# Grafika i animacija

* Crtanje grafike u Androidu
* Primjena osnova OpenGL-a za ugrađene sustave (ES)
* Animiranje s Androidom

U ovom ćemo poglavlju pogledati kako izraditi grafiku pomoću Androidovog grafičkog API-ja, raspravljati o tome kako razviti animacije i istražiti Androidovu podršku za OpenGL standard, kao i upoznati vas s Androidovim novim višeplatformskim grafičkim jezikom visokih performansi RenderScript.

**Crtanje grafike u Androidu**

U ovom ćemo odjeljku pokriti Androidove grafičke mogućnosti i pokazati vam primjere kako napraviti jednostavne 2D oblike. Primijenit ćemo paket android.graphics (pogledajte http://mng.bz/CIFJ), koji pruža sve klase niske razine koje su vam potrebne za stvaranje grafike. Grafički paket podržava stvari kao što su bitmape (koje sadrže piksele), platna (ono što crtanje poziva na crtanje), primitive (kao što su pravokutnici i tekst) i boje (koje koristite za dodavanje boja i stila). Iako ovo nisu jedini grafički paketi, oni su glavni koje ćete koristiti u većini aplikacija. Općenito, koristite Javu za pozivanje grafičkog API-ja za stvaranje složene grafike.

Kako bismo demonstrirali osnove crtanja oblika pomoću Jave i grafičkog API-ja, pogledajmo jednostavan primjer u sljedećem popisu, gdje ćemo nacrtati pravokutnik.

Prvo, moramo uvesti potrebne pakete, uključujući grafiku. Zatim uvozimo ShapeDrawable, koji će podržati dodavanje oblika crtežu, a zatim oblike, koji podržava nekoliko generičkih oblika (uključujući RectShape) koje će se koristiti. Zatim moramo stvoriti, a zatim postaviti View. Nakon toga stvaramo novi ShapeDrawable da bismo dodali u naš Drawable .Nakon što imamo ShapeDrawable, možemo mu dodijeliti oblike. U kodu koristimo RectShape, ali može se koristiti OvalShape, PathShape, RectShape, RoundRectShape ili Shape. Zatim se koristi onDraw() metoda da se nacrta Drawable na platnu. Konačno, koristimo metodu setBounds() Drawable da bismo postavili granicu (okvir) u kojoj ćemo nacrtati naš pravokutnik pomoću metode draw().

Primjer kako dodati grafički element u android bez upotrebe activity\_main.xml

**package** com.example.myapplication15;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
  
  
**import** android.app.Activity;  
  
**import** android.content.Context;  
  
**import** android.graphics.Canvas;  
**import** android.graphics.drawable.ShapeDrawable;  
**import** android.os.Bundle;  
**import** android.view.View;  
  
  
**public class** MainActivity **extends** AppCompatActivity {  
@Override  
 **protected void** onCreate(Bundle icicle) {  
 **super**.onCreate(icicle);  
  
 setContentView(**new** SimpleView(**this**));  
}  
  
**private static class** SimpleView **extends** View {  
 **private** ShapeDrawable **mDrawable** = **new** ShapeDrawable();  
**public** SimpleView(Context context) {  
 **super**(context);  
 setFocusable(**true**);  
 **this**.**mDrawable**.getPaint().setColor(0xffff0000);  
  
 }  
 @Override  
 **protected void** onDraw (Canvas canvas){  
 **int** x = 10;  
 **int** y = 10;  
 **int** width = 300;  
 **int** height = 50;  
 **this**.**mDrawable**.setBounds(x, y, x + width, y + height);  
 **this**.**mDrawable**.draw(canvas);  
 y += height + 5;  
 }  
}  
}

Drugi način je da se to učini putem XML-a. Android omogućuje definiranje okvira za crtanje u XML datoteci resursa.

**Crtanje s XML-om**

Uz Android, mogu se izraditi jednostavni crteži koristeći pristup XML datoteci. To je jednostavniji način i vrijedi imati na umu da se grafika opisana XML-om može programski promijeniti kasnije, tako da XML pruža jednostavan način izrade početnog dizajna koji nije nužno statičan.

Da bi se izradio crtež s XML-om, stvorite jedan ili više Drawable objekata, koji su definirani kao XML datoteke u crtaćem direktoriju (res/drawable). XML za stvaranje jednostavnog pravokutnika izgleda ovako:

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"**>  
<**solid android:color="#FF0000FF"**/>  
</**shape**>  
 *<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"**>  
<**solid android:color="#FF0000FF"**/>  
</**shape**>

S Android XML crtaćim oblicima, zadani je pravokutnik, ali moguće je odabrati drugi oblik pomoću oznake tipa i odabirom vrijednosti oval, pravokutnik, linija ili luk. Da bi se koristio svoj XML oblik, mora se referencirati u izgledu.

*<?***xml version="1.0" encoding="utf-8"***?>*<**ScrollView xmlns:android="http://schemas.android.com/apk/res/android"  
android:layout\_width="fill\_parent"  
android:layout\_height="wrap\_content"**>  
<**LinearLayout  
 android:orientation="vertical"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"**>  
 <**ImageView android:layout\_width="fill\_parent"  
 android:layout\_height="50dip"  
 android:src="@drawable/simplerectangle"** />  
</**LinearLayout**>  
</**ScrollView**>

2. način

MainActivity

**package** com.example.myapplication15;  
  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** android.os.Bundle;  
  
  
  
**public class** MainActivity **extends** AppCompatActivity {  
 @Override  
 **public void** onCreate(Bundle icicle) {  
 **super**.onCreate(icicle);  
 setContentView(R.layout.***activity\_main***);  
 }  
 }

activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**ScrollView xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/izlaz"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 tools:ignore="SpeakableTextPresentCheck"**>  
  
 <**LinearLayout  
 android:id="@+id/layout"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
  
 <**ImageView  
 android:layout\_width="fill\_parent"  
 android:layout\_height="50dip"  
 android:src="@drawable/pravokutnik"** />  
  
 <**ImageView  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/krug"** />  
  
 <**ImageView  
 android:layout\_width="200dp"  
 android:layout\_height="200dp"  
 android:src="@drawable/kruznica"** />  
  
 <**ImageView  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 android:src="@drawable/trokut"** />  
 </**LinearLayout**>  
</**ScrollView**>

krug.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="oval"**>  
  
 <**solid  
 android:color="#576576"**/>  
  
 <**size  
 android:width="120dp"  
 android:height="120dp"**/>  
</**shape**>

kruznica.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="oval"**>  
 <**solid android:color="#00000000"**/>  
 <**padding android:left="10sp" android:top="4sp"  
 android:right="10sp" android:bottom="4sp"** />  
 <**stroke android:width="5dp" android:color="#FFF11FFF"**/>  
</**shape**>

pravokutnik.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"**>  
 <**solid android:color="#FF0000FF"**/>  
</**shape**>

Trokut.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**layer-list xmlns:android="http://schemas.android.com/apk/res/android"** >  
 <**item** >  
 <**rotate  
 android:fromDegrees="45"  
 android:toDegrees="45"  
 android:pivotX="-40%"  
 android:pivotY="87%"** >  
 <**shape  
 android:shape="rectangle"** >  
 <**stroke android:color="@android:color/transparent" android:width="10dp"**/>  
 <**solid  
 android:color="FF00FF00"** />

</**shape**>  
 </**rotate**>  
 </**item**>  
</**layer-list**>

ARGB VRIJEDNOSTI BOJA

Android koristi vrijednosti alfa, crvene, zelene, plave (ARGB) uobičajene u softverskoj industriji za definiranje vrijednosti boja kroz Android API. U RGB-u, boje su definirane kao int sastavljeni od četiri bajta: crvene, zelene i plave, plus alfa. Svaka vrijednost može biti broj od 0 do 255 koji se pretvara u heksadecimalni (heksadecimalni). Alfa označava razinu transparentnosti od 0 do 255.

Na primjer, da bismo stvorili prozirnu žutu boju, mogli bismo početi s alfa od 50,2% prozirnosti, gdje je heksadecimalna vrijednost 0x80: ovo je 128, što je 50,2% od 255. Da bismo dobili žutu, trebamo crvenu plus zelenu. Broj 255 u heksadecimalu za crvenu i zelenu je FF. Plava nije potrebna, pa je njena vrijednost 00. Tako je prozirna žuta 80FFFF00. Ovo može izgledati zbunjujuće, ali dostupne su brojne ARGB karte boja koje prikazuju heksadecimalne vrijednosti mnoštva boja.

Korištenje imenovane boje u datoteci resursa je zgodno kada se radi s uobičajenim bojama (kao što su standardne HTML nazvane web boje, tj. CSS boje ili X Window System i SVG nazivi boja). Datoteka resursa može se stvoriti za definiranje uobičajenih boja za aplikaciju. Ovdje je datoteka resursa za korištenje HTML/CSS naziva boja u Android kodu:

<**resources**>  
<**color name="White"**>#FFFFFFFF</**color**>  
<**color name="Silver"**>#FFC0C0C0</**color**>  
<**color name="Gray"**>#FF808080</**color**>  
<**color name="Black"**>#FF000000</**color**>  
<**color name="Red"**>#FFFF0000</**color**>  
<**color name="Maroon"**>#FF800000</**color**>  
<**color name="Yellow"**>#FFFFFF00</**color**>  
<**color name="Olive"**>#FF808000</**color**>  
<**color name="Lime"**>#FF00FF00</**color**>  
<**color name="Green"**>#FF008000</**color**>  
<**color name="Aqua"**>#FF00FFFF</**color**>  
<**color name="Teal"**>#FF008080</**color**>  
<**color name="Blue"**>#FF0000FF</**color**>  
<**color name="Navy"**>#FF000080</**color**>  
<**color name="Fuchsia"**>#FFFF00FF</**color**>  
<**color name="Purple"**>#FF800080</**color**>  
</**resources**>

Zatim koristiti :

* u java kodu

findViewById(R.id.textView1).setBackgroundColor(getResources().getColor(R.color.Purple))

* u xml kodu

<**TextView android:id="@+id/textView1"  
android:layout\_width="wrap\_content"  
android:layout\_height="wrap\_content"  
android:background="@color/Red"  
android:text="@string/hello\_world"**/>

Što je layer\_list?

Layer je sloj, list je popis, a zatim sloj-popis je značenje popisa slojeva. Popis slojeva se koristi za stvaranje Layer Drawable, Layer Drawable je jedan od resursa za crtanje, tako da popis slojeva stvara graf koji se može crtati.

*<?***xml version="1.0" encoding="UTF-8"***?>*<**layer-list xmlns:android="http://schemas.android.com/apk/res/android"** >  
  
*<!-- Colored rectangle-->*<**item**>  
 <**shape android:shape="rectangle"**>  
 <**size  
 android:width="100dp"  
 android:height="40dp"** />  
 <**solid android:color="#5EB888"** />  
 <**corners android:radius="0dp"**/>  
 </**shape**>  
</**item**>  
  
*<!—The rectangle from the top edge -->  
<!-- Its color should be the same as the layout's background -->*<**item  
 android:top="-40dp"  
 android:bottom="65dp"  
 android:right="-30dp"**>  
 <**rotate  
 android:fromDegrees="45"**>  
 <**shape android:shape="rectangle"**>  
 <**solid android:color="#ffffff"** />  
 </**shape**>  
 </**rotate**>  
</**item**>  
  
*<!-- This rectangle from the lower arrow edge -->  
<!-- Its color should be the same as the layout's background -->*<**item  
 android:top="65dp"  
 android:bottom="-40dp"  
 android:right="-30dp"**>  
 <**rotate  
 android:fromDegrees="-45"**>  
 <**shape android:shape="rectangle"**>  
 <**solid android:color="#ffffff"** />  
 </**shape**>  
 </**rotate**>  
</**item**>  
  
</**layer-list**>

RING

<**shape  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="ring"  
 android:innerRadius="80dp"  
 android:useLevel="false"**>  
  
 <**solid  
 android:color="@color/colorPrimary"**/>  
</**shape**>

**Crtanje u Canvasu**

Klasa Android Canvas inkapsulira bitmape koje se koriste kao površina. Izlaže metode crtanja koje se mogu koristiti za projektiranje. Prvo razjasnimo sljedeće pojmove:

**Bitmapa**: površina na kojoj se crta.

**Paint**: Omogućuje nam da odredimo kako nacrtati primitive na bitmapi. Također se naziva "četkom".

**Canvas**: Opskrbljuje metode crtanja koje se koriste za crtanje primitiva na temeljnoj bitmapi.

Svaki objekt crteža specificira objekt boje za renderiranje. Pogledajmo dostupni popis objekata za crtanje, a oni su sljedeći:

**drawArc**: Ovo crta luk između dva kuta omeđen područjem pravokutnika.

**drawBitmap**: Crta bitmapu na platnu.

**drawRGB**/**drawARGB**/**drawColor**: Ovo ispunjava platno jednom bojom.

**drawBitmapMesh**: crta bitmapu pomoću mreže. Manipulira izgledom mete pomicanjem točaka na njoj.

**drawCircle**: Ovo crta kružnicu na određenom polumjeru sa središtem na danu točku.

**drawLine**: crta liniju (ili niz linija) između točaka.

**drawOval**: crta oval koji je omeđen površinom pravokutnika.

**drawPaint**: ispunjava cijelo platno određenom bojom.

**drawPath**: crta put prema specifikaciji.

**drawPicture**: crta određenu sliku na pravokutnom području.

**drawPosText**: crta tekstualni niz koji specificira pomak svakog znaka.

**drawRect**: Crta pravokutnik.

**drawRoundRect**: crta pravokutnik sa zaobljenim rubovima.

**drawText**: crta tekstualni niz na platnu.

Klasa Paint sastoji se od kista i palete. Omogućuje nam da odaberemo kako ćemo prikazati primitive uvučene u platno metodama crtanja. Možemo kontrolirati boju, stil, font, specijalne efekte itd. može se mijenjati modificiranjem objekta boje.

Na primjer, metoda setColor može se koristiti za odabir boje. Klasa Paint podržava transparentnost tako da se može koristiti za kontrolu raznih nijansi ili efekata. Napravimo jednostavan primjer i pogledajte osnovnu upotrebu platna i boje.

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout      xmlns:android="<http://schemas.android.com/apk/res/android>"      xmlns:tools="<http://schemas.android.com/tools>"      android:id="@+id/idRLView"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".MainActivity">  </RelativeLayout> |
| package com.example.myapplication15; |

import android.os.Bundle;

import android.widget.RelativeLayout;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    // creating a variable for our relative layout

    private RelativeLayout relativeLayout;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // initializing our view.

        relativeLayout = findViewById(R.id.idRLView);

        // calling our  paint view class and adding

        // its view to our relative layout.

        PaintView paintView = new PaintView(this);

        relativeLayout.addView(paintView);

    }

}

**import** android.annotation.SuppressLint;

**import** android.app.Activity;

**import** android.content.Context;

**import** android.graphics.Canvas;

**import** android.graphics.Color;

**import** android.graphics.Paint;

**import** android.util.DisplayMetrics;

**import** android.view.View;

**public** **class** PaintView **extends** View {

    // below we are creating variables for our paint

    Paint otherPaint, outerPaint, textPaint;

    // and a floating variable for our left arc.

**float** arcLeft;

    @SuppressLint("ResourceAsColor")

**public** PaintView(Context context) {

**super**(context);

        // on below line we are initializing our paint variable for our text

        textPaint = **new** Paint(Paint.LINEAR\_TEXT\_FLAG | Paint.ANTI\_ALIAS\_FLAG);

        // on below line we are setting color to it.

        textPaint.setColor(Color.WHITE);

        // on below line we are setting text size to it.

        // In Paint we have to add text size using px so

        // we have created a method where we are converting dp to pixels.

        textPaint.setTextSize(pxFromDp(context, 24));

        // on below line we are initializing our outer paint

        outerPaint = **new** Paint();

        // on below line we are setting style to our paint.

        outerPaint.setStyle(Paint.Style.FILL);

        // on below line we are setting color to it.

        outerPaint.setColor(getResources().getColor(R.color.purple\_200));

        // on below line we are creating a display metrics

        DisplayMetrics displayMetrics = **new** DisplayMetrics();

        // on below line we are getting display metrics.

        ((Activity) getContext()).getWindowManager()

                .getDefaultDisplay()

                .getMetrics(displayMetrics);

        // on below line we are assigning

        // the value to the arc left.

        arcLeft = pxFromDp(context, 20);

        // on below line we are creating

        // a new variable for our paint

        otherPaint = **new** Paint();

    }

    // below method is use to generate px from DP.

**public** **static** **float** pxFromDp(**final** Context context, **final** **float** dp) {

**return** dp \* context.getResources().getDisplayMetrics().density;

    }

    @Override

**protected** **void** onDraw(Canvas canvas) {

**super**.onDraw(canvas);

        // below four lines of code is use to add

        // back color to our screen which is green

        canvas.drawPaint(outerPaint);

        // on below line we are setting color to our paint.

        otherPaint.setColor(Color.WHITE);

        // on below line we are setting style to out paint.

        otherPaint.setStyle(Paint.Style.FILL);

        // below 4 lines of code is use to

        // create white rectangle of screen

        canvas.drawRect(

                getLeft() + (getRight() - getLeft()) / 3,

                getTop() + (getBottom() - getTop()) / 3,

                getRight() - (getRight() - getLeft()) / 3,

                getBottom() - (getBottom() - getTop()) / 3, otherPaint);

        // on below line we are changing the color for our paint.

        otherPaint.setColor(getResources().getColor(R.color.purple\_200));

        // on below line we are drawing a circle and passing

        // width, height, left arc and paint to add color.

        canvas.drawCircle(getWidth() / 2, getHeight() / 2, arcLeft, otherPaint);

        // on below line we are adding text using paint in our canvas.

        canvas.drawText("Geeks for Geeks", (**float**) (getWidth() \* 0.3), (**float**) (getHeight() \* 0.8), textPaint);

    }

}

<**resources**>  
<**string name="app\_name"**>Learn Android with Real Apps</**string**>  
<**string name="draw\_line"**>Draw Line</**string**>  
<**string name="draw\_rectangle"**>Draw Rectangle</**string**>  
<**string name="draw\_circle"**>Draw Circle</**string**>  
<**string name="draw\_text"**>Draw Text</**string**>  
<**string name="draw\_image"**>Draw Image</**string**>  
</**resources**>

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu xmlns:android="http://schemas.android.com/apk/res/android"**>  
  
 <**item  
 android:id="@+id/drawLine"  
 android:title="@string/draw\_line"**></**item**>  
  
 <**item  
 android:id="@+id/drawRectangle"  
 android:title="@string/draw\_rectangle"**></**item**>  
  
 <**item  
 android:id="@+id/drawCircle"  
 android:title="@string/draw\_circle"**></**item**>  
  
 <**item  
 android:id="@+id/drawText"  
 android:title="@string/draw\_text"**></**item**>  
  
 <**item  
 android:id="@+id/drawImage"  
 android:title="@string/draw\_image"**></**item**>  
  
</**menu**>

**package** com.example.myapplication15;

**import** android.content.Context;  
 **import** android.graphics.Canvas;  
 **import** android.graphics.Color;  
 **import** android.graphics.Paint;  
 **import** android.graphics.PorterDuff;  
 **import** android.view.View;  
  
**public class** DrawLine **extends** View {  
  
 Paint **paint** = **new** Paint();  
  
 **public** DrawLine(Context context) {  
 **super**(context);  
 }  
  
 @Override  
 **public void** onDraw(Canvas canvas) {  
 *// Clear Canvas* canvas.drawColor(Color.WHITE, PorterDuff.Mode.MULTIPLY);  
  
 *// Draw Line* **paint**.setColor(Color.RED);  
 **paint**.setStrokeWidth(20);  
 canvas.drawLine(50, 100, 600, 600, **paint**);  
 canvas.drawLine(50, 550, 770, 0, **paint**);  
 }  
  
}

**package** com.example.myapplication15;  
**import** android.content.Context;  
**import** android.graphics.Canvas;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.PorterDuff;  
**import** android.view.View;  
  
**public class** DrawCircle **extends** View {  
  
 Paint **paint** = **new** Paint();  
  
 **public** DrawCircle(Context context) {  
 **super**(context);  
 }  
  
 @Override  
 **public void** onDraw(Canvas canvas) {  
 *// Clear Canvas* canvas.drawColor(Color.***WHITE***, PorterDuff.Mode.***MULTIPLY***);  
  
 *// Draw Circle* **paint**.setColor(Color.***RED***);  
 **paint**.setStrokeWidth(5);  
 canvas.drawCircle(200, 200, 150, **paint**);  
 }

**package** com.example.myapplication15;  
**import** android.content.Context;  
**import** android.graphics.Bitmap;  
**import** android.graphics.BitmapFactory;  
**import** android.graphics.Canvas;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.PorterDuff;  
**import** android.view.View;  
  
**public class** DrawImage **extends** View {  
  
 Paint **paint** = **new** Paint();  
  
 **public** DrawImage(Context context) {  
 **super**(context);  
 }  
  
 @Override  
 **public void** onDraw(Canvas canvas) {  
 *// Clear Canvas* canvas.drawColor(Color.***WHITE***, PorterDuff.Mode.***MULTIPLY***);  
  
 *// Draw Circle* Bitmap bitmap = BitmapFactory.*decodeResource*(getResources(), R.drawable.slika);  
 **paint**.setColor(Color.***RED***);  
 canvas.drawBitmap(bitmap, 100, 100, **paint**);  
 }  
  
}

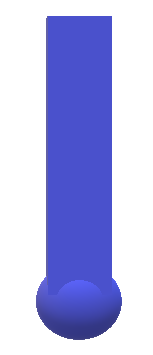
activity\_main

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:background="#ffffff"  
 android:orientation="vertical"  
 tools:ignore="HardcodedText"** >  
  
</**LinearLayout**>

mainActivity

**package** com.example.myapplication15;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.Menu;  
**import** android.view.MenuInflater;  
**import** android.view.MenuItem;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 MenuInflater inflater = getMenuInflater();  
 inflater.inflate(R.menu.***main***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **if** (item.getItemId() == R.id.drawLine) {  
 drawLine();  
 } **else if**(item.getItemId() == R.id.drawRectangle) {  
 drawRectangle();  
 } **else if**(item.getItemId() == R.id.drawCircle) {  
 drawCircle();  
 } **else if**(item.getItemId() == R.id.drawText) {  
 drawText();  
 } **else if**(item.getItemId() == R.id.drawImage) {  
 drawImage();  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **private void** drawImage() {  
 DrawImage drawImage = **new** DrawImage(**this**);  
 setContentView(drawImage);  
 }  
  
 **private void** drawText() {  
 DrawText drawText = **new** DrawText(**this**);  
 setContentView(drawText);  
 }  
  
 **private void** drawCircle() {  
 DrawCircle drawCircle = **new** DrawCircle(**this**);  
 setContentView(drawCircle);  
 }  
  
 **private void** drawRectangle() {  
 DrawRectangle drawRectangle = **new** DrawRectangle(**this**);  
 setContentView(drawRectangle);  
 }  
  
 **private void** drawLine() {  
 DrawLine drawLine = **new** DrawLine(**this**);  
 setContentView(drawLine);  
 }  
  
}

zadatak1



Zadatak 2

