Android programiranje

Marko Arsenović

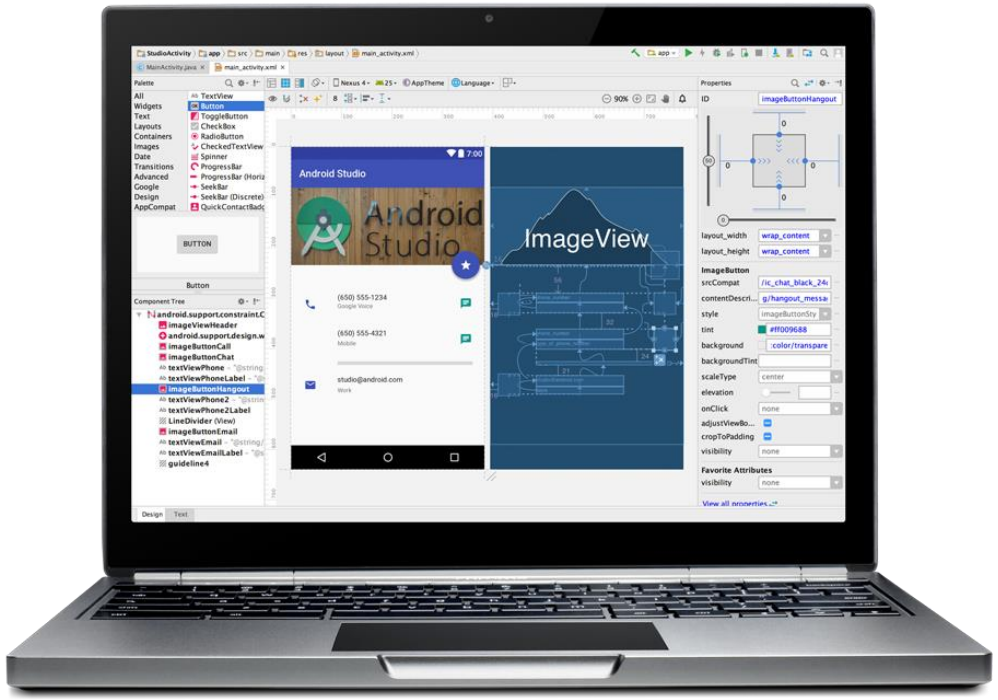
Srđan Sladojević

Sadrzaj

- Android Studio IDE
- Prva aplikacija
- Activity

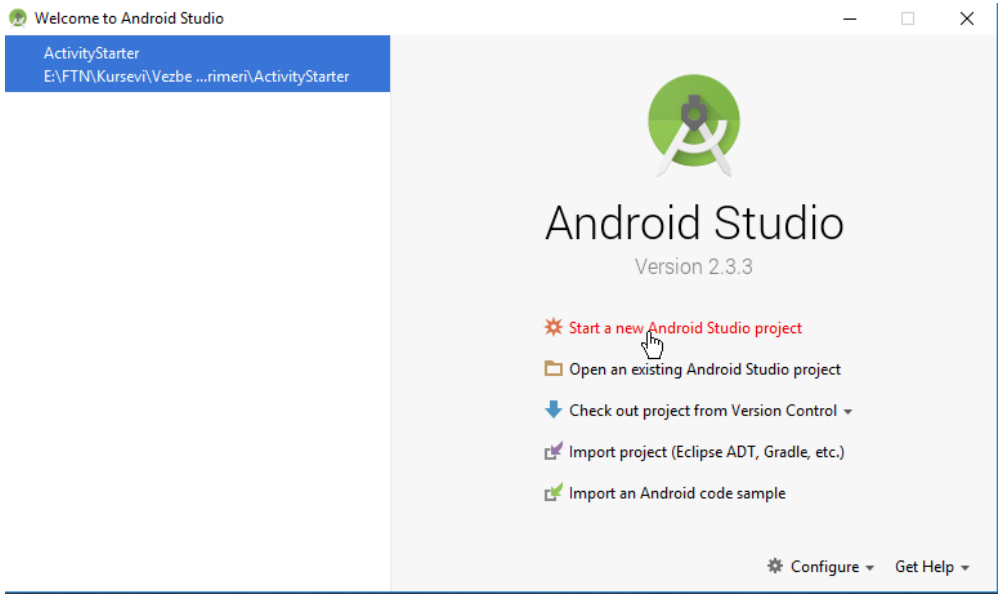
Android Studio IDE

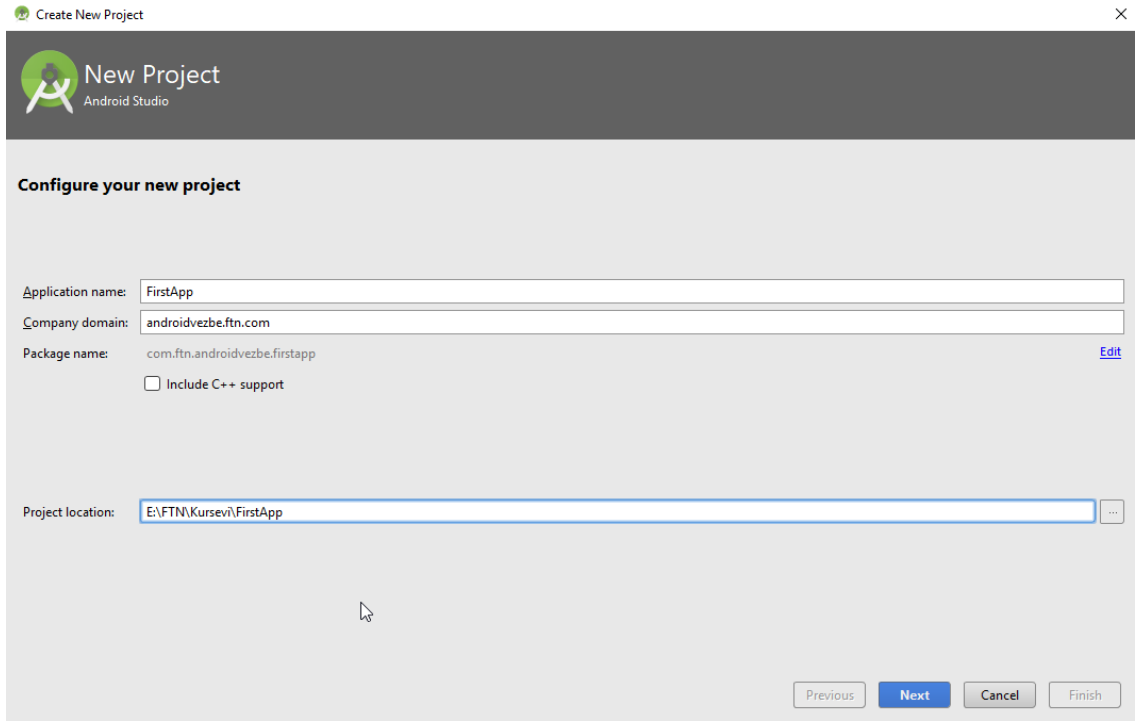
- Android Studio IDE koristi se za razvoj Android aplikacija, postalo je standardno razvojno okruzenje koje je zamenilo zastareo Eclipse ADT resenje.
- Instalacija
- Baziran na IntelliJ IDEA



Android Studio IDE

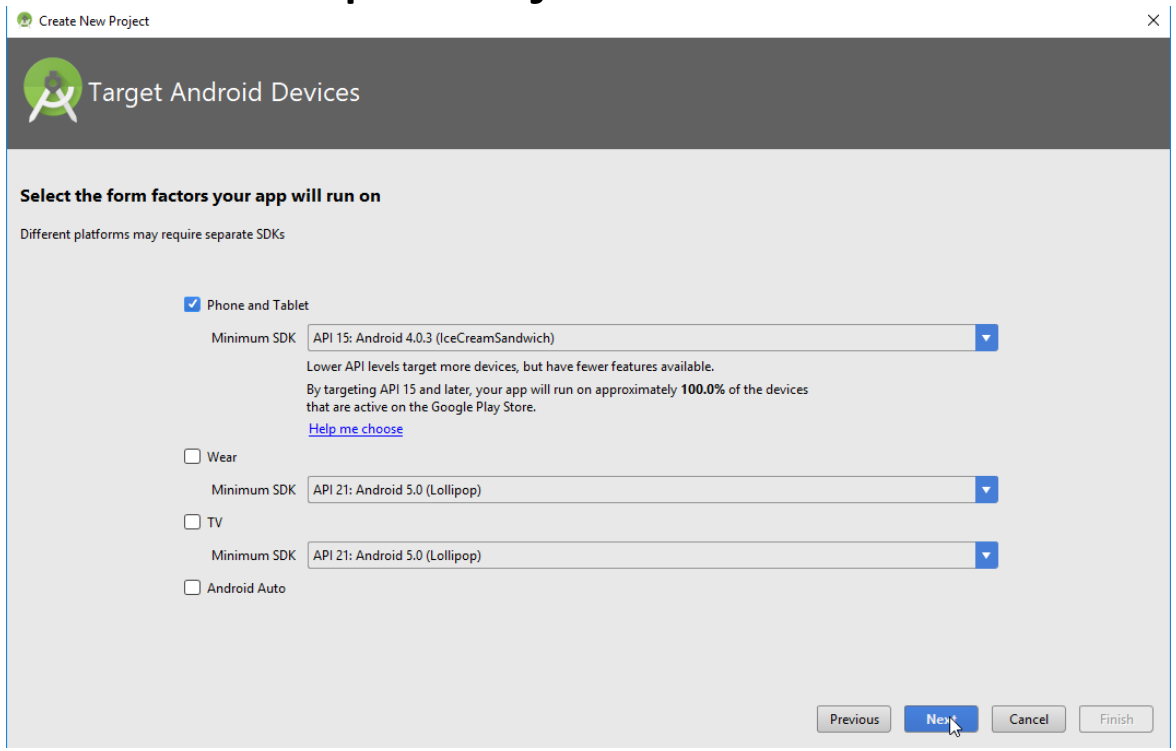
- Prva aplikacija

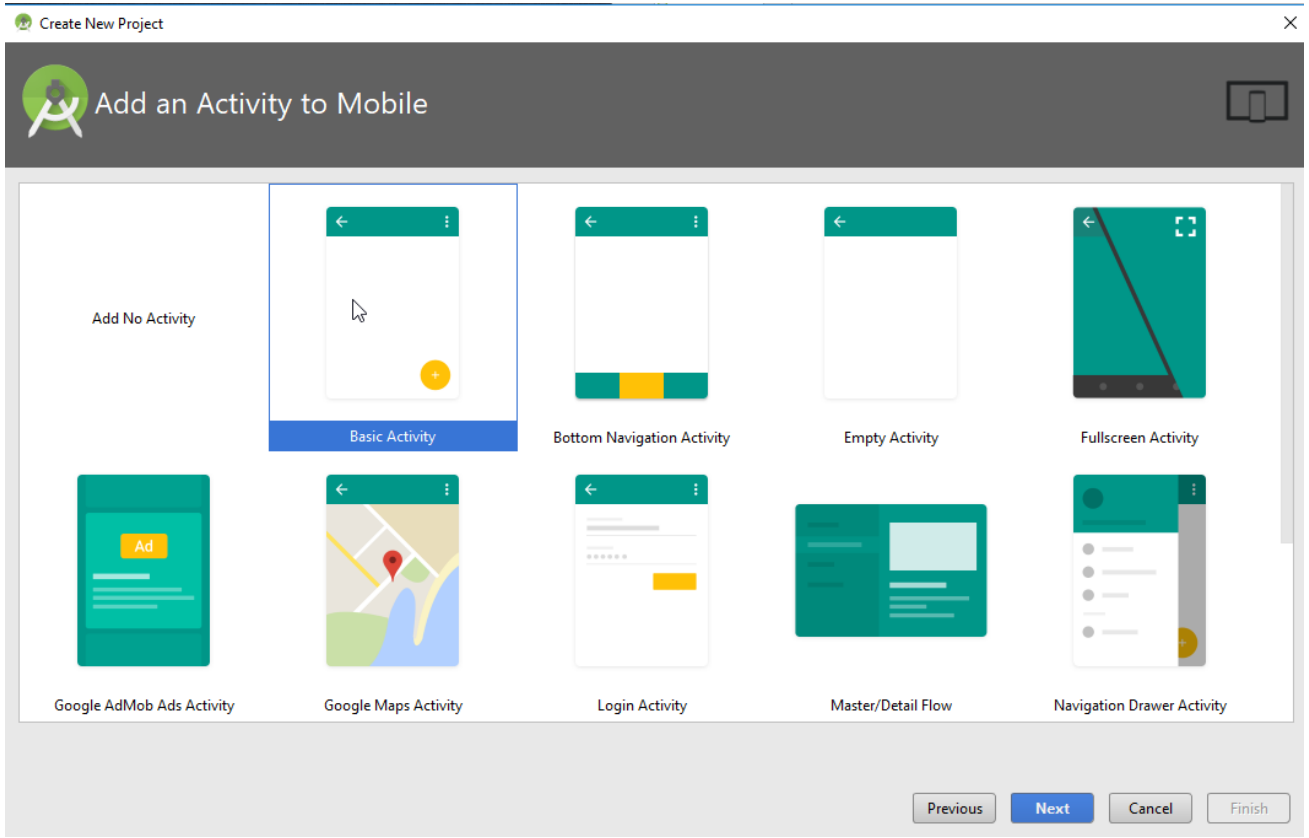




Android Studio IDE

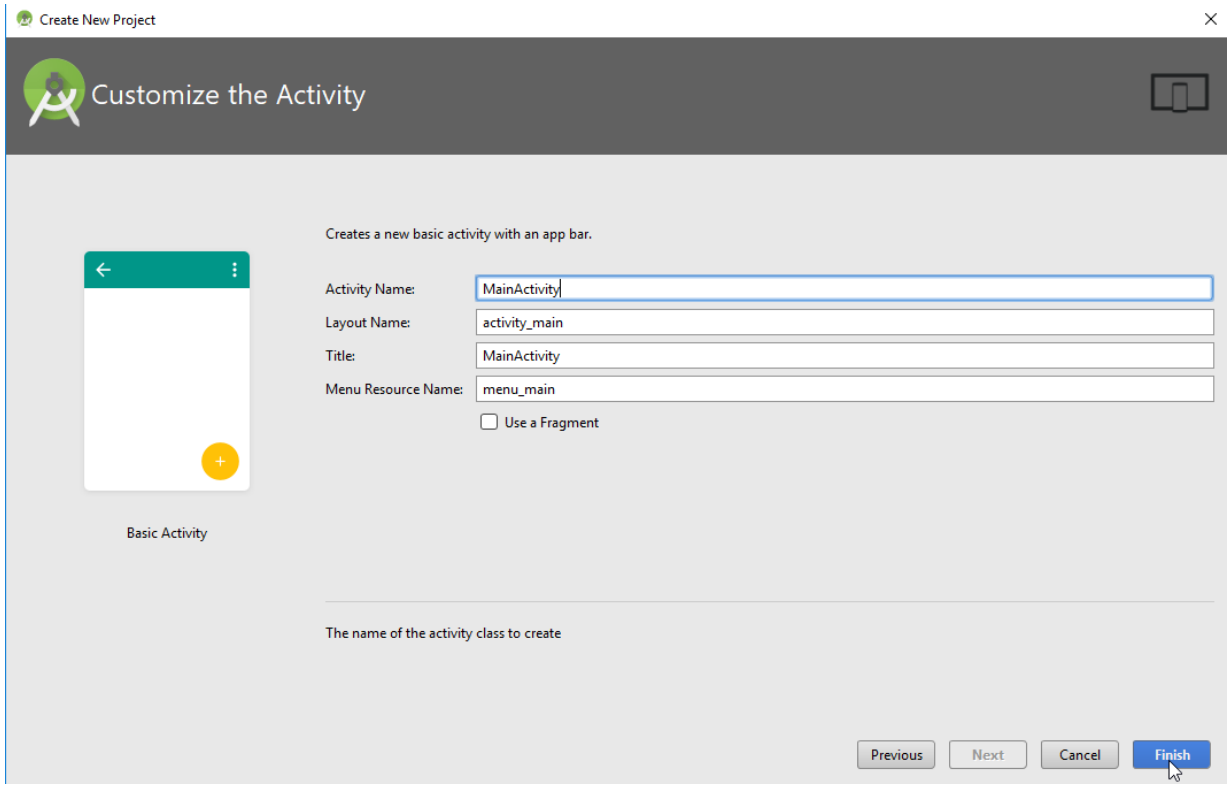
- Prva aplikacija





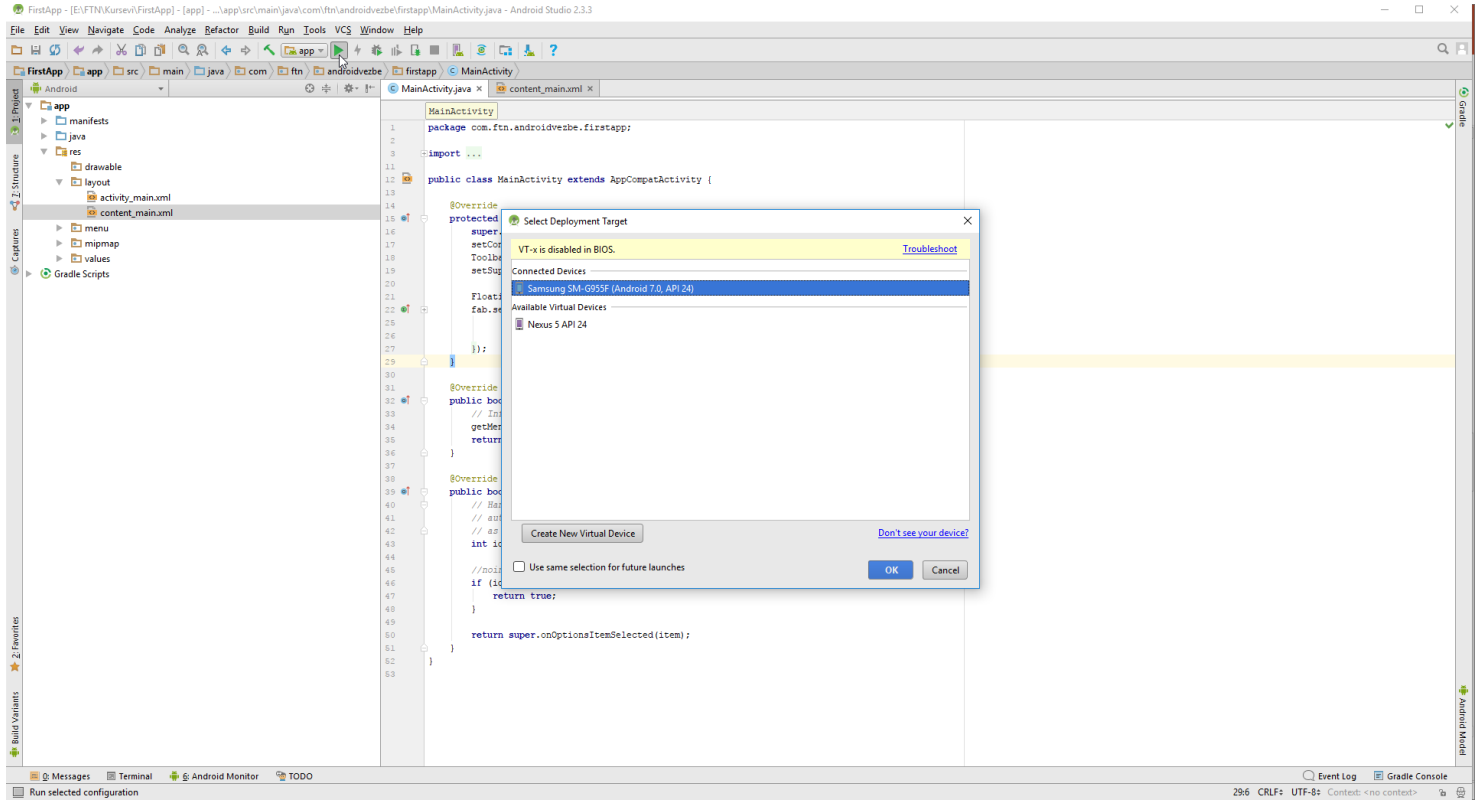
Android Studio IDE

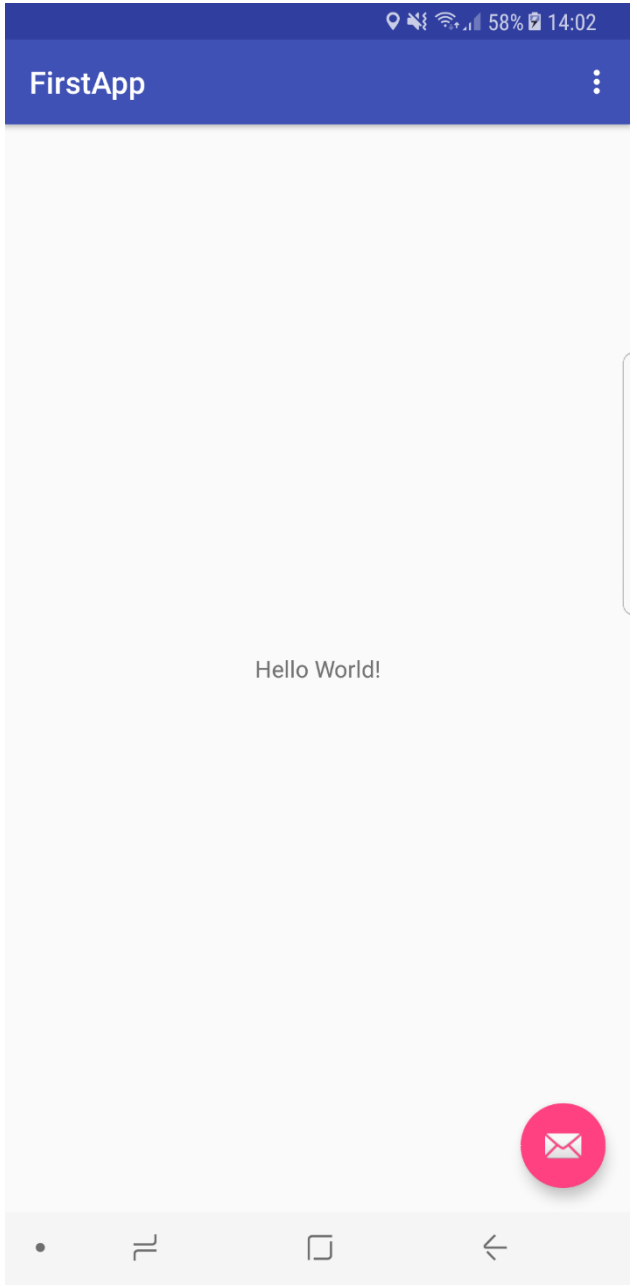
- Prva aplikacija



Android Studio IDE

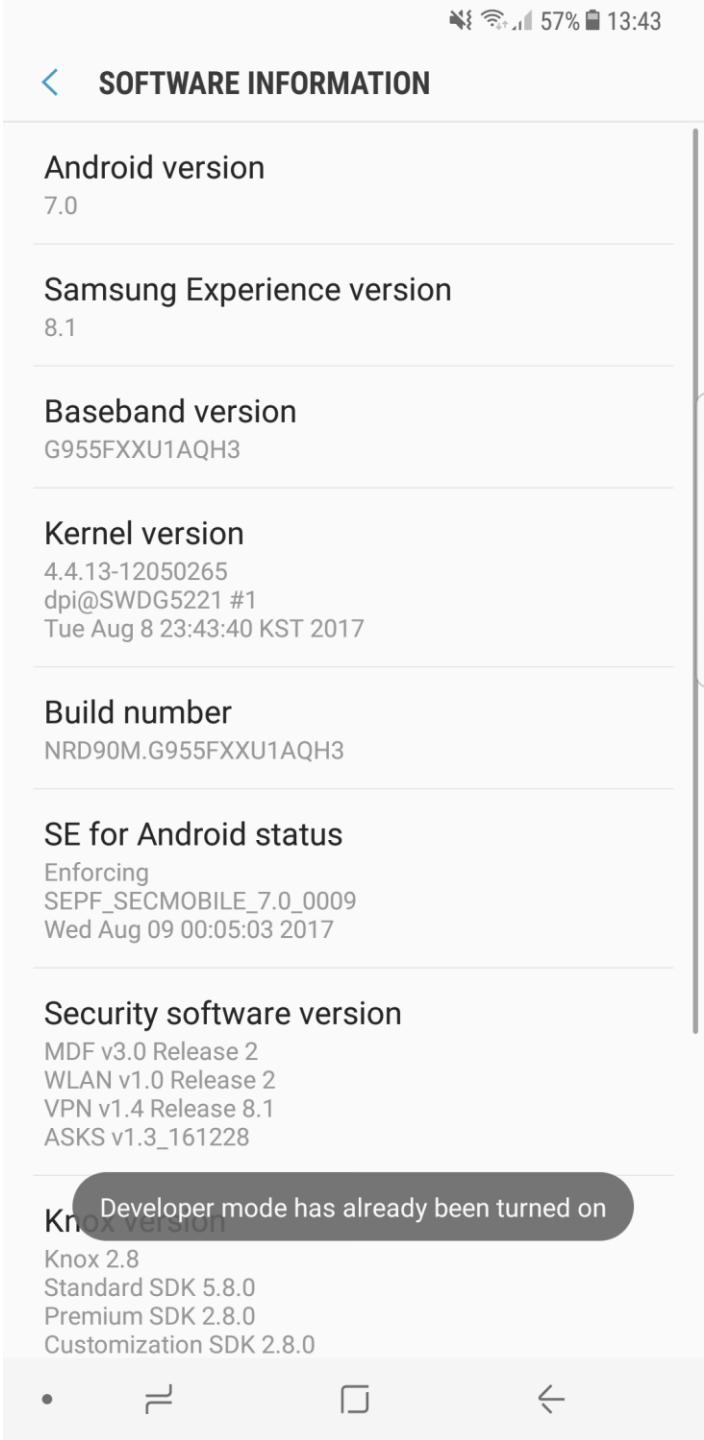
- Prva aplikacija





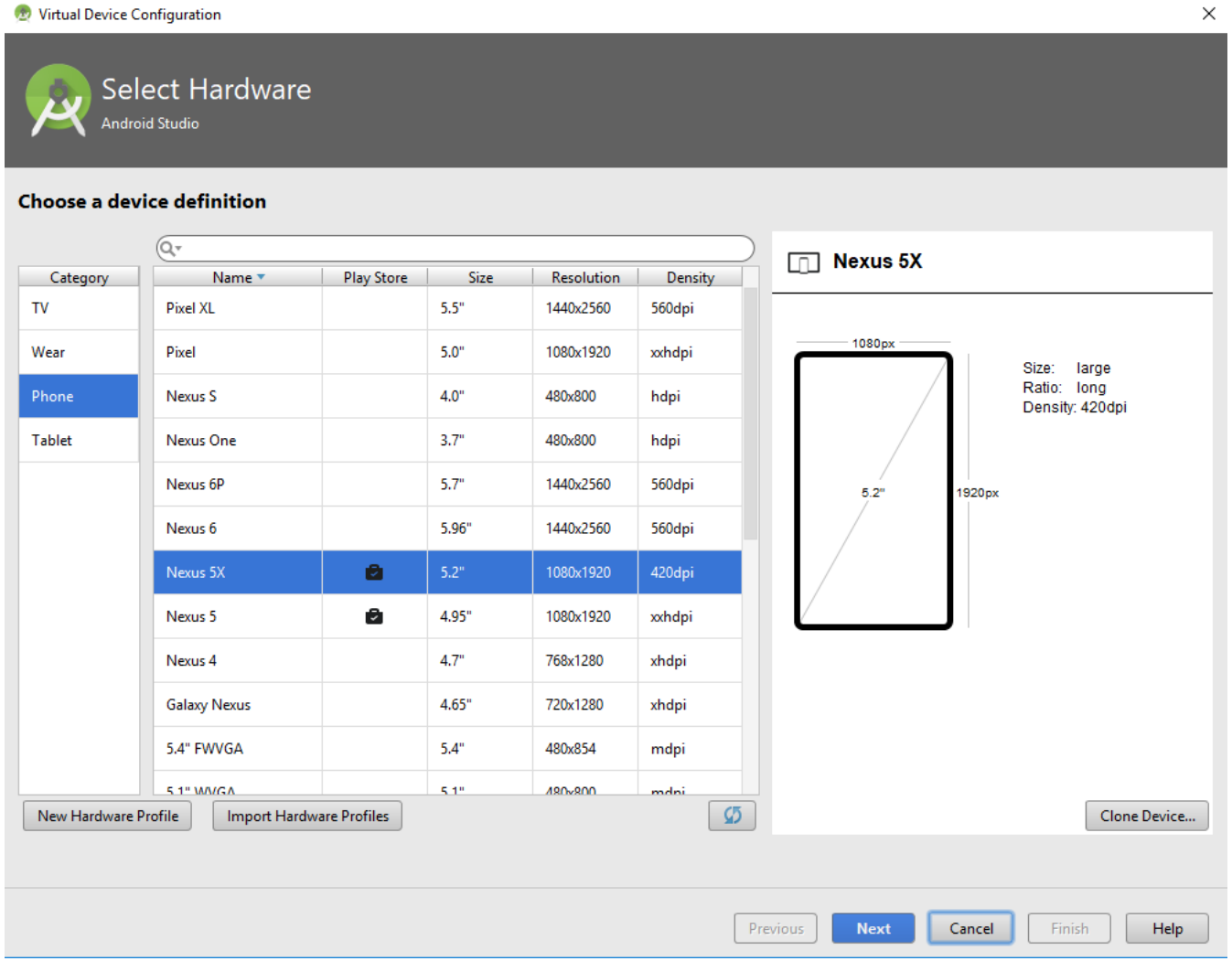
Android Studio IDE

- Prva aplikacija
- Testiranje na uredjaju
- Developer mode na Android mobilnom uredjaju (moze se razlikovati u zavisnosti od uredjaja i verzije Android-a)



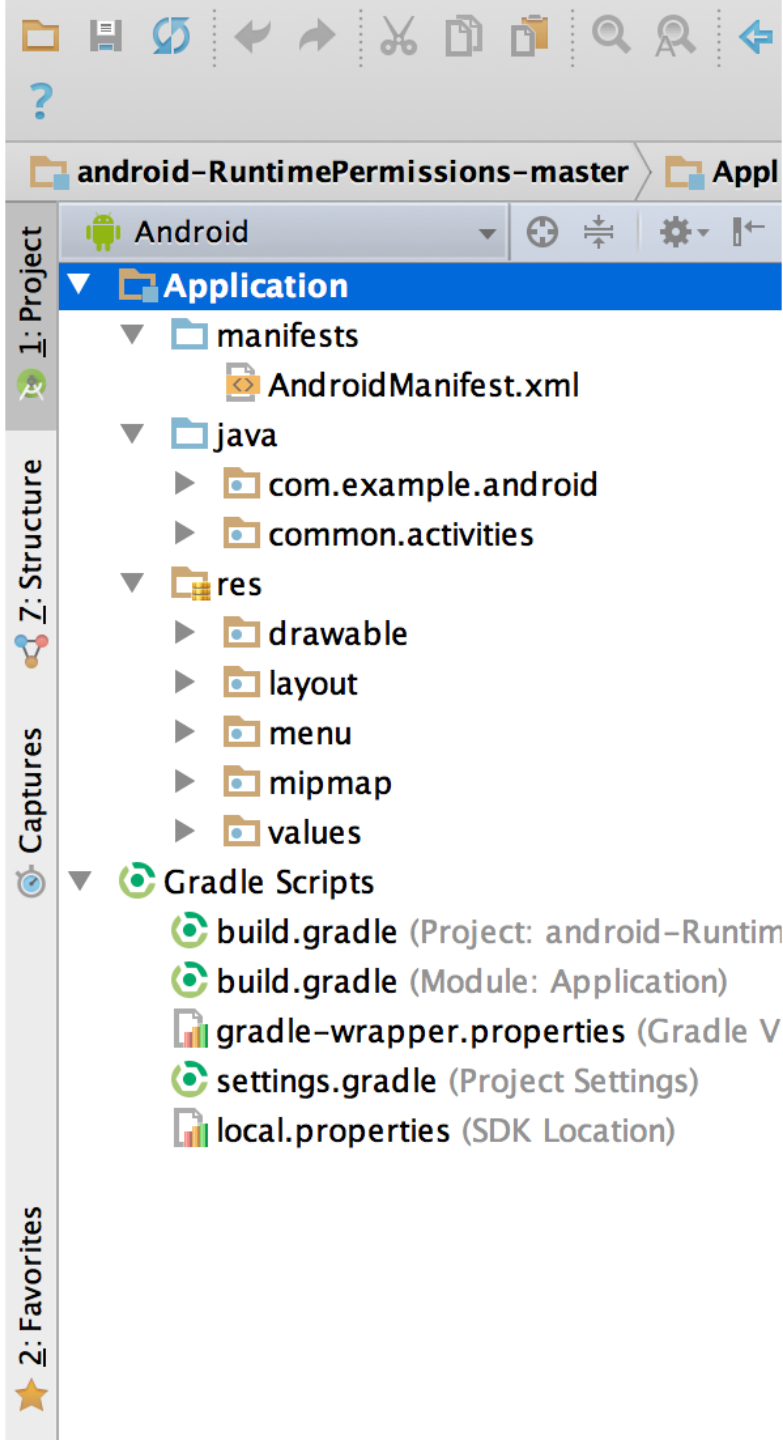
Android Studio IDE

- Prva aplikacija
- Kreiranje emulatora



Android Studio IDE

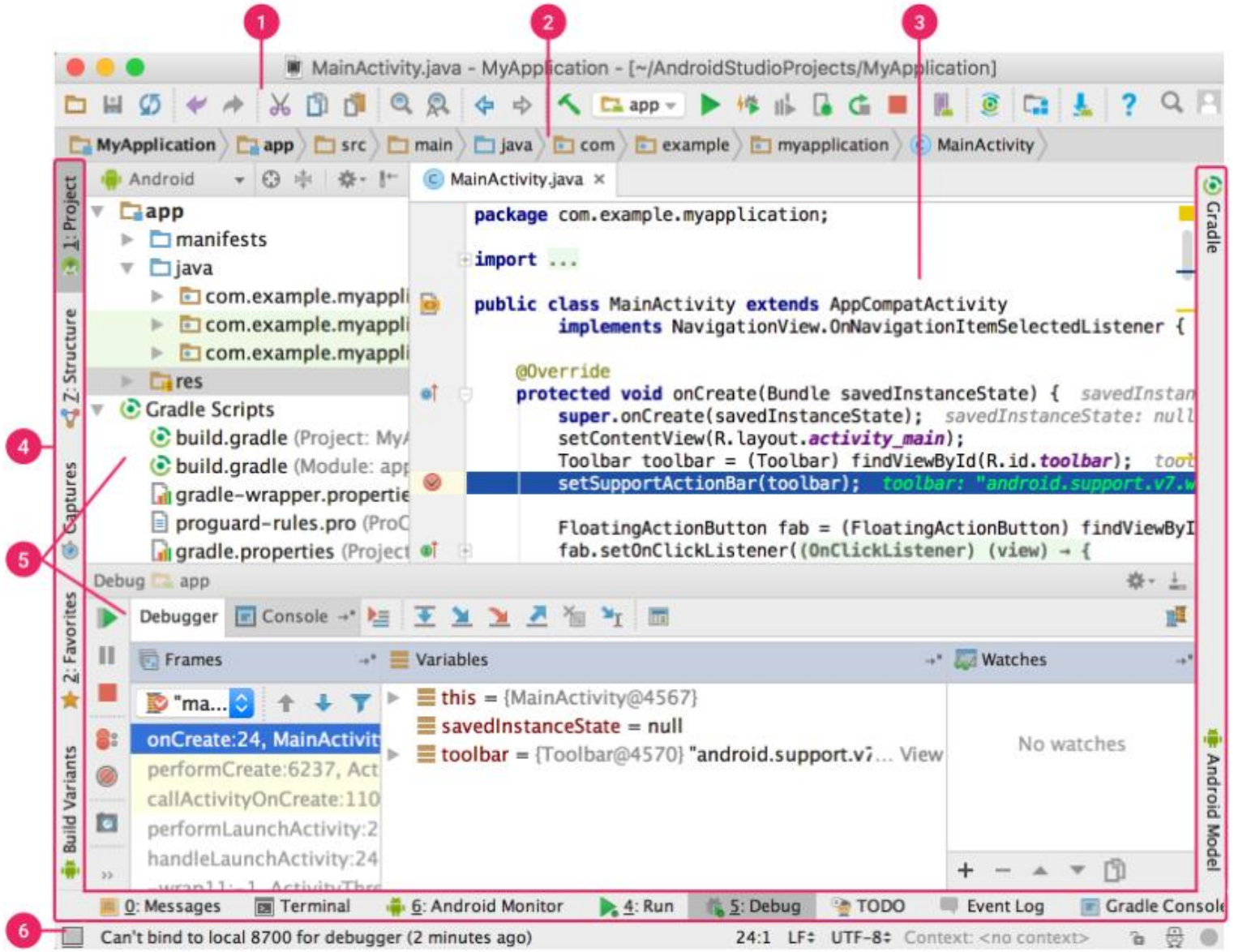
- Struktura projekta
- Android Studio prikazuje izvorni kod i resurse u tri modula:
  - Android app modules (default)
  - Library modules
  - Google App Engine modules
- Svi build fajlovi se nalaze unutar Gradle Scripts
- Android app module sadrzi:
  - manifests – sadrzi AndroidManifest.xml
  - java – sadrzi sve Java izvorne kodove sa JUnit test kodovima
  - res – sadrzi se reusurse: XML layouts, UI strings, bitmap slike



Android Studio IDE

- User Interface

1. toolbar
2. navigation bar
3. editor window
4. tool window bar
5. tool windows
6. status bar



Activity - Deklaracija

- Aktivnosti kao i druge kopmonente aplikacije deklarisani su u XML fajlu koji se zove AndroidManifest.xml. Deklarisanje aktivnosti je nacin na koji govorimo sistemu o nasoj aktivnosti.
- Bilo koja aktivnost ili komponenta koja nije deklarisana u manifestu nece biti ukljucena u aplikaciju, pokusaj pristupa ili upotrebe nedeklarisane komponente ce rezultirati generisanjem izuzetka pri pokretanju.

AndroidManifest.xml

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="ActivityLifecycle"
        android:theme="@style/AppTheme" >
        <activity
            android:name="androidvezbe.ftn.com.activitylifecycle.MainActivity"
            android:label="ActivityLifecycle" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Activity - Zivotni ciklus

- Aktivnost je pojedinacna fokusirana stvar koju korisnik moze da uradi
- Aktivnost predstavlja pojedinacan ekran Android aplikacije
- Komponenta Activity moze da se nalazi u tri stanja:
  - aktivnost se izvrsava (resumed)
  - aktivnost je pauzirana (paused)
  - aktivnost je zaustavljena (stopped)

Activity - Zivotni ciklus

- Aktivnost se izvrsava ako se nalazi u prvom planu i ima fokus.
- Aktivnost je pauzirana ako se druga aktivnost nalazi u prvom planu i ima fokus, ali je prva aktivnost jos uvek vidljiva (zato to je druga aktivnost transparentna ili ne pokriva ceo ekran).
- Pauzirana aktivnost je “ziva” (instanca klase je zadrzana u memoriji i povezana je sa rukovaocem prozora), ali moze biti “ubijena” ako sistem ima jako malo slobodne memorije.
- Aktivnost je zaustavljena ako se nalazi u pozadini (potpuno je prekrivena drugom aktivnoscu).
- Zaustavljena aktivnost je “ziva” (instanca klase je zadrzana u memoriji, ali nije povezana sa rukovaocem prozora), ali moze biti “ubijena” ako sistem ima malo slobodne memorije.

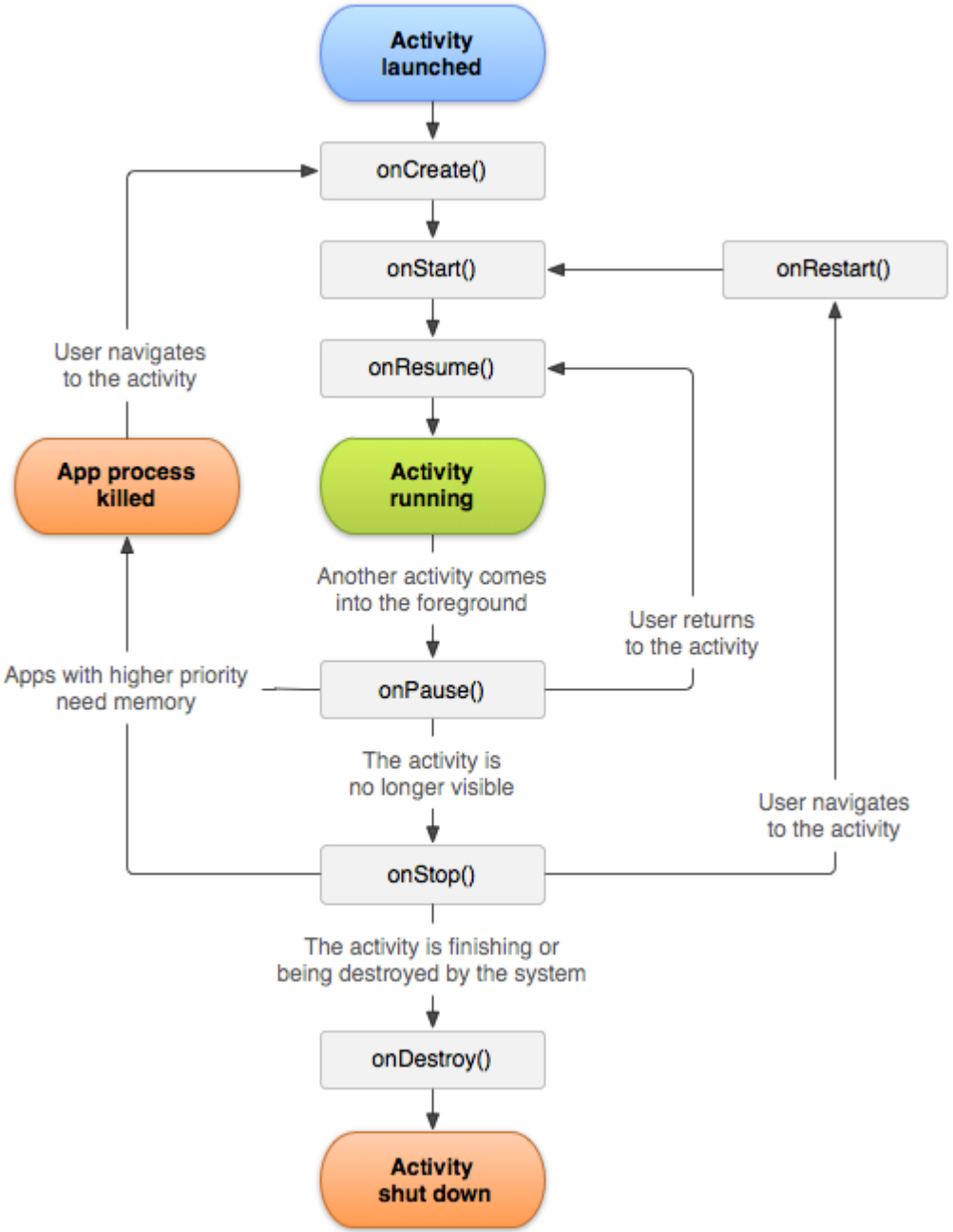
Activity – Zivotni ciklus

- Aktivnost se nalazi u stanju active kada je interfejs dostupan za korisnika. Ovo stanje traje od metode onResume() do metode onPause() koji se javlja kada druga aktivnost prelazi u prednji plan. Ako nova aktivnost ne remeti u potpunosti nasu aktivnost, onda ce aktivnost ostati u stanju paused dok se nova aktivnost ne zavrsi ili ponisti. Tada se odmah poziva metod onResume() I aktivnost nastavlja svoj zadatak.

- Kada nova pokrenuta aktivnost popuni ekran ili ucini nasu aktivnost nevidljivom ota aktivnost ulazi u fazu stopped I njen nastavak ce uvek ukljucivati oziv metode onRestart()

- Ako se aktivnost nalazi u paused ili stoped, OS ce je ukloniti iz memorije kada preostane malo prostora u memoriji ili kada je memorija potrebna za druge aplikacije.

- Mi ne vidimo rezultat onDestory(), jer je aktivnost uklonjena u ovoj tacki. Sa isFinishing() proveravamo da li je aktivost samo pauzirana ili je iskljucena.



Activity - Zivotni ciklus

- onCreate
- Sistem poziva onCreate metodu kada stvara aktivnost.
- Ova metoda treba da zauzme resurse i inicijalizuje komponente neophodne za pravilno funkcionisanje aktivnosti.
- Pozivom setContentView metode iscrtava se korisnicki interfejs.

```java
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Set the user interface layout for this Activity
    // The layout file is defined in the project res/layout/main_activity.xml file
    setContentView(R.layout.main_activity);
```

Activity - Zivotni ciklus

- onRestart metoda se poziva nakon sto je aktivnost zaustavljena, a pre nego sto je ponovo startovana

```
@Override
protected void onRestart() {
    super.onRestart();  // Always call the superclass method first

    // Activity being restarted from stopped state
}
```

Activity - Zivotni ciklus

- Sistem poziva onStart metodu neposredno pre nego sto aktivnost postane vidljiva korisniku.

```
@Override
protected void onStart() {
    super.onStart();  // Always call the superclass method first
```

Activity - Zivotni ciklus

- onResume metoda se poziva neposredno pre nego sto aktivnost pocne interakciju sa korisnikom. U ovom trenutku aktivnost se nalazi na vrhu steka aktivnosti.

```
@Override
public void onResume() {
    super.onResume();  // Always call the superclass method first

}
```

Activity - Zivotni ciklus

- Sistem poziva onPause metodu neposredno pre nego sto pauzira izvrsavanje aktivnosti.
- Ova metoda se obicno koristi za snimanje perzistentnih podataka i zaustavljanje procesa koji zauzimaju procesor.
- Mora biti vrlo brza zato sto sledeca aktivnost ne moze da pocne da se izvrsava sve dok se ova metoda ne zavrsi.

```
@Override
public void onPause() {
    super.onPause();  // Always call the superclass method first


}
```

Activity - Zivotni ciklus

- onStop
- Poziva se kada aktivnost vise nije vidljiva korisniku.

```
@Override
protected void onStop() {
    super.onStop();  // Always call the superclass method first


 }
```

Activity - Zivotni ciklus

- onDestroy
- Poslednja metoda koja se poziva pre nego sto se aktivnost unisti.
- Ova metoda oslobadja zauzete resurse pre nego sto se aktivnost unisti.

- Da bi ikljucili aktivnost direktno, pozivamo njen metod finish() koji poziva onDestroy(). Da bi smo izvrsili istu akcju iz podredjene aktivnosti, koristimo metodu finishFromChild(ActivityChild).

Activity - Zivotni ciklus

- Primer

MainActivity.java

MainActivity | onRestart()

```
6
7   public class MainActivity extends Activity {
8
9       private TextView mTextViewState;
10
11      @Override
12      protected void onCreate(Bundle savedInstanceState) {
13          super.onCreate(savedInstanceState);
14          setContentView(R.layout.activity_main);
15
16          mTextViewState = (TextView)findViewById(R.id.textViewState);
17          mTextViewState.setText("onCreate()\n");
18      }
19
20      @Override
21      protected void onStart() {
22          super.onStart();
23          mTextViewState.append("onStart()\n");
24      }
25
26      @Override
27      protected void onResume() {
28          super.onResume();
29          mTextViewState.append("onResume()\n");
30      }
31
32      @Override
33      protected void onPause() {
34          super.onPause();
35          mTextViewState.append("onPause()\n");
36          if(isFinishing()){
37              mTextViewState.append("...finishing");
38          }
39      }
40
41      @Override
42      protected void onStop() {
43          super.onStop();
44          mTextViewState.append("onStop()\n");
45      }
46
47      @Override
48      protected void onRestart() {
49          super.onRestart();
50          mTextViewState.append("onRestart()\n");
51      }
52
53      @Override
54      protected void onDestroy() {
55          super.onDestroy();
```

62% 14:58

ActivityLifecycle

onCreate()
onStart()
onResume()
onPause()
onStop()
onRestart()
onStart()
onResume()

Activity – pokretanje aktivnosti

- Pokretanje nove aktivnosti pomocu Intent (namera) objekta
- Intent objekat je samo objekat poruke. Intent objekti se mogu upotrebiti za komunikaciju izmedju komponenti aplikacije ali I za komunikaciju izmedju drugih aplikacija na uredjaju.
- Sadrzi svojstva korisna komponenti koja obradjuje nameru (akcija, podaci, dodatne informacije) i sistemu (komponenta, kategorije i oznake).

Activity – pokretanje aktivnosti

- Akcija
- Svojstvo akcija (action) opisuje akciju koja treba da se izvrsi.
- Postavlja se u konstruktoru ili Intent setAction(String action) metodom.
- Preporucuje se koriscenje predefnisanih akcija.

Activity – pokretanje aktivnosti

- Podaci i tip
- Svojstva podaci (data) i tip (type) opisuju podatke koji treba da se obrade i MIME tip tih podataka.
- Postavljaju se u konstruktoru ili Intent setData(URI data), Intent setType(String type) i Intent setDataAndType(Uri data, String type) metodama.
- Zavise od akcije koja treba da se izvrsi.

Activity – pokretanje aktivnosti

- Dodatne informacije (extra)
- Dodatne informacije (extra) potrebne komponenti koja obradjuje nameru opisane su uredjenim parovima (kljuc, vrednost).
- Postavljaju se metodama oblika Intent putExtra(String key, T value).

Activity – pokretanje aktivnosti

- Komponenta
- Opisuje komponentu koja treba da obradi nameru.
- Postavlja se u konstruktoru ili Intent setClassName(String packageName, String className) metodom.
- Ukoliko je ovo svojstvo postavljeno, namera je eksplicitna.

Activity – pokretanje aktivnosti

- Kategorije
- Svojstvo kategorije (categories) opisuje vrstu komponente koja obradjuje nameru.
- Postavlja se Intent addCategory(String category) metodom.
- Jedna namera moze sadrzati vise kategorija.

Activity – pokretanje aktivnosti

- Oznake
- Oznake (flags) sugerisu sistemu kako da startuje aktivnost (npr. kom zadatku treba da pripada) i kako da je tretira nakon sto je startuje (npr. da li treba da se prikaze u spisku nedavnih aktivnosti).
- Postavljaju se metodom Intent setFlags(int args).
- Jedna namera moze sadrzati vise oznaka (onda se oznake postavljaju disjunkcijom predefinisanih vrednosti).

Activity – pokretanje aktivnosti

- Filter namera
- Filter namera (Intent Filter) opisuje mogucnost komponente (namere koje komponenta moze da obradi).
- Sadrzi polja koja odgovaraju svojstvima namere (akcija, podaci, i kategorija).

Activity – pokretanje aktivnosti

- Filter namera
- Kada primi implicitnu nameru da se startuje aktivnost, sistem pronalazi odgovarajuce aktivnosti tako sto poredi nameru i fltere namera na osnovu
  - akcije (akcija specifcirana u nameri mora da odgovara jednoj od akcija specifciranih u flteru),
  - podataka (URI i MIME tip specifcirani u nameri moraju da odgovaraju URI-u i MIME tipu specifciranom u flteru),
  - kategorije (svaka kategorija specifcirana u nameri mora da odgovara jednoj od kategorija specifciranih u flteru; ne mora da vazi obrnuto).
- Namera mora proci sva tri testa da bi bila prosledjena komponenti.
- Jedna komponenta moze sadrzati vise fitera.

Activity – pokretanje aktivnosti

Aktivnost se startuje pozivom startActivity ili startActivityForResult metode.

- Ove metode omoguavaju startovanje navedene aktivnosti (prosledjivanjem eksplicitne namere) ili neke aktivnosti koja je opisana odredjenim svojstvima (prosledjivanjem implicitne namere).

Activity – pokretanje aktivnosti

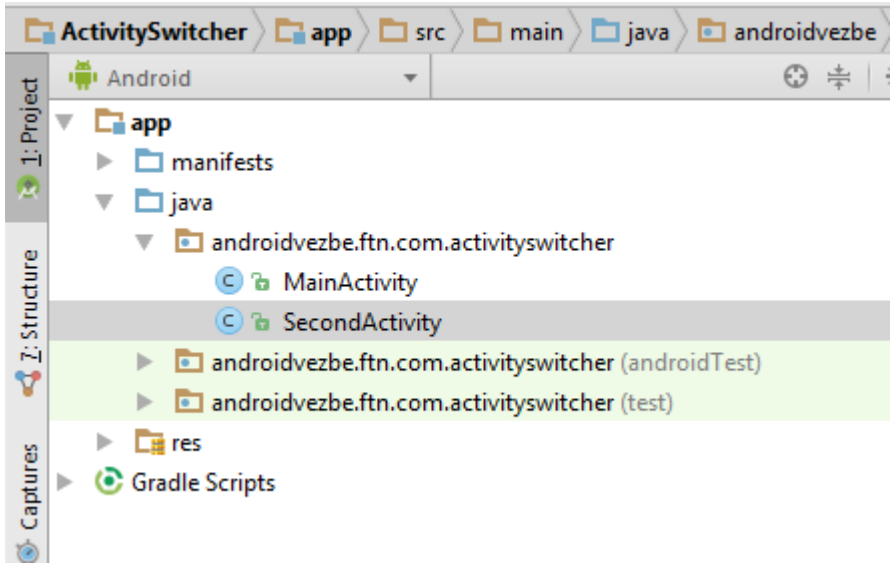
Eksplicitna namera

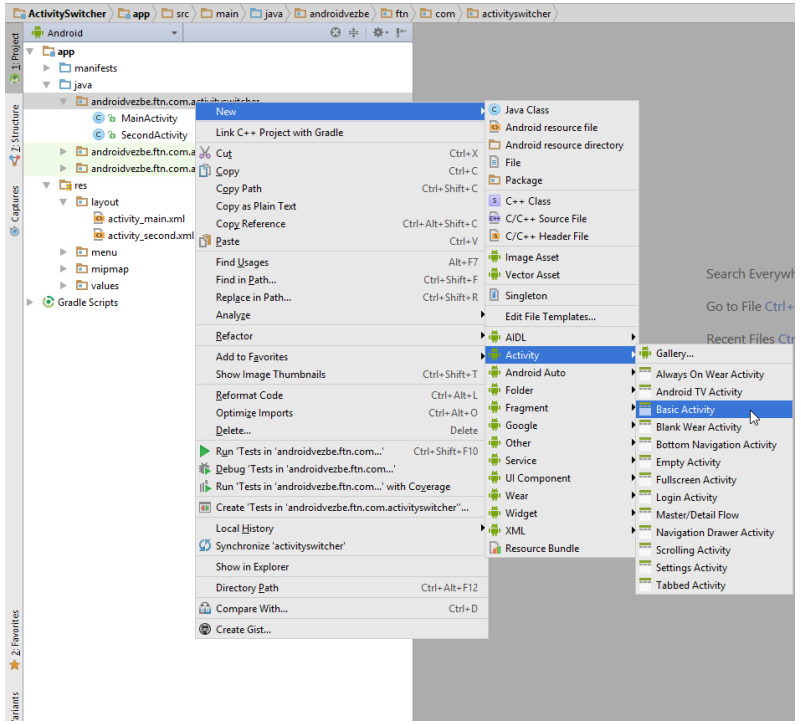
Sluzi za pokretanje specificirane komponente aplikacije, aktivnosti ili servisa u okviru aplikacije npr.

Activity

- Primer ActivitySwitcher: pokretanje druge aktivnosti iz prve

```
public void onClickSwitchActivity(View view) {
    Intent intent = new Intent(this, SecondActivity.class);
    startActivity(intent);
}
```





Activity – pokretanje aktivnosti

Implicitna namera

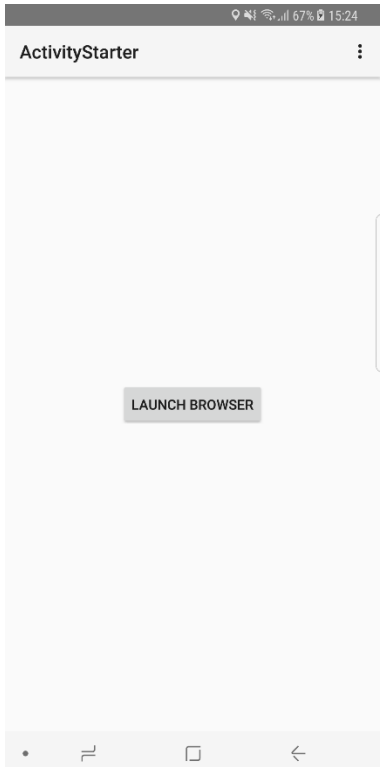
- Specificira akciju koja poziva bilo koju aplikaciju na uredjaju koja je u mogucnosti da obavi odredjenu akciju.

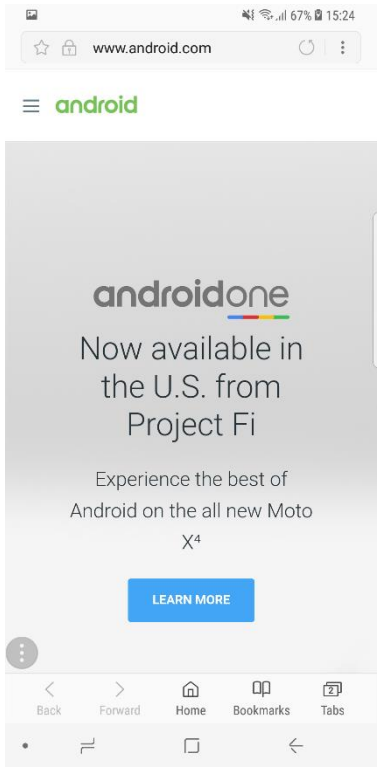
Izvor: https://developer.android.com/guide/components/intents-filters.html

Activity

- Primer ActivityStarter: pokretanja Browser ugradjene aplikacije
- ACTION_VIEW zajedno sa URL-om ukazuje da je namera prikaz web sajtam pa je iz tog razloga pokrenut standardni pretrazivac

```
public void launchIntent(View view) {
    Log.i("MainActivity","launchIntent()");
    Intent intent = new Intent(Intent.ACTION_VIEW);
    intent.setData(Uri.parse("https://www.android.com/"));
    startActivity(intent);
}
```



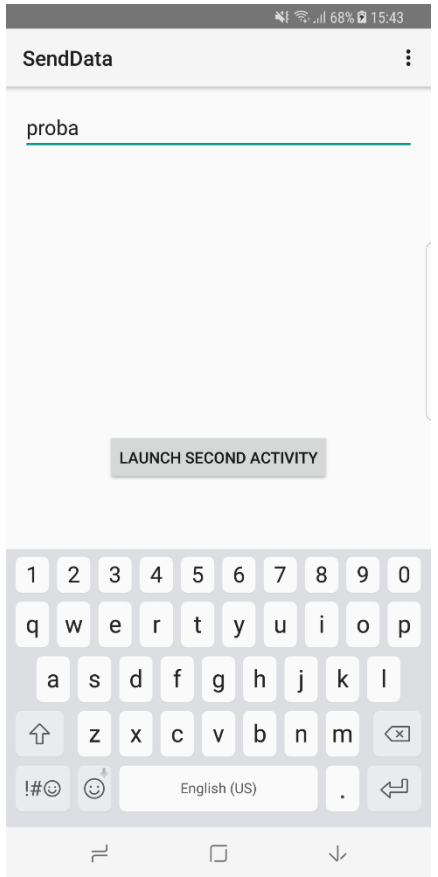


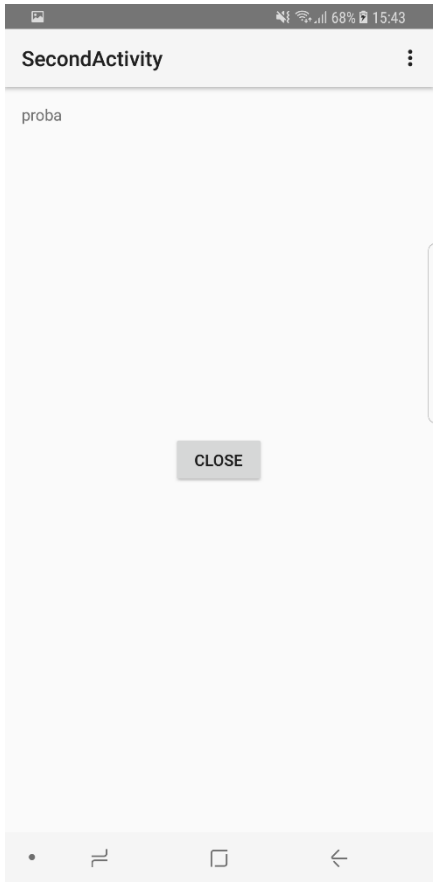
Activity

- Primer SendData: prosledjivanje podataka drugoj aktivnosti

```
public void onClickSwitchActivity(View view) {
    EditText editText = (EditText)findViewById(R.id.editTextData);
    String text = editText.getText().toString();
    Intent intent = new Intent(this, SecondActivity.class);
    intent.putExtra(Intent.EXTRA_TEXT,text);
    startActivity(intent);
}
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second);
    TextView textView = (TextView)findViewById(R.id.textViewText);
    if (getIntent()!=null && getIntent().hasExtra(Intent.EXTRA_TEXT)) {
        textView.setText(getIntent().getStringExtra(Intent.EXTRA_TEXT));
    }
}
```



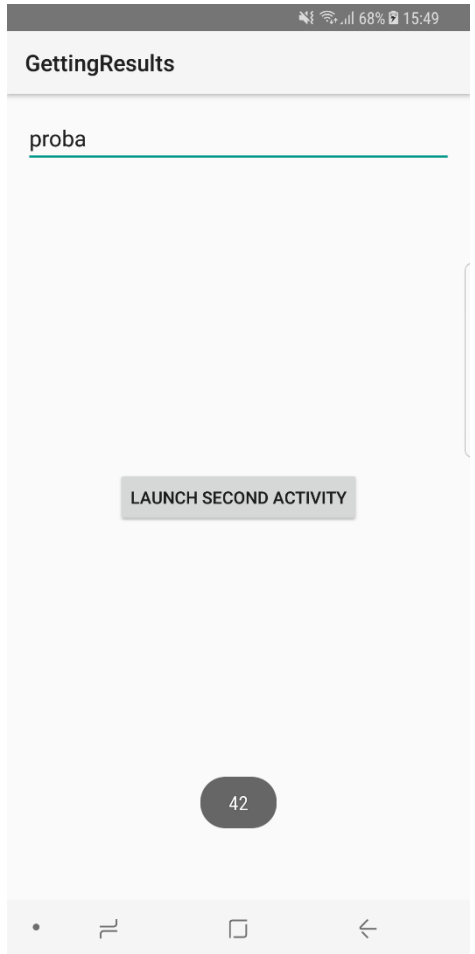


Activity

- Primer GettingResults: vracanje rezultata iz aktivnosti

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode==RESULT_OK) {
        Toast.makeText(this, Integer.toString(data.getIntExtra(REQUEST_RESULT, 0)), Toast.LENGTH_LONG).show();
    }
}

public void onClickSwitchActivity(View view) {
    EditText editText = (EditText)findViewById(R.id.editTextData);
    String text = editText.getText().toString();
    Intent intent = new Intent(this, SecondActivity.class);
    intent.putExtra(Intent.EXTRA_TEXT,text);
    startActivityForResult(intent,1);
}
```

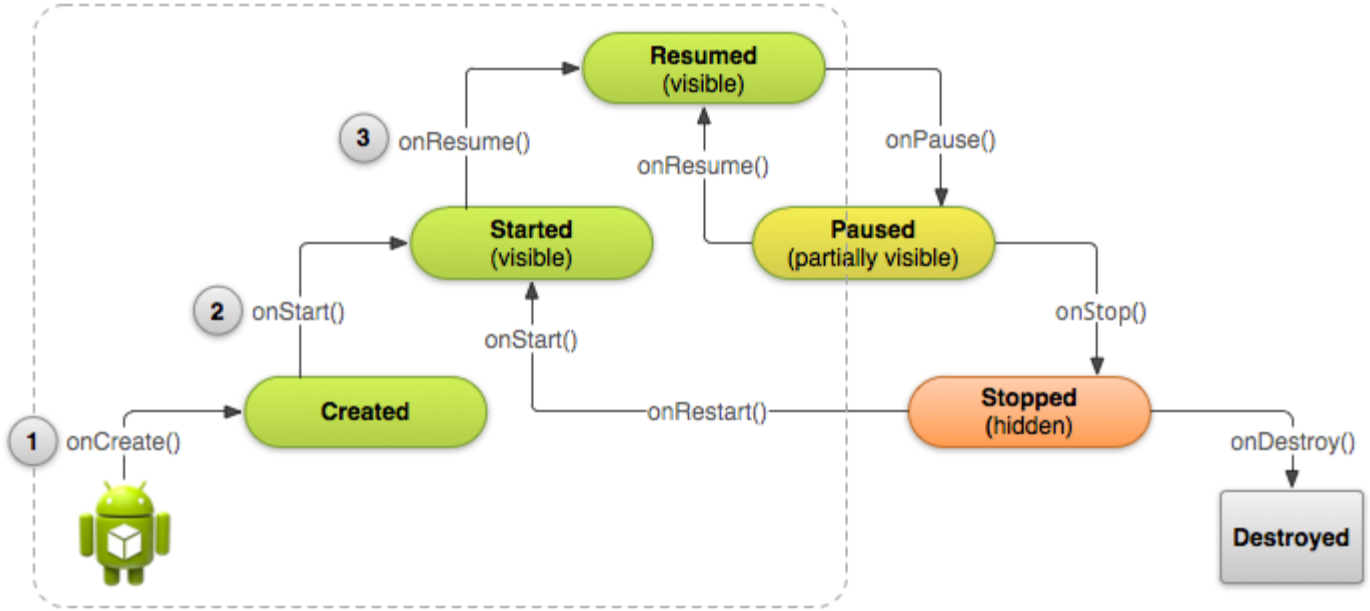
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second);
    TextView textView = (TextView)findViewById(R.id.textViewText);
    textView.setText(getIntent().getStringExtra(Intent.EXTRA_TEXT));
}

public void onClickClose(View view) {
    Intent returnIntent = new Intent();
    returnIntent.putExtra(MainActivity.REQUEST_RESULT,42);
    setResult(RESULT_OK, returnIntent);
    finish();
}
```



Activity – Snimanje stanja aktivnosti

- Kada se aktivnost pauzira ili zaustavi, njeno stanje je sacuvano u memoriji.
- Medjutim, da bi se sacuvalo stanje aktivnosti ako se ona unisti, potrebno je implementirati dodatnu metodu.
- Oprez: Android moze u bilo kom trenutku ubiti aktivnost koja se ne nalazi u prvom planu!

Activity – Snimanje stanja aktivnosti



Snimanje stanja aktivnosti

Activity – Snimanje stanja aktivnosti

- onSaveInstanceState

- Poziva se pre nego sto se aktivnost unisti da bi se snimilo njeno stanje koje se ponovo inicijalizuje u onCreate ili onRestoreInstanceState metodi.

```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {

  // Always call the superclass so it can save the view hierarchy state
  super.onSaveInstanceState(savedInstanceState);

}
```

Activity – Snimanje stanja aktivnosti

- onRestoreInstanceState
- Poziva se posle onStart metode da bi se aktivnost ponovo inicijalizovala iz prethodno snimljenog stanja.

```
public void onRestoreInstanceState(Bundle savedInstanceState) {
    // Always call the superclass so it can restore the view hierarchy
    super.onRestoreInstanceState(savedInstanceState);

}
```

Activity – Snimanje stanja aktivnosti

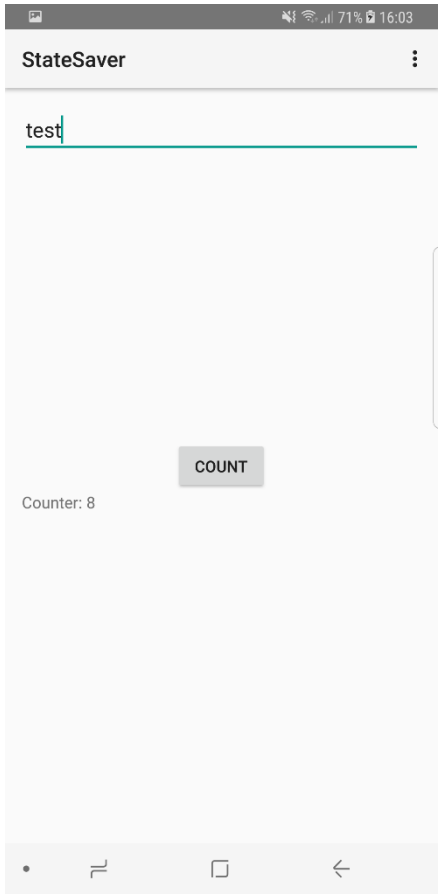
Klasa Bundle sadrzi metode oblika:

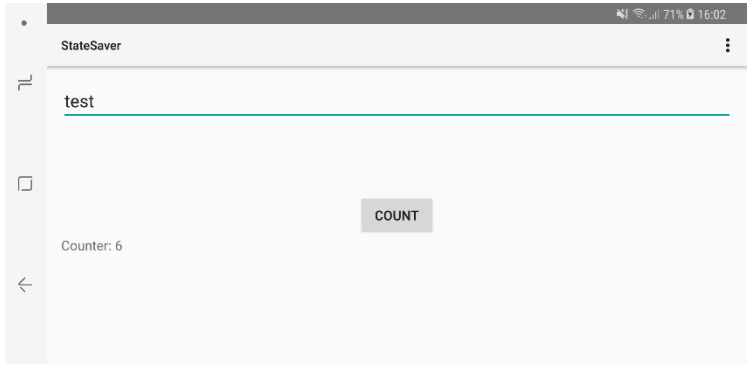
- T getT(String key);
- void putT(String key, T value);

- Podrazumevana implementacija pomenutih metoda poziva onSaveInstanceState metodu nad svakim elementom korisnickog interfejsa sto za rezultat ima cinjenicu da se stanje korisnickog interfejsta automatski snima.

Activity – Snimanje stanja aktivnosti

- Primer StateSaver: cuvanje stanja brojaca






```
static final String KEY_COUNTER = "COUNTER";
private int mCounter=0;

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

    if (savedInstanceState!=null) {
        mCounter = savedInstanceState.getInt(KEY_COUNTER);
    }
}
```

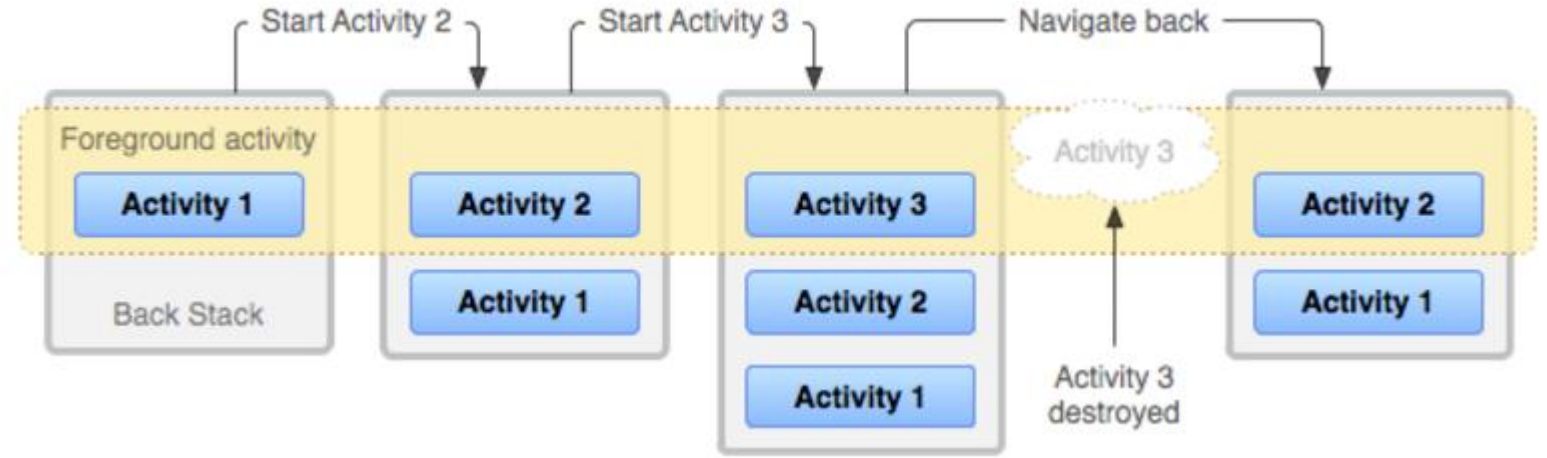
```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putInt(KEY_COUNTER,mCounter);
}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    mCounter=savedInstanceState.getInt(KEY_COUNTER);
}

public void onClickCounter(View view) {
    mCounter++;
    ((TextView) findViewById(R.id.textViewCounter)).setText("Counter: " + Integer.toString(mCounter));
}
```

Activity - Task

- Aplikacija se obicno sastoji iz vise aktivnosti.
- Zadatak (task) je skup aktivnosti sa kojima korisnik intereaguje da bi izvrsio odredjen posao.

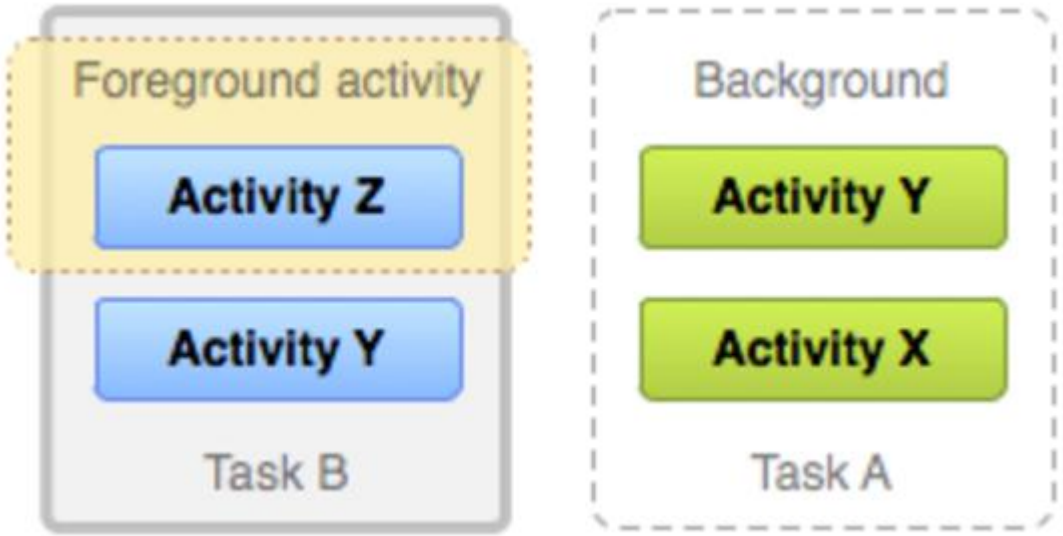
Activity - Povratni stek

- Aktivnosti su uredjene u povratni stek (back stack) u redosledu u kome su startovane.
- Kada se aktivnost startuje, stavlja se na vrh steka i dobija fokus.
- Pritiskom na Back dugme, tekuca aktivnost se skida sa vrha steka i unistava.



Activity – Upravljanje zadacima

- Svakom zadatku odgovara jedan povratni stek.
- Samo jedan zadatak se moze nalaziti u prvom planu u datom trenutku.



Activity – Upravljanje zadacima

- Podrazumevani nacin na koji Android upravlja zadacima i povratnim stekom se moze promeniti:
  - u deklaraciji aktivnosti u AndroidManifest.xml ili
  - postavljanjem odgovarajucih oznaka prilikom startovanja aktivnosti.

Activity – Upravljanje zadacima

- U AndroidManifest.xml moze se dodati launchMode atribut sa vrednostima:
  - standard (sistem startuje novu instancu aktivnosti u tekucem zadatku, tj. zadatku iz koga je startovana).
  - singleTop (ako se instanca aktivnosti vec nalazi na vrhu tekuceg zadatka, sistem joj prosledjuje nameru; u suprotnom startuje novu instancu u tekucem zadatku).
  - singleTask (ako se instanca aktivnosti vec nalazi u nekom zadatku, sistem joj prosledjuje nameru; u suprotnom startuje novu instancu u novom zadatku).
  - singleInstance (isto kao singleTask, osim sto sistem ne startuje druge aktivnosti u zadatku koji sadrzi ovu instancu).

Activity – Upravljanje zadacima

- Prilikom startovanja aktivnosti mogu se postaviti sledece oznake:
  - SINGLE_TOP (isto ponasanje kao singleTop).
  - NEW_TASK (isto ponasanje kao singleTask).
  - CLEAR_TOP (ako se instanca aktivnost vec nalazi u tekucem zadatku, sistem unistava sve aktivnosti koje se nalaze iznad nje i prosledjuje joj nameru; u suprotnom startuje novu instancu u tekucem zadatku).