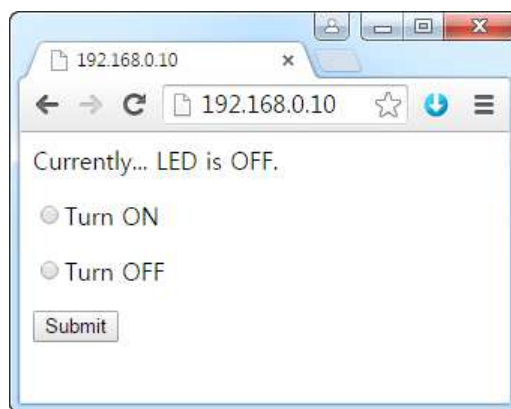
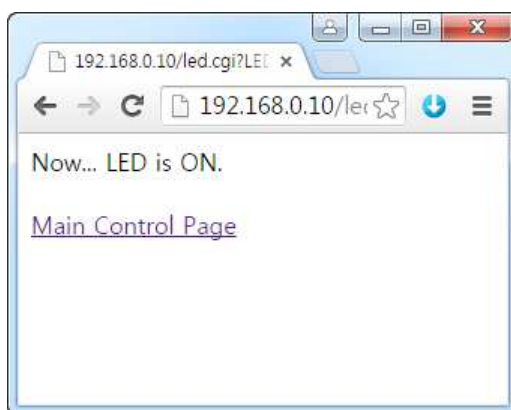


Chapter 273. 이더넷 - 원격 제어

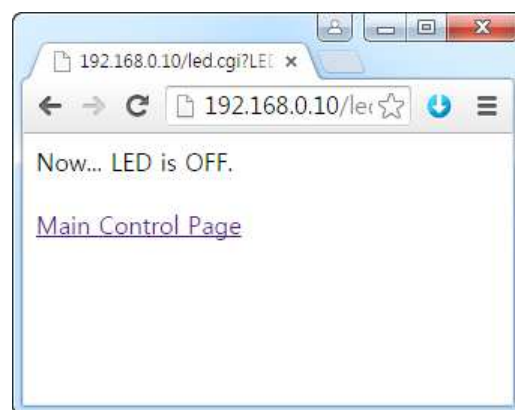
사물인터넷의 응용 분야 중에서 주목을 받고 있는 분야 중 하나가 홈오토메이션(Home Automation)이다. 홈오토메이션에도 여러 가지가 있지만 흔히 이야기되는 것은 스마트폰을 이용해서 집 밖에서도 집안의 전등이나 다양한 기기들을 조작하고 관리할 수 있는 서비스를 말한다. 아두이노 역시 홈오토메이션에 사용될 수 있다. 아두이노가 인터넷에 연결되어 있다면 아두이노에 웹 서버를 구축하고 웹 브라우저를 통해 가정 내 기기들을 조작할 수 있도록 할 수 있다. 이 장에서는 인터넷을 통해 아두이노에 연결된 LED를 제어하는 방법에 대해 알아본다.



(a) 메인 페이지



(b) LED ON 페이지



(c) LED OFF 페이지

그림 1. 인터넷을 통한 LED 제어 페이지 구성

LED 원격 제어를 위한 페이지는 그림 1에서와 같이 3가지로 구성된다. 아두이노에 구축된 웹 서버에 접속하면 현재 LED의 상태를 알려주고, LED를 ON 또는 OFF로 변경할 수 있는 옵션이 표시되는 메인 페이지가 표시된다. 둘 중 하나의 옵션을 선택하고 'Submit' 버튼을 누르면 LED는 선택한 옵션에 맞게 바뀌면서 LED의 상태 변화를 알려주는 페이지로 이동한다. 'Main Control Page' 링크를 누르면 메인 페이지로 이동한다. 메인 페이지는 사용자의 요구를 전달하고 그 결과를 사용자에게 돌려주기 위한 방법 중 하나인 CGI(Common Gateway Interface)를 사용한다. 아두이노에서 메인 페이지를 출력하는 함수는 스케치 1과 같다.

스케치 1. show_main_page

```
void show_main_page(EthernetClient client)
{
    // HTML 헤더
    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: text/html");
    client.println();

    // HTML 내용
    client.println("<html>");
    client.println("<body>");

    client.print("Currently... LED is ");      // 현재 LED 상태 표시
    client.println(statusLED ? "ON." : "OFF.");

    // CGI를 통해 사용자 입력을 받음
    client.println("<br />");
    client.println("<FORM method=\"get\" action=\"/led.cgi\">");
    client.println("<P><INPUT type=\"radio\" name=\"LEDstatus\" value=\"1\">Turn
ON");
    client.println("<P><INPUT type=\"radio\" name=\"LEDstatus\" value=\"0\">Turn
OFF");
    client.println("<P><INPUT type=\"submit\" value=\"Submit\"> </FORM>");

    client.println("</body>");
    client.println("</html>");
}
```

```

    client.println();
}

```

사용자가 라디오 버튼을 선택하고 'Submit' 버튼을 누르면 웹 브라우저는 'GET' 명령을 웹 서버로 전달하며, 'action'에 표시된 CGI로 'name' 및 'value' 쌍으로 이루어진 인자가 추가되어 전달된다.

GET / HTTP/1.1	← 메인 페이지 요청
GET /led.cgi?LEDstatus=1 HTTP/1.1	← LED ON 페이지 요청
GET /led.cgi?LEDstatus=0 HTTP/1.1	← LED OFF 페이지 요청

그림 2. 웹 서버로의 요청 형식

아두이노에서 LED ON 또는 LED OFF 페이지를 출력하는 함수는 스케치 2와 같다. LED ON/OFF 페이지에서는 CGI를 통해 전달되는 LED의 상태를 저장하고, 저장된 값으로 LED의 상태를 바꾼다. 또한 메인 페이지로 돌아갈 수 있는 링크를 표시해 준다.

스케치 2. show_on_off_page

```

void show_on_off_page(EthernetClient client, boolean status)
{
    statusLED = status;           // LED 상태 저장
    digitalWrite(pinLED, status); // LED 상태 바꿈

    // HTML 헤더
    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: text/html");
    client.println();

    // HTML 내용
    client.println("<html>");
    client.println("<body>");

```

```

client.print("Now... LED is ");          // LED 상태 표시
client.println(statusLED ? "ON." : "OFF.");

client.println("<br /><br />");
// 메인 페이지로의 링크 표시
client.println("<a href='\"/\"'>Main Control Page</a>");

client.println("</body>");
client.println("</html>");
client.println();
}

```

웹 서버로 메인 페이지 또는 LED ON/OFF 페이지 요청은 그림 2의 요청 이외에도 다양한 내용이 전달된다. 내용이 전달된 이후에는 개행문자(`\n`과 `\r`의 조합)로 끝나는 빈 줄이 추가되며, 빈 줄을 수신한 이후 웹 서버는 웹 브라우저로 결과를 전송할 수 있다.



그림 3. 웹 서버로 전달되는 내용

스케치 3은 loop 함수를 나타낸 것으로 웹 서버가 한 행을 수신하면 요청의 종류를 알아내고,

빈 줄이 수신된 이후 이를 처리하도록 구성되어 있다.

스케치 3. loop

```
void loop() {  
  // 웹 브라우저의 접속 여부 확인  
  EthernetClient client = server.available();  
  if (client) {                                     // 웹 브라우저가 접속한 경우  
    int response_type = -1;  
    Serial.println("** New client connected...");  
  
    while (client.connected()) {  
      // 줄 단위의 데이터 수신  
      // HTML 요청의 끝을 알아내기 위해 빈 줄이 수신되었는지의 여부를 반환함  
      boolean empty_line = read_one_line(client);  
      Serial.println(buffer);  
  
      if(buffer.indexOf("GET / ") >= 0)  
        response_type = 1;                          // 메인 페이지 요청  
      else if(buffer.indexOf("GET /led.cgi?LEDstatus=1") >= 0)  
        response_type = 2;                          // LED ON 페이지 요청  
      else if(buffer.indexOf("GET /led.cgi?LEDstatus=0") >= 0)  
        response_type = 3;                          // LED OFF 페이지 요청  
  
      if(empty_line) break;                          // HTML 요청이 끝나면 결과 전송  
    }  
  
    switch(response_type){                          // 웹 브라우저의 요청에 따른 결과 전송  
      case 1: show_main_page(client); break;  
      case 2: show_on_off_page(client, true); break;  
      case 3: show_on_off_page(client, false); break;  
    }  
  
    client.stop();                                  // 웹 브라우저 연결 종료  
    Serial.println("** Client disconnected...");  
  }  
}
```

```
}

```

웹 서버로 수신되는 내용을 줄 단위로 읽는 read_one_line 함수는 스케치 4와 같다. 개행문자는 '\r' 이후 '\n' 문자가 수신되므로 이들 문자는 버퍼에 포함시키고 있지 않으며 결과값으로 빈 줄의 수신 여부를 반환한다.

스케치 4. read_one_line

```
boolean read_one_line(EthernetClient client)
{
    buffer = "";                                // 버퍼 비움
    while(client.available()){
        char c = client.read();                 // 바이트 데이터 읽기
        if(c == '\r'){                          // 개행문자를 수신한 경우
            client.read();                      // '\n' 문자 버림
            break;
        }
        buffer = buffer + c;                   // 개행문자 이외의 문자는 버퍼에 저장
    }

    return (buffer.length() == 0);             // 빈 줄 여부 반환
}
```

setup 함수와 전역 변수의 선언은 스케치 5와 같다.

스케치 5. setup & 기타

```
#include <Dhcp.h>
#include <Dns.h>
#include <Ethernet.h>
#include <EthernetClient.h>
#include <EthernetServer.h>
#include <EthernetUdp.h>
```

```
#include <SPI.h>

byte MAC[] = { 0x90, 0xA2, 0xDA, 0x0D, 0xD2, 0x16 };
IPAddress IP(192, 168, 0, 10);
EthernetServer server(80);           // 80번 포트를 이용하는 웹 서버 생성
String buffer = "";                  // 수신 데이터 버퍼

int pinLED = 2;                       // LED 연결 핀
boolean statusLED = false;           // LED 상태

void setup(){
  Serial.begin(9600);

  Ethernet.begin(MAC, IP);           // 고정 IP 사용
  Serial.print("Local IP : ");
  Serial.println(Ethernet.localIP());
  delay(2000);                       // 이더넷 설드의 초기화 대기

  server.begin();                    // 웹 서버 시작

  pinMode(pinLED, OUTPUT);           // LED 연결 핀을 출력으로 설정
  digitalWrite(pinLED, LOW);         // LED는 꺼진 상태로 시작
  statusLED = false;
}
```

웹 서버는 192.168.0.10의 고정 IP 주소를 사용하는 것으로 가정하였으므로 웹 브라우저에서 이 주소로 접속하면 그림 1의 메인 페이지를 확인할 수 있으며, 라디오 버튼을 통해 아두이노에 연결된 LED를 인터넷을 통해 제어할 수 있다. 사설 IP 주소를 사용하므로 외부에서의 접속은 불가능하지만 공유기에 접속된 스마트폰으로도 그림 1의 결과를 확인할 수 있다.

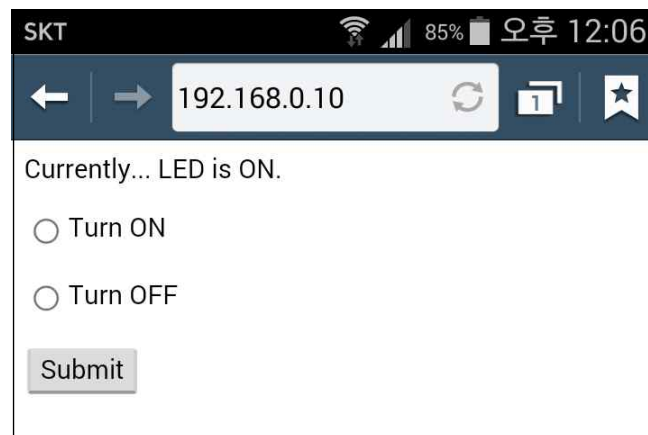


그림 4. 스마트폰으로 LED 제어 페이지에 접속