

```
[1]MyTwokansu1Eng. java
```

```
/*
```

```
-----  
2 次関数のグラフの平行移動（下に凸） 英語版
```

```
Android 4.1 (Jelly Bean)
```

```
Copyright (C) K. Niwa 2021. 9. 19  
-----
```

```
*/
```

```
package jp.kiyo.wuena.mytwokansu1eng;
```

```
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 MyTwokansu1Eng 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 MyTwokansu1Eng(Context context) {
```

```
        super(context);
```

```
    }
```

```
    public MyTwokansu1Eng(Context context, AttributeSet attrs) {
```

```
        super(context, attrs);
```

```
    }
```

```

public MyTwokansu1Eng(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()/2-360)+20, (getHeight()/2-600)+10, (getWidth()/2-
360)+700, (getHeight()/2-600)+1190, paint);
    paint.setAlpha(10000);
    paint.setColor(Color.BLUE);

    for (int i=0;i<2;i++) { //額縁を付ける
        canvas.drawLine((getWidth()/2-360)+20+i, (getHeight()/2-600)+10+i, (getWidth()/2-
360)+20+i, (getHeight()/2-600)+1190-i, paint);
        canvas.drawLine((getWidth()/2-360)+20+i, (getHeight()/2-600)+1190-i, (getWidth()/2-
360)+700-i, (getHeight()/2-600)+1190-i, paint);
        canvas.drawLine((getWidth()/2-360)+700-i, (getHeight()/2-600)+1190-i, (getWidth()/2-
360)+700-i, (getHeight()/2-600)+10+i, paint);
        canvas.drawLine((getWidth()/2-360)+700-i, (getHeight()/2-600)+10+i, (getWidth()/2-
360)+20+i, (getHeight()/2-600)+10+i, paint);
    }

    paint.setColor(Color.BLACK); //実験枠の描画
    canvas.drawRect((getWidth()/2-360)+70, (getHeight()/2-600)+100, (getWidth()/2-
360)+650, (getHeight()/2-600)+680, paint);
    paint.setColor(Color.WHITE);
    canvas.drawRect((getWidth()/2-360)+71, (getHeight()/2-600)+101, (getWidth()/2-

```

```
360)+649, (getHeight()/2-600)+679, paint);
```

```
paint.setColor(Color.BLACK); //座標軸の描画
```

```
canvas.drawLine((getWidth()/2-360)+80, (getHeight()/2-600)+390, (getWidth()/2-360)+640, (getHeight()/2-600)+390, paint);
```

```
canvas.drawLine((getWidth()/2-360)+640, (getHeight()/2-600)+390, (getWidth()/2-360)+640-5, (getHeight()/2-600)+390-5, paint);
```

```
canvas.drawLine((getWidth()/2-360)+640, (getHeight()/2-600)+390, (getWidth()/2-360)+640-5, (getHeight()/2-600)+390+5, paint);
```

```
canvas.drawLine((getWidth()/2-360)+360, (getHeight()/2-600)+110, (getWidth()/2-360)+360, (getHeight()/2-600)+670, paint);
```

```
canvas.drawLine((getWidth()/2-360)+360, (getHeight()/2-600)+110, (getWidth()/2-360)+360+5, (getHeight()/2-600)+110+5, paint);
```

```
canvas.drawLine((getWidth()/2-360)+360, (getHeight()/2-600)+110, (getWidth()/2-360)+360-5, (getHeight()/2-600)+110+5, paint);
```

```
for (int xx=100;xx<640;xx=xx+20) { //x軸メモリの描写
```

```
canvas.drawLine((getWidth()/2-360)+xx, (getHeight()/2-600)+390-3, (getWidth()/2-360)+xx, (getHeight()/2-600)+390+3, paint);
```

```
}
```

```
for (int yy=130;yy<670;yy=yy+20) { //y軸メモリの描写
```

```
canvas.drawLine((getWidth()/2-360)+360-3, (getHeight()/2-600)+yy, (getWidth()/2-360)+360+3, (getHeight()/2-600)+yy, paint);
```

```
}
```

```
paint.setTextSize(23.0f);
```

```
//x軸メモリの描写
```

```
canvas.drawText("5", (getWidth()/2-360)+460-3, (getHeight()/2-600)+400+5+10, paint);
```

```
canvas.drawText("10", (getWidth()/2-360)+560-6, (getHeight()/2-600)+400+5+10, paint);
```

```
canvas.drawText("x", (getWidth()/2-360)+630-10, (getHeight()/2-600)+400+5+10, paint);
```

```
canvas.drawText("0", (getWidth()/2-360)+350-3, (getHeight()/2-600)+400+5+10, paint);
```

```
canvas.drawText("-5", (getWidth()/2-360)+260-3, (getHeight()/2-600)+400+5+10, paint);
```

```
canvas.drawText("-10", (getWidth()/2-360)+160-6, (getHeight()/2-600)+400+5+10, paint);
```

```
//y軸メモリの描写
```

```
canvas.drawText("5", (getWidth()/2-360)+220+5+120-10, (getHeight()/2-600)+290+5,
```

```

paint);
    canvas.drawText("10", (getWidth()/2-360)+220+120-10, (getHeight()/2-600)+190+5,
paint);
    canvas.drawText("y", (getWidth()/2-360)+220+120-10, (getHeight()/2-600)+130, paint);
    canvas.drawText("-5", (getWidth()/2-360)+220+120-10, (getHeight()/2-600)+490+5,
paint);
    canvas.drawText("-10", (getWidth()/2-360)+220-5+120-10, (getHeight()/2-600)+590+5,
paint);

    //y=x^2 のグラフの描画
    paint.setTextSize(25.0f);
    paint.setColor(Color.BLUE);
    canvas.drawText("y = x^2", (getWidth()/2-360)+300-20+120+50-10, (getHeight()/2-
600)+120+50, paint);
    canvas.drawText("2", (getWidth() / 2 - 360) + 315 - 3 + 1+120+50+35-10, (getHeight() /
2 - 600) + 120 - 5+50-10, paint);

    fg=0;

    for (x=-3.6+a;x<=3.7+a;x=x+0.1) {
        y=(x-a)*(x-a)+b;
        px=(int) (360+20*x);
        py=(int) (390-20*y);
        if (fg==0) {
            canvas.drawLine((int) (getWidth()/2-360)+px, (int) (getHeight()/2-
600)+py, (int) (getWidth()/2-360)+px, (int) (getHeight()/2-600)+py, paint);
        }
        else {
            canvas.drawLine((int) (getWidth()/2-360)+oldpx, (int) (getHeight()/2-
600)+oldpy, (int) (getWidth()/2-360)+px, (int) (getHeight()/2-600)+py, paint);
        }
        oldpx=px;oldpy=py;
        fg++;
    }
    canvas.drawText("y = (x+7)^2 - 9", (getWidth()/2-360)+100, (getHeight()/2-
600)+600+20, paint);

```

```
canvas.drawText("2", (getWidth() / 2 - 360) + 160 + 60, (getHeight() / 2 - 600) + 600 - 5 + 20 - 10, paint);
```

```
fg=0;
for (x=-3.6-7;x<=3.7-7;x=x+0.1) {
    y=(x+7)*(x+7)-9;
    px=(int) (360+20*x);
    py=(int) (390-20*y);
    if (fg==0) {

        //y=(x+7)^2-9 のグラフの描画
        paint.setTextSize(25.0f);
        paint.setColor(Color.BLUE);
        canvas.drawLine((int) (getWidth()/2-360)+px, (int) (getHeight()/2-600)+py, (int) (getWidth()/2-360)+px, (int) (getHeight()/2-600)+py, paint);
    }
    else {
        canvas.drawLine((int) (getWidth()/2-360)+oldpx, (int) (getHeight()/2-600)+oldpy, (int) (getWidth()/2-360)+px, (int) (getHeight()/2-600)+py, paint);
    }
    oldpx=px;oldpy=py;
    fg++;
}
```

```
//paint.setColor(Color.BLACK); //実験枠の描画
//canvas.drawLine((getWidth()/2-360)+90+120, (getHeight()/2-600)+100, (getWidth()/2-360)+90+120, (getHeight()/2-600)+400, paint);
//canvas.drawLine((getWidth()/2-360)+90+120, (getHeight()/2-600)+100, (getWidth()/2-360)+390+120, (getHeight()/2-600)+100, paint);
```

```
paint.setColor(Color.BLUE); //表題の表示
paint.setTextSize(30.0f);
canvas.drawText("", (getWidth()/2-360)+60, (getHeight()/2-600)+65, paint);
canvas.drawText("【Parallel Translation", (getWidth()/2-360)+10+50+50-20, (getHeight()/2-600)+55, paint);
canvas.drawText("of Graph of Quadratic Function", (getWidth()/2-
```

```

360)+10+100+50+20+20, (getHeight()/2-600)+85, paint);
    //canvas.drawText("2", (getWidth() / 2 - 360) + 264+50, (getHeight() / 2 - 600) + 50,
    paint);

    paint.setColor(Color.BLACK);    //目標の提示
    paint.setTextSize(30.0f);
    canvas.drawText("■Let's observe that the graph of  $y=(x+7)^2-9$ ", (getWidth()/2-
360)+50, (getHeight()/2-600)+740+20, paint);
    canvas.drawText(" is a translation of the graph of  $y=x^2$  by  $-7$ ", (getWidth()/2-
360)+50, (getHeight()/2-600)+770+20, paint);
    canvas.drawText(" in the x-axis direction and  $-9$  in the y-axis", (getWidth()/2-
360)+50, (getHeight()/2-600)+800+20, paint);
    canvas.drawText(" direction.", (getWidth()/2-360)+50, (getHeight()/2-600)+830+20,
    paint);

    paint.setColor(Color.BLACK);    //説明の表示
    paint.setTextSize(30.0f);
    canvas.drawText("Touch the screen to move the graph of  $y=x^2$ .", (getWidth()/2-360)+50,
    (getHeight()/2-600)+950, paint);
    canvas.drawText("Touch the screen again to stop the graph.", (getWidth()/2-360)+50,
    (getHeight()/2-600)+990, paint);
    canvas.drawText("If you touch it further, it will be initialized.", (getWidth()/2-
360)+50, (getHeight()/2-600)+1030, paint);
    canvas.drawText("When the screen goes dark, touch the title bar !", (getWidth()/2-
360)+50, (getHeight()/2-600)+1070, paint);
    paint.setColor(Color.BLUE);
    paint.setTextSize(30.0f);
    canvas.drawText("Copyright(C) Sohun 2021. 9. 19", (getWidth()/2-360)+150+15,
    (getHeight()/2-600)+1130, 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>=-6.9) {
            a=a-0.1;
        }
        else {
            if (b>=-8.9) {

```

```

        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"?>
<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.mytwokansu1eng.MyTwokansu1Eng

        android:id="@+id/myview1"
        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（下に凸） 英語版

Android 4.1 (Jelly Bean)

Copyright (C) K. Niwa 2021. 9. 19

*/

package jp.kiyo.wuena.mytwokansu1eng;

```
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);
```

```
    }
```

```
}
```