LAB 1

DRAWLINESAMPLE

April 11, 2020

Name:Chenyu Yang
Student ID: 517030910386
Class: F1703015

Contents

1 INTRODUCTION

1.1 Requirement

In this project, we will learn the development of Android applications. We firstly need to show "Hello World!" in the User Interface. Then we will implement a drawing application in which we can draw lines in the screen.

1.2 Environment

1. OpenJDK 64-bit;
2. Android 9.0 (pie);
3. Android Studio 3.3.2;
4. MI 6; (for test)

2 RELATED KNOWLEDGE

2.1 Listener

istener is one of the most common way to get movements applied to a button. Compared with OnClick function, it has higher priority and can handle different actions. A listener is an object that is specially used to monitor and process the events or state changes of other objects. When events happens, corresponding actions are taken immediately.

2.2 Canvas

The Canvas class holds the "draw" calls. To draw something, it needs 4 basic components: a Bitmap to hold the pixels, a canvas to hold the draw calls (written into the bitmap), a drawing primitive (eg: Rect, Path, Text, Bitmap) and a paint to describe the color and style for the drawing.

3 PROCEDURE

3.1 Print 'Hello World!'

The first task is to print 'Hello World!' in the screen. This task is rather easy and without intraction with users. So we can accomplish it just in the *.xml* files, which takes charge of the UI design. We create a TextView in the center of the screen with the string we need to show. (Also, we can put the string in *values/strings.xml*.)

```
<TextView
  android:id="@+id/hello"
  android:layout_width="241dp"
  android:layout_height="58dp"
  android:layout_marginTop="8dp"
  android:layout_marginBottom="8dp"
  android:gravity="center"
  android:text="Hello World!"
  android:textAllCaps="true"
  android:textSize="24sp"
  android:textStyle="bold"
  app:layout_constraintBottom_toBottomOf="parent"
  app:layout_constraintEnd_toEndOf="parent"
  app:layout_constraintHorizontal_bias="0.5"
  app:layout_constraintStart_toStartOf="parent"
  app:layout_constraintTop_toTopOf="parent" />
```

activity_main.xml

Considering the *.xml* file has realized the function we need, the *Activity_main.java* only works to create the page based on *Activity_main.xml*. And it includes an OnClick function which starts the second activity.

```
public class MainActivity extends AppCompatActivity {

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

  //Jump to the next activity
  public void ClickToStart(View view) {
    Intent intent = new Intent(this, DrawLineSample.class);
    startActivity(intent);
  };
};
```

Activity_main.java

3.2 Draw A Line

The second task is more challenging, which requires us to design a drawing application. This task can be seperated into two parts. The first is to monitor the user's touch action while the second is drawing in the screen.

For the former, we can use an OnTouchListener to get the user's action and touch location reletive to the screen. When the user touch the screen, it draws a point at current position, while it draw a line from the previous position to the current when the screen is dragged. Every time the action has been handled, it will record the new location for the next action.

The function returns *True*, which means this action has been handled and will not be passed to ather handlers.

For drawing, we initialize the background bitmap, paint and the canvas in *oncreate()* function. Every time we draw on the background bitmap and pass it to the ImageView. Of course we can reset the background and style of paint.(details are in $4.2)

```
@SuppressLint("ClickableViewAccessibility")
public class DrawLineSample extends AppCompatActivity {
  Intent intent = getIntent(); //get from the old activity

  private ImageView scr;
  private Canvas canvas;
  private Paint p;
  private Bitmap bitmap;
  private DisplayMetrics dm;
  private float[] prev_pos;

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

    DisplayMetrics dm = new DisplayMetrics();
    getWindowManager().getDefaultDisplay().getMetrics(dm); // get the screen size
    prev_pos = new float[2];

    p = new Paint();
    p.setColor(Color.BLACK);              //set the color
    p.setAntiAlias(true);                //set anti-alias, always true
    p.setStrokeCap(Paint.Cap.ROUND);     //set the type of linr
    p.setStrokeWidth(8);                //set the width of line

    canvas = new Canvas();
    bitmap = Bitmap.createBitmap(screen_size[0], screen_size[1], Bitmap.Config.
        ARGB_8888);
    canvas.setBitmap(bitmap);

    scr = findViewById(R.id.draw_layer);
    scr.setOnTouchListener(new View.OnTouchListener() {
      @Override
      public boolean onTouch(View v, MotionEvent event) {
        float x = event.getX();   //record the current location
        float y = event.getY();
        if (event.getAction() == MotionEvent.ACTION_MOVE) {
          canvas.drawLine(x,y,prev_pos[0],prev_pos[1],p);
        }
        if (event.getAction() == MotionEvent.ACTION_DOWN) {
          canvas.drawPoint(x, y, p);
        }
        if (event.getAction() == MotionEvent.ACTION_UP) { }
        //Update the old location and the image;
        prev_pos[0] = x;
```

```
            prev_pos[1] = y;
            scr.setImageBitmap(bitmap);
            return true;
        }
    });
  }
// ......
// More fuctions
// ......
}
```

activity_draw.java

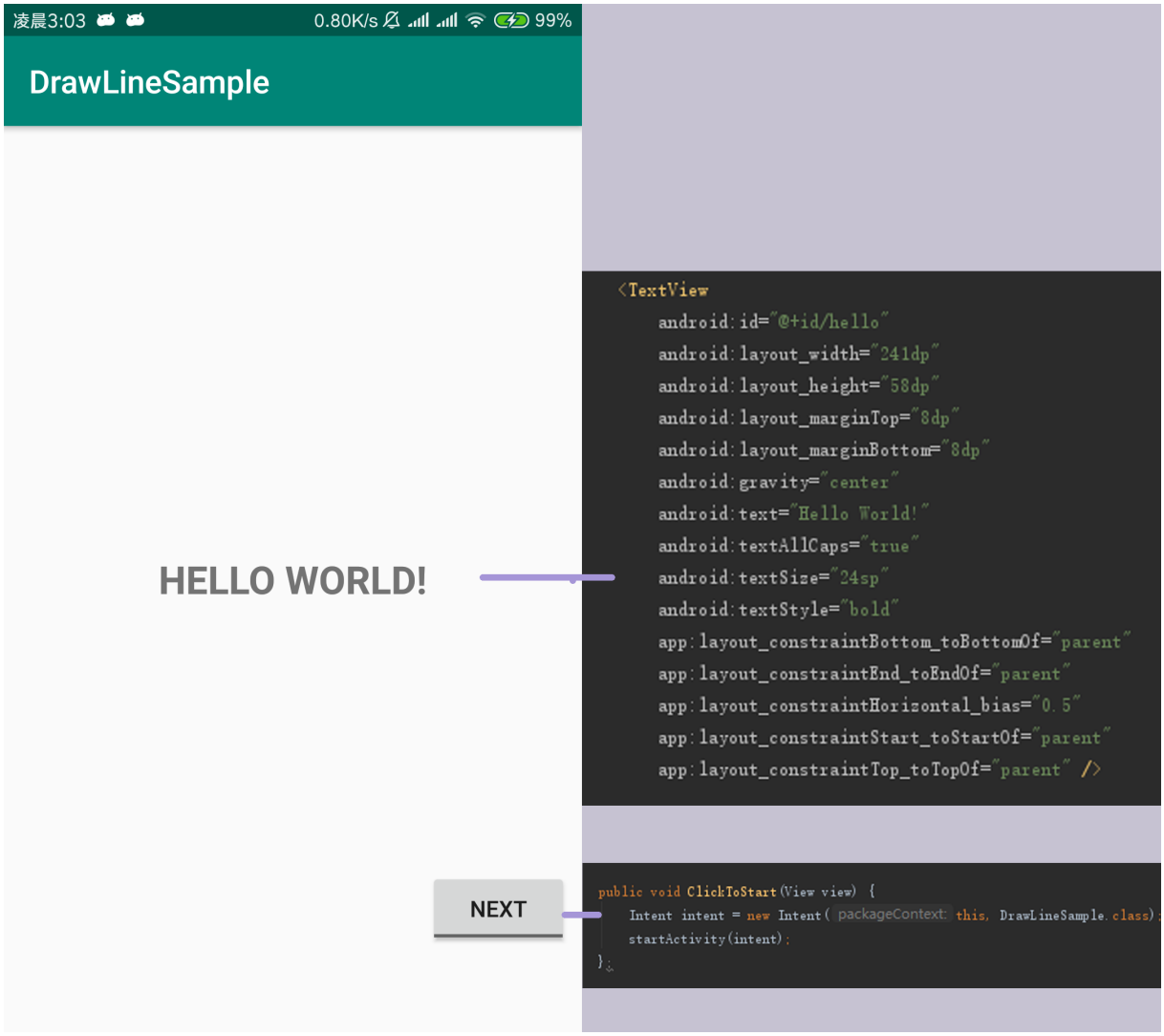
```
    <ImageView
  android:id="@+id/draw_layer"
  android:layout_width="0dp"
  android:layout_height="0dp"
  app:layout_constraintBottom_toBottomOf="parent"
  app:layout_constraintEnd_toEndOf="parent"
  app:layout_constraintHorizontal_bias="1.0"
  app:layout_constraintStart_toStartOf="parent"
  app:layout_constraintTop_toTopOf="parent"
  app:layout_constraintVertical_bias="0.0" />

  <Button // clear screen
  ... />
  <ImageButton // change color
  ... />
```

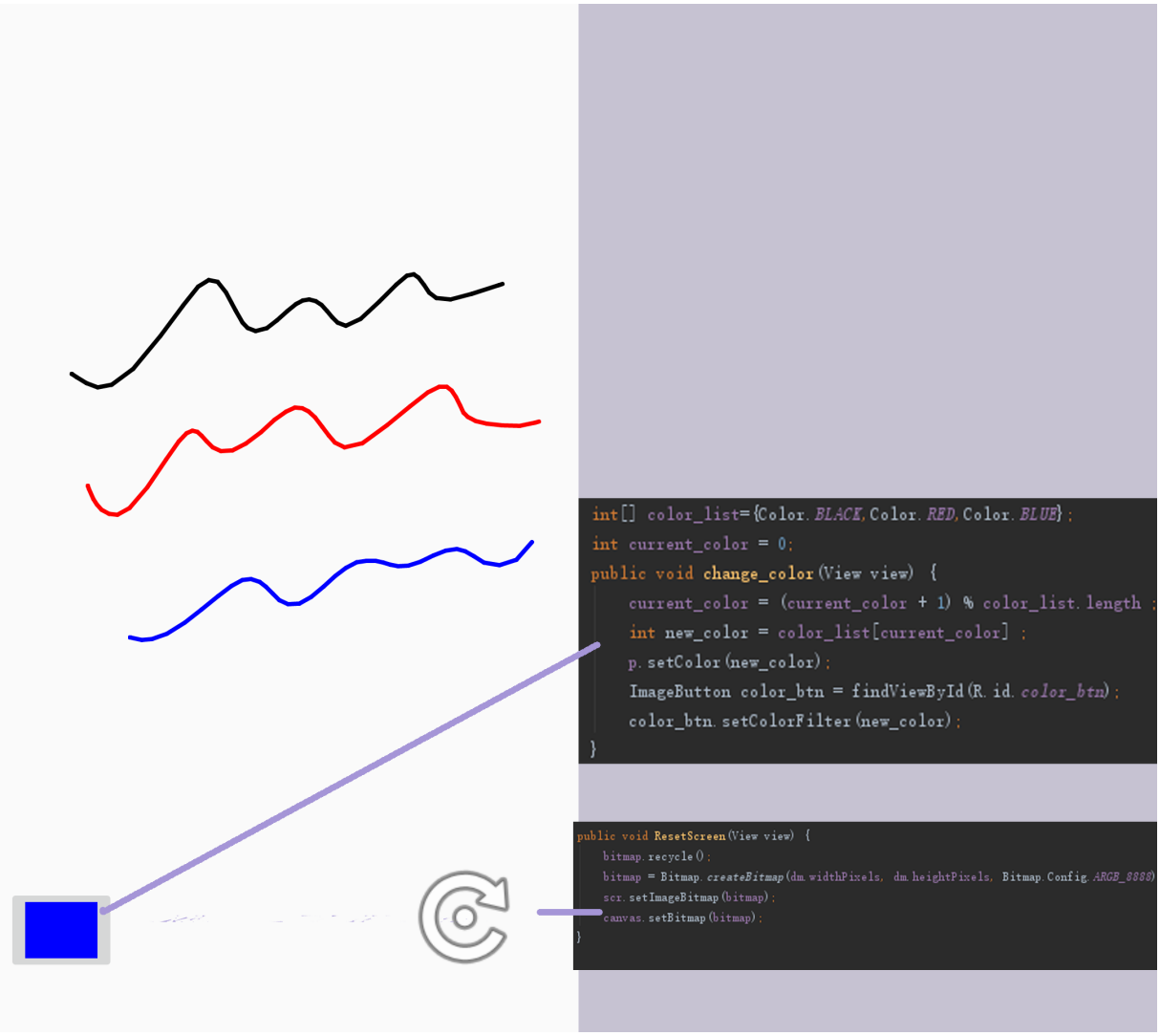
activity_draw.xml

4 RESULT

4.1 Print 'Hello World!'



4.2 Draw A Line



5 Conclusion

In this lab, I learned how to use Android Studio to design a simple mobile application and the usage of Canvas and Lsteners.