［１］MyTwokansu2.java

```
/*
-----------------------------------------------------------------------
    ２次関数のグラフの平行移動（上に凸）
            Android 4.4（Kit Kat）
       Copyright(C) K.Niwa 2019.12.11
-----------------------------------------------------------------------
*/

package jp.kiyo.wuena.mytwokansu2;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.graphics.RectF;
import android.util.AttributeSet;
import android.view.MotionEvent;
import android.view.View;

public class MyTwokansu2 extends View {

    int flag=0;     //グラフの移動(1)、グラフの停止(2)、グラフの初期化(0)初期化　識別子
    double x,y;    //グラフ描写に利用
    int px,py,oldpx,oldpy;   //グラフ描写に利用
    int fg;          //グラフ描写に利用
    double a=0,b=0;          //グラフ描写に利用

    public MyTwokansu2(Context context) {
        super(context);
    }

    public MyTwokansu2(Context context, AttributeSet attrs) {
        super(context,attrs);
    }

    public MyTwokansu2(Context context, AttributeSet attrs,int defStyle) {
        super(context,attrs,defStyle);
```

```
    }

    //onDraw メソッド----------------------------------------------------------------------------
    @Override
    protected void onDraw(Canvas canvas) {

        super.onDraw(canvas);
        canvas.drawColor(Color.WHITE);
        Paint paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setAlpha(50);
        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<2;i++) {        //額縁を付ける
            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);
        }

        paint.setColor(Color.BLACK);          //実験枠の描画
          canvas.drawRect((getWidth()/2-240)+90,  (getHeight()/2-343)+100,(getWidth()
/2-240)+390,(getHeight()/2-343)+400,paint);
        paint.setColor(Color.WHITE);
          canvas.drawRect((getWidth()/2-240)+91,  (getHeight()/2-343)+101,(getWidth()
/2-240)+389,(getHeight()/2-343)+399,paint);

        paint.setColor(Color.BLACK);          //座標軸の描画
          canvas.drawLine((getWidth()/2-240)+100,  (getHeight()/2-343)+250,(getWidth()
/2-240)+380,(getHeight()/2-343)+250,paint);
          canvas.drawLine((getWidth()/2-240)+380,  (getHeight()/2-343)+250,(getWidth()
/2-240)+380-5,(getHeight()/2-343)+250-5,paint);
          canvas.drawLine((getWidth()/2-240)+380,  (getHeight()/2-343)+250,(getWidth()
```

```
/2-240)+380-5,(getHeight()/2-343)+250+5,paint);
            canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+240,(getHeight()/2-343)+390,paint);
            canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+240+5,(getHeight()/2-343)+110+5,paint);
            canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+240-5,(getHeight()/2-343)+110+5,paint);

        for（int xx=110;xx<380;xx=xx+10）｛ //ｘ軸メモリの描写
            canvas.drawLine((getWidth()/2-240)+xx,  (getHeight()/2-343)+250-3,(getWidth
()/2-240)+xx,(getHeight()/2-343)+250+3,paint);
        }
        for（int yy=120;yy<390;yy=yy+10）｛ //ｙ軸メモリの描写
            canvas.drawLine((getWidth()/2-240)+240-3,  (getHeight()/2-343)+yy,(getWidth
()/2-240)+240+3,(getHeight()/2-343)+yy,paint);
        }

        //ｘ軸メモリの描写
        canvas.drawText("5",（getWidth()/2-240)+290-3,（getHeight()/2-343)+260+5, paint);
        canvas.drawText("10",（getWidth()/2-240)+340-6,（getHeight()/2-343)+260+5, paint)
;
        canvas.drawText("ｘ",（getWidth()/2-240)+375,（getHeight()/2-343)+260+5, paint);
        canvas.drawText("O",（getWidth()/2-240)+230-3,（getHeight()/2-343)+260+5, paint);
        canvas.drawText("-5",（getWidth()/2-240)+190-3,（getHeight()/2-343)+260+5, paint)
;
          canvas.drawText("-10",  (getWidth()/2-240)+140-6,  (getHeight()/2-343)+260+5,
paint);

        //ｙ軸メモリの描写
        canvas.drawText("5",（getWidth()/2-240)+220+5,（getHeight()/2-343)+200+5, paint);
        canvas.drawText("10",（getWidth()/2-240)+220,（getHeight()/2-343)+150+5, paint);
        canvas.drawText("ｙ",（getWidth()/2-240)+220,（getHeight()/2-343)+110+5, paint);
        canvas.drawText("-5",（getWidth()/2-240)+220,（getHeight()/2-343)+300+5, paint);
          canvas.drawText("-10",  (getWidth()/2-240)+220-5,  (getHeight()/2-343)+350+5,
paint);

        //y=-2x^2 のグラフの描画
        paint.setColor(Color.BLUE);
        canvas.drawText("ｙ＝-2 ｘ  ",（getWidth()/2-240)+150,（getHeight()/2-343)+370,
paint);
```

```
        canvas.drawText("2", (getWidth()/2-240)+180+16-3+1, (getHeight()/2-343)+350+15,
paint);
        fg=0;
        for (x=-2.6+a;x<=2.6+a;x=x+0.1) {
            y=-2*(x-a)*(x-a)+b;
            px=(int)(240+10*x);
            py=(int)(250-10*y);
            if (fg==0) {
                canvas.drawLine((int)(getWidth()/2-240)+px, (int)(getHeight()/2-343)+py,
(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
            }
            else {
                canvas.drawLine((int)(getWidth()/2-240)+oldpx, (int)(getHeight()/2-343)
+oldpy,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
            }
            oldpx=px;oldpy=py;
            fg++;
        }

        //y=-2(x-6)^2+10 のグラフの描画
        paint.setColor(Color.BLUE);
        canvas.drawText("y =-2( x -6)  +10", (getWidth()/2-240)+250-13, (getHeight()
/2-343)+140, paint);
        canvas.drawText("2", (getWidth()/2-240)+320-20+1, (getHeight()/2-343)+140-5,
paint);
        fg=0;
        for (x=-2.6+6.0;x<=2.6+6.0;x=x+0.1) {
            y=-2*(x-6)*(x-6)+10;
            px=(int)(240+10*x);
            py=(int)(250-10*y);
            if (fg==0) {
                canvas.drawLine((int)(getWidth()/2-240)+px, (int)(getHeight()/2-343)+py,
(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
            }
            else {
                canvas.drawLine((int)(getWidth()/2-240)+oldpx, (int)(getHeight()/2-343)
+oldpy,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
            }
            oldpx=px;oldpy=py;
            fg++;
```

```
        }

        paint.setColor(Color.BLACK);          //実験枠の描画
          canvas.drawLine((getWidth()/2-240)+90,  (getHeight()/2-343)+100,(getWidth()
/2-240)+90,(getHeight()/2-343)+400,paint);
          canvas.drawLine((getWidth()/2-240)+90,  (getHeight()/2-343)+100,(getWidth()
/2-240)+390,(getHeight()/2-343)+100,paint);

        paint.setColor(Color.BLUE);//表題の表示
        paint.setTextSize(25.0f);
        canvas.drawText("",(getWidth()/2-240)+60,(getHeight()/2-343)+65, paint);
        canvas.drawText("【y ＝－ 2(x － 6) ＋ 10 のｸﾞﾗﾌの平行移動】",(getWidth()/2-240)
+30-20,(getHeight()/2-343)+65, paint);
         canvas.drawText("2",  (getWidth()/2-240)+170+5+3-3+1,  (getHeight()/2-343)+50,
paint);

        paint.setColor(Color.BLUE);//目標の提示
        paint.setTextSize(19.0f);
        canvas.drawText("ｙ＝－２(ｘ－６)^2 ＋１０のｸﾞﾗﾌは、 ｙ＝－２ｘ^2",(getWidth
()/2-240)+30,(getHeight()/2-343)+440, paint);
        //canvas.drawText("2",  (getWidth()/2-240)+140+53-5,  (getHeight()/2-343)+440-10,
paint);
        //canvas.drawText("2",(getWidth()/2-240)+300+143-18,(getHeight()/2-343)+440-10,
paint);
        canvas.drawText("のｸﾞﾗﾌをｘ軸方向に＋ 6 だけ、ｙ軸方向に＋ 10 だけ",(getWidth
()/2-240)+30,(getHeight()/2-343)+465, paint);
        canvas.drawText("平行移動したものであることを観察してみよう。",(getWidth()
/2-240)+30,(getHeight()/2-343)+490, paint);

        paint.setColor(Color.BLACK);          //説明の表示
        paint.setTextSize(19.0f);
        canvas.drawText("※ 画面ﾀｯﾁで、ｙ＝－２ｘ^2 のｸﾞﾗﾌが動きます。",(getWidth()
/2-240)+30,(getHeight()/2-343)+550-5, paint);
        //canvas.drawText("2",  (getWidth()/2-240)+290-40-17,  (getHeight()/2-343)+545-5,
paint);
        canvas.drawText("※ もう一度タッチすると止まります。",(getWidth()/2-240)+30,
(getHeight()/2-343)+575-5, paint);
        canvas.drawText("※ 更にタッチすると初期化されます。",(getWidth()/2-240)+30,
(getHeight()/2-343)+600-5, paint);
        canvas.drawText("※ 画面が暗くなったらタイトルバーをタッチ！",  (getWidth()
```

```
/2-240）+30,（getHeight（）/2-343）+625-5, paint）;
        paint.setColor（Color.BLUE）;
        paint.setTextSize（19.0f）;
            canvas.drawText（ "Copyright（C） K.Niwa 2019.11", （getWidth（）/2-240）+110,
（getHeight（）/2-343）+650, paint）;                    //作者・作成年月の表示

        if（flag==1）｛     //識別子が１のとき、グラフが平行移動する
            if（a<=5.9）｛
                a=a+0.1;
            }
            else｛
                if（b<=9.9）｛
                    b=b+0.1;
                }
            }

            invalidate（）; //再描画する（clear & goto onDraw）　そして、この行へ戻っ
てくる。
            //invalidate（）は、onDraw メソッドの中にも記述できる。
            //invalidate（）は、繰り返し処理に利用できる。
            //もちろん、onTouchEvent ソッドの中にも記述できる。
        }
        else if（flag==2）｛//識別子が２のとき、グラフが停止する
            //何も変化を加えない
        }
        else if（flag==0）｛//識別子が０のとき、グラフを初期化する（元の位置に戻す）
            a=0; //初期化する
            b=0; //初期化する
            invalidate（）; //再描画する（clear & goto onDraw）　そして、この行へ戻っ
てくる。
        }

    }//protected void onDraw（Canvas canvas）

        //画面にタッチしたときのイベント処理
-------------------------------------------------------------------------------
    @Override
    public boolean onTouchEvent（MotionEvent event）｛

        flag=flag+1;        //flag に１を加える
```

```
            flag=flag % 3;       //flag に１、２、０を代入する

            invalidate();                                    //再描画する（clear & goto onDraw）
            return false;

        }
}
```

［２］activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
< a n d r o i d x . c o n s t r a i n t l a y o u t . w i d g e t . C o n s t r a i n t L a y o u t
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.mytwokansu2.MyTwokansu2
        android:id="@+id/myfview1"
        android:layout_height="match_parent"
        android:layout_width="match_parent"/>

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

［３］MainActivity.java

```
/*
-----------------------------------------------------------------
    ２次関数のグラフの平行移動（上に凸）
            Android 4.4（Kit Kat）
      Copyright(C) K.Niwa 2019.12.11
-----------------------------------------------------------------
*/

package jp.kiyo.wuena.mytwokansu2;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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