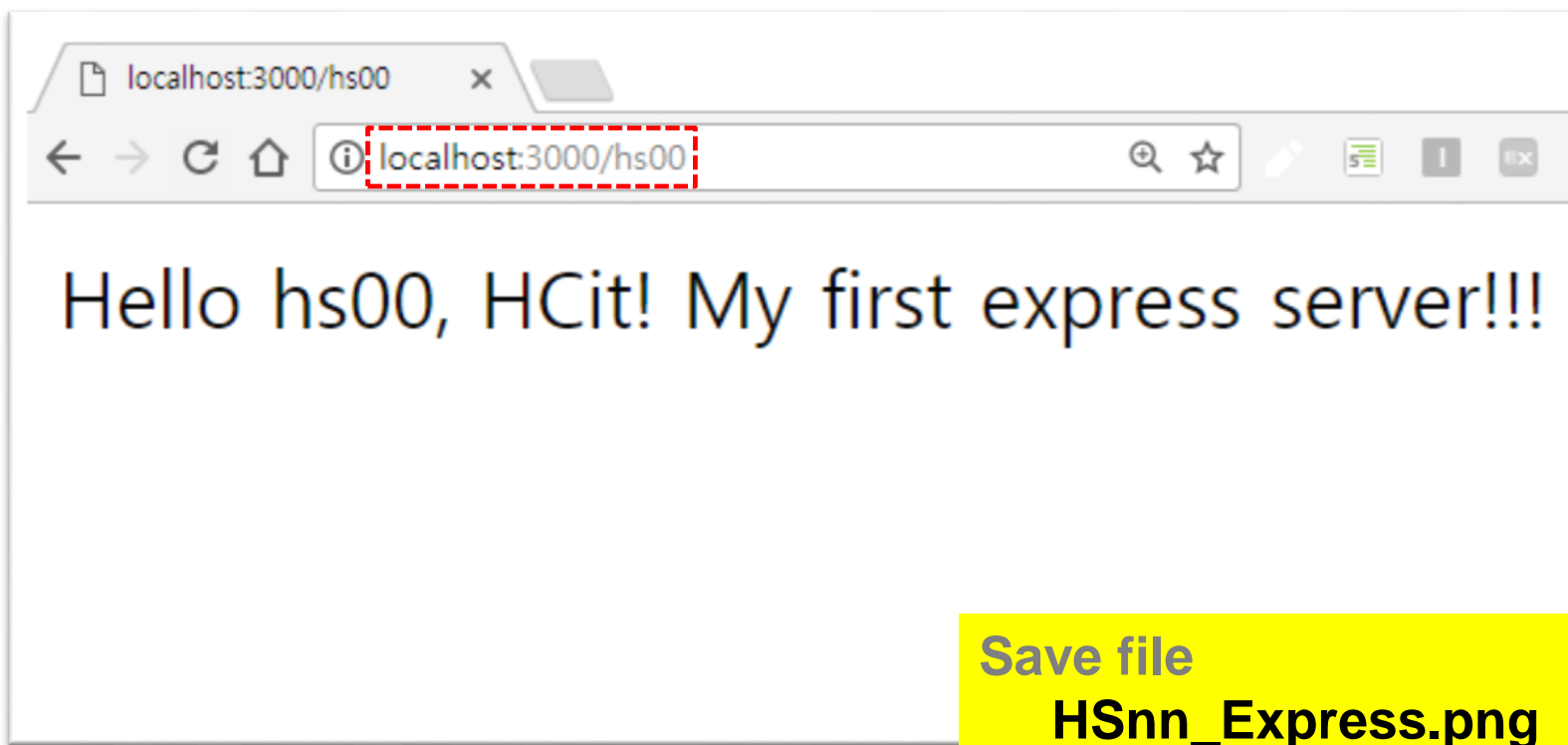
localhost:3000/hcit



^B (실행 전 Tools > Cancel Build 해제 확인!)

[DIY] My ID routing → localhost:hsnn

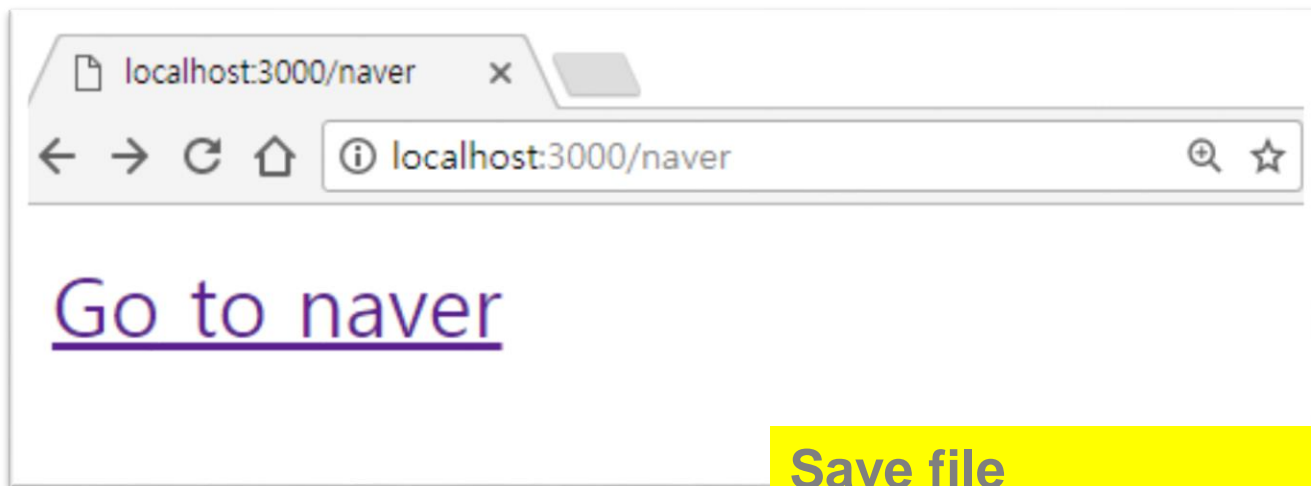
```
app.get('/hs00', function(req, res) {
  res.send('Hello hs00, HCit! My first express server!!!');
});
```



Save file
HSnn_Express.png

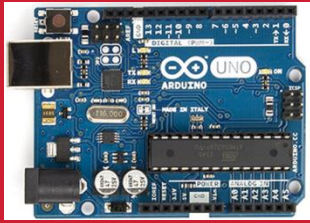
Routing: 라우팅은 애플리케이션 엔드 포인트(**URI**)의 정의, 그리고 **URI**가 클라이언트 요청에 응답하는 방식

[DIY] Go to naver.com



Save file
HSnn_naver.png

[Hint] `Go to naver`



Full Express App using generator



7.2.1 Full Express App using generator

Step 1 : make folder 'express'

Step 2 : go to the folder

Step 3 : `npm install -g express-generator`

Step 4 : `express`

Step 5 : `npm install`

Step 6 : `node ./bin/www`

Step 6 : <http://localhost:3000>



7.2.2 Install Express-generator

1. **Make folder express**
2. **Go to the folder, “express”**
3. **npm install -g express-generator**
 - **express**

```
D:\Portable\NodeJSPortable\Data\hs00>cd ..  
D:\Portable\NodeJSPortable>Data\hs00>md express  
D:\Portable\NodeJSPortable\Data\hs00>cd express  
D:\Portable\NodeJSPortable\Data\hs00\express>npm install -g express-generator  
loadRequestedDeps → fetch ? ??????????????????????????????????????????????????????????????  
loadRequestedDeps → netwo ? ??????????????????????????????????????????????????????????????  
loadRequestedDeps → netwo ? ??????????????????????????????????????????????????????????????  
loadRequestedDeps → netwo ? ??????????????????????????????????????????????????????????????  
loadRequestedDeps → after ? ??????????????????????????????????????????????????????????????  
loadDep:mkdirp → fetchNam ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → r ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → 2 ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → f ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → a ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → a ? ??????????????????????????????????????????????????????????????  
loadDep:sorted-object → f ? ??????????????????????????????????????????????????????????????  
loadDep:brace-expansion → ? ??????????????????????????????????????????????????????????????  
loadDep:brace-expansion → ? ??????????????????????????????????????????????????????????????  
loadDep:brace-expansion → ? ??????????????????????????????????????????????????????????????  
loadDep:brace-expansion → ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → requ ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → 200 ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → fetc ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → netw ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → 200 ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → fetc ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → addl ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → afte ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → netw ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → netw ? ??????????????????????????????????????????????????????????????  
loadDep:concat-map → afte ? ??????????????????????????????????????????????????????????????  
loadDep:minimist → reques ? ??????????????????????????????????????????????????????????????  
loadDep:minimist → 200 ? ??????????????????????????????????????????????????????????????  
extract → gunzTarPerm ? ??????????????????????????????????????????????????????????????  
preinstall → lifecycle ? ??????????????????????????????????????????????????????????????  
finalize:minimatch → fina ? ??????????????????????????????????????????????????????????????  
D:\Portable\NodeJSPortable\Data\express → D:\Portable\NodeJSPortable\Data\node_modules\ex  
press-generator\bin\express-cli.js  
D:\Portable\NodeJSPortable\Data  
├─┬ express-generator@4.16.0  
│ └─ commander@2.13.0  
│   └─ minimatch@3.0.4  
│     ├── brace-expansion@1.1.11  
│     │ └─ balanced-match@1.0.0  
│     └─ concat-map@0.0.1
```



7.2.3 Run Express-generator : express

C:\ NodeJS

D:\Portable\NodeJSPortable\Data\hs00\express>express

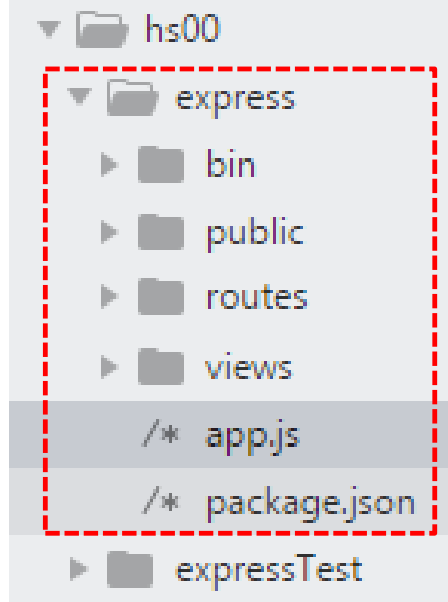
warning: the default view engine will not be jade in future releases
warning: use '--view=jade' or '--help' for additional options

```
create : public\  
create : public\javascripts\  
create : public\images\  
create : public\stylesheets\  
create : public\stylesheets\style.css  
create : routes\  
create : routes\index.js  
create : routes\users.js  
create : views\  
create : views\error.jade  
create : views\index.jade  
create : views\layout.jade  
create : app.js  
create : package.json  
create : bin\  
create : bin\www
```

```
install dependencies:  
> npm install
```

```
run the app:  
> SET DEBUG=express:* & npm start
```

D:\Portable\NodeJSPortable\Data\hs00\express>



‘express’ 명령은 현재 폴더에 Express 서버를 구성해서 초기화한다.



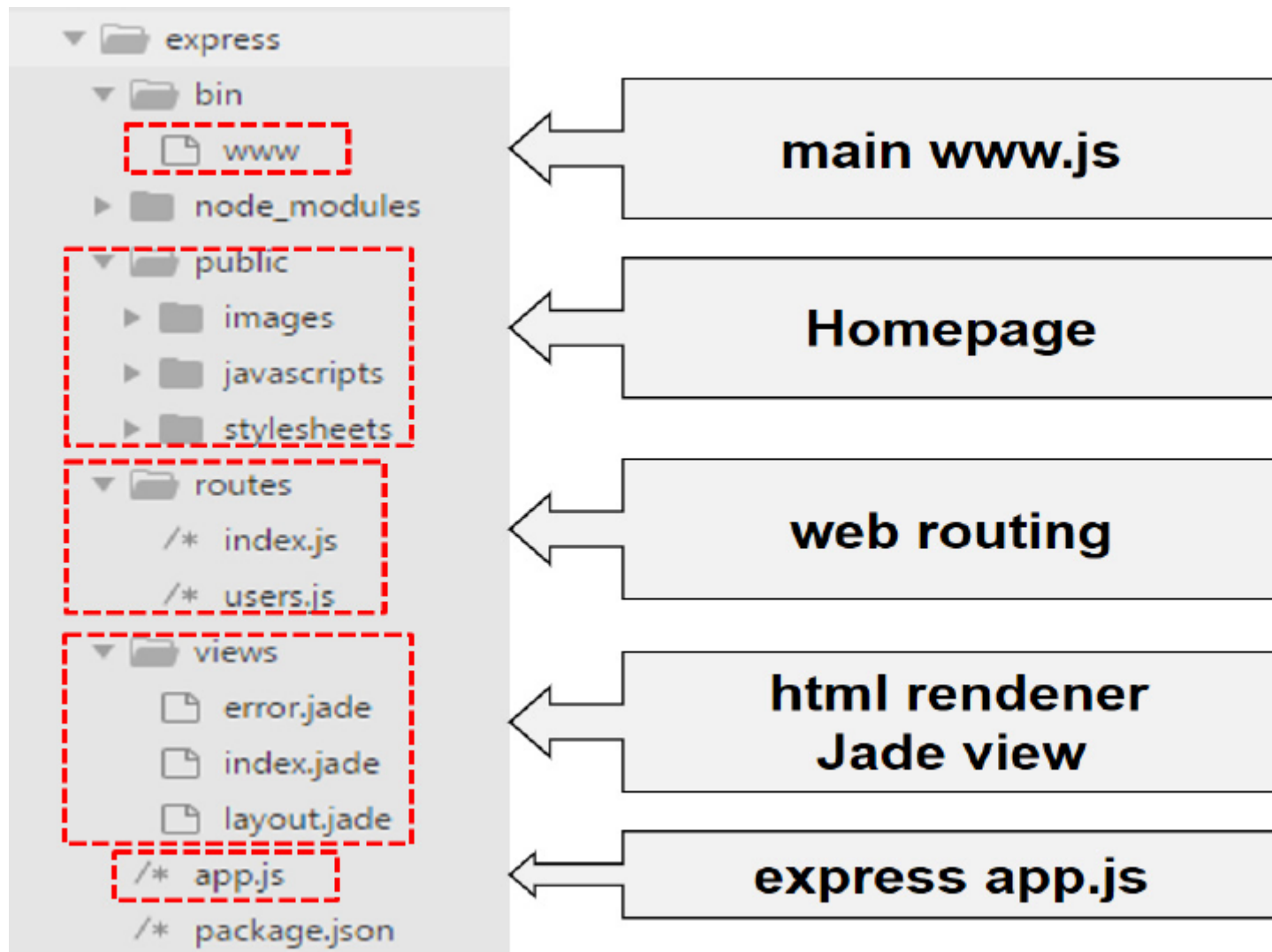
7.2.4 package.json

```
1 {
2   "name": "express",
3   "version": "0.0.0",
4   "private": true,
5   "scripts": {
6     "start": "node ./bin/www"
7   },
8   "dependencies": {
9     "cookie-parser": "~1.4.3",
10    "debug": "~2.6.9",
11    "express": "~4.16.0",
12    "http-errors": "~1.6.2",
13    "jade": "~1.11.0",
14    "morgan": "~1.9.0"
15  }
16 }
```

1. Install node modules that is defined in package.json

2. npm install

7.2.6 Inner structure of express server





7.2.7 run bin/www

1. Install node modules that is defined in package.json
2. npm install
3. Modules were installed in node_modules subfolder.

4. Run node express web

node ./bin/www

or

npm start



7.2.8 test app

cmd: NodeJS - node ./bin/www

```
D:\Portable\NodeJSPortable\Data\hs00\express>node ./bin/www
```

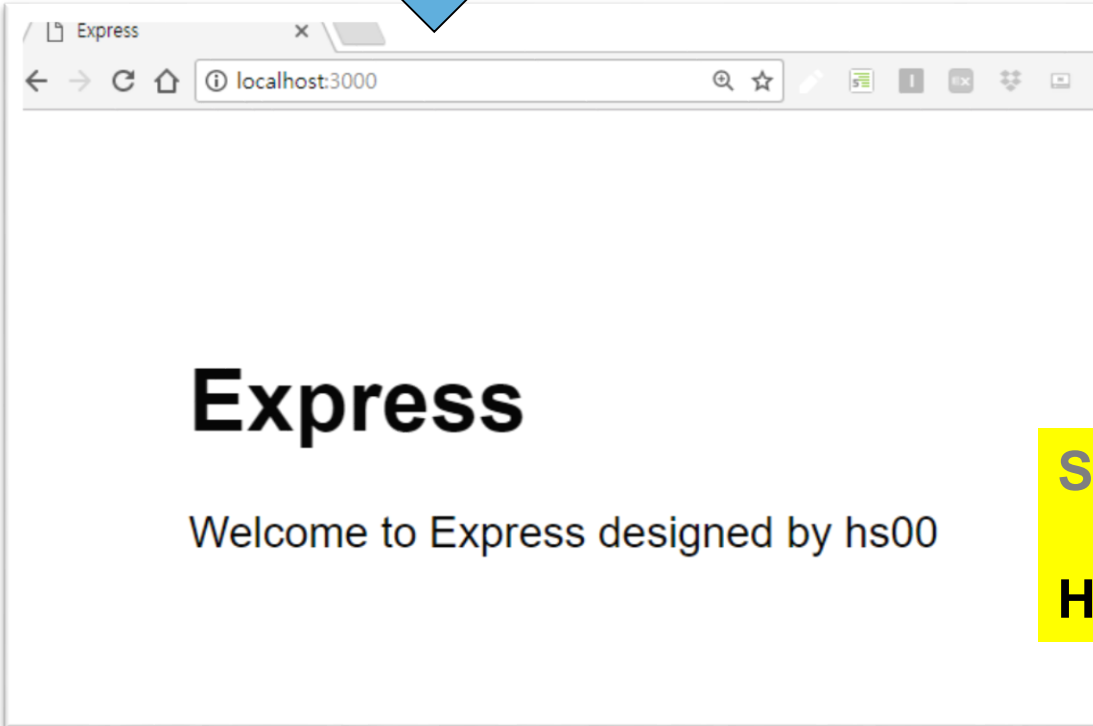
^C

```
D:\Portable\NodeJSPortable\Data\hs00\express>node ./bin/www
```

```
Listening on port: 3000
```

```
GET / 200 296.724 ms - 170
GET /stylesheets/style.css 200 2.919 ms - 111
GET / 200 17.681 ms - 187
GET /stylesheets/style.css 304 0.996 ms - -
GET / 304 25.455 ms - -
GET /stylesheets/style.css 304 0.459 ms - -
```

Web log



Save file

HSnn_Express_App.png



7.2.9 app.js - 1

```
app.js x
1 var createError = require('http-errors');
2 var express = require('express');
3 var path = require('path');
4 var cookieParser = require('cookie-parser');
5 var logger = require('morgan');
6
7 var indexRouter = require('./routes/index');
8 var usersRouter = require('./routes/users');
9
10 var app = express();
11
12 // view engine setup
13 app.set('views', path.join(__dirname, 'views'));
14 app.set('view engine', 'jade');
15
16 app.use(logger('dev'));
17 app.use(express.json());
18 app.use(express.urlencoded({ extended: false }));
19 app.use(cookieParser());
20 app.use(express.static(path.join(__dirname, 'public')));
21
22 app.use('/', indexRouter);
23 app.use('/users', usersRouter);
24
```

routing

View engine
jade

Set routing



7.2.9 app.js - 2

```
app.js x
25 // catch 404 and forward to error handler
26 app.use(function(req, res, next) {
27   next(createError(404));
28 });
29
30 // error handler
31 app.use(function(err, req, res, next) {
32   // set locals, only providing error in development
33   res.locals.message = err.message;
34   res.locals.error = req.app.get('env') === 'development' ? err : {};
35
36   // render the error page
37   res.status(err.status || 500);
38   res.render('error');
39 });
40
41 module.exports = app;
42
```



7.2.10 ./bin/www.js - 1

```
1 #!/usr/bin/env node
2
3 /**
4  * Module dependencies.
5  */
6
7 var app = require('../app');
8 var debug = require('debug')('express:server');
9 var http = require('http');
10
11 /**
12  * Get port from environment and store in Express.
13  */
14
15 var port = normalizePort(process.env.PORT || '3000');
16 app.set('port', port);
17
18 /**
19  * Create HTTP server.
20  */
21
22 var server = http.createServer(app);
23
```

app.js
모듈로
구성된
http
서버를
만든다.



7.2.10 ./bin/www.js - 2

```
24 ▾ /**
25  * Listen on provided port, on all network interfaces.
26  */
27
28  server.listen(port);
29  server.on('error', onError);
30  server.on('listening', onListening);
31  console.log("Listening on port: " + port);
32
33 ▾ /**
34  * Normalize a port into a number, string, or false.
35  */
36
37 ▾ function normalizePort(val) {
38   var port = parseInt(val, 10);
39
40   if (isNaN(port)) {
41     // named pipe
42     return val;
43   }
44
45   if (port >= 0) {
46     // port number
47     return port;
48   }
49
50   return false;
51 }
```

실행 화면 출력을 추가

```
53 /**
54  * Event listener for HTTP server "error" event.
55  */
56
57 function onError(error) { ...
79 }
80
81 /**
82  * Event listener for HTTP server "listening" event.
83  */
84
85 function onListening() { ...
92 }
93
```




7.2.11 routes/index.js

웹 브라우저에서 “**http://localhost:3000**”으로 연결하면 기본 웹 링크인 ‘/’에 지정된 **javascript** 코드인 **index.js**가 실행된다.

routes/index.js

```
index.js x
1 var express = require('express');
2 var router = express.Router();
3
4 /* GET home page. */
5 router.get('/', function(req, res, next) {
6   res.render('index', { title: 'Express' });
7 });
8
9 module.exports = router;
10
```

index.js에는 기본 웹페이지에서 보여주는 페이지에 대한 시각화(**renderer**)가 **views** 폴더의 **index.jade**로 지정된다

7.2.12 views/index.jade

index.jade는 웹페이지 화면의 기본 레이아웃 설정 파일인 **layout.jade**를 이용한다.

그리고 오류가 발생되면 오류를 표현하는 설정은 **error.jade**를 이용한다.

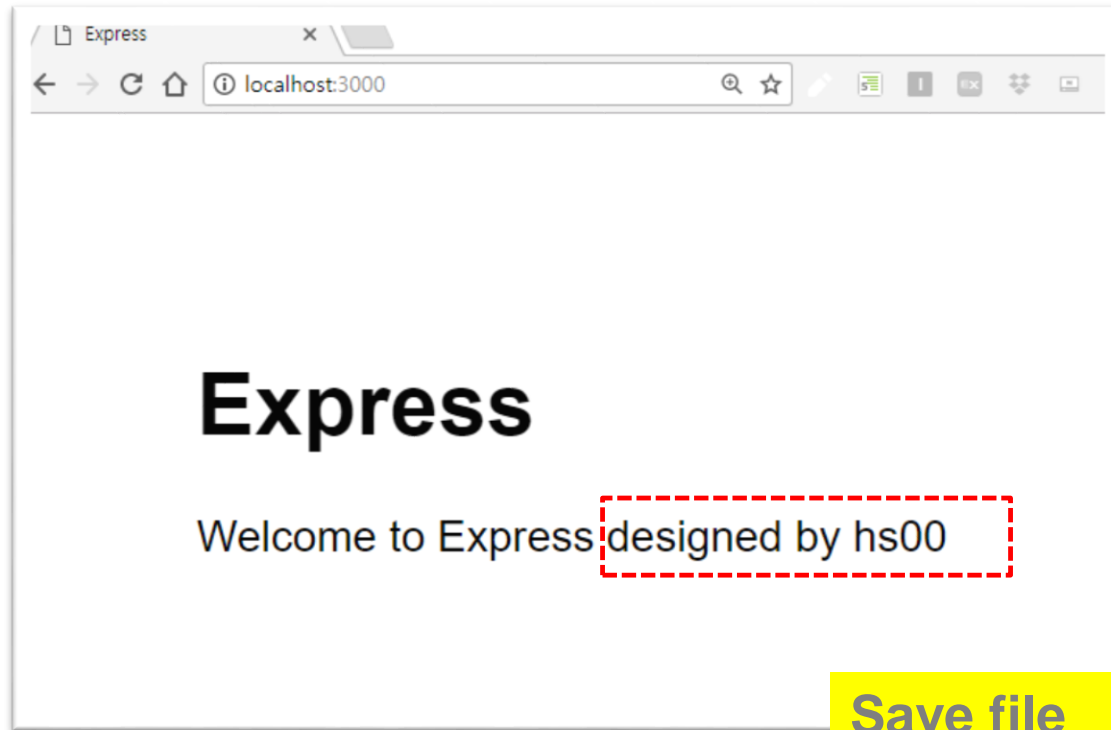
```

index.jade
1 extends layout
2
3 block content
4   h1= title
5   p Welcome to #{title} designed by hs00
6

layout.jade
1 doctype html
2 html
3   head
4     title= title
5     link(rel='stylesheet', href='/stylesheets/style.css')
6   body
7     block content
  
```



7.2.13 result ← localhost:3000

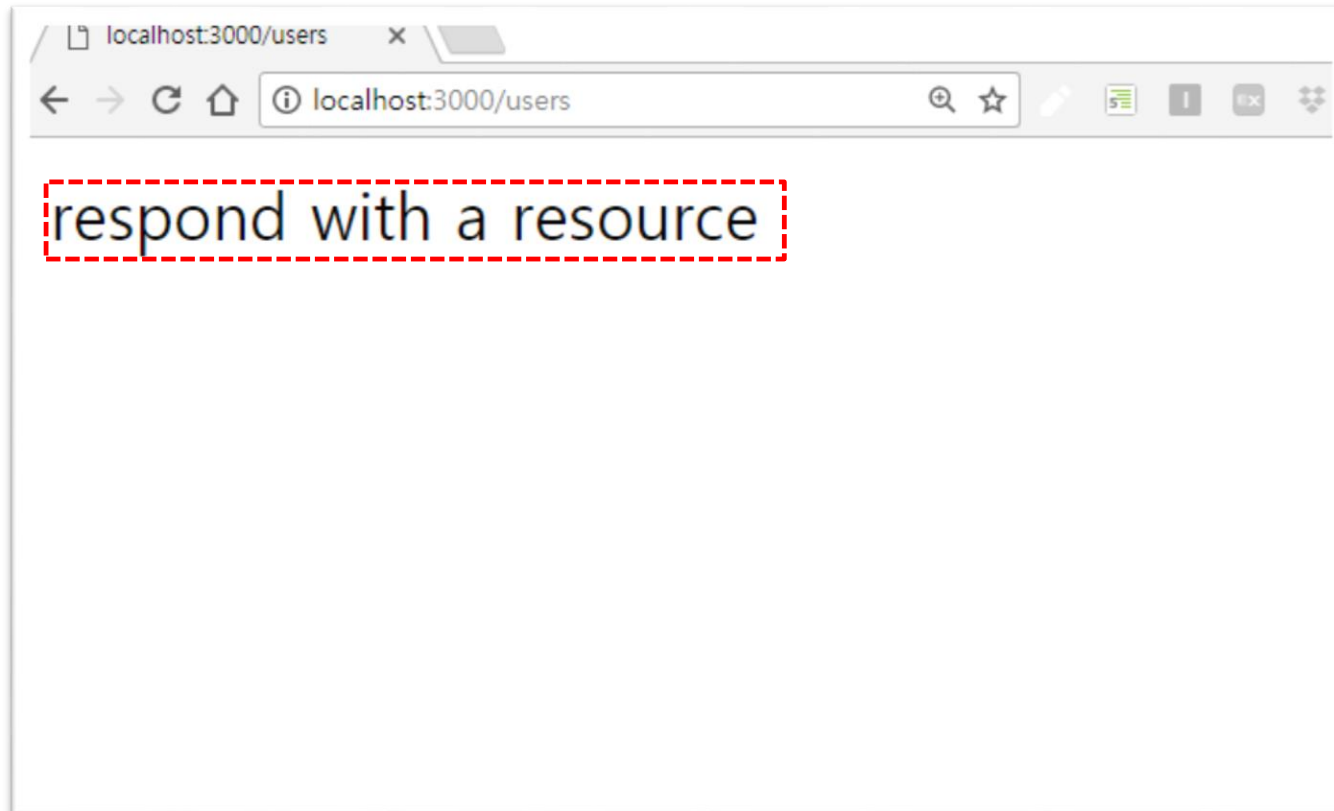


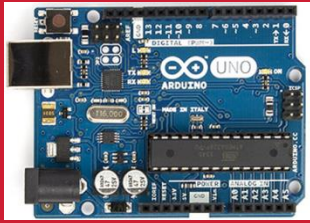
Save file

HSnn_Express_App.png



7.2.14 result ← localhost:3000/users





My Express App using generator



Express options

- e, --ejs** add ejs engine support (defaults to **jade**)
- hbs** add handlebars engine support
- H, --hogan** add hogan.js engine support
- c, --css <engine>** add stylesheet <engine> support (less|stylus|compass)
- f, --force** force on non-empty directory<engine></engine>



7.3.1 Templating using Express-generator

express hs00App -e -c less

NodeJS

```
D:\Portable\NodeJSPortable\Data\hs00\express>cd ..
```

```
D:\Portable\NodeJSPortable\Data\hs00>express hs00App -e -c less
```

warning: option '--ejs' has been renamed to '--view=ejs'

```
create : hs00App\
create : hs00App\public\
create : hs00App\public\javascripts\
create : hs00App\public\images\
create : hs00App\public\stylesheets\
create : hs00App\public\stylesheets\style.less
create : hs00App\routes\
create : hs00App\routes\index.js
create : hs00App\routes\users.js
create : hs00App\views\
create : hs00App\views\error.ejs
create : hs00App\views\index.ejs
create : hs00App\app.js
create : hs00App\package.json
create : hs00App\bin\
create : hs00App\bin\www
```

change directory:

```
> cd hs00App
```

install dependencies:

```
> npm install
```

run the app:

```
> SET DEBUG=hs00app:* & npm start
```

```
D:\Portable\NodeJSPortable\Data\hs00>
```

NodeJS

```
D:\Portable\NodeJSPortable\Data\hs00>dir
```

```
D 드라이브의 볼륨: DATA
볼륨 일련 번호: 7A01-106A
```

```
D:\Portable\NodeJSPortable\Data\hs00 디렉터리
```

```
2018-03-21 오후 05:38 <DIR> .
2018-03-21 오후 05:38 <DIR> ..
2018-03-21 오후 03:57 <DIR> express
2018-03-21 오후 02:39 <DIR> expressTest
2018-03-21 오후 05:38 <DIR> hs00App
2018-03-21 오후 04:29 <DIR> myApp
2018-03-14 오후 02:57 <DIR> server
2018-03-14 오후 04:04 <DIR> start
0개 파일 0 바이트
8개 디렉터리 889,951,232,000 바이트 남음
```

```
D:\Portable\NodeJSPortable\Data\hs00>cd hs00App
```

```
D:\Portable\NodeJSPortable\Data\hs00\hs00App>dir
```

```
D 드라이브의 볼륨: DATA
볼륨 일련 번호: 7A01-106A
```

```
D:\Portable\NodeJSPortable\Data\hs00\hs00App 디렉터리
```

```
2018-03-21 오후 05:38 <DIR> .
2018-03-21 오후 05:38 <DIR> ..
2018-03-21 오후 05:38 1,180 app.js
2018-03-21 오후 05:38 326 package.json
2018-03-21 오후 05:38 <DIR> public
2018-03-21 오후 05:38 <DIR> routes
2018-03-21 오후 05:38 <DIR> views
2개 파일 1,506 바이트
6개 디렉터리 889,951,232,000 바이트 남음
```

```
D:\Portable\NodeJSPortable\Data\hs00\hs00App>
```

7.3.2 Templating using Express-generator

express hs00App -e -c less

```

1 {
2   "name": "hs00app",
3   "version": "0.0.0",
4   "private": true,
5   "scripts": {
6     "start": "node ./bin/www"
7   },
8   "dependencies": {
9     "cookie-parser": "~1.4.3",
10    "debug": "~2.6.9",
11    "ejs": "~2.5.7",
12    "express": "~4.16.0",
13    "http-errors": "~1.6.2",
14    "less-middleware": "~2.2.1",
15    "morgan": "~1.9.0"
16  }
17 }

```




7.3.3 Templating using Express-generator

npm install

Ctrl NodeJS

```
├── is@3.2.1
├── morgan@1.9.0
├── basic-auth@2.0.0
└── on-headers@1.0.1
```

D:\Portable\NodeJSPortable\Data\hs00\hs00App>dir

D 드라이브의 볼륨: DATA

볼륨 일련 번호: 7A01-106A

D:\Portable\NodeJSPortable\Data\hs00\hs00App 디렉터리

2018-03-21	오후 05:46	<DIR>	.
2018-03-21	오후 05:46	<DIR>	..
2018-03-21	오후 05:38		1,180 app.js
2018-03-21	오후 05:38	<DIR>	bin
2018-03-21	오후 05:46	<DIR>	node_modules
2018-03-21	오후 05:38		326 package.json
2018-03-21	오후 05:38	<DIR>	public
2018-03-21	오후 05:38	<DIR>	routes
2018-03-21	오후 05:38	<DIR>	views
		2개 파일	1,506 바이트
		7개 디렉터리	889,934,880,768 바이트 남음

D:\Portable\NodeJSPortable\Data\hs00\hs00App>■

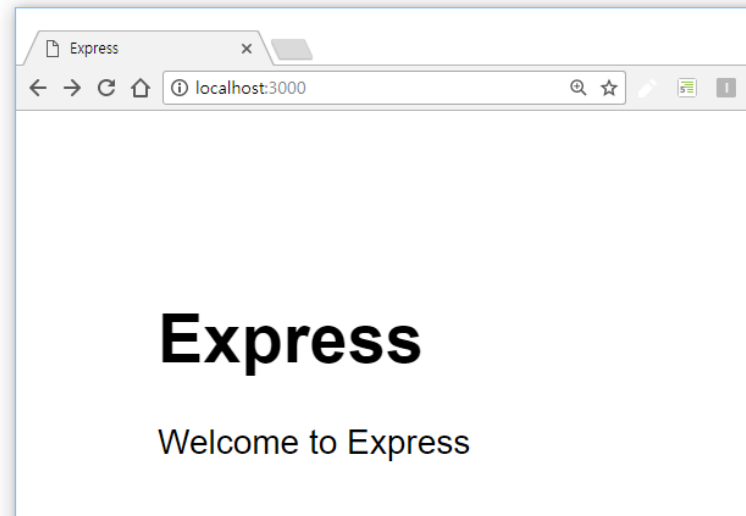


7.3.4 Run template: ejs

npm start or **node ./bin/www**

```
NodeJS - node ./bin/www
D:\Portable\NodeJSPortable\Data\hs00\hs00App>node ./bin/www
GET / 200 12.060 ms - 207
GET /stylesheets/style.css 200 27.866 ms - 87
GET / 304 1.813 ms - -
GET /stylesheets/style.css 304 2.108 ms - -
GET /users 304 1.536 ms - -
GET / 304 0.703 ms - -
GET /stylesheets/style.css 304 0.809 ms - -
```

```
index.ejs
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title><%= title %></title>
5     <link rel='stylesheet' href='/stylesheets/style.css' />
6   </head>
7   <body>
8     <h1><%= title %></h1>
9     <p>Welcome to <%= title %></p>
10  </body>
11 </html>
12
```





[Practice]

◆ [wk03]

- Express App (7.2 Express project)
- Add a new route in **routes/index.js**
- Add a new view: **views/hsnn.jade**
- Upload file name : HSnn_Rpt02.zip



[Practice-1] Modify routes/index.js

Add a new route `/hsnn` in routes/index.js .

FOLDERS

- ▶ hs00
- ▼ hs00
 - ▼ express
 - ▶ bin
 - ▶ node_modules
 - ▶ public
 - ▼ routes
 - /* index.js
 - /* users.js
 - ▶ views
 - /* app.js
 - /* package.json
 - ▶ expressTest
 - ▶ hs00App
 - ▶ myApp
 - ▼ server
 - ▶ File
 - ▶ HTTP
 - ▶ TCP
 - ▶ start

```
index.js x
1  var express = require('express');
2  var router = express.Router();
3
4  /* GET home page. */
5  router.get('/', function(req, res, next) {
6    res.render('index', { title: 'Express' });
7  });
8
9  /* GET my page by /hsnn. -> multi-routing */
10 router.get('/hs00', function(req, res, next) {
11   res.render('hs00', { title: 'Express App',
12                      id: 'HS00',
13                      name: 'Redwoods' });
14   // views/hs00.jade
15 });
16
17 module.exports = router;
```



[Practice-2] Add hsnn.jade in views folder

Add views/**hsnn.jade** in views folder

FOLDERS

- hs00
- hs00
 - express
 - bin
 - node_modules
 - public
 - routes
 - views
 - error.jade
 - hs00.jade
 - index.jade
 - layout.jade
- /* app.js
- /* package.json
- expressTest
- hs00App
- myApp
- server
 - File
 - HTTP
 - TCP
 - start

```
1 extends layout
2
3 block content
4   h1= title
5   p Welcome to #{title}
6   p My id : #{id}
7   center Developed by #{name}
8
```

Express App

localhost:3000/hs00

**Savs as
HSnn_Jade.png**

Express App

Welcome to Express App

My id : HS00

Developed by Redwoods

◆ [Target of this week]

- Complete your works
- Save your outcomes and compress 4 figures

제출파일명 : **HSnn_Rpt02.zip**

- 압축할 파일들

- ① **HSnn_Express.png**
- ② **HSnn_naver.png**
- ③ **HSnn_Express_App.png**
- ④ **HSnn_Jade.png**

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

● References & good sites

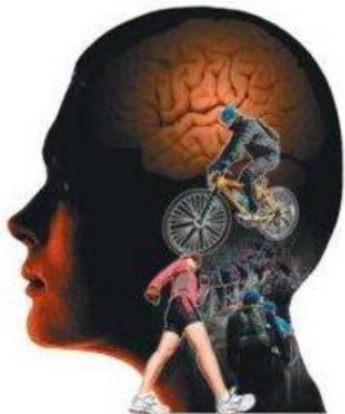
- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling



Features Business Explore Marketplace Pricing

Search GitHub

Sign in or Sign up



Redwoods Yi

Redwoods

Block or report user

📍 GimHae, Republic of Korea

Overview

Repositories 5

Stars 2

Followers 0

Following 0

Pinned repositories

dht22-iot-project

lot 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

Lat

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

Redwoods / Lec Unwatch 1

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#) [Settings](#)

Branch: master [Lec / healthcare-signal-iot /](#) [Create new file](#) [Upload existing file](#)

 Redwoods 2018 wk01 upload Latest	
..	
 src	2018 wk01 upload
 README.md	2018 wk01 upload
 wk01_hs_Intro.pptx	2018 wk01 upload

 [README.md](#)

Lec : Introductionto 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

🏠 HTML 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!

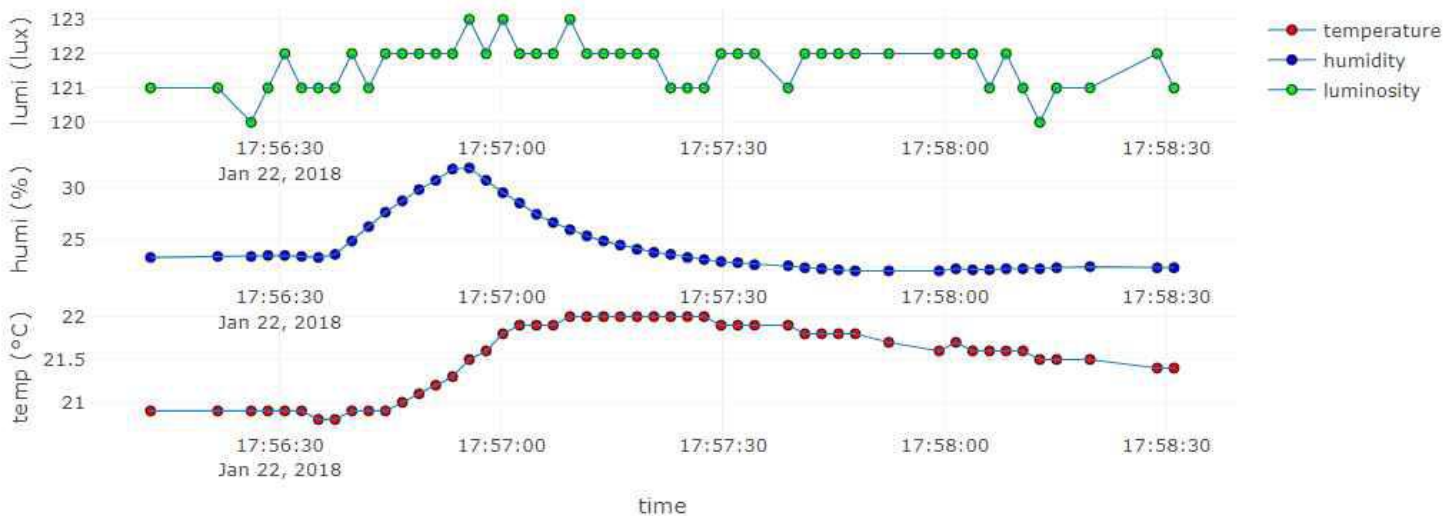
https://www.w3schools.com/nodejs/nodejs_intro.asp

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

