GUI II

Mobilne aplikacije

Stevan Gostojić

Fakultet tehničkih nauka, Novi Sad

22. oktobar 2024.

Pregled sadržaja

Adapteri

- Adapteri povezuju poglede (naslednice `AdapterView` pogleda) i izvore podataka
- Postoje predefinisani adapteri koji povezuju različite poglede (ListView, GridView, Spinner, itd.) i različite izvore podataka (nizove, kolekcije, kursore, itd.)
- Moguće je napraviti adaptere koji povezuju proizvoljan pogled i proizvoljni izvor podataka

ArrayAdapter

- Povezuje `TextView` pogled (ili pogled koji sadrži `TextView` pogled) i niz ili kolekciju
- Automatski se poziva `toString()` metoda svakog objekta u nizu ili kolekciji i njena povratna vrednost se prikazuje u pogledu

activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:orientation="vertical"
9     tools:context=".MainActivity">
10
11    <ListView
12       android:id="@+id/listView"
13       android:layout_width="match_parent"
14       android:layout_height="wrap_content"
15       android:layout_weight="1" />
16
17 </LinearLayout>
18
```

list_item.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <TextView xmlns:android="http://schemas.android.com/apk/res/android"
3     android:id="@+id/textView"
4     android:layout_width="wrap_content"
5     android:layout_height="wrap_content"
6     android:layout_weight="1"
7     android:text="TextView">
8 </TextView>
9
```

MainActivity.java

```
1  public class MainActivity extends AppCompatActivity {
2
3    @Override
4    protected void onCreate(Bundle savedInstanceState) {
5      super.onCreate(savedInstanceState);
6      setContentView(R.layout.activity_main);
7
8      String[] days = {"Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"};
9      ArrayAdapter arrayAdapter = new ArrayAdapter(this,R.layout.list_item,days);
10     ListView listView = findViewById(R.id.listView);
11     listView.setAdapter(arrayAdapter);
12   }
13 }
14
```

CursorAdapter

- `CursorAdapter` je adapter koji koristi kursor kao izvor podataka
- Kursor sadrži rezultat upita nad bazom podataka (više o kurzorima na jednom od narednih časova)

Custom Adapters

- Moguće je definisati adapter koji koristi proizvoljan izvor podataka
- Potrebno je definisati klasu koja nasleđuje `Adapter` ili `BaseAdapter` i redefinisati njene metode

ExampleAdapter.java

```
1  public class ExampleAdapter extends BaseAdapter {
2
3    Activity activity;
4
5    public ExampleAdapter(Activity activity) {
6      this.activity = activity;
7    }
8
9    @Override
10   public int getCount() {
11     // return item count
12   }
13
14   @Override
15   public Object getItem(int position) {
16     // return item at position
17   }
18
19   @Override
20   public long getItemId(int position) {
21     // return item ID at position
22   }
23
24   @Override
25   public View getView(int position, View convertView, ViewGroup parent)
       {
26     // return view at position
27   }
28 }
29
```

ExampleAdapter.java

```
1  @Override
2  public View getView(int position, View view, ViewGroup parent) {
3
4    if (view == null) {
5      view = activity.getLayoutInflater().inflate(R.layout.example_adapter, null);
6    }
7
8    TextView tvName = (TextView) view.findViewById(R.id.tv_name);
9    tvName.setText(...);
10   TextView tvDescription = (TextView) view.findViewById(R.id.tv_description);
11   tvDescription.setText(..);
12   ImageView ivIcon = (ImageView) view.findViewById(R.id.iv_icon);
13   ivIcon.setImageResource(...);
14
15   return view;
16 }
17
```

example_adapter.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent" >
6
7      <ImageView android:id="@+id/iv_icon" />
8
9      <TextView android:id="@+id/tv_name" />
10
11     <TextView android:id="@+id/tv_description" />
12
13 </LinearLayout>
14
15
```

Spinner

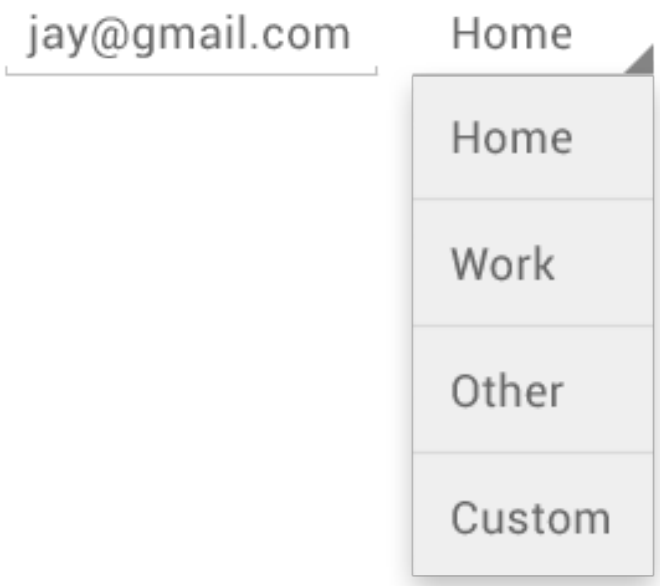


Figure 1: Spinner.

- Spinner pogled prikazuje stavke u meniju (korisnik može da izabere jednu stavku iz menija)
- Stavke se preuzimaju iz adaptera koji je pridružen pogledu

spinner.xml

```
1  <LinearLayout
2    xmlns:android="http://schemas.android.com/apk/res/android"
3    android:orientation="vertical"
4    android:layout_width="match_parent"
5    android:layout_height="match_parent">
6
7   <Spinner
8      android:id="@+id/spinner"
9      android:layout_width="match_parent"
10     android:layout_height="wrap_content" />
11
12 </LinearLayout>
13
```

SpinnerActivity.java

```
1  public class SpinnerActivity extends Activity {
2
3    @Override
4    public void onCreate(Bundle state) {
5      // ...
6      List<String> list = populate();
7      ArrayAdapter adapter =
8        new ArrayAdapter(this,android.R.layout.spinner_item, list);
9      Spinner spinner = (Spinner) findViewById(R.id.spinner);
10     spinner.setAdapter(adapter);
11
12     spinner.setOnItemSelectedListener(
13       new AdapterView.OnItemSelectedListener() {
14         public void onItemSelected(AdapterView<?> parent, View view,
       int position, long id) {
15
16           Intent intent =
17               new Intent(SpinnerActivity.this, SecondActivity.class);
18           intent.putExtra("position", position);
19           intent.putExtra("id", id);
20           startActivity(intent);
21
22         }
23         public void onNothingSelected(AdapterView<?> adapterView) {
24
25         }
26       });
27   }
28 }
29
```

ListView

Figure 2: ListView pogled.

- `ListView` pogled prikazuje listu stavki (koja može da se skroluje)
- Stavke se preuzimaju iz adaptera koji je pridružen pogledu

list_view.xml

```
1  <LinearLayout
2    xmlns:android="http://schemas.android.com/apk/res/android"
3    android:orientation="vertical"
4    android:layout_width="match_parent"
5    android:layout_height="match_parent">
6
7    <ListView
8      android:id="@+id/list_view"
9      android:layout_width="wrap_content"
10     android:layout_height="wrap_content" />
11
12 </LinearLayout>
13
```

ListViewActivity.java

```
1  public class ListViewActivity extends Activity {
2
3    @Override
4    protected void onCreate(Bundle state) {
5
6      // ...
7      List<String> list = populate();
8      ArrayAdapter adapter = new ArrayAdapter(
9        this, android.R.layout.simple_list_item_1, list);
10
11     ListView listView = (ListView) findViewById(R.id.list_view);
12     listView.setAdapter(adapter);
13   }
14 }
15
```

GridView

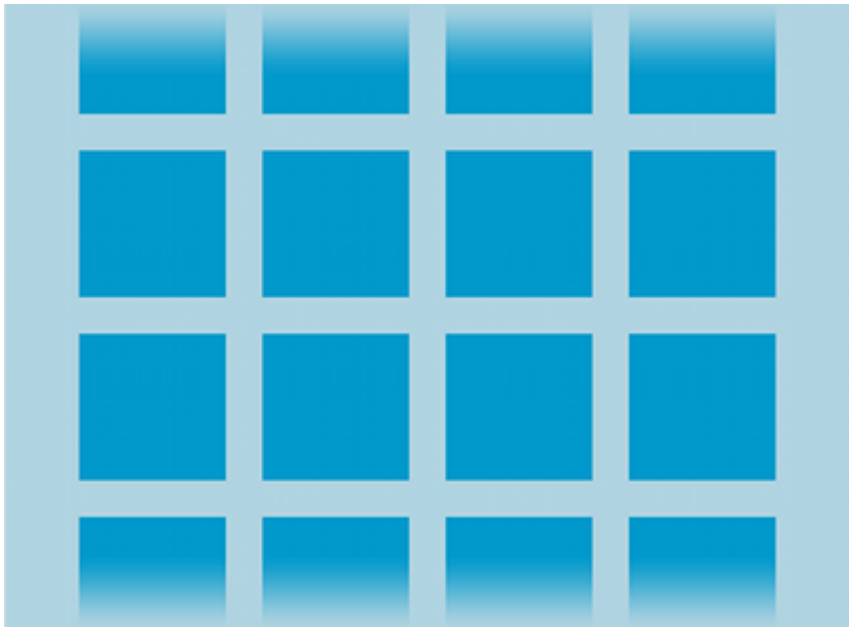


Figure 3: GridView pogled.

- `GridView` pogled prikazuje tabelu stavki (koja može da se skroluje)
- Stavke se preuzimaju iz adaptera koji je pridružen pogledu

grid_view.xml

```
1  <LinearLayout
2    xmlns:android="http://schemas.android.com/apk/res/android"
3    android:orientation="vertical"
4    android:layout_width="match_parent"
5    android:layout_height="match_parent">
6
7    <GridView
8      android:id="@+id/grid_view"
9      android:numColumns="auto_fit"
10     android:gravity="center"
11     android:columnWidth="50dp"
12     android:layout_width="match_parent"
13     android:layout_height="match_parent" />
14
15 </LinearLayout>
16
```

gridview_item.xml

```
1 <CheckedTextView
2    xmlns:android="http://schemas.android.com/apk/res/android"
3    android:id="@android:id/checked_text"
4    android:layout_width="wrap_content"
5    android:layout_height="match_parent"
6    android:layout_weight="0.9"
7    android:gravity="center_vertical" />
8
```

GridViewActivity.java

```
1  public class GridViewActivity extends Activity {
2
3    @Override
4    public void onCreate(Bundle state) {
5      // ...
6      List<String> list = populate() ;
7      ArrayAdapter adapter = new ArrayAdapter(
8        this, R.layout.gridview_item, list);
9
10     GridView gridView = (GridView) findViewById(R.id.grid_view);
11     gridView.setAdapter(adapter);
12   }
13 }
14
```

Pregled sadržaja

Toasts

Figure 4: Toast.

- Toast je pop-up poruka koja automatski nestaje posle određenog vremena
- Korisniku daje povratnu informaciju da je akcija izvršena
- Aktivnost na vrhu povratnog steka ostaje vidljiva i u fokusu
- U novijim verzijama Android platforme preporučljivo je koristiti Snackbar jer dozvoljava interakciju sa korisnikom

MainActivity.java

- Primer za Toast

```
1 Toast toast = Toast.makeText(
2                     getApplicationContext(),
3                     "Hello World!",
4                     Toast.LENGTH_SHORT);
5 toast.show();
6
```

Snackbar

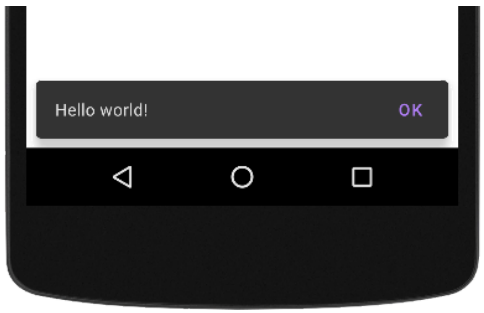


Figure 5: Snackbar.

- Snackbar može sadržati neku akciju (najviše jednu).
- Trajanje prikazivanja može biti `Snackbar.LENGTH_SHORT`, `Snackbar.LENGTH_LONG`, `Snackbar.LENGTH_INDEFINITE`.
- Snackbar se prikazuje preko korenskog rasporeda aktivnosti.
- Snackbar se uklanja protekom vremena, klikom na ponuđenu akciju ili pozivom `dismiss` metode.

MainActivity.java

- Primer za Snackbar

```
1  View parentView = findViewById(R.id.rootLayout);
2  Snackbar snackbar = Snackbar.make(
3                      parentView,
4                      "Hello world!",
5                      Snackbar.LENGTH_INDEFINITE);
6  snackbar.setAction("Undo", new View.OnClickListener() {
7                      @Override
8                      public void onClick(View view) {
9                          // do something
10                     }
11                 });
12 snackbar.show();
13
```

Pregled sadržaja

Obaveštenja

- Obaveštenje (notification) je poruka koja se prikazuje van korisničkog interfejsa aplikacije (u površini za obaveštenja ili fioci za obaveštenja)
- Ne prekida korisnika u izvršavanju tekućeg zadatka
- Obično se prikazuju obaveštenja o vremenski kritičnim događajima ili događajima u kojima učestvuju drugi ljudi
- Moguće je i izvršiti akciju iz obaveštenja

Obaveštenja

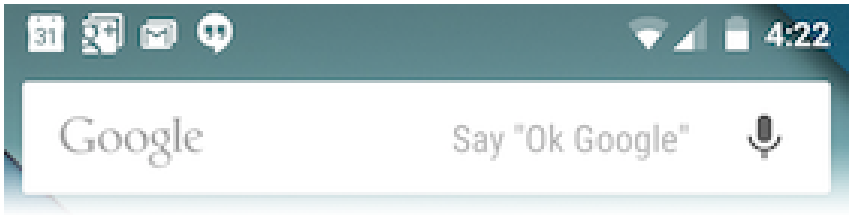


Figure 6: Površina za obaveštenja.

- Obaveštenje se prikazuje kao ikona u površini za obaveštenja (notofication area)

Obaveštenja

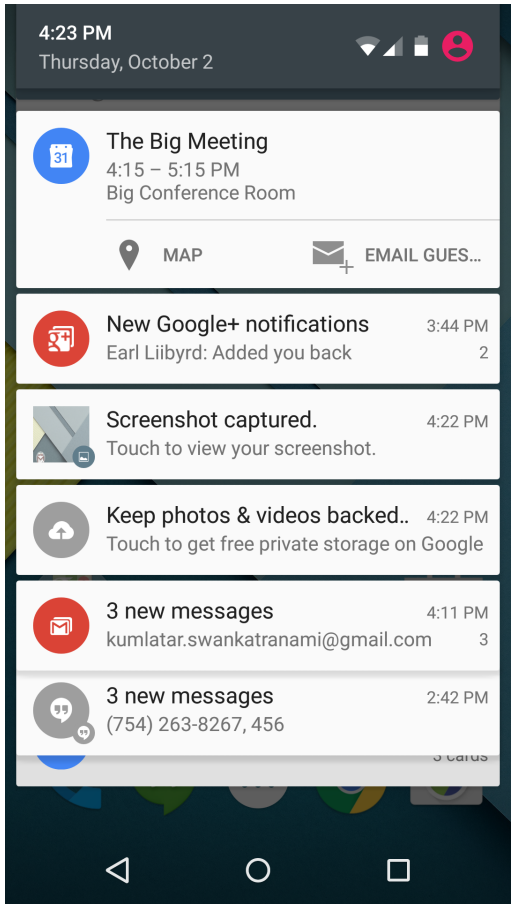
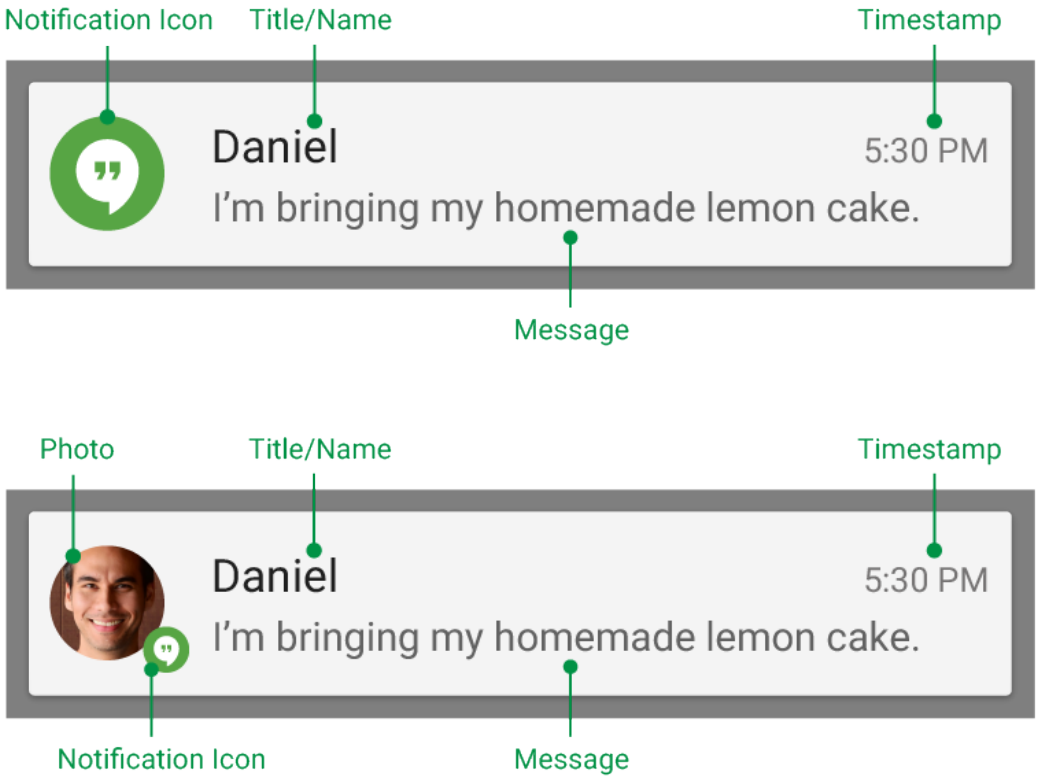


Figure 7: Fioka za obaveštenja.

- Više informacija o obaveštenju prikazuje se u fioci za obaveštenja (notification drawer)

Obaveštenja



- mala ikona
- naslov
- tekst

Figure 8: Osnovni raspored obaveštenja.

Obaveštenja - osnovni primer

- Od API level 26 notifikacije moraju biti pridružene nekom notifikacionom kanalu.
- Korisnik u podešavanjima može izmeniti ponašanje notifikacija koje pripadaju određenom notifikacionom kanalu.
- Pojedinačne notifikacije imaju svoj identifikator putem kojeg ih je moguće naknadno izmeniti.
- Od API level 33 za notifikacije je potrebna permisija `android.permission.POST_NOTIFICATIONS`.

```
1  String CH_ID = "my_notifications";
2  NotificationCompat.Builder builder =
3          new NotificationCompat.Builder(getApplicationContext(), CH_ID);
4  builder.setContentTitle("Hello")
5          .setContentText("Hello World!!!")
6          .setSmallIcon(R.drawable.ic_launcher_foreground);
7
8  int NOTIFICATION_ID = 1;
9  NotificationManager manager =
10   (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
11 manager.notify(NOTIFICATION_ID, builder.build());
12
```

- Android Studio poseduje alat Asset Studio koji olakšava kreiranje ikonica: File -> New -> Image Asset -> Icon Type: Notification Icons

Obaveštenja - dodatna podešavanja

```
1  String CH_ID = "my_notifications";
2  Context context = getApplicationContext();
3  NotificationCompat.Builder builder = new NotificationCompat.Builder(
       context, CH_ID)
4    .setContentTitle(title) // Mandatory property
5    .setLargeIcon(R.drawable.large_icon)
6    .setContentText(text) // Mandatory property
7    .setContentInfo(info)
8    .setSmallIcon(R.drawable.small_icon) // Mandatory property
9    .setWhen(when)
10   .setStyle(new Notification.BigPictureStyle().bigPicture(bigBitmap))
11   .setSound(soundURI)
12   .setLights(color, onDuration, offDuration)
13   .setVibrate(pattern)
14   .setPriority(priority);
15
```

Obaveštenja

Parametar	Opis
color	boja LED (RGB)
onDuration	interval u kome je LED uključena (ms)
offDuration	interval u kome je LED isključena (ms)

Table 1: Parametri setLights metode.

Obaveštenja

Parametar	Opis
pattern	interval u kome telefon vibrira (ms), interval u kome telefon ne vibrira (ms), itd.

Table 2: Parametri setVibrate metode.

- Za korišćenje vibracije je potrebno u Manifest.xml navesti permisiju

```
<uses-permission android:name="android.permission.VIBRATE" />
```

Obaveštenja

Vrednost	Opis
priority	MAX (critical and urgent), HIGH (high priority), DEFAULT (default), LOW (low in urgency), MIN (contextual or background)

Table 3: Parametri setPriority metode.

Obaveštenja - pridruživanje namere

- Ukoliko je potrebno, klikom na notifikaciju se može izvršiti neka akcija. Za to je potrebno da notifikaciji pridružimo odgovarajuću nameru.
- Pošto se ova namera pokreće izvan naše aplikacije, potrebno joj je dodeliti privilegije koje ima i aplikacija. To nam omogućava PendingIntent odn. namera na čekanju.

```
1  // First, we create an explicit intent
2  Intent intent = new Intent(this, ResultActivity.class);
3
4  // Then, we build a pending intent using the explicit intent
5  // requestCode differentiate our intent between other intents
6  int requestCode = (int) System.currentTimeMillis();
7  // this flag cancels an old intent and create new one
8  int flags = PendingIntent.FLAG_CANCEL_CURRENT;
9  PendingIntent pIntent = PendingIntent.getActivity(this, requestCode, intent, flags);
10
11 // Now, we assign the pending intent to the notification builder
12 builder.setContentIntent(pIntent);
13
14 // Finally, we retrieve notification manager and build notification
15 NotificationManager manager =
16   (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
17 manager.notify(id, builder.build());
18
```

Pregled sadržaja

Dijalozi

- Dijalog je modalni prozor koji prikazuje poruku i (opciono) omogućava unos podataka i potvrdu izvršavanja akcije
- Ne zauzima ceo ekran (aktivnost koja prikazuje dijalog se pauzira)
- Postoje predefinisani dijalozi kao što su: AlertDialog, DatePicker i TimePicker
- Moguće je definisati sopstvene dijaloge (preporučljivo je da se umesto klase `Dialog` koristi klasa `DialogFragment` zato što ona vodi računa o životnom ciklusu dijaloga i omogućava ponovno korišćenje dijaloga)

Dijalozi

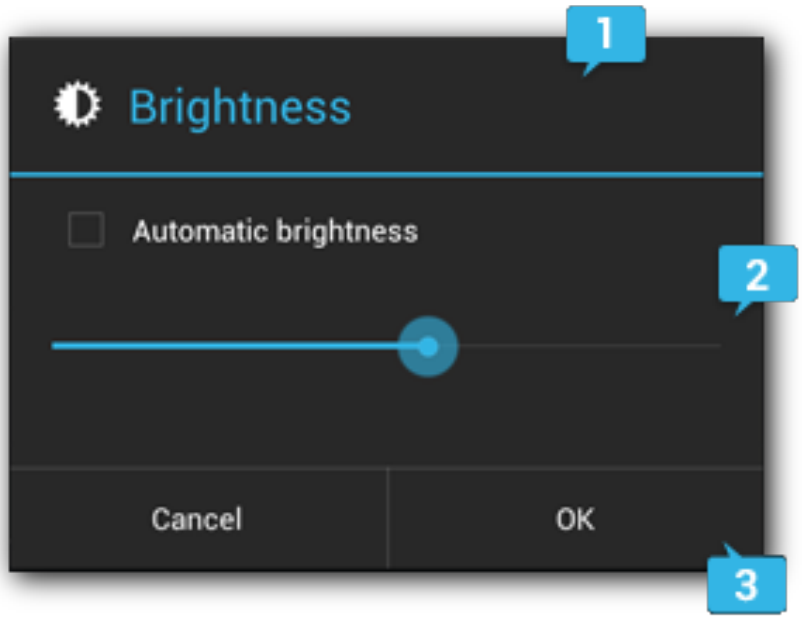


Figure 9: AlertDialog.

AlertDialog sadrži:

1. naslov
2. poruku, listu ili proizvoljan raspored
3. do tri dugmeta (negativno, neutralno i pozitivno)

ExampleActivity.java

```
1  AlertDialog dialog = new AlertDialog.Builder(ExampleActivity.this)
2    .setMessage(R.string.message_id)
3    .setPositiveButton(
4      R.string.btnOK,
5      new DialogInterface.OnClickListener() {
6        public void onClick(DialogInterface dialog, int id) {
7          // ...
8        }
9      })
10   .create();
11 dialog.show();
12
```

Dijalozi

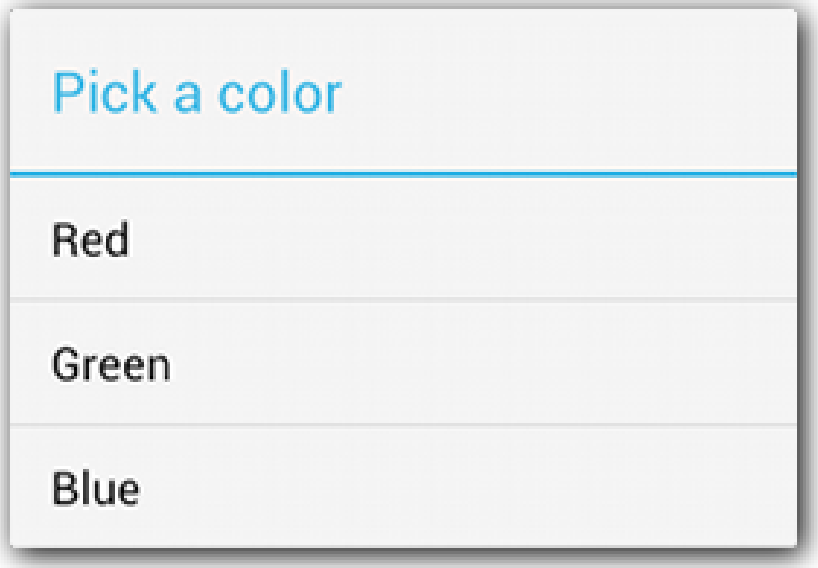


Figure 10: Dijalog sa listom.

- Alert dijalog može da sadrži:
  - listu
  - radio buttons
  - checkboxes
- Stavke se mogu definisati u statičkom nizu ili adapteru

Dijalog sa listom - ExampleDialogFragment.java

```
AlertDialog dialog = new AlertDialog.Builder(getActivity())
    .setTitle(R.string.pick_color)
    .setItems(
      R.array.colors_array,
      new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {
          // ...
        }
      })
    .create();
dialog.show();

```

- U projektu se nizovi mogu definisati kao resursi: File -> New -> Android Resource File -> Resource Type: Values

```
<resources>
    <string-array name="colors_array">
        <item>Red</item>
        <item>Green</item>
        <item>Blue</item>
    </string-array>
</resources>

```

Dijalozi sa rasporedom

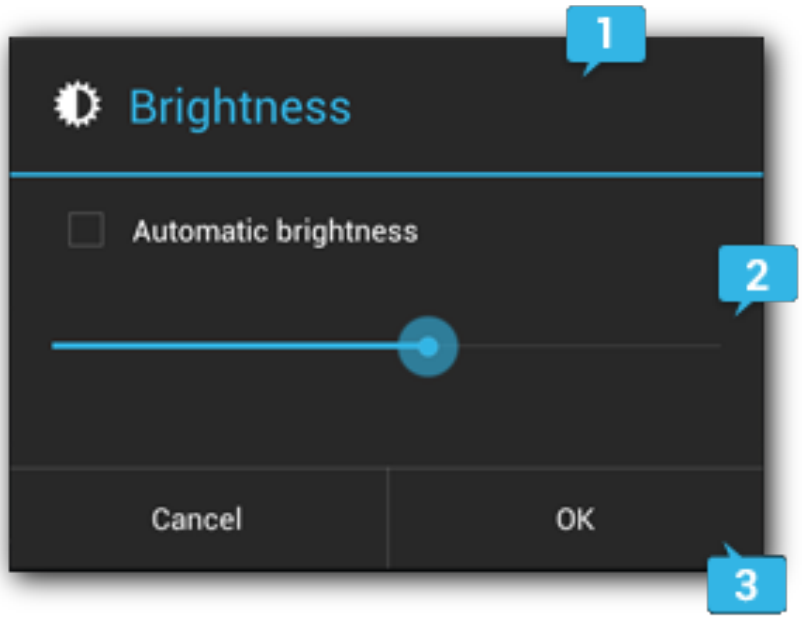


Figure 11: Dijalog sa rasporedom.

- Podrazumevano ponašanje je da raspored zauzme ceo dijalog.
- Ali je ipak moguće dodati i naslov i dugmad.

Dijalog sa rasporedom - ExampleDialogFragment.java

```
1  AlertDialog dialog = new AlertDialog.Builder(getActivity())
2    .setView(
3      getActivity().getLayoutInflater().inflate(R.layout.dialog_layout, null))
4    .setPositiveButton(
5      R.string.btnOK,
6      new DialogInterface.OnClickListener() {
7        @Override
8        public void onClick(DialogInterface dialog, int id) {
9          // ...
10       }
11     })
12   .create();
13 dialog.show();
14
```

TimePicker & DatePicker

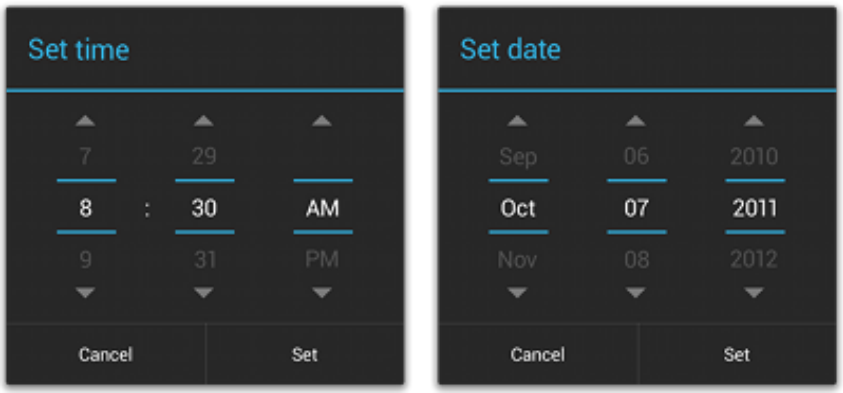


Figure 12: TimePicker & DatePicker.

1. Za unos vremena koristi se predefinisani dijalog TimePicker
2. Za unos datuma koristi se predefinisani dijalog DatePicker

DatePicker

```
1  public class DatePickerFragment
2    extends DialogFragment
3    implements DatePickerDialog.OnDateSetListener {
4
5    @Override
6    public Dialog onCreateDialog(Bundle state) {
7      Calendar c = Calendar.getInstance();
8      int year = c.get(Calendar.YEAR);
9      int month = c.get(Calendar.MONTH);
10     int day = c.get(Calendar.DAY_OF_MONTH);
11     return new DatePickerDialog(getActivity(), this, year, month, day);
12   }
13
14   @Override
15   public void onDateSet(DatePicker view, int year, int month, int day) {
16     // ...
17   }
18 }
19 // month value is zero-based (0,1,2,...)
20
```

TimePicker

```
1  public class TimePickerFragment
2    extends DialogFragment
3    implements TimePickerDialog.OnTimeSetListener {
4
5    @Override
6    public Dialog onCreateDialog(Bundle state) {
7      Calendar c = Calendar.getInstance();
8      int hourOfDay = c.get(Calendar.HOUR_OF_DAY);
9      int minute = c.get(Calendar.MINUTE);
10     return new TimePickerDialog(getActivity(), this, hourOfDay, minute, true);
11   }
12
13   @Override
14   public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
15     // ...
16   }
17 }
18
```

Prikazivanje dijaloga

```
1 public void showDialog() {
2   DialogFragment dialog = new DatePickerFragment();
3   // Tag name is used to save and restore fragment state
4   // and to get a handle to the fragment
5   dialog.show(getSupportFragmentManager(), "tag_name");
6 }
7
```

Pregled sadržaja

Podešavanja

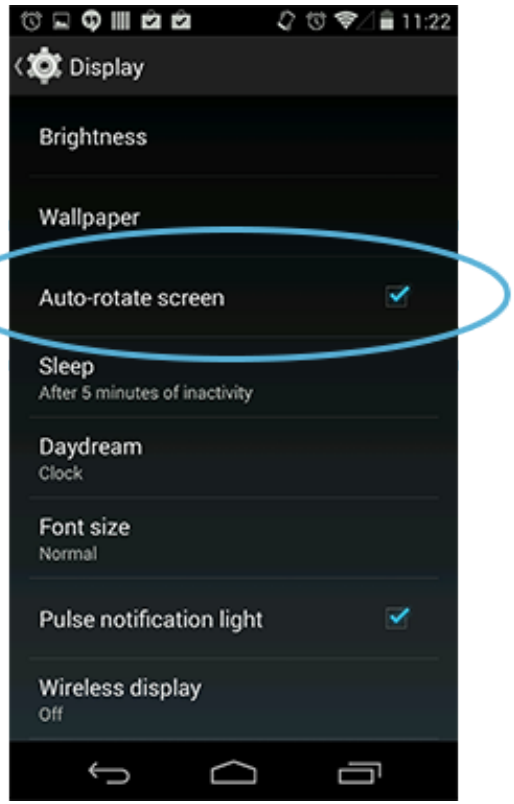


Figure 13: Podešavanja.

- Za podešavanje aplikacije koristi se Preferece API (da bi ponašanje aplikacija bilo konzistentno)
- Različitim tipovima parametrima odgovaraju različiti tipovi kontrola koje nasleđuju Preference klasu
- Kontrole se mogu grupisati u kategorije ili u podekrane
- Vrednosti parametara se automatski učitavaju i snimaju

Podešavanja

Tip parametra	Tip kontrole
Boolean	CheckBoxPreference, SwitchPreference
Float	EditTextPreference
Int	EditTextPreference
Long	EditTextPreference
String	EditTextPreference, ListPreference
Set<String>	MultiSelectListPreference

Table 4: Tipovi kontrola.

res/xml/preferences.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <PreferenceScreen
3     xmlns:android="http://schemas.android.com/apk/res/android">
4
5     <CheckBoxPreference
6        android:key="pref_sync"
7        android:title="@string/pref_sync"
8        android:summary="@string/pref_sync_summ"
9        android:defaultValue="true" />
10
11    <ListPreference
12       android:dependency="pref_sync"
13       android:key="pref_syncConnectionType"
14       android:title="@string/pref_syncConnectionType"
15       android:dialogTitle="@string/pref_syncConnectionType"
16       android:entries="@array/pref_syncConnectionTypes_entries"
17       android:entryValues="@array/
         pref_syncConnectionTypes_values"
18       android:defaultValue="@string/
         pref_syncConnectionTypes_default" />
19
20 </PreferenceScreen>
21
```

- Kreiranje foldera za XML resurse: File -> New -> Folder -> XML Resources Folder
- Dodavanje XML resursa: File -> New -> XML Resource File

Podešavanja

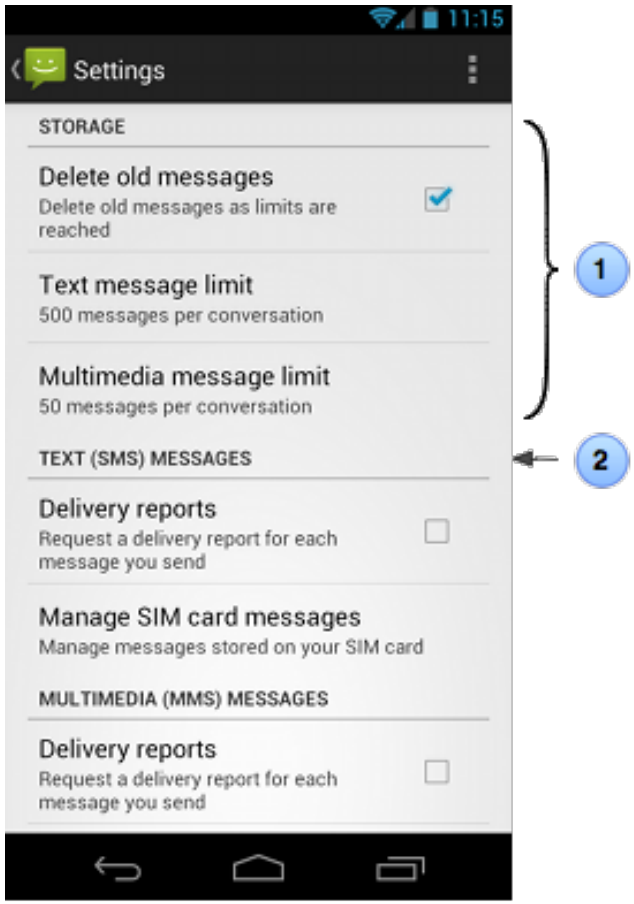


Figure 14: Podešavanja sa naslovima.

Grupisanje podešavanja u kategorije - preferences.xml (1/2)

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <PreferenceScreen
3    xmlns:android="http://schemas.android.com/apk/res/android">
4
5    <PreferenceCategory
6      android:title="@string/pref_sms_storage_title"
7      android:key="pref_key_storage_settings">
8
9      <CheckBoxPreference
10       android:key="pref_key_auto_delete"
11       android:summary="@string/pref_summary_auto_delete"
12       android:title="@string/pref_title_auto_delete"
13       android:defaultValue="false" />
14
```

Grupisanje podešavanja u kategorije - preferences.xml (2/2)

```
1       <EditTextPreference
2           android:key="pref_key_sms_delete_limit"
3           android:dependency="pref_key_auto_delete"
4           android:summary="@string/pref_summary_delete_limit"
5           android:title="@string/pref_title_sms_delete" />
6
7       <EditTextPreference
8           android:key="pref_key_mms_delete_limit"
9           android:dependency="pref_key_auto_delete"
10          android:summary="@string/pref_summary_delete_limit"
11          android:title="@string/pref_title_mms_delete" />
12
13    </PreferenceCategory>
14
15  </PreferenceScreen>
16
```

Kreiranje fragmenta sa prikazom podešavanja

```
1  public class SettingsFragment extends PreferenceFragmentCompat {
2
3    @Override
4    public void onCreatePreferences(Bundle state, String rootKey) {
5      // Load the preferences from an XML resource
6      addPreferencesFromResource(R.xml.preferences);
7    }
8
9    // ...
10
11 }
12
13 // since API level 29, preferences use dependency:
14 // implementation 'androidx.preference:preference:1.2.1'
15
16
```

Aktivnost za prikaz fragmenta sa podešavanjima

```
1  public class SettingsActivity extends AppCompatActivity {
2
3    @Override
4    protected void onCreate(Bundle state) {
5      super.onCreate(state);
6      // Display the fragment as the main content
7      getSupportFragmentManager()
8        .beginTransaction()
9        .replace(android.R.id.content, new SettingsFragment())
10       .commit();
11   }
12
13   // ...
14
15 }
16
```

Pregled sadržaja

Toolbar

Toolbar je element GUI-a koji se (obično) nalazi na vrhu ekrana i obezbeđuje:

- branding aplikacije
- navigaciju
- promenu pogleda
- izvršavanje akcija

Toolbar

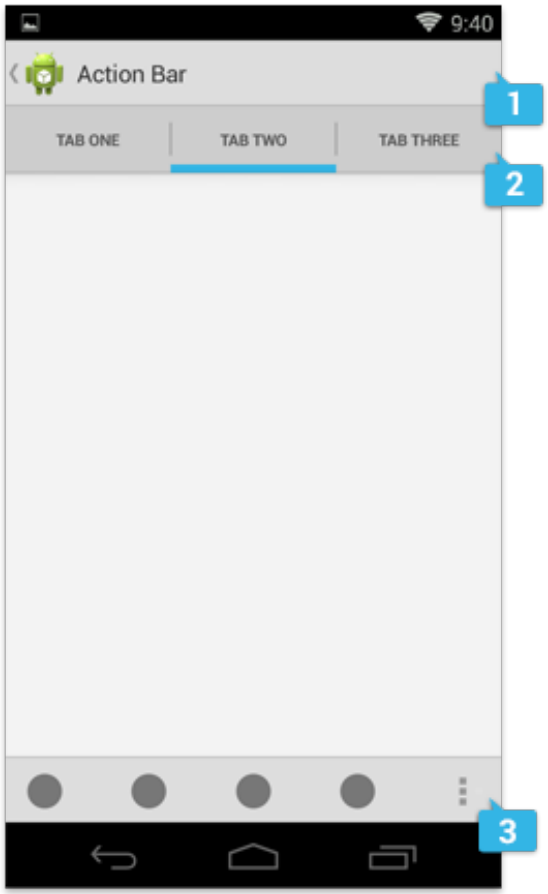


Figure 15: Toolbar.

1. title area (identifikuje aplikaciju)
2. navigation area (omogućava navigaciju)
3. action area (omogućava izvršavanje akcija, ređe korišćene akcije su "prelivene" u meni)

Toolbar

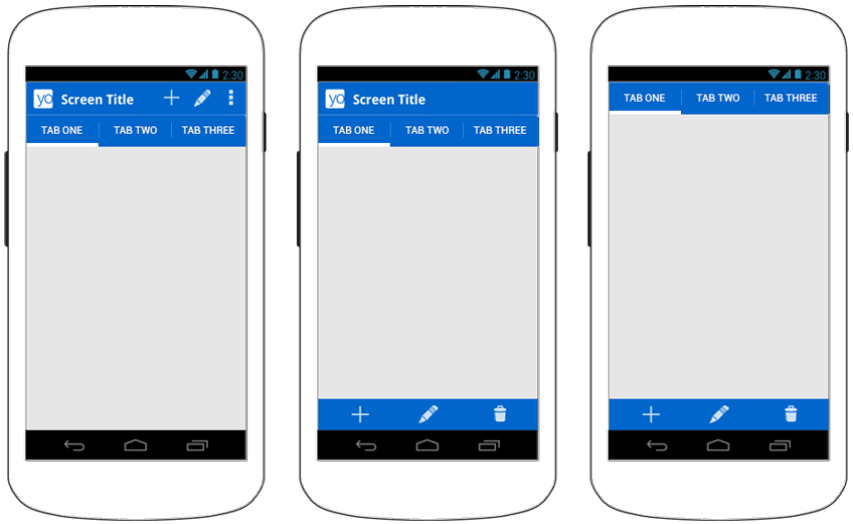


Figure 16: Toolbar.

Može se podeliti u više delova:

1. glavni deo (prikazuje ikonu i omogućava navigaciju)
2. gornji deo (omogućava promenu pogleda)
3. donji deo (omogućava izvršavanje akcija)

Pravljenje toolbar-a

- Dodati appcompat i Material Design zavisnost u projekat
- Dodati toolbar u raspored
- Definisati klasu koja nasleđuje `AppCompatActivity` klasu i u `onCreate()` metodi pozvati `setSupportActionBar()`
- Izabrati temu koja ne koristi ugrađeni `ActionBar`, na primer `AppCompat.NoActionBar`

build.gradle

```
1 dependencies {
2     implementation 'androidx.appcompat:appcompat:1.6.1'
3     implementation 'com.google.android.material:material:1.11.0'
4 }
5
```

layout_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res
       /android"
3    android:layout_width="fill_parent"
4    android:layout_height="fill_parent"
5    android:orientation="vertical">
6
7    <com.google.android.material.appbar.AppBarLayout
8                android:layout_width="match_parent"
9                android:layout_height="wrap_content"
10               android:theme="@style/ThemeOverlay.AppCompat.Dark">
11
12       <androidx.appcompat.widget.Toolbar
13                   android:id="@+id/toolbar"
14                   android:layout_width="match_parent"
15                   android:layout_height="?attr/actionBarSize"
16                   android:background="?attr/colorPrimary"
17                   app:layout_scrollFlags="scroll"/>
18
19   </com.google.android.material.appbar.AppBarLayout>
20
21 </LinearLayout>
22
```

ExampleActivity.java

```
1  public class ExampleActivity extends AppCompatActivity {
2
3    @Override
4    protected void onCreate(Bundle state) {
5      super.onCreate(state);
6      setContentView(R.layout.layout_main);
7      Toolbar toolbar = (Toolbar) findViewById(R.id.
       toolbar);
8      setSupportActionBar(toolbar);
9    }
10
11 }
12
```

Izbor teme koja ne koristi ActionBar

- Izbor teme u AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
  <application android:theme="@style/Theme.AppCompat.Light.NoActionBar">
    <!-- ... -->
  </application>
</manifest>
```

- Izbor teme u podrazumevanim definicijama stilova /res/values/themes.xml

```
<resources xmlns:tools="http://schemas.android.com/tools">
  <!-- Base application theme. -->
  <style name="Theme.MyApplication" parent="Theme.AppCompat.Light.NoActionBar">
    <!-- ... -->
  </style>
</resources>
```

ExampleActivity.java

```
1 ActionBar actionBar = getActionBar();
2 // Hides action bar
3 actionBar.hide();
4 // Shows action bar
5 actionBar.show();
6
```

Izvršavanje akcija

- Za implementaciju dugmadi se koristi mehanizam kontekstno zavisnog menija koji je nasleđen od starijih verzija Androida
- Deklarisati meni kao resurs
- Prikazati meni u `onCreateOptionsMenu` metodi i reagovati na akcije korisnika u `onOptionsItemSelected` metodi aktivnosti

action_bar.xml

```
1  <menu xmlns:android="http://schemas.android.com/apk/res/android">
2
3    <item
4       android:id="@+id/action_search"
5       android:icon="@drawable/ic_action_search"
6       android:title="@string/action_search"
7       android:showAsAction="ifRoom|withText|always" />
8
9    <item
10      android:id="@+id/action_compose"
11      android:icon="@drawable/ic_action_compose"
12      android:title="@string/action_compose" />
13
14 </menu>
15
```

- Android Studio olakšava definisanje menija kao resursa:
  res -> New -> Android Resource File -> Resource Type: Menu

ExampleActivity.java

```
1  public class ExampleActivity extends AppCompatActivity {
2
3    // Menu icons are inflated just as they were with
       actionbar
4    @Override
5    public boolean onCreateOptionsMenu(Menu menu) {
6      // Inflate the menu; this adds items to the action
       bar if it is present.
7      getMenuInflater().inflate(R.menu.menu_main, menu);
8      return true;
9    }
10
11   @Override
12   public boolean onOptionsItemSelected(MenuItem item) {
13     switch (item.getItemId()) {
14       case R.id.action_search:
15         openSearch();
16         return true;
17       case R.id.action_compose:
18         composeMessage();
19         return true;
20       default:
21         return super.onOptionsItemSelected(item);
22     }
23   }
24 }
25
```

Up dugme

- Up dugme se prikazuje u toolbar-u kao strelica na levo i služi za povratak na prethodnu aktivnost
- Za prikazivanje Up dugmeta je potrebno:
  - U `AndroidManifest.xml` povezati (child) aktivnost sa drugom (parent) aktivnošću
  - U (child) aktivnosti pozvati `setDisplayHomeAsUpEnabled` metodu i proslediti joj argument `true`

AndroidManifest.xml

```
1  <manifest ... >
2    <application ... >
3      <!-- ... -->
4      <!-- The main/home activity (has no parent activity) -->
5      <activity android:name="com.example.MainActivity">
6        <!-- ... -->
7      </activity>
8
9      <!-- A child of the main activity -->
10     <activity
11       android:name="com.example.ChildActivity"
12       android:parentActivityName="com.example.MainActivity">
13       <!-- ... -->
14     </activity>
15   </application>
16 </manifest>
17
```

ExampleActivity.java

```
1  public class ExampleActivity extends AppCompatActivity {
2
3    @Override
4    protected void onCreate(Bundle state) {
5      super.onCreate(state);
6      setContentView(R.layout.layout_main);
7      Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
8      setSupportActionBar(toolbar);
9
10     ActionBar actionbar = getSupportActionBar();
11     actionbar.setDisplayHomeAsUpEnabled(true);
12
13     // instead left arrow as up button, an icon can be shown
14     actionBar.setHomeAsUpIndicator(R.drawable.ic_menu);
15   }
16
17 }
18
```

Pregled sadržaja

NavigationDrawer

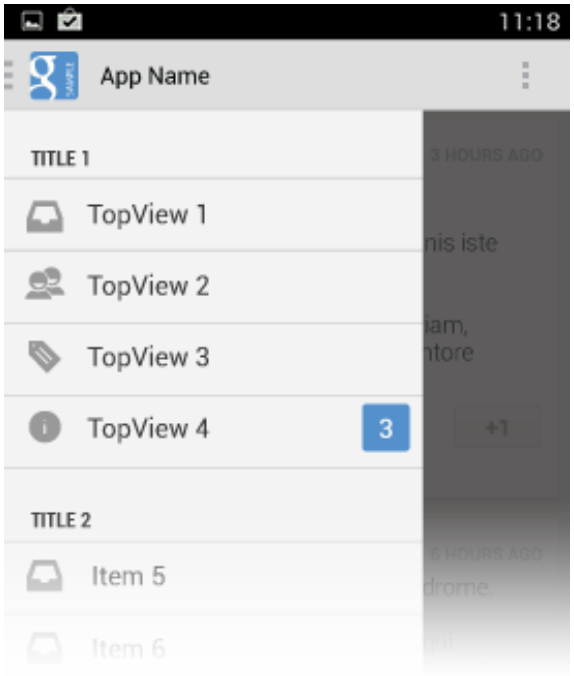


Figure 17: NavigationDrawer

- Deklarisati DrawerLayout raspored kao koreni raspored
- Dodati jedan pogled koji sadrži glavni sadržaj aktivnosti i drugi pogled (NavigationView) koji predstavlja sadržaj fioke za navigaciju
- Kreirati resurs (menu) sa stavkama fioke za navigaciju
- Reagovati na akcije korisnika

layout_main.xml

```
1  <androidx.drawerlayout.widget.DrawerLayout
2    xmlns:android="http://schemas.android.com/apk/res/android"
3    xmlns:app="http://schemas.android.com/apk/res-auto"
4    android:id="@+id/drawer_layout"
5    android:layout_width="match_parent"
6    android:layout_height="match_parent">
7
8    <!-- The main content view -->
9    <LinearLayout
10     android:id="@+id/content_frame"
11     android:layout_width="match_parent"
12     android:layout_height="match_parent" >
13
14   </LinearLayout>
15
16   <!-- The navigation drawer view -->
17   <com.google.android.material.navigation.NavigationView
18           android:id="@+id/navigation_view"
19           android:layout_width="wrap_content"
20           android:layout_height="match_parent"
21           android:layout_gravity="start"
22           app:headerLayout="@layout/drawer_header"
23           app:menu="@menu/drawer"/>
24
25 </androidx.drawerlayout.widget.DrawerLayout>
26
```

menu/drawer.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android">
3
4    <item
5       android:icon="@drawable/icon_01"
6       android:title="@string/menu_item01" />
7    <item
8       android:icon="@drawable/icon_02"
9       android:title="@string/menu_item02" />
10
11 </menu>
12
```

ExampleActivity.java

```
1  public class ExampleActivity extends AppCompatActivity {
2
3    DrawerLayout drawerLayout;
4
5    @Override
6    public void onCreate(Bundle state) {
7      super.onCreate(state);
8      setContentView(R.layout.layout_main);
9
10     drawerLayout = (DrawerLayout) findViewById(R.id.drawer_layout);
11     NavigationView navigationView = (NavigationView) findViewById(R.id.navigation_view);
12     navigationView.setNavigationItemSelectedListener(
13       new NavigationView.OnNavigationItemSelectedListener() {
14         @Override
15         public boolean onNavigationItemSelected(MenuItem menuItem) {
16           menuItem.setChecked(true);
17           drawerLayout.closeDrawers();
18           Toast.makeText(MainActivity.this, menuItem.getTitle(), Toast.LENGTH_LONG).show();
19           return true;
20         }
21     });
22   }
23 }
24
```