Crearea unei aplicații mobile
Folosind Java

Nume: Elecfi Sergiu-Andrei

Grupa: 30233

Cuprins

1.Introducere

În cadrul acestei lucrări, ne propunem să explorăm fundamentele dezvoltării de programe destinate telefoaelor mobile cu sistem de operare Android, cu accent pe facilitățiile de programare oferite de acest sistem de operare. Limbajul de programare folosit va fi Java, unul dintre cele mai utilizate limbaje de dezvoltare al aplicațiilor pe Android.

Scopul principal al acestui ghid este de a oferi o înțelegere detaliată asupra procesului de creare a unei aplicații pentru platforma Android, inclusiv familiarizarea cu mediul de dezvoltare. Pentru a evidenția aceste concepte, vom implementa 3 programe care variază din punctul de vedere al dificultății lor la creare.Astfel, pornind de la afișarea unui simplu "Hellow World!", până la aplicații mai complexe, cum ar fi capturarea și postarea unei fotografii pe platforma Facebook sau o aplicație care furnizează procentul bateriei telefonului.

Prin intermediul acestui ghid, veți dobândi cunoștințele necesare și abilitățile necesare pentru a dezvolta aplicații mobile eficiente și funcționale pe platforma Android, folosind Java ca limbaj de programare principal. Vom explora metodele și tehnici de lucru, astfel încât la finalul acestei lucrări să aveți o înțelegere solidă a procesului de dezvoltare de aplicații Android și să fiți capabili să creați programe personalizate pentru telefoanele mobile cu sistem de operare Android.

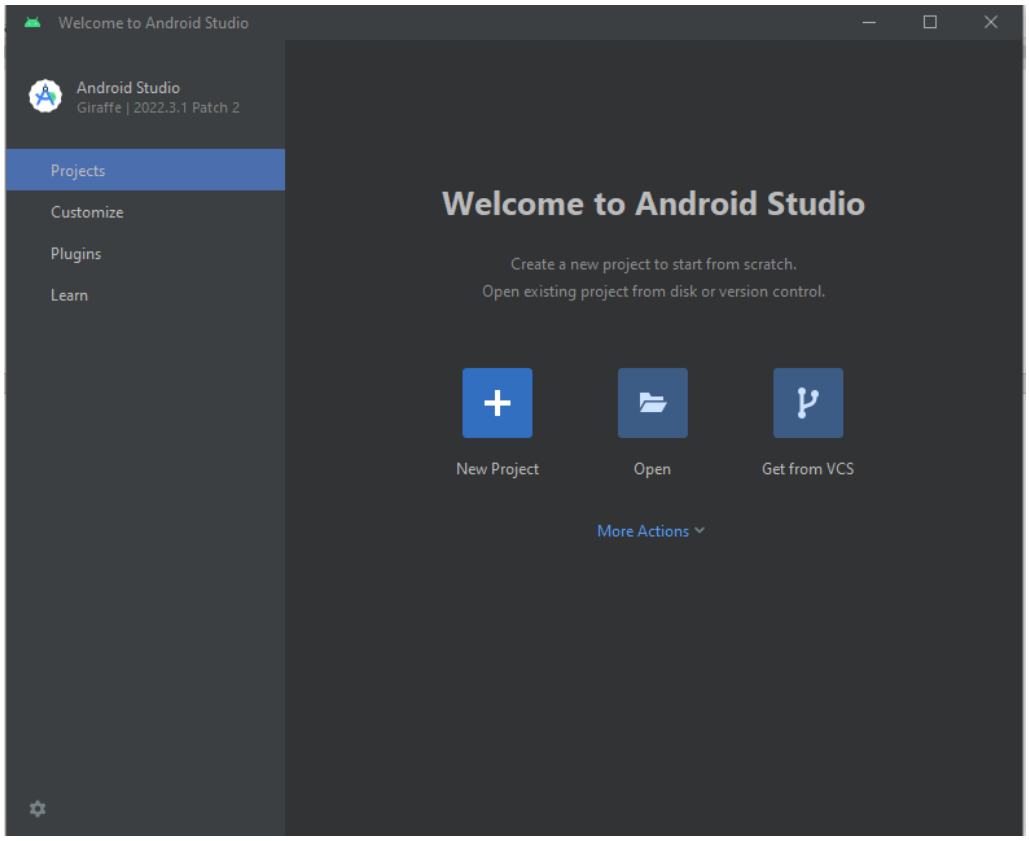
2. Instalare Android Studio

Programul pe care îl recomand pentru crearea aplicațiilor de tip Android este ”Android Studio”. Se pot folosi și alte programe în care să dezvoltăm viitoare aplicații pentru Android în limbajul de programare Java: Eclipse și IntelliJ cu plugin-urile necesare dezolvtării aplicațiilor pentru telefon.

Descăcați Android Studio de pe acest link: https://developer.android.com/studio

Instalarea ar trebuii să decurgă fără probleme, deși unele procese din timpul descărcării ar putea dura mai mult, dar nu este nicio problemă.

În cazul în care instalarea Android Studio s-a terminat, ar trebui să se deschidă o fereastră asemănătoare cu aceasta :

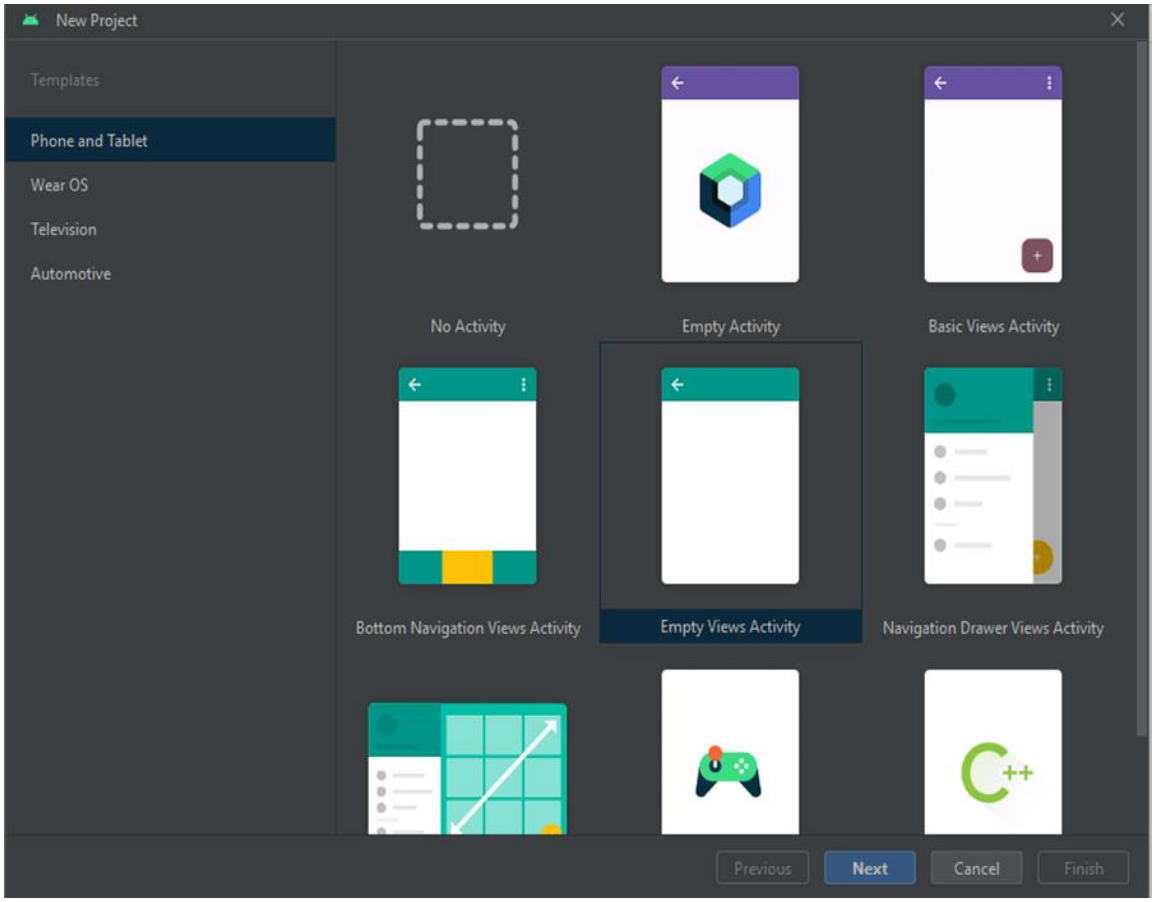


În cazul în care folosiți Eclipse aveți nevoie să instalațiun plugin ADT. Locul de unde îl instalați și pașii instalării le găsiți pe acest link: https://developers.sap.com/tutorials/abap-install-adt.html

În cazul folosirii IntelliJ IDEA, aveți de asemenea nevoie de un plugin în plus. Aici aveți link-ul de descărcare pentru Android Support: https://plugins.jetbrains.com/plugin/1792-android-support

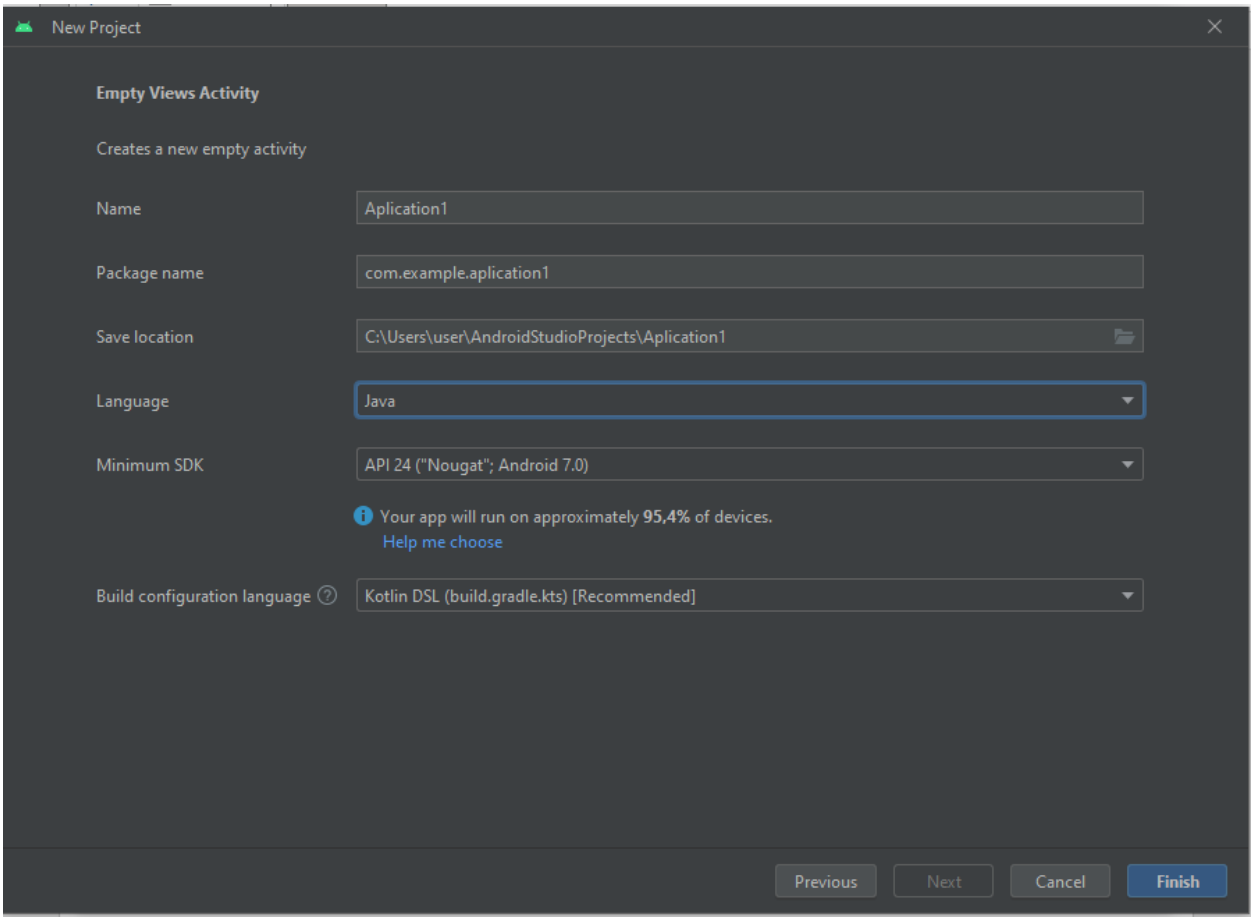
3. Crearea Primei Aplicații

Prima cerință pe care dorim să o realizăm este afișarea unui mesaj de tipul ”Hello World”. Pentru început vom crea noul nostru proiect apăsând pe butonul **New Project** din fereastra de start (în cele mai multe cazuri acest lucru este selectat default, dar cel mai bine este să ne asigurăm). Apoi vom alege ca Template **Phone and Tablet** din partea stângă al meniului ca în imaginea de mai jos. De asemenea, vom folosi ca prim șablon **Empty Views Activity**.

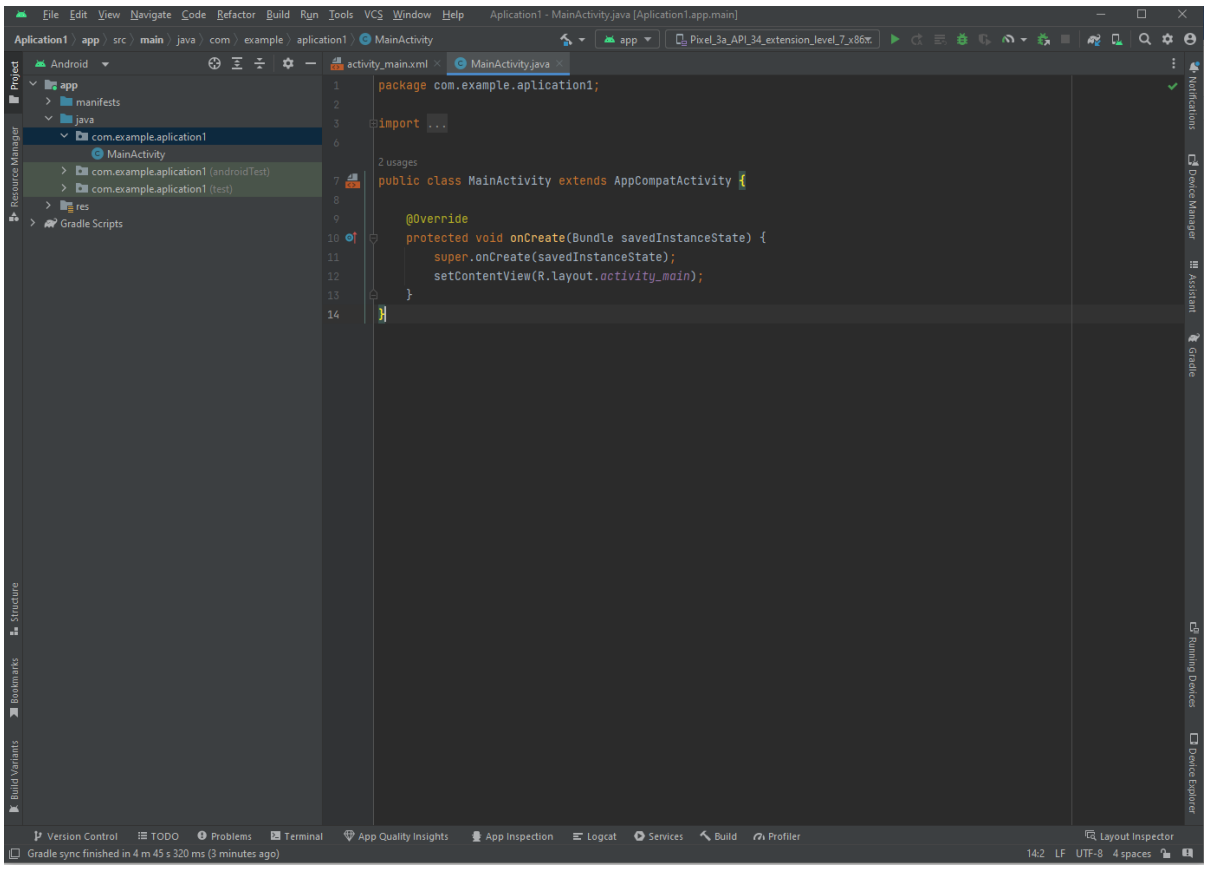


După ce ne-am asigurat că totul este selectat ca în imaginea de mai sus apăsăm butonul **Next** din colțul dreapta din josul ferestrei curente.

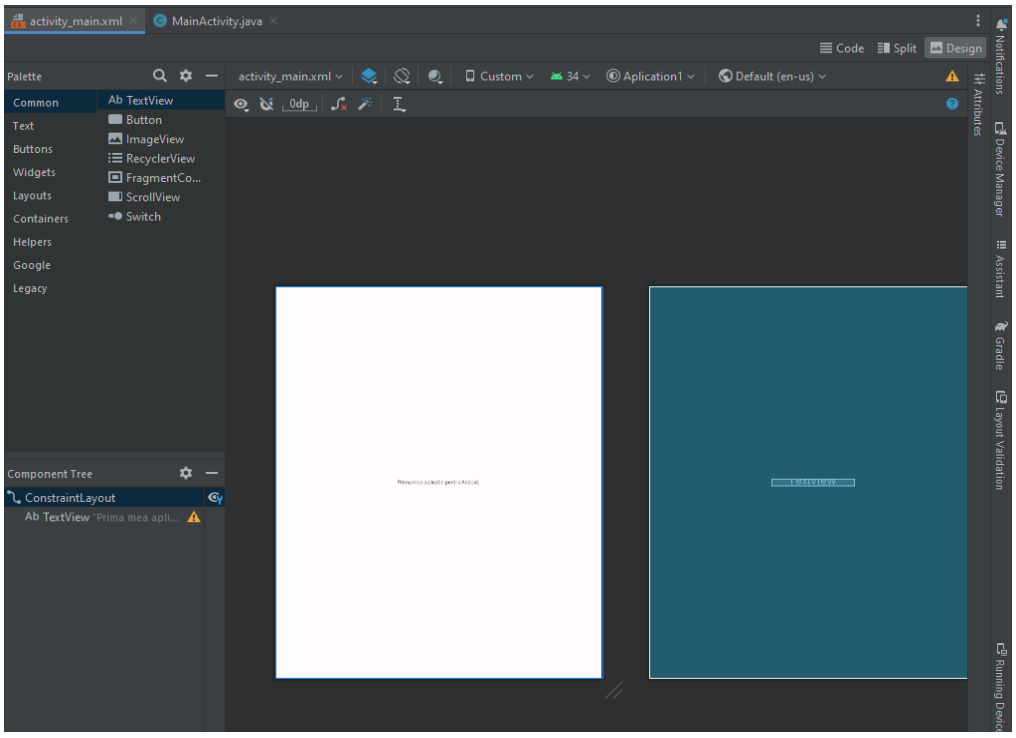
În fereastra următoare redenumim aplicația noastră cu ce nume dorim noi, iar în cazul în care dorim să modificăm locația salvării fișierului putem să o facem, fără a fi vreo problemă. Lucrul la care trebuie să fim atenți este să modificăm limbajul de programare dorit din Kotlin în **Java**, ca în imaginea de mai jos.



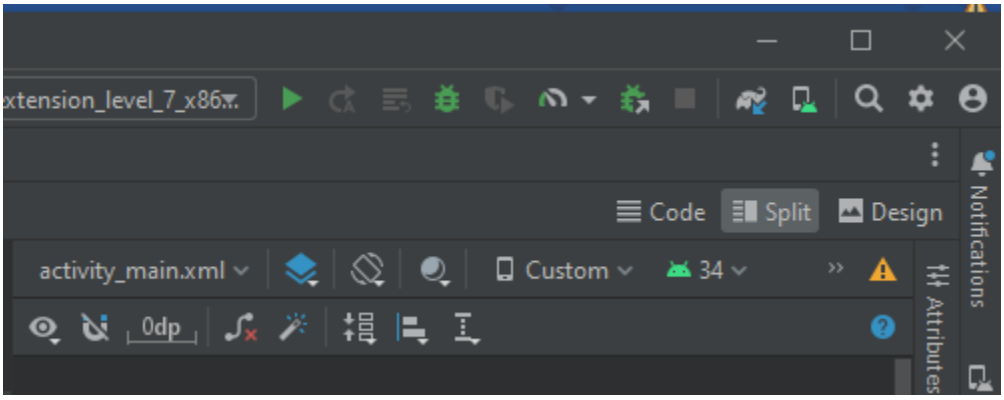
După parcurgerea acestor pași puteți apăsa pe butonul **Finish** in josul paginii. Apoi din nou pe **Finish**, iar proiectul vostru ar trebui să fie creat. Crearea tuturor filelelor din fișier ar putea să dureze ceva, însă acest fapt nu este un lucru care să vă îngrijoreze, timpul de creare și deschidere a fișierelor pe Android Studio diferă de la un calculator la altul. În bara de jos apar fișierele care se descarcă și sunt verificate. La final, după ce întreg fișierul este creat aplicația creată ar trebui să arate așa:



La pasul următor apăsăm pe **activity_mail.xml** care este un fișier deja deschis în Android Studio. Ar trebui să aveți o imagine la fel ca aceasta.

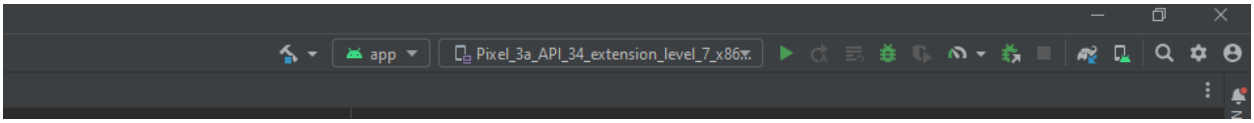


Ca și în orice IDE utilizat de până acum afișarea mesajului ”Hello World!” este dat de aplicație automat, deci noi nu vom avea nevoie să scriem noi cod pentru această cerință. Totuși ca prim exercițiu încercați să schimbați mesajul ”Hellow World!” cu textul ”Prima mea aplicatie Android”. Acest lucru se realizează prin modificarea simplă în cod a mesajului. Codul poate fi vizualizat cu ajutorul apăsării butonului **Code** din colțul dreapta sus al Android Studio.

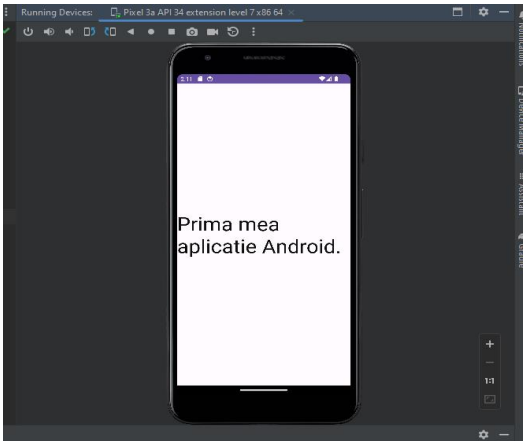


În cazul în care vrem să vedem schimbările în timp real, ne folosim de opțiunea de **Split**, în care putem modifica codul și vedea schimbările aduse în timp real.

După ce ați modificat mesajul afișat, alegeți emulatorul dat de Android Studio, în cazul în care nu este selectat by default, iar apoi apăsăm pe butonul verde de lângă, adică **Run.**



Începerea programului ar putea dura ceva, dar când aplicația este pornită ar trebui să aveți un telefon pe ecran care arată așa:



Pentru a mai testa puțin ce are de oferit Android Studio, încercați rezolvarea următoarelor cerințe:

a) Modificați mărimea textului
b) Modificați culoarea textului
c) Modificați poziția textului
d) Modificați fundalul telefonului când se afișează mesajul

Hint: Căutați prin setări când aceesați pagina de Design a aplicației.

4.Crearea unei aplicații pentru calcularea bacșișului

A doua aplicație pe care dorim să o creăm este cea pentru calcularea bacșișului la restaurant. Dorim ca în aplicație să introducem totalul notei de plată, procentul din nota totală pe care dorim să îl lăsăm ca bacșiș și , de asemenea, numărul de persoane de la masă. După apăsarea unui buton dorim ca bacșișul pe care fiecare persoană trebuie să îl lase să apară pe ecranul telefonului, în aplicația curentă.

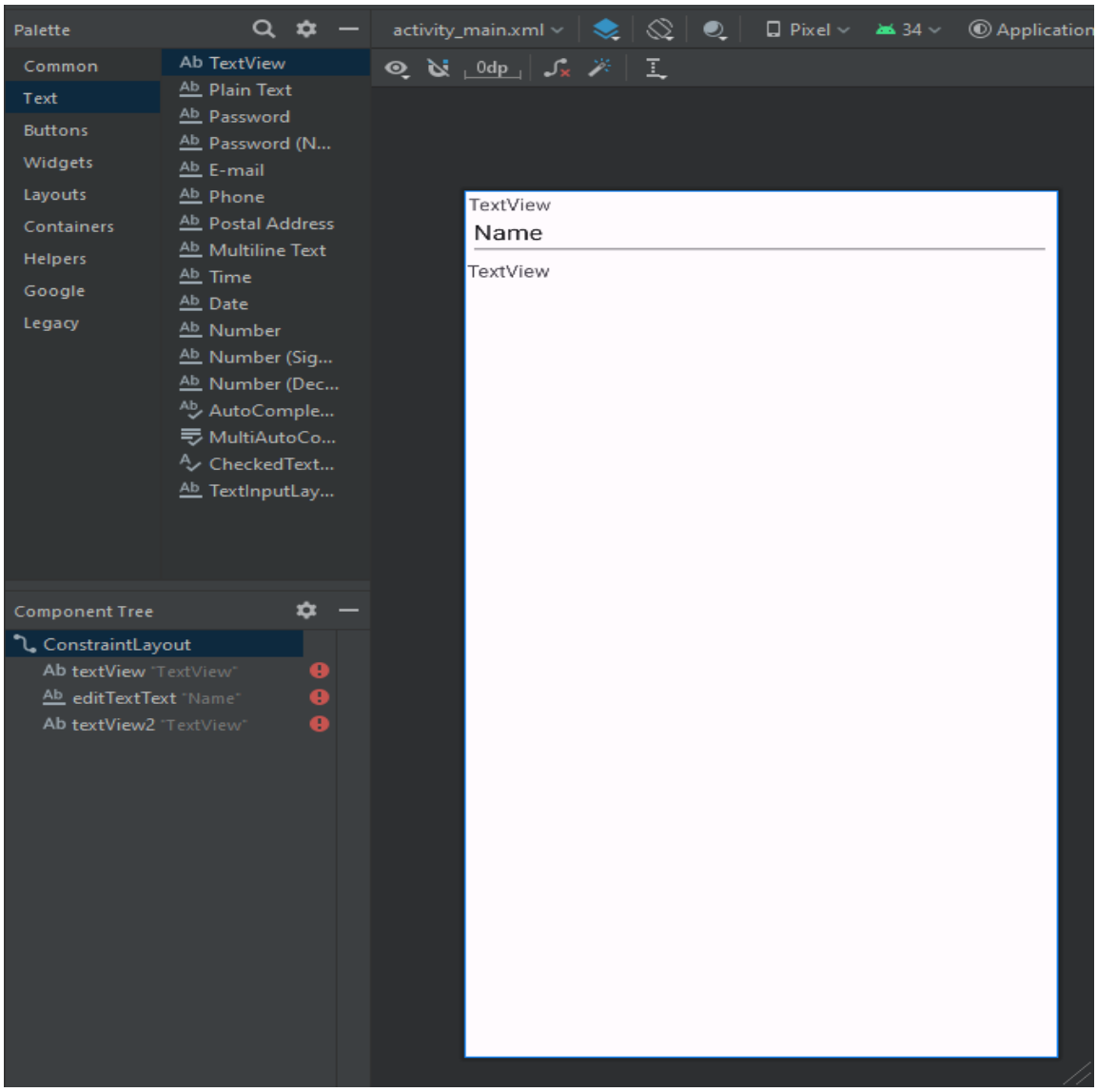
Pentru început, vom creea același template ca la primul nostru proiect în Android Studio. După crearea proiectului, vom începe munca prin a șterge codul din **activity_main.xml** care este generat automat de Android Studio. Acest cod este util doar pentru afișarea mesajului ”Hello World!”.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation = "vertical"
    android:padding = "20dp"
    tools:context=".MainActivity">
```

Tot aici, vom adăuga orientarea telefonului pe verticală, scriind acest cod, în cadrul primului block de linii de cod.

```
android:orientation = "vertical"
```

Următorul pas este crearea vizuală a aplicației. Vom începe prin adăugarea difertitor TextBox-uri în aplicație. Mai întâi ne ducem la design-ul aplicației. Aici avem în partea stângă diferite opțiuni. Cu drag and drop putem trage orice căsuță de text. După ce ați tras câte căsuțe credeți că veți avea nevoie, design-ul vostru ar trebui să arate ceva de genul.



Dacă mergeți înapoi în **activity_main.xml**, puteți observa că a fost adăugat un cod pentru aceste noi lucruri adăugate.

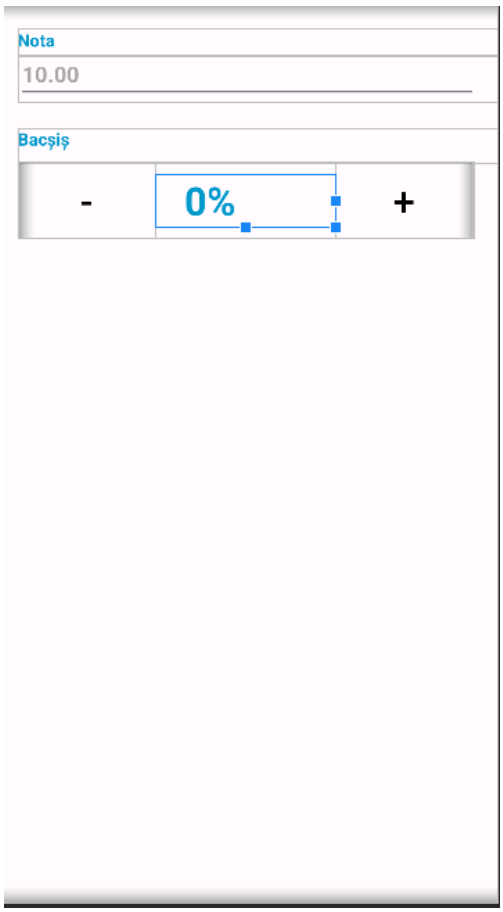
```
<TextView
    android:id="@+id/textView"
    android:layout_width="405dp"
    android:layout_height="18dp"
    android:text="TextView"
    tools:layout_editor_absoluteX="3dp"
    tools:layout_editor_absoluteY="2dp" />

<EditText
    android:id="@+id/editTextText"
    android:layout_width="405dp"
    android:layout_height="38dp"
    android:ems="10"
    android:inputType="text"
    android:text="Name"
    tools:layout_editor_absoluteX="3dp"
    tools:layout_editor_absoluteY="20dp" />

<TextView
    android:id="@+id/textView2"
    android:layout_width="406dp"
    android:layout_height="28dp"
    android:text="TextView"
    tools:layout_editor_absoluteX="2dp"
    tools:layout_editor_absoluteY="58dp" />
```

Puteți adăuga diferite schimbări la nivelul textului, pentru mai multe informații accesați acest link: https://abhiandroid.com/ui/textview#gsc.tab=0. La pasul următor vom crea Layout-ul nostru. În cadrul acestuia vom adăuga fiecare Text de mai sus, însă vom adăuga câteva schimbări, și de asemenea, câte un buton pentru fiecare TextView. Vom adăuga și un now Layout pentru a așeza butoanele și casuțele de text nou adăugate mai ușor. Vom adăuga în noul Layout din proiect un TextArea și două butoane. Acestea vor fi folosite ulterior pentru confirmarea procentului pentru bacșiș.

Pentru a citi mai multe și a înțelege codul puteți citi din aceste resurse:

1. Butoane : https://developer.android.com/develop/ui/views/components/button
2. Layouts https://developer.android.com/develop/ui/views/layout/declaring-layout

Puteți adăuga aceste noi componente cum doriți, la final schema dumneavoastra ar trebui să arate ceva de genul.



Dacă nu reușiți să adăugați butoane și textul astefel încât să ajungeți la aceeași imagine, sau cel puțin una asemănătoare, puteți copia codul de mai jos:

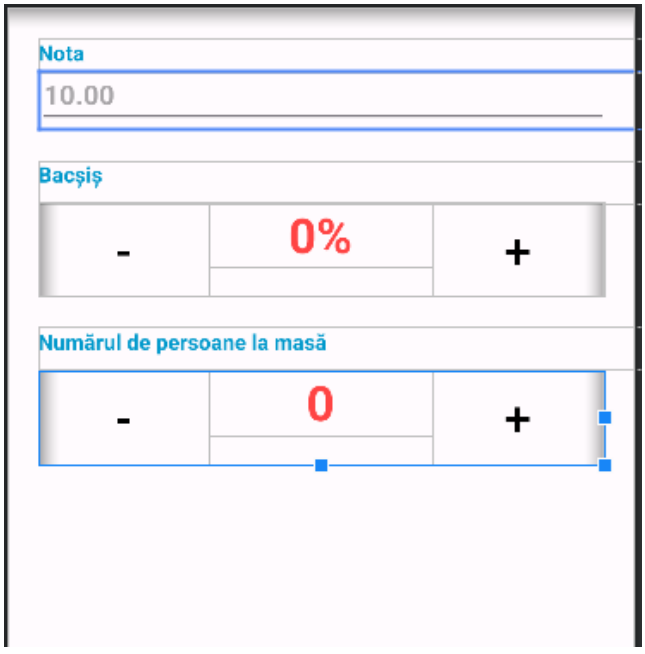
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <Button
        android:id="@+id/b_minus"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:textColor = "@android:color/black"
        android:text="-"
        android:textSize="32sp" />

    <TextView
        android:id="@+id/tv_bacsis"
        android:layout_width="406dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="   0%"
        android:textColor="@android:color/holo_blue_dark"
```

```
        android:textSize="32sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/b_plus"
        style="@style/Widget.AppCompat.Button.Borderless"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor = "@android:color/black"
        android:layout_weight="1"
        android:text="+"
        android:textSize="32sp" />
</LinearLayout>
```

După cum se poate vedea, precum în Java, la fiecare TextView sau fiecare Buton putem adăuga diferite tipuri de scris, culori, poziționare și mărime. La pasul următor vom copia acest cod plus ultimul TextView și îl vom adăuga sub codul acesta, pentru a creea aceleași lucrui pentru numărul de persoane de la masă. În cazul în care așezarea nu se face chiar sub ce aveam deja, încercați să trageți cu ”mâna” de acestea și să le așezati cum doriți.

Atenție mare, toate butoanele și TextView-urile trebuie redenumite, deoarece vor apărea erori iar codul nu va rula. Vă recomand să numiți butoanele și căsuțele de text sugestiv, pentru a fi mai ușor apoi de modificat în cazul în care vor apărea probleme sau se doresc unele modificări.

După ce ați făcut modificăriile enumerate mai sus, proiectul vostru ar trebui să arate așa:

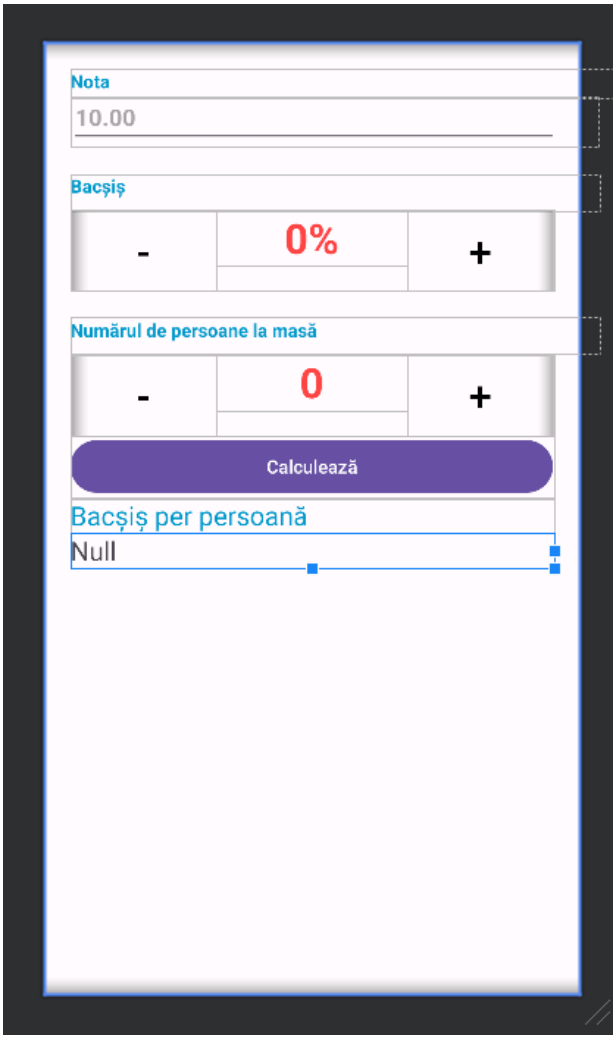


Mai apoi, vom adăuga un nou buton, pe care îl vom folosi pentru a calcula cât bacșiș fiecare persoană va fi nevoită să lase, bazându-ne pe informațiile de mai sus. La codul butonului nu se vor adăuga modificări, doar dacă se doresc modificări la nivel de de design. Codul pentru buton este:

```
<Button
    android:id="@+id/b_calculate"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textColor = "@android:color/white"
    android:text="Calculează" />
```

În final, vom adăuga un nou TextView în care vă apărea bacșișul pe care fiecare persoană trebuie să îl lase, în funcție de datele noastre adăugate mai sus. Vom adăuga încă un TextView pentru a estetică, adică în acest TextView va scrie ”Bacșiș” pentru a ști ce valoarea va apărea.

La final, design-ul vostru ar trebui să arate așa sau apropiat:



Acum, vom trece la partea de scriere în codul Java. Pentru început, vom deschie fișierul **MainActivity.java** din cadrul proiectului**.** Așa ar trebui să arate la început:

```
package com.example.application2;

import ...

2 usages
public class MainActivity extends AppCompatActivity {

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

În cazul în care nu sunteți familiarizați cu limbajul de programare Java sau doriți să mai aprofundați, puteți citi informații și aprofunda tot ce este legat de Swing pe acest link: https://www.javatpoint.com/java-swing

Vom începe prin declararea tuturor elementelor adăugate în cadrul design-ului.

```
EditText et_nota;
TextView textView1, textView2, textView3, textView4, textView5;
Button b_plus,b_minus,b_plus1,b_minus1, b_calculate;
```

În cadrul metodei **onCreate** vom atribui fiecărei variabile create anterior, id-ul butonului/textView-ului din design. Atribuirea se face astfel:

```
name_var = findViewById(R.id.name_in_design);
```

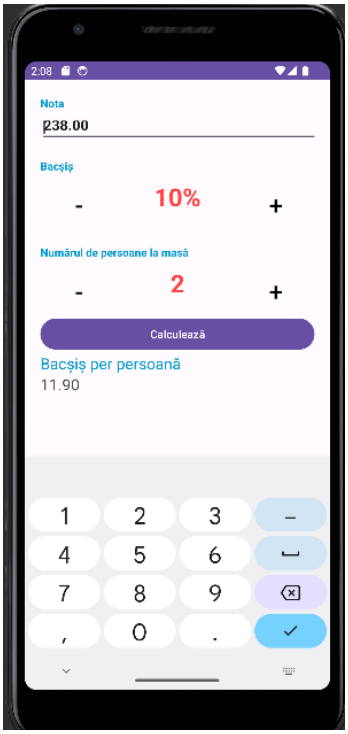
Acum, vom începe să scriem codul pentru fiecare buton. După cum știm din java acest lucru se face cu ActionListener, și facerea de @Override a funcției onClick. De exemplu, pentru scăderea numărului de persoane, codul este:

```
b_minus1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View biew){
        if (persoane > 1){
            persoane --;
            tv_persoane.setText(persoane);
        }
```

În cadrul metodei **b_calculate.setOnClickListener** vom calcula bacșișul final. Aici ne fom folosi de funcțiile deja implementate de Android Studio: toString, valueOf, equals, getText. Vom folosi o funcție din librăria Math, care rotunjește numărul la cel mai apropiat întreg, pentru a nu lucra cu virgulă flotantă. Codul final ar trebuie să arate așa. Să nu uitați și să afișați datele necesare.

```
b_calculate.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View biew){
        String x = et_nota.getText().toString();
        if (!x.equals("")){
            nota = Double.valueOf(x);
            nota *= 100;
            nota = Math.round(nota);
            nota /= 100;

            et_nota.setText(String.format(Locale.getDefault(), "0.2f",
nota));

            bacsis = (nota * procentaj) / 100;
            bacsis *= 100;
            bacsis = Math.round(bacsis);
            bacsis /= 100;

            tv_final.setText(String.format(Locale.US, "0.2f", bacsis));
        }
    }
});
```

Acum putem rula aplicația de acum ca și cea anterioară.



Pentru a exersa ceea ce s-a învățat în crearea acestui proiect, puteți încerca:

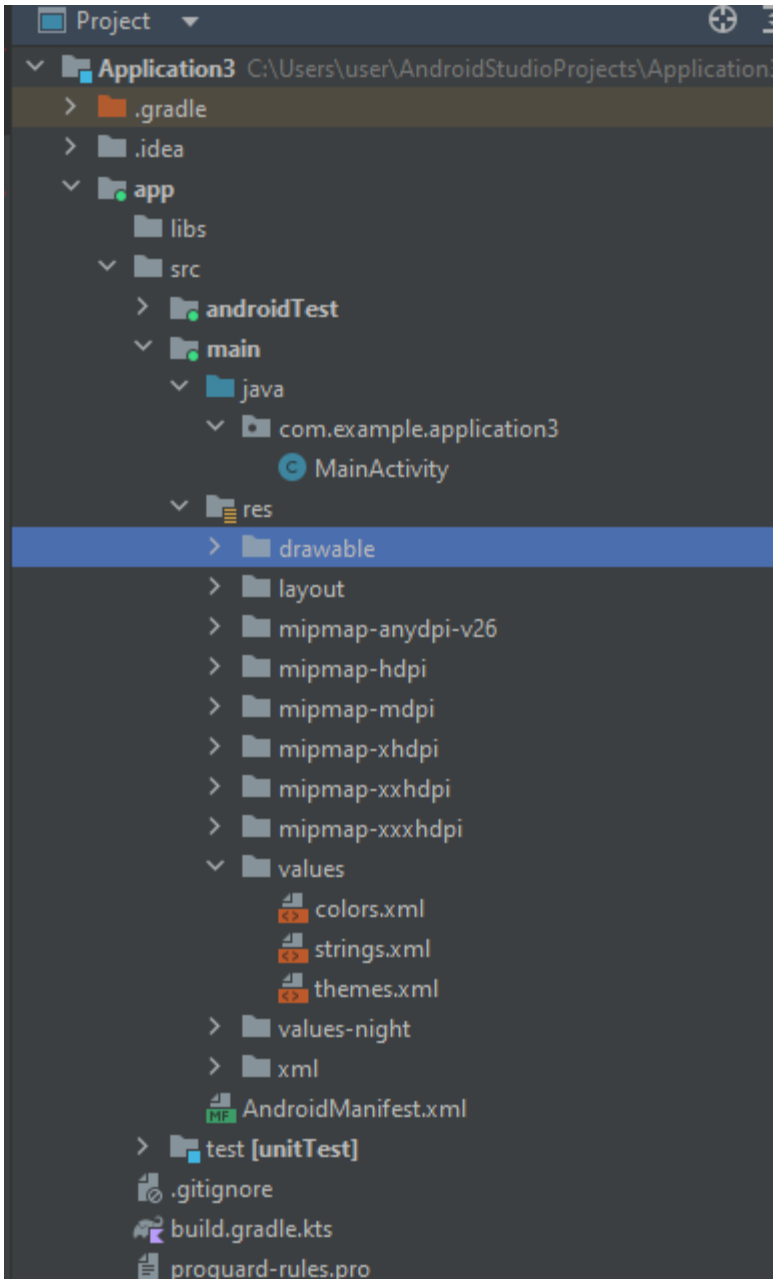
1. Calcularea și afișarea notei de plată totale după adăugarea bacșișului
2. Posibilitatea fiecărei persoane de a alege bacșișul dorit
3. Posibilitatea de a cere rest la nota de plată

5.Crearea ultimei aplicații Android

Pentru cea de a treia aplicație, am dori să adăugam niște poze și să alegem random acea poză. De exemplu putem face un zar și să alegem fața pe care acesta va ateriza sau să ghicim numărul care urmează să fie generat.

În cadrul ultimei noastre aplicații ne vom utiliza de poze cu cărți de joc, și alegerea a două cărți random. Vom lua în evidență și scorul pe care îl avem (vom juca contra computerului și cine are cartea mai mare câștigă runda), și vom adăuga diferite funcționalități precum ”duelul” în cazul egalității între noi și computer sau butonul de RESET în cazul în care dorim să reluăm de la început jocul.

Vom începe prin a crea un nou proiect, îl puteți denumi cum doriți. În cazul în care ați uitat cum să creați un proiect nou, citiți secțiunea 3 din cadrul lucrării. Trebuie să descărcăm o poză cu fiecare carte de joc și cu spatele unei cărți. Dacă doriți puteți să descăcarți și un background pentru aplicație. Link-ul de unde puteți descărca pozele cu cărți este: https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/vector-playing-cards/PNG-cards-1.3.zip



Pentru a descărca spatele unei cărți căutați pur și simplu pe internet și alegeți variante dorită. Atenție, pozele să fie sub format PNG.

După ce ați descărcat tot ce aveți nevoie, trebuie să copiem pozele în interiorul proiectului. Acest lucru îl putem face direct în Android Studio, în folder-ul "drawable". Dacă nu găsiți folderul singuri, puteți să vă orientați după imaginea din stânga.

La pasul următor, pentru cei ce vor să adauge background aplicației lor, acest lucru se rezolvă cu o singură linie de cod în cadrul clasei "activity_main.xml".

```
android:background="@drawable/background"
```

Urmează să creăm un layout de tip linear (LinearLayout) în cadrul acestei clase (dacă se dorește altă așezare a butoanelor și cărțiilor se poate folosi alt tip de layout). În cadrul layout-ului creat vom seta cum vor apărea cărțiile noastre pe

ecranul telefonului, adică formatul în care dorim să le vedem pe ecran. Vom pune la început să apară spatele cărțiilor, deoarece nu dorim să apară cărțiile înainte ca noi să apăsăm START sau DRAW.

```
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/background"
    tools:context=".MainActivity">

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

        <ImageView
            android:id="@+id/iv_card_left"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:scaleType="centerInside"
            android:src="@drawable/back" />
    </LinearLayout>

</RelativeLayout>
```

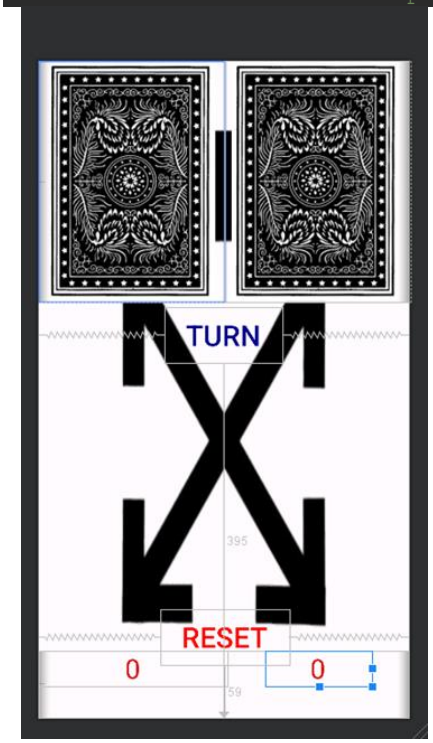
Pe urmă, copiem codul din cadrul cărții din partea stângă a ecranului și pentru cartea din dreapta. După cum puteți observa, în design-ul din partea dreaptă, cărțiile se așează singure pe ecran, fără a fi nevoie de a adăuga instrucțiuni suplimentare. Să nu uitați să schimbați denumirea celei de a doua cărți de joc în cazul în care ați dat copy paste de la celălalt ImageView.

La pasul următor vom adăuga butonul de întoarcere al cărțiilor pe care îl vom numi ”TURN” și acesta se va aflat înafara LinearLayout-ului. Am adăugat butoane în cadrul proiectului trecut, așa că acest lucru nu ar trebui să fie o problemă. În cazul în care nu vă descurcați, codul pentru buton este:

```
<Button
    android:id="@+id/turn"
    style="@style/Widget.AppCompat.Button.Borderless"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="395dp"
    android:text="TURN"
    android:textColor="#00008B"
    android:textSize="32sp" />
```

De reținut să adăugați mai întâi butonul în cadrul design-ului și așezați-l unde doriți. La pasul următor vom crea alt LinearLayout pentru tabela de scor. Vom adăuga două TextView-uri ca și în proiectul precedent, adică nimic nou. Procedeul de adăugare este ca și cel de la ImageView de mai sus și de așezare al acestora în cadrul ecranului este la fel. Codul pentru un TextView este:

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:orientation="horizontal"
    android:layout_height="wrap_content">

<TextView
    android:id="@+id/score_left"
    android:layout_width="328dp"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginEnd="42dp"
    android:layout_marginBottom="35dp"
    android:layout_weight="1"
    android:gravity="center"
    android:text="0"
    android:textColor="#d50c0c"
    android:textSize="30sp" />
```



Design-ul vostru ar trebui să arate ceva de genul, desigur diferit având în vedere dacă ați ales alte imagini și diferite culori la butoane și TextViews.
La partea de design al aplicației nu mai este nimic de făcut, doar dacă se doresc unele înfrumusețări la nivel de cromatică sau imagini. Acum urmează să trecem la partea Java a proiectului.

Ca și în proiectul anterior vom lucra doar în “MainActivity.java”. Pentru început vom crea variabile pentru fiecare ImageView,TextView și Button adăugat în design. Apoi, urmează să preluăm id-ul pentru fiecare dintre aceste variabile, ca și în proiectul precendent folosind comanda:

```
card_right = (ImageView) findViewById(R.id.iv_card_right);
```

În cadrul acestui proiect, nu vom avea atât de scris la setOnClickListener, deoarece avem doar două butoane, cel de reset și cel de ”start”. Vom avea nevoie și de o nouă variabilă de tip ”Random” pentru a genera un număr random și a alege cartea respectivă. Pentru a continua, vom avea nevoie să ne redenumim pozele cu cărți. Recomand numirea de la 2 la 14, 14-le fiind Asul (ex: ”card2”,”card14”), totul pentru a fi mai ușor de manipulat cărțiile. Generăm 2 numere random până la 15 inclusiv.

```
int a = r.nextInt(13) + 2;
int b = r.nextInt(13) + 2;
```

Vom avea nevoie și de două variabile care reține scorul la fiecare moment (trebuie declarate cu valoarea 0). Acum vom trece la compararea cărțiilor, numerelor generate. Avem 3 cazuri:

a) Jucătorul1 câștigă și se adaugă +1 la scorul lui
b) Jucătorul2 câștigă și se adaugă +1 la scorul lui
c) Egalitate, unde nu se atribuie niciun punct nimănui

Mai jos este codul folosit pentru compararea cărțiilor, o comparație foarte simplă.

```
if(a > b){
    score1++;
    score_left.setText(String.valueOf(score1));
}else if(a < b){
    score2++;
    score_right.setText(String.valueOf(score2));
}else {
    Toast.makeText(MainActivity.this,"RĂZBOI",Toast.LENGTH_SHORT).show();
}
```

Ce este nemaivăzut este linia de cod de pe ramura de egalitate. Așa că haideți să aflăm ce face mai exact acea linie de cod:

Toast: Toast este o clasă în Android care oferă o modalitate simplă de a afișa mesaje temporare (de obicei mici) în partea de jos a ecranului.

makeText(): Este o metodă statică a clasei Toast folosită pentru a crea o instanță a obiectului Toast. Această metodă primește trei parametri:

1) Context (în cazul tău, MainActivity.this): Este contextul în care toast-ul va fi afișat. MainActivity.this face referire la activitatea curentă, iar un Context este necesar pentru a afișa un Toast.
2) Textul mesajului ("RĂZBOI"): Este conținutul mesajului pe care dorești să-l afișezi.
3) Durata (Toast.LENGTH_SHORT): Specifică cât timp va fi afișat Toast. În acest caz, Toast.LENGTH_SHORT înseamnă că mesajul va fi afișat pentru o perioadă scurtă de timp.

show(): Este o metodă a obiectului Toast care declanșează afișarea efectivă a mesajului pe ecran. După apelul acestei metode, mesajul Toast va fi afișat pentru durata specificată.

Acum vom trece la alegerea imaginiilor pentru cărți în funcție de numerele generate random. Pentru început încercați voi să scrieți instrucțiunile pentru a selecta imaginea dorită din ”drawable”. În cazul în care nu vă descurcați, mai jos este linia de cod pentru una dintre cărți, linie de cod pe care vom încerca să o înțelegem.

```
int leftImg = getResources().getIdentifier("card" + a,
"drawable",getPackageName());
card_left.setImageResource(leftImg);
```

1)getResources(): Este o metodă a clasei Context din Android, care oferă acces la resursele aplicației, cum ar fi imagini, șiruri de caractere, layout-uri etc.

2)getIdentifier("card" + a, "drawable", getPackageName()): Această metodă este folosită pentru a obține ID-ul resursei corespunzătoare imaginii pe baza unui nume de resursă și tip de resursă. În acest caz, se construiește un nume de resursă prin concatenarea șirului "card" cu valoarea variabilei a. Tipul de resursă este "drawable", iar getPackageName() furnizează numele pachetului aplicației.

i. ->"card" + a: Aceasta construiește numele resursei prin concatenarea șirului "card" cu valoarea variabilei a. De exemplu, dacă a are valoarea 1, numele resursei va fi "card1".
ii. ->"drawable": Specifică tipul de resursă ca fiind o imagine (drawable).

iii. ->getPackageName(): Oferă numele pachetului aplicației.

Această metodă returnează ID-ul resursei sau zero dacă resursa nu este găsită.

3)int leftImg = …: Se stochează ID-ul resursei în variabila leftImg. Această variabilă va fi utilizată pentru a referenția imaginea în continuare.

4)card_left.setImageResource(leftImg): Această linie de cod setează imaginea pentru un obiect ImageView denumit card_left. setImageResource() primește ID-ul resursei și îi atribuie acea imagine ImageView-ului. Astfel, card_left va afișa acum imaginea identificată de leftImg.

Iar după ce am scris și aceste lucruri la butonul de ”TURN”, ne rămâne de făcut doar butonul pentru resetarea jocului. Când vom apăsa acest buton va trebui să se reseteze scorurile, iar cărțile să fie din nou întoarse. Acest lucru se poate realiza cu ajutorul a ceea ce am scris până acum, nimic nou.

Pentru a exersa mai mult puteți încerca să aduceți următoarele schimbări:

1) Puteți adăuga o miză la fiecare joc
2) În cazul în care un jucător câștigă cu o diferență mai mare de 10 puncte primește 2 puncte.
3) Resetarea automată a jocului când unul din jucători ajunge la 10 puncte
4) Adăugarea a încă unui jucător nou.

Resurse folosite în cadrul acestui laborator:

1) https://developer.android.com/codelabs/basic-android-kotlin-compose-first-app#0
2) https://developer.android.com/codelabs/build-your-first-android-app#0
3) https://developer.android.com/studio/intro
4) https://developer.android.com/develop/ui/views/components/button
5) https://developer.android.com/develop/ui/views/layout/declaring-layout
6) https://www.edureka.co/blog/android-studio-tutorial/
7) https://developer.android.com/codelabs/basic-android-kotlin-training-birthday-card-app-image#0