本日の内容

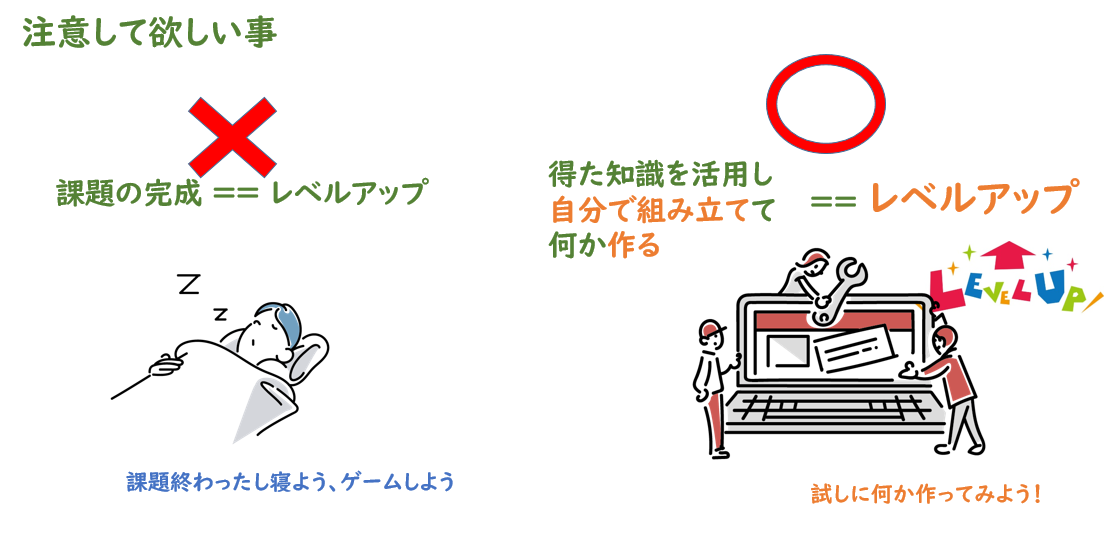
・SoundPool

・画面の向きの固定

公式ドキュメント

<https://developer.android.com/reference/android/media/SoundPool>

今回は音(SE)を実装する方法を学んでいきます。



■SoundPoolとは

音データを再生するクラスの1つ

他にはMediaPlayer , AudioTrackなどがある

・5秒程度のデータを再生出来るのでSEの再生に向いている

mp3,ogg,wavなどの拡張子を再生可能

■プロジェクト作成

- Empty Activity

- プロジェクト名：TrySoundPool

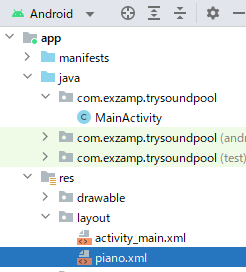
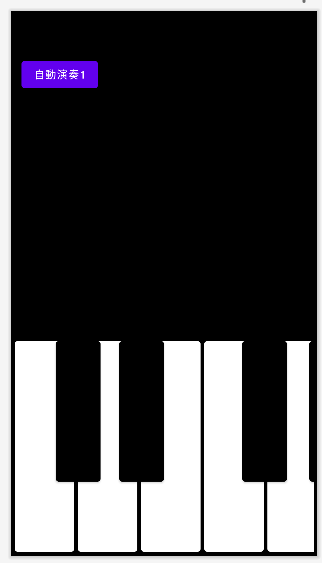
==================================

■xmlファイルの配置

今回使用するxmlは1から作成すると難易度が高い為、今回は授業用に準備したxmlを活用します。

自分でカスタマイズや自作する場合はタグなど意味を読み解いて調べて試しご自身の技術に変換してください。

授業フォルダ内のpiano.xmlをres>layoutフォルダ内にコピーしましょう。

■画面を横に固定

ピアノの画面は用意できましたが縦画面だと使い難そうです。

Androidアプリでは1画面ごとに縦/横画面で

固定する事が可能です。

次ではその設定を行っていきます。

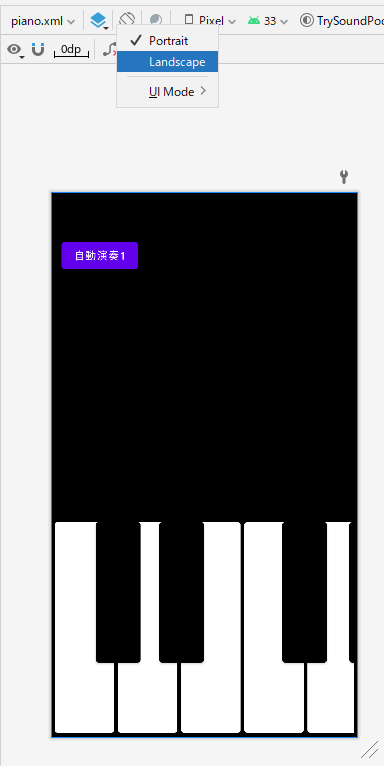
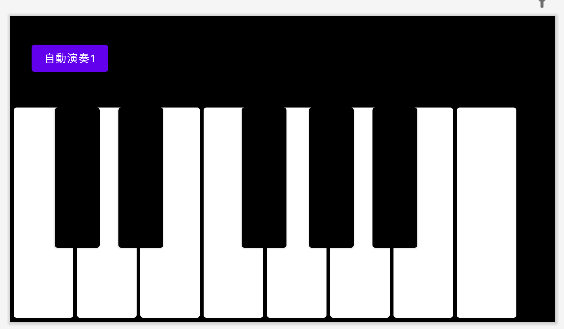
AndroidManifest.xml



MainActivityに対して、画面を横(landscape)で固定する設定です

他にも、縦固定や全画面表示などもあります。(必要になった際に調べてみましょう)

しかし、作業上では縦のままで見にくいので、作業上も横に変更しておきましょう。

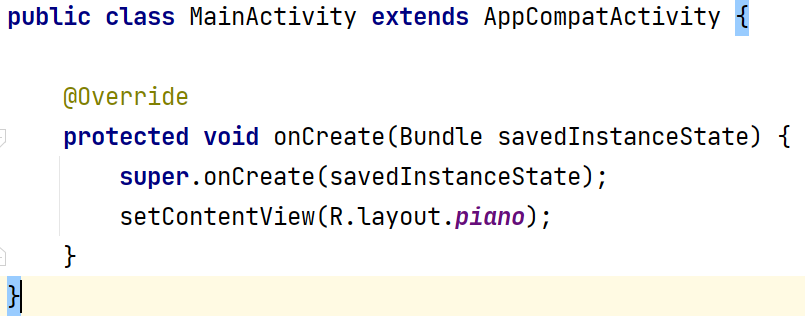
 　　　　　　

■読み込むxmlファイルを指定

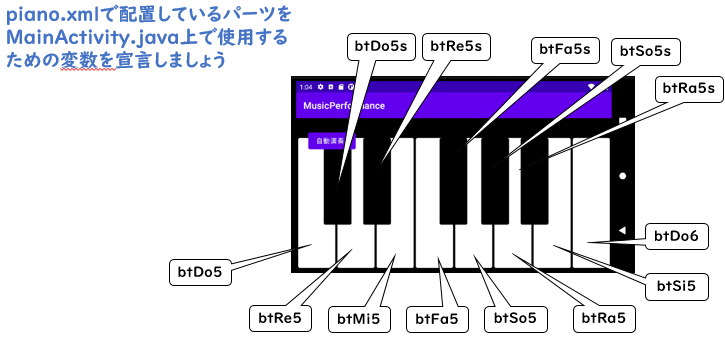
そのまま実行しても表示されるのはHello woldのレイアウトのactivity\_main.xmlになります。

先程コピーしたpiano.xmlで読み込むためには指定する必要があります。

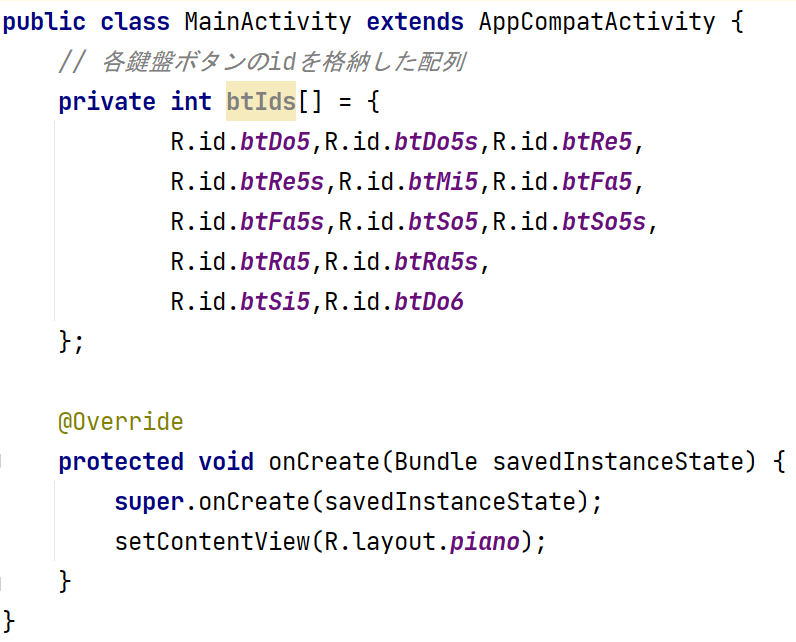
MainActivity.java



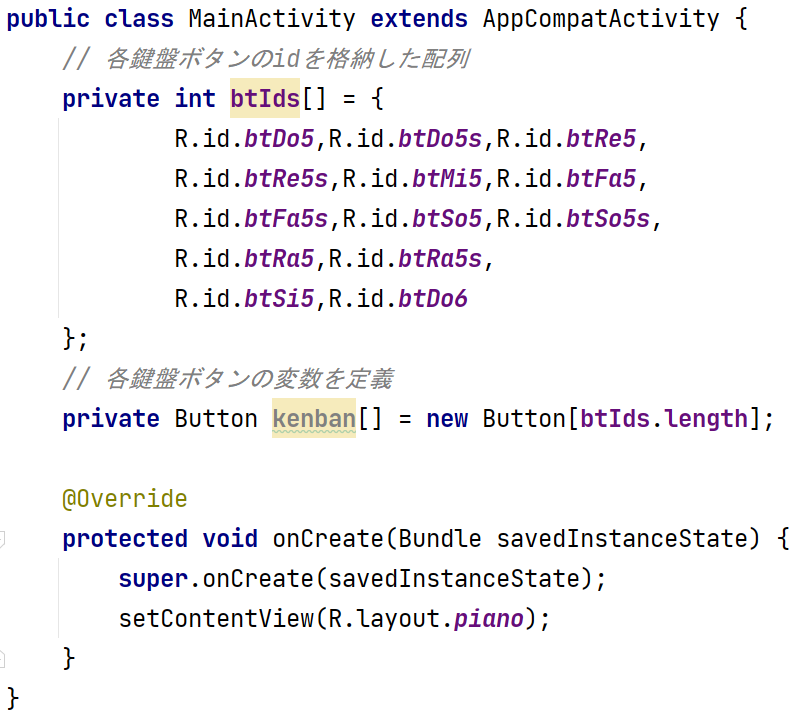
■レイアウトと変数の関連付け



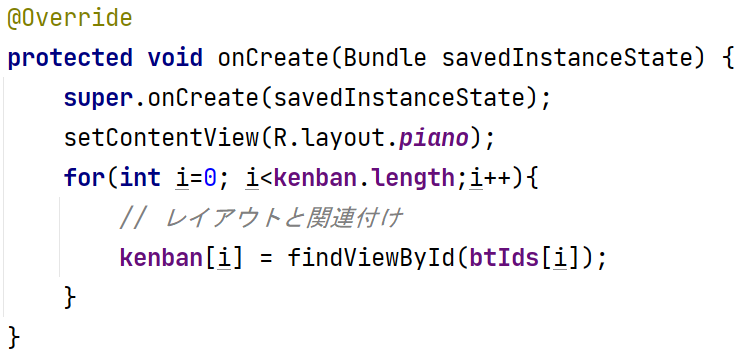
先に各ボタンのidを配列でまとめておきます。



次に各ボタンの変数を配列でまとめて定義してしましましょう



続いて関連付けも行いますが、配列で定義しているのでfor文でまとめて行うことも可能です



■音ファイルの配置

今回は効果音の再生して演奏を実現させるので音ファイルが必要です。

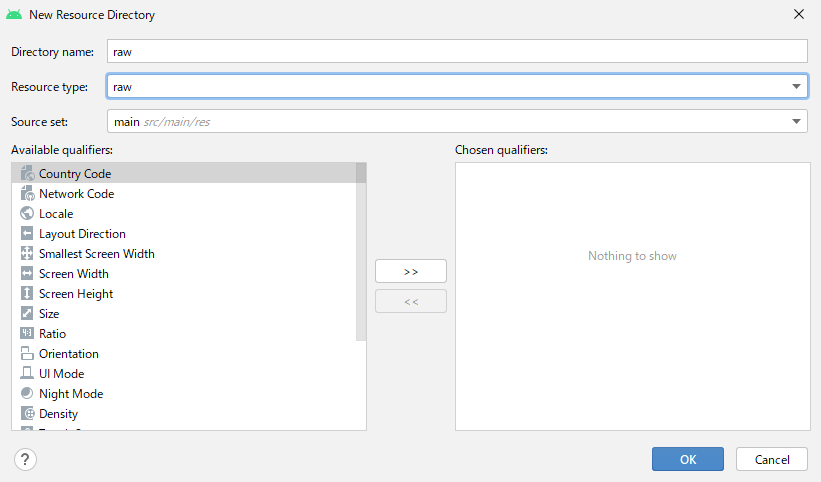
授業フォルダから音ファイルをコピーしプロジェクトへ移しましょう

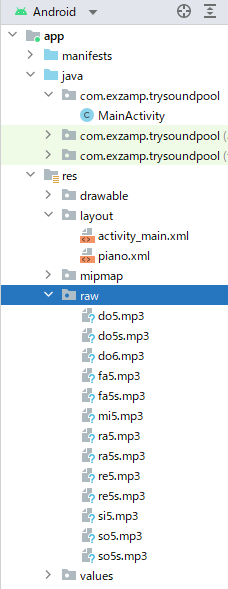
音のファイルは専用のリソースフォルダに配置します。

専用のリソースフォルダrawを作成しましょう。

res>右クリック>new>Android Resouce Directory

Resouce typeをraw>OK

作成したrawフォルダに音ファイルを全てコピーしましょう。



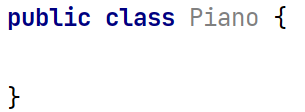
■音を再生するクラスの作成

同じMainActivity内に作成しても良いのですが管理が行いやすいように

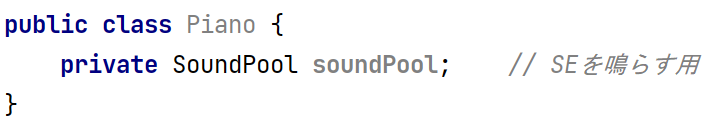
別にPianoクラスを作成しMainActivity内で活用しましょう。

New>Java Class

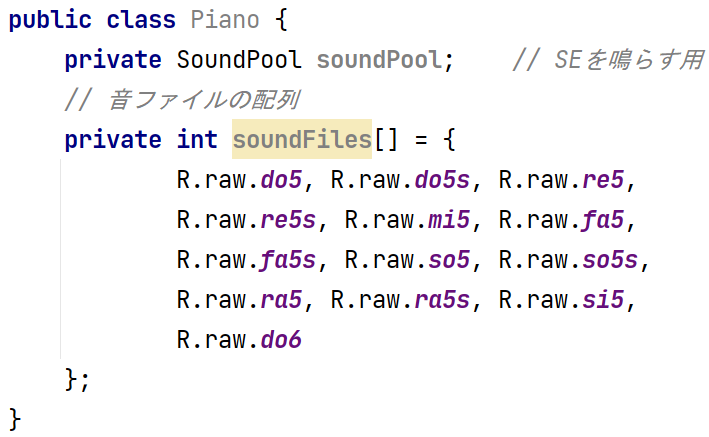
クラス名：Piano



音を鳴らすためにSoundPoolの変数 soundPoolを定義



再生する音ファイルも管理しやすいように配列で定義しておきましょう



Pianoインスタンスの初期設定を行えるようコンストラクタのPianoメソッドを定義



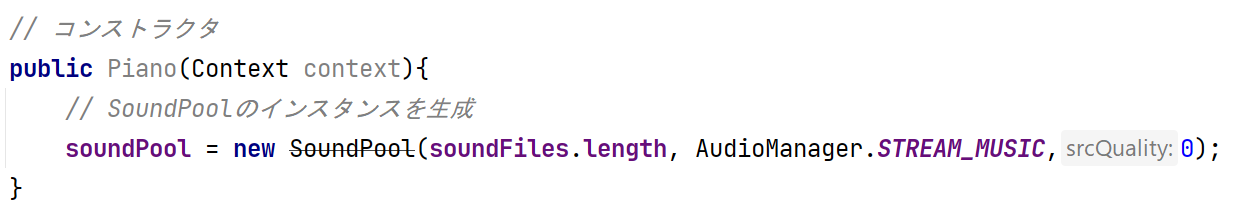
インスタンス生成する処理を追記

**SoundPool(int maxStreams, int streamType, int srcQuality)**

**maxStreams：プールする最大の数**

**streamType：Streamのタイプ（通常はSTREAM\_MUSICを利用する）**

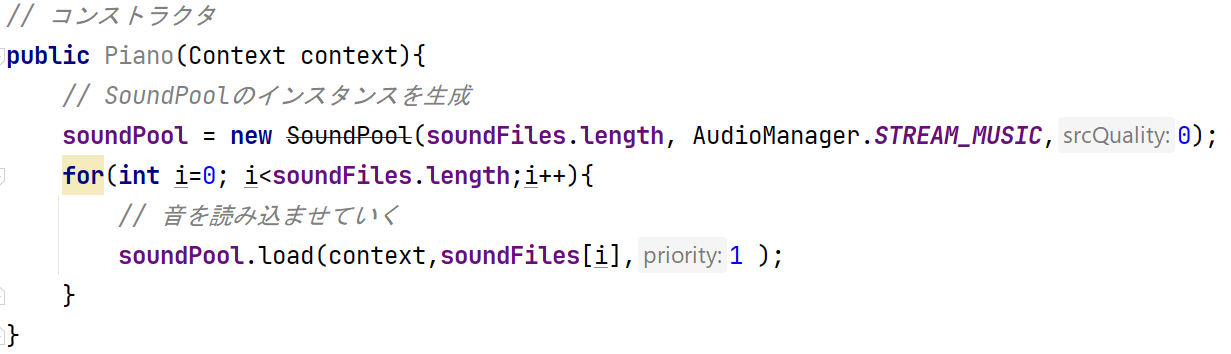
**srcQuality：サンプリングレートのクオリティ（デフォルトは0）**



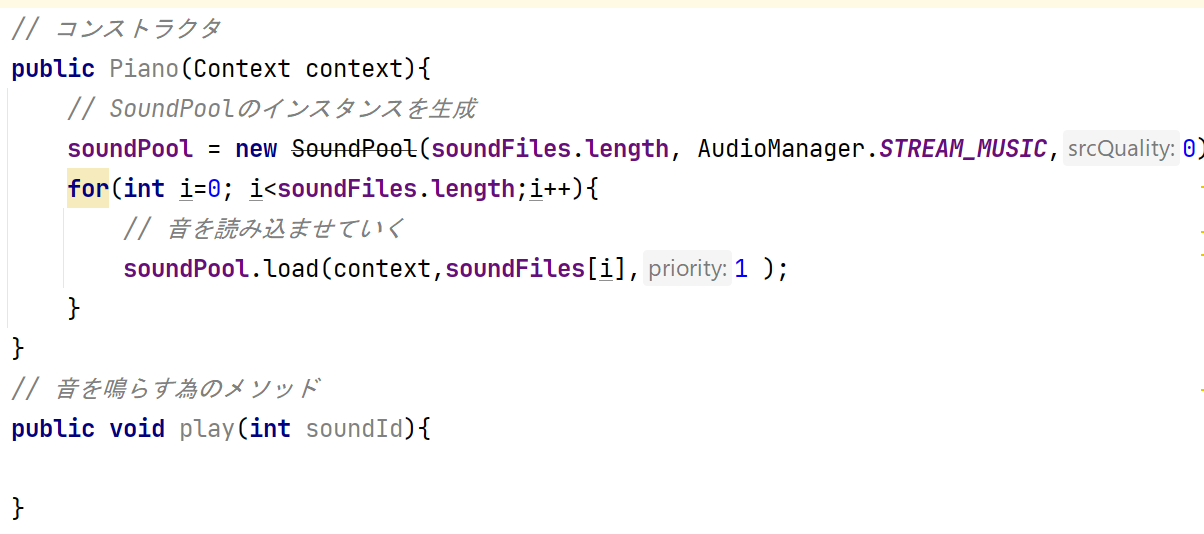
音ファイルの情報と関連付ける

**構文**

**soundPool変数.load(context , 音ファイル , 優先度)**



指定した音ファイルを再生するメソッドplayを定義

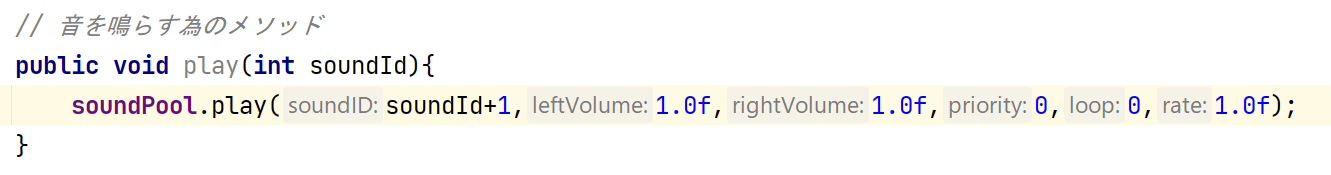


引数の値によって分岐できるよう処理を追記

引数には何番目の鍵盤が押されたかを受け取る(配列は0番目からだが、playは１から添え字が始まる)

**構文**

**soundPool変数.play(何番目の音を再生するか , 左音量 , 右音量 , 優先度 , ループ回数 , 速度)**



これでピアノ音を再生するPianoクラスが完成しました。

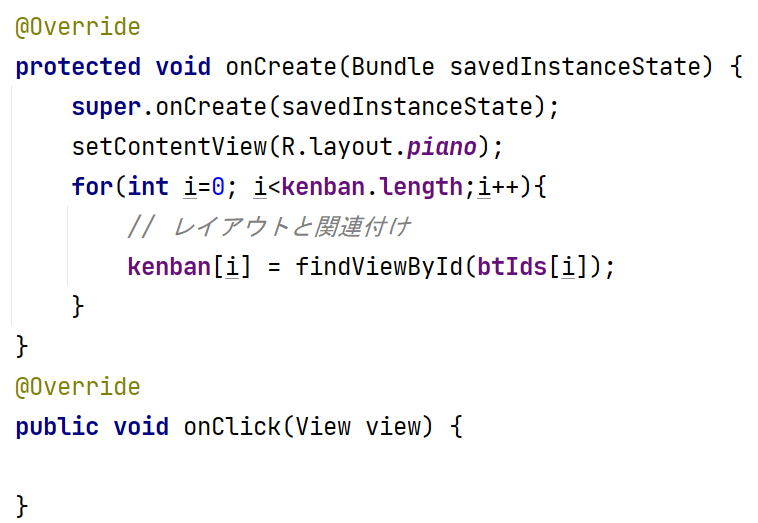
あとはMainActivityにPianoクラスを使用する処理を記述すれば演奏可能です。

■クリックリスナーのインターフェイスを実装

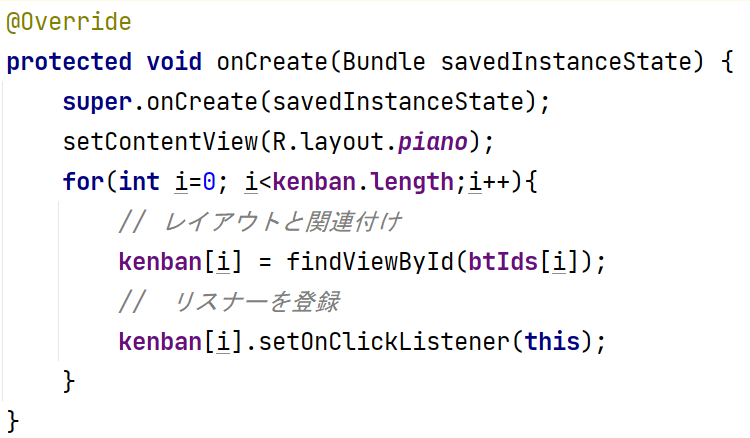
implements View.OnClickListenerを追記



onClickメソッドを実装

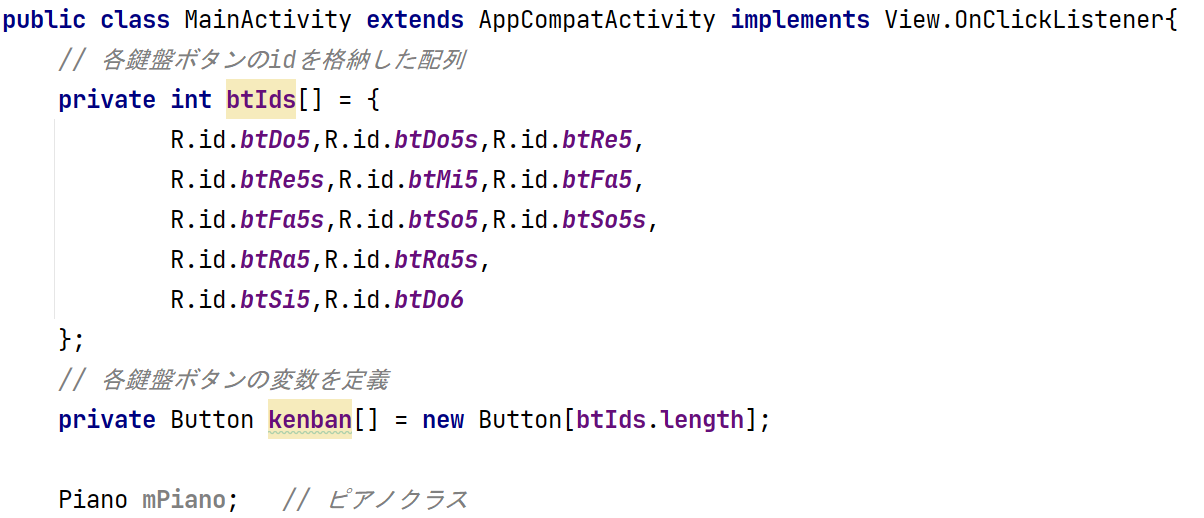


このままでは、ボタンクリックの検知が出来ませんのでリスナーを登録しましょう

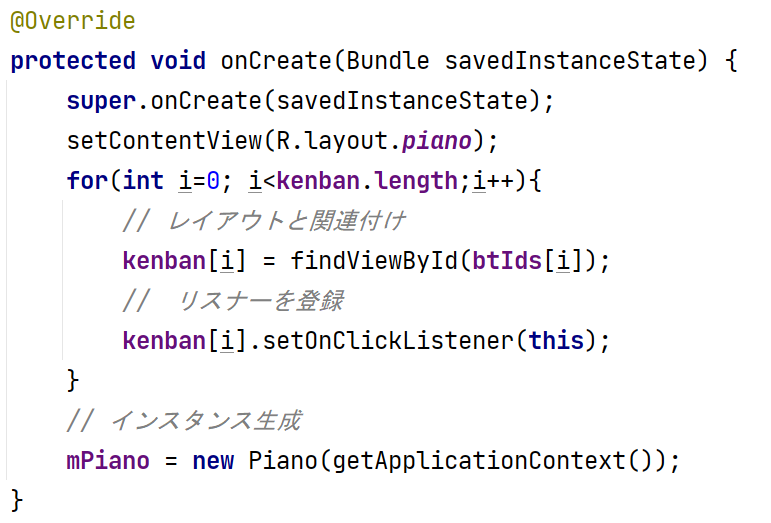


■Pianoクラスを使用して音を鳴らす

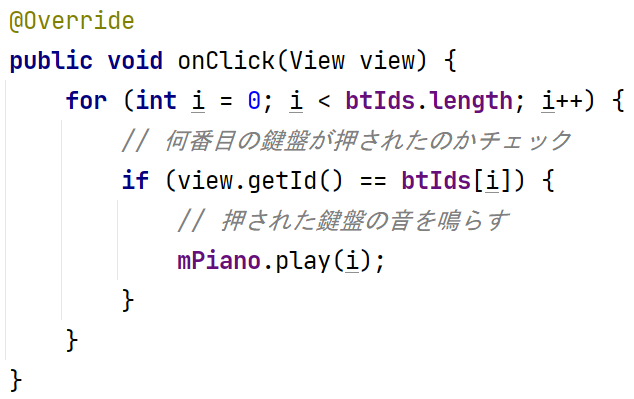
クリックイベント処理の準備も出来ましたので音を鳴らす処理を追記していきましょう



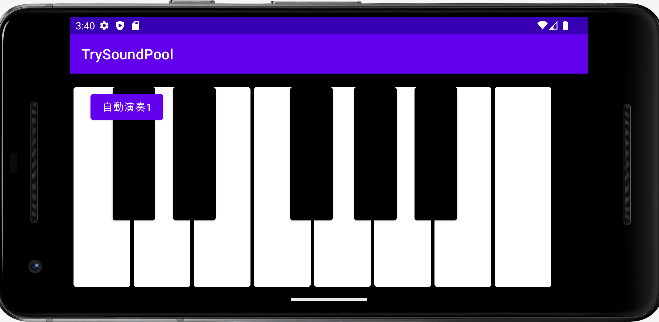
Pianoクラスのインスタンスを生成



何番目の鍵盤が押されたを特定し鳴らす音を指定する



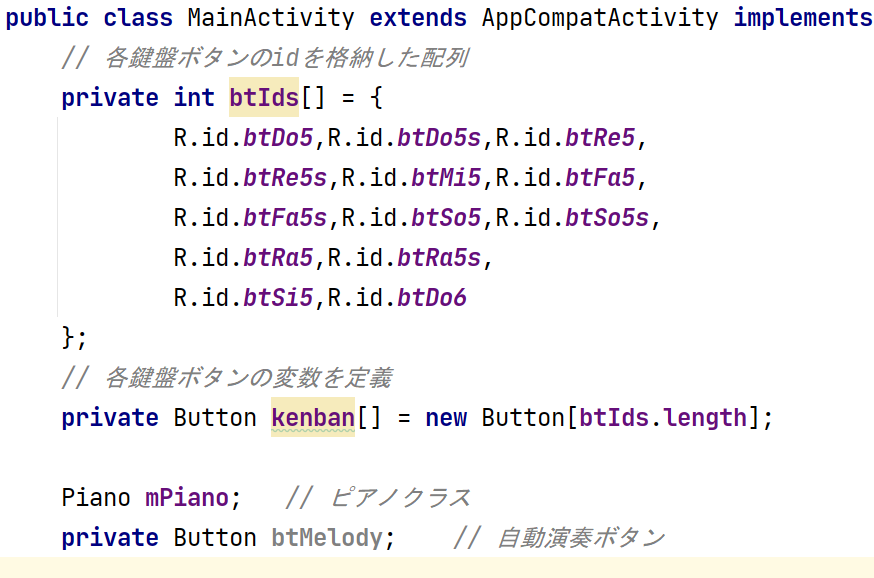
音が鳴るか確認してみましょう！

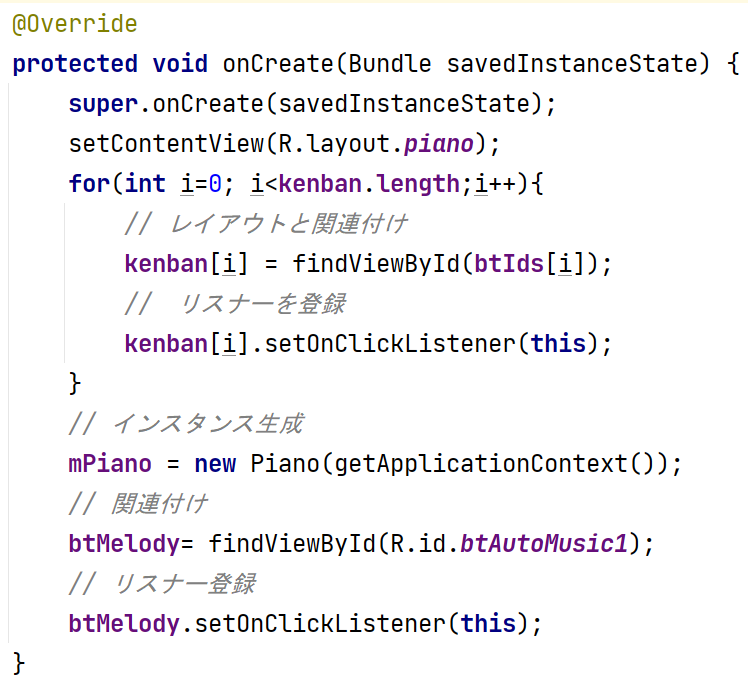


■自動演奏

自動演奏を押下でメロディが流れるように追記してみましょう！

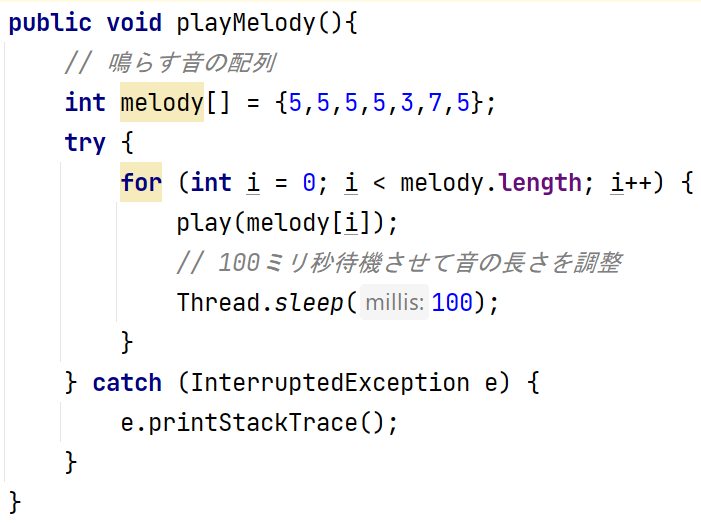
ボタンの変数定義と関連付け、リスナー登録を行います。





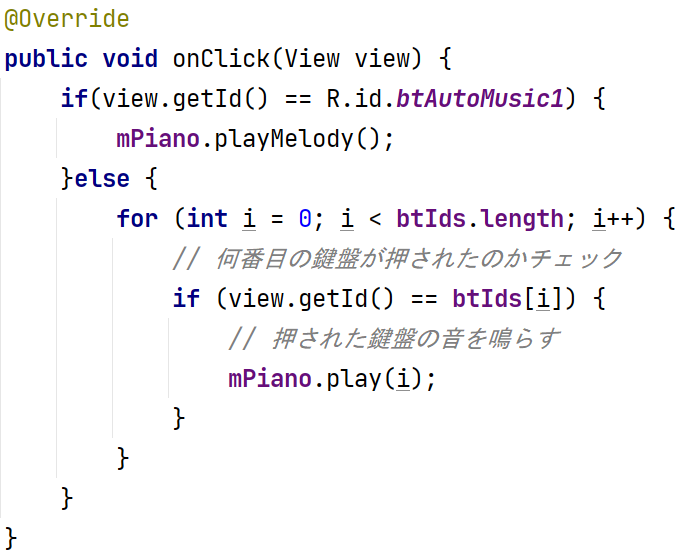
Pianoクラスにメロディとして音を鳴らすメソッドを定義

Piano.java



自動演奏ボタンの時と処理を分岐

MainActivity.java



メロディが鳴るか確認してみましょう！

