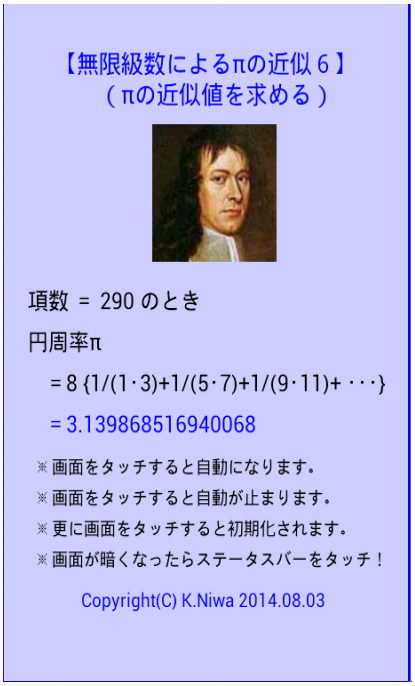
```
//----------------------------------------
//
//          無限級数によるπの近似6
//            （πの近似値を求める）
//                  Ver10
//      Copyright(C) K.Niwa 2014.08.03
//
//----------------------------------------
```

【エミュレータ画面例】



【スマートフォン画面例】
（Android 4.0）

［アプリの概要］

次の近似式を用いて、円周率πの近似値を求めます。

$$\pi = 8\left(\frac{1}{1\cdot 3}+\frac{1}{5\cdot 7}+\frac{1}{9\cdot 11}+\frac{1}{13\cdot 15}+\cdots\right)$$

項の数をどんどん多くしていったとき、円周率πの近似値が求まる様子を観察してみましょう。

［１］Kinjiofp6.java

```
package jp.seitoku.kinjiofp6;

import android.content.Context;
import android.content.res.Resources;
import android.graphics.*;        //*に修正
import android.util.AttributeSet;
import android.view.*;            //*に修正

public class Kinjiofp6 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;                       //πの近似値
        double s;                         //π／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ｘｙ座標に使用
        //*****************************************

        public Kinjiofp6(Context context, AttributeSet attrs, int defStyle) {
                super(context, attrs, defStyle);
```

```
		init(context);
		// TODO 自動生成されたコンストラクター・スタブ
	}

	public Kinjiofp6(Context context, AttributeSet attrs) {
		super(context, attrs);
		init(context);
		// TODO 自動生成されたコンストラクター・スタブ
	}

	public Kinjiofp6(Context context) {
		super(context);
		init(context);
		// TODO 自動生成されたコンストラクター・スタブ
	}

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

	@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)+x1,(getHeight()/2-343)+y1,(getWidth()/2-240)+x2,(getHe
ight()/2-343)+y2, 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 (Kinjiofp6Activity.ritsu != 0) {
			a=(float) (0.9*320/Kinjiofp6Activity.ritsu);	//----------------------------
 ＜画像の拡大・縮小の横の倍率を指定する＞
			b=(float) (0.9*320/Kinjiofp6Activity.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,(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πの近似６】", (getWidth()/2-240)+110-20-20, (getHeight()
/2-343)+80, paint);

                paint.setTextSize(25.0f);

                //x1=(int) ((int)110*rx);
                //y1=(int) ((int)150*ry);
                canvas.drawText("（π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 2014.08.03", (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)+x1, (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π", (getWidth()/2-240)+x1, (getHeight()/2-343)+350, paint);

                x1=(int) ((int)60*rx);
                //y1=(int) ((int)340*ry);
                canvas.drawText("＝8 {1/(1･3)+1/(5･7)+1/(9･11)+ ･･･}", (getWidth()/2-240)+x1, (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)+x1, (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)+x1, (getHeigh
t()/2-343)+y1, 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)+x1, (getHei
ght()/2-343)+y1, 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)+x1, (getH
eight()/2-343)+y1, paint);

                x1=(int) ((int)50*rx);
                y1=(int) ((int)560*ry);
                canvas.drawText("※ 画面が暗くなったらステータスバーをタッチ！" ,(getWidth()/2-240)+x1,
(getHeight()/2-343)+y1, 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］main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:weightSum="1" >

        <jp.seitoku.kinjiofp6.Kinjiofp6
                android:layout_height="match_parent"
```

```
                android:layout_width="match_parent"
                android:id="@+id/myview1">
        </jp.seitoku.kinjiofp6.Kinjiofp6>
</LinearLayout>
```

[３] Kinjiofp6Activity.java

```
package jp.seitoku.kinjiofp6;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

import android.util.DisplayMetrics;       //----------------------------〈画像の拡大・縮小に必要なライブラリ〉

public class Kinjiofp6Activity extends Activity {

    static int ritsu;

        @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.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;// ----------------------------------------------------これで値が取り出せた!
        }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```