Vilniaus Jono Basanavičiaus gimnazija

Mobiliųjų aplikacijų kūrimas

Atliko: 3 klasės mokiniai Evaldas Latoškinas, Justas Venckus, Rokas Norbutas

Vadovas: Informatikos mokytoja Giedrė M. Kvizikevičienė

Vilnius

2016-2017 m.

Įžanga

Gavome užduotį sukurti mobiliąją aplikaciją Android operacinei sistemai. Nusprendėme pasirinkti programą Android Studio, nes ji yra oficiali Android programėlių kūrimo programa, kuri pasižymi patogumu ir kuriai galima surasti daug oficialios mokomosios medžiagos.

Mūsų darbo eiga buvo tokia:

1. Parsisiuntėme programą iš Android Studio tinklapio.
2. Atidarėme programą, išsinagrinėjome, kaip viskas veikia.
3. Pasiskirstėme projekto darbus – programavimą, programos išdėstymą, skaidres/aprašą.
4. Kai dirbame prie projekto, kiekvienu prisėdimu padarome kažką naujo. Suskirstome tai darbais, nufotografuojame vaizdą, aprašome eigą.

Nusprendėme padaryti paprastą verslo simuliacijos programą. Joje:

- Pradedama su tam tikra suma pinigų.
- Galima pirkti ir pardavinėti prekes.
- Prekių kainos nuolat keičiasi.
- Žaidimas baigiasi, kai bankrutuojama

Žaidimas vyksta savaitėmis, kiekvieną savaitę turimas tam tikras veiksmų kiekis, kuris priklauso nuo veiksmų patobulinimo lygio. Per savaitę gaunamas tam tikras pardavimo ir pirkimo pasiūlymų kiekis ir belieka tik žaidėjui nuspręsti, ką jis darys. Baigus savo veiksmus, žaidėjas gali progresuoti į sekančią savaitę. Kas penkias savaites yra mokami mokesčiai, taip atsiranda galimybė bankrutuoti, todėl su savo sprendimais reikia elgtis atsargiai, norint tapti sėkmingu šiame žaidime.

Žaidime vyrauja viena pagrindinė statistika, nusakantį gaunamų pasiūlymų kiekį – populiarumas. Jis gali būti didinamas perkant arba parduodant produktus, išnaudojant visus galimus veiksmus arba perkant patobulinimą.

Žaidimą kūrėme anglų kalba, kadangi taikomės į tarptautinę rinką.

Žaidime yra 5 naudojamos veiklos (kitaip tariant, skirtingi ekranai/meniu) – pagrindinė, pirkimo/pardavimo, patobulinimų, statistikų ir marketingo statistikų. Yra dar ir šešta – pradinė veikla, kurioje pradedamas naujas žaidimas.

Android Studio

Tai yra oficiali *IDE*, skirta Android aplikacijų kūrimui. Šioje programoje yra daugybė funkcijų, patogių programos kūrimui. Parašyta Java kalba. Programoje galima naudoti Java, C ir C++ kalbas. Joje yra daugybe funkcijų, reikalingų ir padedančių sukurti funkcionalią Android programėlę, pvz:

- Android Emulator'ius, leidžiantis per kompiuterį išbandyti Android programėlę.
- Patogus ekrano išdėstymas, galima įdėti mygtuką, teksto lauką ir t.t. per kelias sekundes.
- Galimybė suderinti programą su bet kuria Android versija.

IDE (angl. Integrated Development Environment) – integruota kūrimo aplinka. IDE pasižymi savybėmis:

- Patogi grafinė vartotojo sąsaja;
- Automatinis teksto pabaigimas ir generavimas;
- Galimybė derinti programas;
- Patogus versijų kontrolės sistemų klientas;
- Integruotas kompiliatoriaus valdymas ar pats kompiliatorius;
- Grafinis programos struktūros vaizdavimas

Veikimo principas

Android Studio programėlės veikia tokiu principu:

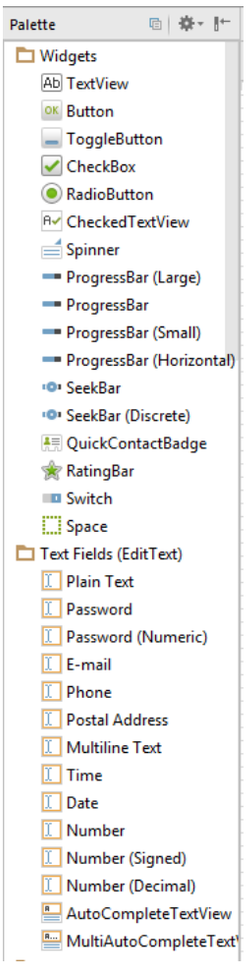
- Programos Android Studio veikia veiklomis – tai vienas aplikacijos ekranas, kuriame galima įvykdyti kokį nors veiksmą. Kiekviena veikla turi skirtinga kodą ir ekrano išdėstymą. Vienas veiklos pavyzdys būtų prisijungimo ekranas kurioje nors programoje, pvz. Facebook.
- Programos dažniausiai susideda iš vienos pagrindinės veiklos ir kitų veiklų, kurios yra tarpusavyje susietos. Pagrindinė veikla yra ta, nuo kurios pradedama programa, kurią mato vartotojas, pirmą kart paleidęs programą.
- Programoje kodas rašomas JAVA kalba.
- Išdėstymas objektų pozicijos yra reliatyvios, t.y. objektai turi būti po, prieš, virš, už kitų objektų (jų pozicijos turi būti susijusios). Tai yra nustatoma gairėse.

JAVA ir OOP

Android Studio programėlės kuriamos JAVA kalba (galima naudoti C ir C++ kalbas, tačiau jos nėra pagrindinės). JAVA kalba yra objektinio programavimo kalba (OOP), t.y. kodas veikia atskirais objektais – kiekvienas objektas turi savo kintamuosiuos, funkcijas, objektus galima susieti. Dauguma kuriamų sudėtingų programų naudojasi būtent šiuo principu programavime. Šio principo pavyzdys gali būti realus pavyzdys – įsivaizduokite, kad automobilis yra vienas bendras objektas, turintis savo savybes (markę, svorį, spalvą, t.t.) ir susideda iš smulkesnių objektų (pvz. variklio, korpuso, ratų..) bei tarpusavyje susietų arba savarankiškų funkcijų (variklio užvedimas, važiavimas..)

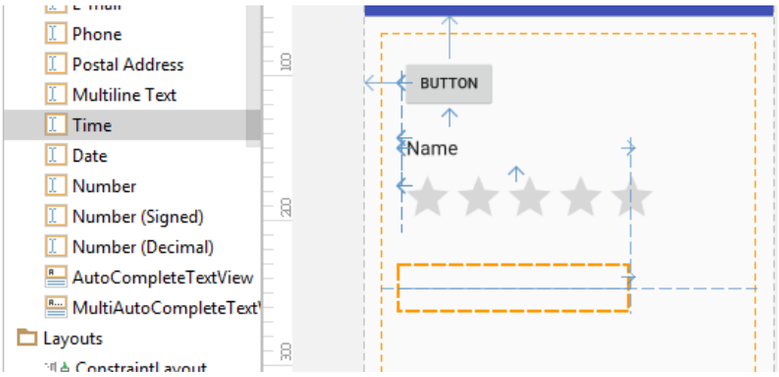
Programos išdėstymas

Paletė

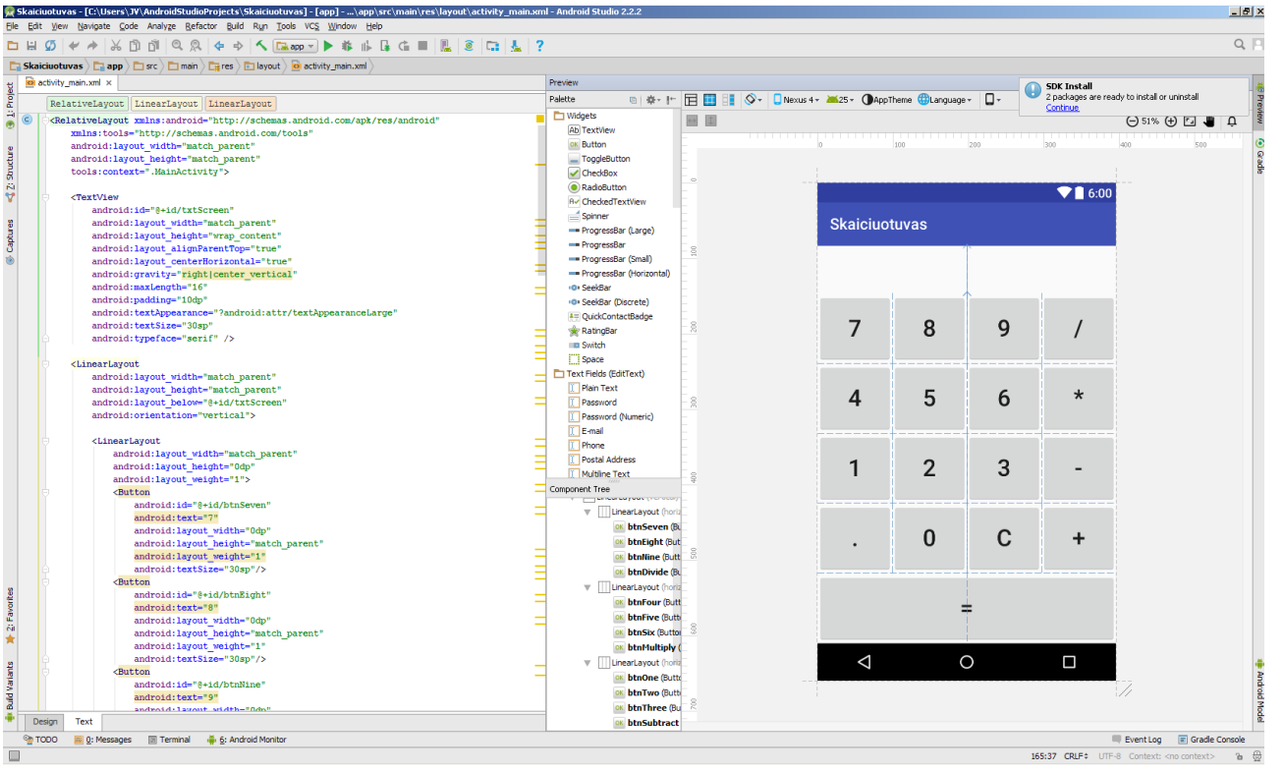


Iš paletės galima paprasčiausiai įstumti objektą į veiklos ekraną. Šis objektas įgauna savo savybes – pavadinimą, poziciją, dydį ir kitas jam budingas savybes. Kiekvieną objektą galima susieti kodu. Šių objektų pavyzdžiai:

- Mygtukai;
- Įvairūs tekstiniai laukai;
- Nuotraukos;
- Chronometras;
- Ir t.t.



XML



XML (Extensible Markup Language) – bendros paskirties duomenų struktūrų bei jų turinio aprašomoji kalba.

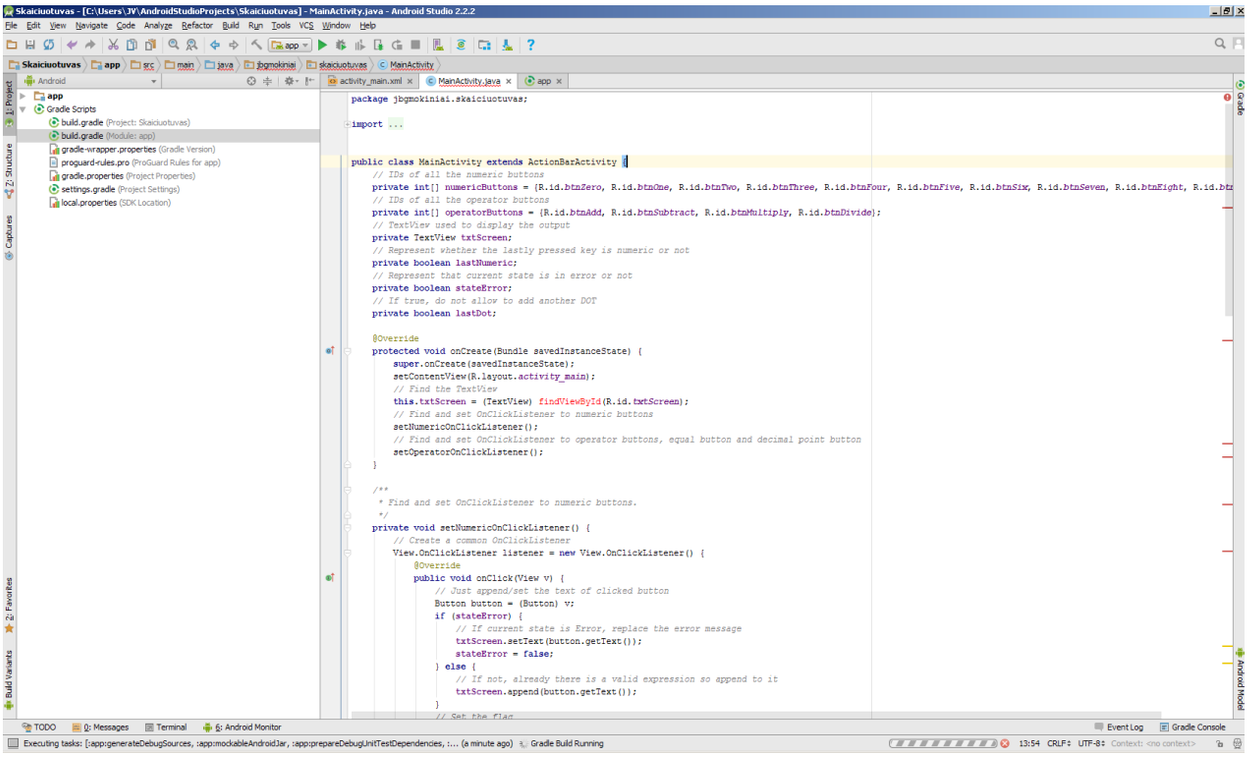
Dauguma failų šioje programoje yra būtent šiuo formatu (pvz veiklos išdėstymas – jį galima prieiti XML formatu arba keisti vizualiai ekrane). Failuose yra rašoma gairėmis, yra struktūra mažėjančia tvarka. Pvz:

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1">
    <Button
        android:id="@+id/btnSeven"
        android:text="7"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:textSize="30sp"/>
    <Button
        android:id="@+id/btnEight"
        android:text="8"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:textSize="30sp"/>
```

Tai mygtukai (Button), esantys LinearLayout (viena mygtukų eilutė skaičiuotuve) skiltį. Kiekvienas mygtukas išskiriamas nauja gaire, o viduj išskiriamos jo savybės.

Veiklos Kodas

Kiekviena veikla turi savo kodą, rašoma JAVA kalba. Šis kodas suteikia veiklai visą funkcionalumą, pvz. paspaudus mygtuką kažkas įvyksta, tarkim, įvedamas skaičius į skaičiuotuvą ir pnš.



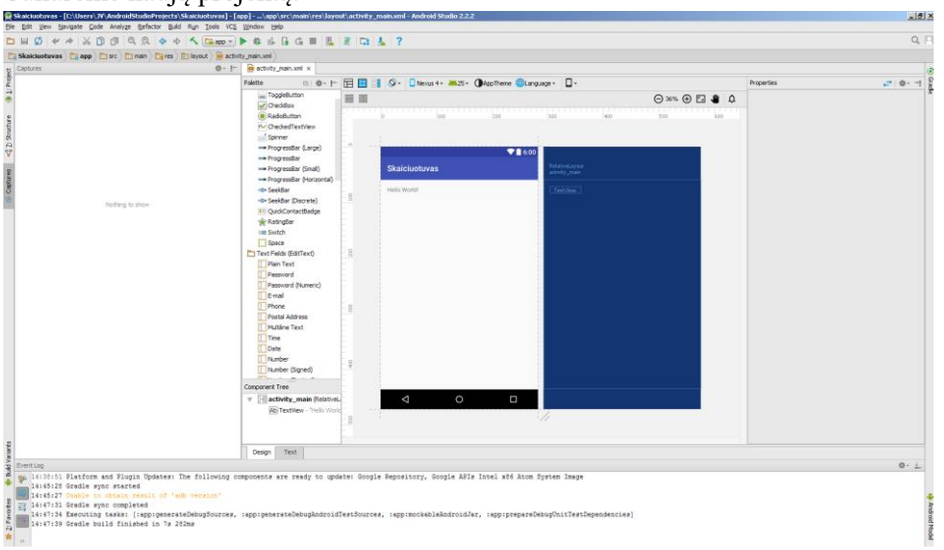
Veiklos kodas veikia tokiu principu:

- **Imports** (jie veikia kaip pvz. C++ include'ai) – tai bibliotekos, kurios leidžia naudotis tam tikromis funkcijomis, dirbti su tam tikrais kintamaisiais ir pnš.
- Kodo faile yra pagrindinė klasė – tai vienas objektas. Veikla kode yra vienas objektas. Ji turi savo kintamuosius, funkcijas.
- Kiekviena veikla gali turėti pagrindinę funkciją OnCreate(). Ši funkcija yra vykdoma tik tada, kai pirmą kartą paleidžiama (kitaip tariant, sukuriama) veikla.

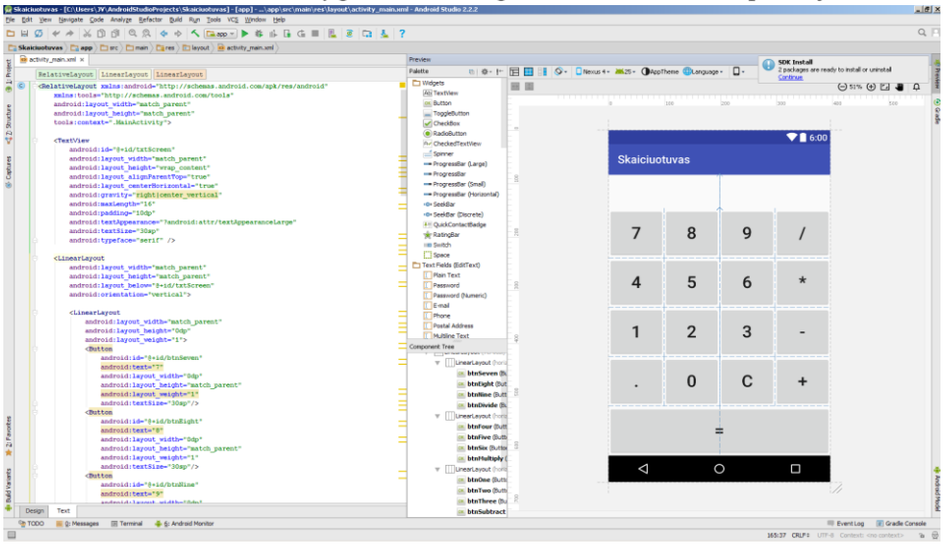
Pirmas praktinis darbas – Skaičiuotuvas

Iš pradžių pradėjome nuo pavyzdinės programos – skaičiuotuvo.

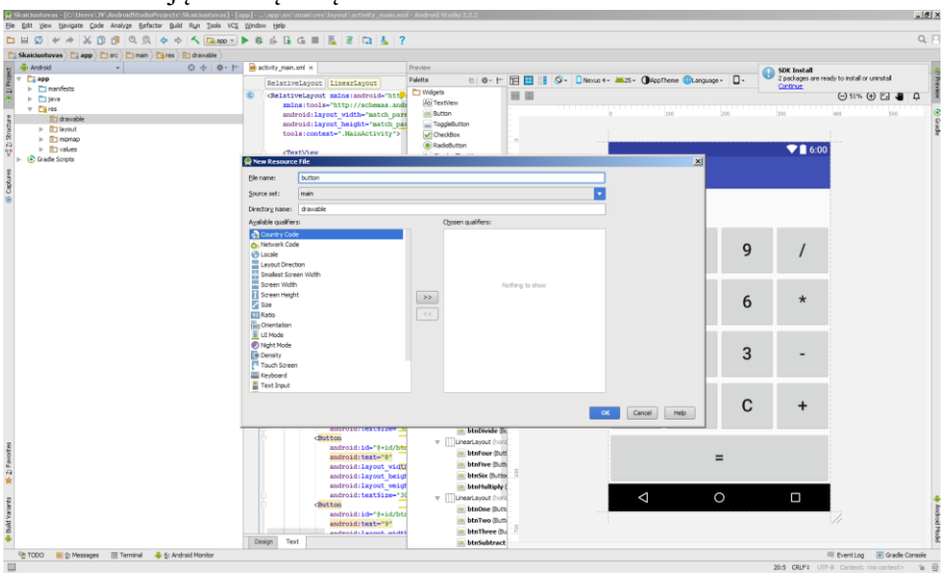
1. Sukūrėme naują projektą.



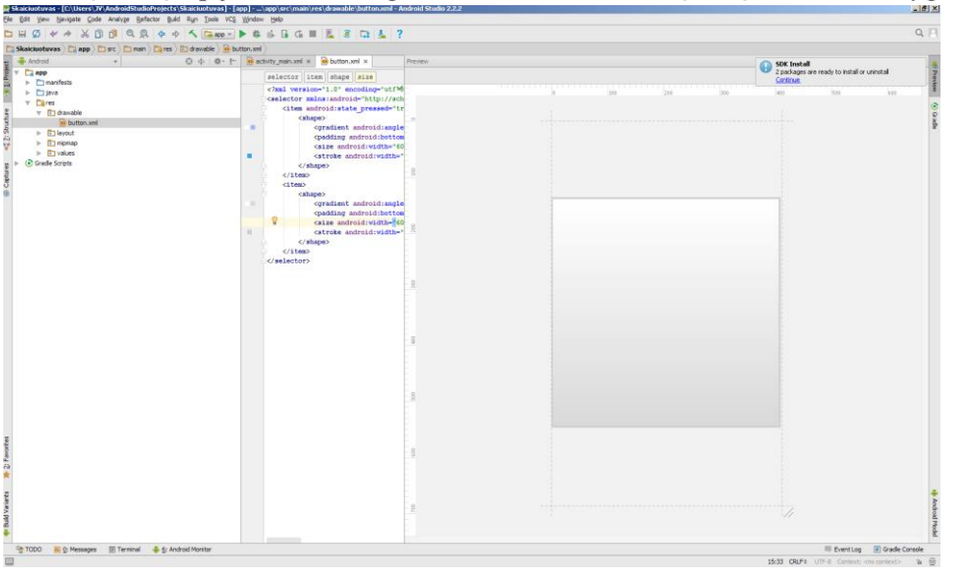
2. Kadangi mygtukai identiški, mes jų po vieną nedėliojome. Veiklos XML faile įkopijavome gaires, kurios sukuria identiškus mygtukus skirtingiems skaičiams bei operacijoms.



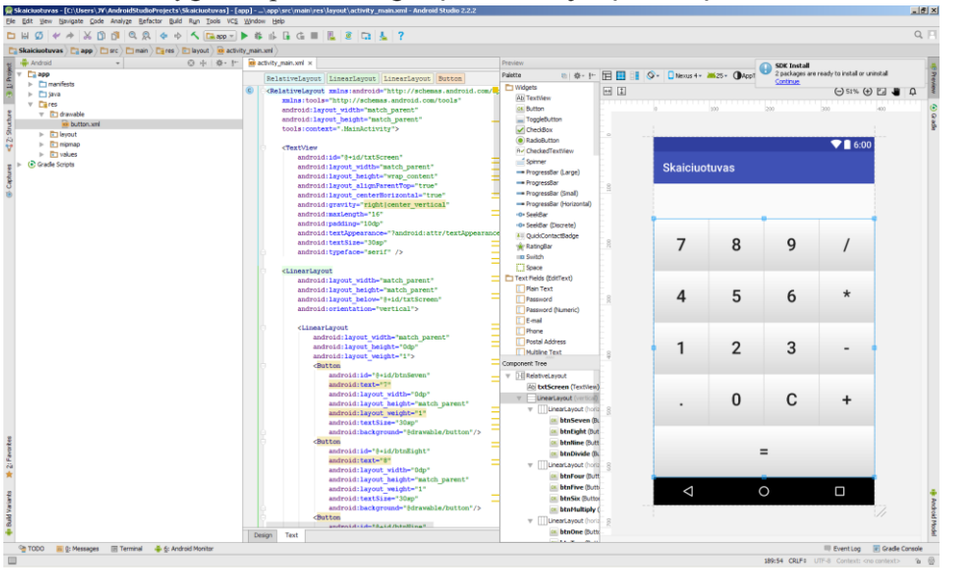
3. Sukūrėme naują resursų failą.



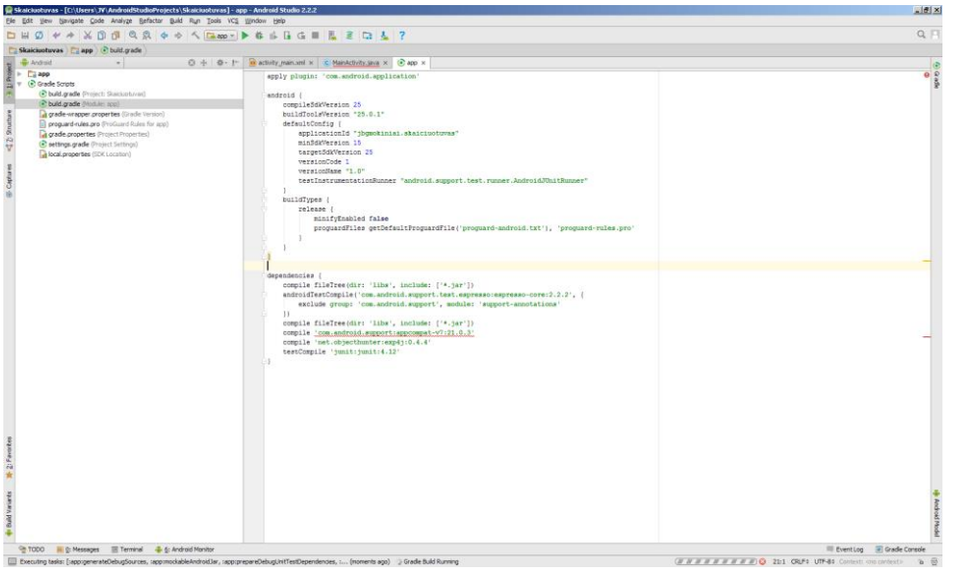
4. Resursų faile įkopijavome XML gaires, kad sukurti dizainą, kurį naudosime mygtukams.



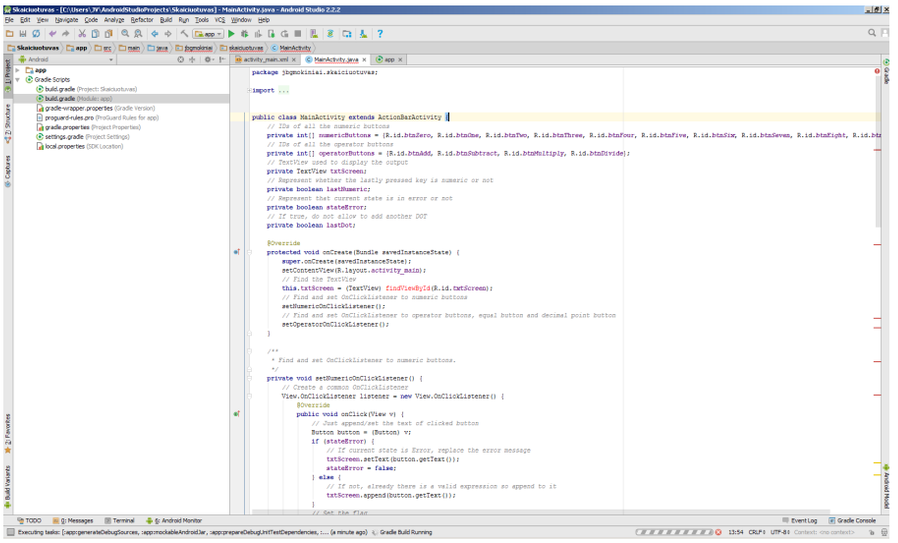
5. Kiekvienam mygtukui prirašėme gairę, kuri uždėjo šį dizainą.



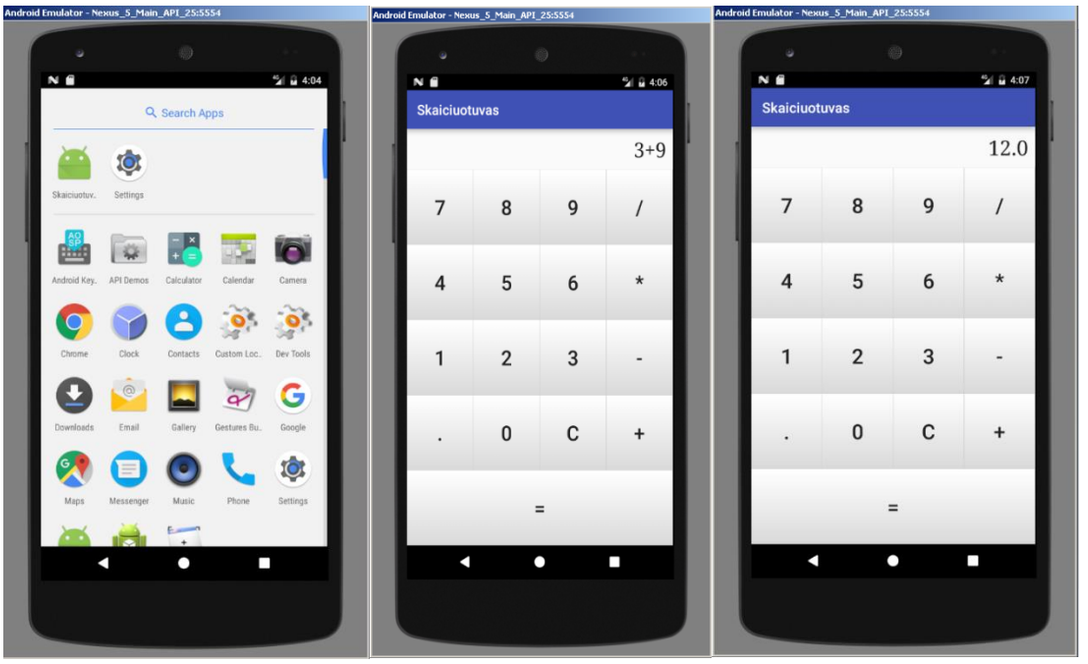
6. Į pagrindinį kompiliavimo failą (build.gradle) įrašėme kodo priklausomybes, kurios leidžia atlikti aritmetinius veiksmus.



7. Į pagrindinės veiklos kodo failą įkopijavome kodą, kuris suteikia mygtukams funkciją.



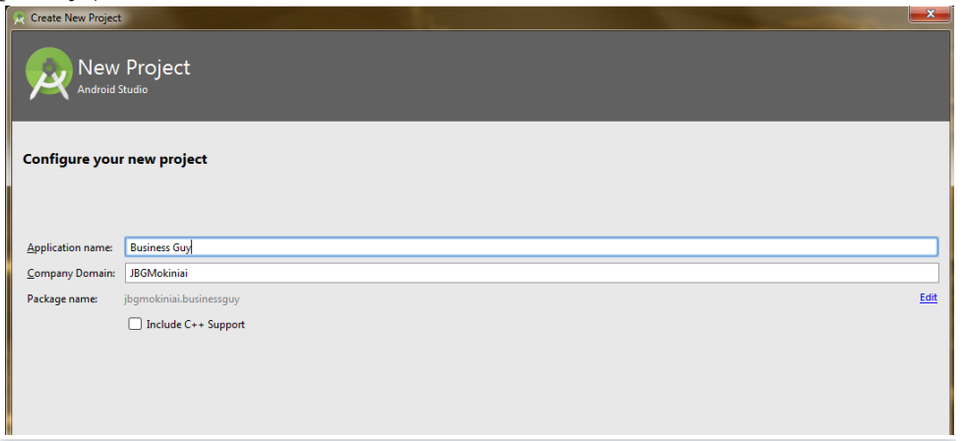
8. Sukompiliavome ir paleidome programą. Paleidome ją per Android emuliacinę programą. Išbandėme, ar veikia.

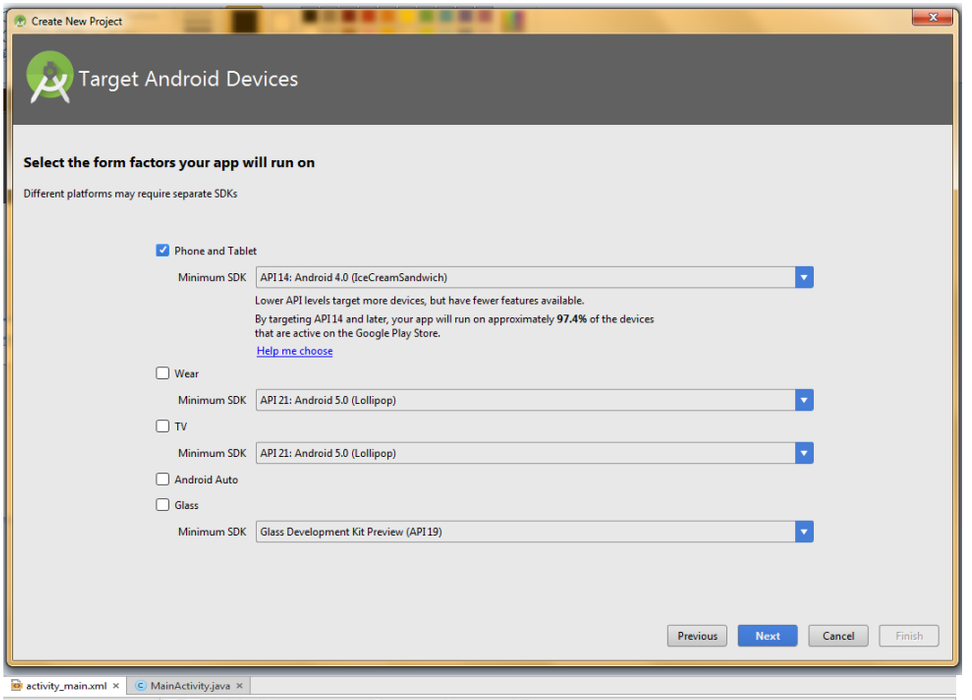


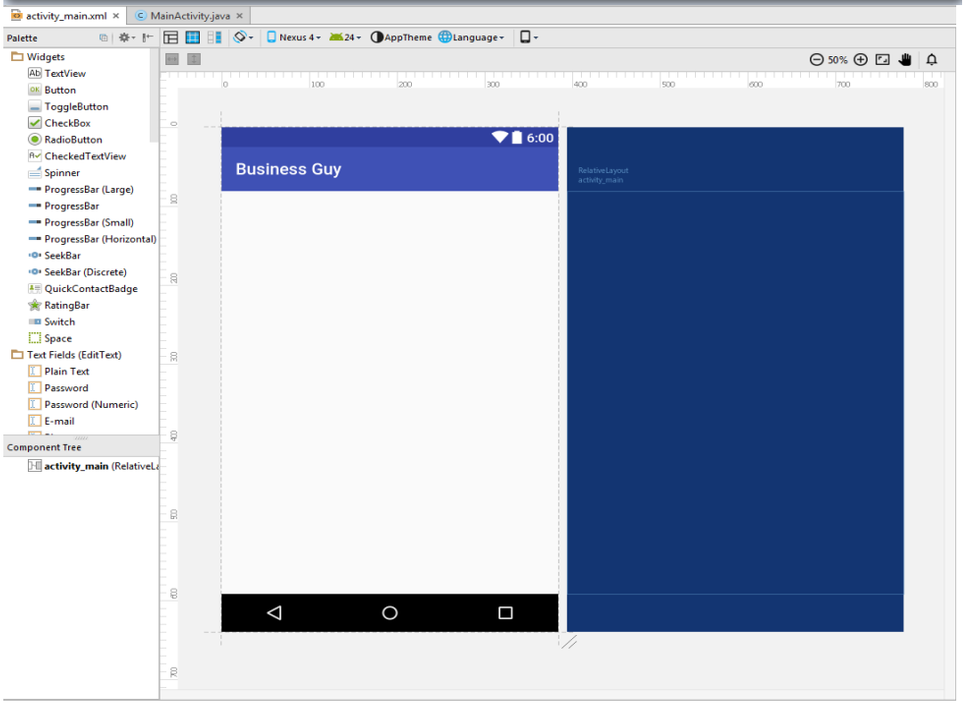
Pagrindinis Projektas – „Business Guy“

Pirmas Darbas – Projekto pradžia, pinigai

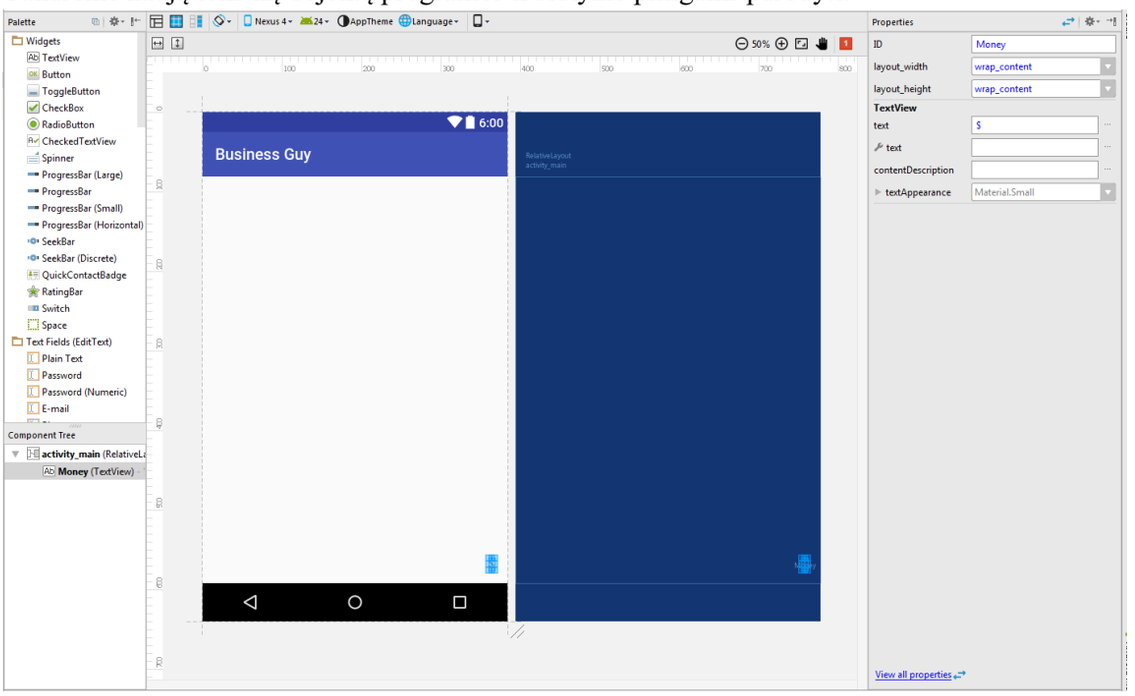
1. Sukūrėme naują projektą. Pritaikėme 4.0 Android sistemai (97,4% dabartinių Android prietaisų palaikys)







2. Sūkūrėme naują tekstinį objektą programos išdėstyme pinigams parodyti.



3. Pagrindinės veiklos kodo faile (MainActivity.Java) įvedėme pinigų kintamąjį ir parašėme kodą, kad atidarius programą būtų pakeistas ankščiau sukurtas pinigų kiekio teksto laukas. (money – pinigų kintamasis, moneytext – teksto lauko pinigų kiekiui parodyti kintamasis. Toliau randamas šis teksto laukas gairėse (MainActivity.XML) ir pakeičiamas ChangeMoneyText() funkcija.)

```
package jbgmokiniai.businessguy;

import ...

public class MainActivity extends AppCompatActivity
{
    public int money = 0;
    private TextView moneytext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.moneytext = (TextView) findViewById(R.id.moneytext);
        ChangeMoneyText();
    }

    public void ChangeMoneyText()
    {
        moneytext.setText("$" + money);
    }
}
```

4. Pagrindinės veiklos XML gairėse pakeitėme pinigų kiekio teksto lauko ID, kad jis sutaptu su tuo, esančiu kode (taip kode randamas teksto laukas TextView)

MainActivity.java

```java
package jbgmokiniai.businessguy;

import ...

public class MainActivity extends AppCompatActivity
{
    public int money = 0;
    private TextView moneytext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
        money = pref.getInt("money", 0);
        this.moneytext = (TextView) findViewById(R.id.moneytext);
        ChangeMoneyText();
        ChangeMoney(5);
    }

    public void ChangeMoneyText()
    {
        moneytext.setText("$" + money);
    }
    public void ChangeMoney(int amount)
    {
        money += amount;
        SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
        prefeditor.putInt("money", money);
        prefeditor.commit();
    }
}
```

```xml
<TextView
    android:text="$"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/moneytext"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />
```

5. Išbandėme per emuliacinę programą. Tekstas buvo pakeistas iš „$“ į „$0“, nes kolkas buvo pinigų kintamojo pradinė vertė buvo nustatyta į 0.

6. Prirašėme kodo:
    a. Galimybė padidinti/sumažinti pinigų kiekį su funkcija ChangeMoney. Į ją įvedamas pinigų kiekis, kurio norima pakeisti. Pakeičiama pinigų suma, išsaugoma į duomenų failą.
    b. Paleidus programą, pinigai nustatomi į tą sumą, kuri yra išsaugota pagrindiniame programos vartotojo duomenų faile „BusinessGuyPrefs“.

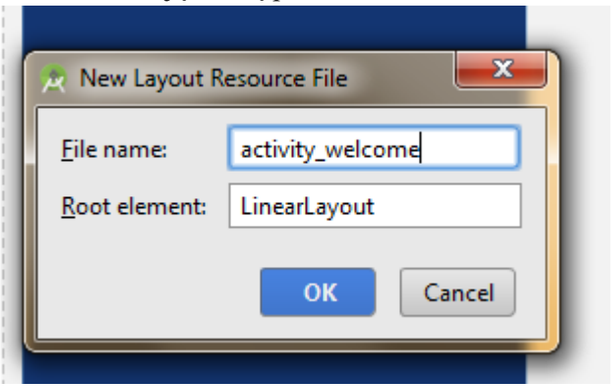
MainActivity.java

```
package jbgmokiniai.businessguy;

import ...

public class MainActivity extends AppCompatActivity
{
    public int money = 0;
    private TextView moneytext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
        money = pref.getInt("money", 0);
        this.moneytext = (TextView) findViewById(R.id.moneytext);
        ChangeMoneyText();
        ChangeMoney(5);
    }

    public void ChangeMoneyText()
    {
        moneytext.setText("$" + money);
    }
    public void ChangeMoney(int amount)
    {
        money += amount;
        SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
        prefeditor.putInt("money", money);
        prefeditor.commit();
    }
}
```

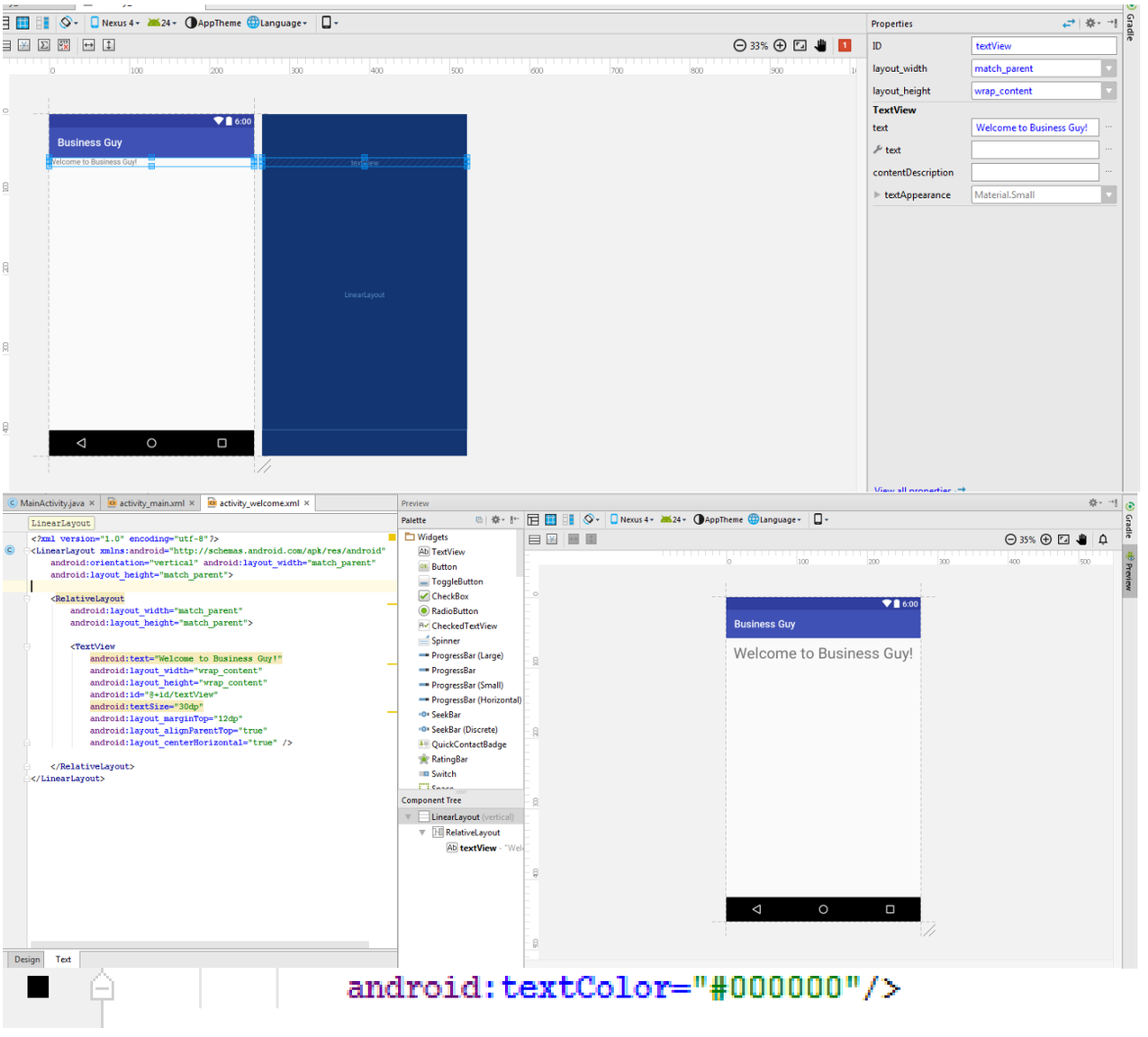
7. Prirašėme kodą, kad kiekviena kartą paleidus programą būtu pridedami $5. Išjungėme ir įjungėme programą porą kartų, kad patikrinti, ar veikia duomenų išsaugojimas.

Antras Darbas – Pradžios Ekranas

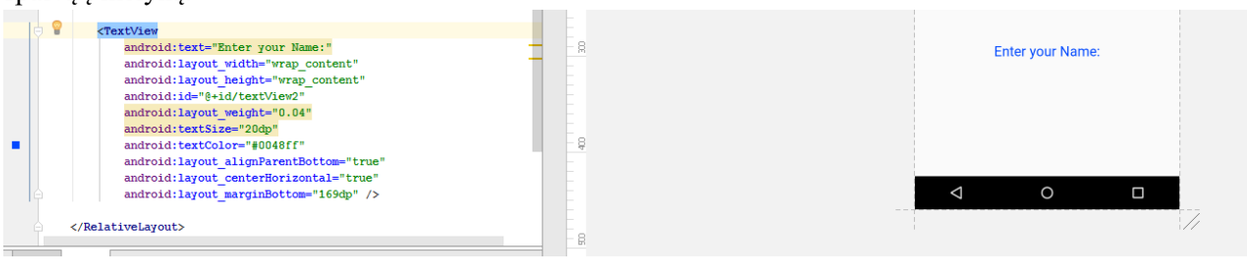
1. Sukūrėme naują veiklą pradžios ekranui.



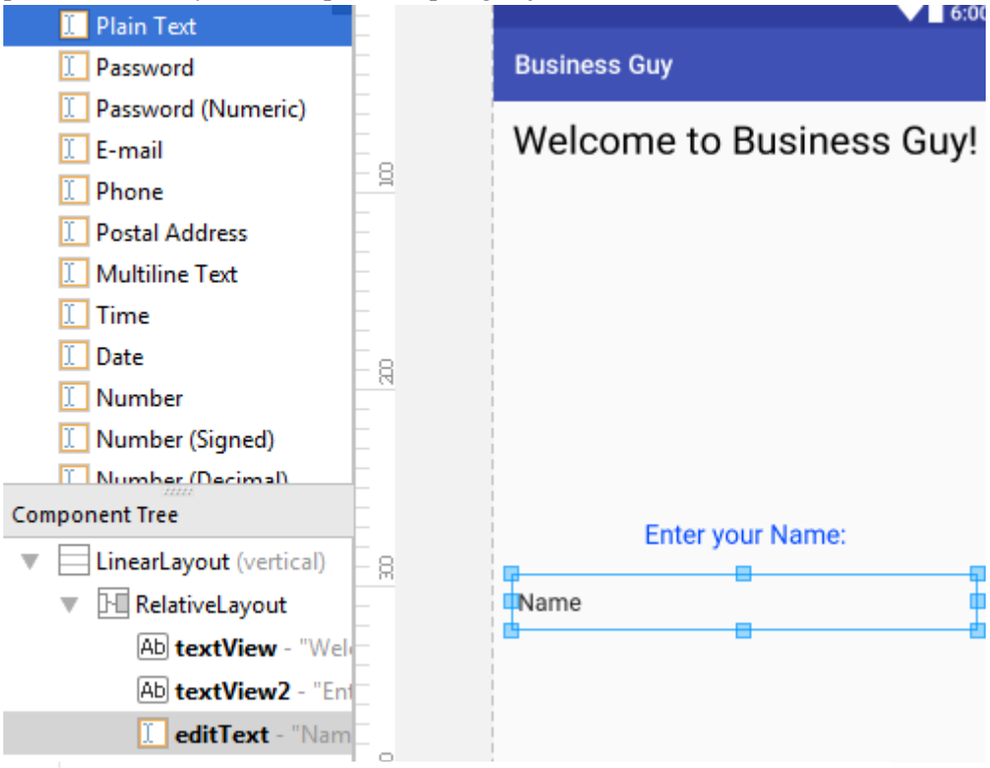
2. Sukūrėme naują teksto objektą iš paletės kuris pasveikina žaidėją atvykusį į žaidimą. Pakeitėme teksto dydį, spalvą per veiklos XML failo gaires (jas pridėjome) ir pakeitėme patį tekstą.



3. Įdėjome naują teksto objektą kuris prašo vartotojo įvesti savo vardą. Per gaires pakeitėme teksto spalvą į mėlyną.

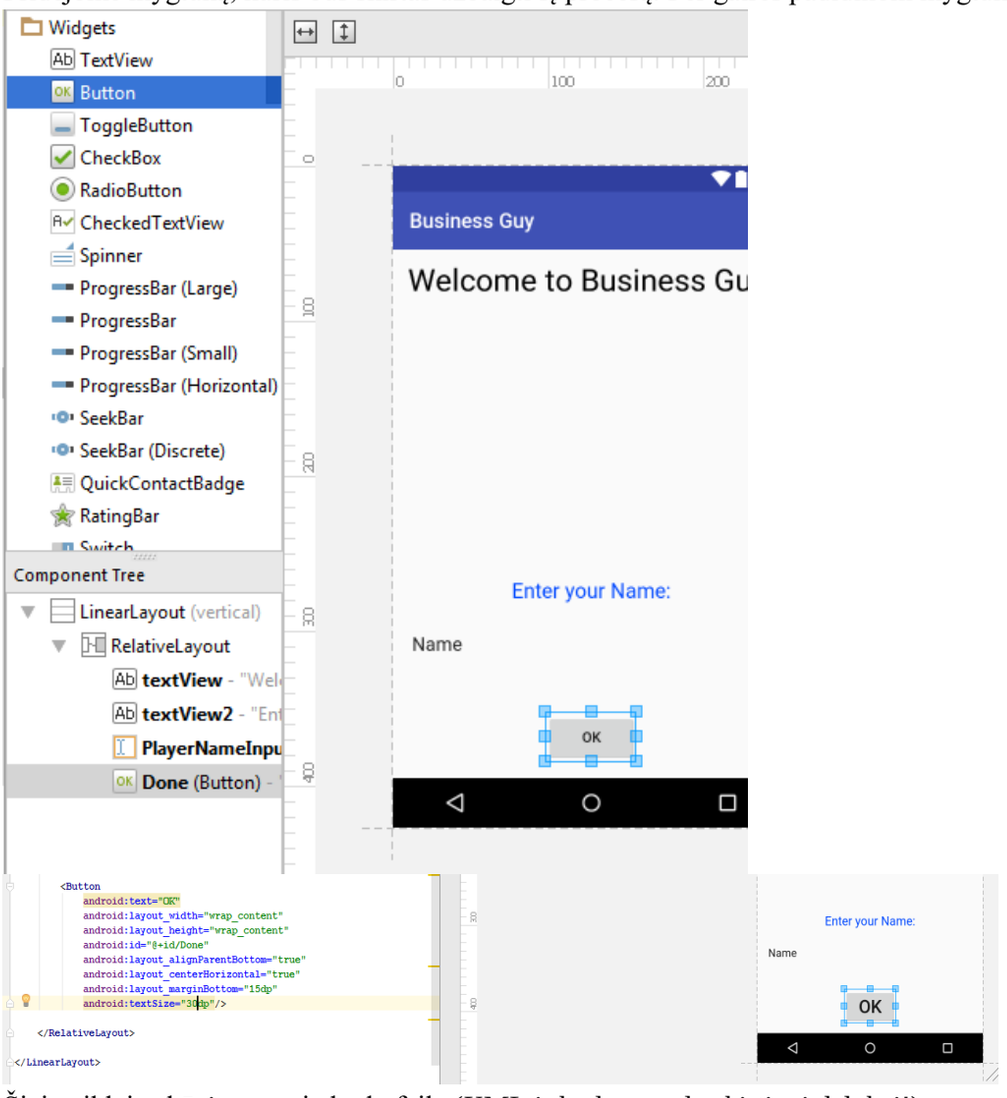


4. Iš paletės įdėjome naują teksto laukelį, į kurį įvedamas tekstas. Jis bus skirtas vardo įvedimui. Jo ID pakeitėme į „PlayerNameInput“ kad patogiai jį rastume kode.

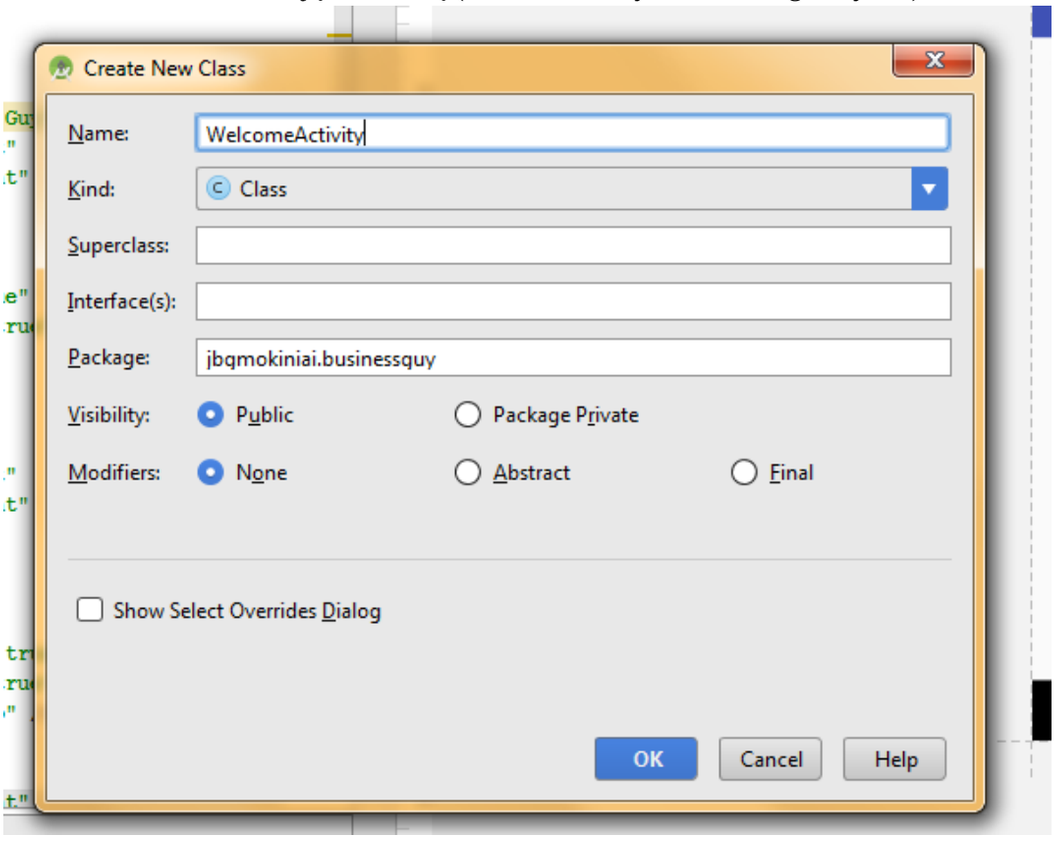



```
    <EditText
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:text="Name"
        android:ems="10"
        android:layout_marginTop="43dp"
        android:id="@+id/PlayerNameInput"
        android:layout_width="1000dp"
        android:layout_alignTop="@+id/textView2"
        android:layout_alignLeft="@+id/textView"
        android:layout_alignStart="@+id/textView"
        android:layout_alignRight="@+id/textView"
        android:layout_alignEnd="@+id/textView" />

</RelativeLayout>
```

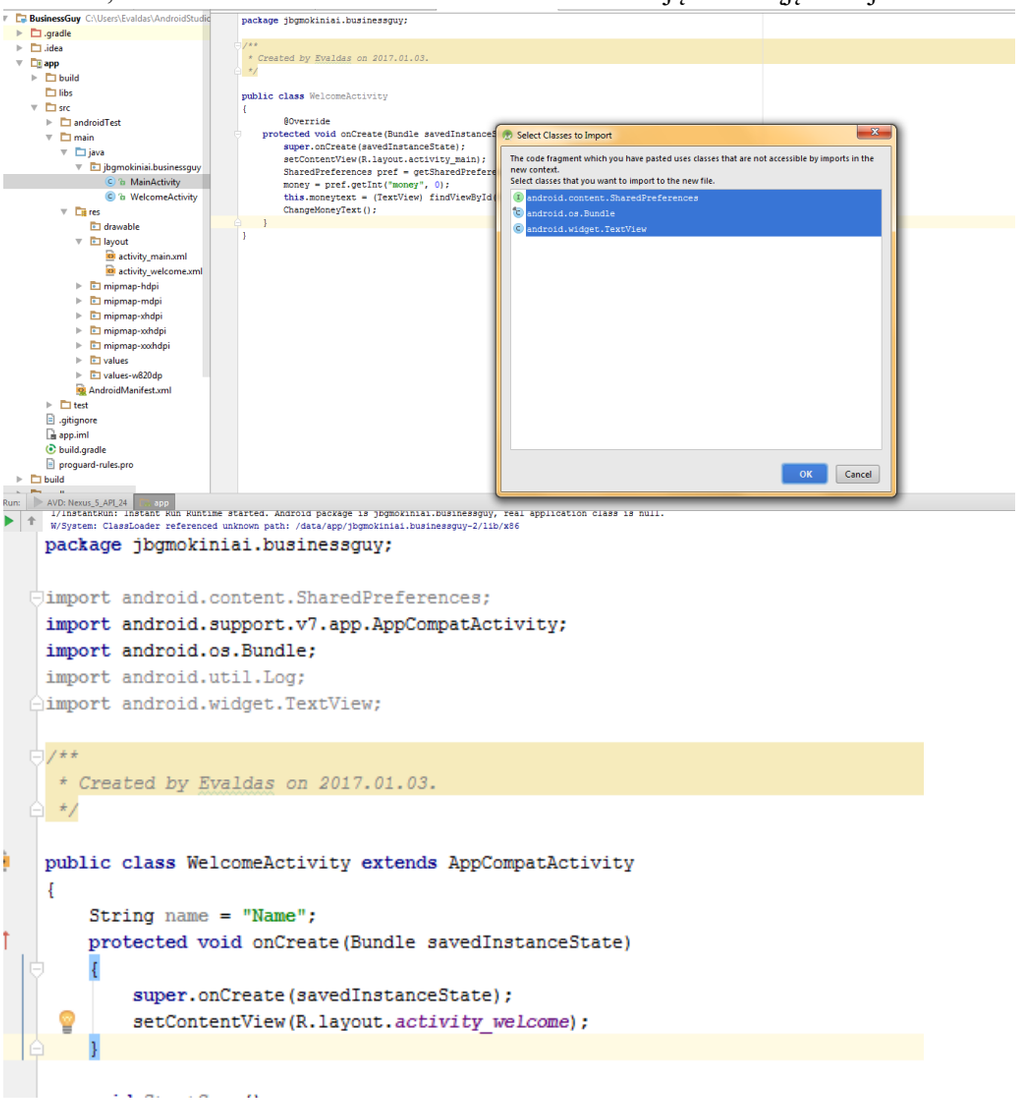
5. Pridėjome mygtuką, kuris bus skirtas užbaigti šį procesą. Per gaires padidinom mygtuko teksto dydį.



6. Šiai veiklai sukūrėme naują kodo failą (XML ir kodas yra du skirtingi dalykai!).

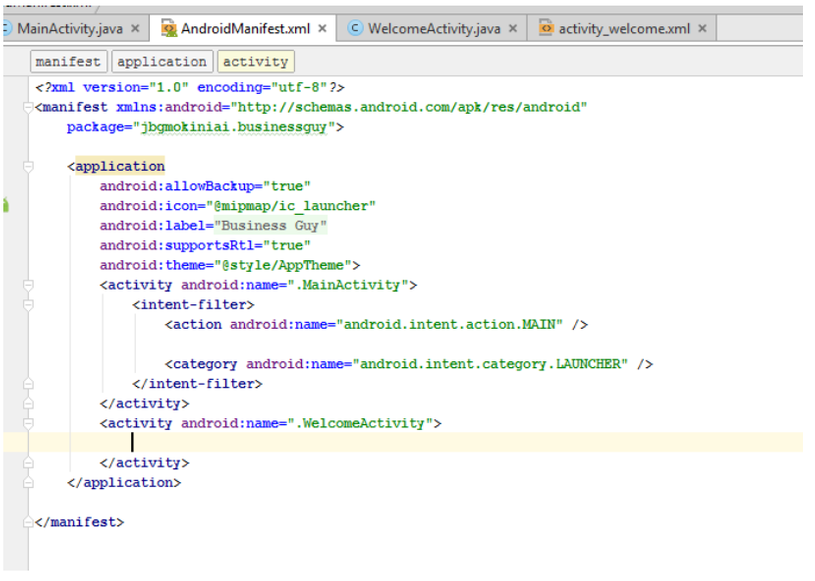


7. Įkopijavome kodą iš pagrindinės veiklos, buvome pasitikti su langu, kad įkeltume Imports (jų reikia, kad veiktų tam tikros funkcijos, būtų galima dirbti su tam tikrais kintamųjų tipais, t.t.). Kodą išvalėme, nes mums reiks visiškai kitokio. Sukūrėme naują kintamąjį žaidėjo vardui.




```
package jbgmokiniai.businessguy;

import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

/**
 * Created by Evaldas on 2017.01.03.
 */

public class WelcomeActivity extends AppCompatActivity
{
    String name = "Name";
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_welcome);
    }
```

8. Į AndroidManifest.XML įrašėme šią naują veikla. To reikia, kad programa žinotų, kad šioje programoje egzistuoja ši veikla.

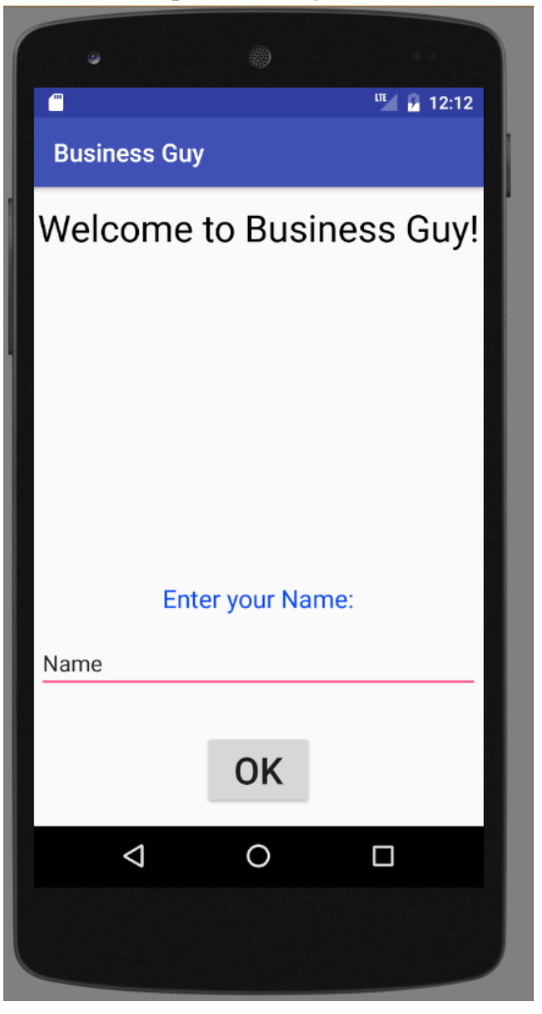



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="jbgmokiniai.businessguy">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Business Guy"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".WelcomeActivity">

        </activity>
    </application>

</manifest>
```

9. Sugrįžome į pagrindinės veiklos kodą. Jame perkodavome veiklą taip, kad iš vartotojo duomenų failo būtų patikrinama, ar žaidimas pradėtas, ar ne. Jeigu nepradėtas, nusiunčiama į pradinį programos ekraną. Jei pradėtas, tai tęsiamas pagr. veiklos darbas.

```
import android.content.Intent;

import org.w3c.dom.Text;

public class MainActivity extends AppCompatActivity
{
    public int money = 0;
    private TextView moneytext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        boolean start = pref.getBoolean("started", false);

        if (start == false)
        {
            startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
        }
        else
        {
            money = pref.getInt("money", 0);
            this.moneytext = (TextView) findViewById(R.id.moneytext);
            ChangeMoneyText();
        }
    }
```

10. Išbandėme tai per emuliaciją. Buvome nusiųsti į pradinį veiklos ekraną.



Trečias darbas – Pradžios ekrano kodas

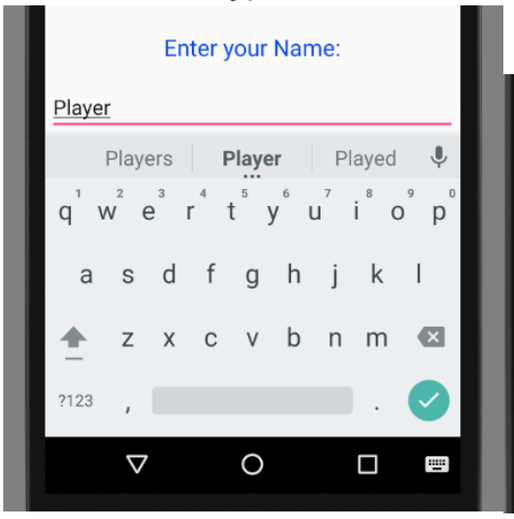
1. WelcomeActivity (pradžios ekrano veikloje) pridėjome naują kintamąjį namefield, iš kurio bus skaitoma įvestas vardas.

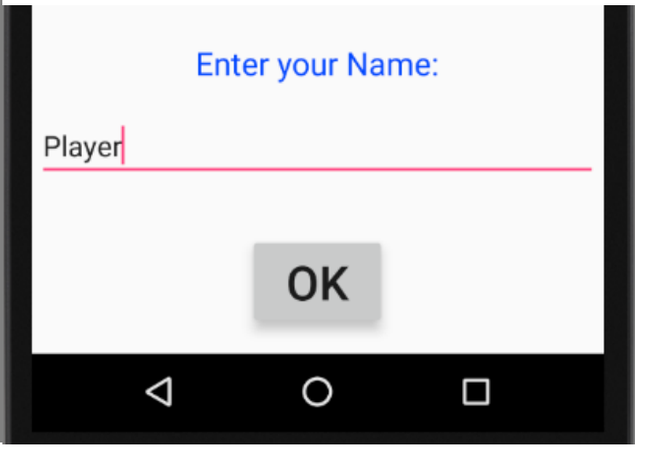
```
    private TextView namefield;

    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_welcome);
        this.namefield = (TextView) findViewById(R.id.PlayerNameInput);
    }
```

2. Prirašėme baigimo mygtukui kodo:
    a. Pirma pridėjome mygtuko „Done“ kintamąjį. Jį suradome per ID („Done“)
    b. Į veiklą prirašėme implenets View.OnClickListener, kuris leidžia naudoti mygtuko paspaudimo funkciją.
    c. Priskyrėme šiam mygtukui („Done“) OnClickListener, suteikiant jam funkciją paspaudus mygtuką.
    d. Funkcijoje OnClick parašėme kodą, kuris pakeičia bool started į true, kas reiškia, kad pradėjus žaidimą, programa aptiks, kad žaidimas pradėtas; pakeitėme žaidėjo varda duomenų faile į įvesta; programa pereina į kitą (pagr.) veiklą.

```
public class WelcomeActivity extends AppCompatActivity implements View.OnClickListener
{
    private TextView namefield;
    private Button Done;

    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_welcome);
        this.namefield = (TextView) findViewById(R.id.PlayerNameInput);
        this.Done = (Button) findViewById(R.id.Done);
        Done.setOnClickListener(this);
    }

    @Override
    public void onClick(View v)
    {
        SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
        prefeditor.putBoolean("started", true);
        prefeditor.putString("name", "" + namefield.getText());
        prefeditor.putInt("money", 1000);
        prefeditor.commit();
        startActivity(new Intent(WelcomeActivity.this, MainActivity.class));
    }
}
```

3. Paleidome emuliaciją ir išbandėme. Suvedėme vardą, paspaudėme „OK“ mygtuką.



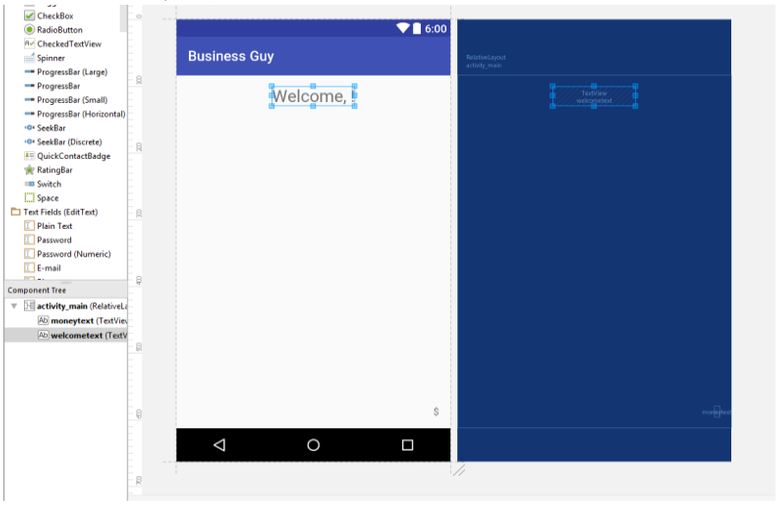


4. Buvome perkelti į kitą veiklą. Perkrovus programą, atsidūrėme nebe pradinėje veikloje, o pagrindinėje.

5. Prirašėme kodo pradinėje veikloje, kad po „OK“ mygtuko paspaudimo pinigai būtų pakeisti į $1000.

```
prefeditor.putInt("money", 1000);
```

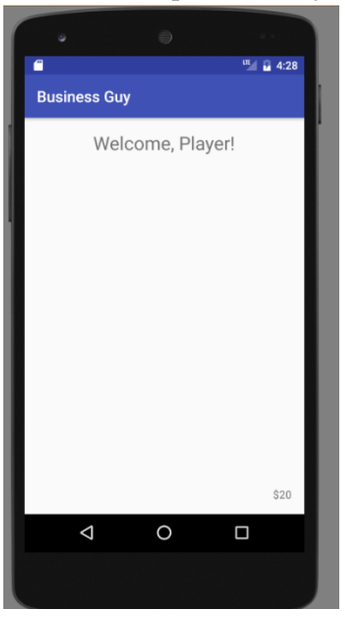
6. Į pagrindinę veiklą įdėjome naują teksto laukelį, kuriame yra pasisveikinimas su žaidėju. Pakeitėme teksto šrifto dydį į 24.



7. Pagr. veiklos kode prirašėme kodo, kad pakeistų tekstą į „Welcome, [žaidėjo vardas]!“. Vardas išgaunamas iš duomenų failo.

```
{
    public int money = 0;
    public String name = "";
    private TextView moneytext;
    private TextView welcometext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        boolean start = pref.getBoolean("started", false);

        if (start == false)
        {
            startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
        }
        else
        {
            money = pref.getInt("money", 0);
            name = pref.getString("name", "");
            this.moneytext = (TextView) findViewById(R.id.moneytext);
            this.welcometext = (TextView) findViewById(R.id.welcometext);
            welcometext.setText("Welcome, " + name + "!");
            ChangeMoneyText();
        }
    }
```

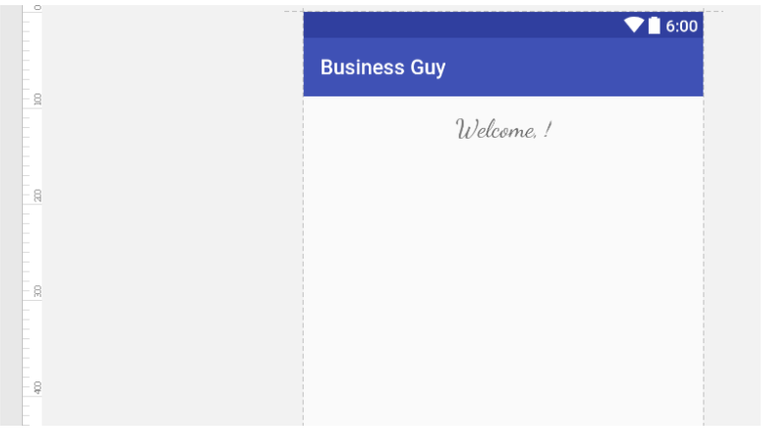
8. Išbandėme tai per emuliaciją.



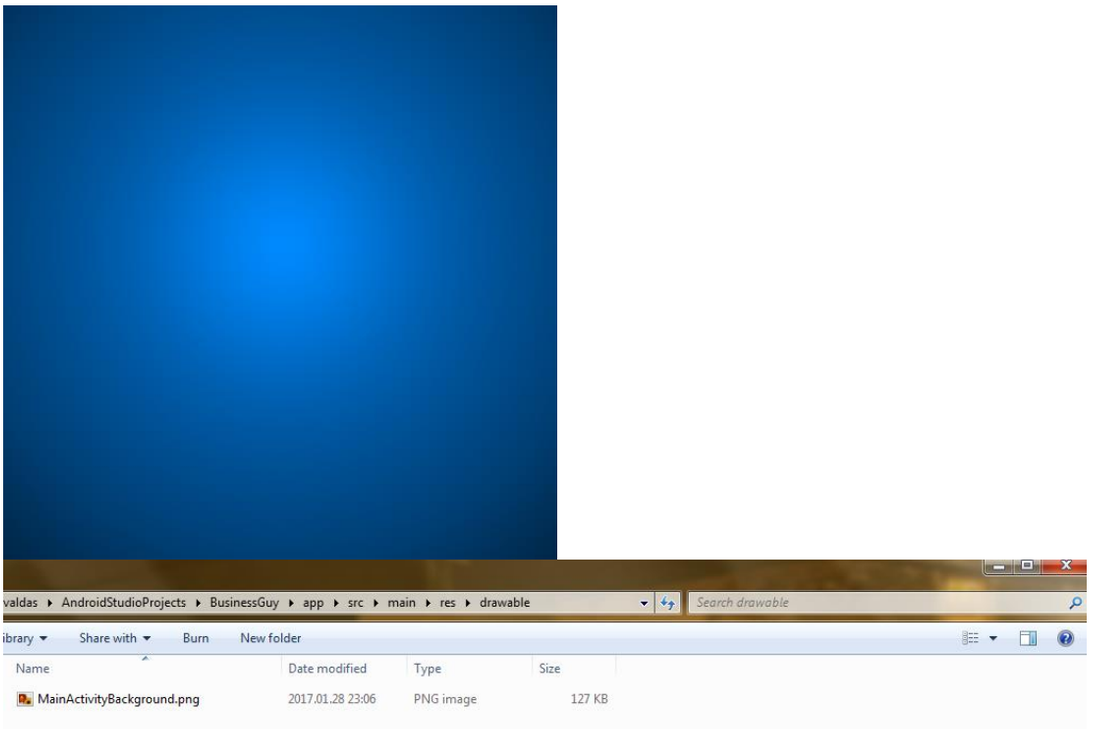
Ketvirtas Darbas – Pagražinimai

1. Pakeitėme pasisveikinimo teksto šrifto stilių į cursive (su gaire android:fontFamily=“cursive“).

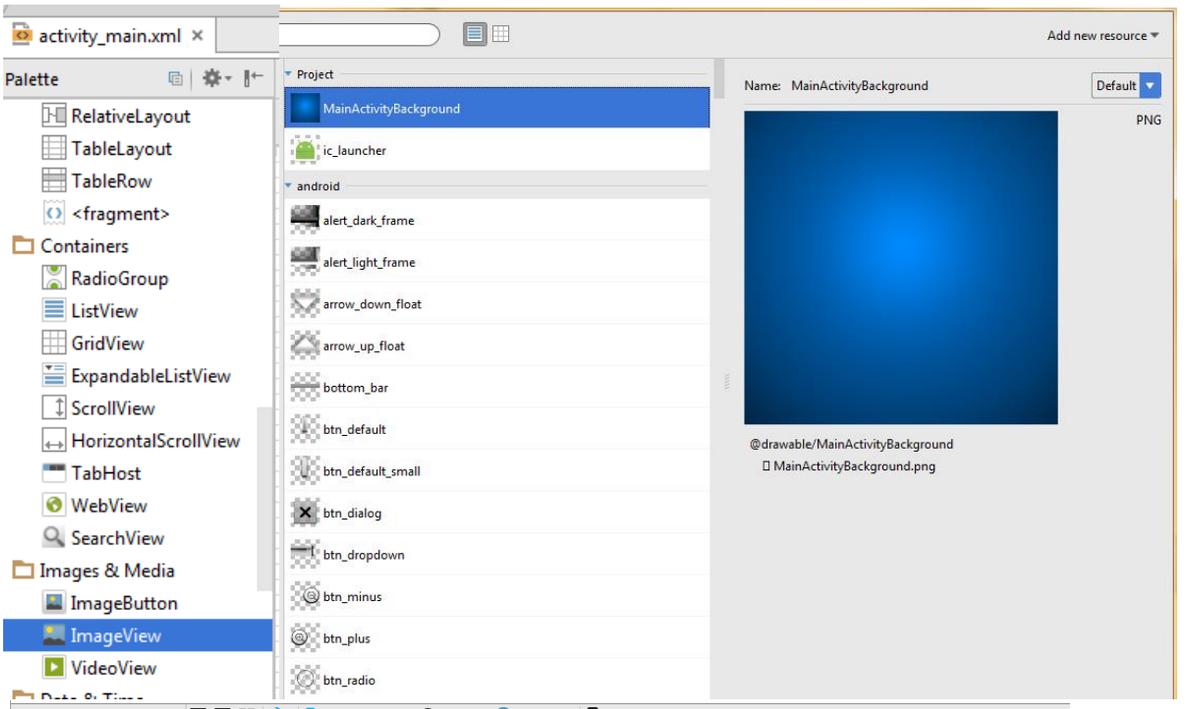
```
android:layout_height="match_parent"
android:paddingBottom="16dp"
android:paddingLeft="64dp"
android:paddingRight="64dp"
android:paddingTop="16dp"
tools:context="jbgmokiniai.businessguy.MainActivity">

<TextView
    android:text="$"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/moneytext"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />

<TextView
    android:text="Welcome, !"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/welcometext"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:textSize="24sp"
    android:fontFamily="cursive"/>
```

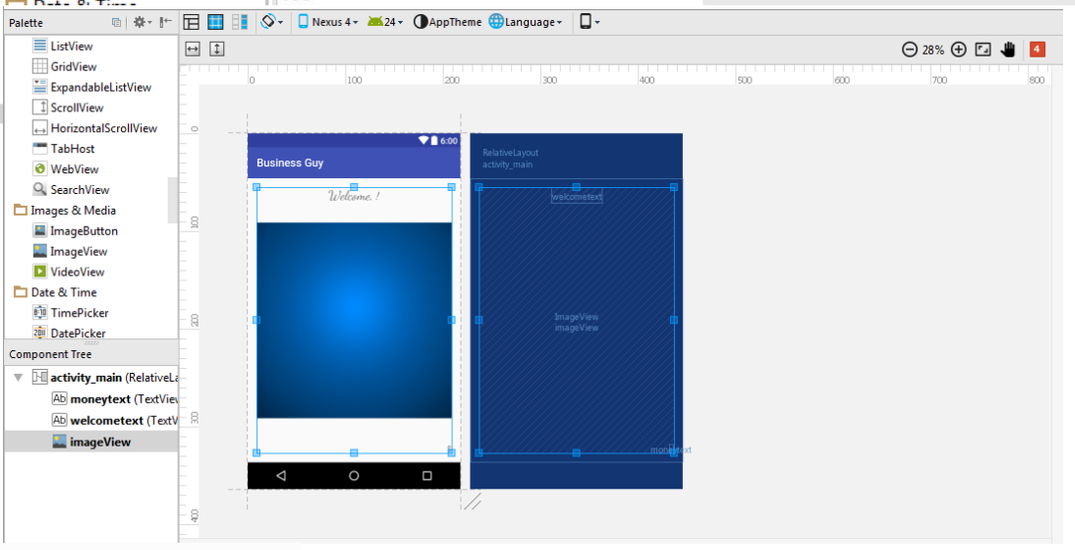


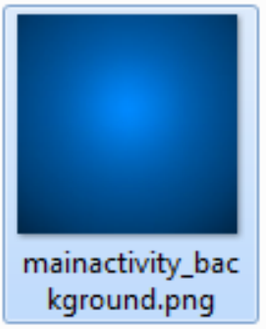
2. Sukūrėme paprastą fono paveikslėlį. Jį įdėjome į res/drawable (čia bus dedami visi programos vaizdiniai resursai)



3. Iš paletės į veiklą įdėjome ImageView objektą. Jam priskyrėme šio įkelto paveikslėlio resursą. Pervadinome resursą visomis mažosiomis raidėmis, nes kitu atveju, programa nepasileis (didžiosiomis raidėmis failai negali būti)

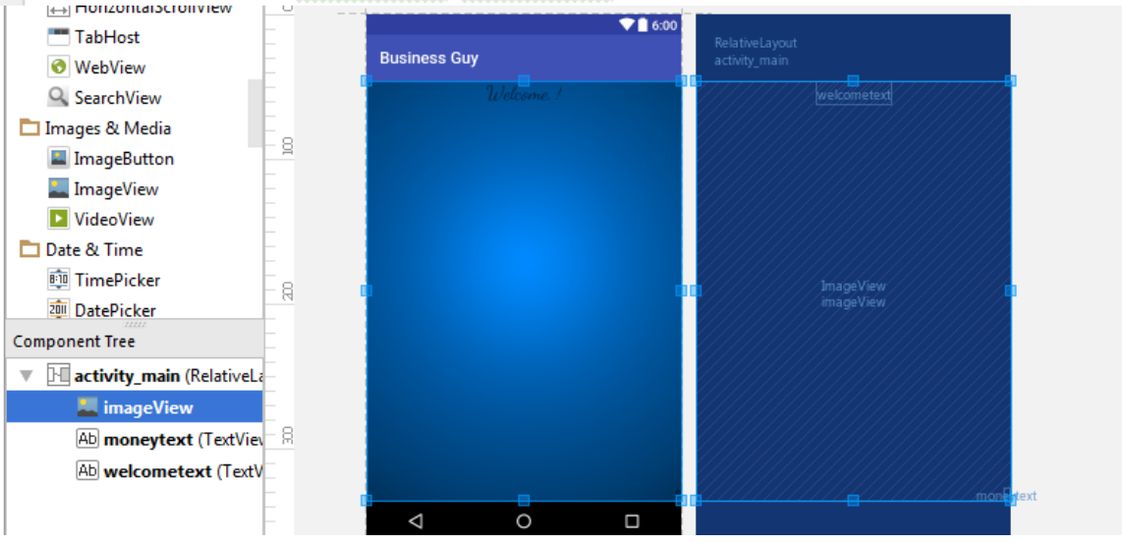




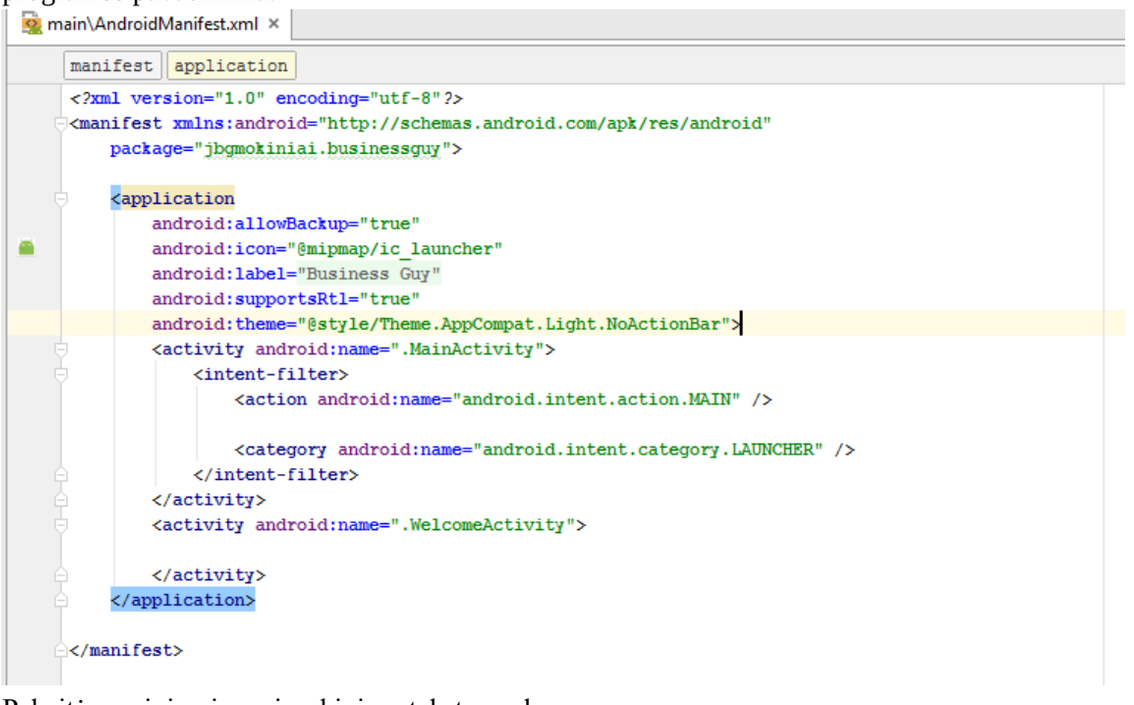


4. Gairėse pakeitėme paveikslėlio ilgį į plotį į maksimalų (kad atitiktų skilties, kurioje jis yra didį) ir prirašėme gairę android:scaleType="centerCrop" kad paveikslėlis būtų centre ir būtų pilnai padidintas. Pagrindinės veiklos gairėse ištrynėme *padding*, kad paveikslėlis būtų per visą ekraną.

```
<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:srcCompat="@drawable/mainactivity_background"
    android:layout_centerVertical="true"
    android:layout_centerHorizontal="true"
    android:id="@+id/imageView"
    android:scaleType="centerCrop"/>
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="jbgmokiniai.businessguy.MainActivity">
```



5. AndroidManifest gairėse pakeitėme programos temą, kad programos viršuje nerodytų juostos su programos pavadinimu.



6. Pakeitėme pinigų ir pasisvekinimo tekstų spalvas.

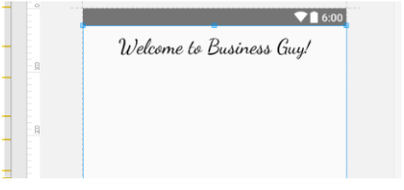
```
<TextView
    android:text="$"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/moneytext"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:textColor="#00ff08"/>

<TextView
    android:text="Welcome, !"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/welcometext"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:textSize="24sp"
    android:textColor="#FFFFFF"
    android:fontFamily="cursive"/>
```

7. Dabar keisime pradinės veiklos išvaizdą, taigi, kadangi jau esame pradėję žaidimą, prirašėme funkciją, kad išvalytų mūsų duomenis, taigi būsime perkelti į pradinę veiklą.

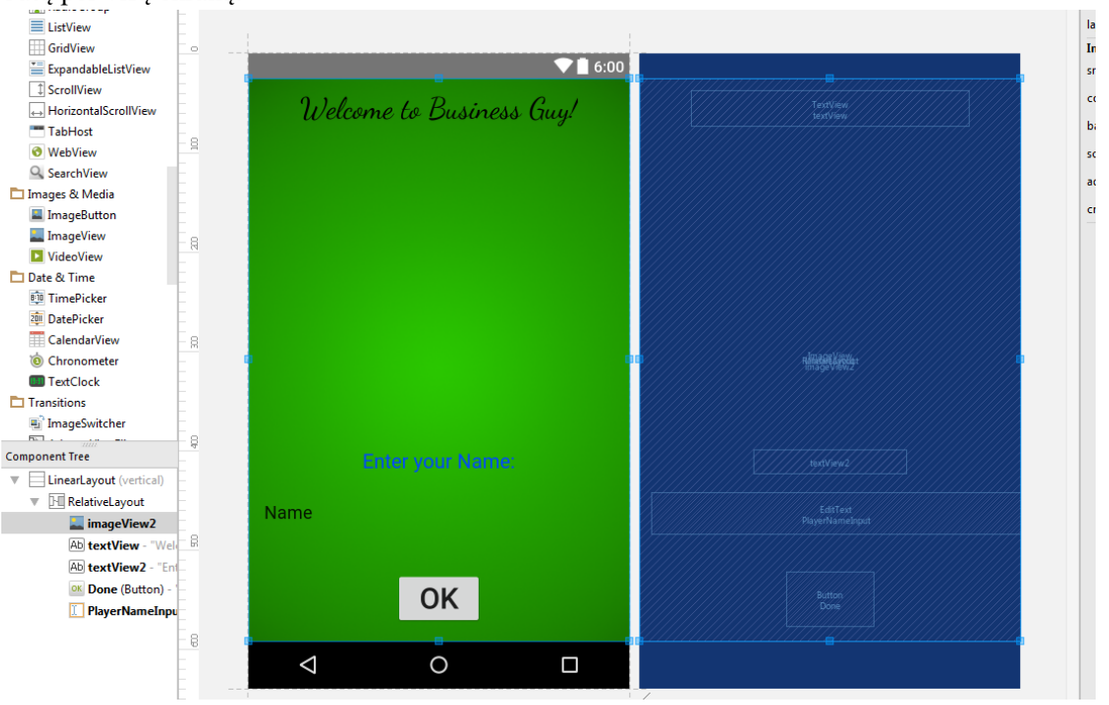
```
    WipeData();
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

    boolean start = pref.getBoolean("started", false);

    if (start == false)
    {
        startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
    }
    else
    {
        money = pref.getInt("money", 0);
        name = pref.getString("name", "");
        this.moneytext = (TextView) findViewById(R.id.moneytext);
        this.welcometext = (TextView) findViewById(R.id.welcometext);
        welcometext.setText("Welcome, " + name + "!");
        ChangeMoneyText();
    }
}

public void ChangeMoneyText() { moneytext.setText("$" + money); }
public void ChangeMoney(int amount)
{
    money += amount;
    SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
    prefeditor.putInt("money", money);
    prefeditor.commit();
}

void WipeData()
{
    SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
    prefeditor.clear();
    prefeditor.commit();
}
```

8. Pradinėje veikloje pakeitėme pasisveikinimo teksto šrifto stilių.

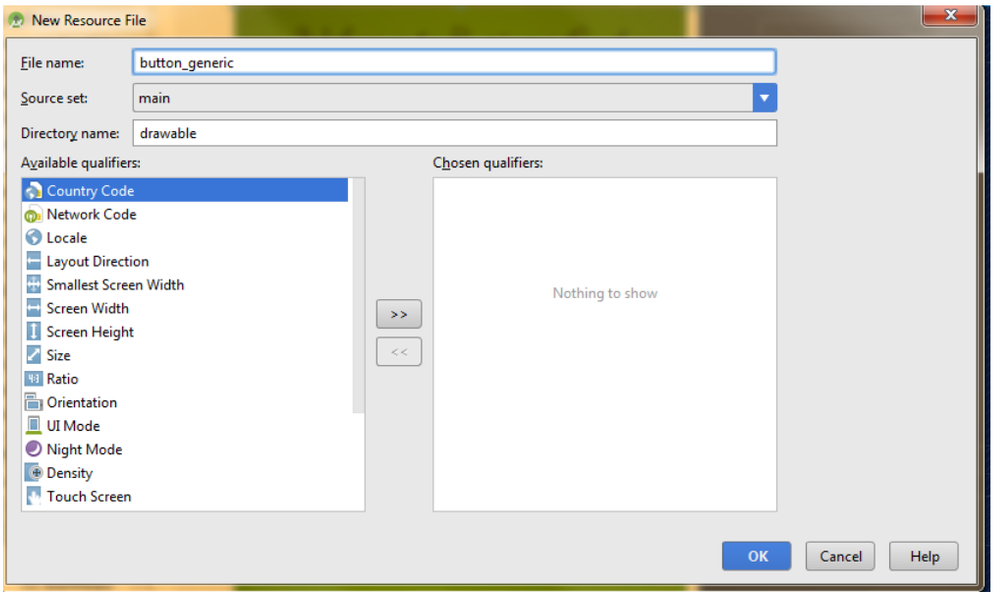
```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:text="Welcome to Business Guy!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:textSize="30dp"
        android:layout_marginTop="12dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textColor="#000000"
        android:fontFamily="cursive"/>
```



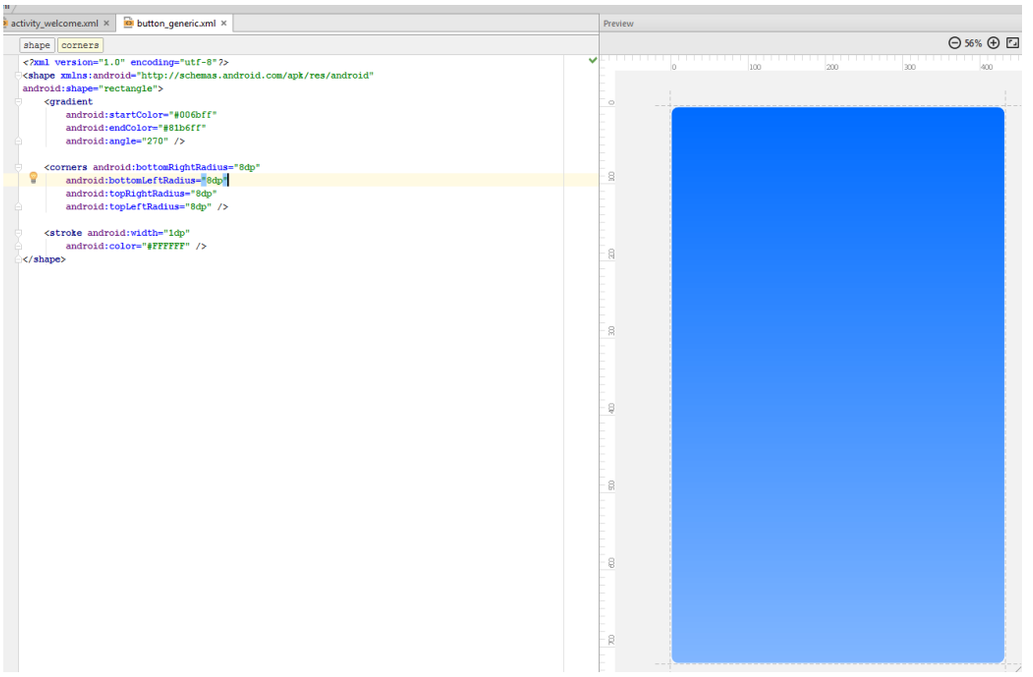
9. Taip pat kaip su praeitu fonu, pridėjome naują fono resursų failą bei pakoregavome gaires, kad jis būtų per visą ekraną.



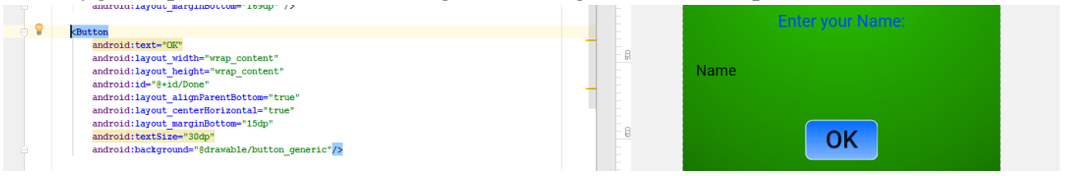
10. Sukūrėme naują resursų failą mygtukui.



11. Gairių failą papildėme gairėmis, kad:
    a. Būtų gradient – t.y. spalvos perėjimo stilius, tai bus mygtuko fonas.
    b. Mygtukas būtų suapvalintas.
    c. Mygtukas tūrėtų apvedžiojimą.



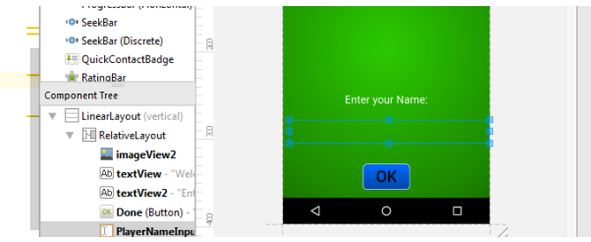
12. Prie mygtuko prirašėme android:background = .. gairę, kad būtų priskirtas šis resursų failas.



13. Galutiniame variante šiek tiek pakeitėme spalvas.

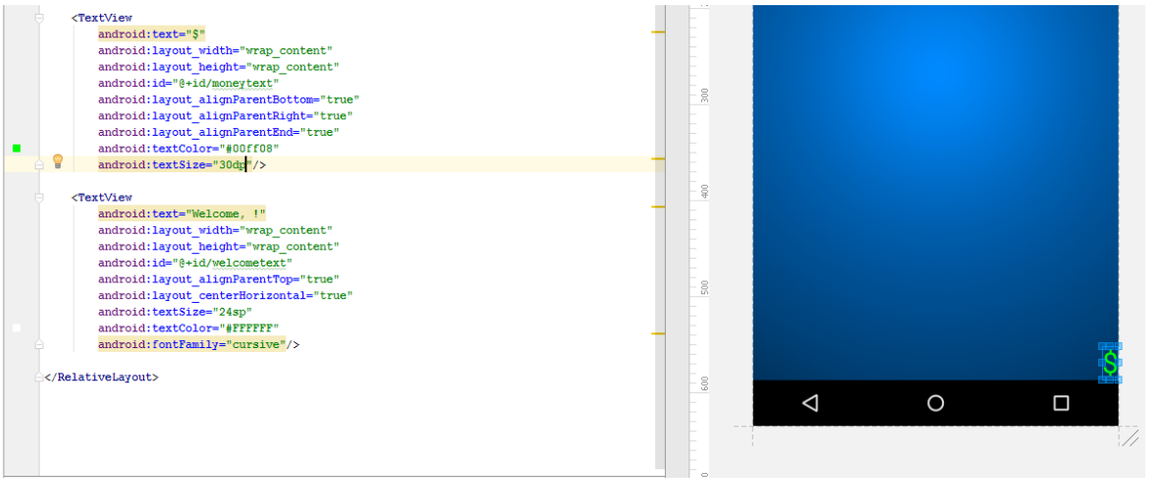
14. Žaidėjo teksto vardo įvedimo laukelyje ištrynėme pradinį tekstą dėl patogumo.

```
    <EditText
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:ems="10"
        android:id="@+id/PlayerNameInput"
        android:layout_width="1000dp"
        android:layout_above="@+id/Done"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginLeft="12dp"
        android:layout_marginStart="12dp"
        android:layout_marginBottom="37dp" />

</RelativeLayout>
```



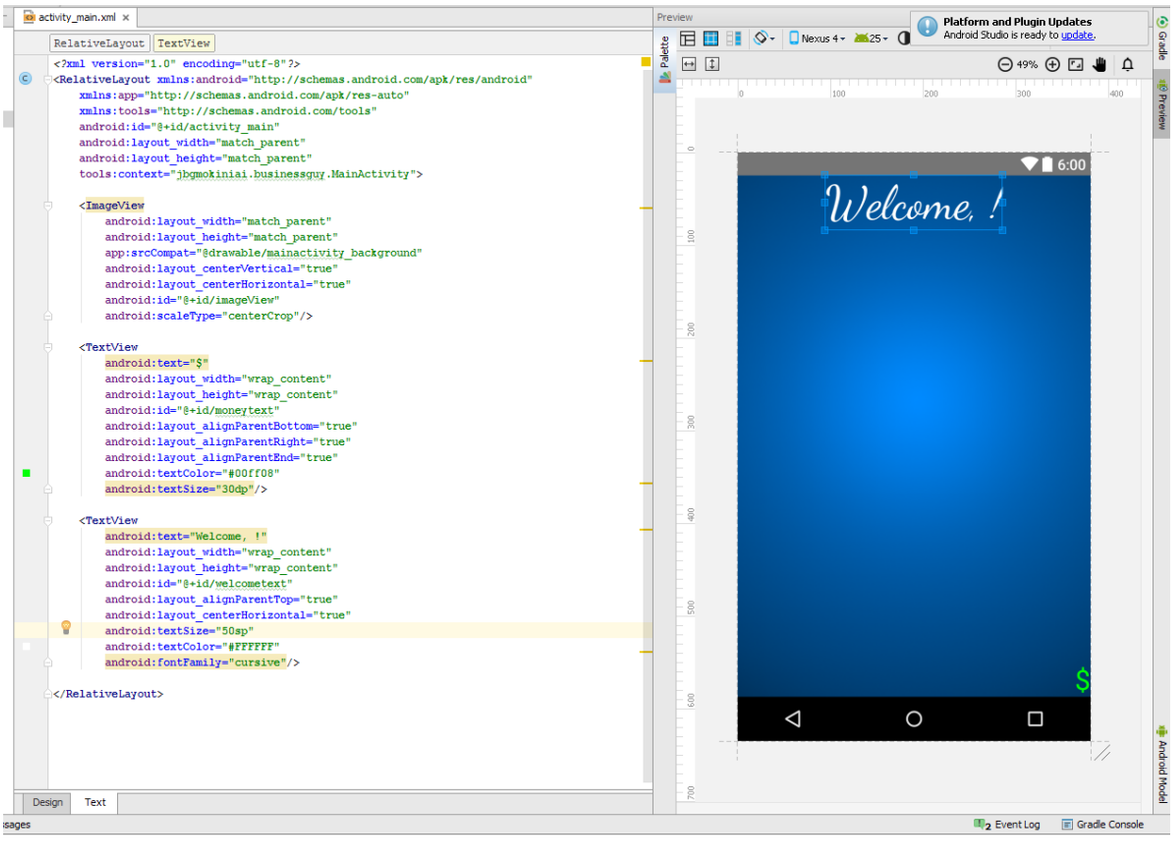
Penktas Darbas – Pagrindinės veiklos išdėstymas

1. Pakeitėm pinigų šrifto dydį.



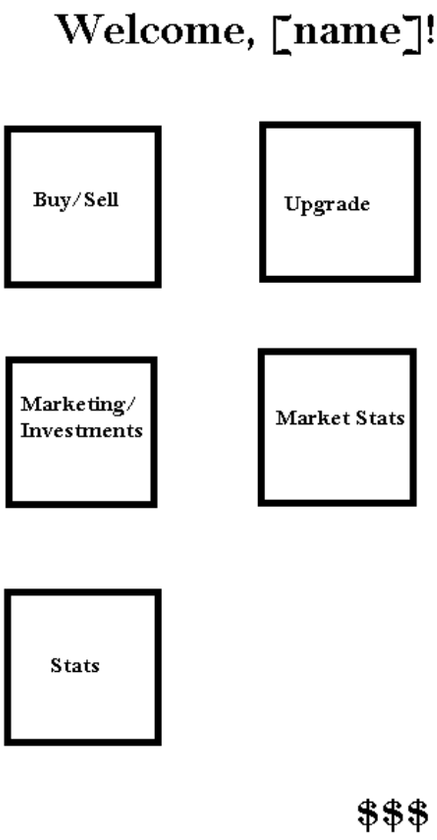
```
    <TextView
        android:text="$"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/moneytext"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#00ff08"
        android:textSize="30dp"/>

    <TextView
        android:text="Welcome, !"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/welcometext"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textSize="24sp"
        android:textColor="#FFFFFF"
        android:fontFamily="cursive"/>

</RelativeLayout>
```

2. Pakeitėm pasisveikinimo teksto dydį.



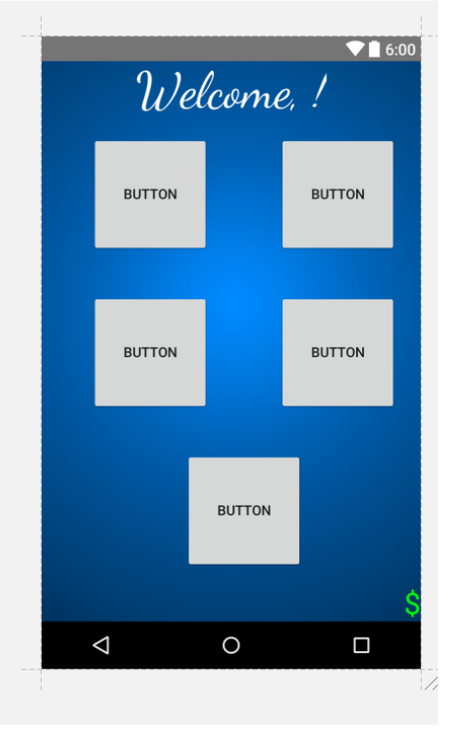
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="jbgmokiniai.businessguy.MainActivity">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:srcCompat="@drawable/mainactivity_background"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/imageView"
        android:scaleType="centerCrop"/>

    <TextView
        android:text="$"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/moneytext"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#00ff08"
        android:textSize="30dp"/>

    <TextView
        android:text="Welcome, !"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/welcometext"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textSize="50sp"
        android:textColor="#FFFFFF"
        android:fontFamily="cursive"/>

</RelativeLayout>
```

3. Per programą Paint pasidarėme mūsų išdėstymo šabloną, pagal kurį dirbsime.

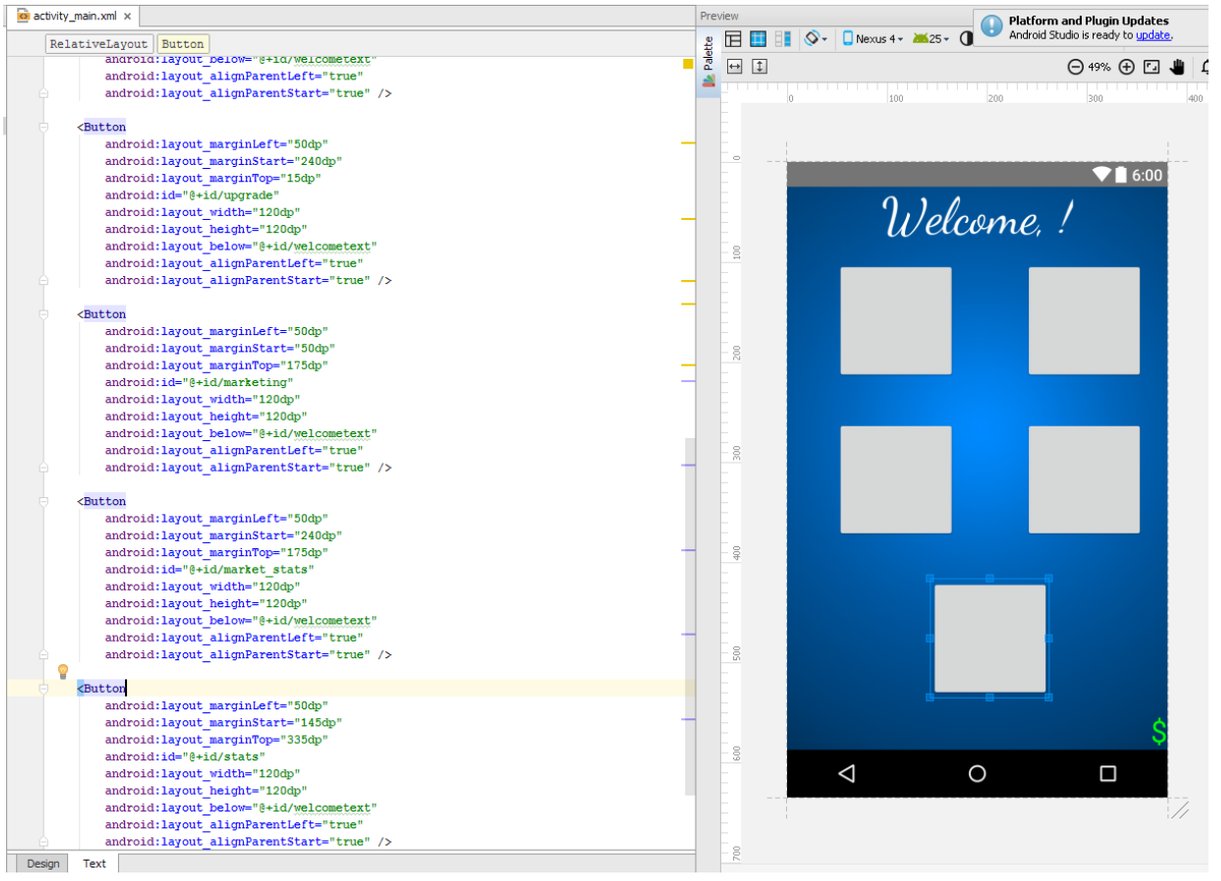


4. Pridėjome 5 identiškus mygtukus, juos išdėsteme remiantis šablonu.

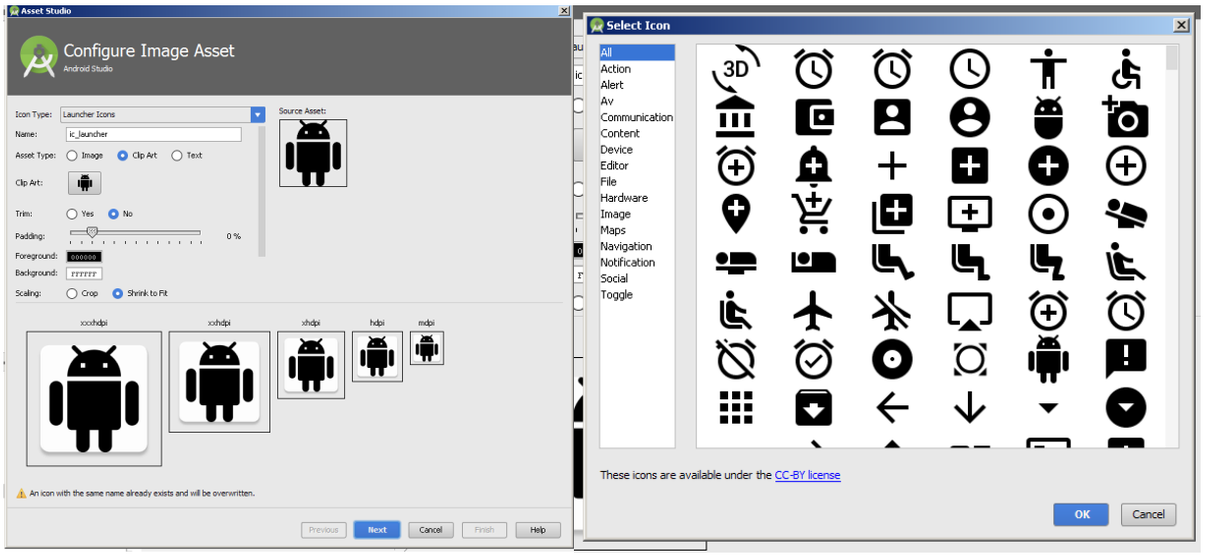
```
<Button
    android:text="Button"
    android:layout_marginLeft="50dp"
    android:layout_marginStart="50dp"
    android:layout_marginTop="15dp"
    android:id="@+id/buy_sell"
    android:layout_width="120dp"
    android:layout_height="120dp"
    android:layout_below="@+id/welcometext"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<Button
    android:text="Button"
    android:layout_marginLeft="50dp"
    android:layout_marginStart="240dp"
    android:layout_marginTop="15dp"
    android:id="@+id/upgrade"
    android:layout_width="120dp"
    android:layout_height="120dp"
    android:layout_below="@+id/welcometext"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<Button
    android:text="Button"
    android:layout_marginLeft="50dp"
    android:layout_marginStart="50dp"
    android:layout_marginTop="175dp"
    android:id="@+id/marketing"
    android:layout_width="120dp"
    android:layout_height="120dp"
    android:layout_below="@+id/welcometext"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<Button
    android:text="Button"
    android:layout_marginLeft="50dp"
    android:layout_marginStart="240dp"
    android:layout_marginTop="175dp"
    android:id="@+id/market_stats"
```

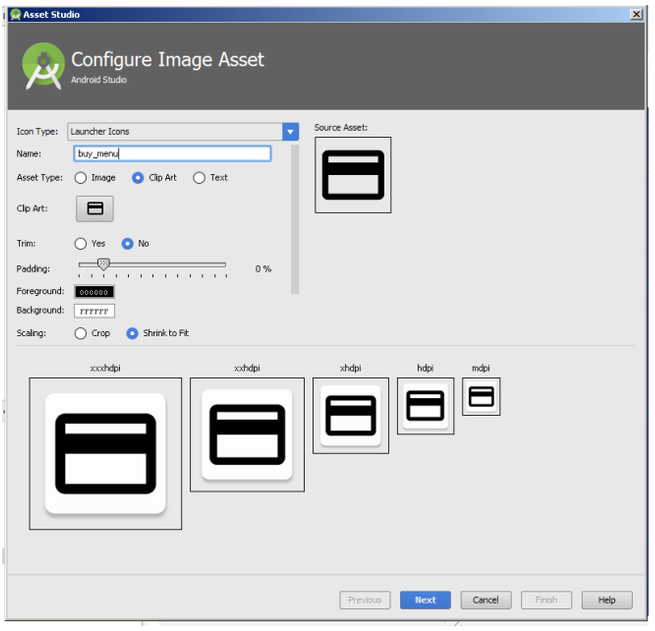


5. Ištrynėme mygtukų viduje esantį tekstą.

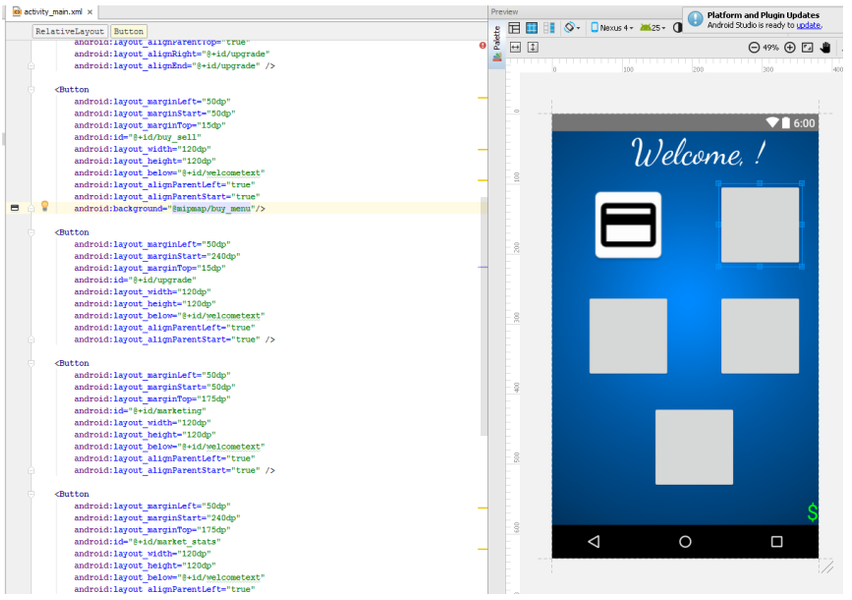


6. Sukūrėme 5 ikonų resursus, kurie bus skirti mygtukam. (Kuriant resursą, yra galimybė pasirinkti iš programoje esančių ikonų, tai ir padarėme)



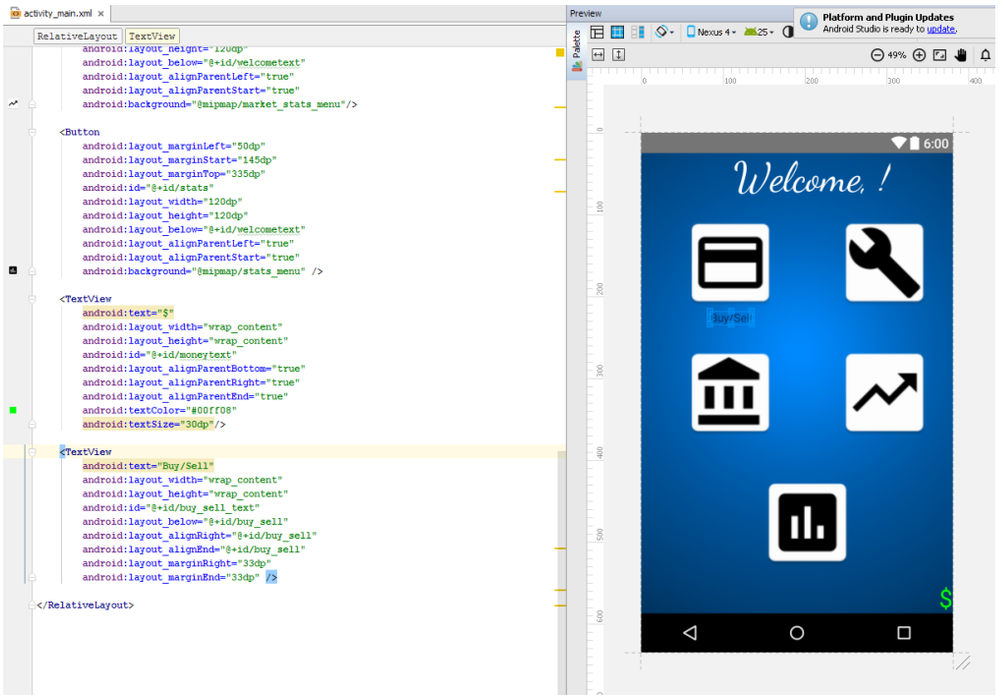


7. Šį resursą priskyrėme mygtukui su gaire android:background.



8. Tai padarėme su visais esančiais mygtukais.

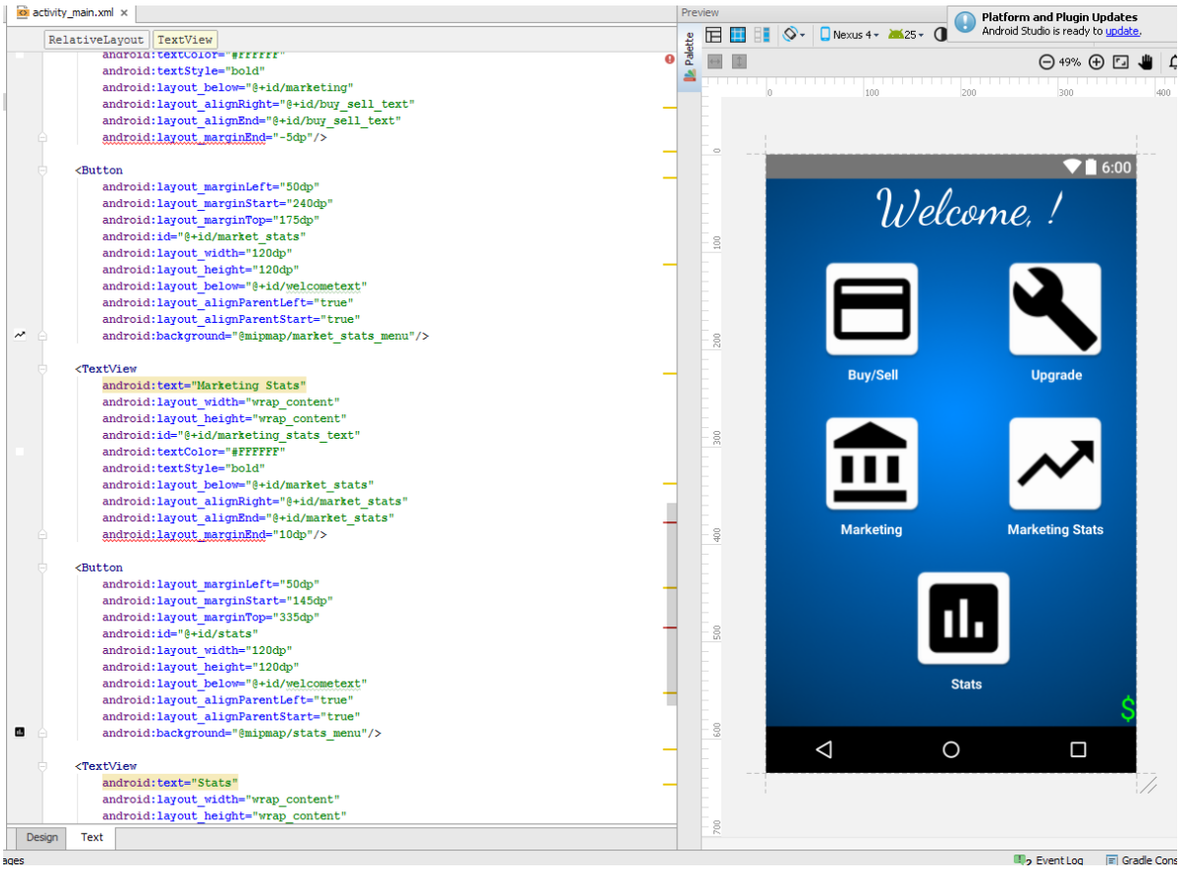
9. Pridėjome teksto objektą po mygtuku. Atitinkamai šį tekstą pakeitėme.



10. Pakeitėme spalvą į baltą ir nustatėme teksto stilių į bold (paryškintą)

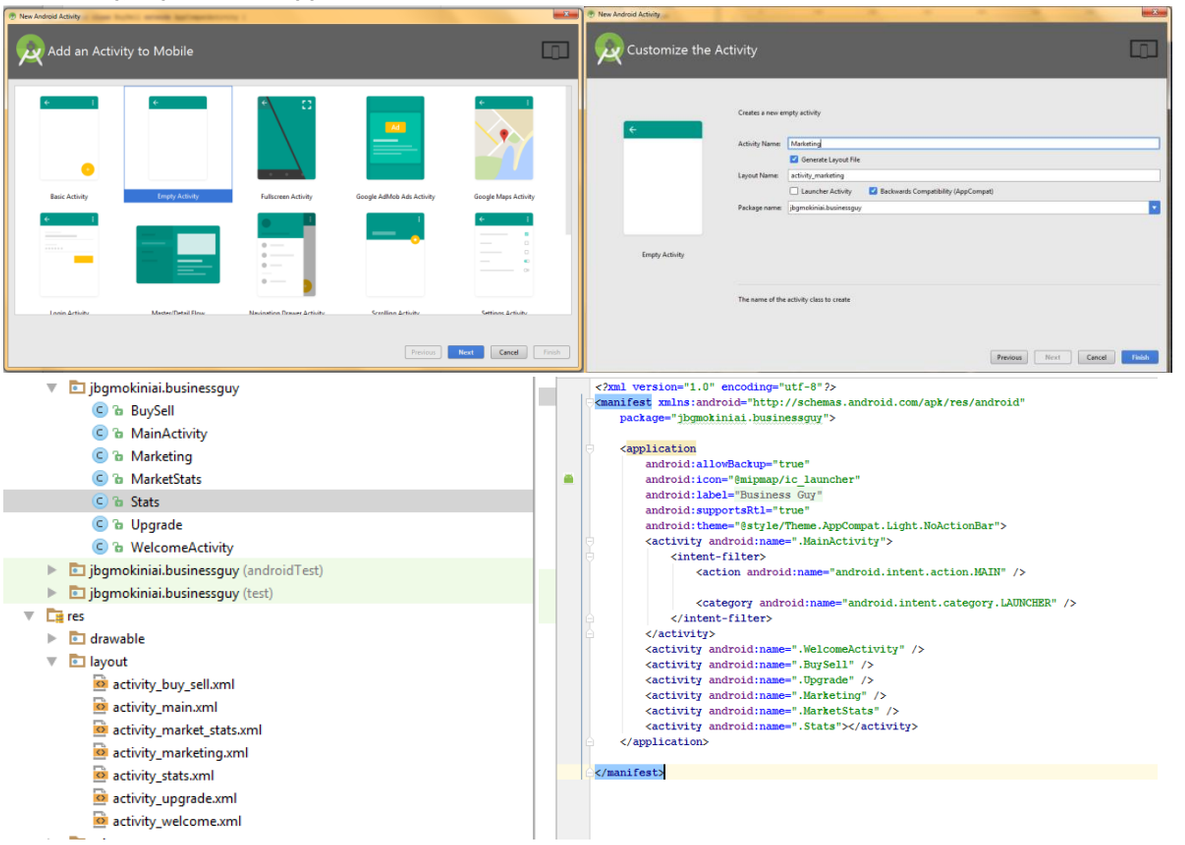
```
<Button
    android:layout_marginLeft="50dp"
    android:layout_marginStart="145dp"
    android:layout_marginTop="335dp"
    android:id="@+id/stats"
    android:layout_width="120dp"
    android:layout_height="120dp"
    android:layout_below="@+id/welcometext"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:background="@mipmap/stats_menu" />

<TextView
    android:text="$"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/moneytext"
    android:layout_alignParentBottom="true"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:textColor="#00ff08"
    android:textSize="30dp"/>

<TextView
    android:text="Buy/Sell"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/buy_sell_text"
    android:layout_below="@+id/buy_sell"
    android:layout_alignRight="@+id/buy_sell"
    android:layout_alignEnd="@+id/buy_sell"
    android:layout_marginRight="33dp"
    android:layout_marginEnd="33dp"
    android:textColor="#FFFFFF"
    android:textStyle="bold"/>
```

11. Tai padarėme su visais mygtukais.

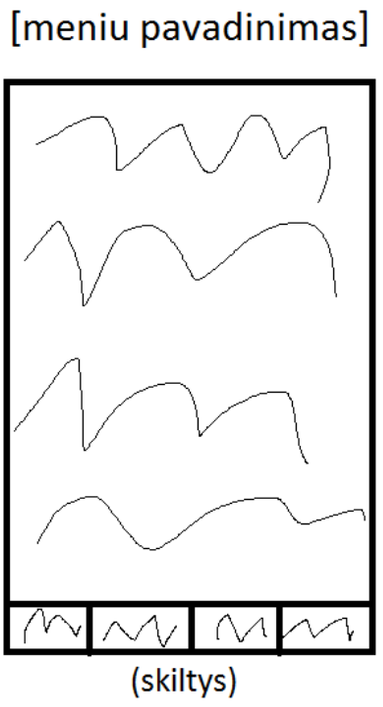


Šeštas Darbas – Kitos veiklos, pagražinimai

1. Sukūrėme veiklas kiekvienai skilčiai (Buy/Sell, Upgrade, Marketing, Marketing Stats ir Stats). Šį kart tai darėme kitu būdu – iškarto sukūrėme veiklą, kas sukuria ir veiklos išdėstymo gairių failą, ir kodo failą, ir įkelia veiklą į AndroidManifest.XML.



2. Meniu reikės išdėstymo, taigi šį darbą pasiskirstėme. Sugalvojome bendrą šabloną, pagal kurį kursime kitus meniu šablonus. Bet prieš tai reikia sukurti veiklų perėjimo funkciją.

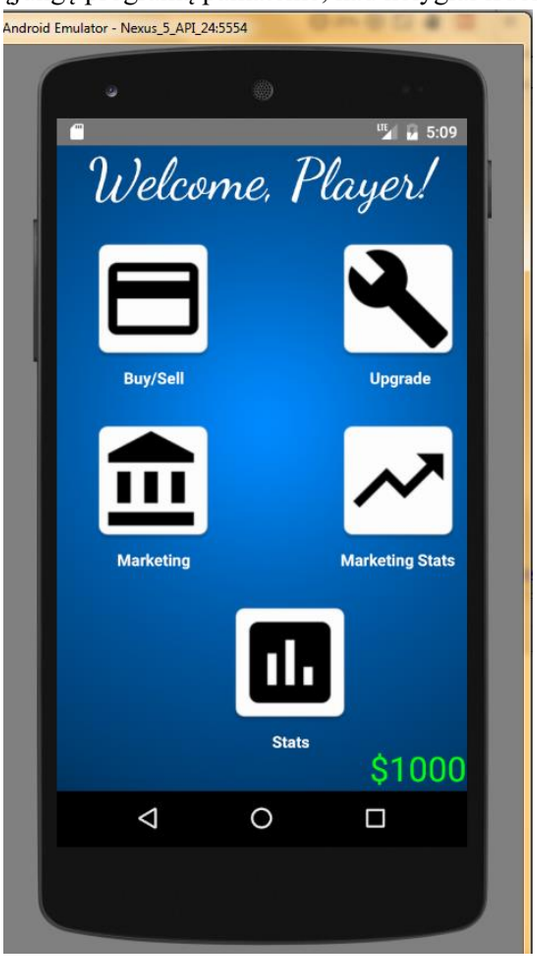


3. Pagrindinės veiklos kode prirašėme kodo, kad galėtume nueiti į kitas veiklas:
    a. Iš pradžių sukūrėme masyvus (sąrašus) mygtukam saugoti ir jų ID saugoti;
    b. Prasidėjus pagrindinei veiklai, surandami šie mygtukai pagal ID masyvą ciklo pagalba;
    c. Pridėjome OnClick funkcionalumą į pagrindinę veiklą, jame parašėme kodo, kad būtų atskiriama, koks mygtuka paspaustas. Šis kodas perkelia mus į kitą veiklą pagal tai, koks mygtukas buvo paspaustas.

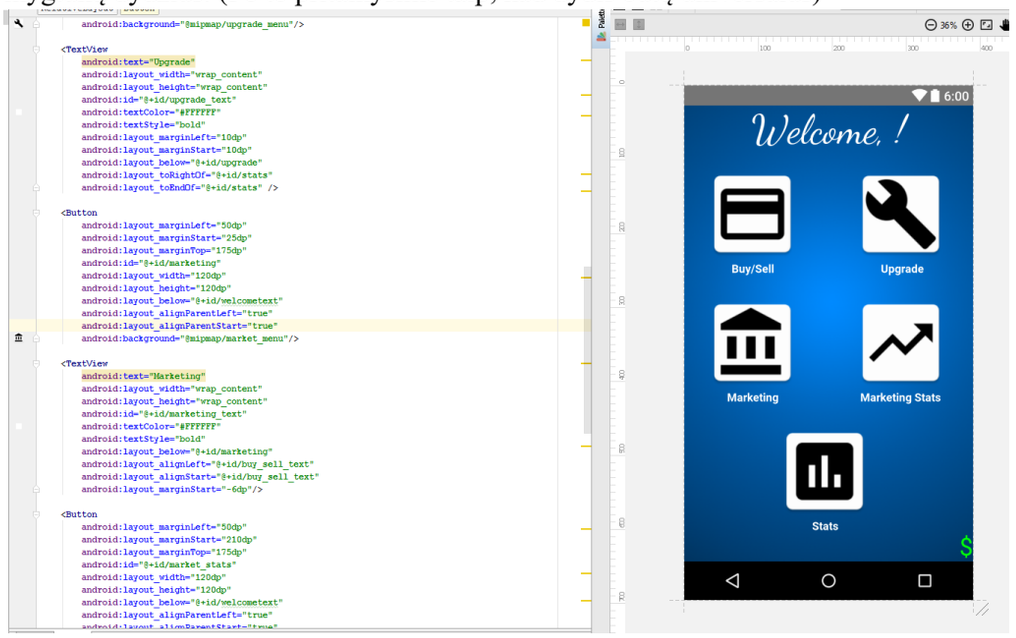
```
public class MainActivity extends AppCompatActivity implements View.OnClickListener
{
    public int money = 0;
    public String name = "";
    private TextView moneytext;
    private TextView welcometext;
    private Button menus[] = new Button[5];
    private int[] menu_ids = { R.id.buy_sell, R.id.upgrade, R.id.marketing, R.id.market_stats, R.id.stats };

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        boolean start = pref.getBoolean("started", false);

        if (start == false)
        {
            startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
        }
        else
        {
            money = pref.getInt("money", 0);
            name = pref.getString("name", "");
            this.moneytext = (TextView) findViewById(R.id.moneytext);
            this.welcometext = (TextView) findViewById(R.id.welcometext);
            welcometext.setText("Welcome, " + name + "!");
            ChangeMoneyText();

            for (int index = 0; index < 5; ++index)
            {
                menus[index] = (Button)findViewById(menu_ids[index]);
                menus[index].setOnClickListener(this);
            }

        }
    }
```

```
    @Override
    public void onClick(View v)
    {
        switch (v.getId())
        {
            case R.id.buy_sell:
                startActivity(new Intent(MainActivity.this, BuySell.class));
                break;
            case R.id.upgrade:
                startActivity(new Intent(MainActivity.this, Upgrade.class));
                break;
            case R.id.marketing:
                startActivity(new Intent(MainActivity.this, Marketing.class));
                break;
            case R.id.market_stats:
                startActivity(new Intent(MainActivity.this, MarketStats.class));
                break;
            case R.id.stats:
                startActivity(new Intent(MainActivity.this, Stats.class));
                break;

            default:
                break;
        }

    }
```

4. Įjungę programą pamatėme, kad nelygiai išdėstyti mygtukai.

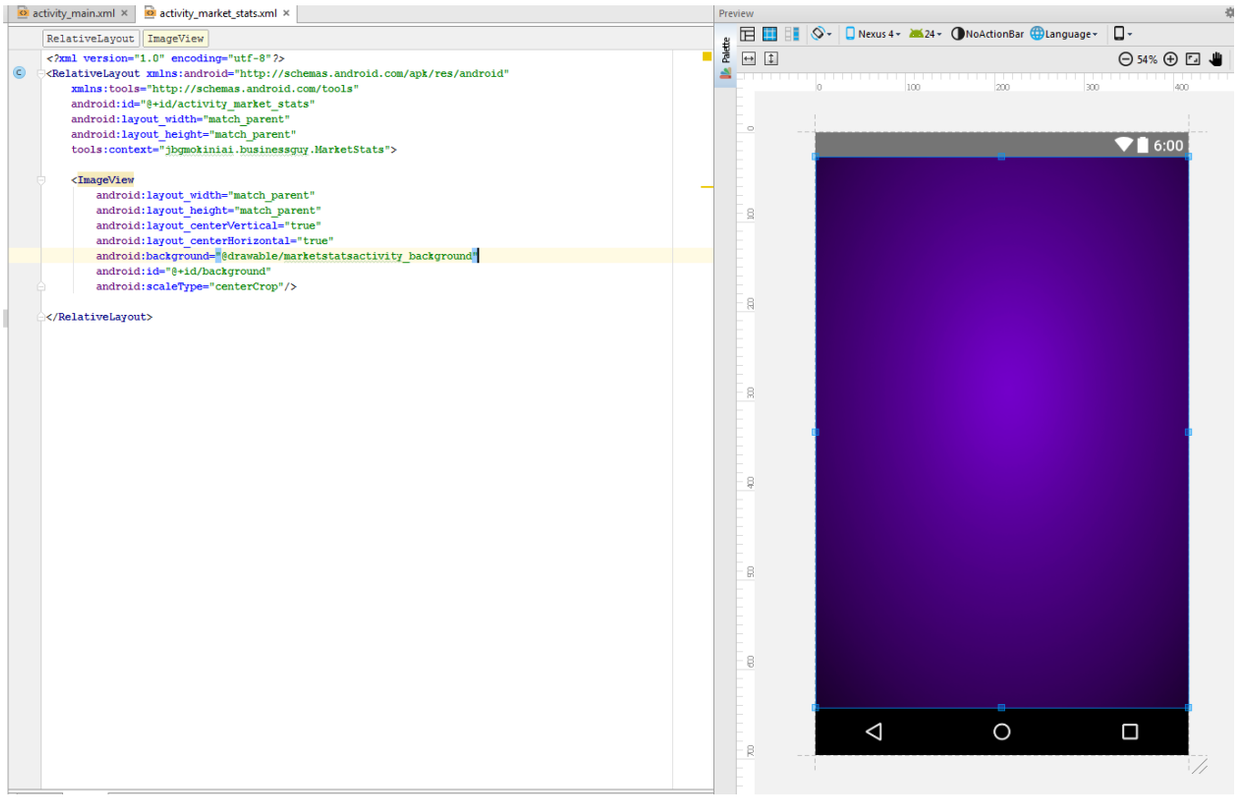


5. Pamažinome ekrano dydį į Nexus 5 (nes mes emuliuojame Nexus 5 per simuliaciją) ir pakeitėme mygtukų dydžius. (Po to pritaikysime taip, kad dydis būtų universalus)



6. Išbandėme, ar veikia parašytas kodas. Paspaudę mygtuką, buvome perkelti į naują veiklą.

7. Kiekvienai veiklai sukūrėme naują fono faila – tai tiesiog praeito fono failas su kita spalva. Jį uždėjome kiekvienai veiklai.



8. Išbandėme, ar mus tikrai perkelia į kitas veiklas.

Septintas Darbas – Ekrano dydžio universalumas

1. Prieš tai minėjome, jog ekrano dydžiui mažėjant/didėjant, mygtukų dydis nekinta. Taigi, mums reikės tai pakoreguoti. Kad mygtukų dydis kistu kartu su ekrano dydžiu, reikia naudoti *LinearLayout* (linijinį išdėstymą). Taigi, sukūrėme LinearLayout pagrindinės veiklos gairėse.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/welcometext"
    android:orientation="vertical">
```

2. Šiame išdėstyme sukūrėme dar vieną LinearLayout (viena eilutė – vienas LinearLayout)

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1">
```

3. Į vieną LinearLayout dėsime vieną eilutę mygtukų (2, 2, 1) Jų proporcingam dydžiui naudojamas android:layout_weight parametras – šio parametro dėka programa nustato, kiek mygtukas turi užimti ekrano.

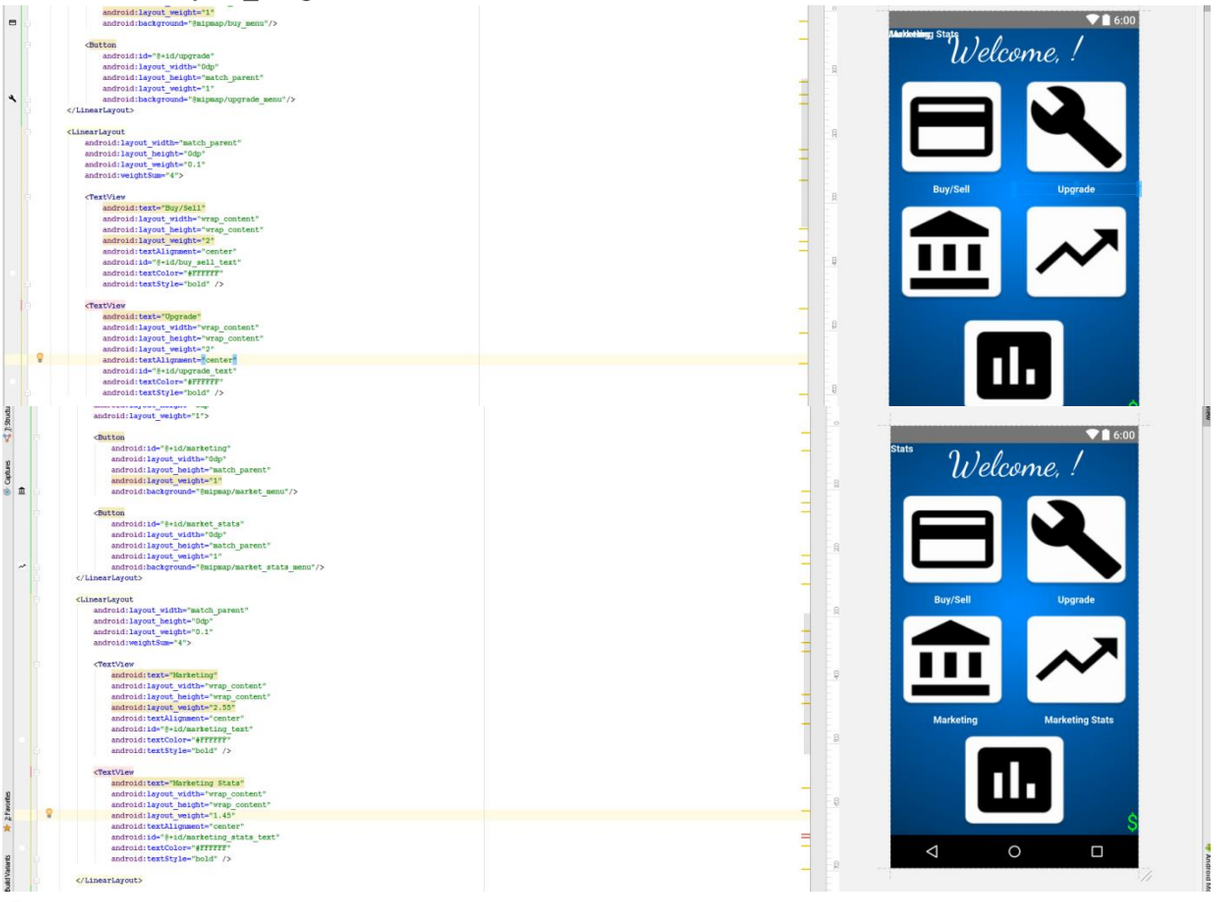
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1">

    <Button
        android:id="@+id/buy_sell"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="@mipmap/buy_menu"/>

    <Button
        android:id="@+id/upgrade"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="@mipmap/upgrade_menu"/>
</LinearLayout>
```

4. Padarę dvi eilutes, trečioje eilutėje tik vienas mygtukas, todėl prirašėme gairę weightSum, kad galėtume naudoti pusę dydžio (kitaip mygtukas būtų išplėstas tiek, kiek užima du mygtukai). Į LinearLayout įrašėme gairę android:gravity=“center_horzintal“ kad mygtuko padėtis būtų per vidurį.

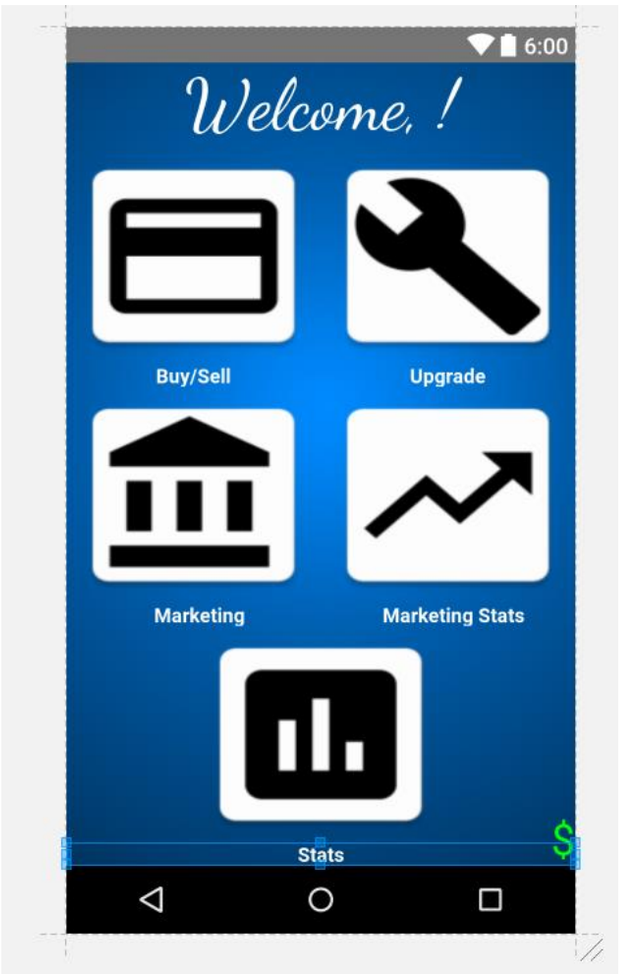
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:weightSum="2"
    android:gravity="center_horizontal">

    <Button
        android:id="@+id/stats"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="@mipmap/stats_menu" />

</LinearLayout>
```

5. Galutinis vaizdas toks.

6. Beliko išdėstyti tekstą. Vienai teksto eilutei – vienas LinearLayout, tačiau naudojame layout_weight šio LinearLayout tik 0.1, nes teksto eilutės ekrane užims mažiau vietos. Kad nustatyti tinkamus tarpus tarp tekstų, reikėjo prirašyti gairę android:weightSum į LinearLayout, o TextView objektuose rasti tinkamas layout_weight reikšmes.



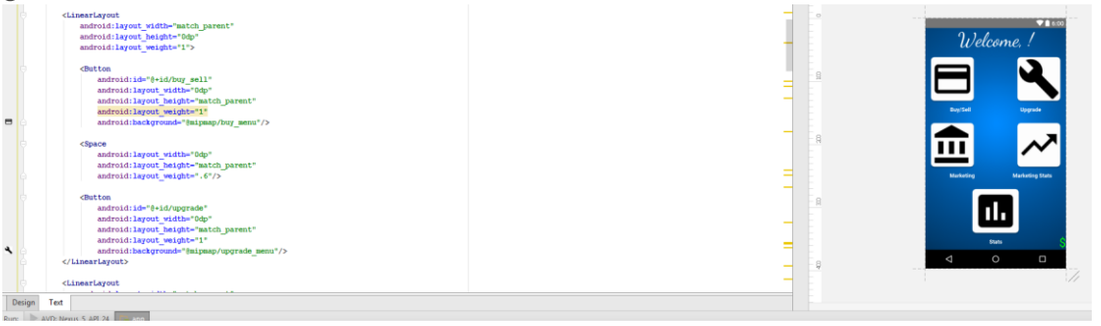
7. Galutinis vaizdas.



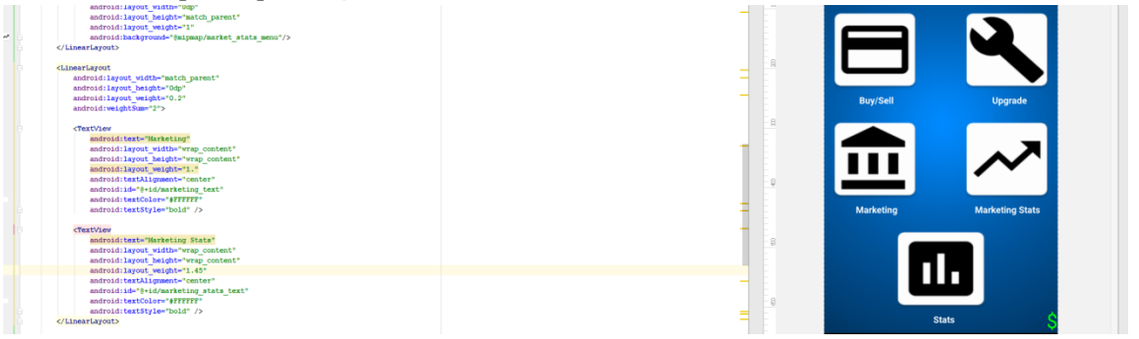
8. Tačiau kažkiek teksto nusikerpa, ir jis yra gan žemai, todėl pakeitėme layout_weight kiekvienos teksto eilutės (LinearLayout) į 0.2.

```
android:layout_weight="0.2"
```

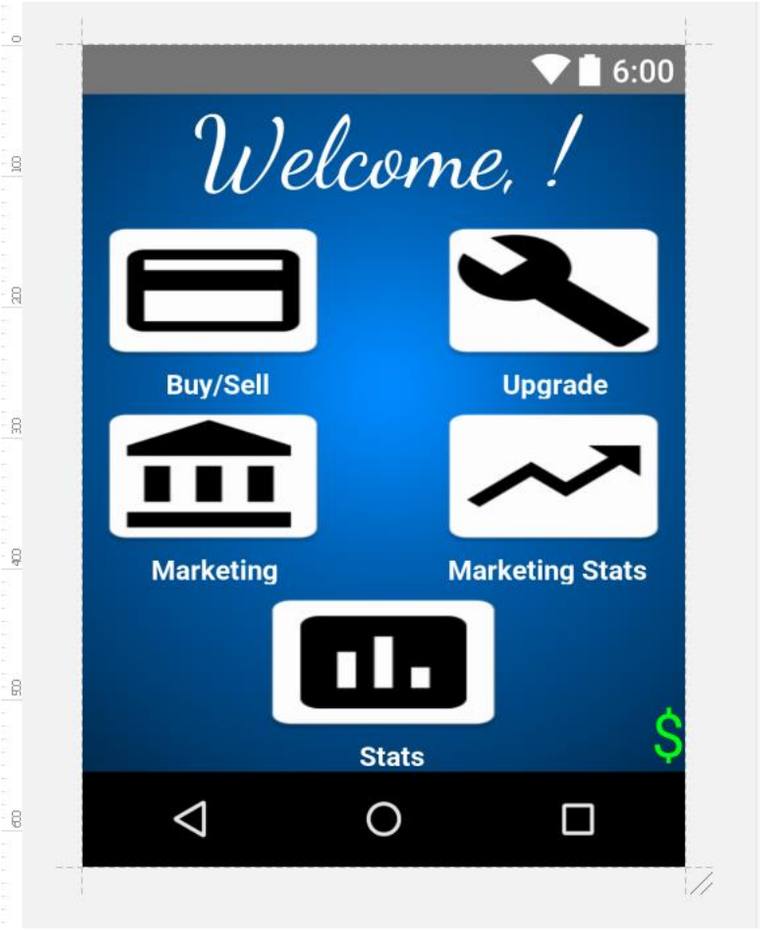
9. Tarp mygtukų pridėjome tarpus (Space gairė), kad jie nebūtų tokie ištempti ir kad atrodytų šiek tiek geriau.



10. Sutvarkėme tarpus tarp tekstų.



11. Išbandėme, kaip veikė šis pritaikymas. Kai pakeitėme ekrano dydį, pasikeitė ir visų mygtukų dydžiai.

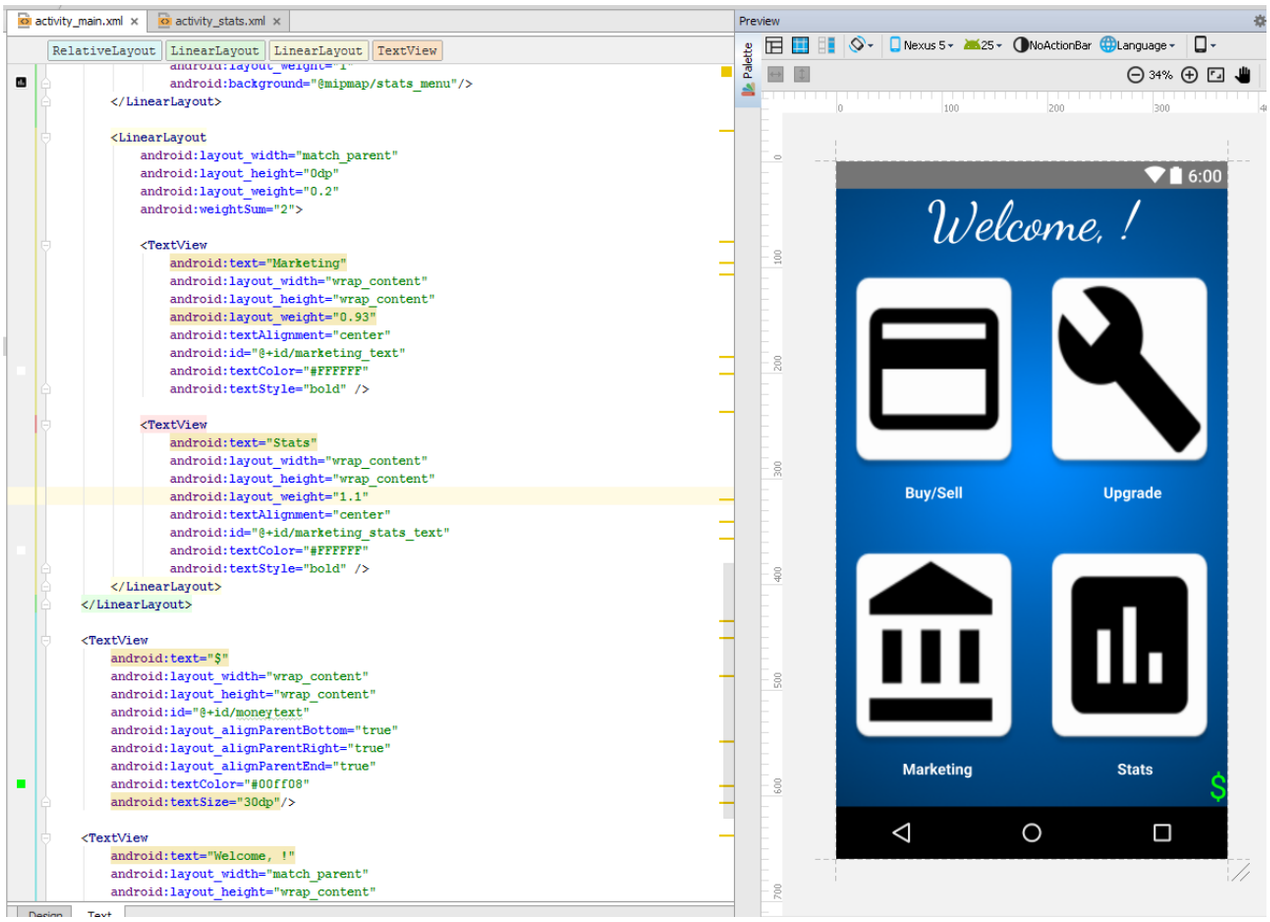


Aštuntas Darbas – Žaidimo idėjos

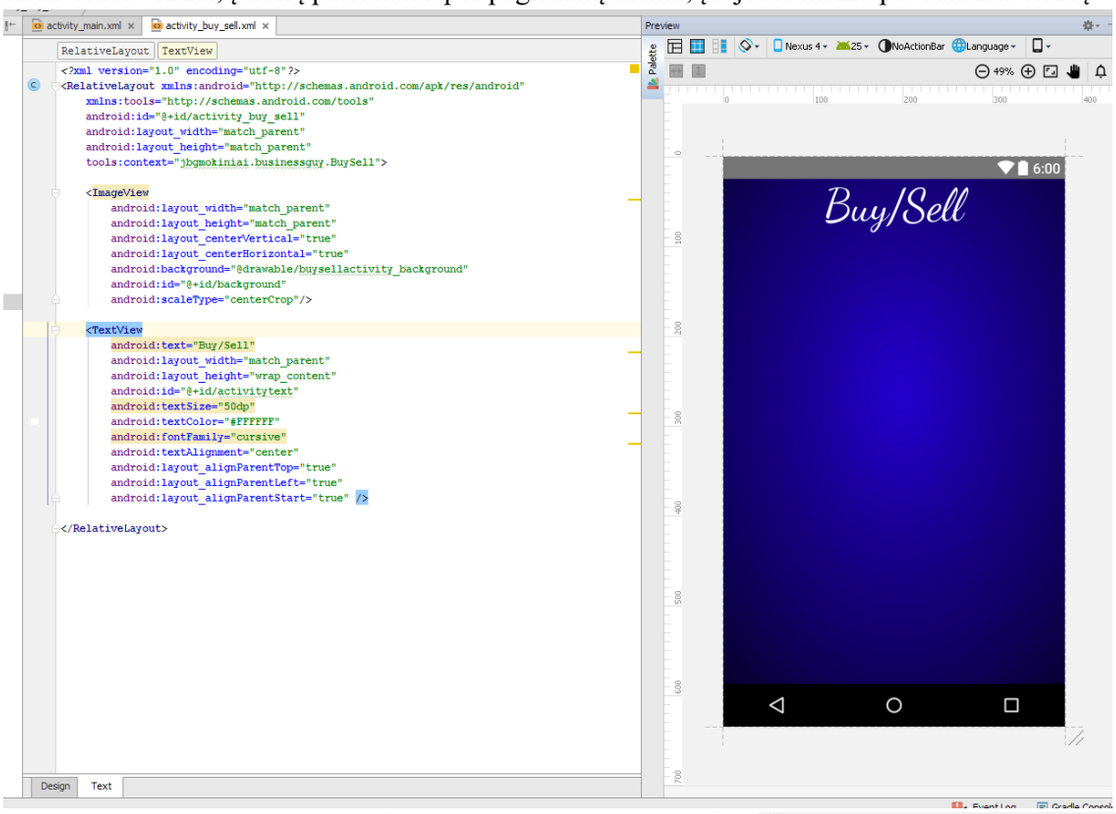
1. Kadangi padarėme pagrindinę ir pradinę veiklą, reikia padaryti kitas veiklas. Taigi, buvo pats metas pradėti susirašinėti idėjas, t.y. kokios bus žaidėjo statistikos, kokias prekes galima bus pirkti ir tt. Pirmiausia sudarėme sąrašą prekių, kurias galima bus pirkti:
    a. Maistas
    b. Elektronika
    c. Auksas
    d. Nafta
    e. Drabužiai
    f. Sporto Prekės
    g. Baldai
    h. Statybines Medziagos
    i. Muzikos Prekės
    j. Indai
    k. Knygos
    l. Medžioklės Reikmenys
    m. Žaislai
    n. Medicinos Prekės
    o. Kuras
    p. Resursai
    q. Augalai
    r. Įrankiai
    s. Buitiniai Reikmenys
    t. Cheminės Medžiagos
2. Tada sudarėme statistikų sąrašą:
    a. Visas uždarbis
    b. Pirktos prekės
    c. Parduotos prekės
    d. Bendri išleisti pinigai
    e. Populiariausia prekė
    f. Populiarumas
3. Marketing statistikų sąrašas (savaitės rezultatai/rinka):
    a. Uždarbis
    b. Pirktos prekės
    c. Parduotos prekės
    d. Išleisti pinigai
    e. Populiarumas
    f. Kainų padidėjimai/sumažėjimai
    g. Populiariausia savaitės prekė
    h. Brangiausia
    i. Pigiausia
4. Patobulinimų sąrašas:
    a. Sandelis (didina leistiną laikyti prekių kiekį)
    b. Sunkvėžimiai (leidžia priimti daugiau pasiūlymų per savaitę)
    c. Marketingas (didina populiarumą)
5. Ir apibrėžėme tam tikras sąvokas:
    a. Populiarumas – Kuo didesnis, tuo daugiau pasiūlymų. Kyla perkant/parduodant arba per upgrade. Kai nedaroma nieko per savaitę, jis mažėja.

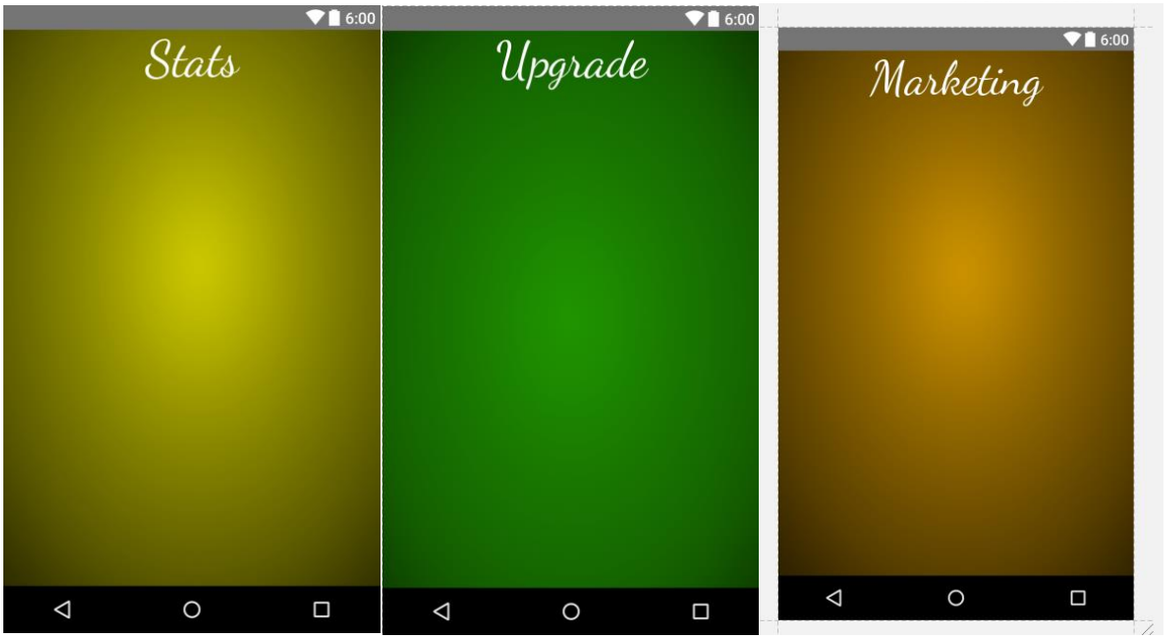
Devintas Darbas – Kitų Veiklų Išdėstymas

1. Šiek tiek persvarstėme idėjas, ir nusprendėme Marketing Stats ir Stats meniu sujungti į vieną.

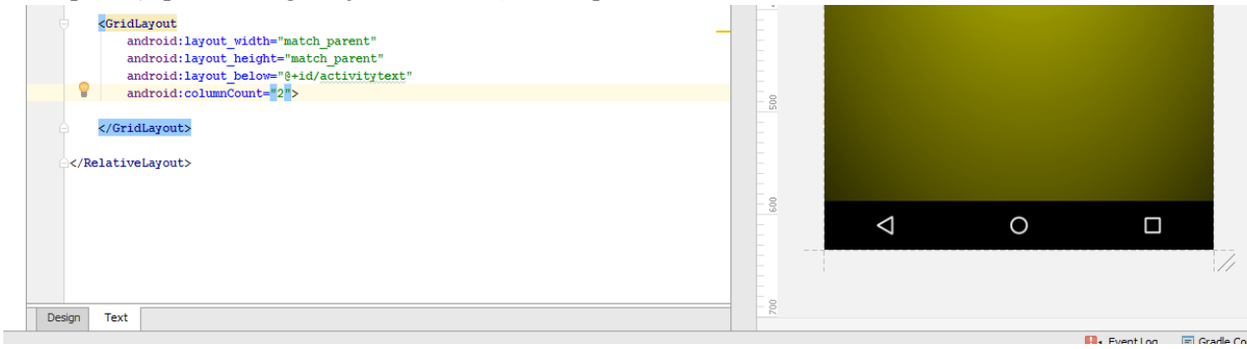

activity_main.xml
activity_stats.xml
RelativeLayout
LinearLayout
LinearLayout
TextView
android:background="@mipmap/stats_menu"/>
</LinearLayout>
<LinearLayout
android:layout_width="match_parent"
android:layout_height="0dp"
android:layout_weight="0.2"
android:weightSum="2">
<TextView
android:text="Marketing"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="0.93"
android:textAlignment="center"
android:id="@+id/marketing_text"
android:textColor="#FFFFFF"
android:textStyle="bold" />
<TextView
android:text="Stats"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1.1"
android:textAlignment="center"
android:id="@+id/marketing_stats_text"
android:textColor="#FFFFFF"
android:textStyle="bold" />
</LinearLayout>
</LinearLayout>
<TextView
android:text="$"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/moneytext"
android:layout_alignParentBottom="true"
android:layout_alignParentRight="true"
android:layout_alignParentEnd="true"
android:textColor="#00ff08"
android:textSize="30dp"/>
<TextView
android:text="Welcome, !"
android:layout_width="match_parent"
android:layout_height="wrap_content"
Design
Text
Preview
Palette
Nexus 5
25
NoActionBar
Language
34%
6:00
Welcome, !
Buy/Sell
Upgrade
Marketing
Stats
$

2. Kiekvienai veiklai, į kurią patenkama per pagrindinį meniu, įdėjome meniu pavadinimo tekstą



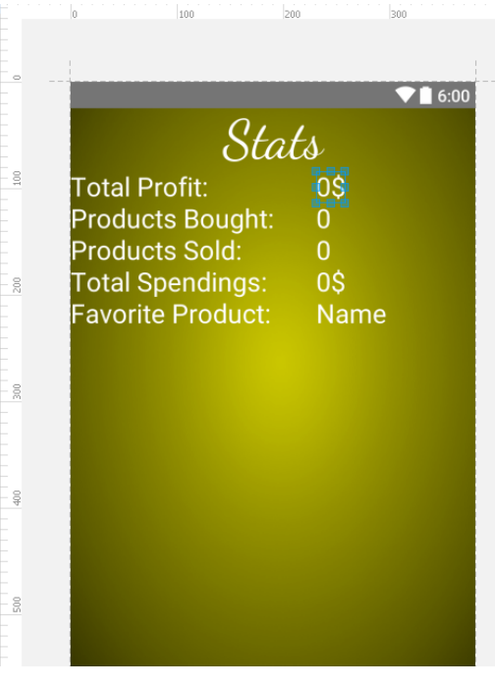


3. Iš pradžių pradėjome dirbti prie Stats veiklos. Įdėjome GridLayout (tai išdėstymas, kuris turi eilutes ir stulpelius), prirašėme gaire jam, kad būtų du stulpeliai.



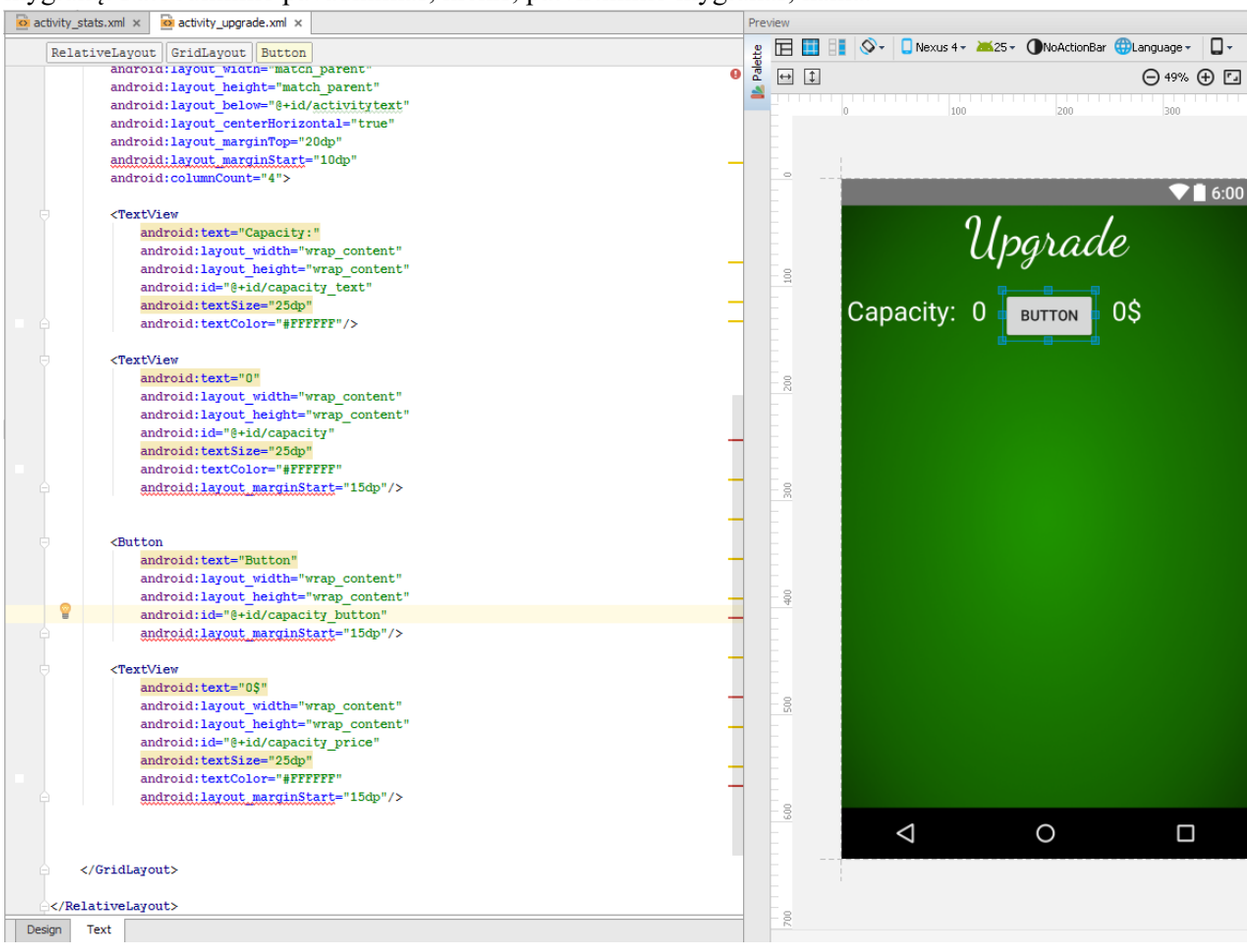
4. Į šį išdėstymą įdėjome po 2 tekstinius objektus kiekvienai statistikai (Statistikos pavadinimas ir kiekis)

```
        android:textColor="#FFFFFF"/>

    <TextView
        android:text="0$"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/total_proffit"
        android:textSize="25dp"
        android:textColor="#FFFFFF"
        android:layout_marginStart="40dp"/>

    <TextView
        android:text="Products Bought:"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/bought_products_text"
        android:textSize="25dp"
        android:textColor="#FFFFFF"/>

    <TextView
        android:text="0"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/bought_products"
        android:textSize="25dp"
        android:textColor="#FFFFFF"
        android:layout_marginStart="40dp" />

    <TextView
        android:text="Products Sold:"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/sold_products_text"
        android:textSize="25dp"
        android:textColor="#FFFFFF"/>

    <TextView
```

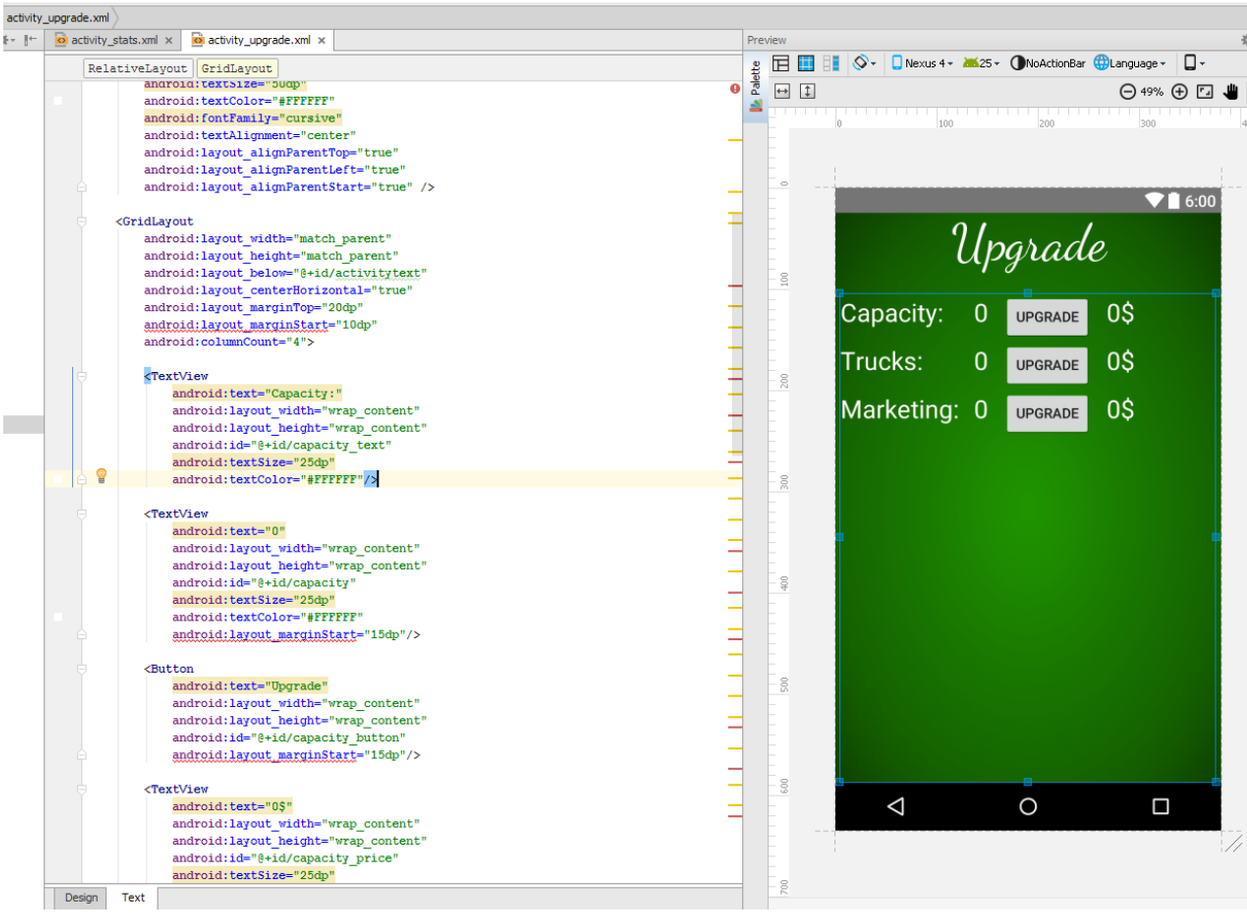


5. Padėjome tarpus tarp kiekvienos eilutės. Taip pat pridėjome tarpą nuo krašto.

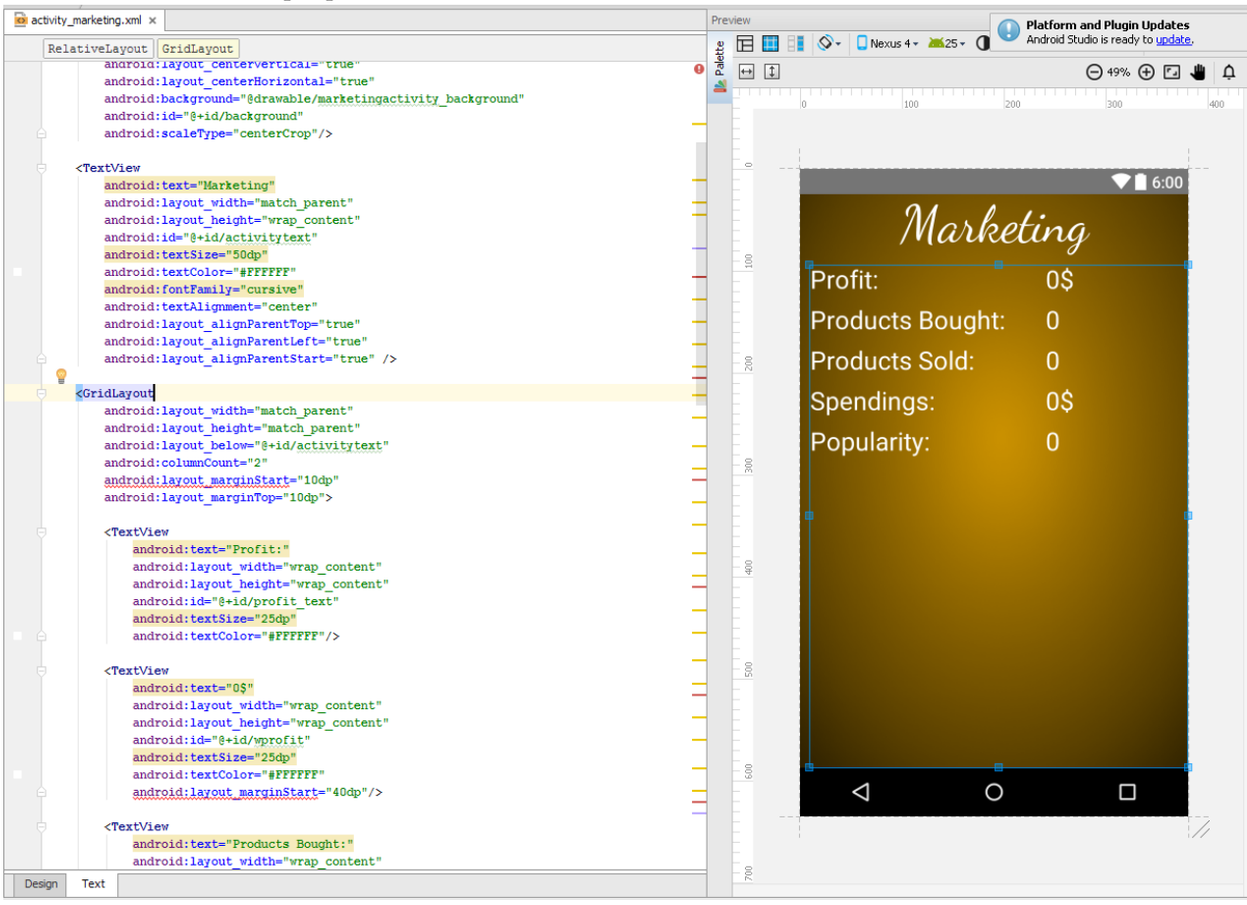
6. Toliau dirbome prie Upgrade veiklos. Kiekvienam patobulinimui šį kart po 3 teksto objektus ir mygtuką: Patobulinimo pavadinimas, kiekis, patvirtinimo mygtukas, kaina.



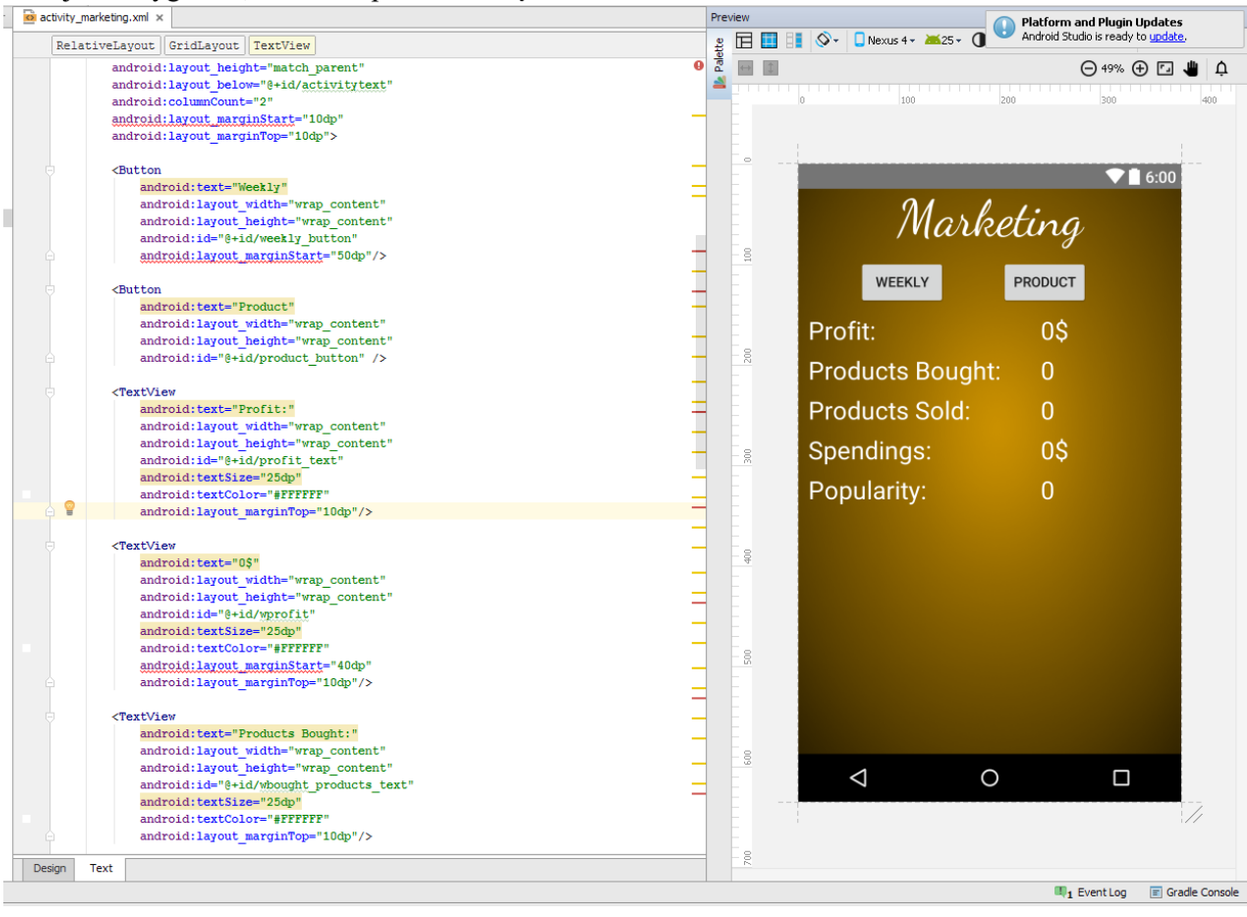
7. Tai padarėme visiems trims patobulinimams.



8. Toliau dirbome prie Marketing veiklos. Šį kart visų marketing statistikų neidėjome, nes savaitės statistikas ir statistikas apie produktus atskirsime skiltimis.



9. Pridėjome mygtukus, kurie leis pakeisti skiltį.



10. Kadangi ištrynėme vieną meniu, tai turėjome pataisyti pagrindinės veiklos kodą. Ištrynėme viską, kas asocijuojasi su market_stats veikla, pakeitėme ciklo kintamąjį nuo 5 iki 4.

```
private int[] menu_ids = { R.id.buy_sell, R.id.upgrade, R.id.marketing, R.id.market_stats, R.id.stats };
```

```
for (int index = 0; index < 4; ++index)
{
    menus[index] = (Button)findViewById(menu_ids[index]);
    menus[index].setOnClickListener(this);
}
```

11. Galiausiai, pridėjome tuščią LinearLayout pagrindinėje veikloje, kad atrodytų šiek tiek gražiau (neištempti mygtukai)

Dešimtas Darbas – Pirkimo/Pardavimo veikla

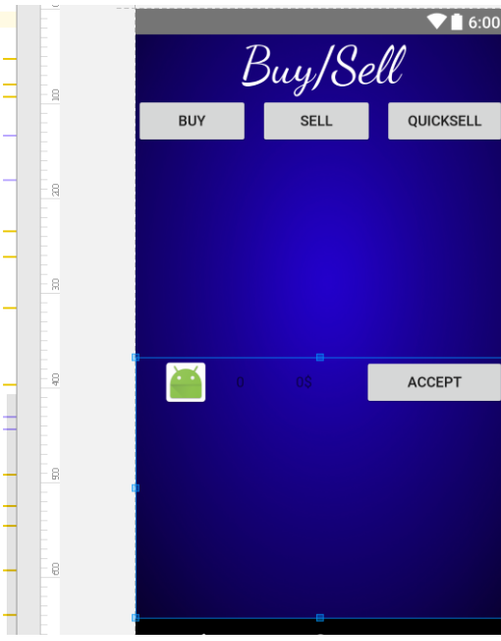
1. Pasidarėme pirkimo/pardavimo veiklos šabloną, pagal kurį darysime šios veiklos išdėstymą.

2. Padarėme bendrą LinearLayout visam ekranui, o po juo – LinearLayout viršutiniams mygtukams.

```
<LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/activitytext"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true">


    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:weightSum="4">

        <Button
            android:text="Buy"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/buy_button"
            android:layout_weight="1"/>

        <Space
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.5" />

        <Button
            android:text="Sell"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/sell_button"
            android:layout_weight="1" />

        <Space
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.5"/>

        <Button
            android:text="Quicksell"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/quicksell_button"
            android:layout_weight="1" />
    </LinearLayout>
```

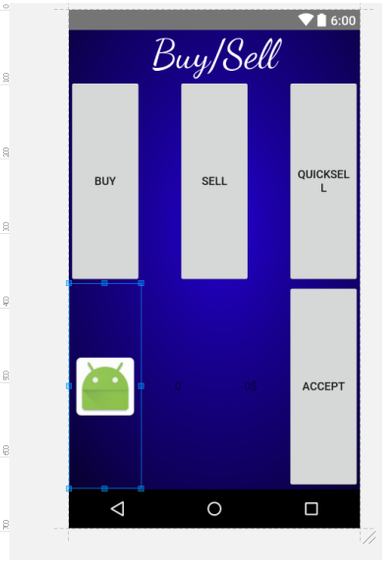
3. Padarėme dar vieną layout produktui. Šis vienas LinearLayout susideda iš 4 objektų: nuotraukos, kiekio, kainos ir patvirtinimo mygtuko.

```
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1"
            android:weightSum="4">

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                app:srcCompat="@mipmap/ic_launcher"
                android:id="@+id/product1_img"
                android:layout_weight="1"/>

            <TextView
                android:text="0"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/product1_name"
                android:layout_weight="1"/>

            <TextView
                android:text="0$"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/product1_price"
                android:layout_weight="1"/>

            <Button
                android:text="Accept"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/product1_accept"
                android:layout_weight="1"/>
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```



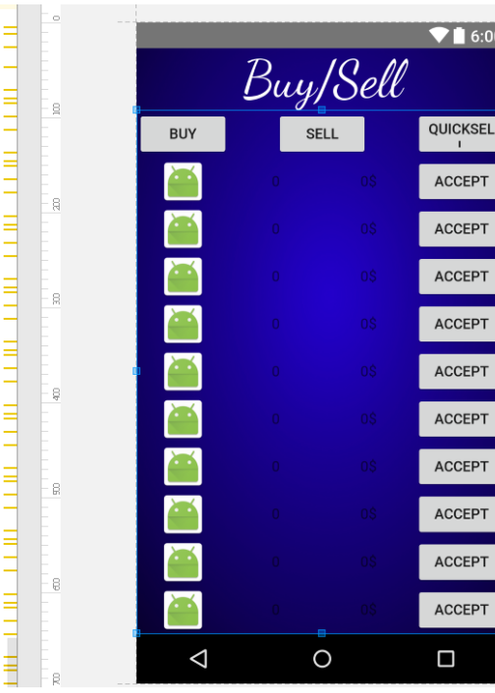
4. Sutvarkėme LinearLayout svorius.

```
                android:layout_height="match_parent"
                android:id="@+id/quicksell_button"
                android:layout_weight="1" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:weightSum="4">

            <ImageView
                android:layout_width="0dp"
                android:layout_height="match_parent"
                app:srcCompat="@mipmap/ic_launcher"
                android:id="@+id/product1_img"
                android:layout_weight="1"/>

            <TextView
                android:text="0"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:gravity="center"
                android:id="@+id/product1_name"
                android:layout_weight="1"/>

            <TextView
                android:text="0$"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:gravity="center"
                android:id="@+id/product1_price"
                android:layout_weight="1"/>

            <Button
                android:text="Accept"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/product1_accept"
                android:layout_weight="1"/>
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```



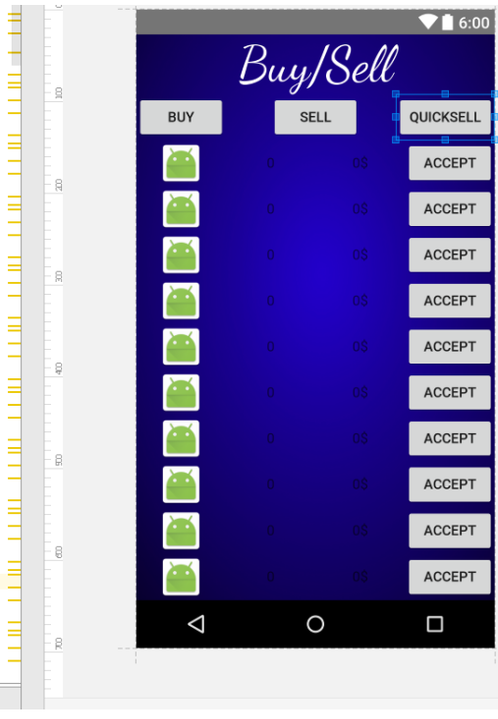
5. Pridėjome dar 9 produkto LinearLayout.

```
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:weightSum="4">

            <ImageView
                android:layout_width="0dp"
                android:layout_height="match_parent"
                app:srcCompat="@mipmap/ic_launcher"
                android:id="@+id/product10_img"
                android:layout_weight="1"/>

            <TextView
                android:text="0"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:gravity="center"
                android:id="@+id/product10_name"
                android:layout_weight="1"/>

            <TextView
                android:text="0$"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:gravity="center"
                android:id="@+id/product10_price"
                android:layout_weight="1"/>

            <Button
                android:text="Accept"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/product10_accept"
                android:layout_weight="1"/>
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```

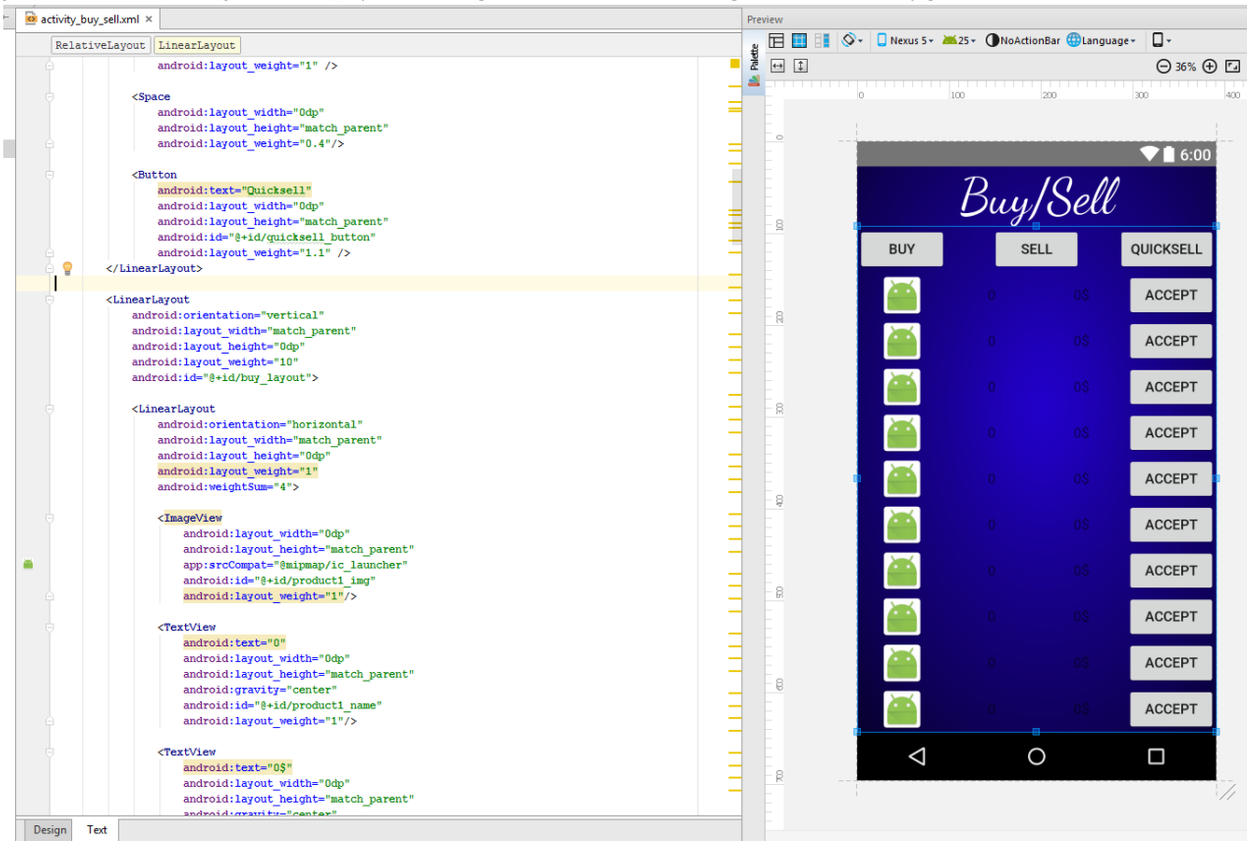


6. Pamažinome tarpą tarp Sell ir Quicksell mygtuko, padidinome Quicksell mygtuko svorį, kad sutilptų visas tekstas.

```
        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:weightSum="4">

            <Button
                android:text="Buy"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/buy_button"
                android:layout_weight="1"/>

            <Space
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="0.5" />

            <Button
                android:text="Sell"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/sell_button"
                android:layout_weight="1" />

            <Space
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="0.4"/>

            <Button
                android:text="Quicksell"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/quicksell_button"
                android:layout_weight="1.1" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
```



7. Visus produktų LinearLayout sudėjome į naują LinearLayout, kurį pavadinome buy_layout. Davėm jam 10 svorio, kad jis užimtų žymiai daugiau ekrano vietos, negu viršutiniai mygtukai.

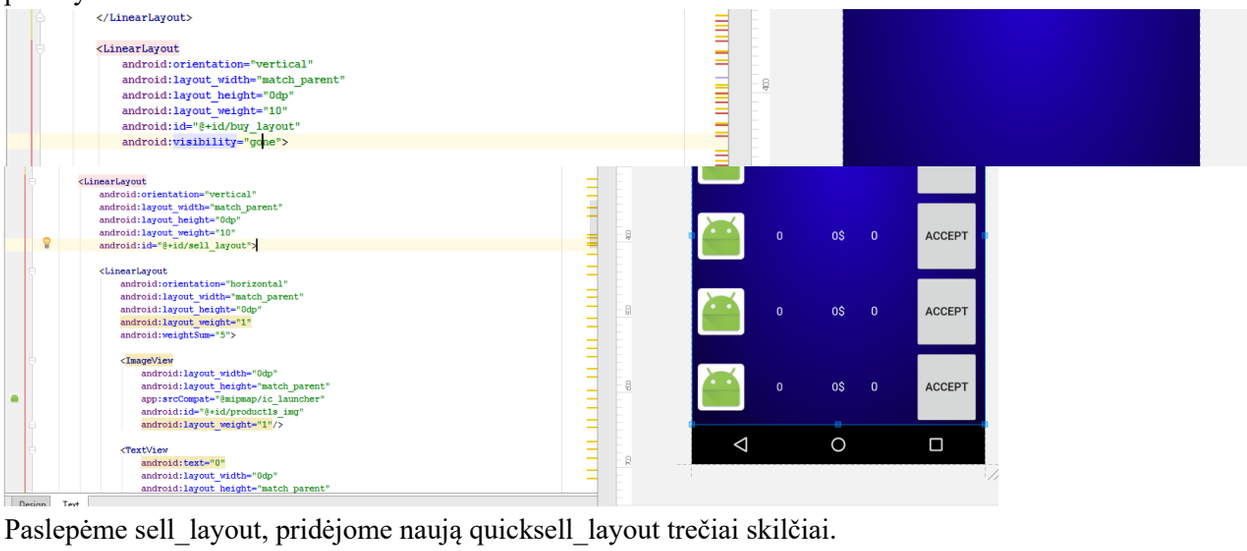


8. Pakeitėme teksto spalvą į baltą.

9. Produktui nusprendėme įdėti dar vieną objektą – tekstinį laukelį, kuriame galima įvesti kiek norima pirkti produkto.

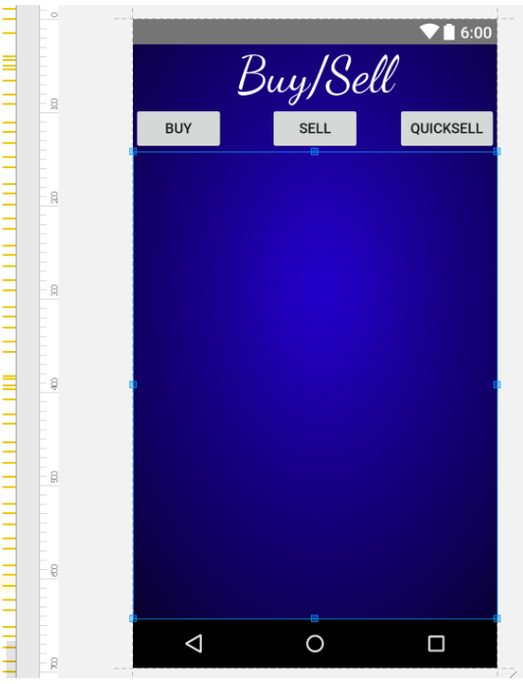
10. Padarėme tai kiekvienam produktui, sumažinome produktų kiekį šiame išdėstyme iki 5 (kad būtų lengviau viską išbandyti)

11. Buy_layout paslepėme su gaire android:visibility. Pridėjome įdentiška buy_layout LinearLayout (su 5 produktų LinearLayout), kurį pavadinome sell_layout. Jis bus skirtas produktų pardavimo pasiūlymams.



12. Paslepėme sell_layout, pridėjome naują quicksell_layout trečiai skilčiai.

```
            <TextView
                android:text="0$"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:gravity="center"
                android:id="@+id/product5s_price"
                android:layout_weight="1"
                android:textColor="#FFFFFF"/>

            <EditText
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:inputType="number"
                android:text="0"
                android:ems="10"
                android:id="@+id/product5s_input"
                android:textColor="#FFFFFF"
                android:layout_weight="0.8"
                android:textSize="15sp"/>

            <Button
                android:text="Accept"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:id="@+id/product5s_accept"
                android:layout_weight="1.1"/>
        </LinearLayout>
    </LinearLayout>

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="10"
        android:id="@+id/quicksell_layout">


    </LinearLayout>

    </LinearLayout>
</RelativeLayout>
```

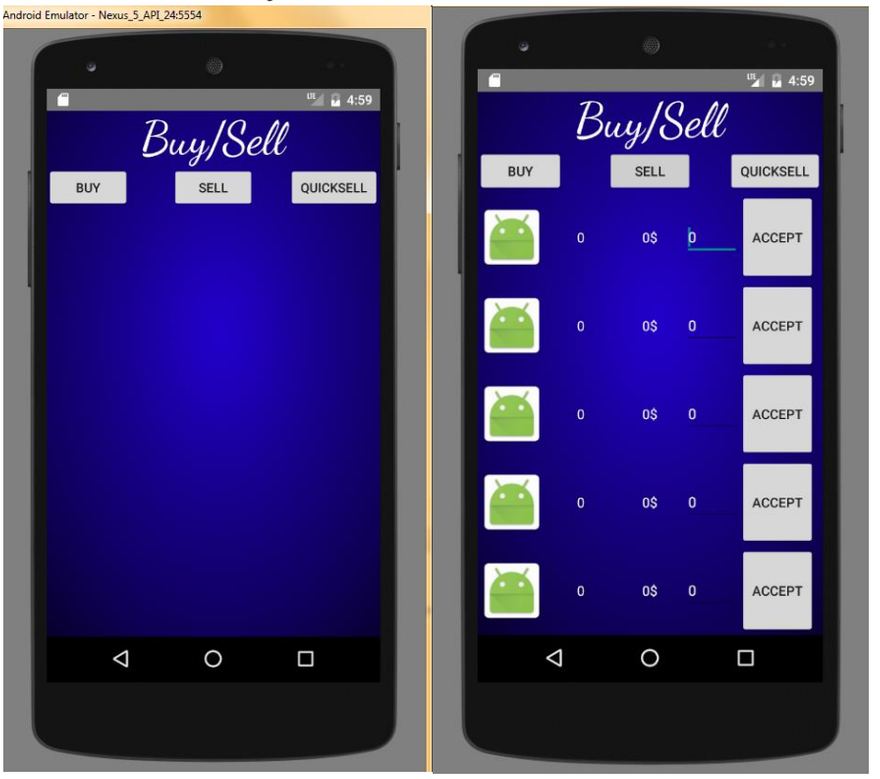


13. Šios veiklos kode parašėme kodą, kad surastų visus pagr. LinearLayout (buy, sell, quicksell), visus viršutinius mygtukus ir funkcionalumą, kad paspaudus mygtuką būtų paslepiami visi LinearLayout ir parodomas tik tas, kurio mygtukas buvo paspaustas.

BuySell.java

```
import ...

public class BuySell extends AppCompatActivity implements View.OnClickListener {

    View layouts[] = new View[3];
    Button layoutbuttons[] = new Button[3];

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_buy_sell);
        layouts[0] = findViewById(R.id.buy_layout);
        layoutbuttons[0] = (Button) findViewById(R.id.buy_button);
        layouts[1] = findViewById(R.id.sell_layout);
        layoutbuttons[1] = (Button) findViewById(R.id.sell_button);
        layouts[2] = findViewById(R.id.quicksell_layout);
        layoutbuttons[2] = (Button) findViewById(R.id.quicksell_button);

        for (int index = 0; index < 3; ++index)
        {
            layoutbuttons[index].setOnClickListener(this);
        }
    }

    @Override
    public void onClick(View v)
    {
        switch (v.getId())
        {
            case R.id.buy_button:
                layouts[1].setVisibility(View.GONE);
                layouts[2].setVisibility(View.GONE);
                layouts[0].setVisibility(View.VISIBLE);
                break;
            case R.id.sell_button:
                layouts[0].setVisibility(View.GONE);
                layouts[2].setVisibility(View.GONE);
                layouts[1].setVisibility(View.VISIBLE);
                break;
            case R.id.quicksell_button:
                layouts[0].setVisibility(View.GONE);
                layouts[1].setVisibility(View.GONE);
                layouts[2].setVisibility(View.VISIBLE);
                break;
        }
    }
}
```

14. Išbandėme emuliatoriuje.



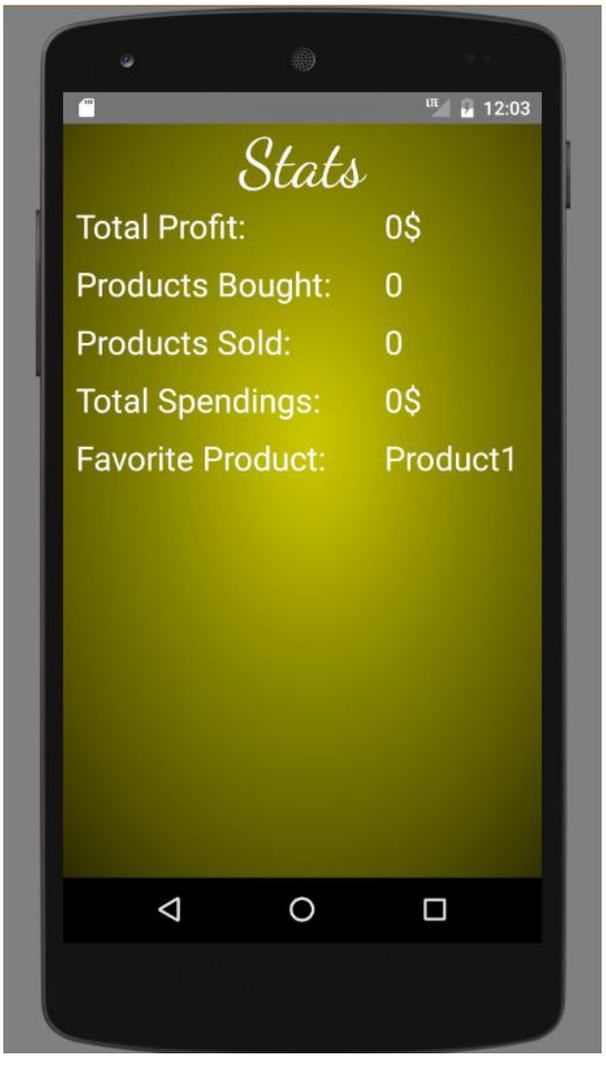
Vienuoliktas Darbas – Veiklų teksto pakeitimas kodu, patvarkymai

1. Pradėjome nuo Stats meniu kodo. Į šios veiklos kodo parašėme kodą, kad kai įeinama į šią veikla, tekstai pakeičiami į atitinkamas statistikas, kurios imamos iš Preferences.

Stats.java

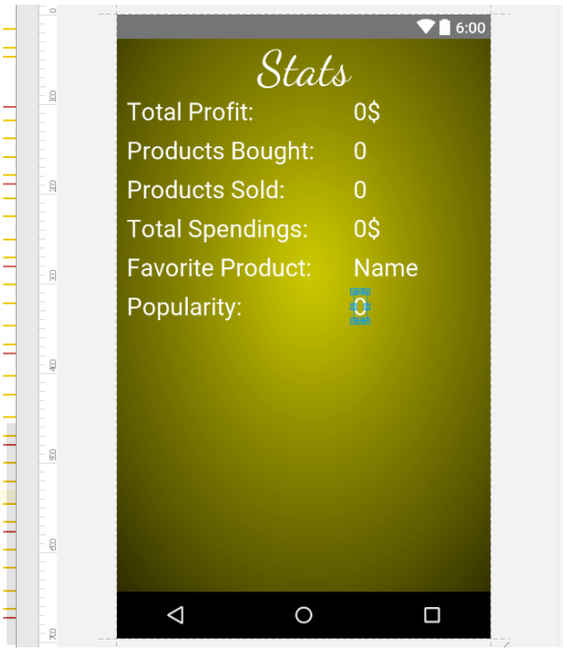
```
public class Stats extends AppCompatActivity {

    int t_profit;
    TextView t_profit_text;
    int prod_bought;
    TextView prod_bought_text;
    int prod_sold;
    TextView prod_sold_text;
    int t_spendings;
    TextView t_spendings_text;
    int f_product;
    TextView f_product_text;

    String product_names[] = {"Product1", "Product2"};


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_stats);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        t_profit = pref.getInt("t_profit", 0);
        prod_bought = pref.getInt("prod_bought", 0);
        prod_sold = pref.getInt("prod_sold", 0);
        t_spendings = pref.getInt("t_spendings", 0);
        f_product = pref.getInt("f_product", 0);

        t_profit_text = (TextView) findViewById(R.id.total_profit);
        prod_bought_text = (TextView) findViewById(R.id.bought_products);
        prod_sold_text = (TextView) findViewById(R.id.sold_products);
        t_spendings_text = (TextView) findViewById(R.id.total_spendings);
        f_product_text = (TextView) findViewById(R.id.favorite_product);
        UpdateStats();
    }

    void UpdateStats()
    {
        t_profit_text.setText(t_profit + "$");
        prod_bought_text.setText("" + prod_bought);
        prod_sold_text.setText("" + prod_sold);
        t_spendings_text.setText(t_spendings + "$");
        f_product_text.setText(product_names[f_product]);
    }
}
```

2. Išbandėme. Iš Favorite Product eilutės galime matyti, kad šis kodas suveikė.



3. Pridėjome populiarumą į Stats veiklą, nes nusprendėme, kad populiarumas bus skaičiuojamas ir bendras, ir 1 savaitės. Tam prirašėme ir kodo veiklos kodo faile.

```
            android:layout_marginStart="40dp"
            android:layout_marginTop="10dp"/>

        <TextView
            android:text="Favorite Product:"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/favorite_product_text"
            android:textSize="25dp"
            android:textColor="#FFFFFF"
            android:layout_marginTop="10dp"/>

        <TextView
            android:text="Name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/favorite_product"
            android:textSize="25dp"
            android:textColor="#FFFFFF"
            android:layout_marginStart="40dp"
            android:layout_marginTop="10dp"/>

        <TextView
            android:text="Popularity:"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/popularity_text"
            android:textSize="25dp"
            android:textColor="#FFFFFF"
            android:layout_marginTop="10dp"/>

        <TextView
            android:text="0"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/popularity"
            android:textSize="25dp"
            android:textColor="#FFFFFF"
            android:layout_marginStart="40dp"
            android:layout_marginTop="10dp"/>
    </GridLayout>
```

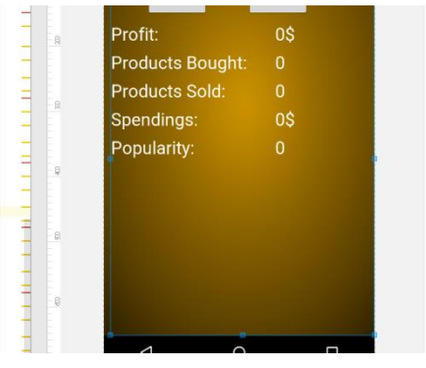


4. Marketing veikloje teko pakeisti Popularity ID, nes čia bus skaičiuojamas savaitinis populiarumas.

```
Stats.java
    int prod_bought;
    TextView prod_bought_text;
    int prod_sold;
    TextView prod_sold_text;
    int t_spendings;
    TextView t_spendings_text;
    int f_product;
    TextView f_product_text;
    int popularity;
    TextView popularity_text;

    String product_names[] = {"Product1", "Product2"};


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_stats);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        t_profit = pref.getInt("t_profit", 0);
        prod_bought = pref.getInt("prod_bought", 0);
        prod_sold = pref.getInt("prod_sold", 0);
        t_spendings = pref.getInt("t_spendings", 0);
        f_product = pref.getInt("f_product", 0);
        popularity = pref.getInt("popularity", 0);

        t_profit_text = (TextView) findViewById(R.id.total_profit);
        prod_bought_text = (TextView) findViewById(R.id.bought_products);
        prod_sold_text = (TextView) findViewById(R.id.sold_products);
        t_spendings_text = (TextView) findViewById(R.id.total_spendings);
        f_product_text = (TextView) findViewById(R.id.favorite_product);
        popularity_text = (TextView) findViewById(R.id.popularity);
        UpdateStats();
    }

    void UpdateStats()
    {
        t_profit_text.setText(t_profit + "$");
        prod_bought_text.setText("" + prod_bought);
        prod_sold_text.setText("" + prod_sold);
        t_spendings_text.setText(t_spendings + "$");
        f_product_text.setText(product_names[f_product]);
        popularity_text.setText("" + popularity);
    }
}
```

5. Į Marketing veiklos kodą prirašėme tokio pačio kodo, kaip Stats veikloje, tik pritaikėme šiai veiklai (kitokie pavadinimai ir t.t.).

```
    <TextView
        android:text="0$"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/wspendings"
        android:textSize="25dp"
        android:textColor="#FFFFFF"
        android:layout_marginStart="40dp"
        android:layout_marginTop="10dp"/>

    <TextView
        android:text="Popularity:"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/wpopularity_text"
        android:textSize="25dp"
        android:textColor="#FFFFFF"
        android:layout_marginTop="10dp"/>

    <TextView
        android:text="0"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/wpopularity"
        android:textSize="25dp"
        android:textColor="#FFFFFF"
        android:layout_marginStart="40dp"
        android:layout_marginTop="10dp"/>
</GridLayout>
```



6. Į pagrindinę veiklą pridėjome naują mygtuką „Next Week“, kuris bus skirtas pabaigti savaitę.

Marketing.java × Stats.java ×

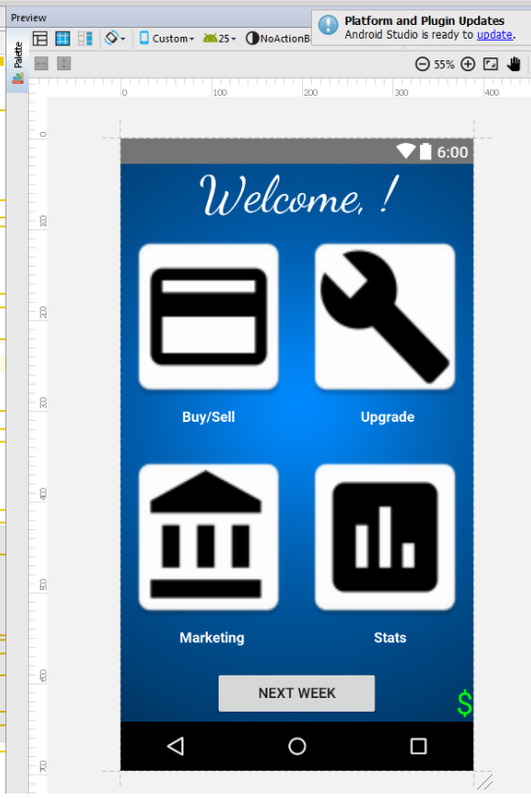
```
package jbgmokiniai.businessguy;

import ...

public class Marketing extends AppCompatActivity {

    int w_profit;
    TextView w_profit_text;
    int w_prod_bought;
    TextView w_prod_bought_text;
    int w_prod_sold;
    TextView w_prod_sold_text;
    int w_spendings;
    TextView w_spendings_text;
    int w_popularity;
    TextView w_popularity_text;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_marketing);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        w_profit = pref.getInt("w_profit", 0);
        w_prod_bought = pref.getInt("w_prod_bought", 0);
        w_prod_sold = pref.getInt("w_prod_sold", 0);
        w_spendings = pref.getInt("w_spendings", 0);
        w_popularity = pref.getInt("w_popularity", 0);

        w_profit_text = (TextView) findViewById(R.id.wprofit);
        w_prod_bought_text = (TextView) findViewById(R.id.wbought_products);
        w_prod_sold_text = (TextView) findViewById(R.id.wsold_products);
        w_spendings_text = (TextView) findViewById(R.id.wspendings);
        w_popularity_text = (TextView) findViewById(R.id.wpopularity);
        UpdateStats();
    }

    void UpdateStats()
    {
        w_profit_text.setText(w_profit + "$");
        w_prod_bought_text.setText("" + w_prod_bought);
        w_prod_sold_text.setText("" + w_prod_sold);
        w_spendings_text.setText(w_spendings + "$");
        w_popularity_text.setText("" + w_popularity);
    }
}
```

7. Į Upgrade veiklos kodą parašėme kodo, kuris pakeistų tekstą į atitinkamas statistikas.

activity_main.xml ×

RelativeLayout LinearLayout LinearLayout

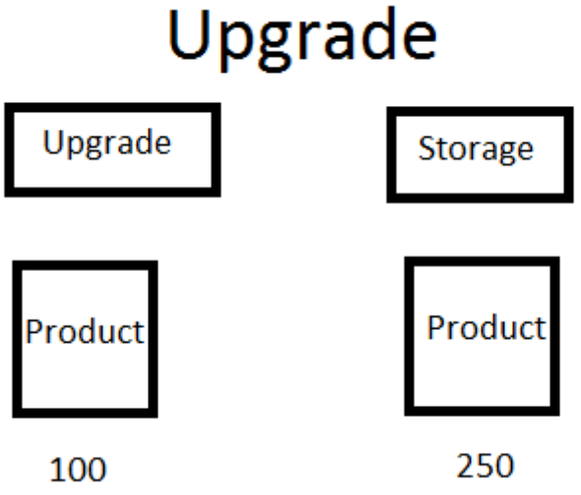
```
                android:layout_weight="0.93"
                android:textAlignment="center"
                android:id="@+id/marketing_text"
                android:textColor="#FFFFFF"
                android:textStyle="bold" />

            <TextView
                android:text="Stats"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1.1"
                android:textAlignment="center"
                android:id="@+id/marketing_stats_text"
                android:textColor="#FFFFFF"
                android:textStyle="bold" />
        </LinearLayout>

        <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="0dp"
                android:layout_weight="0.3"
                android:weightSum="2"
                android:gravity="center">

            <Button
                android:text="next week"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/next_week_button"
                android:layout_weight="0.5" />
        </LinearLayout>
    </LinearLayout>

    <TextView
        android:text="$"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/moneytext"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#00ff08"
        android:textSize="30dp"/>

    <TextView
        android:text="Welcome, !"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```



8. Kadangi Upgrade veikloje daug tuščios, nepanaudotos vietos, nusprendėme čia įdėti sandėlio peržiūra. Nusipaišėme greitą šabloną.

```
Upgrade.java × Stats.java ×
package jbgmokiniai.businessguy;

import ...

public class Upgrade extends AppCompatActivity {

    int capacity_lvl;
    TextView capacity_text;
    int trucks_lvl;
    TextView trucks_text;
    int marketing_lvl;
    TextView marketing_text;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_upgrade);
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);

        capacity_lvl = pref.getInt("capacity_lvl", 0);
        trucks_lvl = pref.getInt("trucks_lvl", 0);
        marketing_lvl = pref.getInt("marketing_lvl", 0);

        capacity_text = (TextView) findViewById(R.id.capacity);
        trucks_text = (TextView) findViewById(R.id.trucks);
        marketing_text = (TextView) findViewById(R.id.marketing);
        UpdateStats();
    }

    void UpdateStats()
    {
        capacity_text.setText("" + capacity_lvl);
        trucks_text.setText("" + trucks_lvl);
        marketing_text.setText("" + marketing_lvl);
    }
}
```

9. Kolkas prie išdėstymo nedirbome, tačiau prirašėme kodą, kuris iš Preferences gautų visų produktų turimą kiekį.



10. Buy Sell veiklos kode padarėme lygiai tą patį, nes ten taip pat reikės šių kintamųjų.

Dvyliktas Darbas – Upgrade veikla

1. Nusprendėme toliau dirbti prie Upgrade veiklos, taigi reikia sugalvoti formule tolesnio patobulinimo kainai nustatyti. Sugalvojome gan paprastą formulę: $lvl * 10000 + 10000$; Čia lvl – patobulinimo dabartinis lygis.
2. Kad veiklos kode būtų mažiau pasikartojančio kodo, padarėme naują klasę, pavadinimu upgrade. Ši klasė turi savo kintamuosiuos, pagal kuriuos bus atskiriami skirtingi patobulinimai. Sukūrus šios klasės objektą, iš kart einama prie konstruktoriaus (public upgrade), kuriam šiuo atveju perleidžiami du duomenys – pavadinimas ir ShardPreferences. Šiais duomenimis pasinaudojant, klasė pakeičia visus kintamuosius į būtent to patobulinimo reikiamam kintamiesiem (randami teksto laukeliai, gaunamas dabartinis lygis). Tolesnėse kodo eilutėse sukūrėme tris šios klasės objektus (kiekvienam patobulinimui – vienas objektas) Su funkcija UpdateStats() bus atnaujinami duomenys įjungiant šią veiklą.

```
public class upgrade
{
    String name;
    int lvl;
    int price;
    TextView lvl_text;
    TextView price_text;

    public upgrade(String n, SharedPreferences pref)
    {
        name = n;
        lvl = pref.getInt(name + "_lvl", 0);
        int upg_id = getResources().getIdentifier(name, "id", getPackageName());
        lvl_text = (TextView) findViewById(upg_id);
        int upg_price_id = getResources().getIdentifier(name + "_price", "id", getPackageName());
        price_text = (TextView) findViewById(upg_price_id);
    }

    void ChangeTexts()
    {
        lvl_text.setText("" + lvl);
        price_text.setText(price + "$");
    }

    void CalculateLvlPrice()
    {
        price = lvl * 10000 + 10000;
    }
}
```

```
upgrade upgrades[] = new upgrade[3];

int products[] = new int[20];
String product_names[] = {"Product1", "Product2"};

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_upgrade);
    SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
    upgrades[0] = new upgrade("capacity", pref);
    upgrades[1] = new upgrade("trucks", pref);
    upgrades[2] = new upgrade("marketing", pref);

    UpdateStats();
}

void UpdateStats()
{
    for (int index = 0; index < 3; ++index)
    {
        upgrades[index].CalculateLvlPrice();
        upgrades[index].ChangeTexts();
    }
}
```

3. Toliau dirbome prie patobulinimo nusipirkimo:
    a. Šiai klasei prirašėme naują funkciją PurchaseUpgrade(), kuri pakelia lygi vienu ir atnaujina teksto laukelius.

```
void PurchaseUpgrade()
{
    lvl++;
    CalculateLvlPrice();
    ChangeTexts();
}
```

    b. Konstruktoriuje prirašėme kodo eilutes, kad rastų šiam patobulinimui reikiamą mygtuką.

```
public upgrade(String n, SharedPreferences pref)
{
    name = n;
    lvl = pref.getInt(name + "_lvl", 0);
    int upg_id = getResources().getIdentifier(name, "id", getPackageName());
    lvl_text = (TextView) findViewById(upg_id);
    int upg_price_id = getResources().getIdentifier(name + "_price", "id", getPackageName());
    price_text = (TextView) findViewById(upg_price_id);
    int upg_button_id = getResources().getIdentifier(name + "_button", "id", getPackageName()
    button = (Button) findViewById(upg_button_id);
}
```

    c. Kiekvienam mygtukui pridėjome OnClickListener(), kad jis turėtų funkcionalumą. (UpdateStats() pakeitėme į InitializeUpgrades())

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_upgrade);
    SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
    upgrades[0] = new upgrade("capacity", pref);
    upgrades[1] = new upgrade("trucks", pref);
    upgrades[2] = new upgrade("marketing", pref);

    InitializeUpgrades();
}

void InitializeUpgrades()
{
    for (int index = 0; index < 3; ++index)
    {
        upgrades[index].CalculateLvlPrice();
        upgrades[index].ChangeTexts();
        upgrades[index].button.setOnClickListener(this);
    }
}

void LoadProducts(SharedPreferences pref)
{
    for (int index = 0; index < 20; ++index)
    {
        products[index] = pref.getInt(product_names[index], 0);
    }
}

@Override
public void onClick(View v)
{

}
```

d. Pridėjome naują kintamąjį į veiklos kodą, kuris bus žaidėjo pinigų kiekis.

```
int money;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_upgrade);
    SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
    upgrades[0] = new upgrade("capacity", pref);
    upgrades[1] = new upgrade("trucks", pref);
    upgrades[2] = new upgrade("marketing", pref);
    money = pref.getInt("money", 0);

    InitializeUpgrades();
}
```

e. onClick funkcijoje parašėme kodą, kuris atpažintų, kurio patobulinimo mygtukas buvo paspaustas. Tada, jeigu užtenka pinigų, tai nuo pinigų atimama patobulinimo kaina, patobulinimas pakeliamas vienu lygiu.

```
@Override
public void onClick(View v)
{
    int id = -1;

    switch (v.getId())
    {
        case R.id.capacity_button:
            id = 0;
            break;
        case R.id.trucks_button:
            id = 1;
            break;
        case R.id.marketing_button:
            id = 2;
            break;

        default:
            break;
    }

    if (id != -1 && money >= upgrades[id].price)
    {
        SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
        money -= upgrades[id].price;
        prefeditor.putInt("money", money);
        prefeditor.commit();
        upgrades[id].PurchaseUpgrade();
    }
}
```

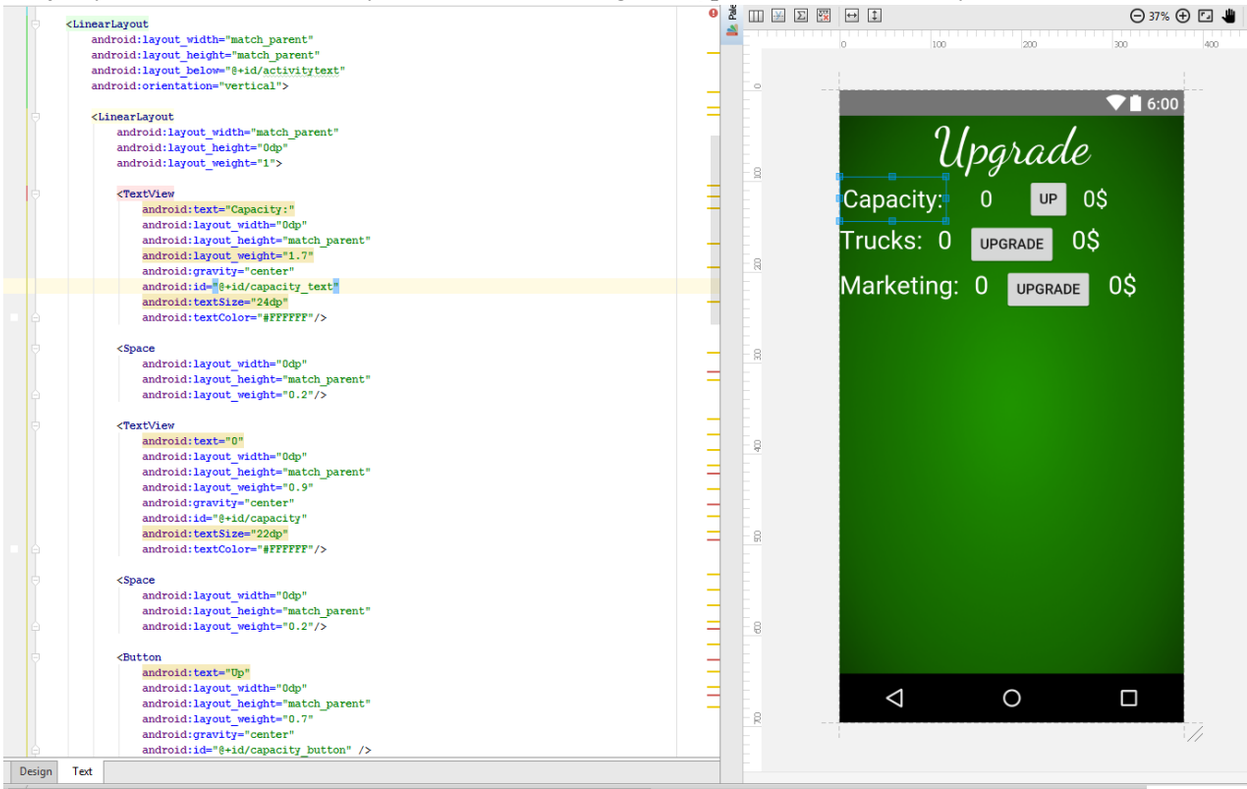
4. Toliau reikia išbandyti, ar visą tai veikia. Taigi pagrindinėje veikloje pasinaudojome prieš tai parašyta funkcija, skirta testavimui, kuri nustato žaidėjo pinigus (ChangeMoney). Nustatėme savo pinigus į 10000. Išbandėme, sėkmingai galėjome nusipirkti patobulinimą, kaina buvo atnaujinta.

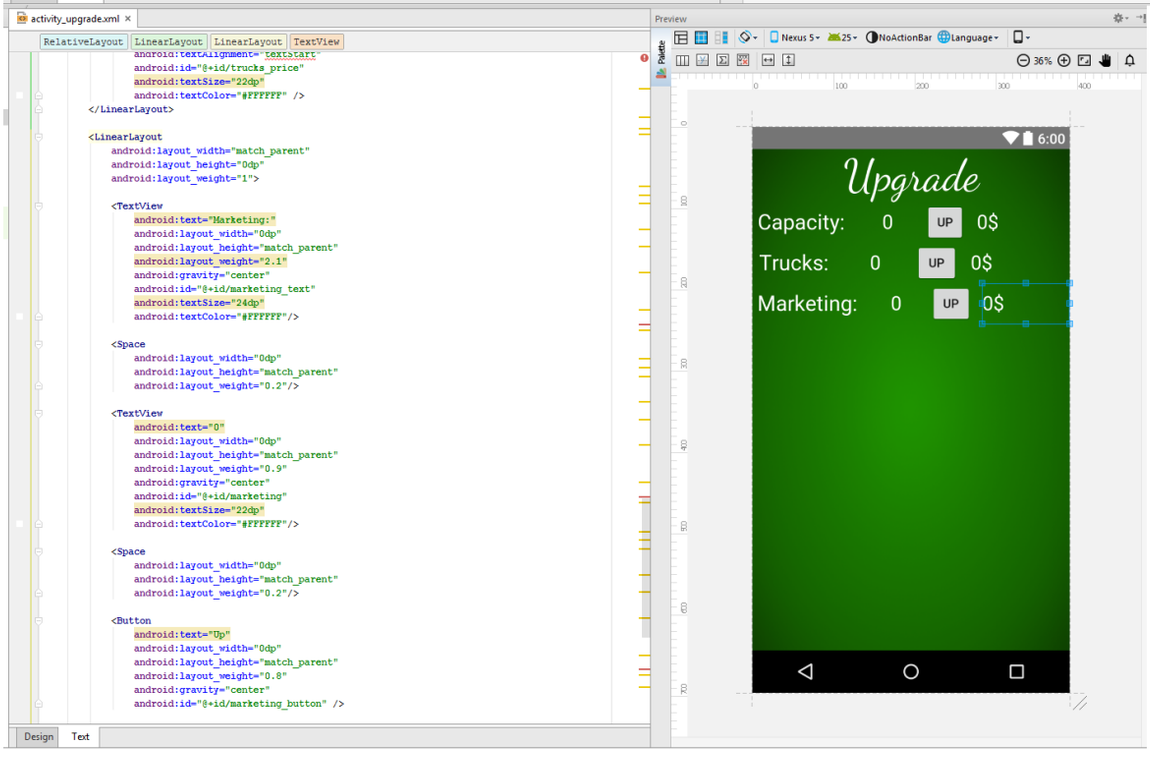
```
public void ChangeMoney(int amount)
{
    money += amount;
    SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
    prefeditor.putInt("money", money);
    prefeditor.commit();
}
```

5. Buvome praleidę vieną dalyka. Į PurchaseUpgrade() funkciją reikia prirašyti kodo, kad lygis būtų išsaugomas duomenų faile.

```
void PurchaseUpgrade()
{
    SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
    lvl++;
    prefeditor.putInt(name + "_lvl", lvl);
    prefeditor.commit();
    CalculateLvlPrice();
    ChangeTexts();
}
```

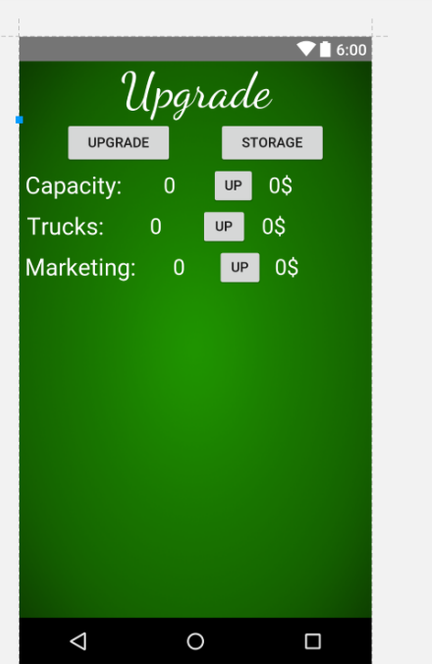
6. Toliau dirbome prie veiklos išdėstymo. Prieš tai naudotas GridLayout nėra geriausias pasirinkimas, nes jo dydis keičiantis ekrano dydžiui nesikeičia. Taigi viską pakeitėme į LinearLayout.





7. Kadangi čia yra dvi skiltys (Upgrade ir Storage), vėlgi darėme tokią pačia sistemą – du LinearLayout, vienas rodomas, kitas paslėptas.

```
            android:text="up"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.8"
            android:gravity="center"
            android:id="@+id/marketing_button" />

        <Space
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="0.2"/>

        <TextView
            android:text="0$"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1.65"
            android:gravity="center"
            android:textAlignment="textStart"
            android:id="@+id/marketing_price"
            android:textSize="22dp"
            android:textColor="#FFFFFF" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="8">
    </LinearLayout>
</LinearLayout>

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="10"
        android:id="@+id/storage_layout"
        android:visibility="gone">
    </LinearLayout>

</LinearLayout>
</RelativeLayout>
```



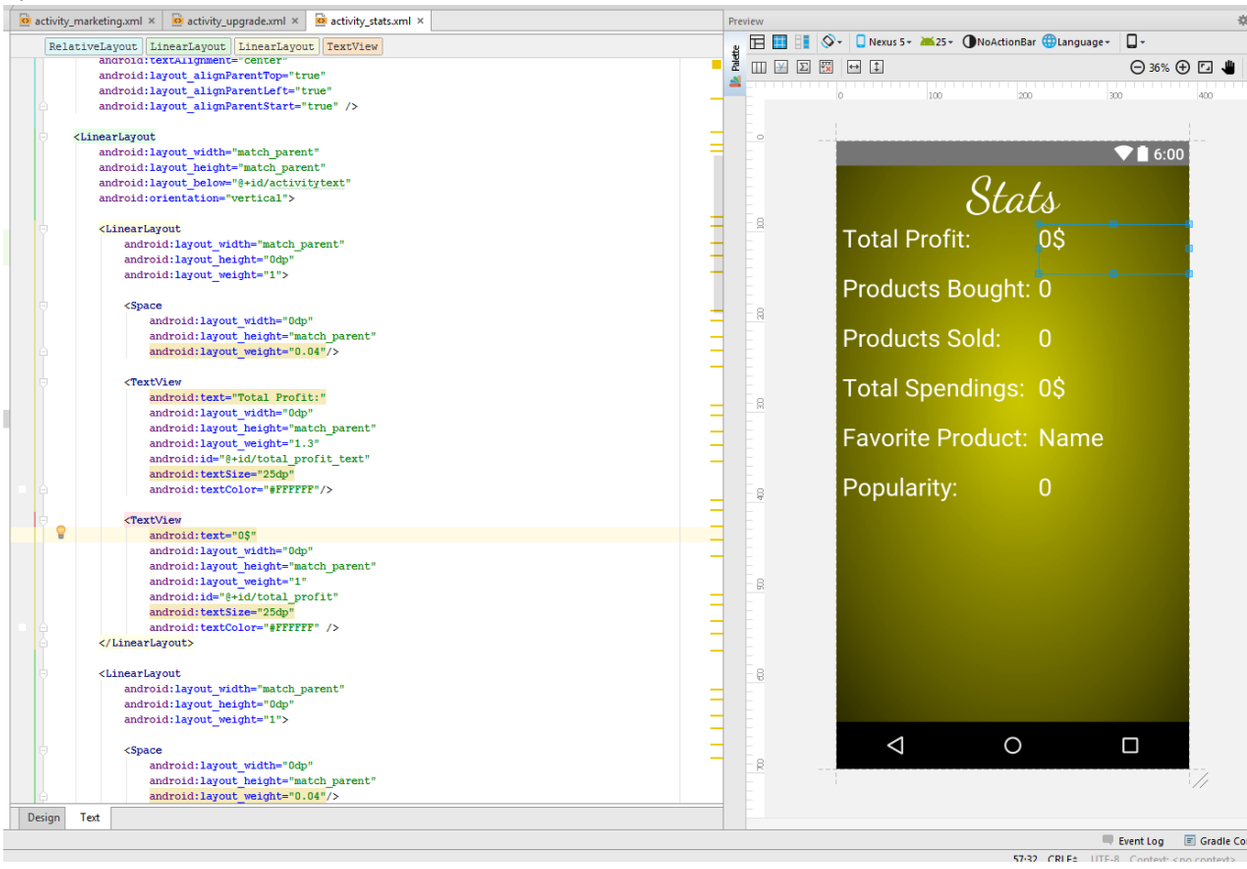
8. Veiklos kode prirašėme kodo, kad būtų rastos veiklos bei mygtukai, mygtukams pridėdami OnClickListener, pridėtas jiem funkcionalumas.

```
View layouts[] = new View[2];
Button layoutbuttons[] = new Button[2];

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_upgrade);
    SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
    layouts[0] = findViewById(R.id.upgrade_layout);
    layoutbuttons[0] = (Button) findViewById(R.id.upgrade_button);
    layouts[1] = findViewById(R.id.storage_layout);
    layoutbuttons[1] = (Button) findViewById(R.id.storage_button);

    for (int index = 0; index < 2; ++index)
        layoutbuttons[index].setOnClickListener(this);
```

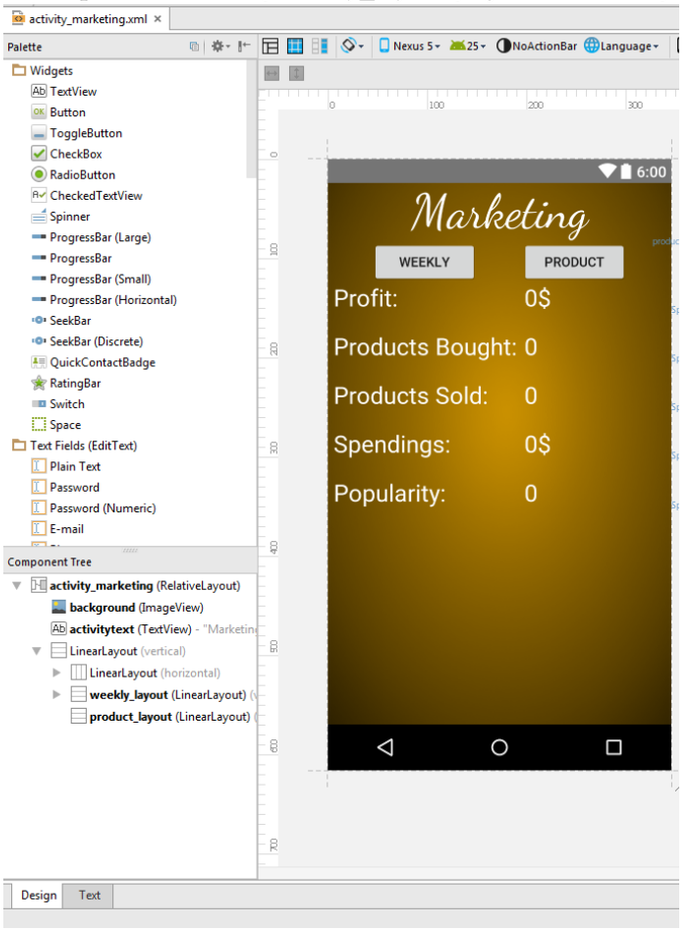
```
@Override
public void onClick(View v)
{
    int id = -1;

    switch (v.getId())
    {
        case R.id.capacity_button:
            id = 0;
            break;
        case R.id.trucks_button:
            id = 1;
            break;
        case R.id.marketing_button:
            id = 2;
            break;
        case R.id.upgrade_button:
            layouts[1].setVisibility(View.GONE);
            layouts[0].setVisibility(View.VISIBLE);
            break;
        case R.id.storage_button:
            layouts[0].setVisibility(View.GONE);
            layouts[1].setVisibility(View.VISIBLE);
            break;

        default:
            break;
    }

    if (id != -1 && money >= upgrades[id].price)
    {
        money -= upgrades[id].price;
        upgrades[id].PurchaseUpgrade();
    }
}
}
```

Tryliktas darbas – Tvarkymai, tolesnis veiklų išdėstymas

1. Stats veiklos GridLayout pakeitėme į LinearLayout, kad keičiant ekrano dydį, keistusi objektų dydžiai.



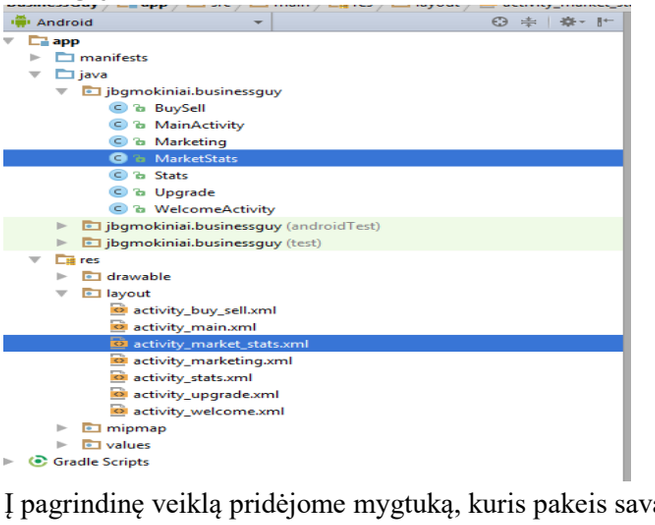
2. Marketing veiklai pridėjome du LinearLayout, kadangi vadovausimės anksčiau naudota sistema, norint pakeisti skiltis. Į weekly_layout įdėjome savaitinių statistikų tekstų objektus.



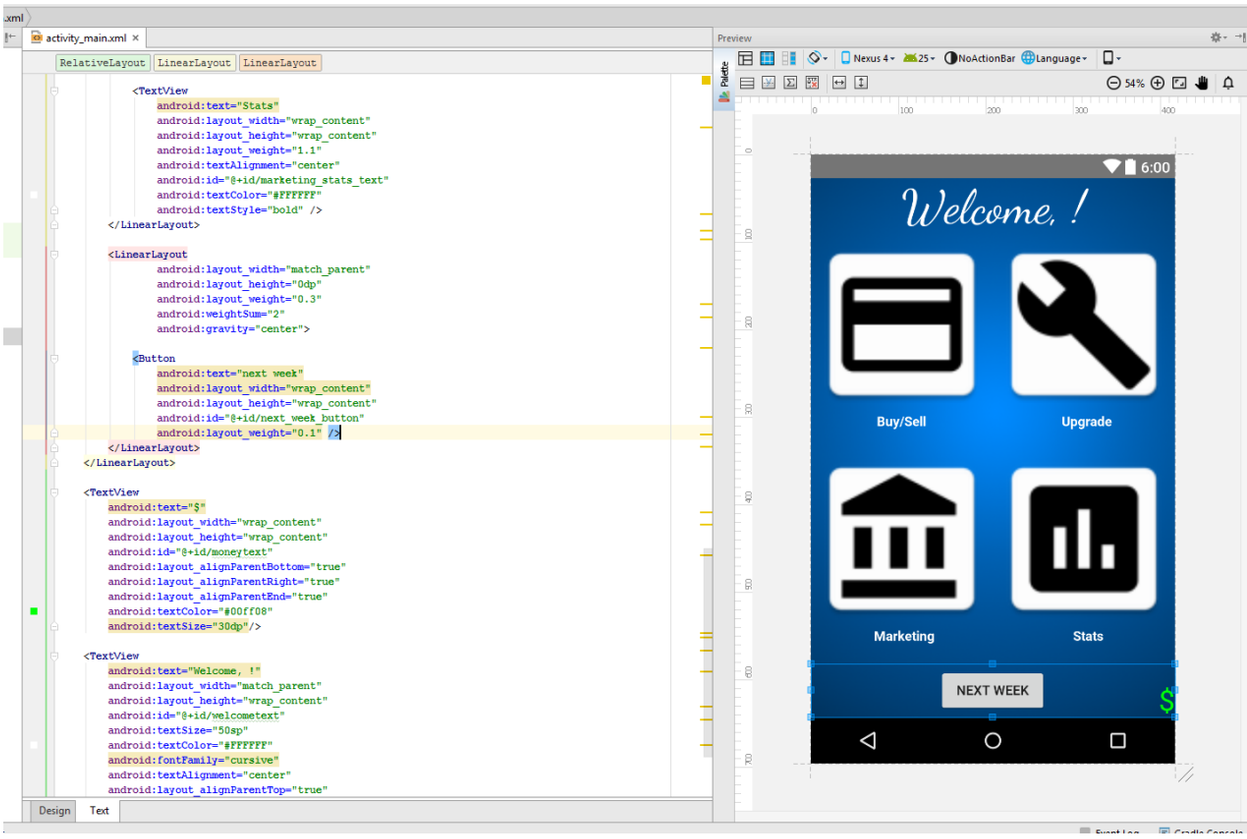
3. Prirašėme kodą, kuris rastų šiuos LinearLayout ir paspaudus mygtukus, pakeistų juos.

```
    layouts[0] = findViewById(R.id.weekly_layout);
    layoutbuttons[0] = (Button) findViewById(R.id.weekly_button);
    layouts[1] = findViewById(R.id.product_layout);
    layoutbuttons[1] = (Button) findViewById(R.id.product_button);

    for (int index = 0; index < 2; ++index)
        layoutbuttons[index].setOnClickListener(this);

    w_profit = pref.getInt("w_profit", 0);
    w_prod_bought = pref.getInt("w_prod_bought", 0);
    w_prod_sold = pref.getInt("w_prod_sold", 0);
    w_spendings = pref.getInt("w_spendings", 0);
    w_popularity = pref.getInt("w_popularity", 0);

    w_profit_text = (TextView) findViewById(R.id.wprofit);
    w_prod_bought_text = (TextView) findViewById(R.id.wbought_products);
    w_prod_sold_text = (TextView) findViewById(R.id.wsold_products);
    w_spendings_text = (TextView) findViewById(R.id.wspendings);
    w_popularity_text = (TextView) findViewById(R.id.wpopularity);
    UpdateStats();
}

void UpdateStats()
{
    w_profit_text.setText(w_profit + "$");
    w_prod_bought_text.setText("" + w_prod_bought);
    w_prod_sold_text.setText("" + w_prod_sold);
    w_spendings_text.setText(w_spendings + "$");
    w_popularity_text.setText("" + w_popularity);
}

@Override
public void onClick(View v)
{
    switch (v.getId())
    {
        case R.id.weekly_button:
            layouts[1].setVisibility(View.GONE);
            layouts[0].setVisibility(View.VISIBLE);
            break;
        case R.id.product_button:
            layouts[0].setVisibility(View.GONE);
            layouts[1].setVisibility(View.VISIBLE);
            break;

        default:
            break;
    }
}
```

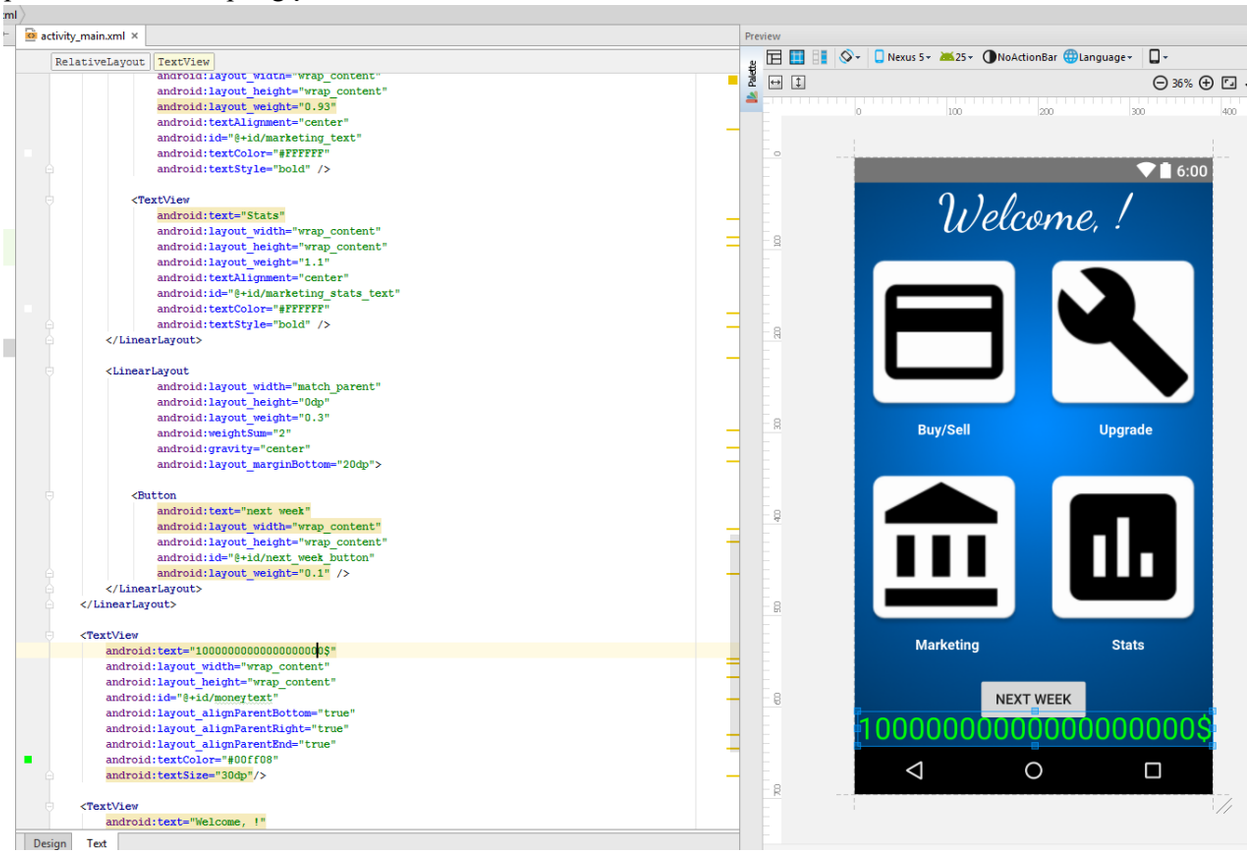
4. Dabar reikėjo ištrinti užsilikusius failus – ištrynėme du failus, susijusius su MarketStats veikla, kadangi jos nebenaudosime.



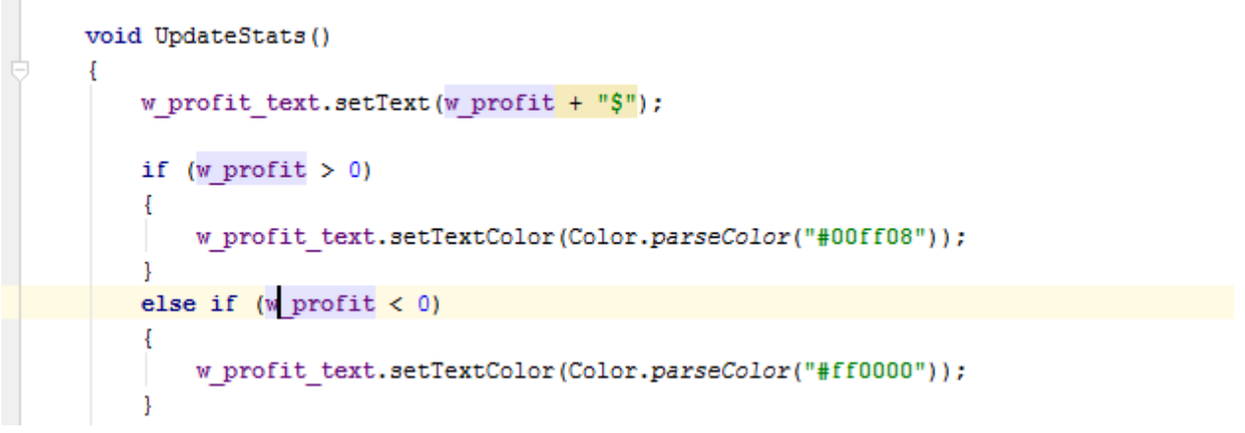
5. Į pagrindinę veiklą pridėjome mygtuką, kuris pakeis savaitę į sekančią.



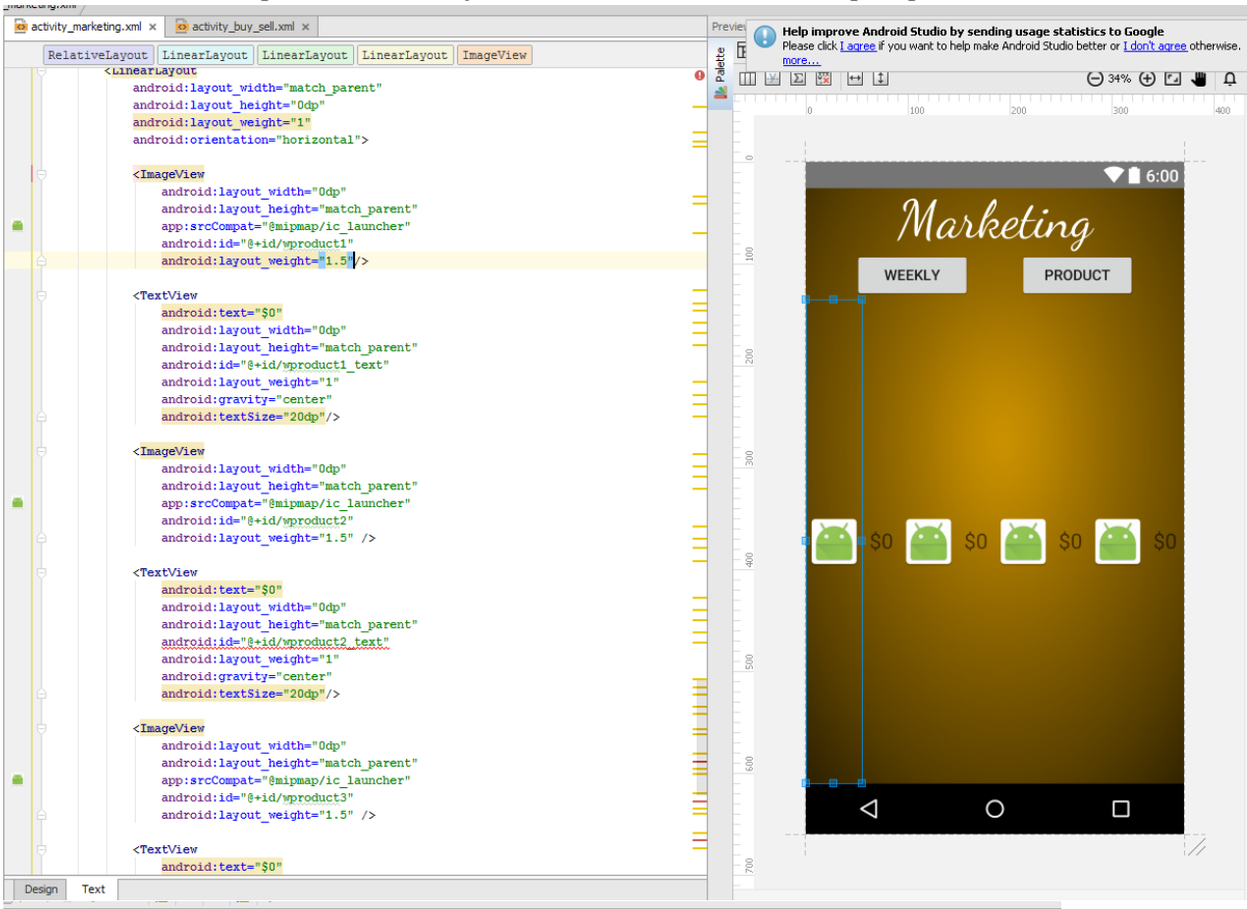
6. Su gaire android:layout_marginBottom pakėleme mygtuko LinearLayout į viršų, kad būtų pakankamai vietos pinigų tekstui.

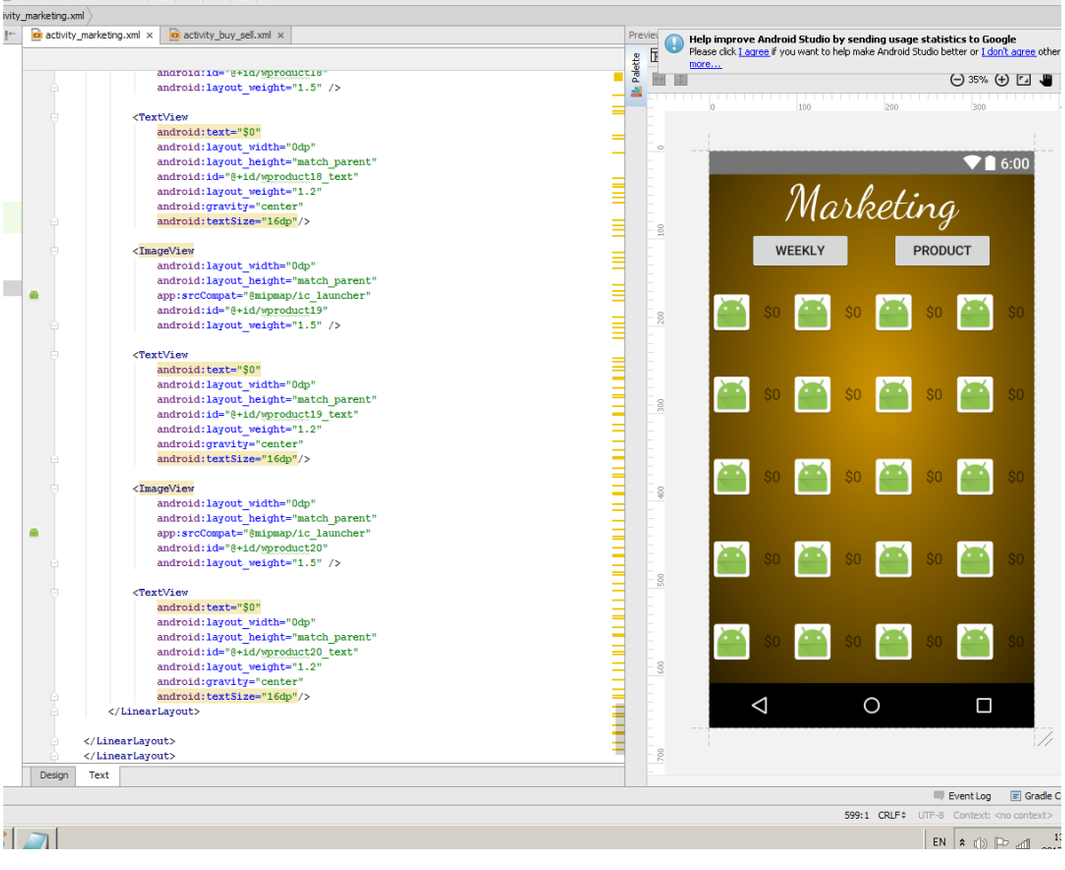


7. I Stats veiklos koda prirašėme kodą, kuris pakeistų pelno teksto spalvą į žalią arba raudoną, atitinkamai pagal tai, ar skaičius teigiamas, ar neigiamas. Marketing veiklos kode padarėme tą patį, tik su savaitės pelno tekstu.

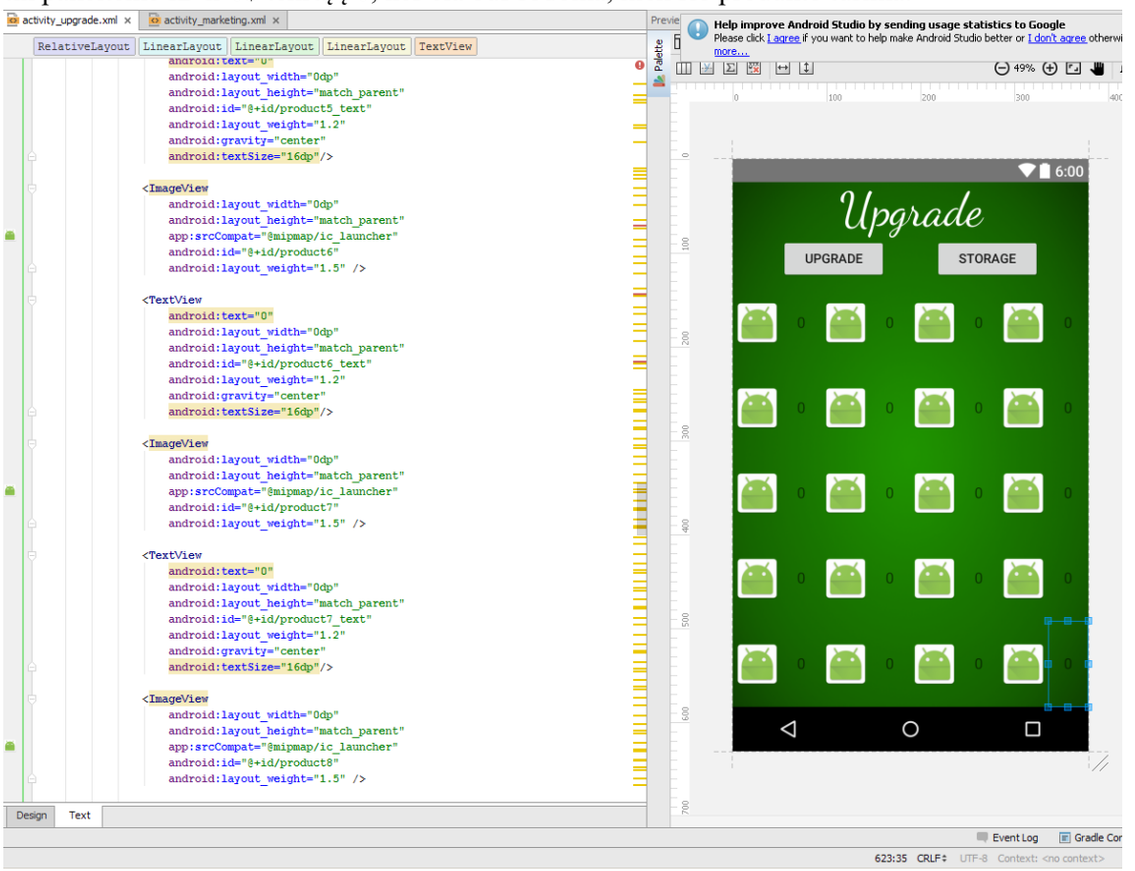


8. Marketing veikloje product_layout išdėstyme susidėliojome vieną eilutę (LinearLayout), kurioje eina nuotrauka ir tekstas kiekvienam produktui. Vienoje eilutėje – vienas produktas. Šioje skiltyje bus rodoma dabartinė produkto kaina ir jo nuotrauka. Padarėme 5 eilutes po 4 produktus.

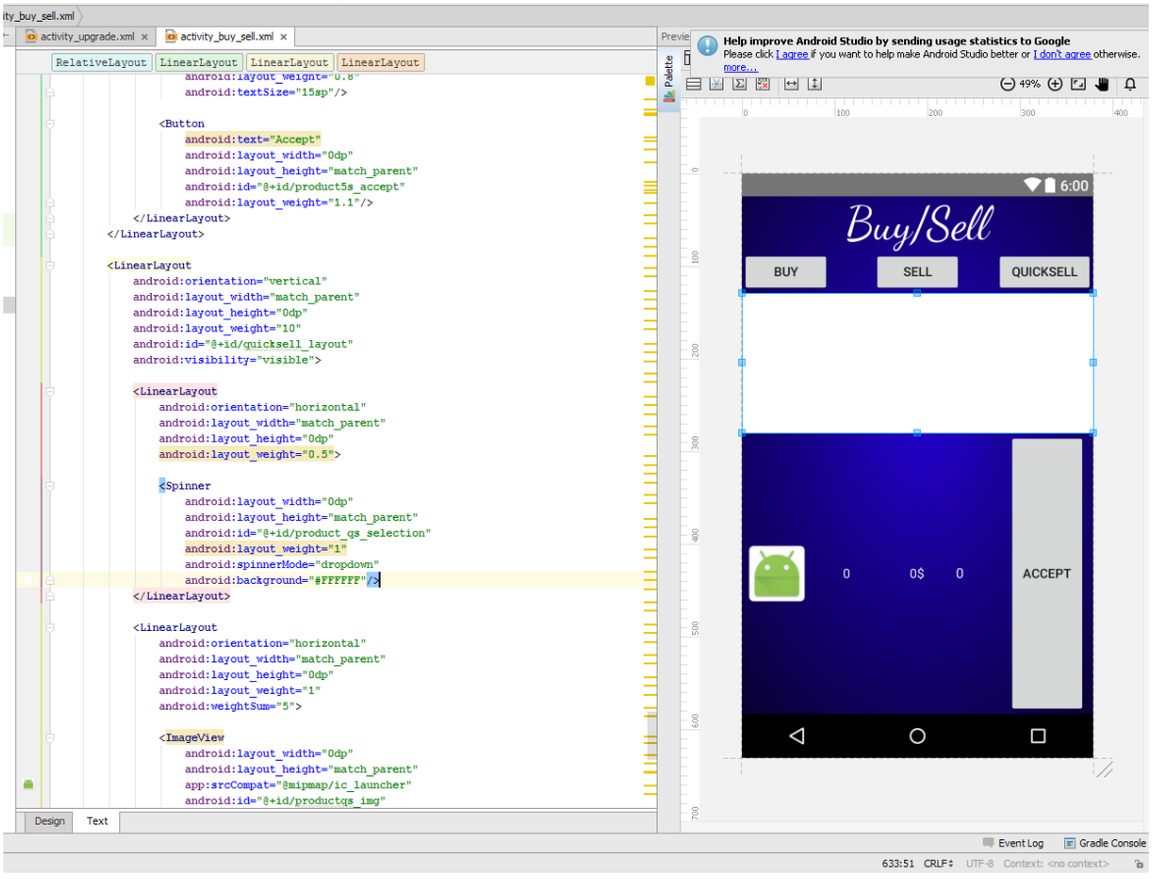




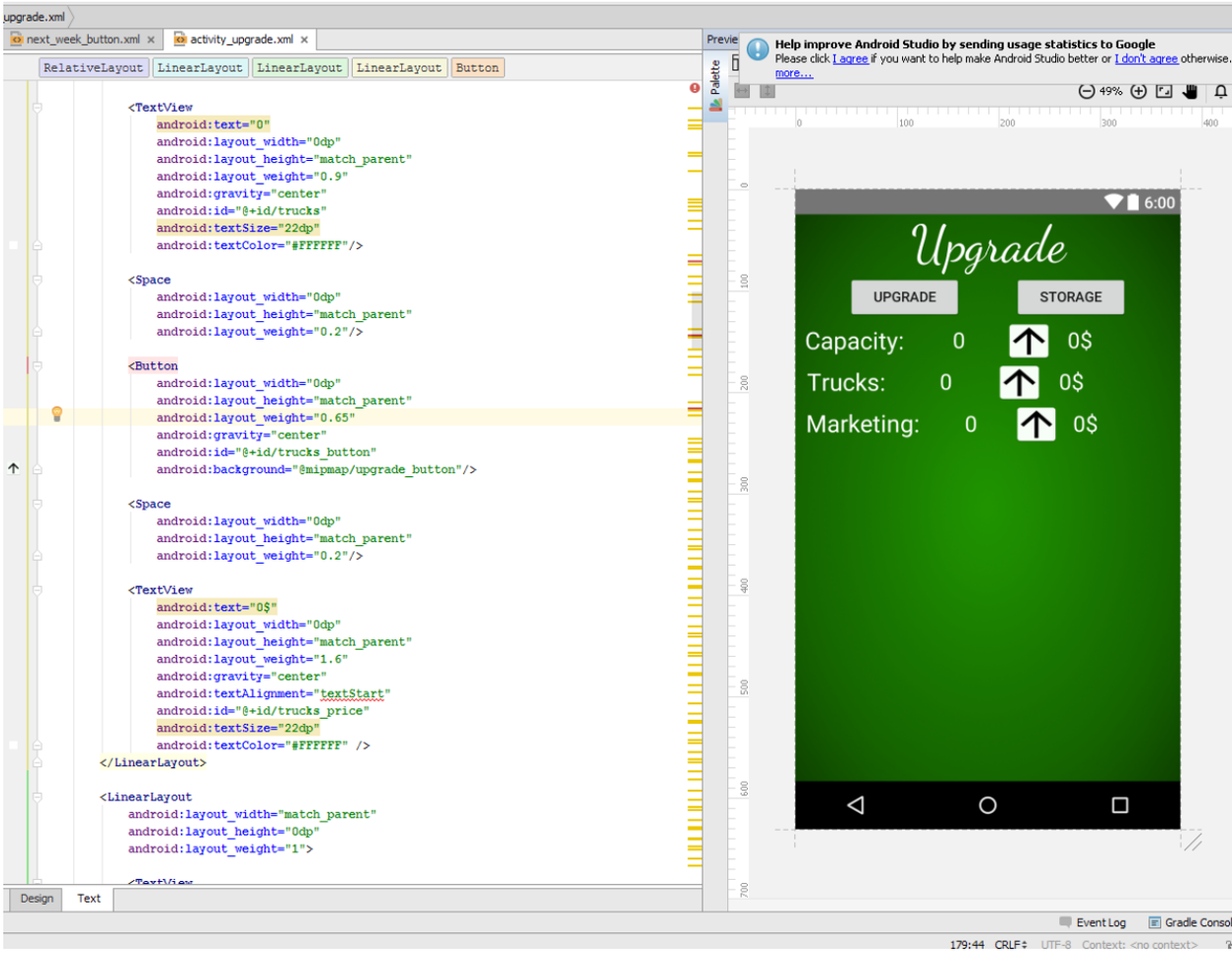
9. Storage veikloje storage_layout išdėstyme įkopijavome tas pačias gaires iš Stats veiklos išdėstymo, tik pakeitėme ID ir 0$ tekstą į 0, nes čia bus rodoma, kiek šio produkto turima.



10. Į buy_sell veiklą quicksell_layout įdėjome *Spinner* - tai objektas, kurį paspaudus atsiveria sąrašas pasirinkimų. Čia bus galima pasirinkti produktą, kurį bus norima parduoti.



11. Į Upgrade veiklą upgrade_layout pakeitėme mygtuko nuotrauką į rodyklę (šios rodyklės nuotrauka buvo rasta tokiu pačiu būdų, kaip ir praeitų mygtukų)



Keturioliktas darbas – Veiklų tekstų pakeitimas į kintamuosius

1. Upgrade veiklos kode prirašėme funkciją LoadProducts, kuri pakeistų produktų kiekių tekstus į turimus kiekius iš duomenų failo.

```
void LoadProducts(SharedPreferences pref)
{
    for (int index = 0; index < 20; ++index)
    {
        products[index] = pref.getInt(product_names[index], 0);
        int prod_id = getResources().getIdentifier("product" + (index+1) + "_text", "id", getPackageName());
        ((TextView) findViewById(prod_id)).setText("" + products[index]);
    }
}
```

2. Marketing veiklos kode prirašėme funkciją, kuri pakeičia produktų kainų tekstus į esamus, ima kintamuosius iš duomenų failo.

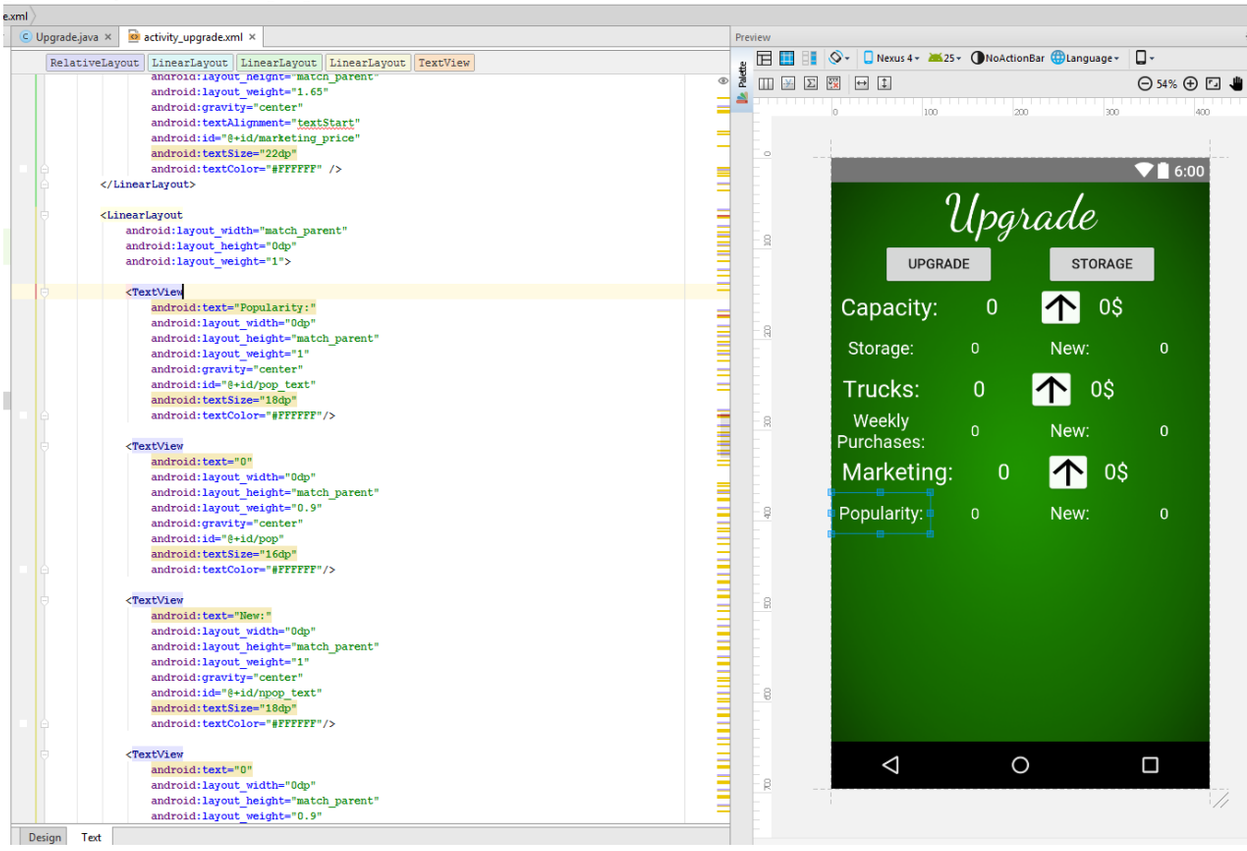
de.java × | Marketing.java ×

```
int w_prod_sold;
TextView w_prod_sold_text;
int w_spendings;
TextView w_spendings_text;
int w_popularity;
TextView w_popularity_text;

View layouts[] = new View[2];
Button layoutbuttons[] = new Button[2];

int product_prices[] = new int[20];
String product_names[] = {"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""};
    void LoadProductPrices(SharedPreferences pref)
    {
        for (int index = 0; index < 20; ++index)
        {
            product_prices[index] = pref.getInt(product_names[index] + "p", 0);
            int prod_id = getResources().getIdentifier("wproduct" + (index+1) + "_text", "id", getPackageName());
            ((TextView) findViewById(prod_id)).setText("" + product_prices[index]);
        }
    }
```

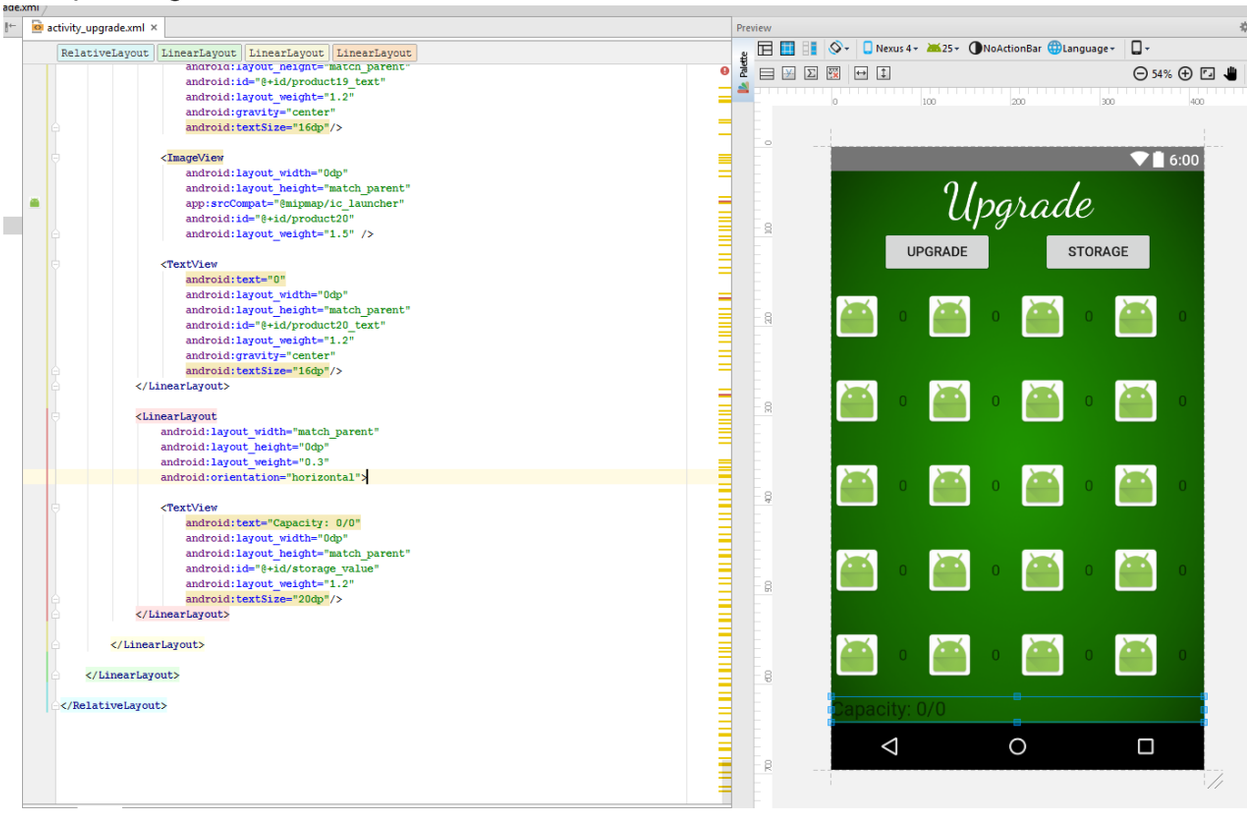
3. Prie šios funkcijos prirašėme kodo, kuris rastų tą produktą, kurio kaina didžiausia, ir tą, kurio kaina mažiausia. Šių produktų nuotraukos spalva pakeičiama į raudoną (mažiausia) ir į žalią (didžiausia)

```
void LoadProductPrices(SharedPreferences pref)
{
    int smallest = 0;
    int smallest_id = 0;
    int biggest = 0;
    int biggest_id = 0;

    for (int index = 0; index < 20; ++index)
    {
        product_prices[index] = pref.getInt(product_names[index] + "p", 0);
        int prod_id = getResources().getIdentifier("wproduct" + (index+1) + "_text", "id", getPackageName());
        ((TextView) findViewById(prod_id)).setText("$" + product_prices[index]);

        if (smallest == 0 || product_prices[index] < smallest)
        {
            smallest = product_prices[index];
            smallest_id = index;
        }

        if (product_prices[index] < biggest)
        {
            biggest = product_prices[index];
            biggest_id = index;
        }
    }

    int smallest_img = getResources().getIdentifier("wproduct" + (smallest_id+1), "id", getPackageName());
    int biggest_img = getResources().getIdentifier("wproduct" + (biggest_id+1), "id", getPackageName());

    ((ImageView) findViewById(smallest_img)).setColorFilter(Color.argb(100, 255, 0, 0));
    ((ImageView) findViewById(biggest_img)).setColorFilter(Color.argb(100, 0, 255, 0));
}
```

4. Upgrade veikloje po kiekvieno patobulinimo eilute pridėjome teksto objektų, kurie nurodo dabartinį parametrą, ir koks bus po patobulinimo.



5. Į Upgrade veiklos storage_layout išdėstymą pridėjome tekstinį objektą, kuris nurodo, kiek produktų turimą ir kiek galima turėti maksimaliai.



6. Į upgrade klasę prirašėme kintamuosius šiems pridėtiems teksto objektams, prie funkcijos ChangeTexts() prirašėme kodą, kuris pakeistų šiuos tekstus ir pridėjome naują kintamąjį „upby“, kuris nurodo, kiek pasikeičia vertė nusipirkus patobulinimą. Kadangi pakeitėme šį kintamąjį, kai deklaruojama klasė, reikia prirašyti dar vieną vertę į skliaustus.

```
public class upgrade
{
    String name;
    int lvl;
    int price;
    int upby;
    TextView lvl_text;
    TextView price_text;
    TextView now_text;
    TextView new_text;
    Button button;

    public upgrade(String n, SharedPreferences pref, int by)
    {
        name = n;
        lvl = pref.getInt(name + "_lvl", 0);
        int upg_id = getResources().getIdentifier(name, "id", getPackageName());
        lvl_text = (TextView) findViewById(upg_id);
        int upg_price_id = getResources().getIdentifier(name + "_price", "id", getPackageName());
        price_text = (TextView) findViewById(upg_price_id);
        int upg_button_id = getResources().getIdentifier(name + "_button", "id", getPackageName());
        button = (Button) findViewById(upg_button_id);
        upby = by;
        int upg_now_id = getResources().getIdentifier(name + "_now", "id", getPackageName());
        now_text = (TextView) findViewById(upg_now_id);
        int upg_new_id = getResources().getIdentifier(name + "_new", "id", getPackageName());
        new_text = (TextView) findViewById(upg_new_id);
    }

    void ChangeTexts()
    {
        lvl_text.setText("" + lvl);
        price_text.setText(price + "$");
        now_text.setText("" + (lvl*upby + upby));
        new_text.setText("" + ((lvl+1)*upby + upby));
    }

    void CalculateLvlPrice() { price = lvl * 10000 + 10000; }

    void PurchaseUpgrade()
    {
        SharedPreferences.Editor prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();
        lvl++;
        prefeditor.putInt(name + "_lvl", lvl);
        prefeditor.commit();
        CalculateLvlPrice();
        ChangeTexts();
    }
}
```

```
upgrades[0] = new upgrade("capacity", pref, 50);
upgrades[1] = new upgrade("trucks", pref, 1);
upgrades[2] = new upgrade("marketing", pref, 100);
```

7. Kadangi populiarumas yra vienintėle vertė, kuri gali kisti be patobulinimo pirkimo, tai ChangeTexts() prirašėme kodą, kuris pakeičia patobulinimų tekstus skirtingai. Prie dabartinės populiarumo vertės pridedama ta vertė, kuria yra padidinama nusipirkus patobulinimą.

```
void ChangeTexts()
{
    lvl_text.setText("" + lvl);
    price_text.setText(price + "$");

    if (name != "marketing")
    {
        now_text.setText("" + (lvl*upby + upby));
        new_text.setText("" + ((lvl+1)*upby + upby));
    }
    else
    {
        SharedPreferences pref = getSharedPreferences("BusinessGuyPrefs", 0);
        now_text.setText("" + (lvl*upby + pref.getInt("popularity", 0)));
        new_text.setText("" + ((lvl+1)*upby + pref.getInt("popularity", 0)));
    }
}
```

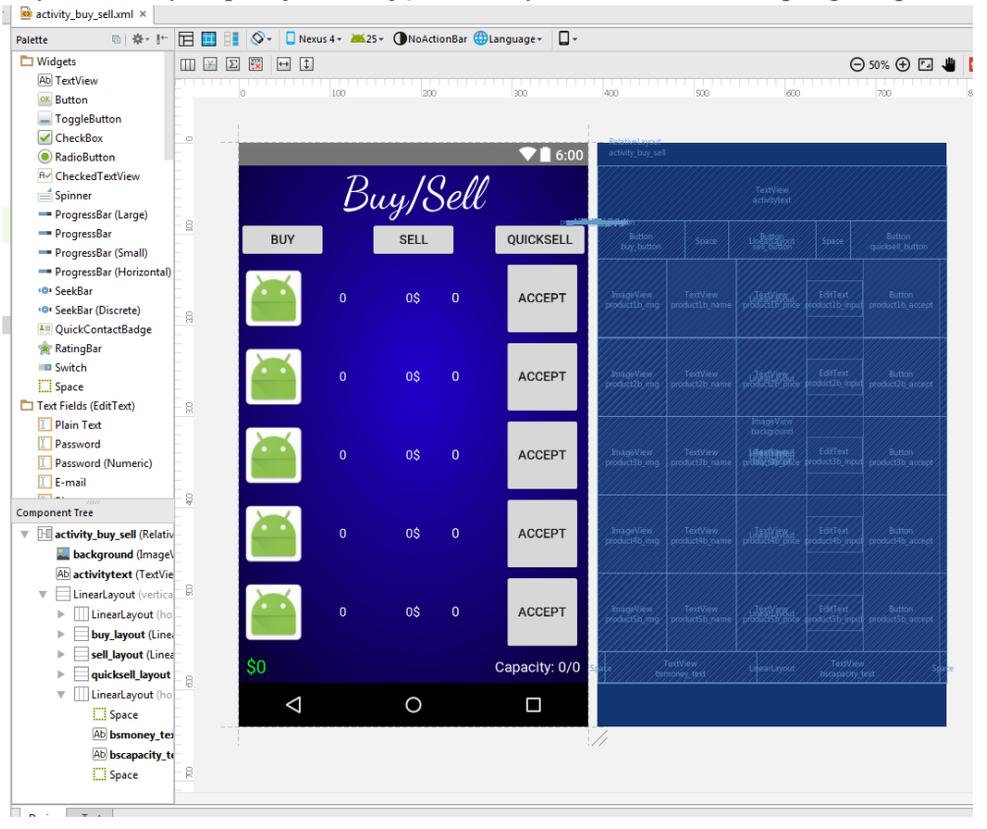
8. Upgrade veikloje dar liko pakeisti Capacity tekstą. Taigi, parašėme tam funkciją, kuri randa šį teksto objektą, apskaičiuoja maksimalų galimą produktų kiekį ir pakeičia tekstą į turimų / max produktų. Kadangi capacity patobulinimas pakeičia maksimalų produktų kiekį, ši funkcija yra vykdoma, kai nusiperkamas patobulinimas.

```
void SetStorageText()
{
    TextView storage_value = (TextView) findViewById(R.id.storage_value);
    int maxProds = upgrades[0].lvl * upgrades[0].upby + upgrades[0].upby;
    storage_value.setText("Capacity: " + totalProds + "/" + maxProds);
}
if (name == "capacity")
    SetStorageText();
```

9. BuySell veikloje prirašėme kodą į LoadProducts, kuris gautų visų produktų turimą kiekį ir maksimalų galimą kiekį.

```
void LoadProducts(SharedPreferences pref)
{
    for (int index = 0; index < 20; ++index)
    {
        products[index] = pref.getInt(product_names[index], 0);
        capacity_current += products[index];
    }

    int upg_lvl = pref.getInt("capacity_lvl", 0);
    capacity = upg_lvl * 50 + 50;
}
```

10. BuySell išdėstyme pridėjome naują LinearLayout, kuriame rodomi pinigai ir produktų kiekis.

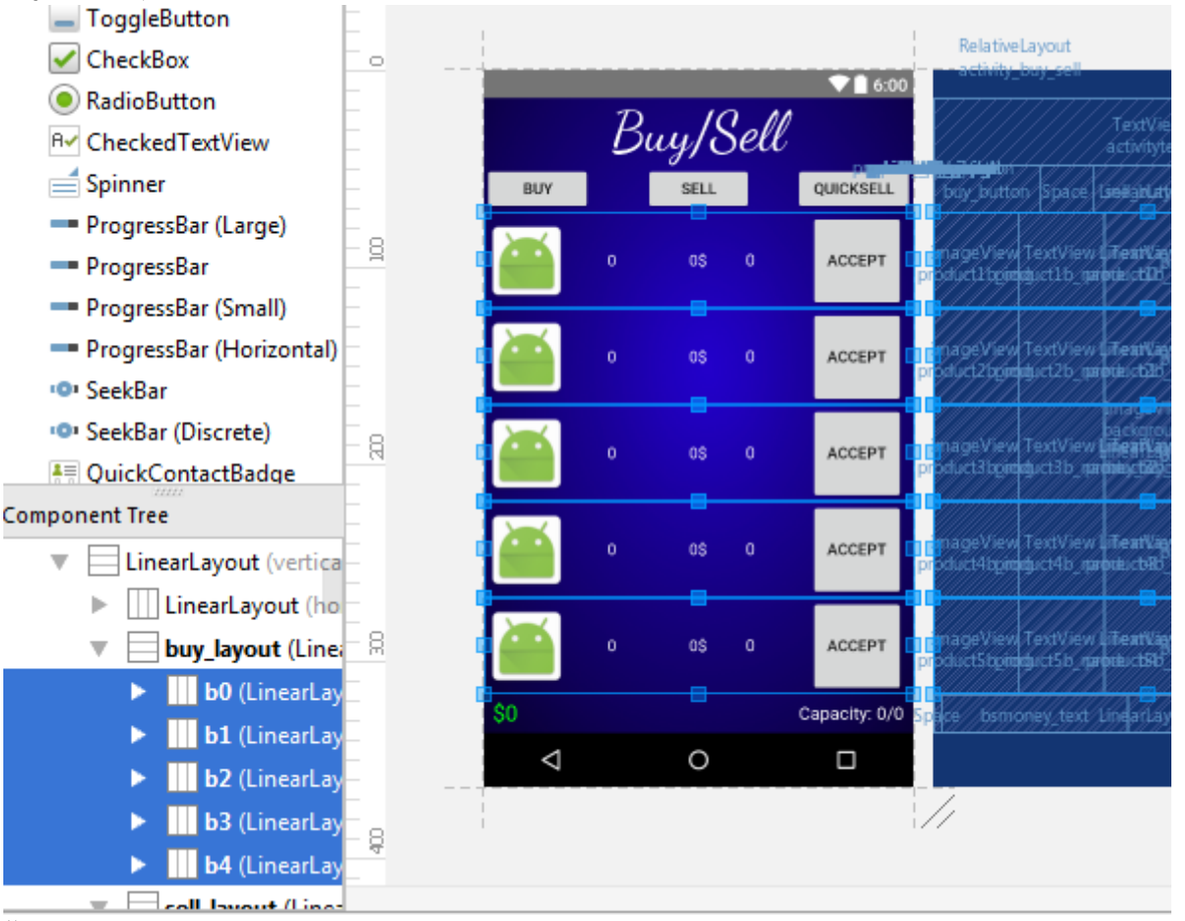


11. Šioje veikloje prirašėme funkciją, kuri rastų šiuos tekstus ir pakeistų juos į reikiamas vertes.

```
void UpdateTexts()
{
    TextView MoneyText = (TextView) findViewById(R.id.bsmoney_text);
    TextView CapacityText = (TextView) findViewById(R.id.bscapacity_text);

    MoneyText.setText(money + "$");
    CapacityText.setText("Capacity: " + capacity_current + "/" + capacity);
}
```

Penkioliktas darbas – Produktų pirkimai, pardavimai

1. BuySell veikloje kiekvieno produkto LinearLayout pavadinimą pakeitėme į b arba s ir skaičių (b – buy, s – sell)



2. Šios veiklos kode sukūrėme vieno produkto klasė, kuri turi kintamuosius, reikiamus vienam produktui: indeksas (skaičius, kuris yra prie anksčiau minėto išdėstymo pavadinimo), id (produkto), kiekis, kaina ir visi šio vieno produkto objektai. Šios klasės konstruktoriuje parašėme kodą, kuris iš duomenų failo ima visus reikiamus kintamuosius ir randa reikiamus objektus. Kad atskirti perkamą produktą nuo parduodamo, naudojamas kintamasis type, kuris yra arba „b“, arba „s“. Jei produkto ID yra –1 (t.y. nėra jokio pasiūlymo), tai produkto eilutė yra paslėpiama.

```
public class bsproduct implements View.OnClickListener
{
    int index;
    int id;
    int count;
    int price;
    View view;
    ImageView image;
    TextView name_view;
    TextView price_view;
    EditText input;
    Button button;
    String type;

    public bsproduct(int ind, String t, SharedPreferences pref)
    {
        id = pref.getInt(t + "product" + ind, -1);
        count = pref.getInt(t + "product_count" + ind, -1);;
        price = pref.getInt(t + "product_price" + ind, -1);;

        int v_id = getResources().getIdentifier(t + ind, "id", getPackageName());
        view = findViewById(v_id);

        String prod = "product" + ind + t + "_";
        int i_id = getResources().getIdentifier(prod + "img", "id", getPackageName());
        image = (ImageView) findViewById(i_id);
        int n_id = getResources().getIdentifier(prod + "name", "id", getPackageName());
        name_view = (TextView) findViewById(n_id);
        int p_id = getResources().getIdentifier(prod + "price", "id", getPackageName());
        price_view = (TextView) findViewById(p_id);
        int in_id = getResources().getIdentifier(prod + "input", "id", getPackageName());
        input = (EditText) findViewById(in_id);
        int b_id = getResources().getIdentifier(prod + "accept", "id", getPackageName());
        button = (Button) findViewById(b_id);

        if (id == -1)
            view.setVisibility(View.GONE);

        type = t;
        index = ind;
    }
```

3. Sukūrėme masyvus perkamiem ir parduodamiem produktam. Padarėme vieną pokytį visose veiklose – pref ir prefeditor (duomenys) padarėme globaliomis kiekvienoje veikloje, kad jas reikėtų deklaruoti tik vieną kartą.

```
bsproduct[] bproducts = new bsproduct[5];
bsproduct[] sproducts = new bsproduct[5];

SharedPreferences pref;
SharedPreferences.Editor prefeditor;
```

4. LoadProducts funkcijoje prirašėme kodą, kuris deklaruotų klasę su kintamaisias (indeksu bei raide, kuri nusako kokio tai tipo produktas, t.y. perkamas ar parduodamas) Šioje vietoje pref nebūtina perleisti, kadangi ji jau padaryta globali.

```
void LoadProducts(SharedPreferences pref)
{
    for (int index = 0; index < 20; ++index)
    {
        products[index] = pref.getInt(product_names[index], 0);
        capacity_current += products[index];
    }

    int upg_lvl = pref.getInt("capacity_lvl", 0);
    capacity = upg_lvl * 50 + 50;

    for (int index = 0; index < 5; ++index)
    {
        bproducts[index] = new bsproduct(index, "b", pref);
        sproducts[index] = new bsproduct(index, "s", pref);
    }
}
```

5. Į produkto klasė parašėme onClick funkcija (kuri veikia dėl to, nes prie šio produkto klasės parašėme implenets View.OnClickListener), kuri prideda funkcionalumo mygtukams. Paspaudus mygtuką:
    a. Imama vertė iš teksto įvedimo laukelio. Tai kiekis, kurį norimą pirkti/parduoti.
    b. Patikrinama, ar įvestas kiekis neviršija maksimalaus.
    c. Nustatoma produkto kaina (išmokama arba mokama)
    d. Patikrinama, ar tai perkamas, ar parduodamas objektas.
    e. Jei tai perkamas objektas, tai patikrinama, ar užtenka pinigų.
    f. Šio produkto kiekis sumažinamas tiek, kiek jo pirkta/parduota. Atimami/pridedami pinigai, į duomenų failą išsaugomi kintamieji.

```
@Override
public void onClick(View v)
{
    int count_input = Integer.parseInt(input.getText().toString());

    if (count_input <= count)
    {
        int t_price = count_input * price;

        if (type == "b")
        {
            if (price >= t_price)
            {
                count -= count_input;
                money -= t_price;
                products[id] += count_input;
                prefeditor.putInt("money", money);
                prefeditor.putInt(product_names[id], products[id]);
                prefeditor.putInt(type + "product_count" + index, count);
                UpdateProduct();
            }
        }
        else
        {
            count -= count_input;
            money += t_price;
            products[id] -= count_input;
            prefeditor.putInt("money", money);
            prefeditor.putInt(product_names[id], products[id]);
            prefeditor.putInt(type + "product_count" + index, count);
            UpdateProduct();
        }
    }
}
```

6. Įjungėme programą patikrinti, ar veikia. Taigi, kadangi pasiūlymai kode dar nesuprogramuoti, tai viskas tuščia.

7. Parašėme funkciją, kuri patikrintų, ar kiekis nelygus 0. Jei lygus 0, tai produkto eilutė paslėpiama. Jei kiekis nelygus 0, tai pakeičiamas kiekio tekstas (kur name_view, ten kiekis. Čia pavadinimo klaida) Šios funkcijos vykdymą pridėjome į mygtuko paspaudimo funkciją.

```
public void UpdateProduct()
{
    if (count == 0)
        view.setVisibility(View.GONE);
    else
    {
        name_view.setText("" + count);
        price_view.setText("" + price);
    }
}
```

```
@Override
public void onClick(View v)
{
    int count_input = Integer.parseInt(input.getText().toString());

    if (count_input <= count_input)
    {
        int t_price = count_input * price;

        if (type == "b")
        {
            if (price >= t_price)
            {
                count -= count_input;
                money -= t_price;
                products[id] += count_input;
                prefeditor.putInt("money", money);
                prefeditor.putInt(product_names[id], products[id]);
                prefeditor.putInt(type + "product_count" + index, count);
                UpdateProduct();
            }
        }
        else
        {
            count -= count_input;
            money += t_price;
            products[id] -= count_input;
            prefeditor.putInt("money", money);
            prefeditor.putInt(product_names[id], products[id]);
            prefeditor.putInt(type + "product_count" + index, count);
            UpdateProduct();
        }
    }
```

8. Veikloje prirašėme veiksmų kintamąjį. Jis nurodo, kiek veiksmų dar galima atlikti šią savaitę.

```
int actions_left;

@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_buy_sell);
    layouts[0] = findViewById(R.id.buy_layout);
    layoutbuttons[0] = (Button) findViewById(R.id.buy_button);
    layouts[1] = findViewById(R.id.sell_layout);
    layoutbuttons[1] = (Button) findViewById(R.id.sell_button);
    layouts[2] = findViewById(R.id.quicksell_layout);
    layoutbuttons[2] = (Button) findViewById(R.id.quicksell_button);

    for (int index = 0; index < 3; ++index)
    {
        layoutbuttons[index].setOnClickListener(this);
    }

    pref = getSharedPreferences("BusinessGuyPrefs", 0);
    prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();

    money = pref.getInt("money", 0);
    actions_left = pref.getInt("actions_left", 0);
```

9. Kai produtkas perkamas ar parduodamas, tai veiksmai sumažinami vienu. Šią eilutę parašėme prie mygtuko paspaudimo funkcijos. (Vietoj price >= t_price turi būti money >= t_price)

```
@Override
public void onClick(View v)
{
    int count_input = Integer.parseInt(input.getText().toString());

    if (count_input <= count && actions_left != 0)
    {
        int t_price = count_input * price;

        if (type == "b")
        {
            if (price >= t_price && count_input <= (capacity - capacity_current))
            {
                count -= count_input;
                money -= t_price;
                products[id] += count_input;
                prefeditor.putInt("money", money);
                prefeditor.putInt(product_names[id], products[id]);
                prefeditor.putInt(type + "product_count" + index, count);
                actions_left--;
                prefeditor.putInt("actions_left", actions_left);
                UpdateProduct();
            }
        }
        else
        {
            count -= count_input;
            money += t_price;
            products[id] -= count_input;
            prefeditor.putInt("money", money);
            prefeditor.putInt(product_names[id], products[id]);
            prefeditor.putInt(type + "product_count" + index, count);
            actions_left--;
            prefeditor.putInt("actions_left", actions_left);
            UpdateProduct();
        }
    }
}
```

10. Grįžome į Upgrade veiklą. Prie patobulinimo pirkimo funkcijos prirašėme kodą, kuris padidina veiksmų kiekį vienu, kai nusiperkamas trucks patobulinimas.

```
void PurchaseUpgrade()
{
    lvl++;
    prefeditor.putInt(name + "_lvl", lvl);
    prefeditor.commit();
    CalculateLvlPrice();
    ChangeTexts();

    if (name == "capacity")
    {
        int actions_left = pref.getInt("actions_left", 0);
        prefeditor.putInt("actions_left", actions_left+1);
    }
}
}
```

11. BuySell veikloje pridėjome teksto objektą, kuris rodys kiek veiksmų liko. Veiklos kode parašėme kodą, kuris pakeičia šį tekstą. Kai nusiperkamas arba parduodamas produktas, šis tekstas yra pakeičiamas.


```
int actions_left;
TextView actions_text;
```

```
money = pref.getInt("money", 0);

actions_text = (TextView) findViewById(R.id.actionstext);
actions_left = pref.getInt("actions_left", 0);
actions_text.setText("Actions left: " + actions_left);
```

```
if (type == "b")
{
    if (price >= t_price && count_input <= (capacity - capacity_current))
    {
        count -= count_input;
        money -= t_price;
        products[id] += count_input;
        prefeditor.putInt("money", money);
        prefeditor.putInt(product_names[id], products[id]);
        prefeditor.putInt(type + "product_count" + index, count);
        actions_left--;
        prefeditor.putInt("actions_left", actions_left);
        actions_text.setText("Actions left: " + actions_left);
        UpdateProduct();
    }
}
else
{
    count -= count_input;
    money += t_price;
    products[id] -= count_input;
    prefeditor.putInt("money", money);
    prefeditor.putInt(product_names[id], products[id]);
    prefeditor.putInt(type + "product_count" + index, count);
    actions_left--;
    prefeditor.putInt("actions_left", actions_left);
    actions_text.setText("Actions left: " + actions_left);
    UpdateProduct();
}
```

12. Produkto klasėje prirašėme kodą, kuris mygtukui pridėtų onClickListener, tai reiškia, jog mygtukas vykdys OnClick() funkciją.

```
int b_id = getResources().getIdentifier(prod + "accept", "id", getPackageName());
button = (Button) findViewById(b_id);
button.setOnClickListener(this);
```

13. Prie pirkimo/pardavimo funkcijos prirašėme eilutę, kuri pakeistų dabartinių turimų produktų kintamąjį (pridėtų arba atimtų)

```
capacity current += count_input;
```

Šešioliktas darbas – Greitas pardavimas, pirkimo/pardavimo testavimas

1. BuySell veiklos kode pridėjome Spinner kintamąjį, jį priskyrėme pagal ID ir parašėme kodo, kuris duotų šiam objektui sąrašą (šiuo atveju prekių sąrašas)

```
Spinner qs_spinner;

@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_buy_sell);
    layouts[0] = findViewById(R.id.buy_layout);
    layoutbuttons[0] = (Button) findViewById(R.id.buy_button);
    layouts[1] = findViewById(R.id.sell_layout);
    layoutbuttons[1] = (Button) findViewById(R.id.sell_button);
    layouts[2] = findViewById(R.id.quicksell_layout);
    layoutbuttons[2] = (Button) findViewById(R.id.quicksell_button);

    for (int index = 0; index < 3; ++index)
    {
        layoutbuttons[index].setOnClickListener(this);
    }

    pref = getSharedPreferences("BusinessGuyPrefs", 0);
    prefeditor = getSharedPreferences("BusinessGuyPrefs", 0).edit();

    money = pref.getInt("money", 0);

    actions_text = (TextView) findViewById(R.id.actionstext);
    actions_left = pref.getInt("actions_left", 0);
    actions_text.setText("Actions left: " + actions_left);

    LoadProducts();
    UpdateTexts();

    qs_spinner = (Spinner) findViewById(R.id.product_qs_selection);
    ArrayAdapter<String> qs_adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item, product_names);
    qs_spinner.setAdapter(qs_adapter);
}
```

2. Prie veiklos pagrindinės klasės prirašėme implements AdapterView.OnItemSelectedListener. Šiam implement‘ui buvo sukurtos dvi funkcijos. Viena iš jų mum reikalinga – bus vykdoma, pasirinkus objektą. Spinner kintamam pridėjome šią funkciją su setOnItemSelectedListener (panašiai, kaip ir su mygtuku)

```
public class BuySell extends AppCompatActivity implements View.OnClickListener, AdapterView.OnItemSelectedListener {

    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {

    }

    @Override
    public void onNothingSelected(AdapterView<?> parent)
    {

    }
qs_spinner.setOnItemSelectedListener(this);
```

3. Sukūrėme naują klasę, panašią į bsproduct klasę. Ši klasė qsproduct bus skirta vienam objektui, t.y. produktui, kuris bus greitai parduodamas. Jis turi kintamuosius, panašius į paprasto produkto: id, kiekis, kaina ir produkto eilutės kintamieji (nuotrauka, tekstai, įvedimo laukelis, mygtukas) Jam taip parašėme panašią funkciją į bsproduct – UpdateProduct(), kuri atnaujina teksto laukelius bei id. Šiai funkcijai visada perleisim 0, nes yra tik viena tokio produkto eilutė.

```
qsprod = new qsproduct();
  public class qsproduct
  {
      int id = 0;
      int count;
      int price;
      ImageView image;
      TextView count_view;
      TextView price_view;
      EditText input;
      Button button;

      public qsproduct()
      {
          image = (ImageView) findViewById(R.id.productqs_img);
          count_view = (TextView) findViewById(R.id.productqs_count);
          price_view = (TextView) findViewById(R.id.productqs_price);
          input = (EditText) findViewById(R.id.productqs_input);
          button = (Button) findViewById(R.id.productqs_accept);

          UpdateProduct(0);
      }

      public void UpdateProduct(int i)
      {
          id = i;
          count = pref.getInt(product_names[id], 0);
          count_view.setText("" + count);
          price = pref.getInt(product_names[id] + "p", 0);
          price_view.setText(price + "$");
      }
  }

  bsproduct[] bproducts = new bsproduct[5];
  bsproduct[] sproducts = new bsproduct[5];
  qsproduct qsprod;
```

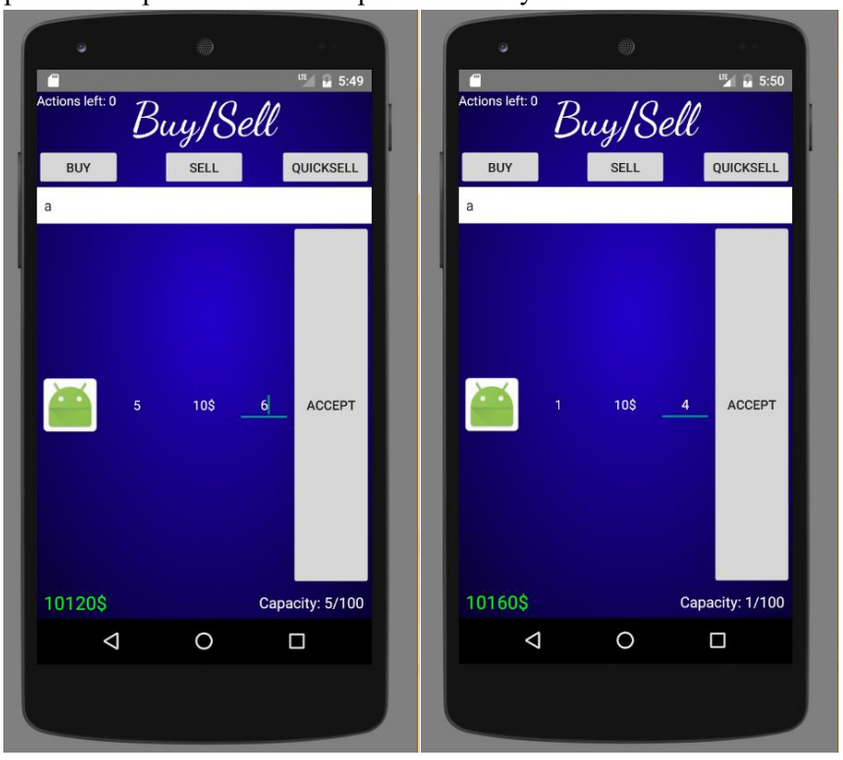
4. Šiai klasei parašėme OnClick funkciją, kuri nuskaito kintamąjį iš teksto įvedimo laukelio ir patikrina, ar įvestas kiekis neviršija turimo produkto kiekio. Jei neviršija, produktas yra parduodamas ir suteikiami pinigai už jį, numinusuojamas parduotų produktų kiekis, atnaujinami tekstai.

```
            button.setOnClickListener(this);

            UpdateProduct(0);
        }

        public void UpdateProduct(int i)
        {
            id = i;
            count = pref.getInt(product_names[id], 0);
            count_view.setText("" + count);
            price = pref.getInt(product_names[id] + "p", 0);
            price_view.setText(price + "$");
        }

        @Override
        public void onClick(View v)
        {
            int count_input = Integer.parseInt(input.getText().toString());

            if (count >= count_input)
            {
                money += price * count_input;
                prefeditor.putInt("money", money);
                count -= count_input;
                prefeditor.putInt(product_names[id], count);
                capacity_current -= count_input;
                prefeditor.commit();
                UpdateTexts();
                UpdateProduct(id);
            }
        }
    }
prefeditor.commit();
```

5. Kad išbandyti, į kodą parašėme testavimui reikalingas eilutes – pridėjome sau produktų, nustatėme produktų kainą

```
prefeditor.putInt(product_names[0], 5);
prefeditor.putInt(product_names[0] + "p", 10);
prefeditor.commit();
```

6. Nuėję į Quicksell skiltį, pabandėme įvesti per didelį kiekį ir parduoti, ko nepavyko padaryti ir tada pabandėme parduoti 4 kiekius produkto. Pavyko.



7. Po to pasiruošėme testuoti pardavimą/pirkimą, ko dar nebandėme daryti. Prieš tai atlikome porą pakeitimų: bsproduct klasėje pakeitėme duomenų pavadinimus iš (tipas + „product“ + tekstas + indeksas) į (tipas + „product“ + indeksas + tekstas). UpdateProduct funkcijoje pakeitėme pinigų teksto formatą, kad būtų dolerio ženklas gale pinigų.

```
public bsproduct(int ind, String t)
{
    id = pref.getInt(t + "product" + ind, -1);
    count = pref.getInt(t + "product" + ind + "_count", -1);;
    price = pref.getInt(t + "product" + ind + "_price", -1);;

    public void UpdateProduct()
    {
        if (count == 0)
            view.setVisibility(View.GONE);
        else
        {
            name_view.setText("" + count);
            price_view.setText(price + "$");
//              image.setImageResource(R.drawable.test);
        }
    }
}
```

8. Parašėme eilutes testiniams duomenims (pakeistas žaidėjo turimas produkto kiekis ir įdėti pirkimo ir pardavimo pasiūlymai)

```
prefeditor.putInt(product_names[0], 10);
prefeditor.putInt("sproduct0", 0);
prefeditor.putInt("sproduct0_count", 100);
prefeditor.putInt("sproduct0_price", 15);
prefeditor.putInt("bproduct0", 0);
prefeditor.putInt("bproduct0_count", 100);
prefeditor.putInt("bproduct0_price", 10);
prefeditor.commit();
```

9. Įjungėme pažiūrėti, ar užkrovė pasiūlymus. Iš tiesų – turėjome pasiūlymą, tačiau neturėjome veiksmų.

10. Parašėme dar vieną testinę eilutę, kuri pakeistų žaidėjo likusių veiksmų kiekį.

```
prefeditor.putInt("actions_left", 10);
```

11. Išbandėme parduoti ir pirkti produktus, įvesti per dideles vertes. Viskas pavyko.

Septynioliktas darbas – Verčių įvedimas, savaitės progresijos pagrindas

1. Visose veiklose kur tai naudojama, įvedėme vertes į masyvus – produktų pavadinimai, jų pradinės kainos.

```
String product_names[] = { "Food", "Electronics", "Gold", "Oil",
        "Clothes", "Sports Goods", "Furniture", "Construction Materials",
        "Music Goods", "Crockery", "Books", "Hunting Equipment",
        "Toys", "Medicine", "Gas", "Resources",
        "Plants", "Tools", "Household Items", "Chemicals"};
int product_prices[] = {10, 200, 100, 30,
        50, 30, 110, 90,
        60, 20, 20, 70,
        15, 80, 20, 120,
        25, 25, 10, 80};
```

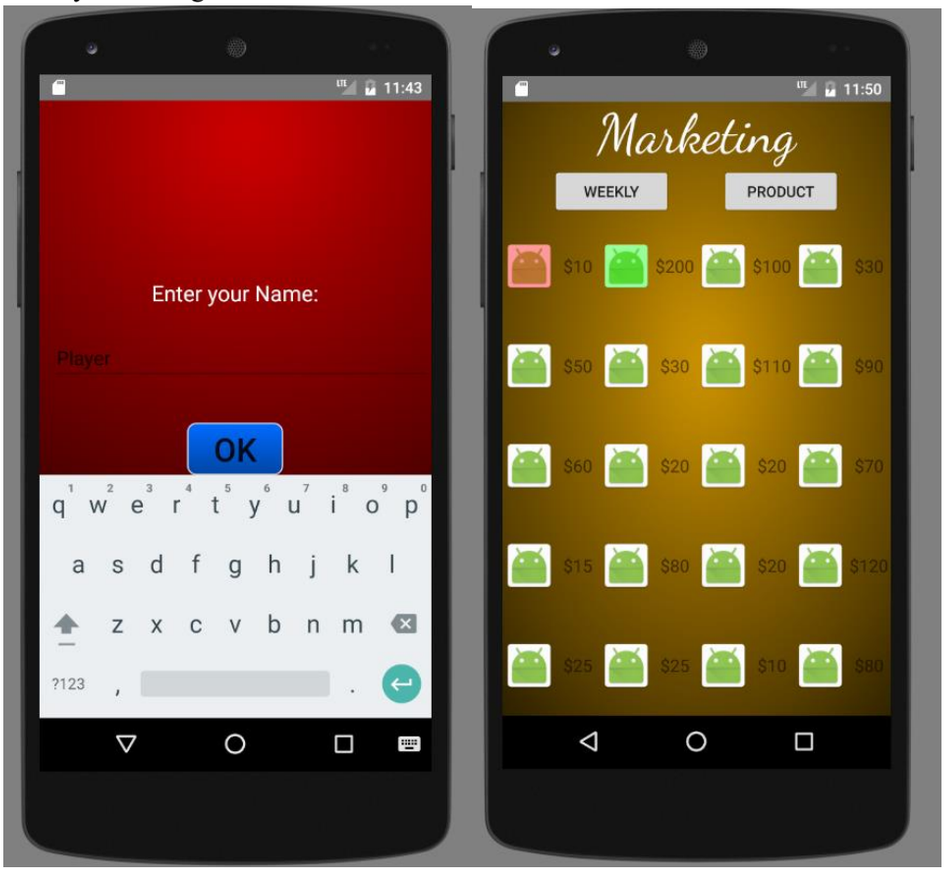
2. Pradinėje veikloje (vardo įvedimas) parašėme kodo eilutes, kurios produkto kainą nustato į pradinę vertę nurodyta masyve.

```
for (int index = 0; index < 20; ++index)
{
    prefeditor.putInt(product_names[index] + "p", product_prices[index]);
}
```

3. Padarėme keitimą Marketing veiklos kode, kuris teisingai nustatytu didžiausią kainą turinti produktą.

```
    if (product_prices[index] > biggest)
    {
        biggest = product_prices[index];
        biggest_id = index;
    }
}
```

4. Su anksčiau parašytu funkcijų pagalba, pradėjome žaidimą iš naujo ir patikrinome, ar kainos buvo nustatytos teisingai. Veikė.



5. Pradinėje veikloje įvedėme porą pradinių duomenų – populiarumas ir veiksmų skaičius.

```
prefeditor.putInt("popularity", 100);
prefeditor.putInt("actions_left", 2);
```

6. Pagrindinėje veikloje įvedėme savaitės kintamąjį. Jis bus saugomas duomenų faile. Taip pat parašėme funkciją, kuri pakeistų savaitę į sekančią.

```
int week = 0;
week = pref.getInt("week", 0);
```

7. Į OnClick funkciją prirašėme dar vieną įmanoma mygtuko paspaudimo atvejį – kitos savaitės mygtukas. Tam mygtukui pridėjome OnClickListener.

```
public void AdvanceWeek()
{
    week++;
    prefeditor.putInt("week", week);


    prefeditor.commit();
}
```

8. Toliau dirbome prie kitos savaitės perėjimo. Pradėjome nuo pirmo žingsnio – populiarumo pakeitimo. Populiarumas imamas iš duomenų failo. Jeigu panaudoti visi įmanomi veiksmai, tai populiarumas padidinamas 10čia, o jeigu nepadarytas nei vienas – sumažinimas 10čia. Populiarumas išsaugomas duomenų faile.

```
((Button) findViewById(R.id.next_week_button)).setOnClickListener(this);
case R.id.next_week_button:
    AdvanceWeek();
    break;
```

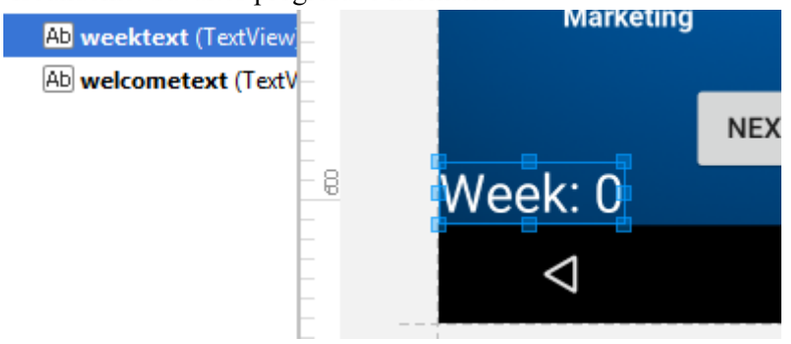
9. Parašėme patikrinimą, ar žaidimas pralaimėtas. Jei pinigų mažiau, negu 0, tai žaidimas pradedamas iš naujo. WipeData() funkciją papildėme finish() kodo eilute, kuri uždaro pagrindinę veiklą (į ją grįžti negalima, kol ji vėl nepradedama)

```
public void AdvanceWeek()
{
    week++;
    prefeditor.putInt("week", week);

    int popularity = pref.getInt("popularity", 0);

    // No actions done = -popularity / All actions done = +popularity
    int actions_max = 2 + pref.getInt("trucks_lvl", 0);
    int actions_left = pref.getInt("actions_left", 0);

    if (actions_left == actions_max)
    {
        popularity -= 10;
    }
    else if (actions_left == 0)
    {
        popularity += 10;
    }

    prefeditor.putInt("popularity", popularity);
```

10. Pridėjome naują teksto objektą į pagrindinę veiklą, kuris rodys dabartinę savaitę. Šį kintamąjį įrašėme į veiklos kodą, parašėme eilutę jam rasti ir prirašėme į UpdateTexts() funkciją eilutę, kuri pakeičia šį tekstą.

```
    if (money < 0)
    {
        WipeData();
    }

void WipeData()
{
    prefeditor.clear();
    prefeditor.commit();
    startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
    finish();
}
```

11. Išbandėme. Savaitės progresavo normaliai.



12. Priėjome dar vieną žingsnį savaites progresavime – jeigu savaitės skaičius dalus iš 5 be liekanos, tada nuskaičiuojami mokesčiai iš žaidėjo pinigų. Šiems mokesčiams nustatyti sugalvojome formulę $100\$ + ((trucks\_lvl + capacity\_lvl + popularity\_lvl)^2 * 50\$)$

```
// Week % 5 == 0, then pay taxes
if (week % 5 == 0)
{
    int taxes = (int) (100 + pow((pref.getInt("trucks_lvl", 0) + pref.getInt("capacity_lvl", 0) + pref.getInt("marketing_lvl", 0)), 2) * 50);
    money -= taxes;
    prefeditor.putInt("money", money);
}
```

Aštuonioliktas darbas – Tolesnis savaičių progresavimas

1. Prieš tęsiant darbą, nusprendėme padaryti vieną pakeitimą – pinigų sumas daryti tikslesnes, t.y. per kablelį. Ankščiau naudojome int, o dabar naudosime float, kad būtų skaičių po kablelių. Pakeitėme visas kodo eilutes visoje programoje, susijas su pinigais (pakeitėme iš int į float)

```
public float money = 0;   prefeditor.putFloat("money", 1000);
price = pref.getFloat(t + "product" + ind + "_price", -1);;
```

2. Toliau dirbome prie pirkimo/pardavimo pasiūlymų pridėjimo. Iš pradžių parašėme ciklą, kuris panaikintų visus dabartinius pasiūlymus (kadangi sekančią savaitę jų įmanoma gauti mažiau)

```
// Add Offers
for (int index = 0; index < 5; ++index)
{
    prefeditor.putInt("bproduct" + index, -1);
    prefeditor.putInt("sproduct" + index, -1);
    prefeditor.putInt("bproduct" + index + "_count", -1);
    prefeditor.putInt("sproduct" + index + "_count", -1);
    prefeditor.putFloat("bproduct" + index + "_price", -1);
    prefeditor.putFloat("sproduct" + index + "_price", -1);
}
```

3. Toliau parašėme kodo eilutes, kurios:
    a. Nustato minimalų ir maksimalų pasiūlymų kiekį pagal populiarumą (kas 200 pop. +1 min, kas 100 pop. +1 max)
    b. Toliau parašėme kodo eilutes, kurios atsitiktinai ima skaičių tarp minimalaus ir maksimalaus kiekio.
    c. Toliau parašėme kitus apribojimus – minimalūs ir maksimalūs įmanomi kiekiai ir nepelningumo ir pelningumo maksimumai procentais (kiek kaina gali būti didesnė arba mažesnė lūžio taško (nei pelninga, nei ne – t.y. nieko neuždirbama) ribos).

```
int offer_min = 1 + (popularity / 200);
int offer_max = 2 + (popularity / 100);

Random random = new Random();
int offers_b = random.nextInt((offer_max - offer_min + 1)) + offer_min;
int offers_s = random.nextInt((offer_max - offer_min + 1)) + offer_min;

int min_count = 10 + popularity / 20;
int max_count = 50 + popularity / 10;

float max_price_swing = (2f + popularity / 100f) / 100f;
float max_nonprofit = (10f - popularity / 200f) / 100f;
```

4. Toliau parašėme ciklą pirkimo ir pardavimo pasiūlymams atskirai. Pradėjome nuo pirkimo:
    a. Nustatoma prekės ID atsitiktinai (nuo 0 iki 19);
    b. Nustatomas kiekis nuo minimalaus iki maksimalaus atsitiktinai;
    c. Nustatoma minimali ir maksimali kaina pagal pelningumo ir nepelningumo maksimumus, o iš to nustatoma tikra kainą atsitiktinai tarp šios ribos. Šis skaičius suapvalinamas šimtainių tikslumu.
    d. Šie duomenys įraošmi į duomenų failą.

```
for (int index = 0; index < offers_b; ++index)
{
    int p_id = random.nextInt(20);
    int p_count = random.nextInt(max_count + 1) + min_count;
    float base_price = pref.getFloat(product_names[index] + "p", 0);

    float p_min = base_price * (1f - max_price_swing);
    float p_max = base_price * (1f + max_nonprofit);
    float p_price = random.nextFloat() * (p_max - p_min) + p_min;

    p_price = Float.parseFloat(String.format("%.2f", p_price));

    prefeditor.putInt("bproduct" + index, p_id);
    prefeditor.putInt("bproduct" + index + "_count", p_count);
    prefeditor.putFloat("bproduct" + index + "_price", p_price);
}
```

5. Pardavimo ciklas mažai kuo skiriasi, tiesiog sukeisti nepelningumo ir pelningumo maksimumai, kadangi čia pardavimo operacija.

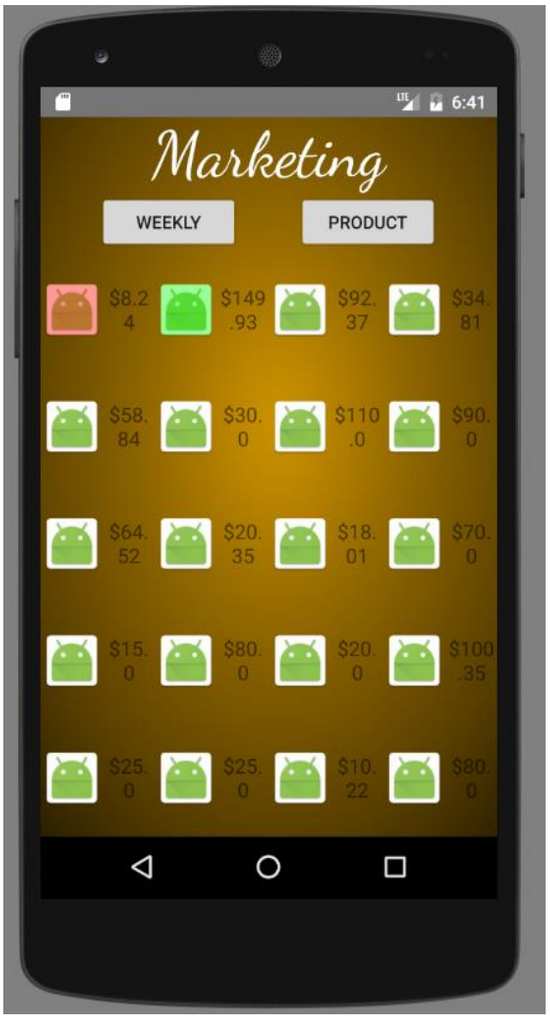
```
for (int index = 0; index < offers_s; ++index)
{
    int p_id = random.nextInt(20);
    int p_count = random.nextInt(max_count + 1) + min_count;
    float base_price = pref.getFloat(product_names[index] + "p", 0);

    float p_min = base_price * (1f - max_nonprofit);
    float p_max = base_price * (1f + max_price_swing);
    float p_price = random.nextFloat() * (p_max - p_min) + p_min;

    p_price = Math.round(p_price * 10) / 10;

    p_price = Float.parseFloat(String.format("%.2f", p_price));

    prefeditor.putInt("sproduct" + index, p_id);
    prefeditor.putInt("sproduct" + index + "_count", p_count);
    prefeditor.putFloat("sproduct" + index + "_price", p_price);
}
```

6. Išbandėme savaitės progresiją. Gavome pasiūlymus.

7. Toliau reikėjo prieiti prie kito žingsnio – kainų keitimas. Tai vyksta tokia tvarka:
    a. Sukuriamas masyvas 20čiai produktų, visų vertės nustatomos į „false“;
    b. Maksimalus pakitimų kiekis – 10. Jis išrenkamas atsitiktinai nuo 0 iki 10;
    c. Toliau cikle išrenkamas skaičius nuo 0 iki 19 (tai – produkto ID). Jeigu šio produkto kaina jau buvo keista, tai daromas sekantis keitimas, jei ne tai;
    d. Masyve pažymima, kad pakito produkto kaina. Iš duomenų imama dabartinė produkto kaina ir nauja minimali ir maksimali įmanoma kaina (gali pakisti tik 20%).
    e. Tarp minimumo ir maksimumo nustatoma nauja vertė. Jinai yra suapvalinama ir išsaugota į duomenų failą.

```
// Change Prices
Boolean[] changed = new Boolean[20];

for (int index = 0; index < 20; ++index)
{
    changed[index] = false;
}

int changes = random.nextInt(11);

for (int index = 0; index < changes; ++index)
{
    int p_id = random.nextInt(20);

    if (!changed[p_id])
    {
        changed[index] = true;

        float p_price = pref.getFloat(product_names[p_id] + "p", 0);

        float p_min = p_price * 0.8f;
        float p_max = p_price * 1.2f;

        p_price = random.nextFloat() * (p_max - p_min) + p_min;

        p_price = Float.parseFloat(String.format("%.2f", p_price));

        prefeditor.putFloat(product_names[p_id] + "p", p_price);
    }
}
```

8. Išbandėme. Kainos iš tiesų pasikeitė po savaitės progresavimo.



9. Kadangi žaidimui pradedant reikia gauti pasiūlymus, pradėjus žaidimą reikia progresuoti savaitę, tačiau nepakeisti kai kurių duomenų. Taigi, parašėme tikrinimus, kad nemažėtų populiarumas dėl veiksmų, jei tai būtent toks progresavimas ir kad nebūtų keičiamos kainos.

```
if (actions_left == actions_max && week != 1)
{
    popularity -= 10;

    if (popularity < 0)
        popularity = 0;
}
```

```
if (week == 1)
    changes = 0;
```

```
if (week == 0)
    AdvanceWeek();
```

10. Kadangi aritmetiniai veiksmai su float yra netikslūs (pvz. 2.0 – 1.0 gali tapti 0.99999999999), tai po visų aritmetinių veiksmų reikalingas apvalinimas.

```
money -= t_price;
money = Float.parseFloat(String.format("%.2f", money));
```

11. Po pirkimo/pardavimo buvome pamiršę išsaugoti naują veiksmų kiekį į duomenų failą. Prirašėme šią eilutę.

```
prefeditor.putInt("actions_left", actions_max);
```

12. Grįžus iš tam tikrų veiklų, pagrindinės veiklos kintamieji nėra pakeičiami, todėl pereinant į naują veiklą reikia prirašyti finish() eilutę, kad būtų uždaroma pagrindinė veikla. Kiekvienai veiklai, iš kurios galima sugrįžti į pagr. veiklą reikėjo parašyti onBackPressed funkciją, kuri grįžtų į pagr. veiklą ir užbaigtų (finish) šią.

```
@Override
public void onBackPressed()
{
    startActivity(new Intent(this, MainActivity.class));
    finish();
}
```

```
}Override
ublic void onClick(View v)
{
    switch (v.getId())
    {
        case R.id.buy_sell:
            startActivity(new Intent(MainActivity.this, BuySell.class));
            finish();
            break;
        case R.id.upgrade:
            startActivity(new Intent(MainActivity.this, Upgrade.class));
            finish();
            break;
        case R.id.marketing:
            startActivity(new Intent(MainActivity.this, Marketing.class));
            finish();
            break;
        case R.id.stats:
            startActivity(new Intent(MainActivity.this, Stats.class));
            finish();
            break;
        case R.id.next_week_button:
            AdvanceWeek();
            break;
```

13. Buvome pamiršę įrašyti parduodamo/perkamo produkto naują kiekį į duomenų failą. Tai padarėme.

```
public void UpdateProduct()
{
    prefeditor.putInt(type + "product" + index + "_count", count);
```

Devynioliktas darbas – Statistikos, patobulinimai

1. Toliau dirbėjome prie statistikų sekimo. Produkto pirkimo funkcijoje parašėme kodą, kuris sektų pelno, išlaidų ir nupirktų produktų statistikas. (Perkant produktą, nuo pelno numinusuojami išleisti pinigai)

```
float t_profit = pref.getFloat("t_profit", 0);
t_profit -= t_price;
t_profit = Float.parseFloat(String.format("%.2f", t_profit));
prefeditor.putFloat("t_profit", t_profit);

float t_spendings = pref.getFloat("t_spendings", 0);
t_spendings += t_price;
t_spendings = Float.parseFloat(String.format("%.2f", t_spendings));
prefeditor.putFloat("t_spendings", t_spendings);

int prod_bought = pref.getInt("prod_bought", 0);
prod_bought += count_input;
prefeditor.putInt("prod_bought", prod_bought);
```

2. Pardavimo kode parašėme kodo, kuris sektų pelno ir parduotų produktų statistikas. (Pardavus produktą, prie pelno pridedami gauti pinigai)

```
float t_profit = pref.getFloat("t_profit", 0);
t_profit += t_price;
t_profit = Float.parseFloat(String.format("%.2f", t_profit));
prefeditor.putFloat("t_profit", t_profit);

int prod_sold = pref.getInt("prod_bought", 0);
prod_sold += count_input;
prefeditor.putInt("prod_sold", prod_sold);
```

3. Kad būtų nustatyta mėgstamiausia prekė, mes nusprendėme įrašyti į duomenis kiekvienam produktui atliktą veiksmų (pardavimo ir pirkimo) kiekį. Šis kiekis pliusuojamas vienu tada, kai produktas nupirktas arba parduotas.

```
int prod_actions = pref.getInt(product_names[id] + "a", 0);
prod_actions++;
prefeditor.putInt(product_names[id] + "a", prod_actions);
```

4. Stats veiklos kode prirašėme kodą, kuris duomenų faile randa mėgstamiausią prekę pagal atliktą veiksmų kiekį.

```
f_product = getFProduct();

int getFProduct()
{
    int biggest = 0;
    for (int index = 0; index < 20; ++index)
    {
        if (pref.getInt(product_names[index] + "a", 0) > biggest)
            biggest = index;
    }

    return biggest;
}
```

5. Išbandėme. Viskas veikia.

6. Toliau rašėme kodą savaitinėm staistikom. Savaitinių statistikų duomenys sekami taip pat, kaip ir bendrų statistikų, tik naudojami kiti kintamieji.

```
float w_profit = pref.getFloat("w_profit", 0);
w_profit -= t_price;
w_profit = Float.parseFloat(String.format("%.2f", w_profit));
prefeditor.putFloat("w_profit", w_profit);

float w_spendings = pref.getFloat("w_spendings", 0);
w_spendings += t_price;
w_spendings = Float.parseFloat(String.format("%.2f", w_spendings));
prefeditor.putFloat("w_spendings", w_spendings);

int w_prod_bought = pref.getInt("w_prod_bought", 0);
w_prod_bought += count_input;
prefeditor.putInt("w_prod_bought", w_prod_bought);
```

```
float w_profit = pref.getFloat("w_profit", 0);
w_profit += t_price;
w_profit = Float.parseFloat(String.format("%.2f", w_profit));
prefeditor.putFloat("w_profit", w_profit);

float w_spendings = pref.getFloat("w_spendings", 0);
w_spendings += t_price;
w_spendings = Float.parseFloat(String.format("%.2f", w_spendings));
prefeditor.putFloat("w_spendings", w_spendings);

int w_prod_sold = pref.getInt("w_prod_sold", 0);
w_prod_sold += count_input;
prefeditor.putInt("w_prod_sold", w_prod_sold);
```

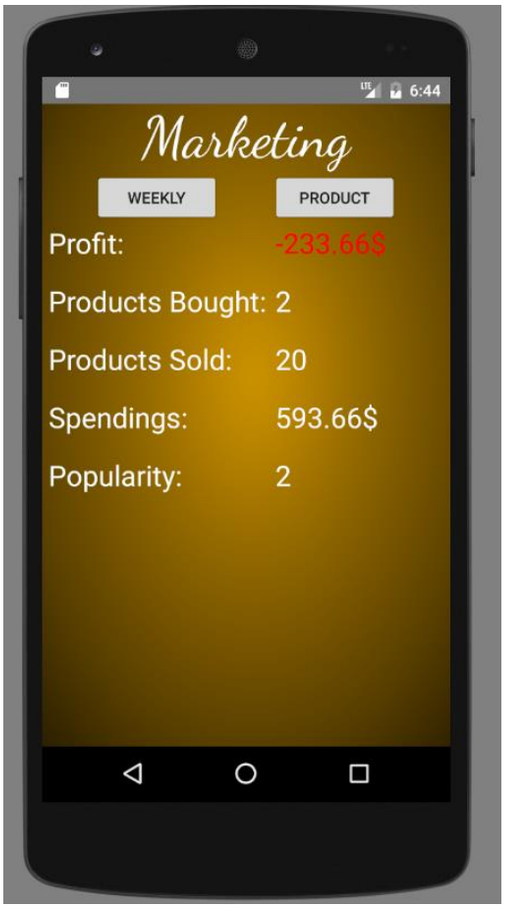
7. Prirašėme kodą, kuris prideda populiarumą po produkto veiksmo. Už kiekvienus 200$ pardavimo/pirkimo yra gaunamas 1 populiarumo taškas.

```
int popularity = pref.getInt("popularity", 0);
int w_popularity = pref.getInt("w_popularity", 0);
w_popularity += t_price / 200;
popularity += w_popularity;
prefeditor.putInt("w_popularity", w_popularity);
prefeditor.putInt("popularity", popularity);
```

8. Kadangi šios statistikos yra tik savaitinės, jos yra išvalomos savaitės pabaigoje.

```
// Wipe weekly stats
prefeditor.putFloat("w_profit", 0);
prefeditor.putInt("w_prod_bought", 0);
prefeditor.putInt("w_prod_sold", 0);
prefeditor.putFloat("w_spendings", 0);
prefeditor.putInt("w_popularity", 0);
```

9. Išbandėme. Veikia.



10. Toliau suvokėme, kad kode liko šiek tiek klaidų. Taigi, jas taisėme. Pirma iš jų – jeigu per didelis populiarumas, tai įmanoma, jog prekės kaina bus mažesnė už jos standartinę kainą. Tai ištaisėme. (Jei skaičius neigiamas, jis tiesiog paverčiamas nuliu)

```
float max_nonprofit = (5f - popularity / 200f) / 100f;

if (max_nonprofit < 0f)
    max_nonprofit = 0f;

price = pref.getFloat(product_names[id] + "p", 0) * 0.8f;
price = Math.round(price * 100.0f) / 100.0f;
```

11. Visame kode pakeitėme apvalinimo kodo eilutes. Ši kodo eilutė yra praktiškesnė ir efektyvesnė.

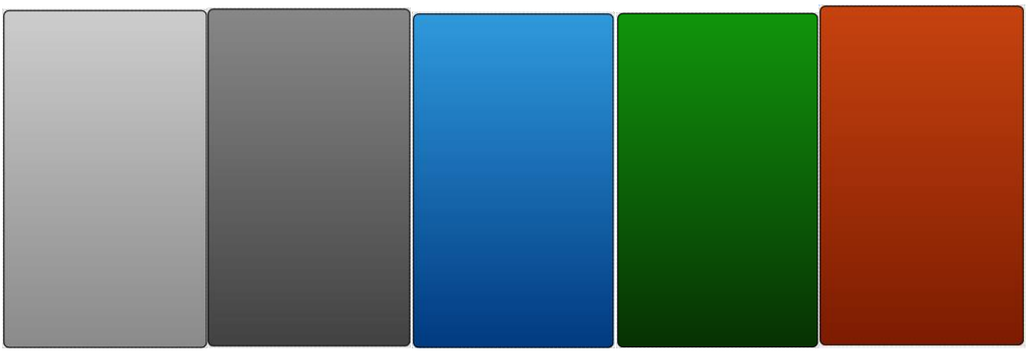
```
p_price = Math.round(p_price * 100.0f) / 100.0f;
```

12. Pardavimo funkcijoje buvo klaida, jog buvo galima parduoti produktą net jei žaidėjas jo neturi. Tai ištaisėme su patikrinimu, ar žaidėjas turi to produkto tiek, kiek reikia.

```
if (pref.getInt(product_names[id], 0) >= count_input)
{
    count -= count_input;
```

Dvidešimtas darbas – Paskutiniai patobulinimai, užbaigimas

1. Nusprendėme, jog reikia pagražinti programoje esančius mygtukus. Sukūrėme penkis gairių failus ir juose parašėme gaires mygtuko stiliaus sukurti. (Šie mygtukai sukurti pagal tą patį formatą)

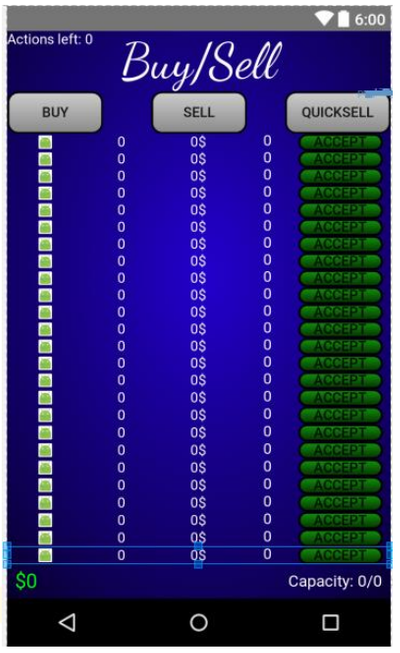


```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <gradient
        android:startColor="#c6420e"
        android:endColor="#7c1b00"
        android:angle="270"/>

    <corners android:bottomRightRadius="10dp"
        android:bottomLeftRadius="10dp"
        android:topRightRadius="10dp"
        android:topLeftRadius="10dp" />

    <stroke android:width="2dp"
        android:color="#000000"/>
</shape>
```

2. Šiuos mygtuku stilius uždėjome atitinkamiems mygtukams.

3. Toliau mum reikėjo pridėti daugiau produktų objektų, kadangi dabar yra tik penki, o pasiūlymų bus galima gauti tikrai daugiau. Taigi, sudeliojome šių objektų 25 ir kiekvienam iš jų prirašėme gairę android:padding kad teksto įvedimo lauko tekstas būtų aiškiai matomas (kitaip matoma tik pusę skaičiaus)

```
<EditText
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:inputType="number"
    android:text="0"
    android:ems="3"
    android:id="@+id/product5b_input"
    android:textColor="#FFFFFF"
    android:layout_weight="0.8"
    android:textSize="15sp"
    android:gravity="center"
    android:padding="1dp"/>
```



4. Beliko padaryti pakeitimus kode. Taigi, visu pirma padidinome perkamų ir parduodamų produktų masyvų dydžius į 25 ir uždėjome limitą, jog jeigu bus siūloma daugiau, negu 25 produktai, tai kiekis vistiek bus pakeistas į 25. Dar prirašėme kodo eilutę, kuri po produkto pirkimo/pardavimo pakeistų teksto laukelio skaičių į 0.

```
bsproduct[] bproducts = new bsproduct[25];
bsproduct[] sproducts = new bsproduct[25];
if (offers_b > 25)
    offers_b = 25;

if (offers_s > 25)
    offers_s = 25;
```

```
input.setText("0");
```

5. Taigi,s beliko parašyti kodo, kuris pakeistų mygtuko spalvą, kai yra įjungta tam tikra skiltis (pvz BuySell veikloje). Kadangi veiklos yra visada įjungiamos pirmoje skiltyje, tai pirmo mygtuko stilių pakeitėme į paspausto mygtuko stilių. Toliau įvedėme naują kintamąjį, kuris rodo, kuri skiltis dabar įjungta ir parašėme kodo, kuris pakeistų praeitos skilties mygtuko stilių į įprastą, o dabartinės – į papspausto mygtuko stilių. Šį kodą pritaikėme visoms veiklos, kuriose yra skiltys (BuySell, Upgrade, Marketing)

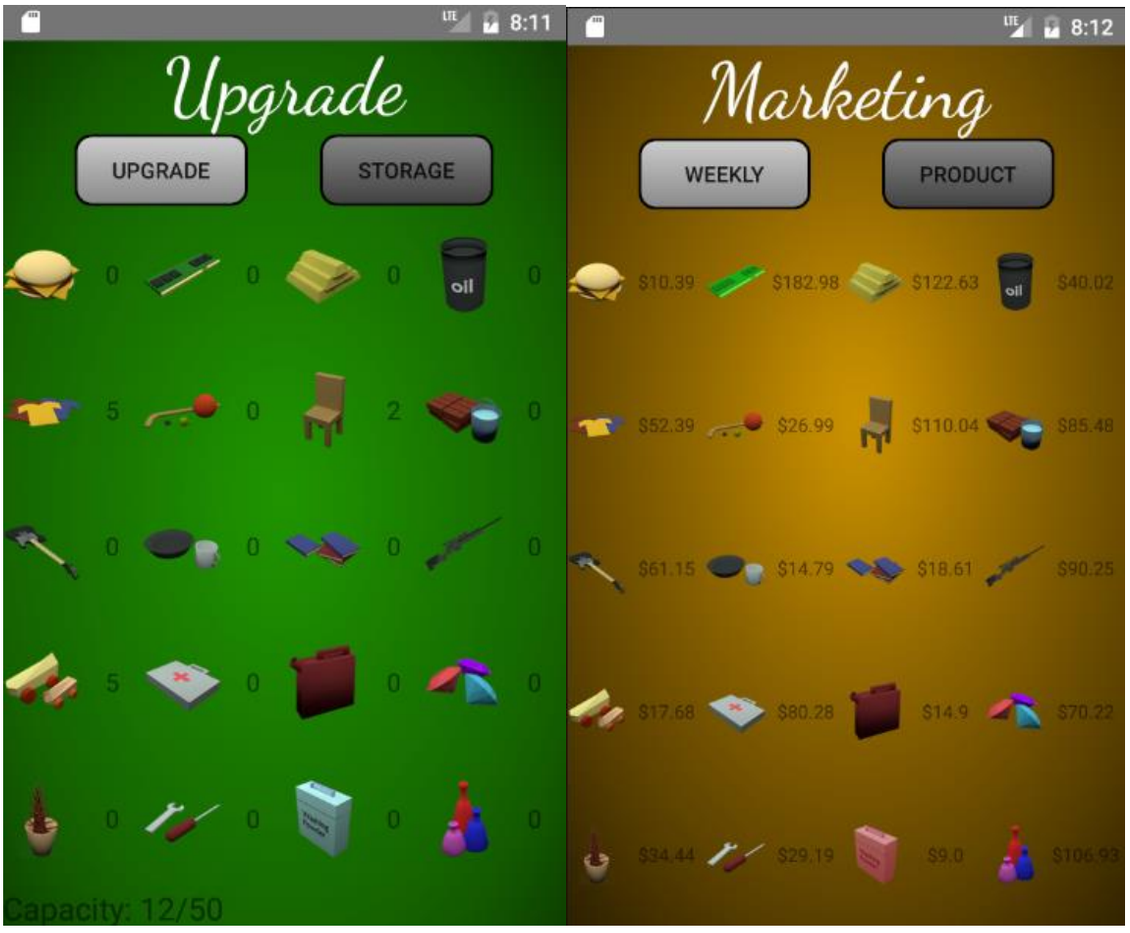
```
<Button
    android:text="Buy"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:id="@+id/buy_button"
    android:layout_weight="1"
    android:background="@drawable/category_pressed_button" />
```


```
int active_layout = 0;
```

```
@Override
public void onClick(View v)
{
    layouts[active_layout].setVisibility(View.GONE);
    layoutbuttons[active_layout].setBackgroundResource(R.drawable.category_button);
    switch (v.getId())
    {
        case R.id.buy_button:
            active_layout = 0;
            break;
        case R.id.sell_button:
            active_layout = 1;
            break;
        case R.id.quicksell_button:
            active_layout = 2;
            break;
    }
    layouts[active_layout].setVisibility(View.VISIBLE);
    layoutbuttons[active_layout].setBackgroundResource(R.drawable.category_pressed_button);
}
```

6. Išbandėme.

7. Sumodeliavome produktų modelius, padarėme nuotraukas, įkėleme į projektą.



8. Parašėme kodo, kuris pasiulymų produktų nuotraukas pakeistų į reikiamas.

```
public void UpdateProduct()
{
    prefeditor.putInt(type + "product" + index + "_count", count);
    if (count == 0)
        view.setVisibility(View.GONE);
    else
    {
        name_view.setText("" + count);
        price_view.setText(price + "$");
        String image_name = "product" + (id+1);
        int prodID = getResources().getIdentifier(image_name , "drawable", getPackageName());
        image.setImageResource(prodID);
    }
}
}
```

9. Padarėme paskutinius taisymus – parašėme koo, kad produkto negalima būtų pirkti/parduoti, jei įvestas skaičius 0. Tada pakeitėme atimama ir predama populiarumą dėl veiksmų į 5 vietoj 10, kad šiek tiek palengvinti žaidimą, kai nedaromi veiksmai. Sumažinome mokesčių dydžius (pradinis $50 vietoj $100).

```
count_input != 0)
```

```
    popularity -= 5;

    if (popularity < 0)
        popularity = 0;
}
else if (actions_left == 0)
{
    popularity += 5;
}
```

```
int taxes = (int) (50 + pow((pref.getInt("trucks_lvl", 0) + pref.getInt("capacity_lvl", 0) + pref.getInt("marketing_lvl", 0)), 2) * 50);
```

Išvados

- Mums pasisekė sukurti gerai veikiantį žaidimą.
- Išmokome naudotis Android Studio programa, kurti programos išdėstymą bei įdiegti funkcionalumą.
- Internete radome pakankamai informacijos, kadangi Android Studio yra oficiali Android kūrimo programa ir ji yra gan populiari.
- Išmokome pasiskirstyti darbus kuriant programėlę, gautus rezultatus sujungti.
- Be praeitos patirties šį darbą padaryti būtų buvę labai sunku, kadangi prireikė plačių programavimo žinių. Taigi, žmogui, kuriančiam programą galime parekomenduoti prieš tai mokytis reikiamos programavimo kalbos.

Šaltiniai

- https://en.wikipedia.org/wiki/Android_Studio
- https://lt.wikipedia.org/wiki/Integruota_k%C5%ABrimo_aplinka
- https://developer.android.com/training/index.html
- https://www.tutorialspoint.com/android/android_studio.htm
- http://www.javahelps.com/2015/03/android-simple-calculator.html
- https://lt.wikipedia.org/wiki/Objektinis_programavimas
- https://lt.wikipedia.org/wiki/XML
- http://stackoverflow.com/questions/11688689/save-variables-after-quitting-application
- http://stackoverflow.com/questions/5821051/how-to-display-the-value-of-a-variable-on-the-screen
- http://stackoverflow.com/questions/23024831/android-shared-preferences-example
- http://stackoverflow.com/questions/4531396/get-value-of-a-edit-text-field
- https://developer.android.com/training/basics/firstapp/starting-activity.html
- http://stackoverflow.com/questions/20156733/how-to-add-button-click-event-in-android-studio
- http://stackoverflow.com/questions/3307090/how-to-add-background-image-to-activity
- http://stackoverflow.com/questions/29047902/how-to-add-an-image-to-the-drawable-folder-in-android-studio
- http://stackoverflow.com/questions/4905370/what-are-the-differences-between-linearlayout-relativelayout-and-absolutelayou
- http://stackoverflow.com/questions/26492522/how-do-i-remove-the-title-bar-in-android-studio
- http://stackoverflow.com/questions/17054000/cannot-resolve-symbol-r-in-android-studio
- http://stackoverflow.com/questions/3687315/deleting-shared-preferences
- http://stackoverflow.com/questions/7690416/android-border-for-button
- http://stackoverflow.com/questions/10266595/how-to-make-a-round-button
- http://stackoverflow.com/questions/12166559/how-to-add-a-gradient-to-buttons-in-android-through-xml
- http://stackoverflow.com/questions/25905086/multiple-buttons-onclicklistener-android
- http://stackoverflow.com/questions/15642104/array-of-buttons-in-android
- http://stackoverflow.com/questions/20000560/android-layout-width-half-of-parent
- http://stackoverflow.com/questions/18051472/how-to-center-the-content-inside-a-linear-layout
- https://developer.android.com/guide/topics/ui/layout/gridview.html

- http://stackoverflow.com/questions/432037/how-do-i-center-text-horizontally-and-vertically-in-a-textview-on-android
- http://stackoverflow.com/questions/6173400/how-to-hide-a-button-programmatically
- http://stackoverflow.com/questions/7053738/what-is-meant-by-ems-android-textview
- http://stackoverflow.com/questions/1957831/center-a-button-in-a-linear-layout
- http://stackoverflow.com/questions/6679434/android-findviewbyid-with-a-variant-string
- http://stackoverflow.com/questions/20121938/how-to-set-tint-for-an-image-view-programmatically-in-android/27929217
- http://stackoverflow.com/questions/10614696/how-to-pass-parameters-to-onclicklistener
- http://stackoverflow.com/questions/4903515/how-do-i-return-an-int-from-edittext-android
- http://stackoverflow.com/questions/5089300/how-can-i-change-the-image-of-an-imageview
- http://stackoverflow.com/questions/2784081/android-create-spinner-programmatically-from-array
- http://stackoverflow.com/questions/6078157/random-nextfloat-is-not-applicable-for-floats
- http://stackoverflow.com/questions/5312334/how-to-handle-back-button-in-activity
- https://developer.android.com/studio/run/device.html
- http://stackoverflow.com/questions/22833515/rounding-to-6-decimal-places-using-math-round-method-in-java-android

Turinys