MOSIS - Laboratorijska vežba 1

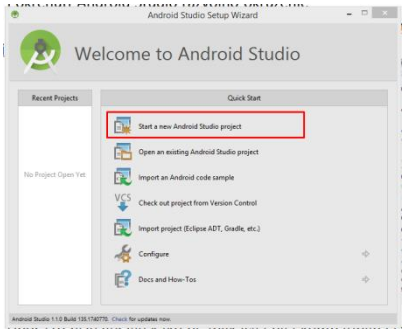
Laboratorijske vežbe iz predmeta MOSIS su koncipirane tako da prate računske vežbe i služe za proveru koncepata programiranja aplikacija namenjenih Android operativnom sistemu koji su na njima obrađeni. Imajući u vidu da je Android kompletan operativni system, teško je obraditi sve koncepte koji se prilikom programiranja aplikacija javljaju u toku jednog semestra, tako da će pažnja biti usmerena na one ključne dok se za naprednije mora konsultovati literatura (npr. ona preporučena na Moodle stranici kursa).

Laboratorijska vežba 1 je uvodna i namenjena je upoznavanju sa osnovnim konceptima programiranja za Android operativni sistem namenjen „pametnim telefonima“.

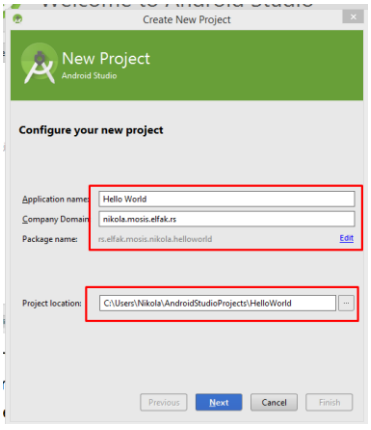
Ciljevi ove vežbe su:

1. Upoznavanje sa *Android Studio* razvojnim okruženjem
2. Kreiranje Hello World Android projekta
3. Upoznavanje sa strukturom i bitnim fajlovima android projekta
4. Kreiranje emulatora korišćenjem *AVD* menadžera
5. Kreiranje emulatora korišćenjem *Genymotion* emulatora
6. Pokretanje aplikacije
7. Provera i upoznavanje sa životnim ciklusom aplikacije

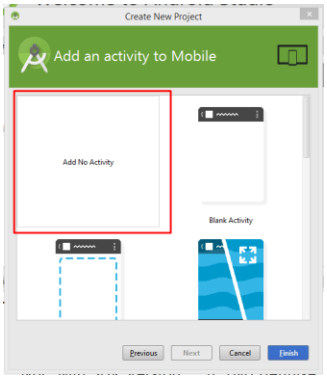
1. Upoznavanje sa *Android Studio* razvojnim okruženjem
   a. Pokrenuti *Android Studio* razvojno okruženje
   b. Upoznati se sa glavnim delovima okruženja (meni, stavke menija itd.)
2. Kreiranje *Hello World Android* projekta
   a. Kreirati novi projekat korišćenjem *Quick Start* prozora ili (*File->New Project->Android project*).
   b. Definisati Application name i definisati ime Company domain na osnovu koga će biti kreiran package name.
      Application Name: Hello World
      Company domain: *ime*.mosis.elfak.rs
   c. Izaberite lokaciju svog projekta (odaberite D particiju na laboratorijskom računaru).



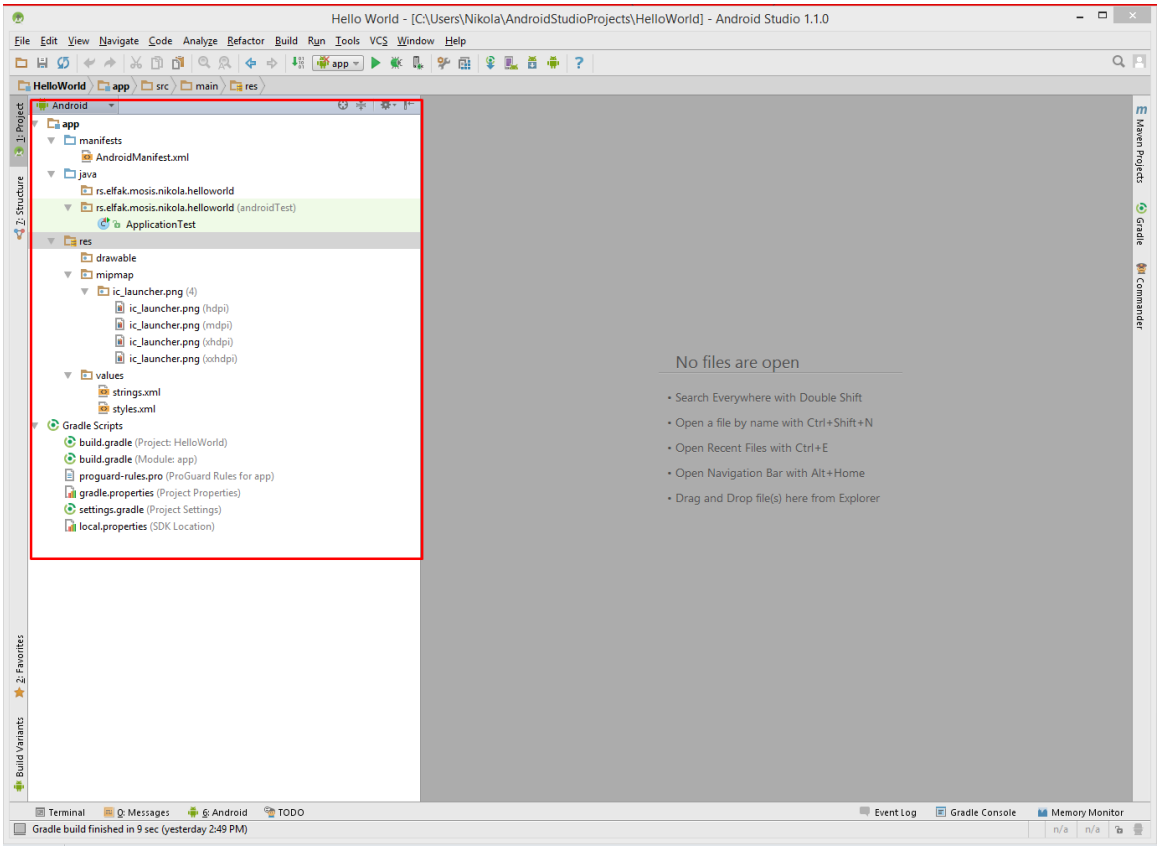
   d. Na sledećem ekranu wizarda izabrati željenu verziju androida kao minimalnu verziju sa kojom će projekat biti kompatibilan. Npr. *API 8 Android 2.2 (Froyo)* iz spiska *Minimum SDK* za *Phone and Tablet*. Generalno se preporučuje izbor što niže verzije kako bi aplikacija bila kompatibilna sa što većim brojem postojećih sistema. *Android Studio* daje procenu koliko uređaja će biti obuhvaćeno izabranom verzijom.



e. Izaberite *Add No Activity* kao tip *Activity*-a koji će inicijalno biti dodat u aplikaciju i klinite na *Finish* dugme.



f. Nakon što je kreiranje projekta završeno, biće prikazano okruženje sa kreiranim projektom.

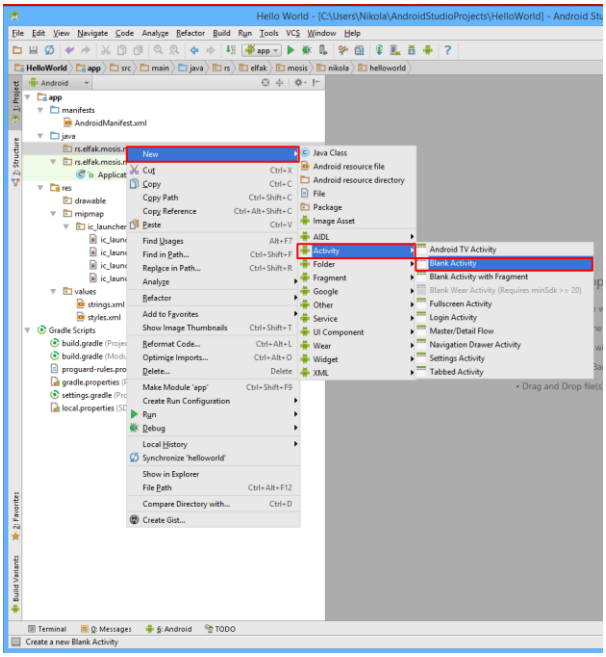


3. Detaljno proučite kreiranu aplikaciju. Hijerarhija direktorijuma je inicijalno u Android Studiju značajno jednostavnija nego kod projekta kreiranog u Eclipse razvojnom okruženju:
    a. *app*: folder koji sadrži fajlove namenjene izvornom kodu aplikacije i kao i resurse potrebne aplikaciji. Ovaj folder sadrži nekoliko potfoldera.
        i. *java*: sadrži hijerarhiju paketa i .java klase koje se dodaju u projekat.
            1. *rs.elfak.mosis.nikola.helloworld*: Sadrži sve klase projekta.
            2. *rs.elfak.mosis.nikola.helloworld (Android test)*: Sadrži klase u kojima se pišu testovi projekta
        ii. *manifests*: folder u kome se nalazi AndroidManifest.xml fajl.
        iii. *res*: ovaj folder će sadržati sve resurse koje će aplikacija koristiti. U startu sadrži tri osnovna foldera koje okruženje podrazumevano kreira. Svaki foldera može da ima više različitih verzija u zavisnosti od namene (multimedija, orijentacija ekrana, regionalna podešavanja jezika itd.)
            1. *drawable*: sadrži slike i animacije koje se koriste u aplikaciji (moguće je umesto jednog imati više ovakvih foldera u zavisnosti od rezolucija ekrana mobilnih telefona koje je potrebno podržati). Ovakvi folderi počinju rečju *drawable* i sadrže slike za ekrane različite gustine piksela) .
            2. *mipmap*: sadrži folder u kome se nalaze ikone aplikacije namenjene različitim gustinama ekrana (*hdpi*, *mdpi*, *xhdpi* i *xxhdpi*).

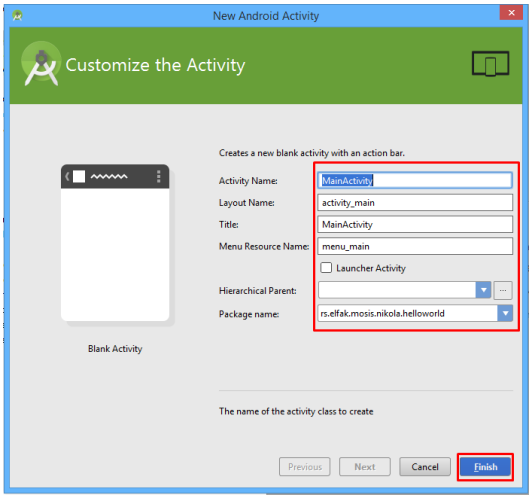
3. values: sadrži value type resurse poput stringova, celih brojeva, nizova (sve konstante koje se koriste u kodu, preferira se njihovo navodjenje u resursima u odnosu na kod)
    a. Inicijalno sadrži strings.xml i styles.xml u kome su definisani stringovi i stilovi potrebni HelloWorld aplikaciji.

b. *Gradle Scripts*: folder koji sadrži skripte namenjene *Gradle build* sistemu. *Android Studio* sadrži *Gradle build* sistem u kome je moguće podešavati izvore koda (source control koji se koristi, biblioteke koje se preuzimaju sa servera itd.), izvore resursa, skinove aplikacije. Gradle skripte se programiraju korišćenjem *Groovy* skripti. U ovoj vežbi ćemo koristiti podrazumevana podešavanja za build aplikacije.

4. *Activity* je jedinstvena, fokusirana aktivnost koju korisnik može izvršavati. Koristi se za interakciju sa korisnikom. *Activity* služi za kreiranje prozora (podloge) na koju je moguće smestiti elemente korisničkog interfejsa sa kojima korisnik interreaguje. Dodati u aplikaciju *MainActivity.java*.
    a. Desnim klikom kliknuti na package aplikacije i dodati *Blank Activity*.



    b. Prihvatite podrazumevani unos i kliknite *Finish*.



c. Primetićete da je dodata klasa *MainActivity.java* u *package* aplikacije kao i novi potfolder *layout* sa *activity_mail.xml* fajlom. Potfolder *layout* sadrži sve xml opise korisničkog interfejsa i može imati svoje pandane za različite veličine ekrana i/ili orijentacije (layout-large, layout-xlarge-land itd.). Korisniči interfejs *Activity*-a je moguće menjati direktno из dizajnera ili menjanjem njegovog xml opisa. Dodat je i folder *menu* koji u kome je definisan meni dodatog *Activity*-a. *Activity* dodat u *Android Studio* okruženju podrazumevano ima *Action Bar* i *menu* o kojima će biti reči na narednim laboratorijskim vežbama.

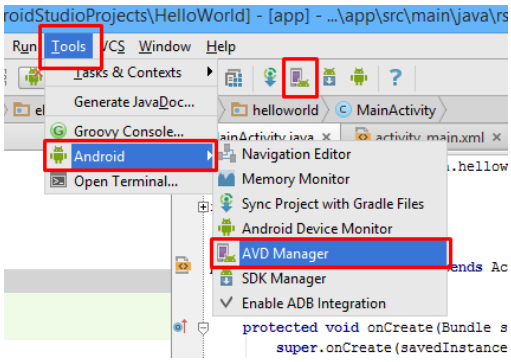
5. *Activity* je jedinstvena, fokusirana aktivnost koju korisnik može izvršavati. Koristi se za interakciju sa korisnikom.

   i.
   ii. gen: sadrži fajlove koje ADT automatski generiše.
   iii. R.java: autogenerisana klasa koja služi za pristup resursima.

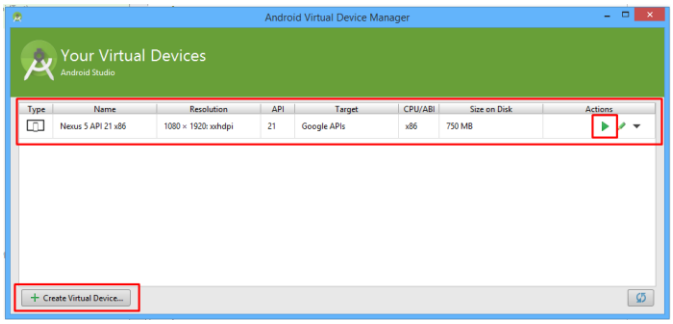
   b. assets: sadrži asset fajlove, slično kao kod resursa, ali im se pristupa kao fajlovima korišćenjem AssetManager klase preko koje se učitavaju u vidu strimova.
   c. AndroidManifest.xml: svaki projekat mora da sadrži ovaj fajl istog imena i sadrži sve podatke neophodne za izvršavanje aplikacije (privilegije, dozvole, listu Activity, Service, Broadcast Reciever i Content Provider klasa itd.)

   d. default.properties: ovaj fajl sadrži podešavanja projekta.

6. Pre pokretanja aplikacije potrebno je napraviti emulator, ovo nije potrebno vršiti svaki put već samo onda kada je potrebno kreirati emulator sa odgovarajućim karakteristima projekta.
   a. Startujte *AVD Manager* (*Tools > Android > AVD Manager* ili izaberite ikonu toolbar-a).



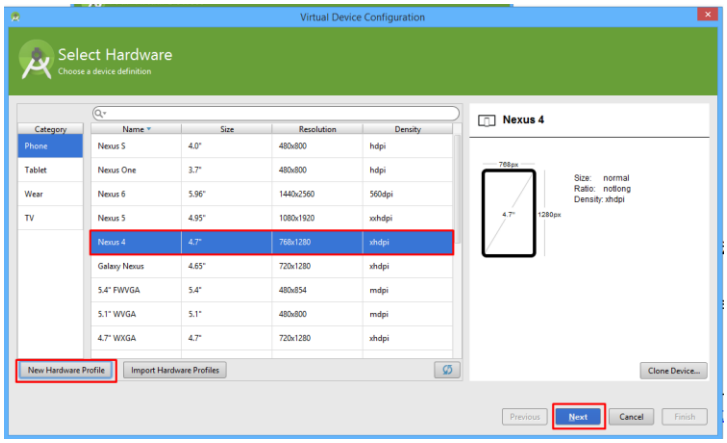
b. Redom su prikazani kreirani emulatori pri čemu je inicijalno kreiran jedan.



c. Startovanje kreiranog emulatora se vrši klikom na *play* dugme

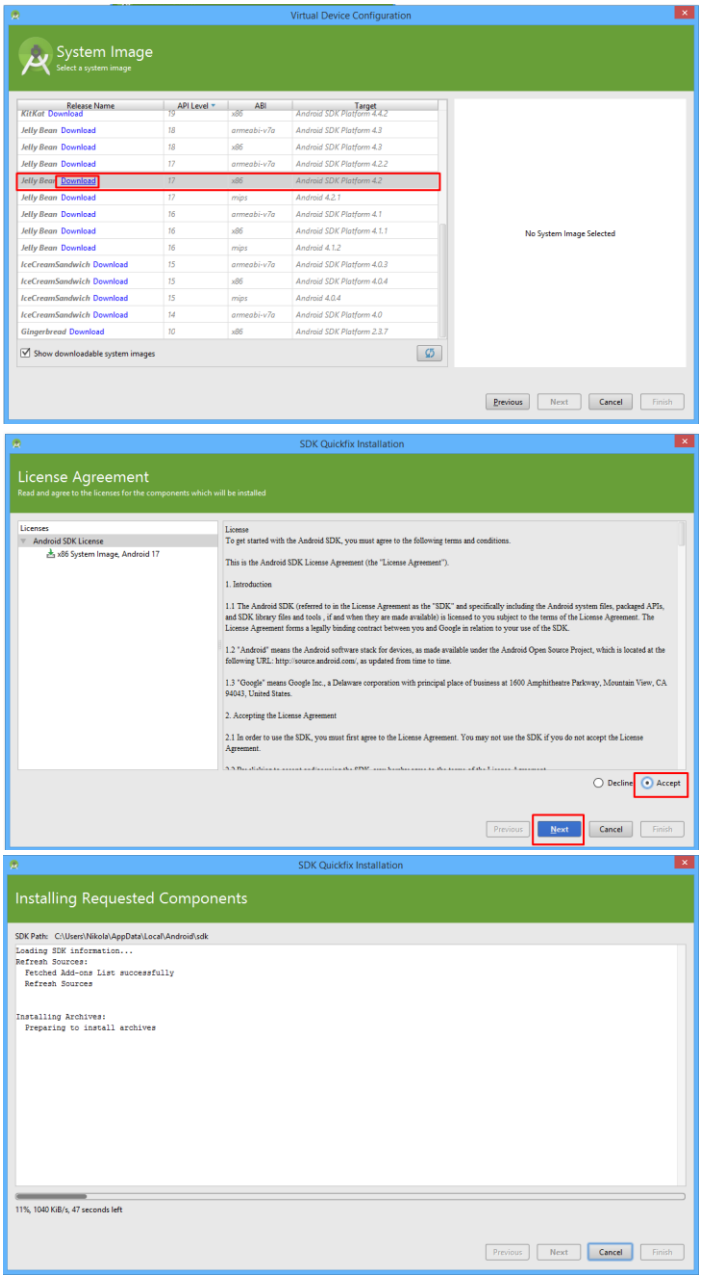
d. Ukoliko želite da kreirate novi, kliknite *Create Virtual Device* dugme.

e. Izaberite jednan od unapred preipremljenih hardverskh profile ili kreirajte željeni hardverski profil klikom na dugme *New Hardware Profile*.

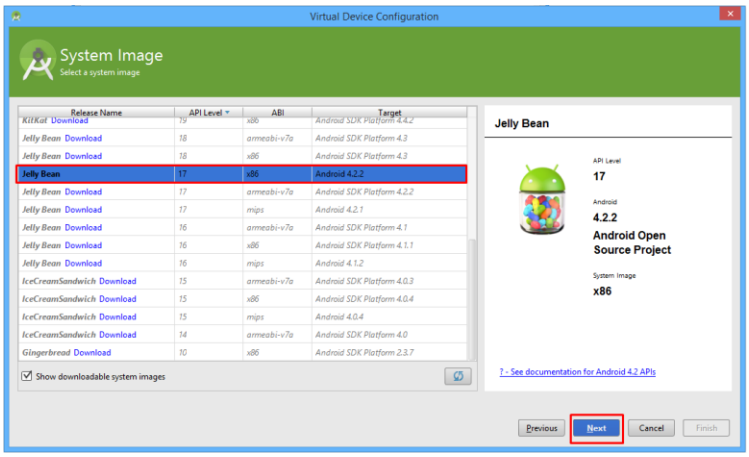


f. Kao primer, izaberite Nexus 4 hardverski profil i kliknite na dugme *Next.*

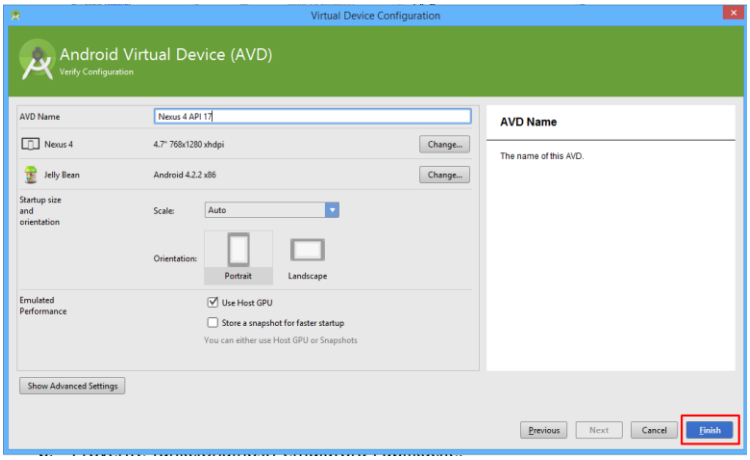
g. Izaberite verziju androida koju želite da koristite na emulatoru (Npr. originalnu *Android* 4.2 verziju za *Nexus 4*). Ukoliko ta verzija ne postoji na lokalnoj verziji *Android SDK*, potrebno ju je skinuti sa Internet-a klikom na dugme *Download.*



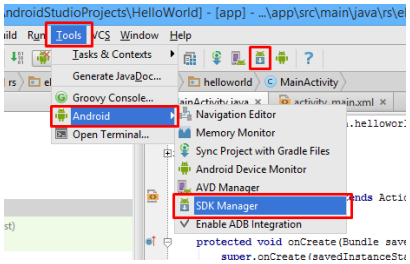
h. Nakon završetka preuzimanja izaberite verziju *Lollipop* i kliknite dugme *Next*.



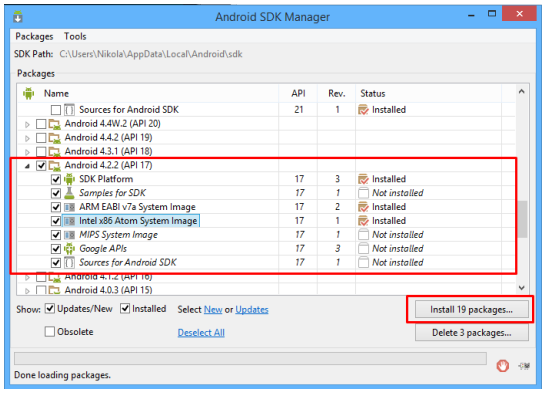
i. Potvpdite podešavanja koja su unapred izabrana i kliknite na dugme *Finish*.

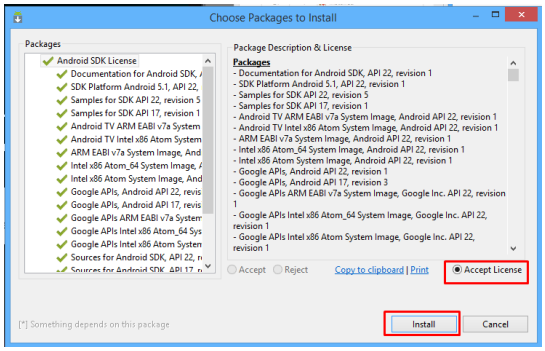


j. Rezultat kreiranja emulatora će biti prikazan u dijalogu. Ukoliko emulator sa datim podešavanjima nije učitan, kao što je prikazano na slici, potrebno je instalirati ciljanu Android verziju korišćenjem *SDK Manager* aplikacije. a. Startujte *SDK Manager* (*Tools > Android > SDK Manager* ili izaberite ikonu toolbar-a)



k. Izaberite instalaciju Android verzije 17 čekiranjem odgovarajuće stavke a zatim kliknite na dugme *Install*.

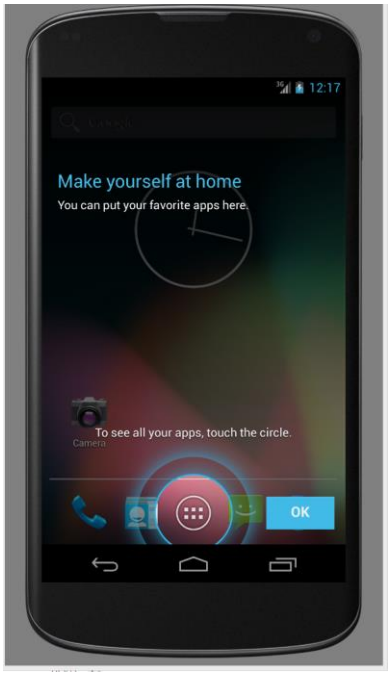




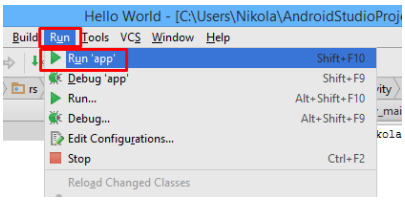
l. Ukoliko i nakon instaliranja odgovarajuće verzije korišćenjem *SDK Manager*-a restartujte *Android Studio*

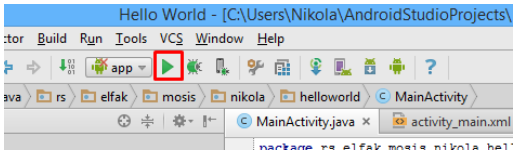
m. Pokrenite kreirani emulator klikom na *play* dugme.

n. Emulatoru je potrebno vreme da se pokrene. Nemojte ga gasiti tokom razvoja. Svaki put kada budete pokrenuli aplikaciju iz okruženja, ona će biti ponovo učitana.

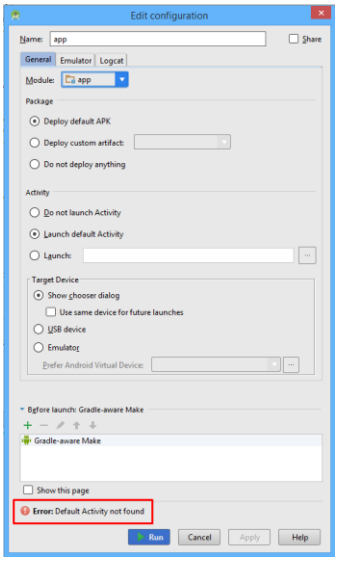


7. Pokrenite aplikaciju izborom *Run -> Run* ili izaberite ikonicu iz *toolbar*-a.

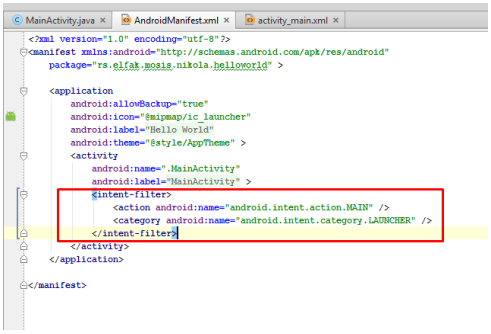




a. Pošto je *MainActivity.java* klasa dodata naknadno, okruženje je ne prepoznaje kao podrazumevani *Activity* aplikacije.

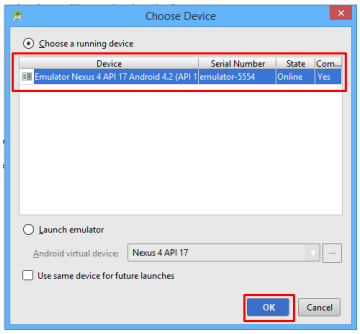


b. Potrebno je editovati *AndroidManifest.xml*. Otvoriti *AndroidManifest.xml* i u delu gde je naveden *MainActivity* dodati kod kao na slici.



c. Pokrenuti aplikaciju ponovo.

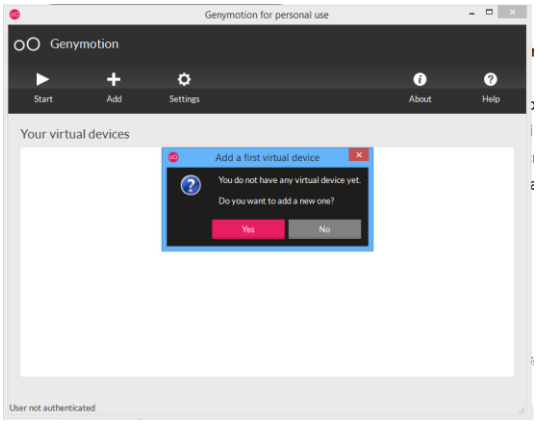
d. Izabrati emulator željeni emulator i kliknuti dugme *OK*.



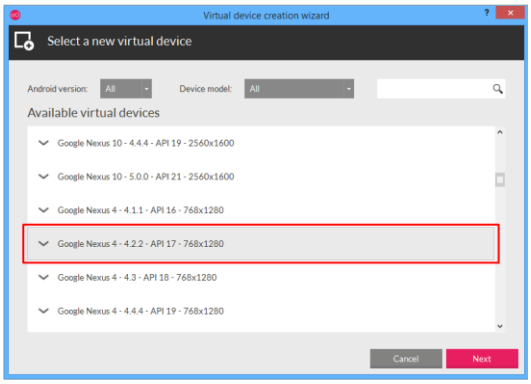
e. Čestitamo, ovo je Vaša prva Android aplikacija (ukoliko niste ranije probali nešto slično).

f. Proverite funkcionalnosti emulatora i aplikacije.

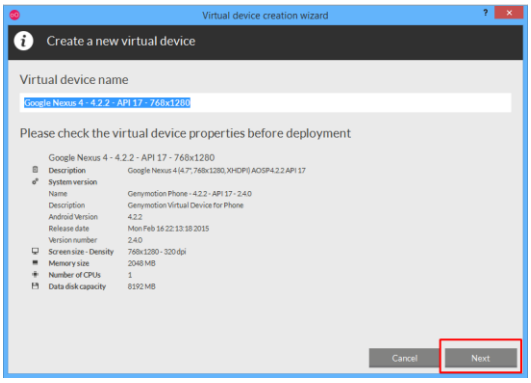
8. Alternativno, na laboratorijskim vežbama ćemo koristiti *Genymotion* emulator. Ovaj emulator se izvršava na Oracle Virtual Box virtuelnoj mašini i često je jako brz i udobrniji za testiranje. Zahteva procesore koji hardverski podržavaju virtuelizaciju. Ukoliko na Vašem računaru ne postoji hardverska podrška za virtuelizaciju, nije moguće koristiti ovaj emulator. Ukoliko virtuelizacija nije uključena, uključite je u podešavanjima za *BIOS* matične ploče.
   a. Pokrenite *Genymotion* emulator. Ukoliko nema kreiranih emulatora kreirajte novi emulator.



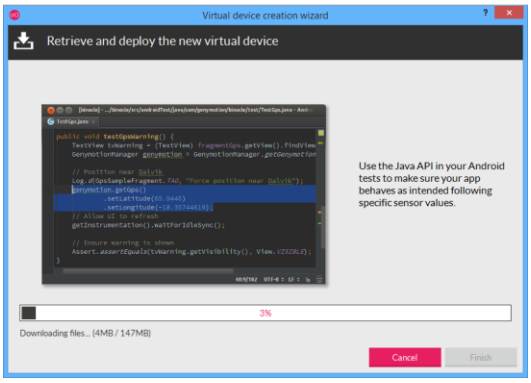
   b. Izaberite jedan od ponuđenih uređaja (Npr. Google Nexus 4 – 4.2.2 – API 17).



c. Potvrdite postavke uređaja klikom na dugme *Next*.

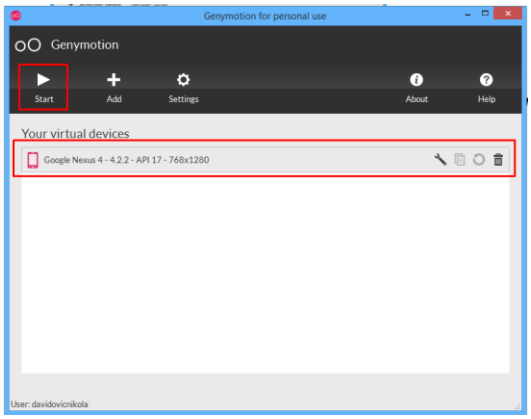


d. Virtualni uređaj se automatski preuzima sa servera *Genymotion*-a.

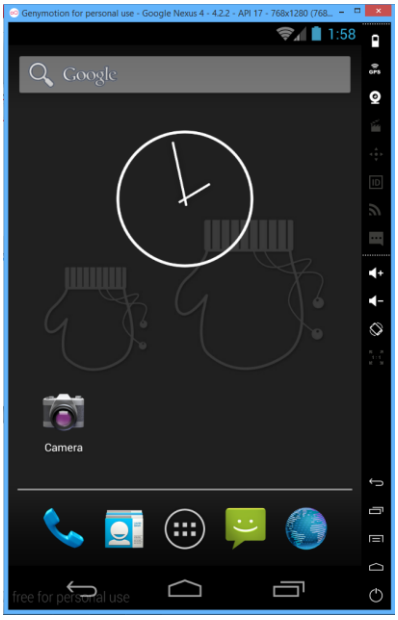


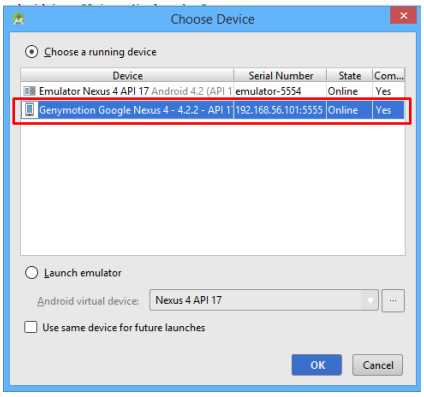
e.

f. Startovati emulator izborom odgovarajuće virtuelne mašine i kliknuti na dugme *play.*

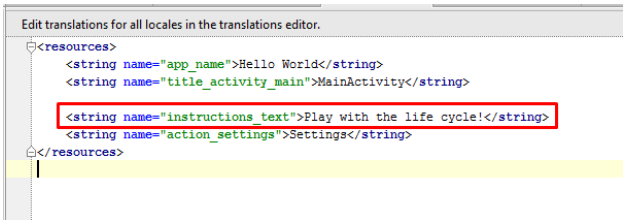


g. Startovani emulator je moguće koristiti iz *Android Studio* okruženja

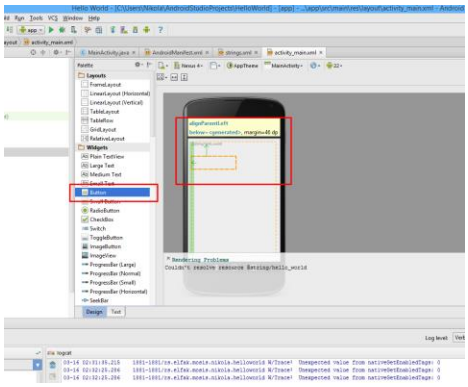




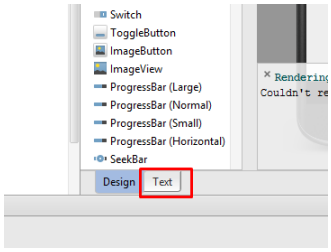
9. Upoznajte se sa životnim ciklusom Android aplikacije:
    a. Otvorite strings.xml i u njemu promenite hello_world resurs kao na slici. Dodajte I natpis koji će biti prikazan na dugmetu.



    b. Otvorite *activity_main.xml*. Uočite da je inicijalno otvoren dizajner korisničkog interfejsa. Iz dizajnera dodajte *Button* element.

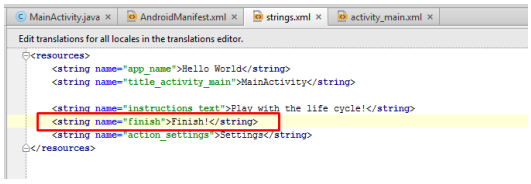


c. Otvorite *activity_main.xml* fajl izvorom *Text tab*-a.



d. Proučite *activity_main.xml.* Izmenite ime string resursa.

e. Dodajte u strings.xml text dugmeta.



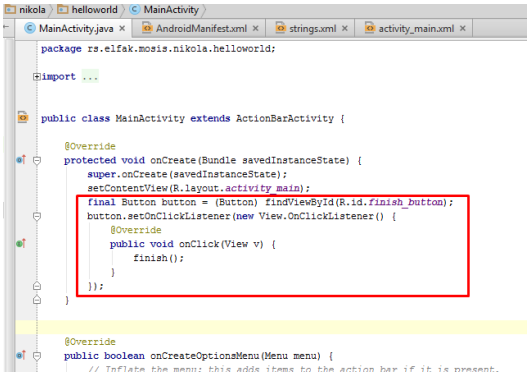
f. Povežite tekst dugmeta *string* resursa sa dodatim dugmetom.

g. Izmenite *id* dodatog dugmeta u *finish_button*.

```
    android:layout_height="wrap_content"
    android:id="@+id/textView" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/finish"
    android:id="@+id/finish_button"
    android:layout_below="@+id/textView"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="58dp" />
```

h. U MainActivity.java klasi dodajte odgovarajuće importe i dodajte onClick metod kreiranom dugmetu.



```
package rs.elfak.mosis.nikola.helloworld;

import ...

public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final Button button = (Button) findViewById(R.id.finish_button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
```

i. Za svaku ključnu metodu vezanu za životni ciklus android aplikacije napravite Override (onRestart(), onStart(), onResume(), onPause(), onStop(), onDestroy()).

j. Proverite kako se aplikacija ponaša.

k. Uskladite kod kao što je prikazano na slici. Svaki metod treba da ima poziv Toast.makeText(this, "*ime_funkcije_u_kojoj_se_nalazi*", Toast.LENGTH_SHORT).show(); i odgovarajući poziv metodi nadklase (Npr. super.onStart()).

```
        }

        return super.onOptionsItemSelected(item);
    }

    @Override
    public void onRestart(){
        super.onRestart();
        Toast.makeText(this, "onRestart", Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onStart(){
        super.onStart();
        Toast.makeText(this, "onStart", Toast.LENGTH_SHORT).show();
    }

    @Override
    public  void  onResume()
    {
        super.onResume();
        Toast.makeText(this, "onResume", Toast.LENGTH_LONG).show();
    }

    @Override
    public void onPause()
    {
        Toast.makeText(this, "onPause", Toast.LENGTH_SHORT).show();
        super.onPause();
    }

    @Override
    public void onStop()
    {
        Toast.makeText(this, "onStop", Toast.LENGTH_SHORT).show();
        super.onStop();
    }

    @Override
    public void onDestroy()
    {
        Toast.makeText(this, "onDestroy", Toast.LENGTH_SHORT).show();
        super.onDestroy();
    }
}
```

l. Pratite dešavanje aplikacije u emulatoru nakon pozivanja telefonskog broja, pritiska na Back dugme itd.