

# GUI I

## Mobilne aplikacije

# Pregled sadržaja

- 1 Resursi i konfiguracije uređaja
- 2 Grafički korisnički interfejs
- 3 Pogledi
- 4 Rasporedi

# Resursi i konfiguracije uređaja

- Android aplikacija je skup slabo povezanih komponenti
- Komponente pored klasa mogu da sadrže i resurse (deklaracije GUI, tekst, rastersku i vektorsku grafiku, audio i video klipove, itd.)
- Resurse treba eksternalizovati da bi se omogućilo prilagođavanje aplikacije različitim konfiguracijama uređaja (dimenzije, rezolucija i orijentacija ekrana, jezik i region, itd.) i lakša sinhronizacija između programera i grafičkih dizajnera.

# Tipovi resursa

Tip	Opis
anim	animacije
drawable	vektorska ili rasterska grafika
layout	deklaracije grafičkog korisničkog interfejsa
raw	"sirovi" podaci (audio i video klipovi)
values	proste vrednosti (nizovi, boje, eksternalizovani stringovi, stilovi, itd.)
xml	XML dokumenti

Table 1: Tipovi resursa.

# Eksternalizovanje stringova

- Primer hardcode-iranog stringa u layout-u

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical">
4
5   <TextView android:id="@+id/text" android:text="Hello World!"/ >
6
7
8 </LinearLayout>

```

---

- Primer eksternalizovanog stringa u layout-u

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical">
4
5   <TextView android:id="@+id/text" android:text="@string/hello_world"/ >
6
7
8 </LinearLayout>

```

- Navođenje stringa kao resursa u strings.xml

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <string name="hello_world">Hello World!</string>
4 </resources>

```

# Resursi

- Svaki resurs identifikovan je tipom i nazivom
- Android generiše jedinstveni identifikator svakog resursa (nalazi se u R klasi)
- Resursima se može pristupiti iz Java koda (`R.layout.main`, `R.string.hello_world`) ili iz XML koda (`@layout/main`, `@string/hello_world`)

# Konfiguracije uređaja

- Postoji veliki broj uređaja (sa različitim hardverskim karakteristikama) koji koriste Android platformu i veliki broj verzija Android platforme
- Resursi se mogu definisati za različite konfiguracije uređaja (npr. ekran niske, srednje i visoke rezolucije)
- Različitim konfiguracijama uređaja odgovaraju resursi koji se nalaze u direktorijumima sa različitim sufiksima (ldpi, mdpi, hdpi)
- Moguće je istovremeno definisati resurse za više tipova konfiguracija (ekran visoke rezolucije u noćnom modu)

# Konfiguracija uređaja

Tip	Vrednosti
language and region	en, fr, en-rUS, fr-rFR, fr-rCA, itd.
screen size	small, normal, large, xlarge
screen orientation	port, land
screen pixel density	ldpi, mdpi, hdpi, xhdpi, xxhdpi, xxxhdpi, nodpi, tvdpi
UI mode	car, desk, television, appliance, watch
touchscreen type	notouch, finger
night mode	night, notnight
platform version (API level)	v1, v2, v3, itd.

Table 2: Tipovi konfiguracije uređaja.



# Konfiguracija uređaja

```
ExampleApp/  
2   res/  
    drawable/  
4     icon.png  
    drawable-ldpi/  
6     icon.png  
    drawable-mdpi/  
8     icon.png  
    drawable-hdpi/  
10    icon.png  
    drawable-night-hdpi/  
12    icon.png
```

# Konfiguracija uređaja - preporuke

- Aplikacija bi uvek trebalo da sadrži podrazumevane resurse odn. resurse koji su nezavisni od konfiguracije
- Za različite rezolucije ekrana bi trebalo pripremiti slike različitih rezolucija
- Za različite veličine ili orijentacije ekrana bi trebalo pripremiti različite rasporede GUI-a
- Potrebno je i internacionalizovati stringove da bi se omogućila lokalizacija aplikacije na različite jezike

# Pregled sadržaja

- 1 Resursi i konfiguracije uređaja
- 2 Grafički korisnički interfejs
- 3 Pogledi
- 4 Rasporedi

## GUI

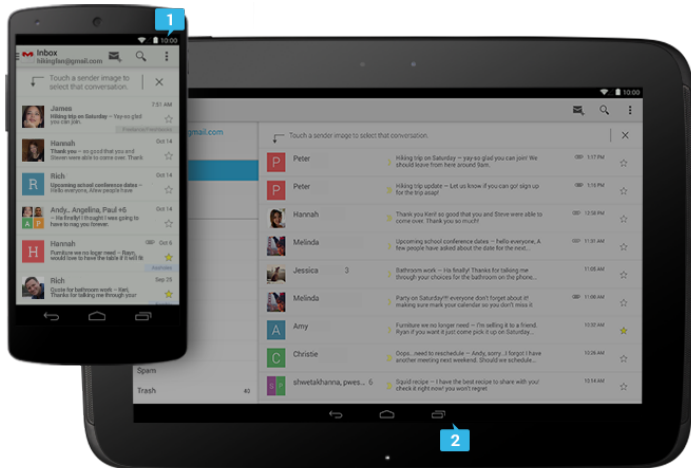


Figure 1: Statusna linija i navigaciona linija.

# GUI

- Home Screen
- All Apps
- Overview Space (Recents Screen)
- Notifications
- App Screen

# GUI

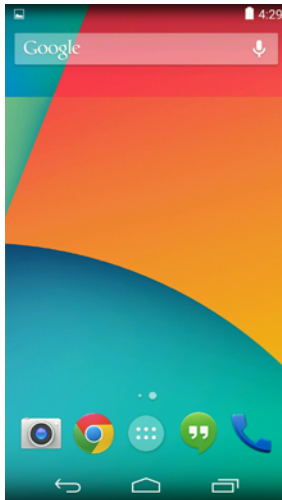


Figure 2: Home Screen.

## GUI

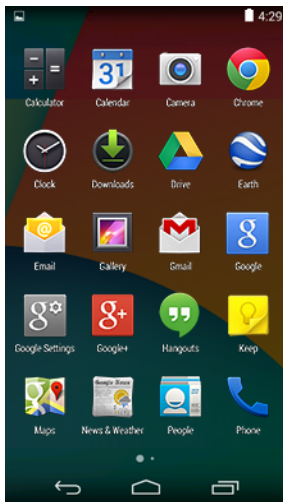


Figure 3: All Apps.

# GUI

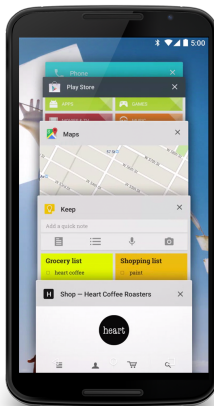


Figure 4: Overview Space



# GUI

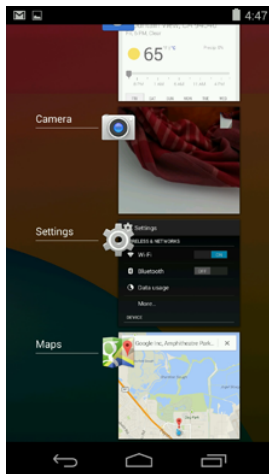


Figure 5: Recents Screen.

# GUI

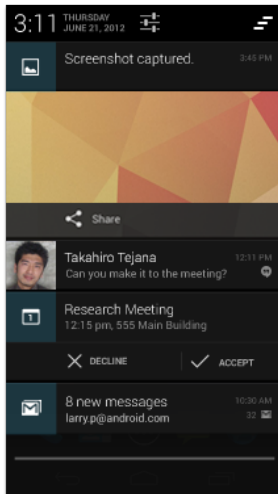


Figure 6: Obaveštenja.

# GUI

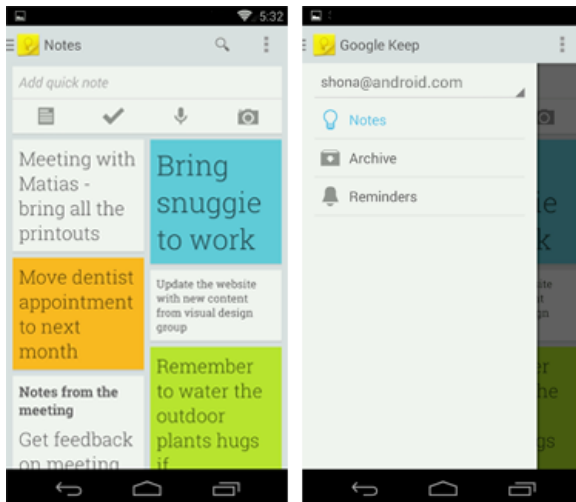


Figure 7: Toolbar, Navigation Drawer & Content Area.

# GUI



Figure 8: Toobar (App Bar).

- ① app icon
- ② view control
- ③ action buttons
- ④ action overflow

# GUI

## Gestovi

- Touch
- Long press
- Swipe/Drag
- Long press drag
- Double touch
- Double touch drag
- Pinch open
- Pinch close

# Navigacija

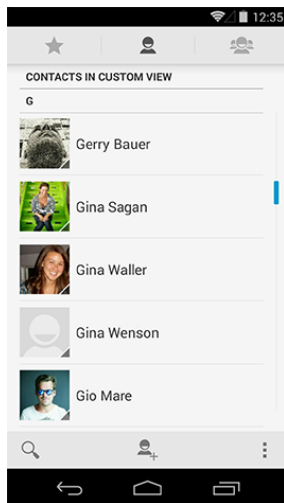


Figure 9: Kolekcija entiteta.

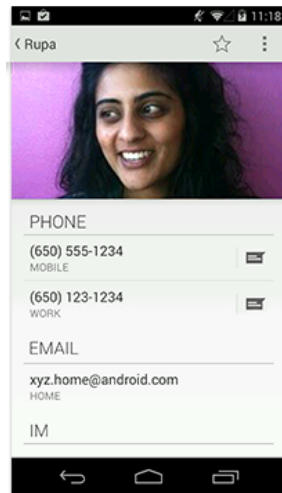


Figure 10: Pojedinačni entitet.

# Navigacija

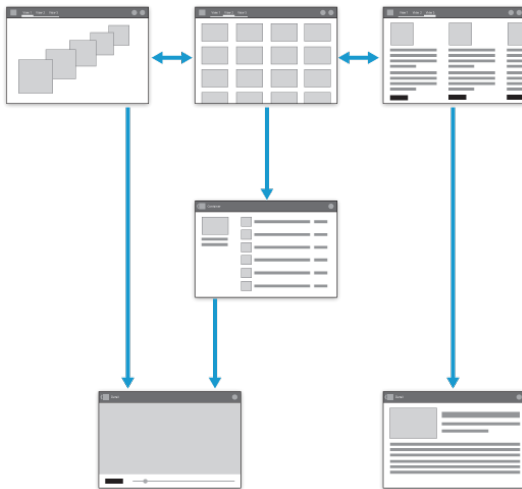


Figure 11: Struktura GUI-a.

# Pregled sadržaja

- 1 Resursi i konfiguracije uređaja
- 2 Grafički korisnički interfejs
- 3 Pogledi**
- 4 Rasporedi



# Pogledi i rasporedi

- Grafički korisnički interfejs bilo koje aktivnosti može se predstaviti hijerarhijom pogleda (view) i rasporeda (layout)
- Pogledi predstavljaju elemente GUI-a
- Rasporedi su pogledi koji sadrže druge poglede i određuju kako se oni raspoređuju na ekranu

# Pogledi i rasporedi

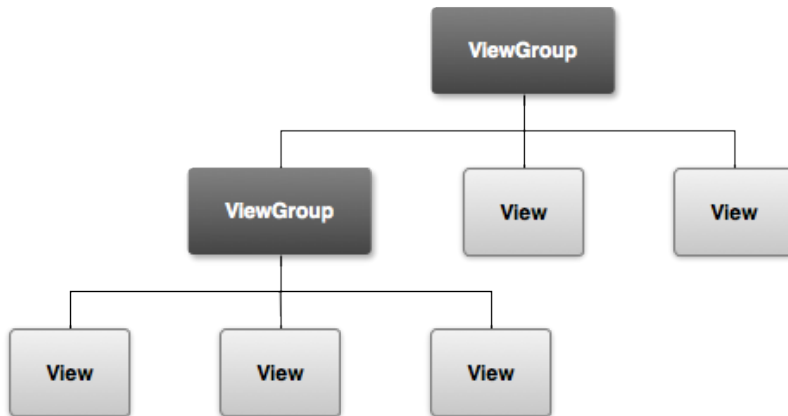


Figure 12: Hijerarhija pogleda i rasporeda GUI-a.

# Pogledi

- Elementi Android GUI-a su pogledi (View)
- Android sadrži predefinisane tipove pogleda (npr. labela, tekstualna polja, dugmad, itd.)
- Moguće je definisati nove tipove pogleda (mada se obično koriste predefinisani)

# Svojstva pogleda

- Stanje pogleda određeno je njegovim svojstvima (atributima)
- Postoje svojstva koja su zajednička za sve tipove pogleda (npr. vidljivost, transparentnost, itd.), a neki tipovi pogleda mogu da sadrže i posebna svojstva

# Događaji

- Pogledi mogu obrađivati različite događaje (dodir, pritisak tastera, promenu fokusa, itd.)

Događaj	Opis
onTouch()	the user performs an action qualified as a touch event (e.g. ACTION_DOWN, ACTION_MOVE, ACTION_UP)
onClick()	the user clicks the item
onLongClick()	the user either touches and holds the item
onFocusChange()	the user navigates onto or away from the item

Table 3: Važniji događaji.

# Pravljenje pogleda

- Pogledi se mogu definisati instanciranjem objekata u Java kodu ili dodavanjem elemenata u XML kodu
- Na sličan način mogu se postaviti svojstva i obrađivači događaja pogleda

# Pravljenje pogleda u Java kodu

## ExampleActivity.java

```
public class ExampleActivity extends Activity {  
2   protected void onCreate(Bundle state) {  
  
4       Button button = new Button(this);  
       LinearLayout layout = new LinearLayout(this);  
6       layout.addView(button);  
       setContentView(layout);  
8   }  
   }  
10 }
```

# Pravljenje pogleda u XML kodu

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/
  android"
  android:id="@+id/linear_layout"
4  android:orientation="vertical">

6  <Button android:id="@+id/button" />

8 </LinearLayout>
```

---

```
public class ExampleActivity extends Activity {
2  protected void onCreate(Bundle state) {

4      setContentView(R.layout.main);
  }
6 }
```



# Postavljanje svojstava u Java kodu

ExampleActivity.java

```
public class ExampleActivity extends Activity {  
2   protected void onCreate(Bundle state) {  
  
4       Button button = (Button) findViewById(R.id.button);  
       button.setLayoutParams(new LayoutParams(  
6           ViewGroup.LayoutParams.WRAP_CONTENT,  
           ViewGroup.LayoutParams.WRAP_CONTENT));  
7       button.setText(R.string.button_text);  
8  
10      LinearLayout layout = new LinearLayout(this);  
       layout.addView(button);  
12  
       setContentView(layout);  
14  }  
16  }
```

# Postavljanje svojstava

## layout.xml

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res /
    android"
        android:id="@+id/linear_layout"
4     android:orientation="vertical">

6     <Button
        android:id="@+id/button"
8         android:layout_width="wrap_content"
        android:layout_height="wrap_content"
10        android:text="@string/button_text" />

12 </LinearLayout>

```

---

```

    public class ExampleActivity extends Activity {
2        protected void onCreate(Bundle state) {

4            setContentView(R.layout.main);
        }
6    }

```

# Obrada događaja

## ExampleActivity.java

```
1 public class ExampleActivity extends Activity {
2     protected void onCreate(Bundle bundle) {
3
4         Button button = new Button(this);
5         button.setOnClickListener(new View.OnClickListener() {
6             public void onClick(View v) {
7                 Toast.makeText(ExampleActivity.this, "Button clicked", Toast.
8                     LENGTH_SHORT).show();
9             }
10        }
11
12        LinearLayout layout = new LinearLayout(this);
13        layout.addView(button);
14
15        setContentView(layout);
16    }
```

# Obrada događaja

## layout.xml

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res /
    android"
3      android:id="@+id/linear_layout"
4      android:orientation="vertical">
5
6      <Button
7          android:id="@+id/button"
8          android:layout_width="wrap_content"
9          android:layout_height="wrap_content"
10         android:text="@string/button_text"
11         android:onClick="click" />
12
13 </LinearLayout>
14

```

---

```

1  public class ExampleActivity extends Activity {
2
3      protected void onCreate(Bundle state) {
4          setContentView(R.layout.main);
5      }
6
7      public void click(View view) {
8          Toast.makeText(this, "Button clicked", Toast.LENGTH_SHORT).show();
9      }
10 }

```

# Tipovi pogleda

- TextView
- ImageView
- EditText
- Button
- RadioButton
- ToggleButton
- Checkbox

# Hello Android!

Figure 13: TextView.

- Pogled TextView prikazuje tekst i omogućava njegovo kopiranje

# TextView

```
<TextView
2   android:id="@+id/email_address"
   android:layout_width="match_parent"
4   android:layout_height="wrap_content"
   android:text="@string/hello_world" />
6
```

# ImageView



Figure 14: ImageView.

- Pogled ImageView prikazuje proizvoljnu sliku iz različitih izvora
- Omogućava i skaliranje, odsecanje, primenu filtera, itd.



# ImageView

```
<ImageView
2   android:id="@+id/icon"
   android:layout_width="wrap_content"
4   android:layout_height="wrap_content"
   android:src="@drawable/my_image" />
6
```

# EditText

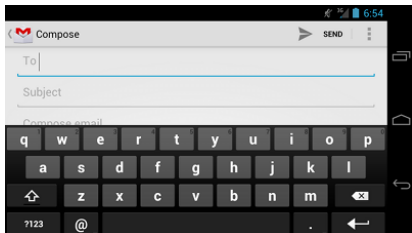


Figure 15: EditText.

- Pogled EditText omogućava unos teksta
- Pored unosa teksta, omogućava i niz drugih aktivnosti kao što su označavanje, isecanje, kopiranje, itd.
- Moguće je specificirati tip tastature (normalna, numerička, telefonska, itd.) ili ponašanje tastature (automatsko pretvaranje početnih slova reči u velika slova, itd.)

# EditText

```
<EditText
2   android:id="@+id/email_subject"
   android:layout_width="match_parent"
4   android:layout_height="wrap_content"
   android:hint="@string/subject"
6   android:inputType="text" />
```

# Button



Figure 16: Button.

- Pogled Button prikazuje tekst ili sliku koja simbolizuju akciju
- Kada korisnik pritisne dugme generiše se click događaj
- Metoda koja obrađuje ovaj događaj specificira se onClick atributom i mora biti sadržana u aktivnosti kojoj je dugme pridruženo

# Button

```

<Button
2   android:id="@+id/button_id"
   android:layout_width="wrap_content"
4   android:layout_height="wrap_content"
   android:text="@string/button_text"
6   android:drawableLeft="@drawable/button_icon"
   android:onClick="alarm" />
8

```

---

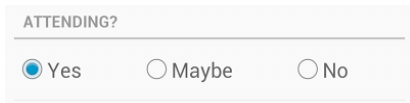
```

public void alarm(View view) {
2   Uri notification = RingtoneManager.getDefaultUri(
      RingtoneManager.TYPE_NOTIFICATION);
4
   Ringtone r = RingtoneManager.getRingtone(
6     getApplicationContext(), notification);

8   r.play();
   }
10

```

# RadioButton



ATTENDING?

☒ Yes    ☐ Maybe    ☐ No

Figure 17: RadioButton.

- Pogled `RadioButton` omogućava korisniku da izabere jednu opciju iz skupa više opcija
- Svaka opcija predstavljena je objektom klase `RadioButton` koji su grupisani objektom klase `RadioGroup`

# RadioButton i RadioGroup

```

<RadioGroup
2   android:layout_width="match_parent"
   android:layout_height="wrap_content"
4   android:orientation="vertical">

6   <RadioButton
       android:id="@+id/yes"
8       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
10      android:text="@string/yes"
       android:onClick="onRadioButtonClicked"/ >

12
   <RadioButton
       android:id="@+id/maybe"
14      android:layout_width="wrap_content"
       android:layout_height="wrap_content"
16      android:text="@string/maybe"
       android:onClick="onRadioButtonClicked"/ >

18
20   ...

22 </RadioGroup>

```

# RadioButton

```
public void onRadioButtonClicked(View view) {  
2   RadioButton rb = (RadioButton) view;  
   if (rb.isChecked()) {  
4       switch (rb.getId()) {  
           case R.id.yes:  
6               // ...  
               break;  
8           case R.id.maybe:  
               // ...  
10              break;  
           case R.id.no:  
12              // ...  
               break;  
14       }  
   }  
16 }
```



# ToggleButton



Figure 18: ToggleButton.

- Pogled ToggleButton omogućava korisniku da promeni podešavanje između dva stanja

# ToggleButton

```
<ToggleButton
2   android:id="@+id/togglebutton"
   android:layout_width="wrap_content"
4   android:layout_height="wrap_content"
   android:textOn="Vibrate on"
6   android:textOff="Vibrate off"
   android:onClick="onToggleClicked"/ >
8
```

---

```
public void onToggleClicked(View view) {
2   ToggleButton tb = (ToggleButton) view;
   if (tb.isChecked()) {
4       // Enable vibrate
   } else {
6       // Disable vibrate
   }
8 }
```

# CheckBox

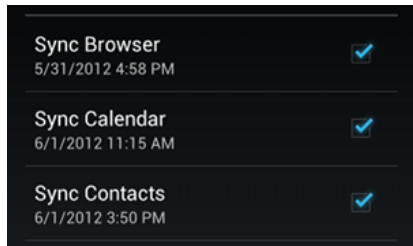


Figure 19: CheckBox.

- Pogled CheckBox omogućava korisniku da izabere jednu ili više opcija iz skupa opcija
- Opcije se obično prikazuju u vertikalnoj listi.

# CheckBox

```
<CheckBox
2   android:id="@+id/checkbox_browser"
   android:layout_width="wrap_content"
4   android:layout_height="wrap_content"
   android:text="@string/sync_browser"
6   android:onClick="onCheckboxClicked" />
```

---

```
public void onCheckboxClicked(View view) {
2   CheckBox cb = (CheckBox) view;
   if (cb.isChecked()) {
4       // ...
   }
6 }
```

# Pregled sadržaja

- 1 Resursi i konfiguracije uređaja
- 2 Grafički korisnički interfejs
- 3 Pogledi
- 4 Rasporedi**

# Raspored

- Raspored (layout) je pogled koji sadrži druge poglede i raspoređuje ih po ekranu
- Kao i svaki drugi pogled, može se definisati proceduralno u Java kodu (instanciranjem klase) ili deklarativno u XML kodu (dodavanjem elementa)

## main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/
  android"
  android:layout_width="match_parent"
4  android:layout_height="match_parent"
  android:orientation="vertical">
6
  <TextView
8    android:id="@+id/text"
    android:layout_width="wrap_content"
10   android:layout_height="wrap_content"
    android:text="Hello, I am a TextView" />
12
  <Button
14    android:id="@+id/button"
    android:layout_width="wrap_content"
16   android:layout_height="wrap_content"
    android:text="Hello, I am a Button" />
18 </LinearLayout>
```

# Svojstva pogleda

- Slično CSS box modelu, svaki pogled ima geometriju pravougaonika
- Poziciju i dimenzije pogleda određuje vrsta rasporeda koji ga sadrži i svojstva pogleda (koja mogu da zavise od vrste rasporeda)
- Neka svojstva (npr. padding i margin) ne zavise od vrste rasporeda
- Neka svojstva (npr. layout\_width i layout\_height) zavise od vrste rasporeda



# Merne jedinice

Oznaka	Naziv	Opis
dp	density-independent pixels	apstraktna merna jedinica koja odgovara veličini piksela na ekranu rezolucije 160 dpi
sp	scale-independent pixels	kao i dp, samo što se piksel skalira i u odnosu na faktor skaliranja fonta
pt	tačka	1/72 in
px	piksel	fizička veličina piksela na ekranu
mm	milimetar	
in	inč	

Table 4: Merne jedinice.

# Svojstva pogleda

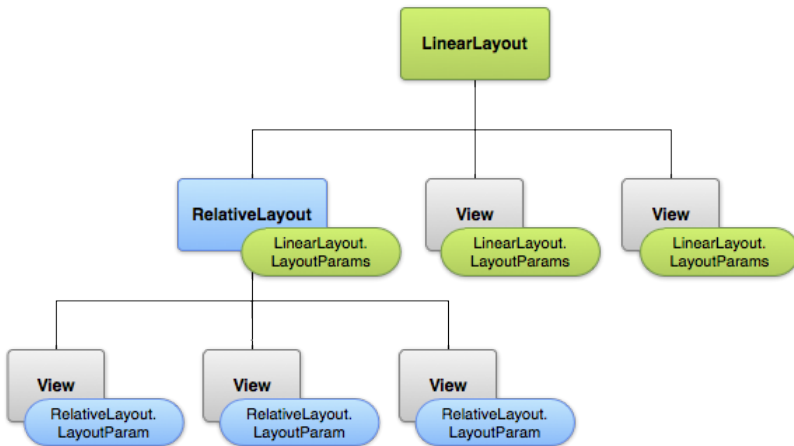


Figure 20: Svojstva pogleda.

# Iscrtavanje pogleda

- Svaki pogled iscrtava sebe i svoju decu
- Iscrtavanje pogleda izvršava se u dva prolaza:
  - prolazu merenja (measure pass)
  - prolazu raspoređivanja (layout pass)

# Vrste rasporeda

- `AbsoluteLayout`
- `GridLayout`
- `FrameLayout`
- `LinearLayout`
- `RelativeLayout`
- `DrawerLayout`
- `ConstraintLayout`
- `CoordinatorLayout`
- ...

# LinearLayout



Figure 21: Linearni raspored.

- Linearni raspored (LinearLayout) je raspored koji raspoređuje decu u jednom pravcu (vertikalno ili horizontalno)
- Deca linearnog rasporeda raspoređena su jedno pored drugog, tako da vertikalni raspored ima samo jedno dete po vrsti (a horizontalni samo jedno dete po koloni)

# Linearni raspored

```

2  <?xml version="1.0" encoding="utf-8"?>
3  <LinearLayout xmlns:android=
4      "http://schemas.android.com/apk/res/android"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:paddingLeft="16dp"
8      android:paddingRight="16dp"
9      android:orientation="vertical">
10     <EditText android:layout_width="match_parent"
11         android:layout_height="wrap_content"
12         android:hint="@string/to" />
13
14     <EditText android:layout_width="match_parent"
15         android:layout_height="wrap_content"
16         android:hint="@string/subject" />
17
18     <EditText android:layout_width="match_parent"
19         android:layout_height="0dp"
20         android:layout_weight="1"
21         android:gravity="top"
22         android:hint="@string/message" />
23
24     <Button android:layout_width="100dp"
25         android:layout_height="wrap_content"
26         android:layout_gravity="right"
27         android:text="@string/send" />
28 </LinearLayout>

```

To

Subject

Message

Send

# LinearLayout

Svojstvo	Opis
<code>layout_weight</code>	assigns an "importance" value to a view in terms of how much space it should occupy on the screen
<code>gravity</code>	specifies how an object should position its content, on both the X and Y axis, within its own bounds
<code>orientation</code>	use "horizontal" for a row, "vertical" for a column orientation

Table 5: Svojstva linearnog rasporeda.

# RelativeLayout



Figure 22: Relativni raspored.

- Relativni raspored (RelativeLayout) je raspored koji raspoređuje decu relativno u odnosu na sebe i jedno na drugo
- Pozicija pogleda može se specificirati u odnosu na elemente istog hijerarhijskog nivoa (levo, desno, iznad ili ispod drugog pogleda) ili u odnosu na roditelja (poravnat sa levom, desnom, gornjom ili donjom ivicom)



# Relativni raspored

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android=
   "http://schemas.android.com/apk/res/android"
4   android:layout_width="match_parent"
   android:layout_height="match_parent"
6   android:paddingLeft="16dp"
   android:paddingRight="16dp">
8
9   <EditText android:id="@+id/name"
10      android:layout_width="match_parent"
11      android:layout_height="wrap_content"
12      android:hint="@string/reminder" />
13
14  <Spinner android:id="@+id/dates"
15      android:layout_width="0dp"
16      android:layout_height="wrap_content"
17      android:layout_below="@id/name"
18      android:layout_alignParentLeft="true"
19      android:layout_toLeftOf="@+id/times" />
20
21  <Spinner android:id="@id/times"
22      android:layout_width="96dp"
23      android:layout_height="wrap_content"
24      android:layout_below="@id/name"
25      android:layout_alignParentRight="true" />
26
27  <Button android:layout_width="96dp"
28      android:layout_height="wrap_content"
29      android:layout_below="@id/times"
30      android:layout_alignParentRight="true"
31      android:text="@string/done" />
32 </RelativeLayout>

```

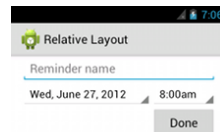


Figure 23: Relativni raspored.

# Svojstva relativnog rasporeda

Svojstvo	Opis
<code>layout_alignParentTop</code>	If "true", makes the top edge of this view match the top edge of the parent
<code>layout_centerVertical</code>	If "true", centers this child vertically within its parent
<code>layout_below</code>	Positions the top edge of this view below the view specified with a resource ID
<code>layout_toRightOf</code>	Positions the left edge of this view to the right of the view specified with a resource ID

Table 6: Svojstva relativnog rasporeda.

# ConstraintLayout

- Ograničavajući raspored (ConstraintLayout) je raspored koji omogućava određivanje pozicije i veličine pogleda na fleksibilan način
- Pozicije i veličine pogleda određuju se na osnovu ograničenja u odnosu na druge poglede, roditeljski raspored ili nevidljive vođice
- Ograničavajući raspored specijalizuje relativni raspored i prilagođen je radu u vizuelnom okruženju Android Studio
- Za ovaj raspored je u build.gradle skripti potrebna zavisnost:

```
dependencies {  
2     implementation 'androidx.constraintlayout:constraintlayout:2.1.3'  
    }  
4
```

## main.xml

```

2 <?xml version="1.0" encoding="utf-8"?>
3 <androidx.constraintlayout.widget.ConstraintLayout
4     xmlns:android="http://schemas.android.com/apk/res/android"
5     xmlns:app="http://schemas.android.com/apk/res-auto"
6     xmlns:tools="http://schemas.android.com/tools"
7     android:layout_width="match_parent"
8     android:layout_height="match_parent"
9     app:layout_behavior="@string/appbar_scrolling_view_behavior"
10    tools:context="com.journaldev.constraintlayoutplaying.MainActivity"
11    >
12    <Button
13        android:id="@+id/button"
14        android:layout_width="wrap_content"
15        android:layout_height="wrap_content"
16        android:text="Button"
17        app:layout_constraintBottom_toBottomOf="parent"
18        app:layout_constraintLeft_toLeftOf="parent"
19        app:layout_constraintRight_toRightOf="parent"
20        app:layout_constraintTop_toTopOf="parent" />
21 </androidx.constraintlayout.widget.ConstraintLayout>

```

# ConstraintLayout

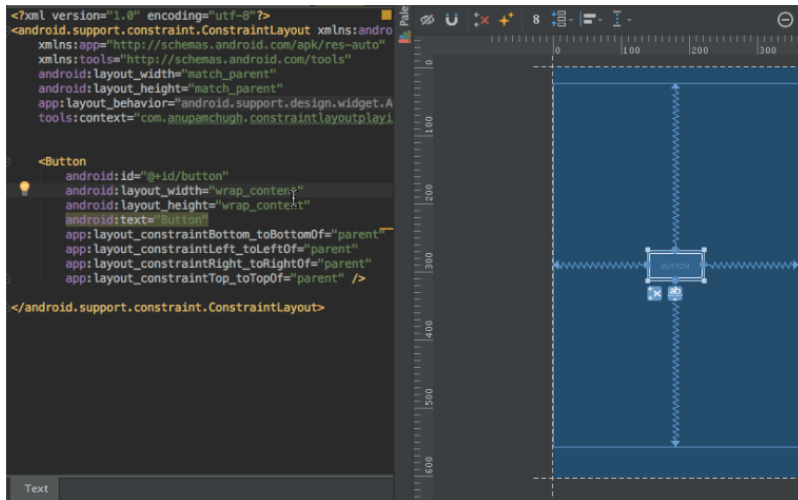


Figure 24: Ograničavajući raspored.

# FrameLayout

- FrameLayout raspored prikazuje više pogleda koji će biti raspoređeni jedan na drugom (poslednji koji je dodat biće raspoređen na vrhu)
- Veličina FrameLayout rasporeda odgovara veličini najvećeg pogleda koga sadrži (ako to dozvoljava roditelj)

# main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <FrameLayout
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     xmlns:android="http://schemas.android.com/apk/res/android">
6
7     <ImageView
8         android:id="@+id/iv_logo"
9         android:layout_height="match_parent"
10        android:layout_width="match_parent"
11        android:src="@drawable/logo" />
12
13    <TextView
14        android:id="@+id/tv_label"
15        android:text="@string/label"
16        android:layout_height="match_parent"
17        android:layout_width="match_parent" />
18
19 </FrameLayout>
```

# FrameLayout

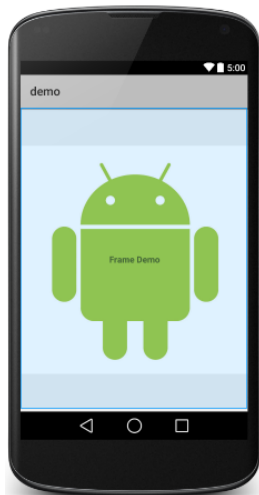


Figure 25: Okvir raspored.



# CoordinatorLayout

- Koordinirajući raspored (CoordinatorLayout) upravlja interakcijom između pogleda koje sadrži
- Obično se koristi kao koreni raspored aktivnosti ili fragmenta

## main.xml

```

1  <android.support.design.widget.CoordinatorLayout
2  xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
3  android:id="@+id/coordinator_layout"
   android:layout_width="match_parent"
4  android:layout_height="match_parent">

5
6  <android.support.design.widget.AppBarLayout
   android:id="@+id/app_bar"
7  android:layout_width="match_parent"
   android:layout_height="wrap_content">

8
9
10  <android.support.v7.widget.Toolbar
   android:layout_width="match_parent"
11  android:layout_height="?attr/actionBarSize"/>

12
13 </android.support.design.widget.AppBarLayout>

14
15 <!-- Main content -->

16
17 <android.support.design.widget.FloatingActionButton
18  android:layout_width="wrap_content"
   android:layout_height="wrap_content"
19  android:layout_margin="16dp"
   android:contentDescription="@string/add_item"
20  android:src="@drawable/ic_add_24dp"
   app:layout_anchor="@id/app_bar"
21  app:layout_anchorGravity="bottom|right|end"/>

22
23 </android.support.design.widget.CoordinatorLayout>

```

# CoordinatorLayout

