

OTP를 이용한
무인 택배 보관함

Made by 강통코딩

Back-end개발 (1)

수원대학교 정보통신학과 졸업작품 보고서 - 5~6주차

목차

I. Node.js Server

II. Web Application(HTML)

III. 소켓통신(Server – Web)

IV. Mosquitto Server(MQTT)

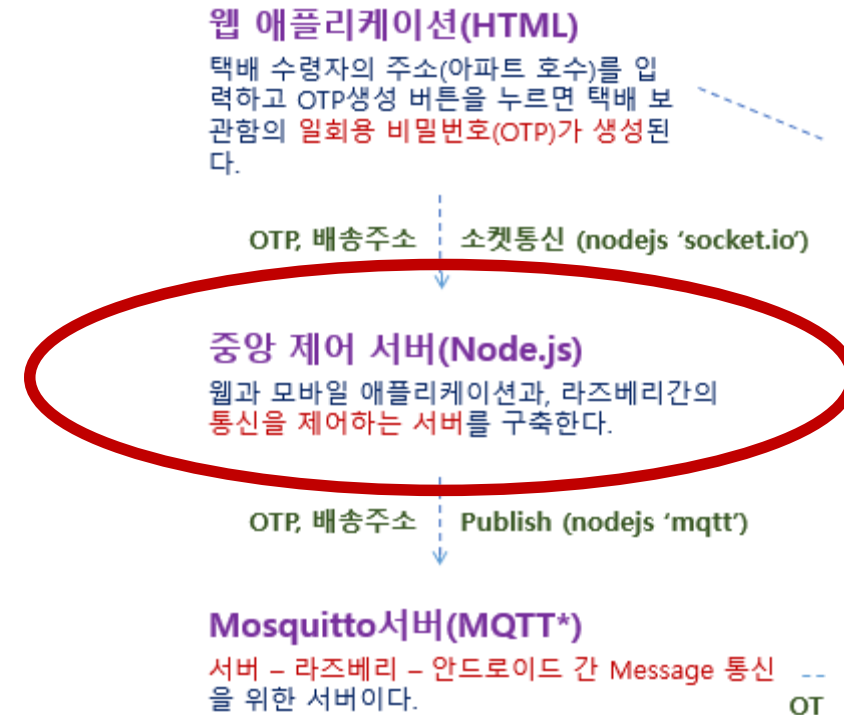
5주차

V. RaspberryPi

6주차

VI. Android

1. Node.js Server



Node.js Server

NPM(Node Package Manager)

- Npm init
- Dependencies 목록에 필요한 모듈을 추가한다.

'http' : 웹서버를 만들기 위한 모듈

'mqtt' : mqtt서버와 연결하고 publish, subscribe하기 위한 모듈

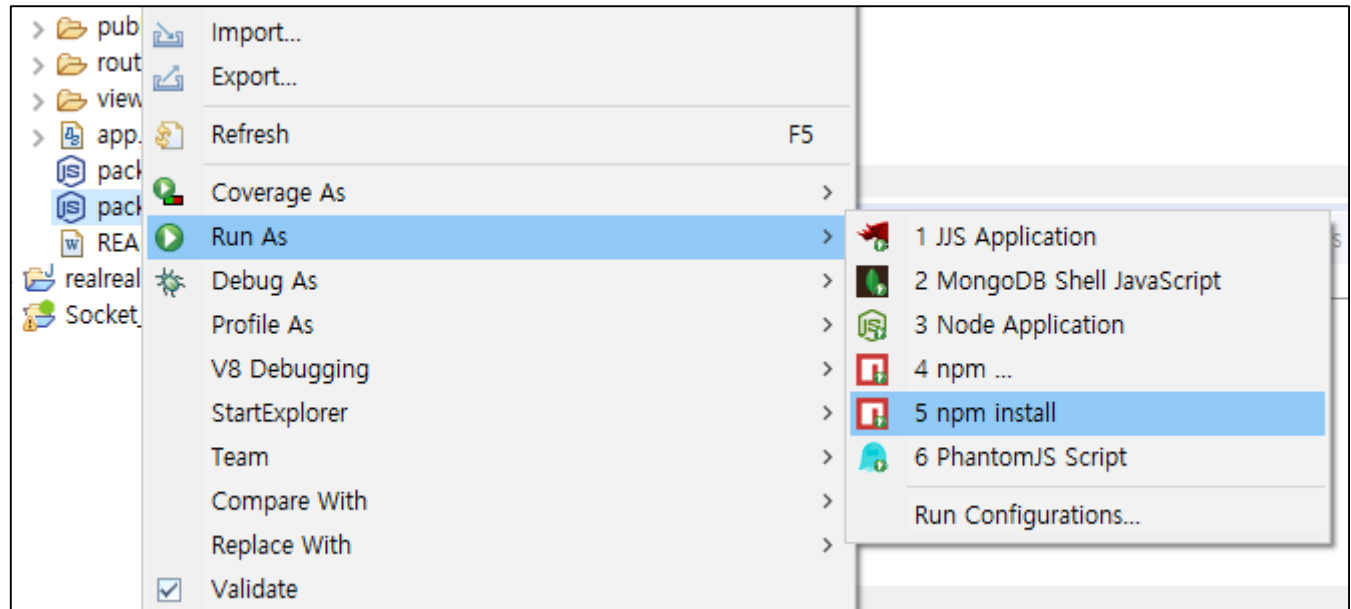
'socket.io' : 소켓을 생성해서 HTML과 통신하기 위한 모듈

```
1 {  
2   "name": "otpbox",  
3   "version": "0.0.0",  
4   "private": true,  
5   "scripts": {  
6     "start": "node ./bin/www"  
7   },  
8   "dependencies": {  
9     "cookie-parser": "~1.4.4",  
10    "debug": "~2.6.9",  
11    "express": "~4.16.1",  
12    "http-errors": "~1.6.3",  
13    "jade": "~1.11.0",  
14    "morgan": "~1.9.1",  
15    "mqtt": "^2.14.0",  
16    "socket.io": "^2.0.4"  
17  }  
18 }
```

Node.js Server

NPM install

- 추가한 모듈을 설치한다.



Node.js Server

제어 서버 구축

- Node.js의 모듈 'http'를 이용해서 서버 객체 만든다.

```
4 var app = require('../app');  
5 var debug = require('debug')('otpbox:server');  
6 var http = require('http');  
7  
8 var port = normalizePort(process.env.PORT || '3000');  
9 app.set('port', port);  
10  
11 var server = http.createServer(app);  
12
```

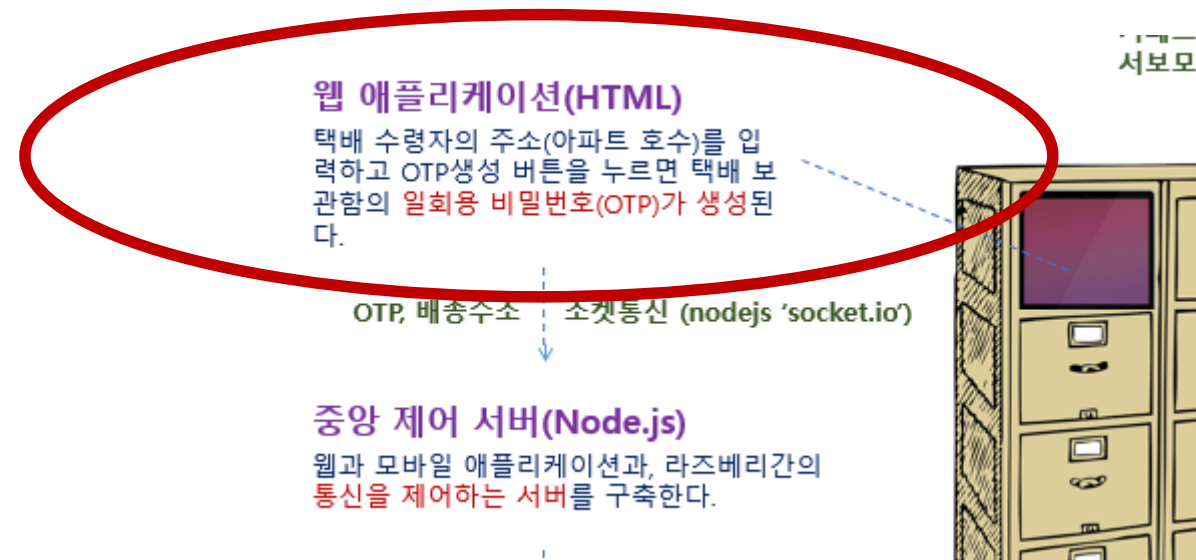
Node.js Server

제어 서버 구축

- 서버 객체에서 생성한 포트를 listen해서 서버를 접속 가능하게 만든다. (port : 3000)

```
39 server.listen(port);|
40 server.on('error', onError);
41 server.on('listening', onListening);
```

2. Web Application(HTML)



Web Application(HTML)

OTP생성 (javascript)

- 4자리 랜덤 비밀번호를 생성한다.

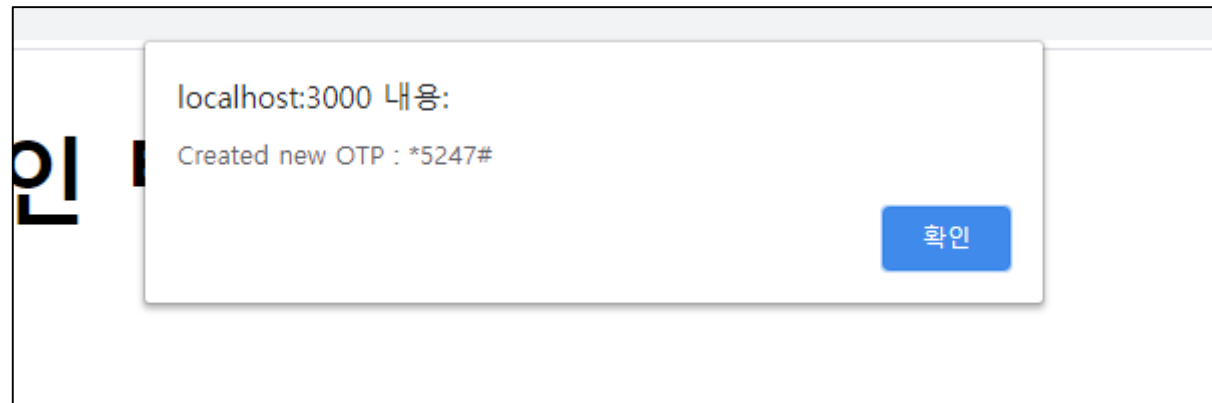
```
//OTP 생성 알고리즘
var randomValue = "0123456789";

for(i=1; i<=4; i++){
    randomPoint = Math.floor(Math.random()*10);
    Pwdchar = randomValue.charAt(randomPoint);
    if(i == 1){
        newPassword = Pwdchar;
    }
    else{
        newPassword += Pwdchar;
    }
}

//OTP 생성 알림 메시지
alert("Created new OTP : "+newPassword+"#");
```

Web Application(HTML)

OTP생성 사진



Web Application(HTML)

타이머 생성 (javascript)

- 비밀번호가 생성되면 타이머가 생성된다.

```
//타이머
//1초에 한번씩 msg_time() 함수 실행
window.onload = function TimerStart(){ tid=setInterval('msg_time()',1000) };

//타이머 함수
function msg_time() {
    //남은 시간 계산
    m = "남은시간 : "+Math.floor(SetTime / 60) + "분 " + (SetTime % 60) + "초";

    //남은 시간 표시
    var msg = "<font size = 2 color='red'>" + m + "</font>" ;
    document.all.ViewTimer.innerHTML = msg;    // ViewTimer div영역 출력

    //일회용 비밀번호 표시
    var psw = "<font size = 4 color='green'>" + "*" + newPassword + "#" + "</font>";
    document.all.OTPpw.innerHTML = psw;

    //설정된 시간에서 1초씩 감소
    SetTime--;

    //타이머가 끝났을 경우
    if (SetTime < 0) {
        clearInterval(tid);
        document.all.OTPpw.innerHTML = null;
        document.all.ViewTimer.innerHTML = null;
        //alert("종료");
    }
}
```

Web Application(HTML)

타이머 생성 사진

OTP를 이용한 무

비밀번호

*4421#

남은시간 : 0분 50초

Web Application(HTML)

주소 선택 (javascript)

- 택배 수령자의 주소를 입력받는다.

```
<h4>주소 선택</h4>
<form id="formname" name="formname" class="address">
  <p><input type="radio" name="add" value="501">501호</input></p>
  <p><input type="radio" name="add" value="502">502호</input></p>
  <p><input type="radio" name="add" value="503">503호</input></p>
</form>
```

Web Application(HTML)

주소 선택 사진

주소 선택

☐ 501호

☐ 502호

☐ 503호

Web Application(HTML)

Web App 시연영상

- Filename: ./html시연영상.mp4



OTP를 이용한 무인 택배 보관함

비밀번호



주소 선택

☐ 501호

☐ 502호

☐ 503호

OTP 생성

OTP

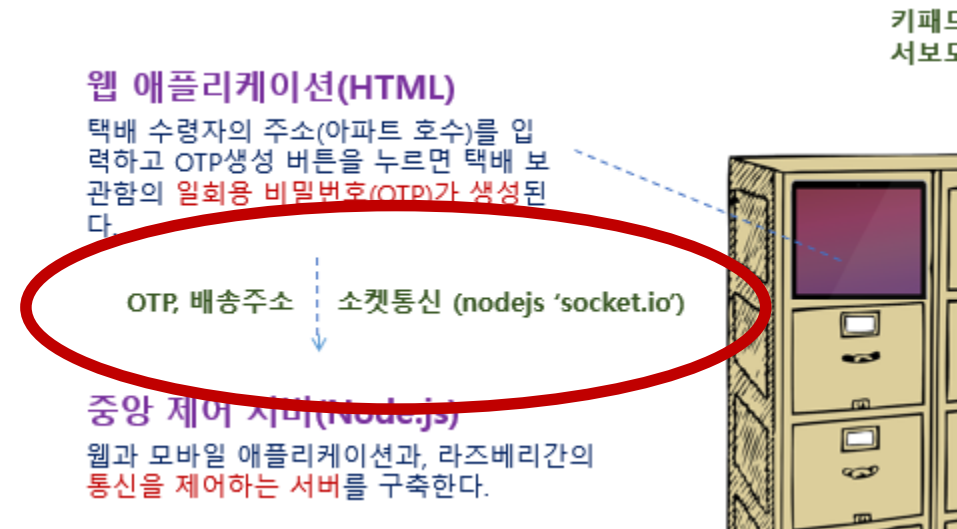
by강통코딩

OTP를 이용한 무인 택배 보관함

team강통코딩

Editor : Eclipse

3. 소켓통신(Server - HTML)



소켓통신(Server - HTML)

소켓통신 서버코드

- 'Socket.io' 모듈을 이용하여 소켓통신
- 'create_OTP' 메시지를 받으면
내용을 console창에 띄움

```
33 // 일회용 비밀번호 생성(HTML소켓통신)
34 var io=require("socket.io")(server);
35 io.on("connection", function(socket){
36     console.log("socket connected");
37     socket.on("create_OTP", function(pw){
38         console.log("create_OTP , message : ",pw);
39         //MQTT서버로 생성된 OTP를 전송하는 코드 추가
40         client.publish("otpbox", pw);
41     });
42 });
```

소켓통신(Server - HTML)

소켓통신 HTML코드

- JQuery를 이용하여 자바스크립트를 서버로 전송
- 주소와 비밀번호를 서버로 전송 (create_OTP)
- 주소와 비밀번호를 한꺼번에 보내기 위해서 JSON으로 파싱

```
<script type="text/javascript" src="/socket.io/socket.io.js"></script>
<script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
<script type="text/javascript">
//소켓통신 연결
var socket=null;
$(document).ready(function(){
    socket=io.connect(); // 3000port
});

//라디오버튼 주소 전달
var st = $(":input:radio[name=add]:checked").val();

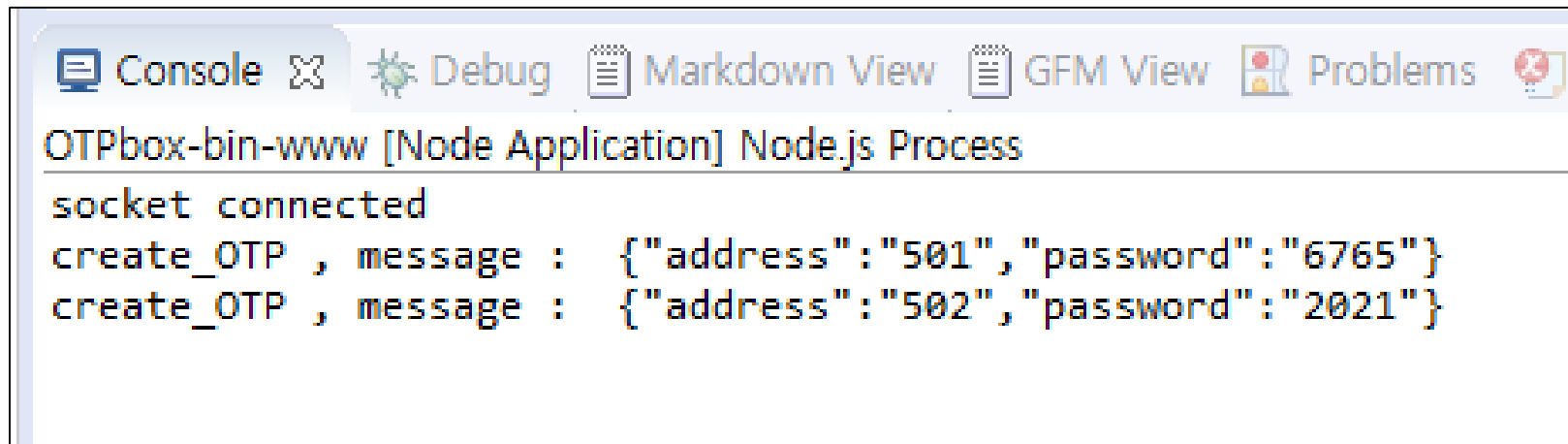
//소켓통신으로 주소와 OTP를 서버에 전송하기 위해 JSON으로 파싱
var ParseJSON = {
    address : st,
    password : newPassword
};

//서버로 전송
socket.emit("create_OTP", JSON.stringify(ParseJSON));
```

소켓통신(Server - HTML)

소켓통신 사진

- 다시 mqtt서버로 보내야하기 때문에 JSON데이터를 추출하지 않음.

A screenshot of a Node.js application's console window. The window has a title bar with tabs for 'Console', 'Debug', 'Markdown View', 'GFM View', and 'Problems'. The 'Console' tab is active, showing the following output:

```
OTPbox-bin-www [Node Application] Node.js Process
socket connected
create_OTP , message : {"address":"501","password":"6765"}
create_OTP , message : {"address":"502","password":"2021"}
```

Editor : cmd

4. Mosquitto Server(MQTT)

중앙 제어 서버(Node.js)

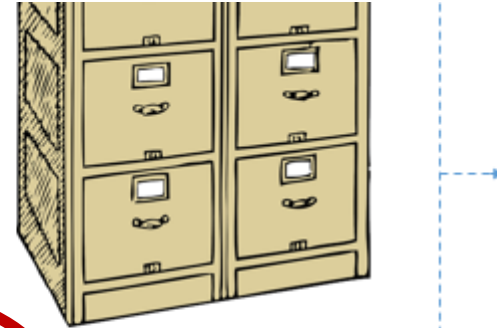
웹과 모바일 애플리케이션과, 라즈베리간의 통신을 제어하는 서버를 구축한다.

OTP, 배송주소 Publish (nodejs 'mqtt')

Mosquitto서버(MQTT*)

서버 - 라즈베리 - 안드로이드 간 Message 통신을 위한 서버이다.

OTP, 배송주소 Subscribe (python 'paho')



Mosquitto Server(MQTT)

Mosquitto Server 실행

- 1883번 포트로 통신

```
C:\Users\Jinwoo Do\dev\MQTTProject\mosquitto>mosquitto -v
1588672702: mosquitto version 1.6.9 starting
1588672702: Using default config.
1588672702: Opening ipv6 listen socket on port 1883.
1588672702: Opening ipv4 listen socket on port 1883.
```

Mosquitto Server(MQTT)

Node.js Server와 통신

- Mosquitto서버와 연결
- HTML에서 소켓통신으로 비밀번호와 주소를 받으면 Mosquitto서버에 Publish (topic : otpbox)

```
25 //MQTT서버 연결
26 var mqtt = require("mqtt");
27 var client=mqtt.connect("mqtt://192.168.0.17");
28 client.on("connect",function(){
29     console.log("MQTT connected");
30 });
```

```
33 // 일회용 비밀번호 생성(HTML소켓통신)
34 var io=require("socket.io")(server);
35 io.on("connection", function(socket){
36     console.log("socket connected");
37     socket.on("create OTP", function(pw){
38         console.log("create_OTP , message : ",pw);
39         //MQTT서버 생성된 OTP를 전송하는 코드 추가
40         client.publish("otpbox", pw);
41     });
42 });
```

Mosquitto Server(MQTT)

Node.js Server와 통신 사진

```
21
22 var server = http.createServer(app);
23
24
25 //MQTT서버 연결
26 var mqtt = require("mqtt");
27 var client=mqtt.connect("mqtt://192.168.0.96");
28 client.on("connect",function(){
29     console.log("MQTT connected");
30 });
31
32
33 // 일회용 비밀번호 생성(HTML소켓통신)
34 var io=require("socket.io")(server);
35 io.on("connection", function(socket){
36     console.log("socket connected");
37     socket.on("create_OTP", function(pw){
38         console.log("create_OTP , message : ",pw);
39     });
40     //MQTT서버로 생성된 OTP로 전송하는 코드 추가
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

OTPbox-bin-www [Node Application] Node.js Process

```
MQTT connected
socket connected
create_OTP , message : {"address":"502","password":"4138"}
```

```
명령 프롬프트 - mosquitto_sub -t otpbox -p 1883
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Jinwoo Do>cd dev
C:\Users\Jinwoo Do\dev>cd MQTTProject
C:\Users\Jinwoo Do\dev\MQTTProject>cd mosquitto
C:\Users\Jinwoo Do\dev\MQTTProject\mosquitto>mosquitto_sub -t otpbox -p 1883
{"address":"502","password":"4138"}
```

