다양한 전공과 오랜 대학생활을 통해 데이터 구조 및 프로그램 설계에 특화된 Android 개발자 지원자 김진희의 포트폴리오입니다.

목차

I. 참여 프로젝트 목록

- LoRa 기반 무선 에너지 미터계 개발 (펌웨어, 트리구조 설계, Linux, C, Python) - 2017년 9월 ~ 2018년 8월
- ICT 기반의 융복합 극지 환경 관측 시스템 개발 (데이터 분석, GUI, Java) - 2017년 3월 ~ 2018년 8월
- 2016년 정보사회학과 학술제 ISM (데이터 사이언스, 시각화, R) - 2016년 9월 ~ 12월
- 전자통신공학과 캡스톤디자인 (**Android**, SQLite, SQL, Java, C++) - 2015년 9월 ~ 2016년 5월
- 전자통신공학과 과목 - 마이크로프로세서 응용, 문자 코드 생성기 (Java, C) – 2015년 3월 – 6월
- Dx-Ball 게임 모방 (Windows API, Linked List, C/C++) - 2012년 7월 ~ 8월

2012년부터 다양한 프로젝트를 통해서 프로그램 전체를 설계하고 개발해왔고, 특히 비교적 최근 참여한 프로젝트인 'LoRa 기반 무선 에너지 미터계 개발' 에서는 트리(Tree) 형태의 데이터 구조(Data Structure)를 설계하거나 딕셔너리(Dictionary) 구조를 많이 활용해왔습니다.

Android 앱을 개발한 프로젝트는 '전자통신공학과 캡스톤디자인'으로 한 번 있었습니다.

II. Android 개발 프로젝트

전자통신공학과 캡스톤디자인 - 2015년 9월 ~ 2016년 5월

- 블루투스 신호 세기를 계산해서 사용자의 운동기구 탑승 여부를 확인하고 입력한 정보를 운동기구에 자동으로 설정되게 하는 앱

구현한 기능

- 블루투스 신호 세기 계산 및 사용자 위치 인식
- 블루투스 데이터 송수신
- SQLite 데이터베이스 저장 (설정, 저장 버튼)
- SQLite 데이터베이스 읽어오기 (운동기록, 달력 탭)
- 좌우 스와이프로 탭 넘기기, 스크롤(운동기록)
- 모든 UI 컴포넌트 기능

스크린샷과 동일한 결과를 내는 소스 코드는 유실한 상태로,
현재 소스코드는 버그 수정이 필요한 상태입니다.

https://github.com/kjh0311/kim_jin_hee-projects/tree/master/2016%20-%20Android%2C%20Data%20Science

II. Android 개발 프로젝트

전자통신공학과 캡스톤디자인 - 2015년 9월 ~ 2016년 5월

- 블루투스 신호 세기를 계산해서 사용자의 운동기구 탑승 여부를 확인하고 입력한 정보를 운동기구에 자동으로 설정되게 하는 앱

개발 도구

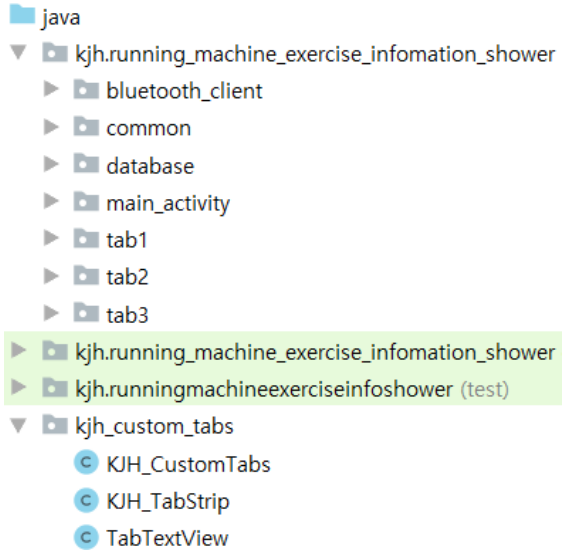
- Android Studio
- Android SDK Platform 21 (revision: 2)
- Build Tools: 23.0.2
- Gradle: 3.2.1

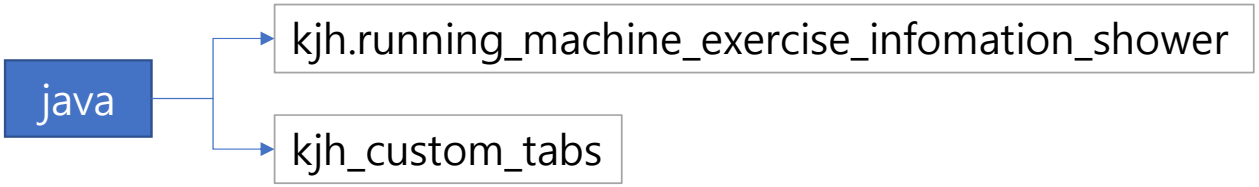
II. Android 개발 프로젝트

전자통신공학과 캡스톤디자인 - 2015년 9월 ~ 2016년 5월

- 블루투스 신호 세기를 계산해서 사용자의 운동기구 탑승 여부를 확인하고 입력한 정보를 운동기구에 자동으로 설정되게 하는 앱

코드 구조 – 자바 최상위 패키지





java 파일은 kjh.running_machine_exercise_infomation_shower 패키지에 총 7개이 패키지를 넣었고, kjh_custom_tabs 패키지는 그 패키지 밖에 만들었습니다.

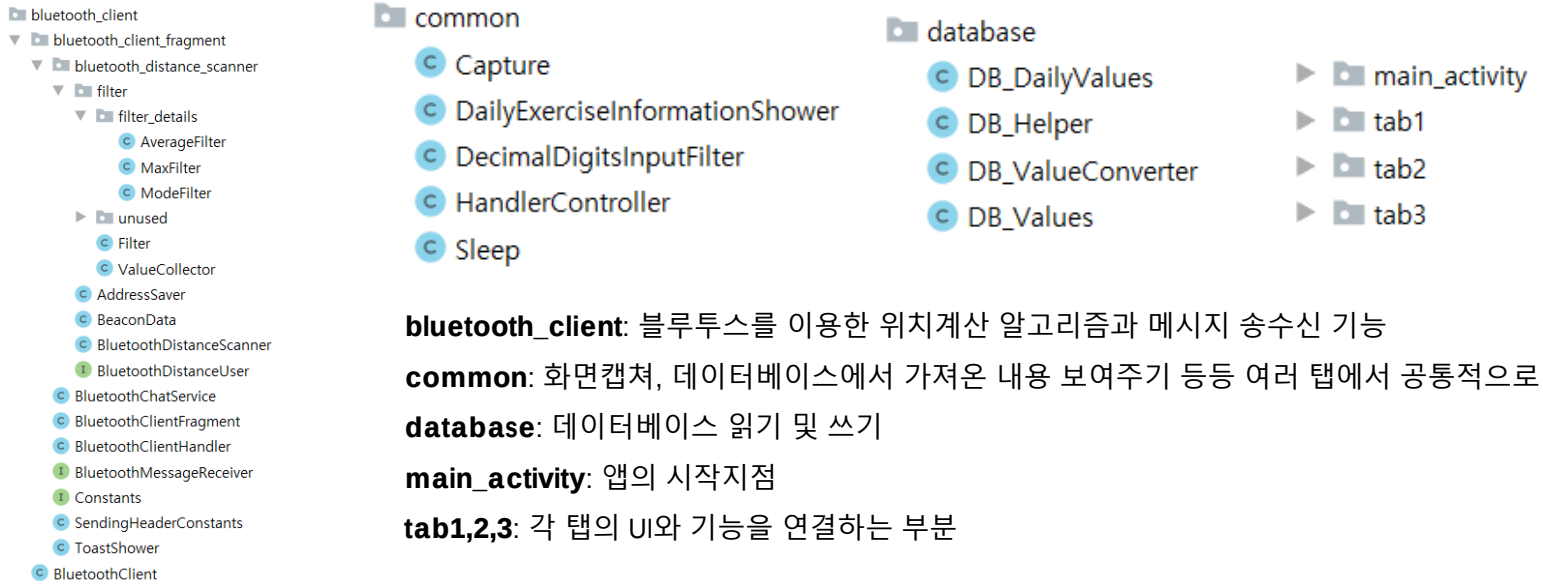
kjh_custom_tabs 패키지의 클래스는 구글에서 제공한 소스 파일인 SlidingTabLayout.java를 수정하고 기능을 추가해서 만든 클래스로 나름대로 다른 데에서 끌어다가 유용하게 쓸 수 있도록 만든 모듈이고, **소스 파일 상단에 한국어 주석으로 사용방법을 설명하였습니다.**

II. Android 개발 프로젝트

전자통신공학과 캡스톤디자인 - 2015년 9월 ~ 2016년 5월

- 블루투스 신호 세기를 계산해서 사용자의 운동기구 탑승 여부를 확인하고 입력한 정보를 운동기구에 자동으로 설정되게 하는 앱

코드 구조 – 패키지 내부



bluetooth_client: 블루투스를 이용한 위치계산 알고리즘과 메시지 송수신 기능

common: 화면캡쳐, 데이터베이스에서 가져온 내용 보여주기 등등 여러 탭에서 공통적으로 사용할 기능들

database: 데이터베이스 읽기 및 쓰기

main_activity: 앱의 시작지점

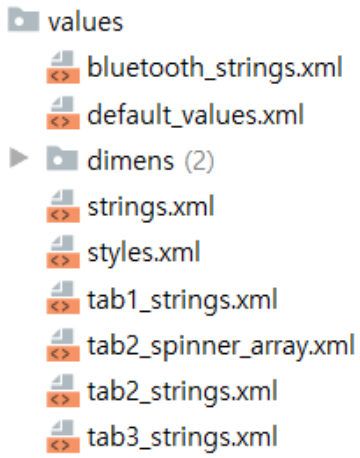
tab1,2,3: 각 탭의 UI와 기능을 연결하는 부분

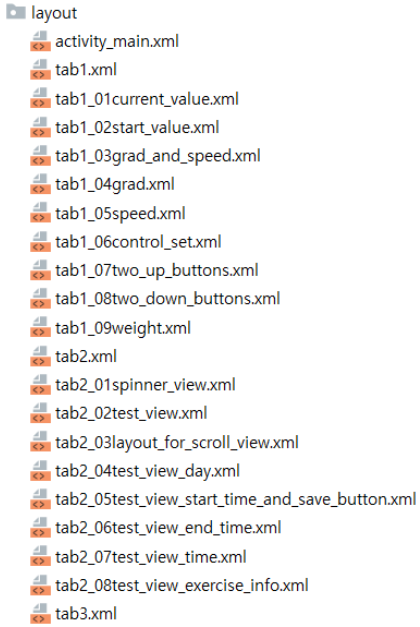
II. Android 개발 프로젝트

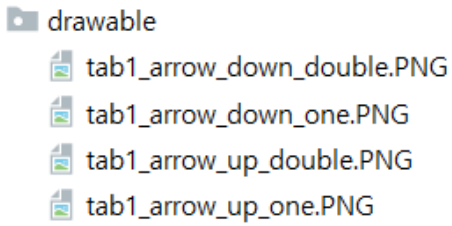
전자통신공학과 캡스톤디자인 - 2015년 9월 ~ 2016년 5월

- 블루투스 신호 세기를 계산해서 사용자의 운동기구 탑승 여부를 확인하고 입력한 정보를 운동기구에 자동으로 설정되게 하는 앱

코드 구조 – UI







drawable: 첫 번째 탭에 보이는 화살표 아이콘 파일

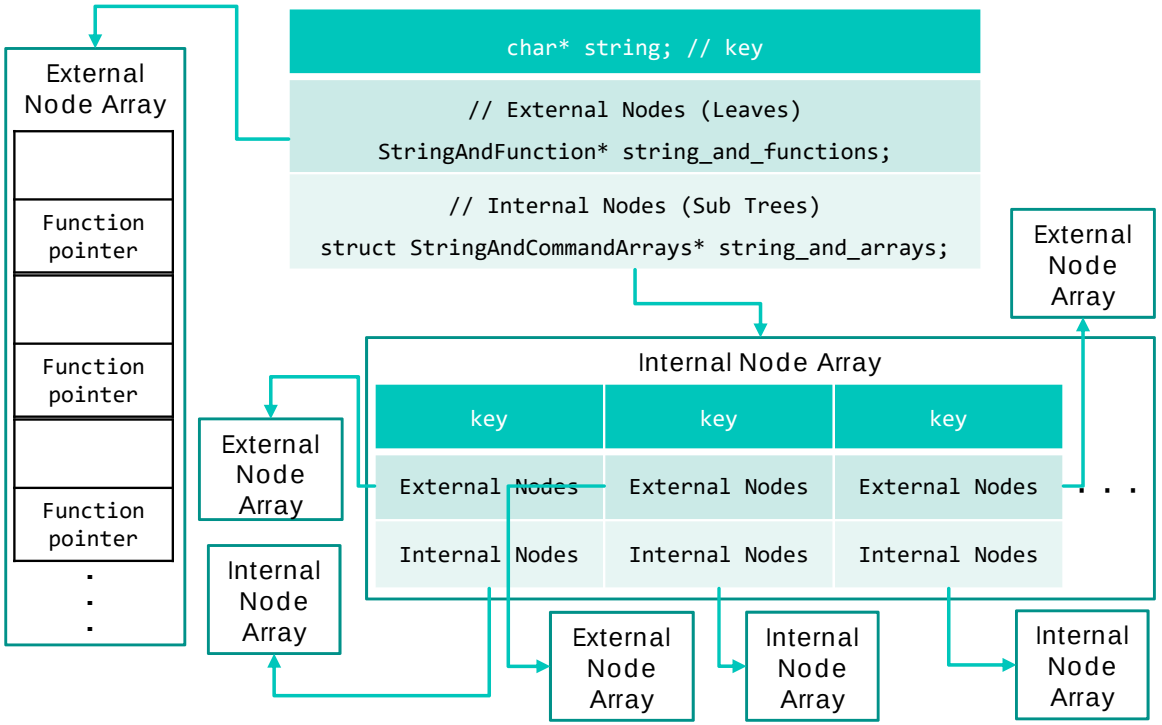
values: 화면에 표시할 글자들

layout: 화면에 보여줄 컴포넌트를 정의

III. 데이터 구조 설계/활용

LoRa 기반 무선 에너지 미터계 개발 - 2017년 9월 ~ 2018년 8월

- 트리(tree) 구조를 이용한 명령어 해석기 - 설계 (C언어)




```
typedef struct {
    char* string; // key
    // External Nodes (Leaves)
    StringAndFunction* string_and_functions;
    // Internal Nodes (Sub Trees)
    struct StringAndCommandArrays* string_and_arrays;
}
StringAndCommandArrays;
```

Internal Node 구현

```
typedef LorawanError_t (*Function)(char* param);
typedef struct {
    char* string;
    Function function;
} StringAndFunction;
```

External Node 구현

명령어 해석기 개발을 위해 트리형태로 자료구조를 설계하고, 트리는 구조체와 함수포인터, 배열에 대한 포인터를 이용해서 구현하였습니다.

https://github.com/kjh0311/kim_jin_hee-projects/tree/master/2017~2018%20-%20Graduate

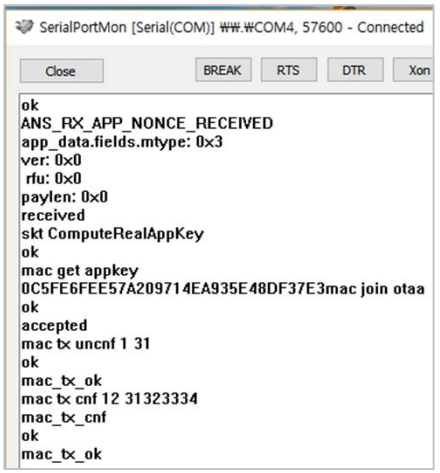
III. 데이터 구조 설계/활용

LoRa 기반 무선 에너지 미터계 개발 - 2017년 9월 ~ 2018년 8월

- 트리(tree) 구조를 이용한 명령어 해석기 - 활용 예시 (C언어)

2.4 Media Access Controller (MAC) Commands .....
- 2.4.1 mac reset .....
- 2.4.2 mac tx <type> <portno> <data> .....
- 2.4.3 mac join <mode> .....
- 2.4.4 mac save .....
- 2.4.5 mac forceENABLE .....
- 2.4.6 mac pause .....
- 2.4.7 mac resume .....
- 2.4.8 MAC Set Commands .....
  - 2.4.8.1 mac set devaddr <address> .....
  - 2.4.8.2 mac set deveui <devEUI> .....
  - 2.4.8.3 mac set appeui <appEUI> .....
  - 2.4.8.4 mac set nwkskey <nwkSessKey>
  - 2.4.8.5 mac set appskey <appSessKey>
  - 2.4.8.6 mac set appkey <appKey> .....
  - 2.4.8.7 mac set pwridx <pwrIndex> .....

구현할 명령어 모음 (일부)



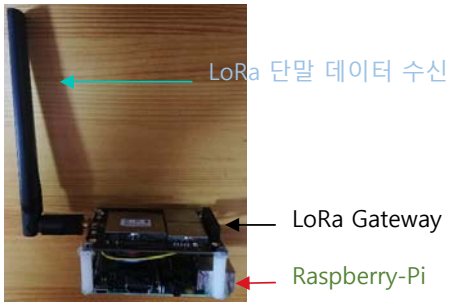
SKT 테스트베드에서 실험결과

- 가장 왼쪽에 있는 그림은 Microchip 사의 LoRa Development Kit의 설명서에서 발췌한 내용의 일부입니다.
- 그 내용을 참고로 명령어 해석기를 직접 설계해서 개발하였고, LoRaWAN Specification을 KR region 파라미터에 따라 맞게 구현했는지 확인하기 위해서 SKT사의 테스트베드에서 제가 개발한 명령어 해석기를 활용하였습니다.
- **skt ComputeRealAppKey** 명령은 SKT 테스트베드에서 사용할 목적으로 만든 명령어입니다.

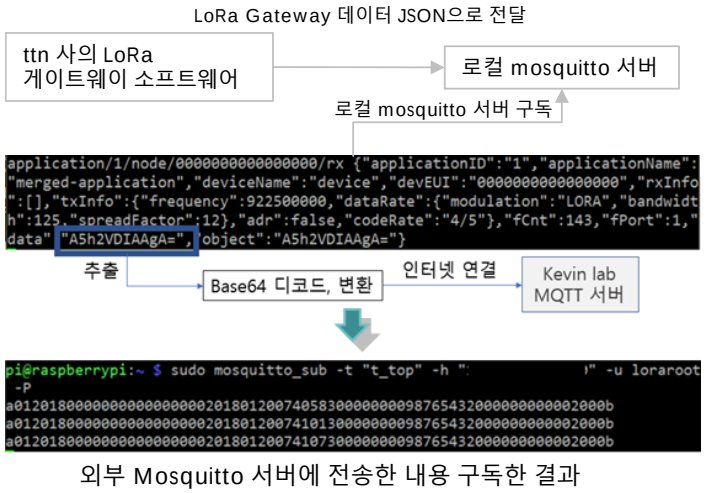
III. 데이터 구조 설계/활용

LoRa 기반 무선 에너지 미터계 개발 - 2017년 9월 ~ 2018년 8월

- JSON <-> Dictionary 변환 및 활용 (Python)



하드웨어 연결



외부 Mosquitto 서버에 전송한 내용 구독한 결과

- 이 프로그램을 개발함으로서, 시스템 간에 통신할 때는 **JSON** 형태로 통신하고, HEX 데이터는 **Base64**로 인코딩해서 JSON 문자열 안에 포함시키는 것이 효율적인 방법이라는 것을 알았고, JSON 문자열은 Dictionary 구조로 변환하여 key 값으로 접근하여 다룬다는 점을 터득했습니다.

- **ttn사의 게이트웨이 소프트웨어**는 로컬 mosquitto 서버에 LoRa Gateway가 받은 패킷을 **JSON** 문자열을 출력하는데,

이 프로그램은 그 내용을 읽어서 **JSON** 문자열에서 **BASE64**로 인코딩된 데이터를 디코드하여 아래 그림에 출력되는 형태의 문자열로 변환하여 인터넷 상에 있는 외부 **Mosquitto** 서버로 전송하는 프로그램 입니다.

다음 python 코드는 JSON으로 전달된 데이터를 Dictionary로 변환하여 'data'라는 키에 해당하는 base64 데이터를 디코드합니다.

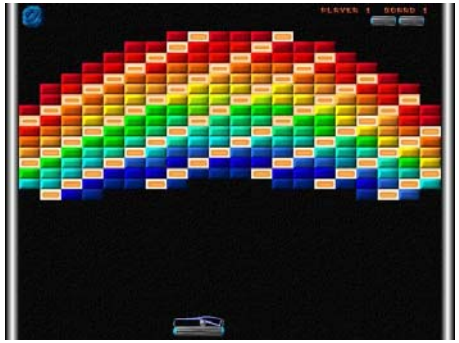
```
json_string = get_json_string(mqtt_line)
dictionary = json.loads(json_string)
```

```
base64_data = dictionary['data']
hex_received_data = base64.b64decode(base64_data)
```

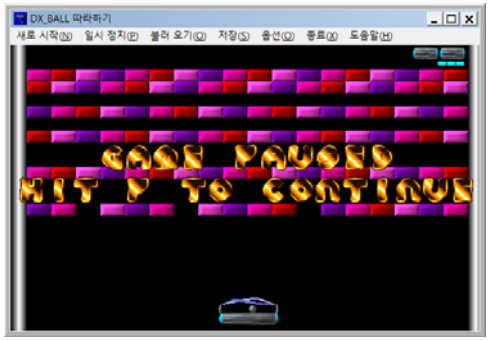
III. 데이터 구조 설계/활용

Dx-Ball 게임 모방 (2012년 7~8월)

- 연결 리스트(Linked List) (C/C++)



원본



모방

```
BallStruct* CreateBall(float x,float y,float dx, float dy){
    EnterCriticalSection(&Ball_CRT);
    struct BallStruct NewBall;
    NewBall.x=x;
    NewBall.y=y;
    NewBall.dx=dx;
    NewBall.dy=dy;
    NewBall.Grabbed=FALSE;
    NewBall.next=NULL;
    NewBall.back=NULL;  //초기값이 반드시 있어야함
    BallCount++;
    if(FirstBall == NULL)
    {// 새 메모리 할당
        FirstBall=(BallStruct*)malloc(sizeof(BallStruct));
        *FirstBall = NewBall;   // 새로 만든 미사일을 할당된 메모리 공간으로 복사
        LastBall = FirstBall;   // Tail도 Head와 동일한 위치로 업데이트
    }
    else{//헤드가 아닌 경우
        // 새로운 공간을 할당하여 Tail->Next에 붙임

        NewBall.back = LastBall;
        LastBall->next = (BallStruct*)malloc(sizeof(BallStruct));
        LastBall = LastBall->next; // 새 공간을 마지막으로 지정
        *LastBall = NewBall;
    }
    LeaveCriticalSection(&Ball_CRT);
    return LastBall;    // 맨 마지막 것(새로 만든 것)을 반환값으로 지정
}
```

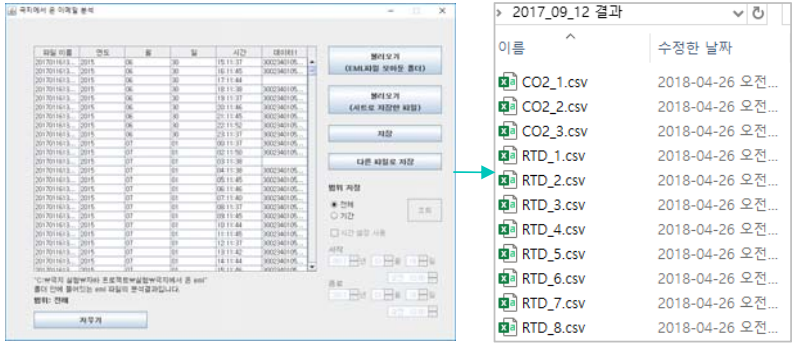
오른쪽의 코드는 게임에서 아이템 습득 시 공 개수 추가하는 코드로 연결 리스트(Linked List)를 이용해서 구현한 것 입니다.

https://github.com/kjh0311/kim_jin_hee-projects/tree/master/2012%20(Beginner)%20-%20Imitation%20of%20DX-Ball%20Game

IV. 프로그램 설계

ICT 기반의 융복합 극지 환경 관측 시스템 개발 (2017년 3월 ~ 2018년 8월)

- **다량의 이메일 파일(EML) 읽어서 필요한 데이터를 가지고서 스프레드 시트 파일(CSV) 자동으로 생성하는 프로그램 개발, GUI 개발 (Java)**




```
public OneEmlFileParser(String string_output_csv_dir)
{
    rtd = new OneSbdPartParser("RTD", Type.SHORT, 12, 4, 1, string_output_csv_dir);
    tc = new OneSbdPartParser("TC", Type.SHORT, 7, 8, 0.25, string_output_csv_dir);
    co2 = new OneSbdPartParser("CO2", Type.SHORT, 1, 8, 5000.0/4095.0, string_output_csv_dir);
    wc = new OneSbdPartParser("WC", Type.FLOAT, 12, 2, 1, string_output_csv_dir);

    mail_session = getMailSession();
}
```

클래스와 생성자를 이용한 논리 간소화

OneSbdPartParser 클래스를 통해 여러 종류의 센서데이터를 추출해서 파일로 저장하도록 작성

- 극지에서 기후환경에 대한 센서수집 데이터를 기록한 이메일 파일들을 폴더째로 불러와서 데이터별로 csv 파일로 정리하도록 하는 프로그램입니다.

https://github.com/kjh0311/kim_jin_hee-projects/tree/master/2017~2018%20-%20Graduate

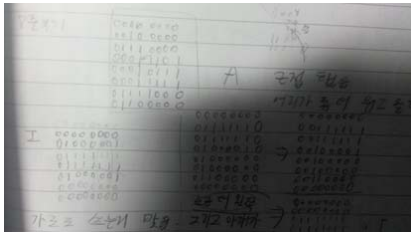
IV. 프로그램 설계

전자통신공학과 - 마이크로프로세서 응용 (2015년 3~6월)

- **문자 코드 생성기 제작 - 과정 (Java)**

1. 알파벳 A를 표기하는 다음 코드를 종이에 적어 분석

   const char font_A[8] = { 0x00,0x40,0x70,0x1D,0x17,0x1F,0x78,0x60 }; // A



개발 당시 찍은 사진입니다.

2. 종이에 분석한 것을 다음과 같이 자바로 구현

- 만들어 져야 할 코드
{0X10, 0X12, 0XD5, 0XD5, 0XD5, 0X12, 0X10, 0X00}
- 위 배열은 각 행 별로 값이 있으며, 그 값은 그 행에 어떤 열에 점을 찍어야 할 지를 보여준다.

```
int result[] = new int[Main.cols];
for(int i=0;i<Main.rows;i++){
  for(int j=0;j<Main.cols;j++){
    if (Main.record[i][j]){ // 점 찍은 경우
      result[j] |= 1 << i;
    }
  }
}
```

```
private String getHex(int iv){
  String hex = "";
  hex = Integer.toHexString(iv).toUpperCase();
  if (hex.length()==1)
    hex = "0"+hex;
    hex = "0X"+hex;
  return hex;
}
```

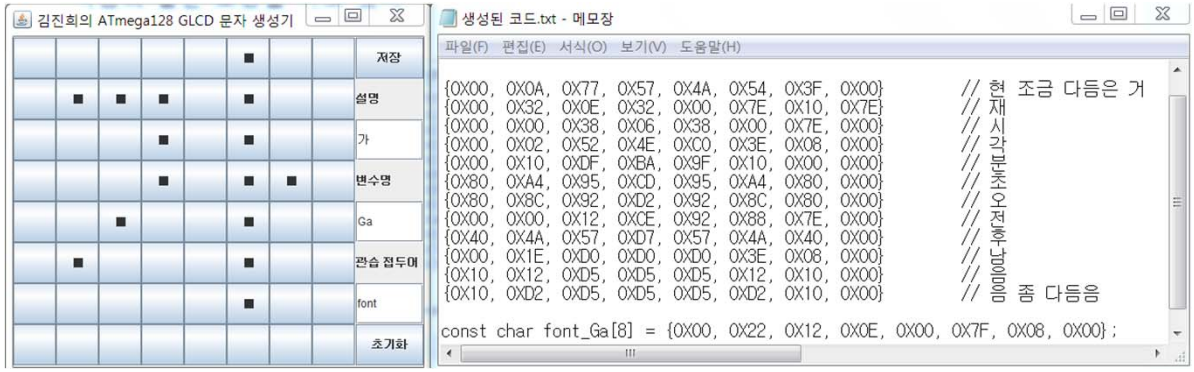
```
String s = "{";
String explainText;
s += getHex(result[0]);
for (int i=1; i<Main.rows;i++){
  s += ", ";
  s += getHex(result[i]);
}
s += "};";
```

https://github.com/kjh0311/kim_jin_hee-projects/tree/master/2015%20-%20Microprocessor/Application%20of%20microprocessor

IV. 프로그램 설계

전자통신공학과 - 마이크로프로세서 응용 (2015년 3~6월)

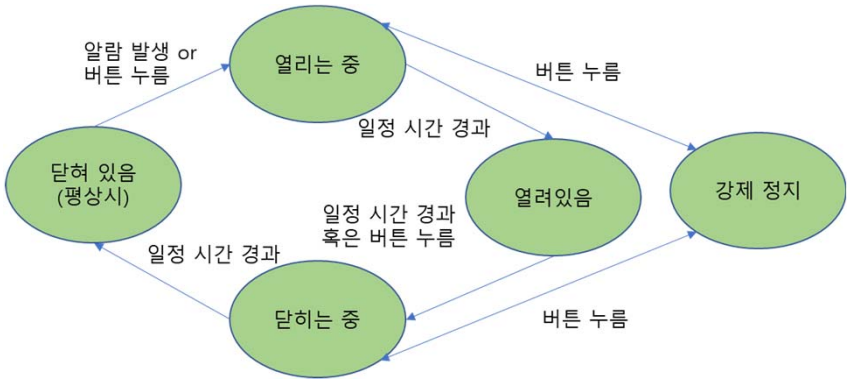
- **문자 코드 생성기 제작 - 결과 (Java)**
  - 한글문자 표시를 위해,
    알파벳 A를 비롯해 폰트를 표현하도록 제공된 다음과 같은 코드를 분석하여 문자코드 생성기를 제작하였습니다.
    const char font_A[8] = { 0x00,0x40,0x70,0x1D,0x17,0x1F,0x78,0x60 }; // A



IV. 프로그램 설계

전자통신공학과 - 마이크로프로세서 응용 (2015년 3~6월)

- **스텝모터 작동 프로그램 설계에 FSM (Finite State Machine) 그래프 도입 - 1 (C언어)**




```
#define       DOOR_CLOSED   0
#define       DOOR_OPENING 1
#define       DOOR_OPENED   2
#define       DOOR_CLOSING 3
// 버튼을 누를 경우 발생
#define       DOOR_STOPPED 4

typedef unsigned char State;
typedef struct
{
    State mode;
    SecondAndTenMili openingStartTime;
    SecondAndTenMili openedStartTime;
    SecondAndTenMili closingStartTime;
} DoorInfo;
```

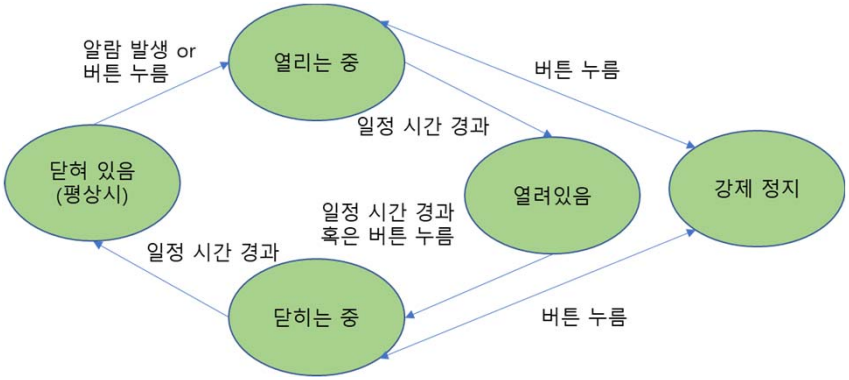
```
DoorInfo Door = {DOOR_CLOSED, 0, 0};
ShortInt DoorControl(bool keyPressed, ShortInt keyBuf)
{
    if ( (keyPressed && keyBuf == DOOR_BUTTON) ||
            doorTimeExpired(&Door)    ) {
        Door.mode = ToNextState(&Door, keyPressed);
    }
    return DoorControlForState(&Door);
}
```

- 왼쪽과 같이 그래프를 그려서 그 내용을 오른쪽과 같이 구현하였습니다. (구현 내용은 다음페이지까지 계속)

IV. 프로그램 설계

전자통신공학과 - 마이크로프로세서 응용 (2015년 3~6월)

- **스텝모터 작동 프로그램 설계에 FSM (Finite State Machine) 그래프 도입 - 2 (C언어)**




```
bool doorTimeExpired(DoorInfo *pInfo)
{
    switch(pInfo->mode)
    {
    case DOOR_CLOSED:
        return checkAlarmTime();
    case DOOR_OPENING:
        return checkDoorOpeningTime
            (pInfo->openingStartTime);
    case DOOR_OPENED:
        return checkDoorOpenedTime
            (pInfo->openedStartTime);
    case DOOR_CLOSING:
        return checkDoorClosingTime
            (pInfo->closingStartTime);
    default:
        return false;
    }
}
```

```
State ToNextState(DoorInfo *pInfo, ShortInt keyPressed)
{
    switch(pInfo->mode){
    case DOOR_CLOSED:
        return openStart(pInfo);
    case DOOR_OPENING:
        return openedStart(pInfo, keyPressed);
    case DOOR_STOPPED: // 강제로 멈춘경우
    case DOOR_OPENED:
        return closeStart(pInfo, keyPressed);
    case DOOR_CLOSING:
        return closingToNext(keyPressed);
    default:
        return pInfo->mode;
    }
}
```

감사합니다