목차

1. 프로젝트 주제 및 선정 이유 **.................................................. 1**

2. 개발 언어 **................................................................................ 1**

3. 프로젝트 일정 **......................................................................... 2**

4. 간단한 소스코드 **..................................................................... 3**

5. 최종 진행상황 **......................................................................... 5**

**6. 참고자료 ................................................................................. 6**

**최종 프로젝트 보고서**

**1. 프로젝트 주제 및 선정 이유**

▶프로젝트 주제: C# VS Arduino

네트워크 시간에 프로젝트로 통신 프로그램을 이번에 만들게 되었는데, 뭔가 신기한

아이디어가 없을까? 고민을 하다가 문득 이번 학기 무선통신 시간에 새롭게 배우게 된

Arduino언어를 이용하여 C#이랑 연관 지어 통신을 하여 메시지를 보내면 간단하게

LCD에 띄우면 좋을 것 같다고 생각되어서 선정하게 되었다.

**2. 개발 언어**

▶C# 언어

마이크로소프트가 닷넷 프레임워크의 일부로 만든 객체 지향 프로그래밍 언어이고,

C++에 기본을 둔 언어로, 비주얼 베이직이나 자바(JAVA)와도 비슷하다.

▶아두이노

물리적인 세계를 감지하고 제어할 수 있는 인터랙티브 객체들과 디지털 장치를 만들기 위한 도구로, 간단한 마이크로컨트롤러 보드를 기반으로 한 오픈 소스 컴퓨팅 플랫폼과 소프트웨어 개발 환경을 말한다.

**3. 프로젝트 일정**

주제 선정 (11/05~11/07)

↓

1차 코딩 (DB 설계 및 로그인 구현) (11/11~11/15)

↓

1차 코딩 수정 (DB 오류 및 DB 사용X) (11/18~23)

↓

2차 코딩 (프로토콜 설계) 및 교수님 컨펌(11/26~30)

↓

교수님 2차 컨펌 / 3차 코딩 (프로토콜 구현)(12/03~06)

↓

마무리 (프로토콜 수정 및 다듬기) (12/08~12)

**4. 간단한 소스코드(프로토콜 설계)**

// & 기준으로 짜른다.

string[] tokens = text.Split('&');

// 항상 데이터의 0번째와 1번째는

// 상태 코드, id가 각각 들어간다.

string St\_code = tokens[0];

string us\_ID = tokens[1];

// 전송받은 데이터에 대한 ACK를 보내기 위한 변수

string Message = null;

byte[] Msg = null;

Socket socket = null;

switch (St\_code)

{

// 로그인

case "100":

clientNum++;

// 연결 완료되었다는 메세지를 띄워준다.

AppendText(txtHistory, "서버와 연결되었습니다.");

AppendText(txtHistory, string.Format("[접속{0}]ID:{1}:{2}",

clientNum, us\_ID, obj.WorkingSocket.RemoteEndPoint.ToString()));

//ID 체크 하는 부분

if (us\_ID == "kdw0439")

{

string us\_PW = tokens[2];

//PW 체크하는 부분

if (us\_PW == "rkddkwl125")

{

//리스트에 추가하는 부분은 로그인 시에만 하면 됨

connectedClients.Add(us\_ID, obj.WorkingSocket);

//현재 id(key)에 해당하는 Socket값을 socket변수에 반환 시켜 준다.

connectedClients.TryGetValue(us\_ID, out socket);

//로그인 성공 메시지

Message = "101&Success&로그인 OK&";

//TextBox에 메시지 출력

AppendText(txtHistory, Message);

//전송할 메시지를 UTF8로 인코딩

Msg = Encoding.UTF8.GetBytes(Message);

//현재 socket값에 메시지 전송

SendTo(socket, Msg);

}

// 로그인 실패(맞지 않는 PW)

// 로그인 실패시에는 ID를 Dic에 추가하지 않는다.

else if (us\_PW != "rkddkwl125")

{

//잘못된 PW 입력하여 실패 메시지 출력

Message = "103&False&PW error&";

// TextBox에 메시지 출력

AppendText(txtHistory, Message);

//메시지를 UTF8로 인코딩

Msg = Encoding.UTF8.GetBytes(Message);

//현재 sock값에 메시지 전송

SendTo(obj.WorkingSocket, Msg);

// 로그인에 실패한 소켓 값은 닫는다.

obj.WorkingSocket.Close();

}

}

// 로그인 실패(맞지 않는 ID)

// 로그인 실패시에는 ID를 Dic에 추가하지 않는다.

else if (us\_ID != "kdw0439")

{

//id 오류 메시지 출력

Message = "102&False&id error&";

// TextBox에 메시지 출력

AppendText(txtHistory, Message);

//메시지를 UTF8로 인코딩

Msg = Encoding.UTF8.GetBytes(Message);

//현재 sock값에 메시지 전송

SendTo(obj.WorkingSocket, Msg);

// 로그인에 실패한 소켓 값은 닫는다.

obj.WorkingSocket.Close();

}

break;

// 아두이노(LCD) 메시지 전송

case "200":

string us\_Msg = tokens[1];

// 연결 완료되었다는 메세지를 띄워준다.

AppendText(txtHistory, "서버와 연결되었습니다.");

AppendText(txtHistory, string.Format("[접속{0}] Message:{1} :{2}",

clientNum, us\_Msg, obj.WorkingSocket.RemoteEndPoint.ToString()));

if (us\_Msg.Length < 21)

{

connectedClients.Add(us\_Msg, obj.WorkingSocket);

//현재 id(key)에 해당하는 Socket값을 socket변수에 반환 시켜 준다.

connectedClients.TryGetValue(us\_Msg, out socket);

//메시지 전송 성공

Message = "201&Success&message OK&";

arduSerialPort.Write("#TEXT"+us\_Msg+ "#\n");

//TextBox에 메시지 출력

AppendText(txtHistory, Message);

//전송할 메시지를 UTF8로 인코딩

Msg = Encoding.UTF8.GetBytes(Message);

//현재 sock값에 메시지 전송

SendTo(socket, Msg);

}

else

{

// 메시지 전송 실패

Message = "400&False&message fail&";

AppendText(txtHistory, Message);

//메시지를 UTF8로 인코딩

Msg = Encoding.UTF8.GetBytes(Message);

SendTo(obj.WorkingSocket, Msg);

// 실패한 소켓 값은 닫는다.

obj.WorkingSocket.Close();

}

break;

**5. 최종 진행상황**

Server 폼과 로그인 클라이언트 폼을 띄우고 우선 서버를 Open한 뒤, 서버에 사전에

설정한 아이디랑 비밀번호를 입력하면 101&Success&로그인 OK& 이라는 프로토콜

메시지, 아이디를 잘못 입력할 경우 102&False&id error& 프로토콜 메시지를 보내고

비밀번호를 잘못 입력할 경우 103&False&PW error& 프로토콜 메시지를 보낸다.

그리고 로그인 성공하면 로그인에 성공했습니다. 라는 메시지 박스가 나타나고 내서

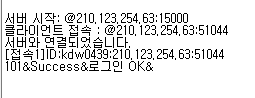
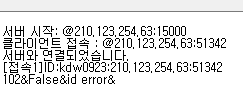
다음 폼인 Arduino을 띄워서 앞에 프로토콜 메시지처럼 만약 메시지 전송을 성공하면

201&Success&message OK&라는 프로토콜 메시지를 보내고 메시지 전송을 실패하면

400&False&message fail&라는 프로토콜이 나타나도록 해주었다.

근데 메시지 전송이 될 경우 Arduino LCD에 띄우는 부분을 아쉽게도 구현하지

못하였는데, 아직 Arduino랑 C#을 연동하는 지식이 많이 부족하다는 것을 느꼈다.

**참고 자료**

임지영 교수님, “MultiChat - chatting\_Dictionary”, 2018.11.17,

<https://classroom.google.com/c/MTUyOTc2Njk4NDFa/m/MjQ1MDY4NTgyMTVa/details>

김지하, “ChatClient”, 2018.12.11, <https://github.com/KimYaKi/C-NetworkProgramming>

WhiteAT, “C# 아두이노 간의 WIFI 통신으로 LCD 제어”, 2015.05.22,

<http://docs.whiteat.com/?p=808>