【実習７】サウンド

１　ソースコード

（１）テンプレート　Jump の①の箇所に　Jump070View　を入力してください。

（２）次のアプリケーションを新規作成してください。

Jump060View をコピー＆ペーストして、ファイル名を「Jump070View」に変更してください。

プロジェクト名：JumpPro????(年組席)　　アプリケーション名：Jump070View

```
/*    年    組    席  名前
 * Jump070View
 *     サウンド
 */
・・・
import java.util.Random;
import android.media.MediaPlayer;

public class Jump070View extends SurfaceView
                implements SurfaceHolder.Callback,
                           MediaPlayer.OnCompletionListener
{
    ・・・
    private int num;//ブロック数
    private MediaPlayer player;//プレーヤー

    public Jump070View(Context context)
    {
        ・・・
    }
    //サーフェイス生成時に呼ばれる
    public void surfaceCreated(SurfaceHolder holder)
    {
        ・・・
    }
    public void draw(Canvas canvas)
    {
        ・・・
    }
    //ボールの移動
    private void moveBall()
    {
        ・・・

        if(ballY>barY){playSound();init=S_GAMEOVER;}

        //バーとの衝突判定
        int r=hitLineRect(
                ballX-ballVX,ballY-ballVY,ballX,ballY,
                barX,barY-ballHEIGHT,barWIDTH,barHEIGHT);
        if (r>=0)
```

```
        {
            if(r==0) ballVY=-4-rand(2);//上
            if(r==1) ballVY=-ballVY;//下
            if(r==2 || r==3) ballVX=-ballVX;//左、右
            playSound();
        }
        //ブロックとの衝突判定
        for (int i=0;i<BLOCK.length;i++)
        {
            if (block[i]==0) continue;
            int r1=hitLineRect(
                ballX-ballVX,ballY-ballVY,ballX,ballY,(i%13)*blockWIDTH,
                                50+(i/13)*blockHEIGHT,blockWIDTH,blockHEIGHT);
            if (r1>=0)
            {
                if (r1==0 || r1==1) ballVY=-ballVY;//上、下
                if (r1==2 || r1==3) ballVX=-ballVX;//左、右
                block[i]=0;//ブロックの消去
                score+=10;
                playSound();
                break;
            }
        }
    }
      ・・・
    //数値→文字列
    private static String num2str(int num,int len)
    {
        String str=""+num;
        while(str.length()<len) str="0"+str;
        return str;
    }

    //メディアプレイヤーの生成と再生
    public void playSound()
    {
        try
        {
            if(ballY>barY){
                player=MediaPlayer.create(getContext(), R.raw.cancel5);
            }else{
                player=MediaPlayer.create(getContext(), R.raw.decide17);
            }
            player.seekTo(0);
            player.start();
            player.setOnCompletionListener(this);
        }catch(Exception e){
        }
    }
    //サウンドの停止
    public void stopSound()
    {
        try
        {
            player.stop();
```

```
            player.setOnCompletionListener(null);
            player.release();//サウンドリソースのメモリ解放
            player=null;
        }catch(Exception e){
        }
    }
    //サウンド再生終了時に呼ばれる
    public void onCompletion(MediaPlayer MediaPlayer){stopSound();}
}
```

２　サウンドファイルの準備

「res」フォルダに「raw」フォルダを生成し、その中に MP3、Midi、Ogg 形式のサウンドファイルを配置してください、

３　サウンドファイルの再生と停止

3.1 メディアプレイヤーの生成

```
player=MediaPlayer.create(getContext(), R.raw.cancel5);
player.setAudioStreamType(AudioManager.STREAM_MUSIC);
```

サウンドを再生するには、まずメディアプレイヤーを生成します。同時に使用できるトラックは、4 つまでです。

※SoundPool クラス

MediaPlayer と同じく、一般的なサウンド・フォーマットを再生できます。同時に使用できるトラックは、２５６までです。ループ再生も可能です。MediaPlayer とちがって、読み込み時に PCM データへデコードします。

そのため、再生キューの反応がいいです。デコードしたデータは、ネイティブ・ヒープに溜められてしまうため、データが大きくなると、Android 全体を重くしてしまいます。だから、BGM などの長いデータには向きません。

3.2　サウンドの再生

```
player.setLooping(true);
player.seekTo(0);
player.start();
```

①ループ再生をするには、player.setLooping(true); を指定してください。

②seekTo()メソッドで再生位置を先頭に戻します。

③start()メソッドで再生を開始します。

3.3　サウンド再生完了リスナーの指定

```
implements   MediaPlayer.OnCompletionListener
     ・
     ・
player.setOnCompletionListener(this);
     ・
     ・
//サウンド再生終了時に呼ばれる
public void onCompletion(MediaPlayer MediaPlayer){stopSound();}
```

サウンドの再生が完了した時、stopSound()メソッドを実行して後処理を行うように、サウンド再生完了リスナー「MediaPlayer.OnCompletionListener」を

implements します。サウンド再生完了リスナーを設定するには、
「setOnCompletionListener」メソッドを使います。

サウンド再生完了時には MediaPlayer.OnCompletionListener インタフェースの onCompletion メソッドが呼ばれます。

※MediaPlayer での再生は、アプリ終了後も継続されます。そのため、
surfaceDestroyed メソッドや surfaceChanged メソッドで再生停止するようにしています。そうしないと、アプリを切り替えられた時に、サウンドが再生され続けてしまいます。

3.4 サウンドの停止

```
player.stop();
player.setOnCompletionListener(null);
player.release();//サウンドリソースのメモリ解放
player=null;
```

①サウンドの停止を行うには、MediaPlayer クラスの stop()メソッドを使います。

②setOnCompletionListener メソッドに null を指定して、リスナーを解除します。

③release()メソッドでメモリを解放します。

④player=null によりオブジェクトに null を指定して、参照を解除しています。