

[ 1 ] MyKinjiofp6.java

/\*

-----  
無限級数による  $\pi$  の近似 6

Android 4.4 (Kit Kat)

Copyright (C) K.Niwa 2019.12.10  
-----

\*/

package jp.kiyo.wuena.mykinjiofp6;

import android.content.Context;  
import android.graphics.Canvas;  
import android.graphics.Color;  
import android.graphics.Paint;  
import android.graphics.Rect;  
import android.util.AttributeSet;  
import android.view.View;  
import android.content.res.Resources; //画像用  
import android.graphics.\*;  
import android.view.\*;

public class MyKinjiofp6 extends View {

private Bitmap bitmap1 = null;

int flag=0; //自動識別子  
int ct=0; //分子・分母の項の数  
int count; //ループカウンター  
double pai; //  $\pi$  の近似値  
double s; //  $\pi / 8$  を求める過程での無限級数

/\*\*  
//double rx; //x 座標の画面の幅に対する比率  
//double ry; //y 座標の画面の高さに対する比率  
//int x1,y1,x2,y2; //キャスト後の x y 座標に使用  
\*\*/

public MyKinjiofp6 (Context context) {  
super (context);

```

        init(context);
    }

    public MyKinjiofp6(Context context, AttributeSet attrs) {
        super(context,attrs);
        init(context);
    }

    public MyKinjiofp6(Context context, AttributeSet attrs,int defStyle) {
        super(context,attrs,defStyle);
        init(context);
    }

    private void init(Context context) {
        Resources res = context.getResources();          //画像用
        bitmap1 = BitmapFactory.decodeResource(res, R.drawable.gregory);          //画像用
    }

    @Override
    protected void onDraw(Canvas canvas) {
        // TODO 自動生成されたメソッド・スタブ

        float a=0;
        float b=0;

        super.onDraw(canvas);
        canvas.drawColor(Color.WHITE);
        Paint paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setAlpha(50);

        /*******
        //rx=super.getWidth()/480; //x座標の画面の幅に対する比率
        //ry=super.getHeight()/690; //y座標の画面の高さに対する比率
        //x1=(int) ((int) 10*rx);
        //y1=(int) ((int) 10*ry);
        //x2=(int) ((int) 470*rx);
        //y2=(int) ((int) 675*ry);
        canvas.drawRect((getWidth()/2-240)+10,(getHeight()/2-343)+10,(getWidth()/2-240)
+470,(getHeight()/2-343)+675, paint);

```

```

//*****

paint.setAlpha (10000);
paint.setColor (Color.BLUE);

for (int i=0;i<3;i++) {
    canvas.drawLine ((getWidth () /2-240) +10+i, (getHeight () /2-343) +10+i, (getWidth
() /2-240) +10+i, (getHeight () /2-343) +675-i, paint);
    canvas.drawLine ((getWidth () /2-240) +10+i, (getHeight () /2-343) +675-i, (getWidth
() /2-240) +470-i, (getHeight () /2-343) +675-i, paint);
    canvas.drawLine ((getWidth () /2-240) +470-i, (getHeight () /2-343) +675-i, (getWidth
() /2-240) +470-i, (getHeight () /2-343) +10+i, paint);
    canvas.drawLine ((getWidth () /2-240) +470-i, (getHeight () /2-343) +10+i, (getWidth
() /2-240) +10+i, (getHeight () /2-343) +10+i, paint);
}

if (MainActivity.ritsu != 0) {
    a=(float) (0.7*320/MainActivity.ritsu); //----- < 画像の拡大・縮小の横の倍率を指定する >
    b=(float) (0.7*320/MainActivity.ritsu); //----- < 画像の拡大・縮小の縦の倍率を指定する >
}
else {
    a=(float) 1.0;
    b=(float) 1.0;
}

Matrix Mat = new Matrix (); //----- < 画像を拡大・縮小する >

Mat.postScale (a, b); //-----
Bitmap bitmap2 = Bitmap.createBitmap ( //-----
    bitmap1,0,0, //-----
    bitmap1.getWidth (), //-----
    bitmap1.getHeight (), //-----
    Mat,true //-----
); //-----

if (bitmap2 != null) {
    canvas.drawBitmap (bitmap2, (getWidth () /2-240) +180-10, (getHeight () /2-343)
+130, paint);
}

```

```

}

paint.setTextSize(25.0f);

//x1=(int) ((int) 100*rx);
//y1=(int) ((int) 120*ry);
    canvas.drawText("【無限級数による  $\pi$  の近似 6】", (getWidth()/2-240)
+110-20-20+13, (getHeight()/2-343)+80, paint);

paint.setTextSize(25.0f);

//x1=(int) ((int) 110*rx);
//y1=(int) ((int) 150*ry);
    canvas.drawText(" ( $\pi$  の近似値を求める)", (getWidth()/2-240)+120, (getHeight()
/2-343)+110, paint);

paint.setColor(Color.BLUE);
paint.setTextSize(19.0f);

//x1=(int) ((int) 120*rx);
//y1=(int) ((int) 600*ry);
    canvas.drawText("Copyright (C) K.Niwa 2019.11.23", (getWidth()/2-240)+100,
(getHeight()/2-343)+600, paint);

//----- 計算部始まり -----

ct++;
s=s+(double) 1/((4*ct-3)*(4*ct-1));

pai=(double) 8*s;

//----- 計算部終わり -----

paint.setColor(Color.BLACK);
paint.setTextSize(23.0f);

//x1=(int) ((int) 40*rx);
//y1=(int) ((int) 250*ry);

```

```

        canvas.drawText("項数 = "+ct+" のとき" ,(getWidth()/2-240)+40, (getHeight()
/2-343)+310, paint);
        //canvas.drawText("x 率="+getWidth() ,x1, y1, paint);
        //canvas.drawText(" y 率= "+getHeight() ,x1, y1, paint);

        //x1=(int) ((int) 40*rx);
        //y1=(int) ((int) 300*ry);
        canvas.drawText("円周率  $\pi$ " , (getWidth()/2-240)+40, (getHeight()/2-343)+350,
paint);

        //x1=(int) ((int) 60*rx);
        //y1=(int) ((int) 340*ry);
        canvas.drawText("=  $8 \{1/(1 \cdot 3)+1/(5 \cdot 7)+1/(9 \cdot 11)+ \dots\}$ " , (getWidth()/2-240)+60,
(getHeight()/2-343)+390, paint);

        paint.setColor(Color.BLUE);

        //x1=(int) ((int) 60*rx);
        //y1=(int) ((int) 380*ry);
        canvas.drawText("="+pai ,(getWidth()/2-240)+60, (getHeight()/2-343)+430, paint);

        paint.setColor(Color.BLACK);
        paint.setTextSize(18.0f);

        //x1=(int) ((int) 50*rx);
        //y1=(int) ((int) 470*ry);
        canvas.drawText("※ 画面をタッチすると自動になります。" ,(getWidth()/2-240)
+50, (getHeight()/2-343)+470, paint);

        //x1=(int) ((int) 50*rx);
        //y1=(int) ((int) 500*ry);
        canvas.drawText("※ 画面をタッチすると自動が止まります。" ,(getWidth()/2-240)
+50, (getHeight()/2-343)+500, paint);

        //x1=(int) ((int) 50*rx);
        //y1=(int) ((int) 530*ry);
        canvas.drawText("※ 更に画面をタッチすると初期化されます。" ,(getWidth()
/2-240)+50, (getHeight()/2-343)+530, paint);

        //x1=(int) ((int) 50*rx);

```

```

        //y1=(int) ((int) 560*ry);
        canvas.drawText("※ 画面が暗くなったらタイトルバーをタッチ!",(getWidth()
/2-240)+50, (getHeight()/2-343)+560, paint);

        if (flag==1) {                //flag=1 で自動になる      flag=2 で自動が止まる
flag=0 で初期化する
            invalidate(); //表示を更新する
        }

    }//protected void onDraw(Canvas canvas)

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        flag++;
        flag = flag % 3;
        if (flag==0) {
            ct=0;                //項数
            s=0;                //πを求める過程で使用
        }

        invalidate(); //表示を更新する
        return false;

    }//public boolean onTouchEvent(MotionEvent event)

} //public class MyPai6 extends View

```

[ 2 ] activity\_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

```
<jp.kiyo.wuena.mykinjiofp6.MyKinjiofp6
    android:id="@+id/myfview1"
    android:layout_height="match_parent"
    android:layout_width="match_parent"/>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

[ 3 ] MainActivity.java

```
/*
```

```
-----
    無限級数による  $\pi$  の近似 6
    Android 4.4 (Kit Kat)
    Copyright (C) K.Niwa 2019.12.10
    -----
```

```
*/
```

```
package jp.kiyo.wuena.mykinjiofp6;
```

```
import androidx.appcompat.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

```
import android.util.DisplayMetrics;    //<画像の拡大・縮小に必要なライブラリ>
```

```
import android.app.Activity;
```

```
import android.view.Menu;
```

```
public class MainActivity extends AppCompatActivity {
```

```
    static int ritsu;
```

```
    @Override
```

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

    DisplayMetrics metrics = new DisplayMetrics(); // <端末の情報を取得する>
    getWindowManager().getDefaultDisplay().getMetrics(metrics);
    StringBuilder buffer = new StringBuilder();
    buffer.append("densityDpi (ドット数 / インチ) : " + String.valueOf
(metrics.densityDpi) + "\n");
    ritsu=metrics.densityDpi;
}
}
```