IOT 시스템

- Web Browser를 이용한 RaspberryPi 센서 제어 -

학과: 국제스포츠레저학부

학번: 201500483

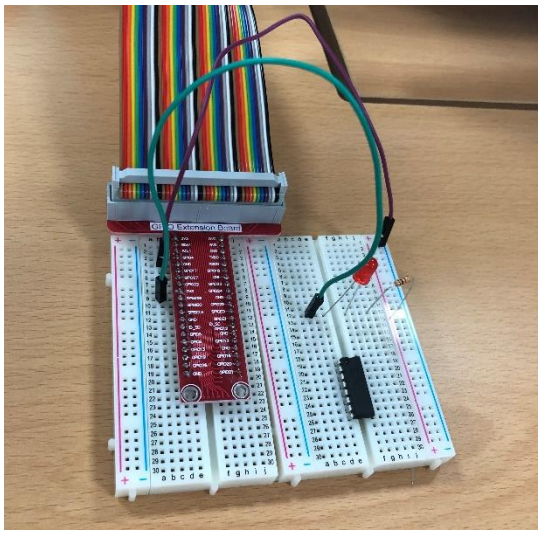
이름: 김서연

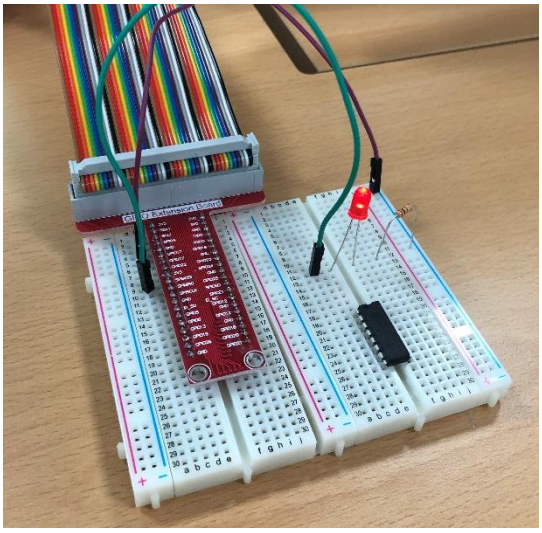
1. Node.js를 이용한 LED Blink

- 학습내용: 라즈베리파이와 GPIO를 사용하여 LED를 깜빡이게 할 것이다. GPIO를 제어하기 위해 On/Off 모듈과 함께 Node.js를 사용한다. LED 조명을 켜기 위해 GPIO핀을 '출력'으로 사용하고, 코드를 작성하여 켜거나 끄도록 할 것이다.

On/Off 모듈을 설치하기 위해 [npm install onoff] 를 입력해주었고 blink.js 파일을 작성하여 blink.js를 실행하도록 하였다. 첫번째 과제에서 수행한 것을 바탕으로 다음과 같이 브레드보드 회로부를 연결하고 라즈베리파이를 부팅한다. 전압이 흘러갈 수 있도록 LED와 연결하고 저항을 두어 전압이 다시 GND로 흘러갈 수 있도록 한다.

이처럼 node.js를 이용하여 라즈베리파이에 연결된 센서 정보를 얻어 LED와 같은 엑츄에이터를 제어할 수 있는 실습을 진행해보았다.





blink.js

File Edit Search Options Help

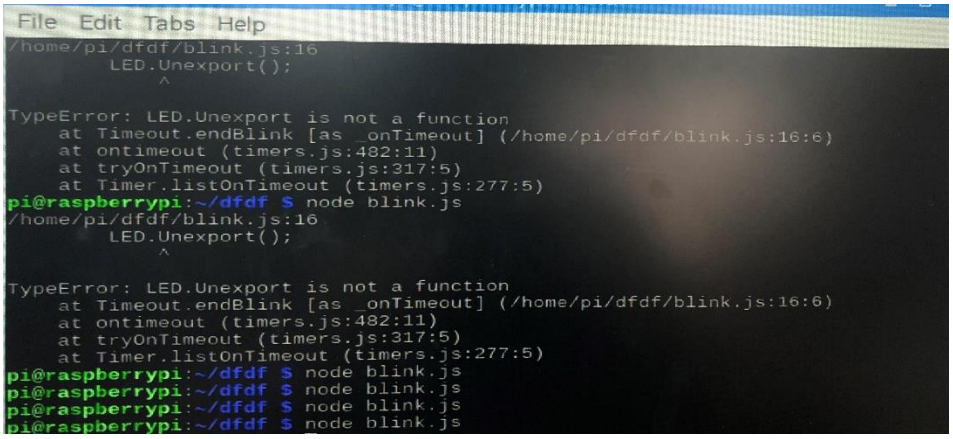
```
var Gpio = require('onoff').Gpio;
var LED = new Gpio(4, 'out');
var blinkInterval = setInterval(blinkLED, 250);

function blinkLED(){
        if(LED.readSync() === 0) {
        LED.writeSync(1);
        } else {
        LED.writeSync(0);
        }
}

function endBlink(){
        clearInterval(blinkInterval);
        LED.writeSync(0);
        LED.unexport();
}

setTimeout(endBlink, 10000);
```

- 프로그램 실행

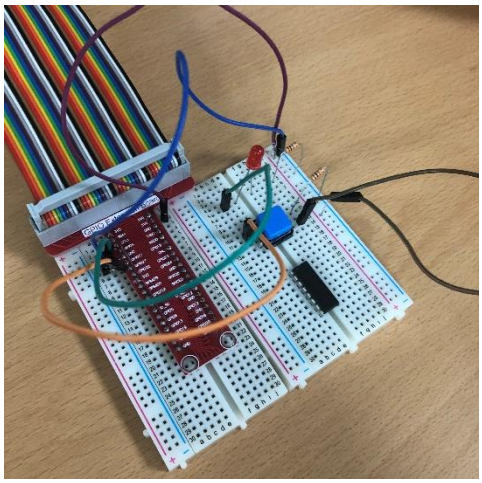


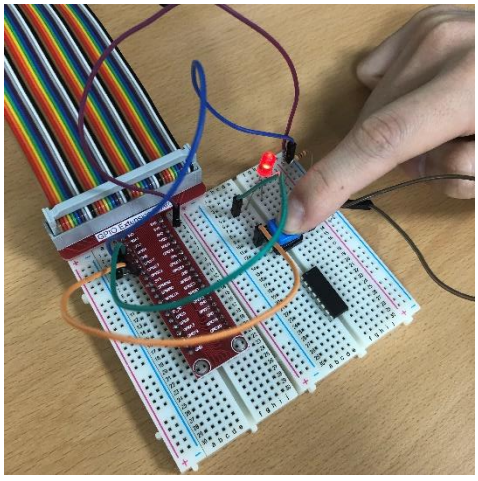
2. Node.js를 이용한 LED Blink (2)

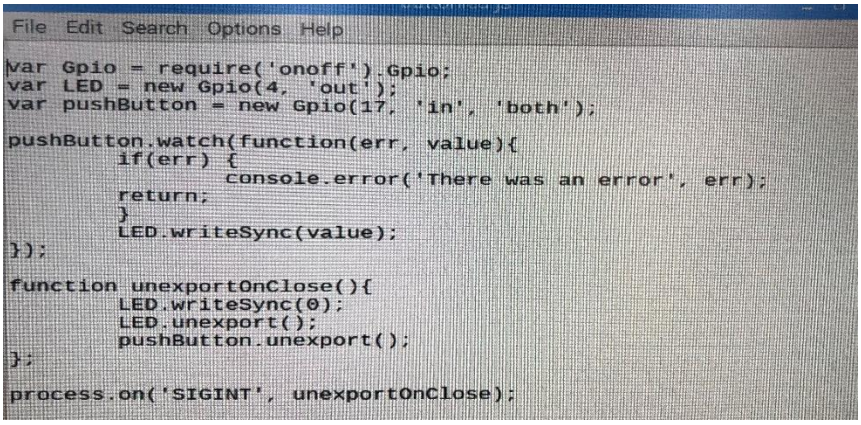
- 학습내용: 앞선 실습을 바탕으로 본 실습 또한 라즈베리파이와 GPIO를 사용하여 LED를 깜빡이도록 할 것이며 GPIO 핀을 출력으로 사용하였다. 또 다른 PushButton GPIO 핀을 입력으로 사용할 것이다. 브레드보드에 연결된 버튼을 누르면 LED가 켜지도록 회로를 구성하고 그에 맞는 코드를 작성하였다.

전압이 흐를 수 있도록 버튼의 네 다리 중 하나와 연결하였으며 다리 전체가 브레드보드에 연결되어 있기 때문에 어느 곳을 꽂아도 무방하다. 버튼을 연결하여 트렌치 전체에 맞도록 하기 위해 11행과 13행에 연결한다. 다음과 같이 구성하였다면 buttionled.js 코드를 실행시킨다.

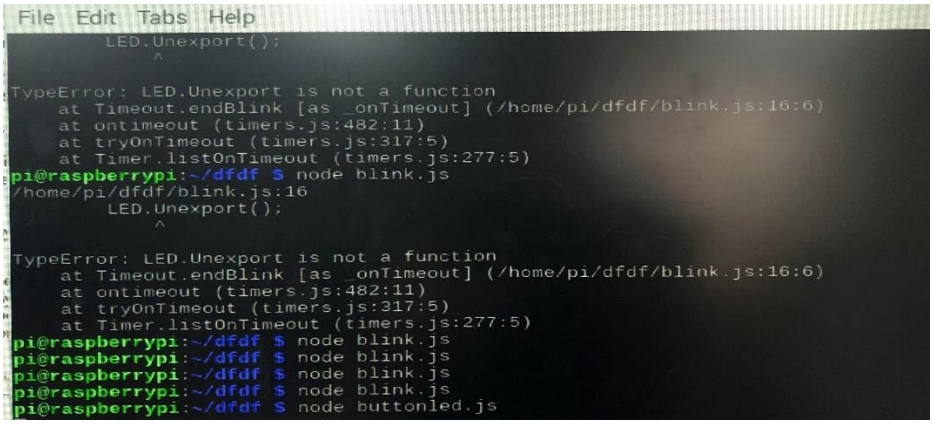
노드로 센서 정보를 얻고 LED와 버튼을 제어하여 Node.js를 이용한 라즈베리파이 제어를 이해할 수 있었다.







- 프로그램 실행



3. Node.js 기반 Web Server

- 학습내용: Node.js를 실행할 수 있도록 HTML 파일을 제공할 수 있는 webserver.js파일을 작성하여 실습을 진행한다. 현재 디렉토리에서 HTML 파일에 사용할 수 있는 새 디렉토리 Public을 생성한다. 요청된 파일을 열고 클라이언트에 내용을 반환하는 Node.js 파일을 생성하는데, 이상이 있으면 404의 오류를 던지도록 설계한다.

Public 파일로 이동하고, Index.html 파일을 다음과 같이 작성한다. 이 파일은 아직 어떤 기능도 갖추지 못하는데, 웹 서버가 작동하는지 확인하기 위해 다시 현재 디렉토리로 위치를 이동해 webserver.js를 실행시킨다. [RaspberryPi_IP]:8080을 사용하여 브라우저에서 웹 사이트를 열어 index.html이 제대로 작동하는지 확인한다.

webserver.js

File Edit Search Options Help

```
var http = require('http').createServer(handler);
var fs = require('fs');

http.listen(8080);

function handler(req, res) {
        fs.readFile(_dirname+'/public/index.html', function(err, data){
        if(err){
                res.writeHead(404, {'Content-Type': 'text/html'});
                return res.end("404 Not Found");
        }
        res.writeHead(200, {'Contect-Type': 'text/html'});
        res.write(data);
        return res.end();
        });
}
```

index.html

File Edit Search Options Help

```
<!DOCTYPE html>
<html>
<body>
<h1>Control LED light</h1>
<input id="light" type="checkbox">LED
</body>
</html>
```

e-Class System | Microsoft PowerPoint | 10.10.3.57:8080

Not secure | 10.10.3.57:8080

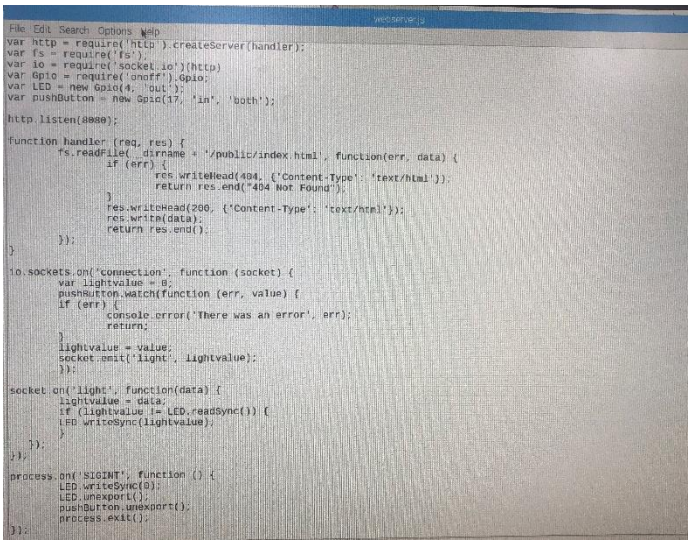
Control LED light

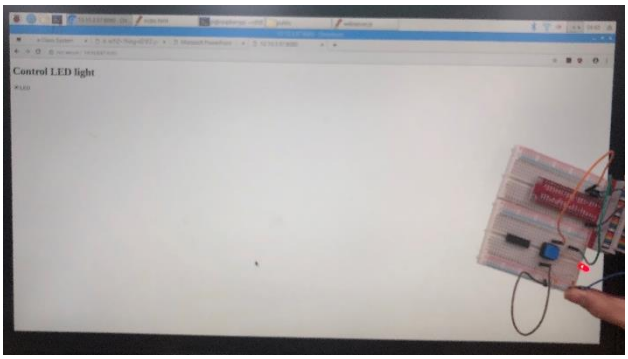
☑ LED

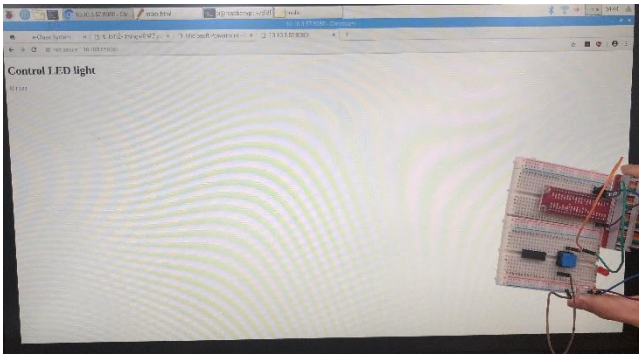
4. Node.js 기반 Web Server (2)

- 학습내용: 웹 서버가 가동되고 실행된다면 WebSocket을 활용한 실습을 진행할 수 있다. 웹 서버를 설정한 상태에서 라즈베리파이 시스템 패키지를 최신 버전으로 업데이트해야한다. 설치된 모든 패키지를 최신 버전으로 업그레이드하여 socket.io의 최신 버전을 다운로드하고 설치한다. 웹 서버에 WebSocket을 추가하여 기존 파일을 수정하고, index.html 또한 수정하여 LED를 제어하도록 한다.

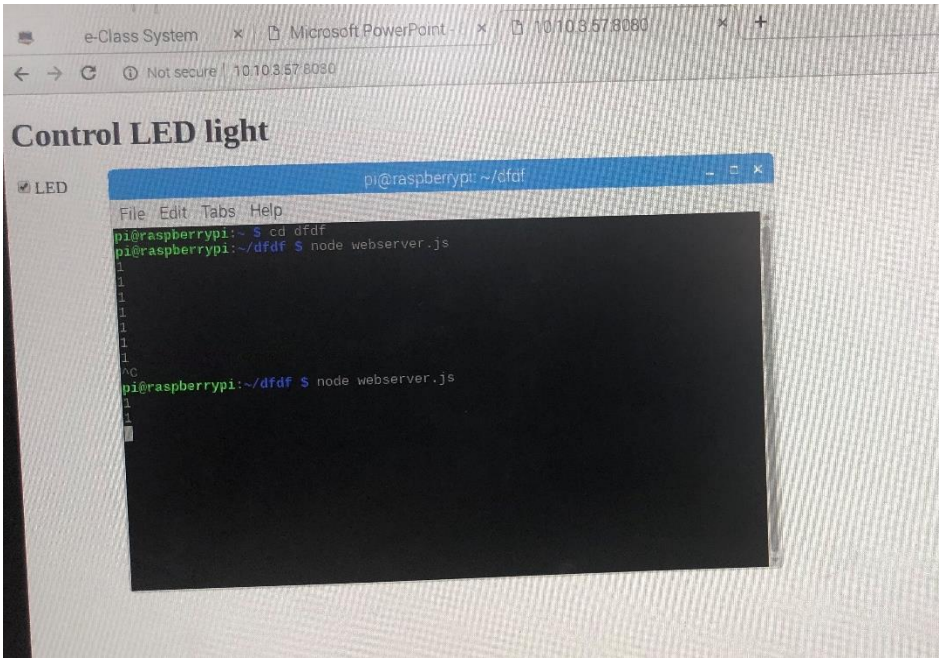
http://[RaspberryPi_IP]:8080/을 사용하여 브라우저에서 웹 사이트 열고 서버가 모든 변경 사항을 라스베리 파이 상의 콘솔로 출력해야 한다. 이때 클라이언트가 서버에 변경 사항을 보내고 있으며, 서버가 응답하고 있다.







- 프로그램 실행


e-Class System
Microsoft PowerPoint
10.10.3.57:8080
Not secure | 10.10.3.57:8080
Control LED light
LED
pi@raspberrypi: ~/dfdf
File Edit Tabs Help
pi@raspberrypi:~ $ cd dfdf
pi@raspberrypi:~/dfdf $ node webserver.js
1
1
1
1
1
1
1
^C
pi@raspberrypi:~/dfdf $ node webserver.js
1
1