

# Android programiranje

**Marko Arsenović**  
**Srđan Sladojević**

# Sadržaj

- Resources
- Views
- Widgets
- Layouts
- Styles
- Themes



# Resursi i konfiguracija uredjaja

- Android aplikacije mogu da sadrze i resurse (tekst, slike, audio klipove, itd.)
- Resurse treba eksternalizovati da bi se omogucilo:
  - prilagodjavanje aplikacije razlicitim konfiguracijama uredaja(dimenzije, rezolucija i orijentacija ekrana, jezik, region, itd.)
  - laksa sinhronizacija izmedu programera i grafickih dizajnera



# Resursi

Tip	Opis
drawable	vektorske ili rasterske slike
layout	deklaracije grafickog korisnickog interfejsa
raw	“sirovi” podaci
values	proste vrednosti (nizovi, boje, stringovi, stilovi, itd.)
xml	proizvoljni XML dokumenti

```
MyProject/  
    src/  
        MainActivity.java  
    res/  
        drawable/  
            graphic.png  
        layout/  
            main.xml  
            info.xml  
        mipmap/  
            icon.png  
        values/  
            strings.xml
```



# Resursi

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



# Resursi

## drawable

```
<?xml version="1.0" encoding="utf-8"?>
<bitmap xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@drawable/icon_ca" />
```

## layout

```
<?xml version="1.0" encoding="utf-8"?>
<merge>
    <include layout="@layout/main_ltr"/>
</merge>
```

## strings

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello</string>
    <string name="hi">@string/hello</string>
</resources>
```



# Konfiguracija uredjaja

- Resursi se mogu definisati za razlicite konfiguracije uredjaja (ekran niske, srednje, visoke rezolucije)
- Razlicitim konfiguracijama uredjaja odgovaraju resursi koji se nalaze u direktorijumima sa razlicitim sufiksima (ldpi, mdpi, hdpi)
- Moguce je istovremeno definisati resurse za vise tipova konfiguracije (ekranu visoke rezolucije u nocnom modu odgovara sufiks night-hdpi)
- Uvek treba definisati podrazumevane resurse



```
res/  
  drawable/  
    icon.png  
    background.png  
  drawable-hdpi/  
    icon.png  
    background.png  
  ...
```

# Konfiguracija uredjaja

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

Tipovi konfiguracije uredjaja





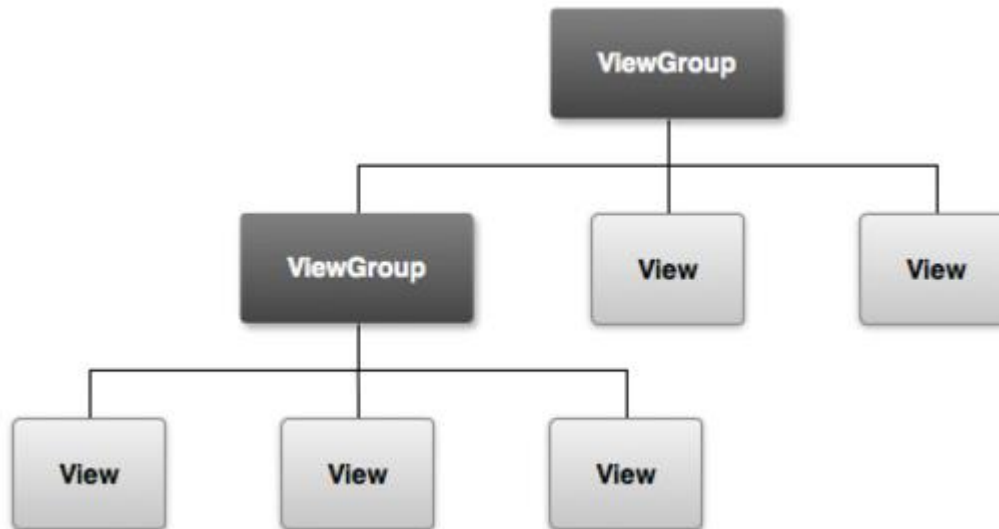
# Konfiguracija uredjaja

- Za različite rezolucije ekrana bi trebalo pripremiti slike razlicitih rezolucija
- Za različite velicine ekrana bi trebalo pripremiti različite rasporede GUI-a



# Views i Layouts

- Graficki korisnicki interfejs bilo koje aktivnosti moze se predstaviti hijerarhijom pogleda (View) i rasporeda (Layout)
- Pogledi predstavljaju komponente GUI-a, a rasporedi sadrze poglede i odredjuju kako se oni rasporedjuju na ekranu



Hijerarhija pogleda i rasporeda GUI-a



# Views

- Android API sadrzi razlicite tipove view-ova (labele, tekstualna polja, dugmad, itd.)
- Razliciti tipovi view-ova sadrze razlicita svojstva koja odredjuju njihovo stanje (vidljivost, transparentnost, itd.) i mogu obradjivati razlicite dogadjaje (dodir, pritisak tastera, promenu fokusa, itd.)



# Views

## Vaznija svojstva

Svojstvo	Opis
id	Ime identifikacije view-a
clickable	Definise da li view reaguje na klik događaj
focusable	Kontrolise da li view ima fokus
visibility	Kontrolise inicijalnu vidljivost view-a
alpha	Kontrolise inicijalnu transparentnost

## Vazniji događaji

Događaj	Opis
touch	Akcija kvalifikovana kao touch event-press, relase...
click	Korisnik pritiska komponentu
long click	Korisnik pritiska i drži komponentu
focus change	Korisnik navigira ka ili od komponente
key	Korisnik je fokusiran na komponentu i pritiska ili pusta hardware key na uređaju



# Pravljenje View-a

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



# Widgets

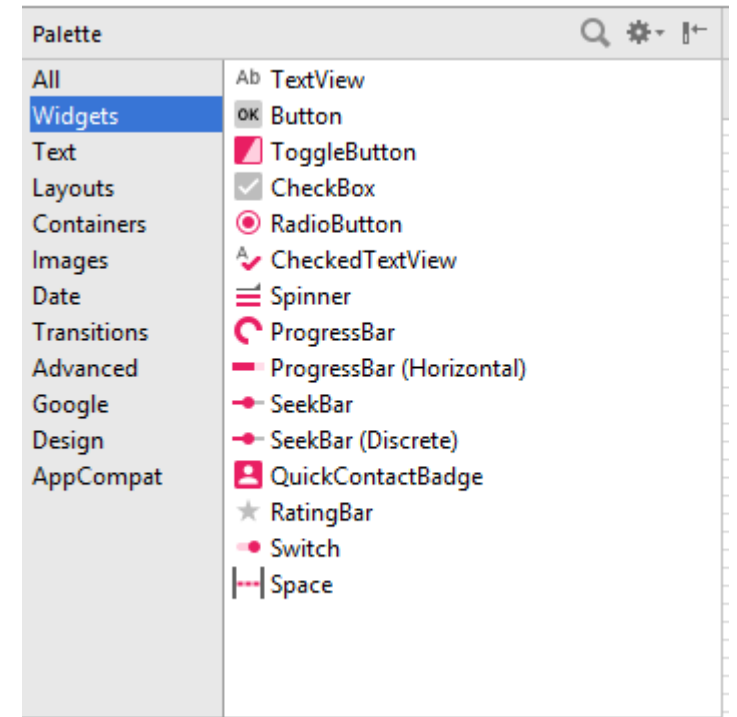
- Termin widget se može odnositi na koliko različitih koncepata u Androidu. Kada se o njima govori, većina ljudi misli na application widgets, koji se mogu obično videti na početnom ekranu – mini aplikacije koje obezbeđuju podset funkcionalnosti glavne aplikacije.
- Widget se može odnositi i na specijalizovane elemente koji su postavljeni u fajlovima rasporeda.
- Android SDK obezbeđuje veliki broj predefinisani widget-a izvedeni iz View klase.



# Widgets

Tipovi:

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



# Widgets

- Komponenta **TextView** prikazuje tekst i omogućava njegovo kopiranje

```
<TextView  
    android:id="@+id/email_address"  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:text="@string/hello_world" />
```





# Widgets

- Komponenta **ImageView** prikazuje proizvoljnu sliku iz razlicitih izvora
- Omogucava i skaliranje, odsecanje, primenu filtera, itd.

```
<ImageView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:src="@drawable/my_image"/>
```



# Widgets

- Komponenta `EditText` omogućava unos teksta
- Pored unosa teksta, omogućava i niz drugih aktivnosti kao sto su označavanje, isecanje, kopiranje, itd.
- Moguce je specificirati tip tastature (normalna, numericka, telefonska, itd.) ili ponasanje tastature (automatsko pretvaranje pocetnih slova reci u velika slova, itd.)

```
<EditText
    android:id="@+id/email_address"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/password_hint"
    android:inputType="text|textPassword" />
```



# Widgets

Vrednost	Opis
text	normalna tastatura
textEmailAddress	normalna tastatura sa @ znakom
textUri	normalna tastatura sa / znakom
number	osnovna numericka tastatura
phone	telefonska tastatura
textCapWords	normalna tastatura koja automatski pretvara pocetno slovo recenice u veliko slovo
textAutoCorrect	normalna tastatura koja ispravlja ceste pravopisne greske
textPassword	normalna tastatura koja unesene znakove prikazuje kao tacke
textMultiLine	normalna tastatura koja omogucava korisnicima da unose tekst u vise redova

Vrednosti inputType atributa



# Widgets

- Komponenta **Button** prikazuje tekst ili sliku koja simbolizuju akciju
- Kada korisnik pritisne dugme generise se click dogadaj
- Metoda koja obraduje ovaj dogadaj specicira se **onClick** atributom i mora biti sadrzana u aktivnosti kojoj je dugme pridruzeno

Alarm

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    ... />
```



```
<ImageButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/button_icon"
    ... />
```



```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    android:drawableLeft="@drawable/button_icon"
    ... />
```



# Widgets

- Komponenta **Button** koja obraduje događaj specificiran sa **onClick** atributom

```
<?xml version="1.0" encoding="utf-8"?>
<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/button_send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_send"
    android:onClick="sendMessage" />
```

```
/** Called when the user touches the button */
public void sendMessage(View view) {
    // Do something in response to button click
}
```



# Widgets

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

ATTENDING?

☒ Yes ☐ Maybe ☐ No



```
<?xml version="1.0" encoding="utf-8"?>
<RadioGroup xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton android:id="@+id/radio_pirates"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/pirates"
        android:onClick="onRadioButtonClicked"/>
    <RadioButton android:id="@+id/radio_ninjas"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/ninjas"
        android:onClick="onRadioButtonClicked"/>
</RadioGroup>
```

# Widgets

- Komponenta **RadioButton**

```
<?xml version="1.0" encoding="utf-8"?>
<RadioGroup xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton android:id="@+id/radio_pirates"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/pirates"
        android:onClick="onRadioButtonClicked"/>
    <RadioButton android:id="@+id/radio_ninjas"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/ninjas"
        android:onClick="onRadioButtonClicked"/>
</RadioGroup>
```

```
public void onRadioButtonClicked(View view) {
    // Is the button now checked?
    boolean checked = ((RadioButton) view).isChecked();

    // Check which radio button was clicked
    switch(view.getId()) {
        case R.id.radio_pirates:
            if (checked)
                // Pirates are the best
            break;
        case R.id.radio_ninjas:
            if (checked)
                // Ninjas rule
            break;
    }
}
```



# Widgets

- Komponenta **ToggleButton** omogucava korisniku da promeni podesavanje izmedu dva stanja



*Toggle buttons*



*Switches (in Android 4.0+)*

```
ToggleButton toggle = (ToggleButton) findViewById(R.id.togglebutton);
toggle.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
        if (isChecked) {
            // The toggle is enabled
        } else {
            // The toggle is disabled
        }
    }
});
```

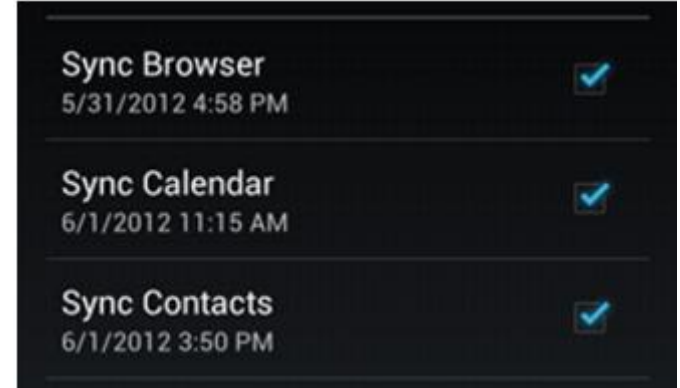




# Widgets

- Komponenta **CheckBox** omogućava korisniku da izabere jednu ili vise opcija iz skupa opcija
- Opcije se obicno prikazuju u vertikalnoj listi.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <CheckBox android:id="@+id/checkbox_meat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/meat"
        android:onClick="onCheckboxClicked"/>
    <CheckBox android:id="@+id/checkbox_cheese"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/cheese"
        android:onClick="onCheckboxClicked"/>
</LinearLayout>
```



# Widgets

- Komponenta **CheckBox**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <CheckBox android:id="@+id/checkbox_meat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/meat"
        android:onClick="onCheckboxClicked"/>
    <CheckBox android:id="@+id/checkbox_cheese"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/cheese"
        android:onClick="onCheckboxClicked"/>
</LinearLayout>
```



```
public void onCheckboxClicked(View view) {
    // Is the view now checked?
    boolean checked = ((CheckBox) view).isChecked();

    // Check which checkbox was clicked
    switch(view.getId()) {
        case R.id.checkbox_meat:
            if (checked)
                // Put some meat on the sandwich
            else
                // Remove the meat
                break;
        case R.id.checkbox_cheese:
            if (checked)
                // Cheese me
            else
                // I'm lactose intolerant
                break;
        // TODO: Veggie sandwich
    }
}
```

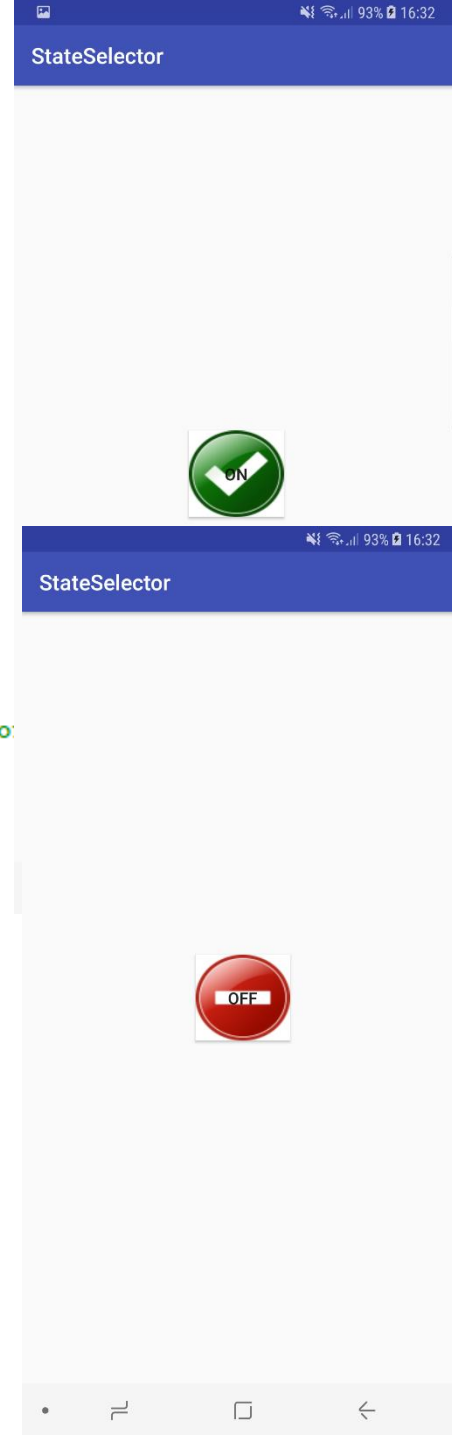
# Widgets

- Primer StateSelector: upotreba grafika za prikazivanje stanja dugmeta koriscenjem Android State Selector-a

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
  <!--
    <item
      android:drawable="@android:color/darker_gray"
      android:state_checked="true"/>
    <item
      android:drawable="@android:color/white"
      android:state_checked="false"/>
  -->
  <item
    android:drawable="@drawable/checked_on"
    android:state_checked="true"/>
  <item
    android:drawable="@drawable/checked_off"
    android:state_checked="false"/>
</selector>
```

```
<ToggleButton
  android:id="@+id/toggleButton"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="true"
  android:background="@drawable/state_selector"
  android:text="New ToggleButton" />
```

```
▼ res
  ▼ drawable
    checked_off.png
    checked_on.png
    state_selector.xml
  ▼ layout
    activity_main.xml
```

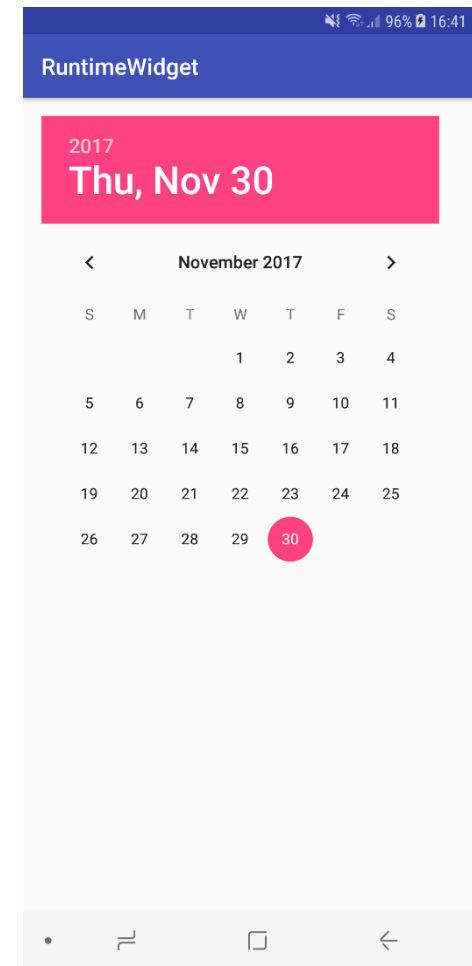


# Widgets

- Primer RuntimeWidget – kreiranje komponente u vreme pokretanja

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

    RelativeLayout layout = (RelativeLayout) findViewById(R.id.layout);
    DatePicker datePicker = new DatePicker(this);
    layout.addView(datePicker);
}
```



# Views

- Custom View:
  1. Kriirati novu klasu koja prosiruje klasu View
  2. Kreirati sopstveni konstruktor
  3. Zameniti vrednosti metoda `onMeasure()`, standardna implementacija ce vratiti elicinu 100x100
  4. Zameniti vrednosti metoda `onDraw()` I standardna implemntacija nece iscrtati nista
  5. Difinisti prilagodjene metode I listener-e
  6. Implementirati prilagodjenu funkcionalnost



# Views

- Primer CustomView

```
public class CustomView extends View {  
  
    final Paint mPaint = new Paint();  
  
    public CustomView(Context context) {  
        super(context);  
        mPaint.setColor(Color.BLACK);  
        mPaint.setTextSize(30);  
    }  
  
    @Override  
    protected void onDraw(Canvas canvas) {  
        super.onDraw(canvas);  
        setBackgroundColor(Color.DKGRAY);  
        canvas.drawText("Custom View Text", 100, 100, mPaint);  
    }  
}
```

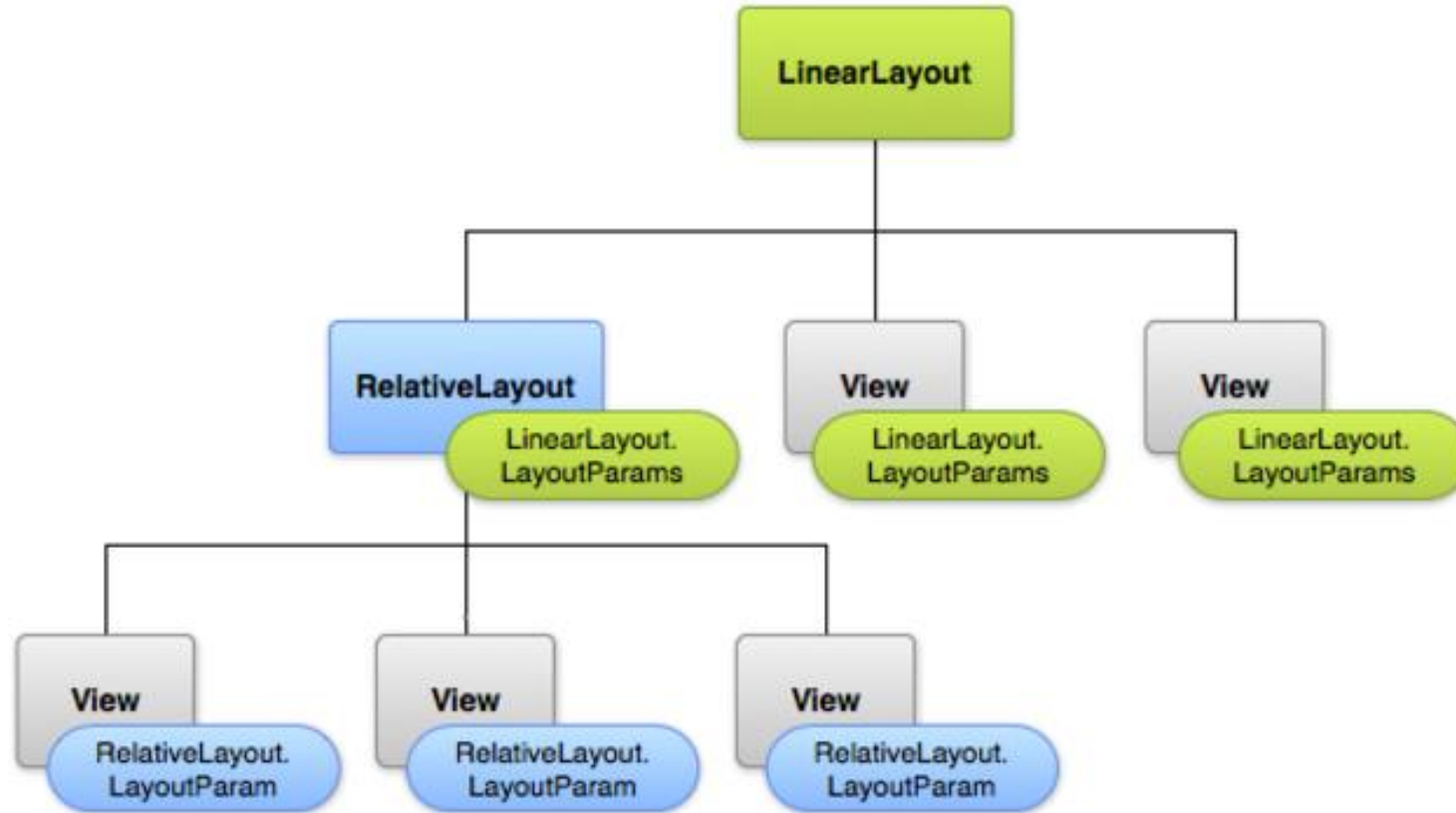


# Views - Svojstva

- Pogled ima geometriju pravougaonika (poziciju i dimenzije)
- Poziciju i dimenzije pogleda odredjuje vrsta rasporeda koji ga sadrzi i svojstva pogleda koja zavise od vrste rasporeda
- Ova svojstva imaju oblik `layout_*`
- Pozicija i dimenzije se izrazavaju u jedinici density-independent pixels (dp)
- Jedan dp odgovara velicini piksela pri rezoluciji od 160 dpi
- Postoje dve specijalne vrednosti svojstva `layout_width` i `layout_height`
  - `wrap_content` (pogled se prilagodjava dimenzijama dece)
  - `fill_parent` (pogled se prilagodjava dimenzijama roditelja)



# Svojstva View-a



Svojstva pogleda





# Views - Iscrtavanje

- Svaki view iscrtava sebe i svoju decu
- Iscrtavanje pogleda izvrsava se u dva prolaza:
  - prolazu merenja (measure pass)
  - prolazu rasporedjivanja (layout pass)



# Layouts

- U Androidu je korisnički interfejs definisan u rasporedu (layouts). Layout može da bude deklarisan u XML fajlu ili kreiran dinamički u kodu.
- Fajlovi layout-a su sacuvani u direktorijum /res/layout i referencirani u kodu pomoću identifikatora `R.layout.<ime layout-a bez ekstenzije>`
- Android obezbeđuje korisne raznovrsne klase Layouts koje sadrže i organizuju pojedinačne elemente aktivnosti



# Layouts

- Objekat ViewGroup je objekat kontejner koji sluzi kao osnovna klasa za Androidovu porodicu Layout klasa.
- Views elementi koji su postavljeni u layout formiraju hierarhiju u kojoj je layout root elemenat.



# Layouts

- Android obezbedjuje ugradjene layout-e dizajnirane za specificne namene:
- RelativeLayout – omogucava da prikazi budu pozicionirani u odnosu na druge elemente
- LinearLayout – moze da poredja prikaze vertikalno ili da ih poravna horizontalno u zavisnosti od odredjene orijentacije
- TableLayout – za rasporedjivanje mreza prikaza



# Layouts

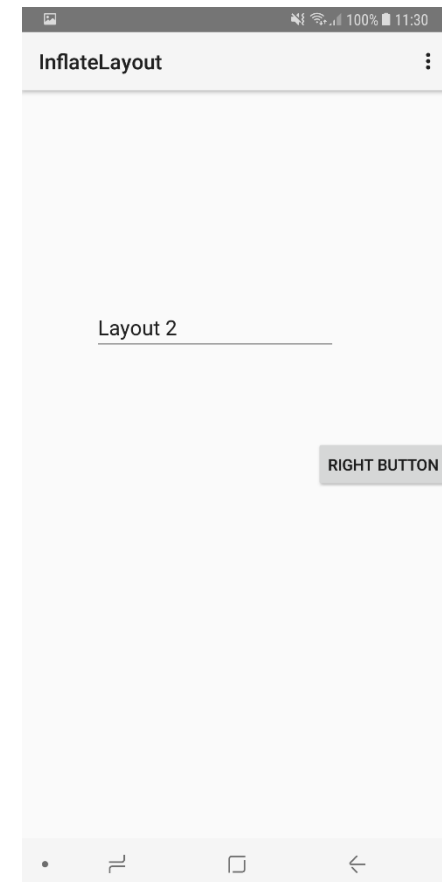
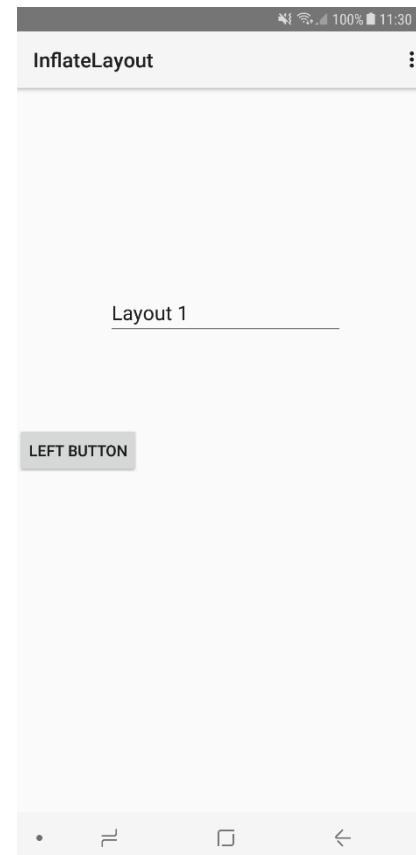
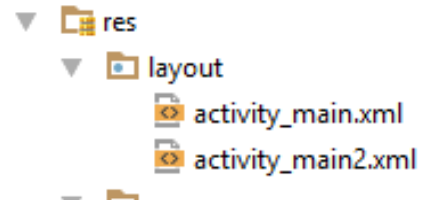
- Layout i klase ViewGroup mogu da budu ugnjezdeni jedni unutar drugih za kreiranje slozenih konfiguracija.
- Za upravljanje vidzetime, lisama, tabelama, galerijama I drugih formata prikaza postoji vise desetina razlicitih objekata Layout klase, pored toga uvek se moze izvesti novi Layout iz osnovne klase da bi smo kreirali sopstveni raspored.



# Layouts

- Primer InflateLayout: Definisanje i podizanje rasporeda
- setContentView()

```
public void onClickLeft(View view) {  
    setContentView(R.layout.activity_main2);  
}  
  
public void onClickRight(View view) {  
    setContentView(R.layout.activity_main);  
}
```



# Layouts

- 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 od ili ispod drugog pogleda) ili u odnosu na roditelja (poravnat sa levom ili donjom ivicom)



# Layouts

- Primer RelativeLayout - koristan za smanjivanje broja ugnjezdenih rasporeda, vazno za memoriju i obradu

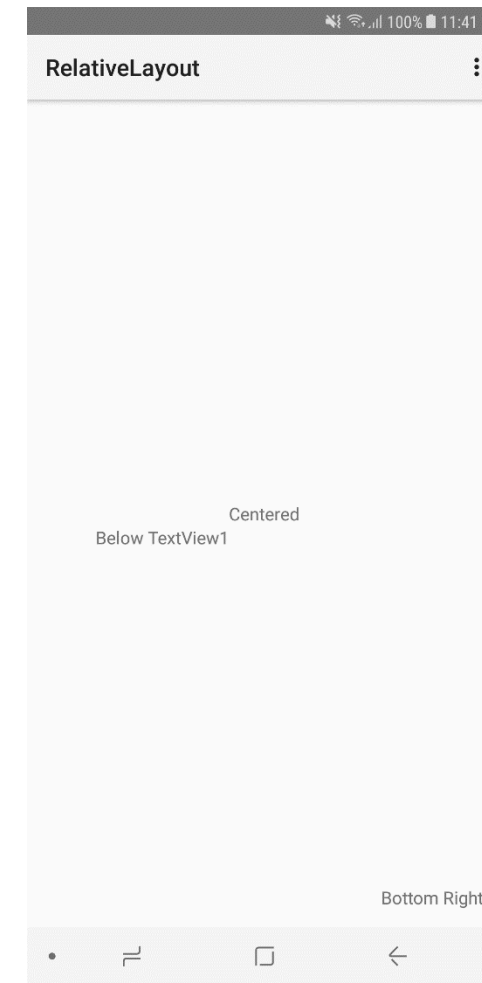
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent" android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp" tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Centered"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Below TextView1"
        android:layout_below="@+id/textView1"
        android:layout_toLeftOf="@+id/textView1" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Bottom Right"
        android:layout_alignParentBottom="true"
        android:layout_alignParentEnd="true" />

</RelativeLayout>
```





# Layouts

- Primer RelativeLayout - koristan za smanjivanje broja ugnjezdenih rasporeda, vazno za memoriju i obradu

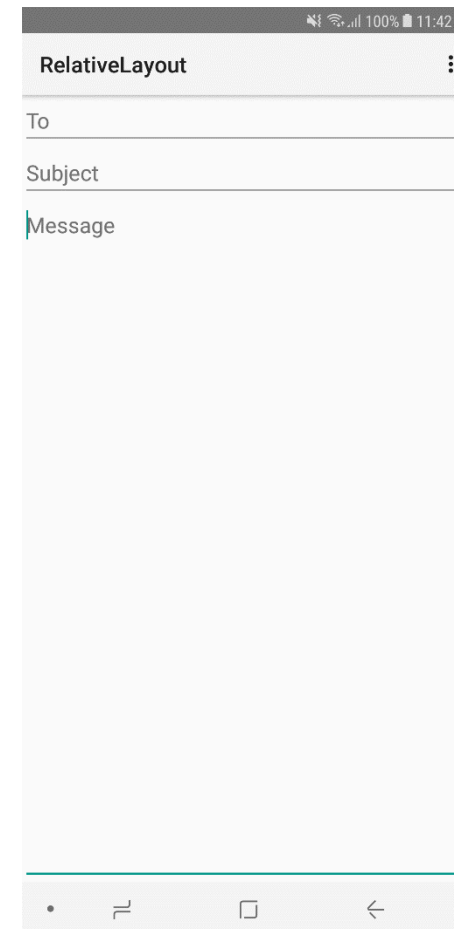
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent" android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp" tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Centered"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Below TextView1"
        android:layout_below="@+id/textView1"
        android:layout_toLeftOf="@+id/textView1" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Bottom Right"
        android:layout_alignParentBottom="true"
        android:layout_alignParentEnd="true" />

</RelativeLayout>
```



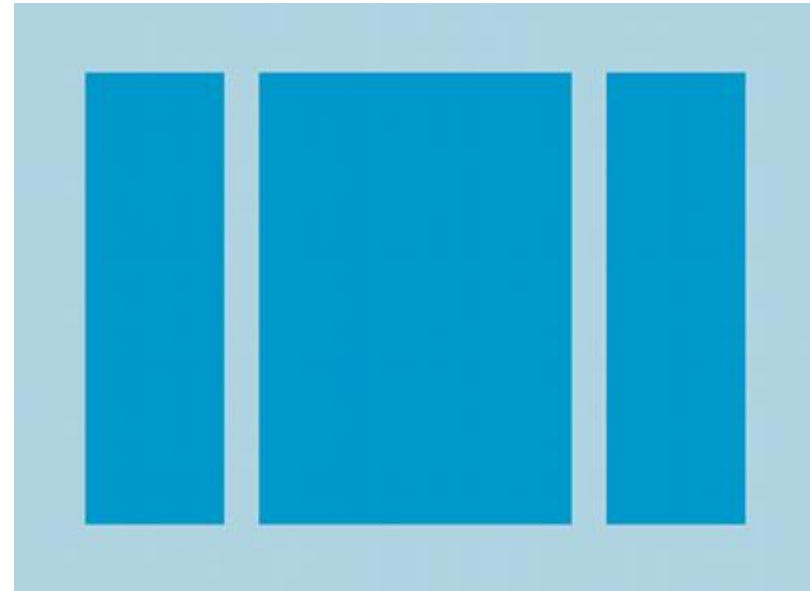
# Layouts

- ConstraintLayout
- [Link](#)



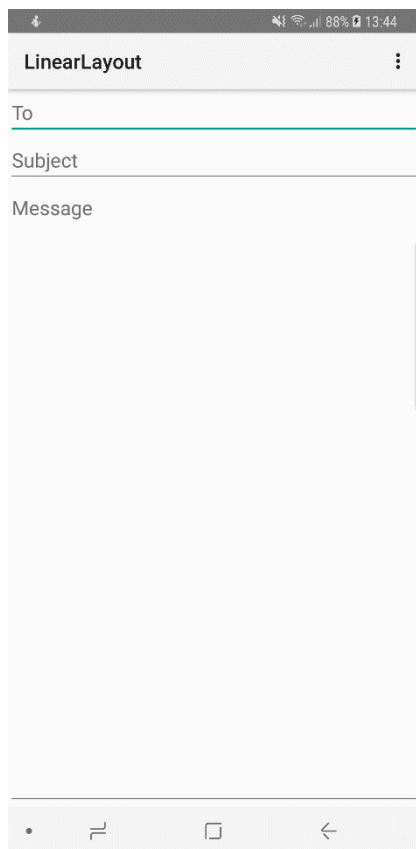
# Layouts

- 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)



# Layouts

- Primer LinearLayout:



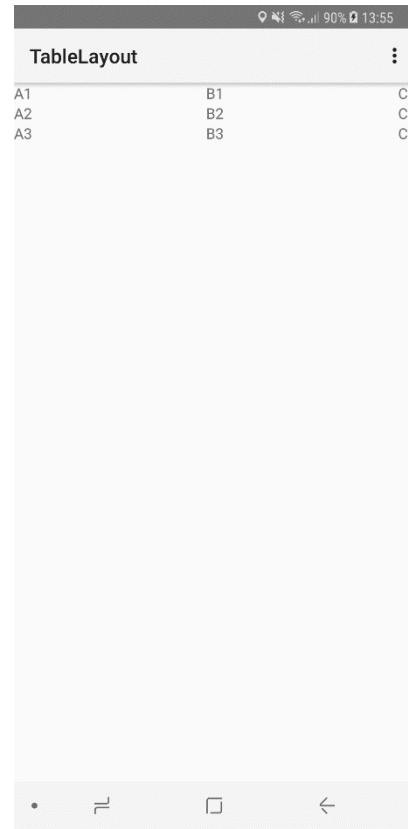
```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2     android:orientation="vertical"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent">
5     <EditText
6         android:id="@+id/editTextTo"
7         android:layout_width="match_parent"
8         android:layout_height="wrap_content"
9         android:hint="To" />
10    <EditText
11        android:id="@+id/editTextSubject"
12        android:layout_width="match_parent"
13        android:layout_height="wrap_content"
14        android:hint="Subject" />
15    <EditText
16        android:id="@+id/editTextMessage"
17        android:layout_width="match_parent"
18        android:layout_height="0dp"
19        android:layout_weight="1"
20        android:gravity="top"
21        android:hint="Message" />
22 </LinearLayout>
```

- LinearLayout ima ključnu funkciju koju raspored RelativeLayout ne nudi: `layout_weight`.
- Omogućujemo prikazu da dinamički menja veličinu na osnovu dostupnog prostora.
- Opcije uključuju mogućnost da prikaz popuni ceo preostali prostor (ako prikaz ima veću širinu), da više prikaza bude postavljeno u određeni prostor (ako svi imaju istu širinu) ili da prikazi budu proporcionalno razmaknuti po njihovoj širini.



# Layouts

- Primer kreiranja tabele: TableLayout
- TableLayout obezbedjuje redove i kolone dodate dinamicki dok se gradi tabela



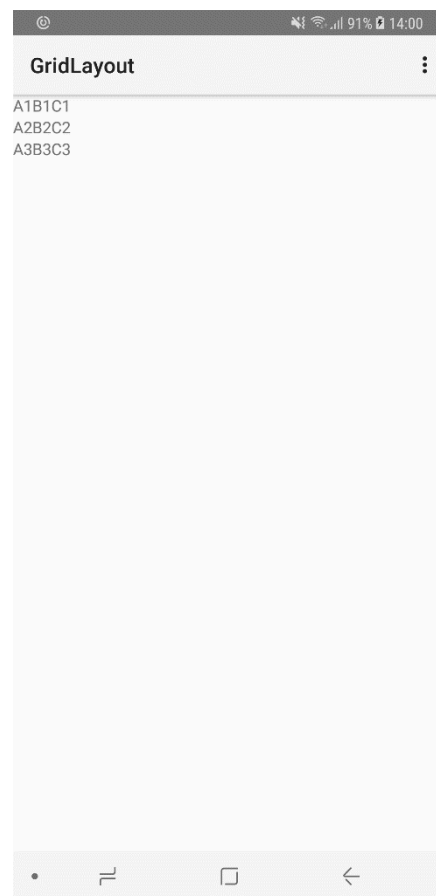
A1	B1	C1
A2	B2	C2
A3	B3	C3

```
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="1"
    android:orientation="vertical">
    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="A1"
            android:id="@+id/textView1" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="B1"
            android:layout_gravity="center"
            android:id="@+id/textView2" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="C1"
            android:id="@+id/textView3" />
    </TableRow>
    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="A2"
            android:id="@+id/textView4" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="B2"
            android:layout_gravity="center"
            android:id="@+id/textView5" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="C2"
            android:id="@+id/textView6" />
    </TableRow>
```



# Layouts

- Primer kreiranja tabela: GridLayout
- U rasporedu GridLayout koristi se suprotan pristup.
- Broj redova i kolona se odredjuju prilikom kreiranja tabele
- Ne treba da odredjujete informaciju za red ili kolinu, Android ce automatski dodati svaki element u celiju prema rasporedu.



```
1 <GridLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:layout_width="match_parent"
4   android:layout_height="match_parent"
5   android:columnCount="3"
6   android:rowCount="3">
7   <TextView
8       android:layout_width="wrap_content"
9       android:layout_height="wrap_content"
10      android:text="A1"
11      android:id="@+id/textView1"
12  />
13   <TextView
14       android:layout_width="wrap_content"
15       android:layout_height="wrap_content"
16      android:text="B1"
17      android:id="@+id/textView2"
18  />
19   <TextView
20       android:layout_width="wrap_content"
21       android:layout_height="wrap_content"
22      android:text="C1"
23      android:id="@+id/textView3" />
24   <TextView
25       android:layout_width="wrap_content"
26       android:layout_height="wrap_content"
27      android:text="A2"
28      android:id="@+id/textView4" />
29   <TextView
30       android:layout_width="wrap_content"
31       android:layout_height="wrap_content"
32      android:text="B2"
33      android:id="@+id/textView5" />
34   <TextView
35       android:layout_width="wrap_content"
36       android:layout_height="wrap_content"
37      android:text="C2"
38      android:id="@+id/textView6" />
39   <TextView
40       android:layout_width="wrap_content"
41       android:layout_height="wrap_content"
42      android:text="A3"
43      android:id="@+id/textView7" />
```

# Layouts

- ListView i GridView
- Naslednici grupe ViewGroup, ali se cesce koriste kao elementi
- Umesto da se definisu svi moguci elementi koji mogu da ih popune u vreme dizajna, sadrzaji su kreirani dinamicki iz podataka koji se prosledjuju elementu



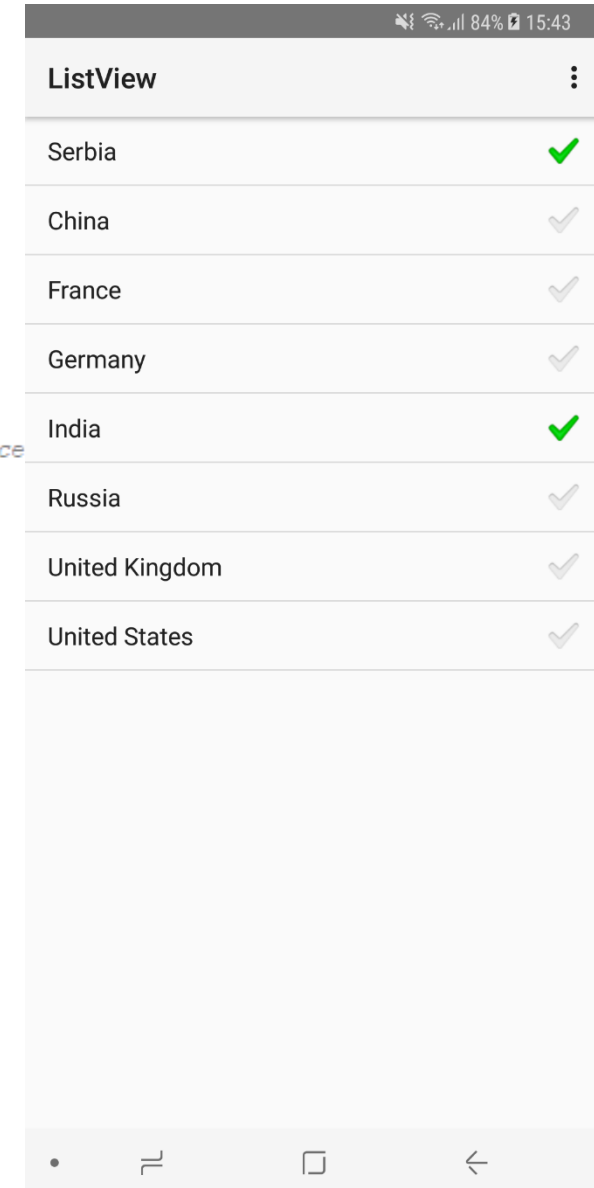
# Layouts

- Primer ListView

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    String[] countries = new String[]{"Serbia", "China", "France", "Germany", "India", "Russia", "United Kingdom", "United States"};
    //ListAdapter countryAdapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, countries);
    ListAdapter countryAdapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_checked, countries); //sa multiple choice
    setListAdapter(countryAdapter);

    getListView().setOnItemClickListener((parent, view, position, id) -> {
        String s = ((TextView) view).getText() + " " + position;
        Toast.makeText(getApplicationContext(), s, Toast.LENGTH_SHORT).show();
    });

    getListView().setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
}
```

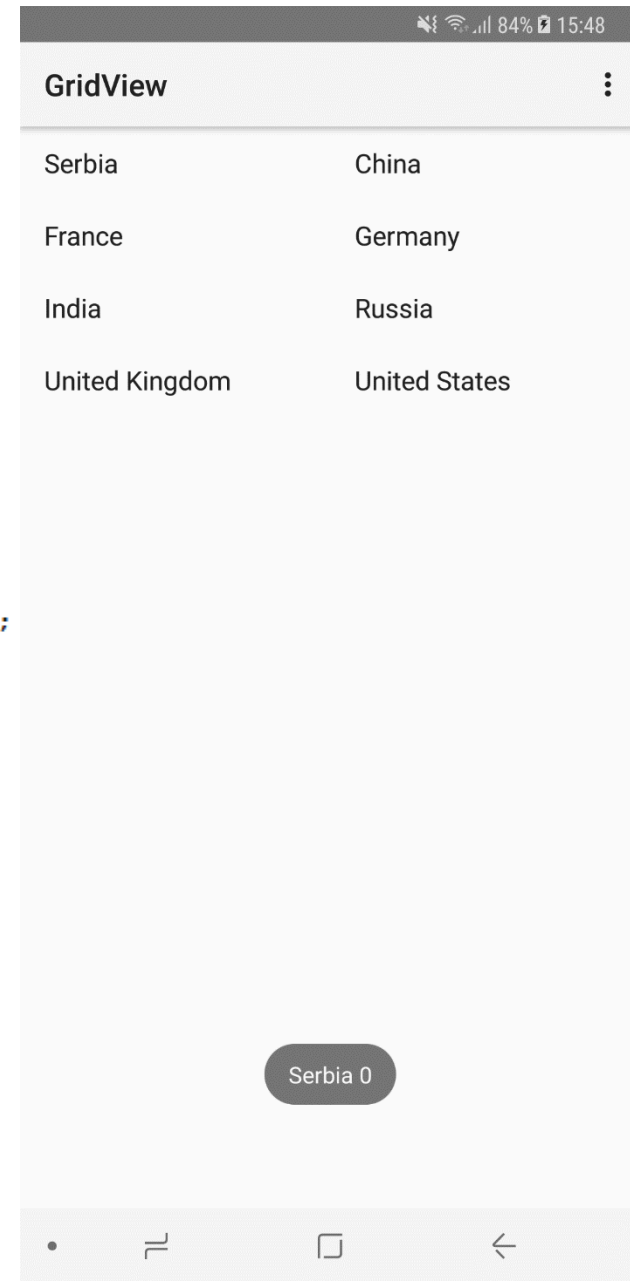




# Layouts

- Primer GridView

```
public class MainActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        GridView gridView = new GridView(this);  
        setContentView(gridView);  
        String[] countries = new String[]{"Serbia", "China", "France", "Germany", "India", "Russia", "United Kingdom", "United States"};  
        ListAdapter countryAdapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, countries);  
        gridView.setAdapter(countryAdapter);  
        gridView.setNumColumns(2);  
        gridView.setOnItemClickListener((parent, view, position, id) → {  
            String s = ((TextView) view).getText() + " " + position;  
            Toast.makeText(getApplicationContext(), s, Toast.LENGTH_SHORT).show();  
        });  
    }  
}
```

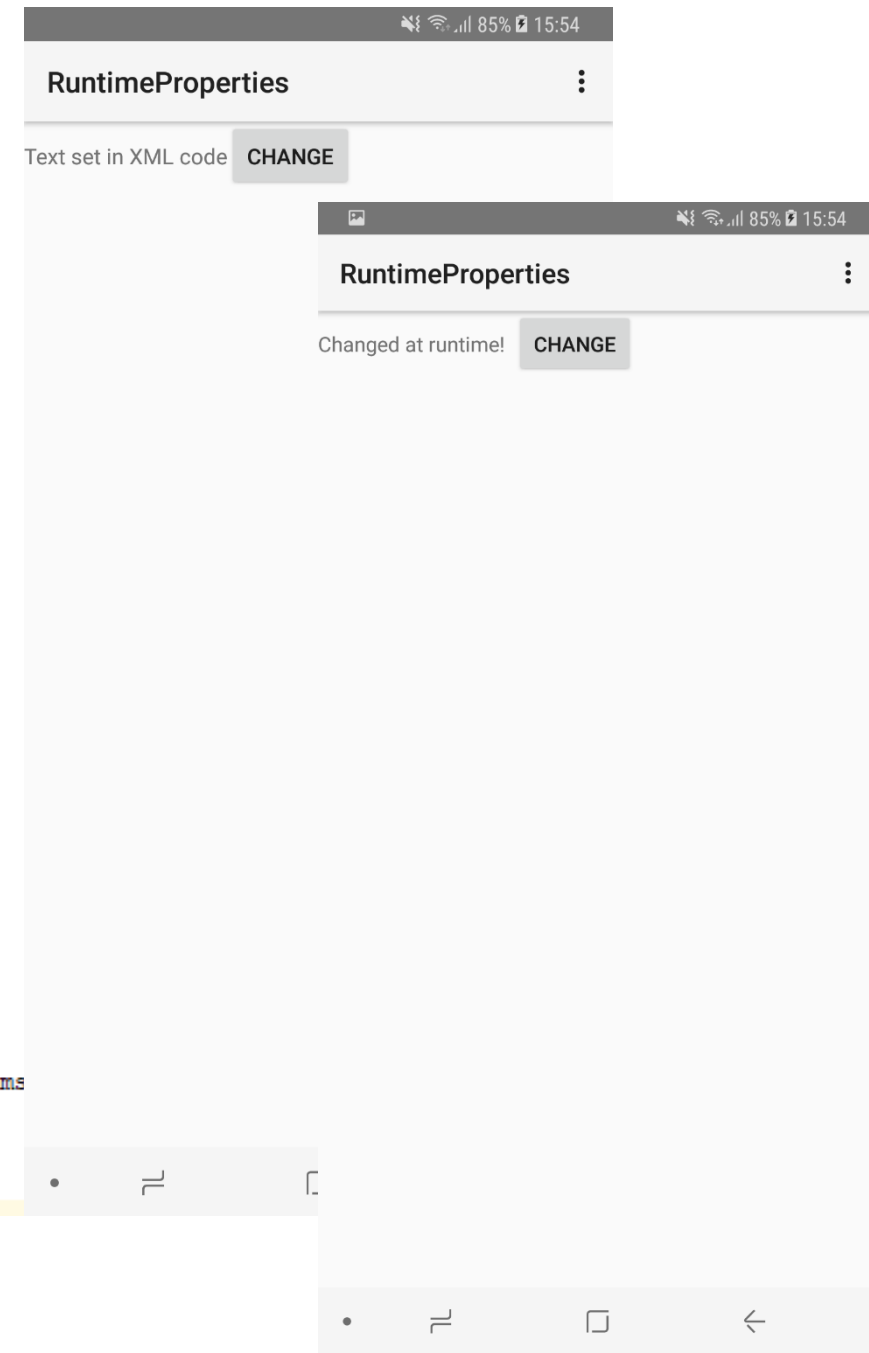


# Layouts

- Primer RuntimeProperties – menjanje parametara Layout-a u toku pokretanja
- Pozeljan je “separation of concerns” u razvoju aplikacija, ali podržano je u Androidu da se korisnicki interfejs promeni iz Java koda.

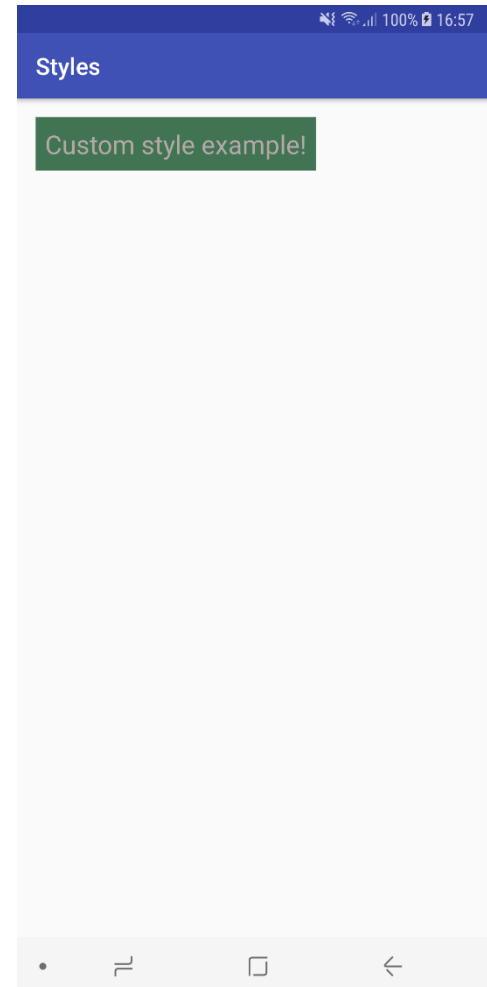
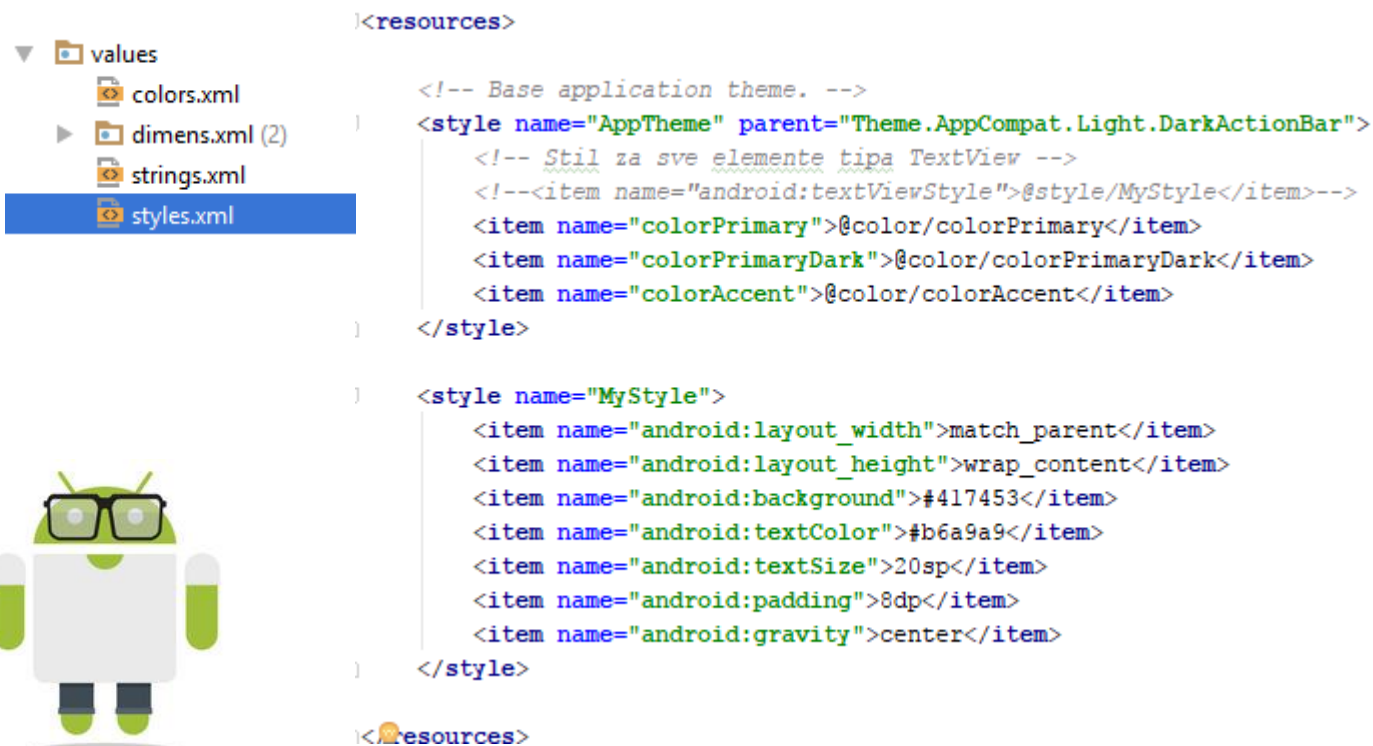


```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    Button button = (Button)findViewById(R.id.button);  
    button.setOnClickListener((view) -> {  
        ((TextView)findViewById(R.id.textView)).setText("Changed at runtime!");  
        LinearLayout.LayoutParams params = (LinearLayout.LayoutParams)view.getLayoutParams();  
        params.leftMargin += 20;  
    });  
}
```



# Styles

- Stil je kolekcija parametara za definisanje izgleda prikaza.
- Kreiranje stila je jednostavno – izvuci podesavanja iz Layout-a i postaviti ih u izvor stila.



```
<TextView
    style="@style/MyStyle"
    android:text="Custom style example!"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
</RelativeLayout>
```

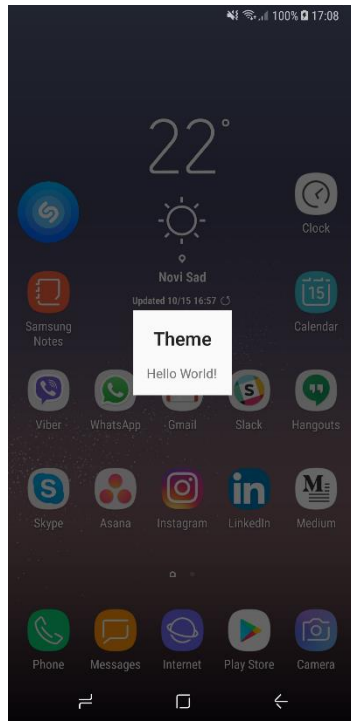
# Themes

- Tema je stil primenjen na aktivnost ili celu aplikaciju.
- Da bi se podesila tema, koristi se atribut `android:theme` u `AndroidManifest.xml`
- Atribute theme se primenjuje na element `<Application>`, kao i na elemente `<Activity>`.
- Svi prikazi unutar odredjenog elementa ce biti stilizovani pomocu specificovane teme.



# Themes

- Primer Themes:
- Nova tema MyDialog, nasledjuje snovnu AppTheme
- Deklarisana tema dodeljena je u aktivnost u **AndroidManifest.xml**



```
<resources>

<!-- Base application theme. -->
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>

<!--Nasledjujemo AppTheme, sistemska tema definisana u kodu-->
<style name="AppTheme.MyDialog">
    <item name="android:windowIsFloating">true</item>
</style>

</resources>
```

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Theme"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity"
            android:theme="@style/AppTheme.MyDialog">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

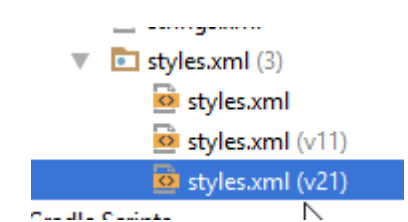
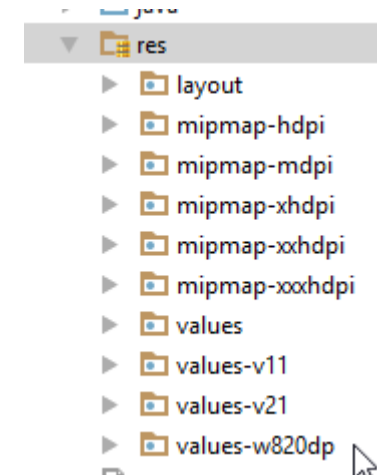
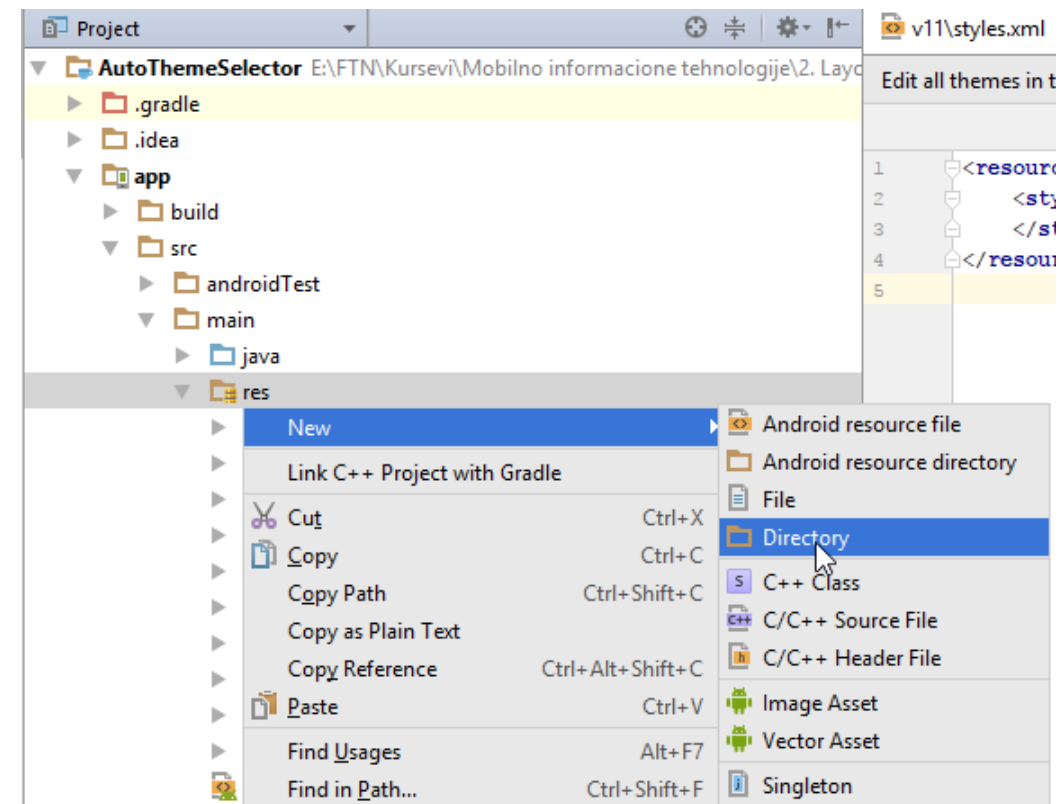
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```



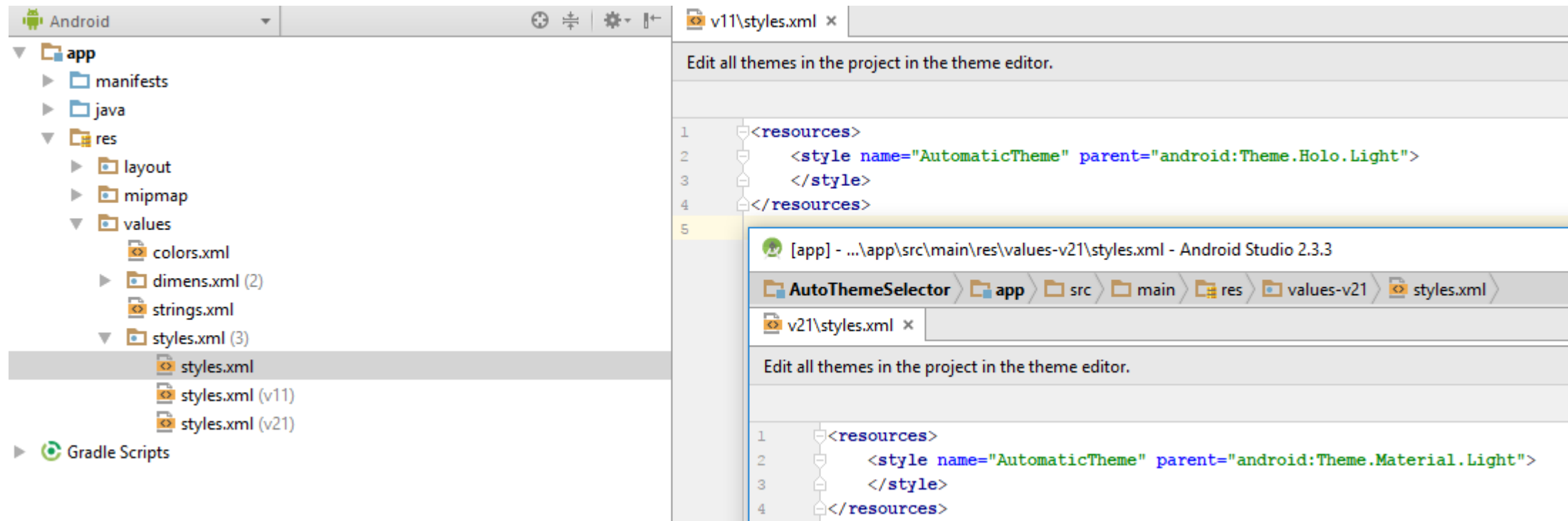
# Themes

- Primer AutoThemeSelector – selektovanje teme na osnovu Android verzije:
1. MainActivity prosiruje Activity a ne AppCompatActivity
  2. Ukloniti AppTheme iz styles.xml, dodati novu temu
  3. Kreirati dva nova direktorijuma za API 11 i 21. Project prikaz umesto Android prikaza unutar Android Studio IDE.



# Themes

- Primer AutoThemeSelector – selektovanje teme na osnovu Android verzije



The screenshot displays the Android Studio interface with the theme editor open for the `v11\styles.xml` file. The left sidebar shows the project structure with the `values` folder expanded, highlighting `styles.xml`. The main editor area shows the XML code for the theme, which is set to `android:Theme.Holo.Light`. A breadcrumb trail at the bottom indicates the path: `AutoThemeSelector > app > src > main > res > values-v21 > styles.xml`. The code editor shows the following XML structure:

```
1 <resources>
2   <style name="AutomaticTheme" parent="android:Theme.Holo.Light">
3   </style>
4 </resources>
```

The status bar at the top right shows the time as 17:19 and 100% battery.



