

Chapter 259. 피아노

1. 아두이노

스마트폰과 아두이노를 블루투스로 연결하고 스마트폰에 표시된 피아노 건반을 눌러 해당 음이 아두이노에서 재생되는 미니 피아노를 만들어 보자. 단음 재생을 위해 아두이노 우노에 스피커를 그림 1과 같이 연결한다.

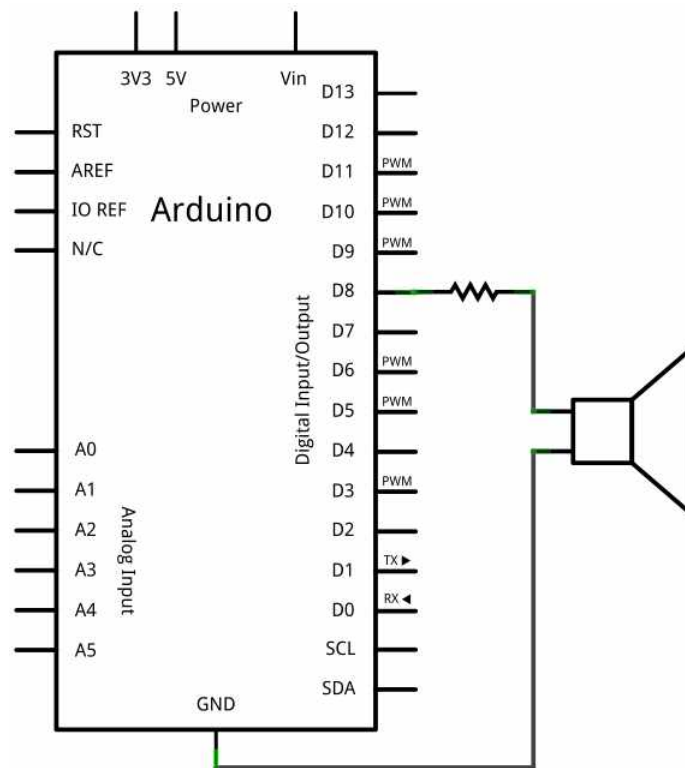


그림 1. LED 연결 회로도

단음 재생을 위해서는 tone 함수를 사용할 수 있다. tone 함수는 50% 듀티 사이클과 지정된 주파수의 구형파(square wave)를 이용하여 스피커나 피에조(piezo) 부저로 음을 재생한다. tone 함수를 사용하기 위해서는 재생하고자 하는 음의 주파수와 밀리초 단위의 재생 시간을 지정해 주어야 한다. 음 높이에 따른 재생 주파수의 예는 다음과 같다.

```

#define NOTE_C4 262          // 도
#define NOTE_D4 294          // 레
#define NOTE_E4 330          // 미
#define NOTE_F4 349          // 파
#define NOTE_G4 392          // 솔
#define NOTE_A4 440          // 라
#define NOTE_B4 494          // 시
#define NOTE_C5 523          // 도

```

스케치 1은 그림 1과 같이 연결된 스피커를 통해 학교종 멜로디를 재생하는 스케치의 예이다.

스케치 1. 학교종 멜로디 재생

```

#define NOTE_C4 262
#define NOTE_D4 294
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_G4 392
#define NOTE_A4 440
#define NOTE_B4 494
#define NOTE_C5 523

int speakerPin = 8;          // 스피커 연결 핀

int melody[] = {             // 학교종 멜로디. 0은 침표
    NOTE_G4, NOTE_G4, NOTE_A4, NOTE_A4, NOTE_G4, NOTE_G4, NOTE_E4, 0,
    NOTE_G4, NOTE_G4, NOTE_E4, NOTE_E4, NOTE_D4, 0,
    NOTE_G4, NOTE_G4, NOTE_A4, NOTE_A4, NOTE_G4, NOTE_G4, NOTE_E4, 0,
    NOTE_G4, NOTE_E4, NOTE_D4, NOTE_E4, NOTE_C4, 0
};

int noteDuration = 250;

void setup() {

```

```

}

void loop() {
  for (int thisNote = 0; thisNote < sizeof(melody) / sizeof(int); thisNote++) {
    tone(speakerPin, melody[thisNote], noteDuration);    // 멜로디 재생
    delay(noteDuration);
  }

  while(true){ };
}

```

2. UI 디자인

아두이노에 단음 재생을 위해 스피커 연결이 완료되었으니 어플리케이션의 UI를 만들어보자. 안드로이드 스튜디오에서 새 프로젝트를 선택하고 프로젝트 이름은 'Bluetooth_Piano'로 하자. 프로젝트 생성을 위한 옵션은 디폴트값을 사용한다. 건반을 나타내기 위해서는 버튼을 사용하며 각 버튼이 눌러진 경우 해당 값을 아두이노로 전송하여 해당 음을 재생하도록 한다. 어플리케이션의 UI는 가로 모드를 사용하여 8개의 버튼을 선형 레이아웃으로 배치한다.

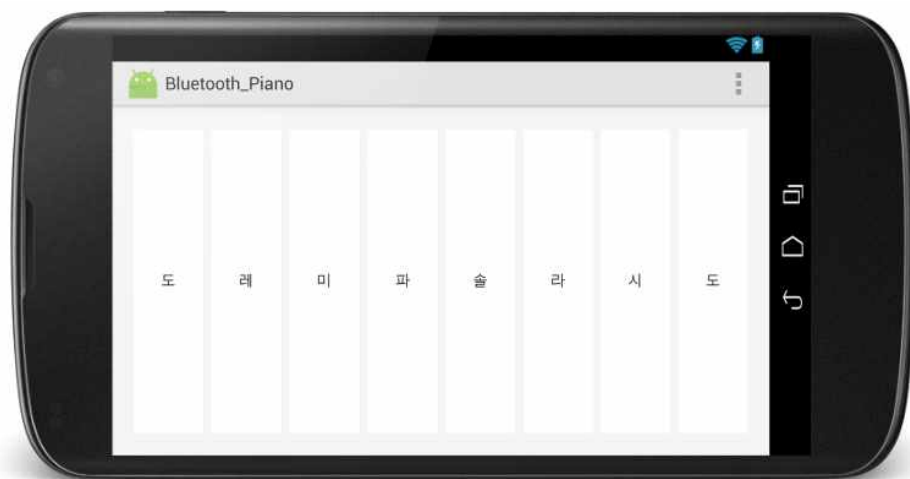


그림 2. 피아노 건반 배치

코드 2. UI 디자인

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    android:gravity="center" >
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="도"
    android:id="@+id/button1"
    android:layout_weight="1" />
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="레"
    android:id="@+id/button2"
    android:layout_weight="1" />
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="미"
    android:id="@+id/button3"
    android:layout_weight="1" />
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
```

```
        android:text="파"
        android:id="@+id/button4"
        android:layout_weight="1" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="술"
    android:id="@+id/button5"
    android:layout_weight="1" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="라"
    android:id="@+id/button6"
    android:layout_weight="1" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="시"
    android:id="@+id/button7"
    android:layout_weight="1" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:text="도"
    android:id="@+id/button8"
    android:layout_weight="1" />

</LinearLayout>
```

3. 블루투스 통신

블루투스 통신을 위한 어플리케이션의 기본 코드는 블루투스 프로그래밍 장의 코드를 사용한다. package 문장 이후 모든 코드를 블루투스 기본 코드로 대체하자. AndroidManifest.xml 파일에서 블루투스 사용을 위한 퍼미션을 최상위 요소 아래에 추가하는 것도 잊지 말아야 한다.

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

먼저 버튼이 눌러진 경우 이벤트를 처리하기 위해 MainActivity 클래스가 OnClickListener를 구현하도록 하자. OnClickListener를 구현하면 onClick 멤버 함수를 반드시 구현하여야 하므로 onClick 함수 정의 역시 추가하자. 다음은 onCreate 함수에서 버튼의 이벤트 처리를 MainActivity 클래스가 담당하도록 지정해 주어야 한다.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button btn;
    btn = (Button)findViewById(R.id.button1);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button2);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button3);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button4);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button5);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button6);
    btn.setOnClickListener(this);
    btn = (Button)findViewById(R.id.button7);
```

```
btn.setOnClickListener(this);  
btn = (Button)findViewById(R.id.button8);  
btn.setOnClickListener(this);  
  
checkBluetooth();  
}
```

버튼이 눌리지면 MainActivity 클래스의 onClick 함수가 호출되므로 onClick 함수에서는 현재 눌려진 버튼에 대한 정보를 블루투스를 통해 아두이노로 전달하면 된다.

```
@Override  
public void onClick(View v) {  
    String str = new String();  
  
    switch(v.getId()){  
        case R.id.button1:  
            str = "1";  
            break;  
        case R.id.button2:  
            str = "2";  
            break;  
        case R.id.button3:  
            str = "3";  
            break;  
        case R.id.button4:  
            str = "4";  
            break;  
        case R.id.button5:  
            str = "5";  
            break;  
        case R.id.button6:  
            str = "6";  
            break;  
        case R.id.button7:
```

```

        str = "7";
        break;
    case R.id.button8:
        str = "8";
        break;
    }

    sendData(str);
}

```

어플리케이션을 스마트폰에 설치하고 아두이노에는 스케치 3을 업로드 하자. 스케치 3은 블루투스를 통해 수신된 데이터를 시리얼 모니터에 표시하고 해당 음을 재생하는 예이다.

스케치 3. 블루투스 수신 테스트

```

#include <SoftwareSerial.h>

#define NOTE_C4  262
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_G4  392
#define NOTE_A4  440
#define NOTE_B4  494
#define NOTE_C5  523

// SoftwareSerial(RX, TX) 형식으로 블루투스 모듈과 교차하여 연결된다.
SoftwareSerial BTSerial(2, 3);

int speakerPin = 8;           // 스피커 연결 핀
int noteDuration = 250;

int NOTES[8] = {
    NOTE_C4, NOTE_D4, NOTE_E4, NOTE_F4,
    NOTE_G4, NOTE_A4, NOTE_B4, NOTE_C5 };

```

```
String note_name[] = { "Do", "Re", "Mi", "Fa", "Sol", "La", "Ti", "Do" };

void setup() {
  Serial.begin(9600);          // 컴퓨터와의 시리얼 통신 초기화
  BTSerial.begin(9600);       // 블루투스 모듈 초기화
}

void loop() {
  if(BTSerial.available()){    // 블루투스 데이터 수신
    byte data = BTSerial.read();
    if(data >= '1' && data <= '8'){ // 범위 내의 값인 경우
      tone(speakerPin, NOTES[data - '1'], noteDuration); // 음 재생
      Serial.println(note_name[data - '1']);              // 시리얼 모니터 출력
    }
  }
}
```

어플리케이션에서 건반을 누르면 해당 음의 이름이 시리얼 모니터로 출력되면서 음이 재생되는 것을 확인할 수 있다.

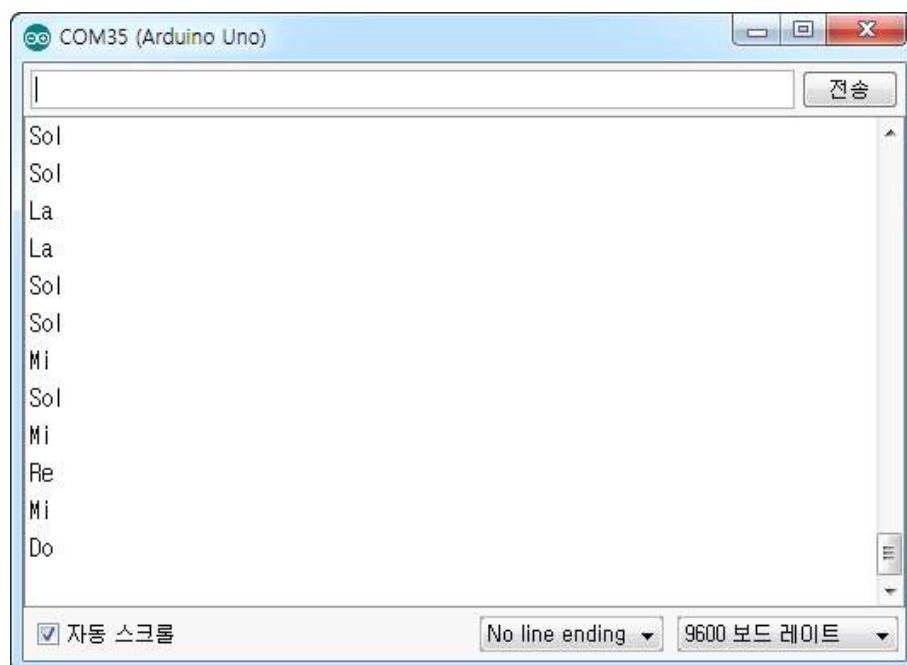


그림 3. 스케치 3 실행 결과