PT&SCHE

Introduction of part-time and short cycle studies in Serbia

Funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

UNIVERZITET METROPOLITAN BEOGRAD

KI205 - JAVA 8: PROGRAMIRANJE U JAVI NA ANDROID PLATFORMI

Primena pogleda za prikazivanje slika i menija

Lekcija 04

PRIRUČNIK ZA STUDENTE

KI205 - JAVA 8: PROGRAMIRANJE U JAVI NA ANDROID PLATFORMI

Lekcija 04

PRIMENA POGLEDA ZA PRIKAZIVANJE SLIKA I MENIJA

- Primena pogleda za prikazivanje slika i menija
- Poglavlje 1: Upotreba menija sa pogledima
- Poglavlje 2: Pogledi za prikazivanje slika
- Poglavlje 3: Dodatni pogledi
- Poglavlje 4: Domaći zadatak 4
- Zaključak

UVOD

Pogledi za prikazivanje slika i menija su ugrađeni u veliki broj Android aplikacija.

Cilj ove lekcije jeste savladavanje načina primene novih, napredniji, vrsta pogleda koji se koriste u kreiranju *ozbiljnijih* Android aplikacija. Konkretno, biće naučeno kako se primenjuju pogledi kojima se prikazuju slike na mobilnom ekranu, a zatim i načini za kreiranje kontekstnih i menija sa opcijama. Razlog za obrađivanje ove teme je veoma jednostavan, ne postoji "ozbiljna" android aplikacija u kojoj nije implementirana neka slika ili nije ugrađen meni za izbor neke od opcija. Čak, šta više, sve Android aplikacije, koje se kreiraju pod Android API koji odgovara verziji operativnog sistema Android 6.0, mogu po podrazumevanim vrednostima u MainActivity.java klasi da imaju ugrađenu metodu za kreiranje praznog menija i metodu za izbor iz menija. U daljem radu se ove metode redefinišu, meni se popuni i definišu se akcije koje se dešavaju kada se izabere neka stavka iz menija. Takođe, u lekciji će biti govora o primeni pogleda kojima je omogućeno prikazivanje trenutnog vremena i web sadržaja.

Shodno prethodno navedenom, u ovoj lekciji biće govora o:

- Primeni Gallery, ImageSwitcher, GridView i ImageView pogleda za prikazivanje slika;
- Načinima prikazivanja kontekstnih i menija sa opcijama;
- Korišćenju pogleda AnalogClock i DigitalClock;
- Prikazivanju web sadržaja pomoću WebView pogleda.

Nakon savladavanja ove lekcije studenti će biti osposobljeni da koriste naprednije poglede u vlastitim Android aplikacijama. Prikazivaće slike i koristiće ih kroz razne animacije, aktivno će koristiti menije, bilo da su oni standardni ili kontekstni, moći će na različite načine da prikazuju tekuće vreme u vlastitim android aplikacijama. Posebno, studenti će moći, primenom WebView pogleda, da integrišu na različite načine web sadržaj u Android aplikacije koje sami kreiraju. Ovom lekcijom se polako ulazi u naprednije Android koncepte i poglede.

Poglavlje 1

Upotreba menija sa pogledima

UVOD U MENIJE

Menijem je omogućeno prikazivanje sadržaja koji nije direktno vidljiv u UI.

Meni, u svakoj ozbiljnoj Android aplikaciji, predstavlja nezaobilazan alat. Moguće ga je sresti u različitim oblicima i sa različitima načinima prikazivanja vlastitih stavki. Primenom menija povećava se preglednost glavne aktivnosti i na istom prostoru korisniku se nudi veći broj dostupnih opcija.

Meniji su kontrole korisničkog interfejsa kojima je omogućeno prikazivanje sadržaja koji nisu direktno vidljivi u glavnom korisničkom interfejsu Android aplikacije. Android operativni sistem podržava dva tipa menija:

- Meniji sa opcijama – Ovi meniji prikazuju informacije koje se odnose na tekuću aktivnost. Meni se aktivira klikom na taster za meni;
- Kontekstni meniji – Meniji nose informacije koji se odnose na konkretan pogled aktivnosti. Da bi meni bio aktiviran, neophodno je da se klikne na određeni deo ekrana, odnosno odgovarajući pogled, i da se zadrži na njemu, u smislu dugačkog klika. Nakon ovakve intervencije korisnika na glavnu aktivnost, otvara se meni sa odgovarajućim opcijama.

Sledećim video materijalom je dat uvod u menije sa opcijama.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Sledećim video materijalom je dat uvod u kontekstne menije.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRIMENA POMOĆNIH METODA

Pre rada sa menijima, neophodno je kreirati dve pomoćne metode – za prikazivanje liste stavki i upravljanje događajima nakon izbora stavke menija.

Pre nego što programer kreira meni sa opcijama, ili kontekstni meni, prinuđen je da kreira dve pomoćne metode. Zadatak ovih metoda je prikazivanje liste stavki i upravljanje događajima

nakon izbora stavke menija. Ove metode su deo klase aktivnosti projekta i u konkretnom slučaju biće nazvane CreateMenu() i MenuChoice(). Ove metode, primenom alata Android Studio IDE mogu unapred, kao prazne, biti uključene u glavnu klasu aktivnosti. Prilikom kreiranja novog projekta i izborom šablona za novu aplikaciju pod nazivom Basic Activity (sledeća slika) navedeno će biti realizovano.

Slika 1.1 Izbor šablona Basic Activity

U nastavku, klikom na opciju Next, Android Studio nudi definisanje naziva za XML GUI datoteku, glavnu klasu aktivnosti, naslov i, na samom kraju, naziv menija. Navedeno je prikazano sledećom slikom.

Slika 1.2 Imena glavnih JAVA i XML datoteka, naslova i menija

Konačno, prazne metode za rad sa menijima, dodate su u glavnu klasu aktivnosti, a to je prikazano sledećim listingom.

```
@Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.menu, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }

        return super.onOptionsItemSelected(item);
    }
```

POMOĆNE METODE - FUNKCIONISANJE

CreateMenu() preuzima argument tipa Menu, a MenuChoice argument tipa MenuItem.

Kreiranje pomoćnih metoda predstavlja polazni osnov za kreiranje aplikacije za rad sa menijima u Android operativnom sistemu.

CreateMenu() metoda preuzima argument tipa Menu i dodaje niz stavki menija. Za dodavanje stavke u meni, potrebno je kreirati objekat klase MenuItem i izvršiti metodu add() objekta Menu (pogledati priloženi kod metode). Metoda *add()* poseduje četiri argumenta:

1. ***groupID*** – identifikator grupe kojoj stavka menija pripada;
2. ***itemId*** – jedinstveni identifikator stavke grupe;
3. ***order*** – redosled u kome stavka treba da bude prikazana;
4. ***title*** – tekst koji se prikazuje za određenu stavku menija.

Metoda setAlphabeticShortcut(),u primeru (vidi priloženi kod metode), je iskorišćena za podešavanje prečice na tastaturi za izbor konkretne stavke manija. Metodom setIcon() definiše se slika koja se prikazuje kao određena stavka menija.

Metoda MenuChoice() preuzima MenuItem argument, proverava njegov Id i ispituje koja je stavka liste izabrana.
U primeru je iskorišćena Toast poruka za prikazivanje rezultata izbora stavke na ekranu mobilnog uređaja

MENI SA OPCIJAMA - PRIMER

Funkcionisanje menija sa opcijama zahteva implementaciju izvesnih specifičnih metoda.

Da bi meni sa opcijama bio prikazan u izvesnoj aktivnosti, neophodno je implementirati metode onCreateOptionsMenu() i onOptionsItemSelected(). Prva metoda se izvršava kada korisnik klikne na taster MENU. U konkretnom primeru, pomoćna metoda onCreateMenu() izvršava se za prikazivanje menija sa opcijama. Nakon izbora, stavke iz menija, izvršava se metoda onOptionsItemSelected(), koja implementira metodu menuChoice() za prikazivanje izabrane stavke i sprovođenje predviđene akcije.

Za kreiranje aplikacija sa menijima, neophodno je uključivanje izvesnog broja paketa sa odgovarajućim klasama u klasu aktivnosti projekta. Sledećim listingom date su klase za podršku rada sa menijima.

```
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
```

Sledećim kodom, izolovanim iz glavne klase, date su redefinisane početne prazne metode za rad sa menijima u Android aplikaciji. Metode su proširene kodom u skladu sa prethodno navedenim zahtevima.

```
@Override
    public boolean onOptionsItemSelected(MenuItem item){
        return MenuChoice(item);
    }
    private void CreateMenu(Menu menu){
        menu.setQwertyMode(true);
        MenuItem mnu1 = menu.add(0, 0, 0, "Stavka 1");{

            mnu1.setAlphabeticShortcut('a');
            mnu1.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu2 = menu.add(0, 1, 1, "Stavka 2");{
            mnu2.setAlphabeticShortcut('b');
            mnu2.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu3 = menu.add(0, 2, 2, "Stavka 3");{
            mnu3.setAlphabeticShortcut('c');
            mnu3.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu4 = menu.add(0, 3, 3, "Stavka 4");{
            mnu4.setAlphabeticShortcut('d');
        }
        menu.add(0, 4, 4, "Stavka 5");
        menu.add(0, 5, 5, "Stavka 6");
        menu.add(0, 6, 6, "Stavka 7");
    }
    private boolean MenuChoice(MenuItem item){
        switch (item.getItemId()) {
            case 0:
                Toast.makeText(this, "Kliknuli ste na stavku  1",
                        Toast.LENGTH_LONG).show();
                return true;
            case 1:
                Toast.makeText(this, "Kliknuli ste na stavku  2",
                        Toast.LENGTH_LONG).show();
                return true;
            case 2:
                Toast.makeText(this, "Kliknuli ste na stavku  3",
                        Toast.LENGTH_LONG).show();
                return true;
            case 3:
                Toast.makeText(this, "Kliknuli ste na stavku  4",
                        Toast.LENGTH_LONG).show();
                return true;
            case 4:
                Toast.makeText(this, "Kliknuli ste na stavku  5",
                        Toast.LENGTH_LONG).show();
                return true;
            case 5:
                Toast.makeText(this, "Kliknuli ste na stavku  6",
                        Toast.LENGTH_LONG).show();
                return true;
            case 6:
```

```
                Toast.makeText(this, "Kliknuli ste na stavku  7",
                        Toast.LENGTH_LONG).show();
                return true;
            }
            return false;
        }
```

KONTEKSTNI MENI - PRIMER

Kontekstni meni je povezan sa pogledom definisanim za određenu aktivnost.

Pored menija sa opcijama, u Android aplikacijama, moguće je koristiti i kontekstne menije. Kontekstni meni je povezan sa pogledom definisanim za određenu aktivnost i prikazuje se kada korisnik drži pritisnutu određenu stavku (pogled). Na primeru će biti pokazano prikazivanje kontekstnog menija u situaciji kada korisnik par sekundi vrši pritisak na dugme. Podrška za kreiranje menija dugim klikom na dugme je ugrađena u onCreate() metodu glavne klase aktivnosti (videti dole priloženi kod).

Za prikazivanje kontekstnog menija, neophodno je izvršavanje metodesetOnCreateContextMenuListener() za *Button* pogled. Nakon klika, i zadržavanja dugmeta, poziva se metoda onCreateContextMenu() koja izvršava jednu drugu metodu, createMenu(), za prikazivanje kontekstnog menija.

Metode za selektovanje stavki menija i prikazivanje rezultata izbora, funkcionišu analogno meniju sa opcijama.

Još bi trebalo napomenuti da je moguće izvršiti metodu setQwertyMode()da bi definisane prečice sa tastature, ukoliko postoje, bile funkcionalne.

Za rad sa kontekstnim menijem, u klasu aktivnosti aplikacije, neophodno je ugraditi i sledeći kod.

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

        Button btn = (Button) findViewById(R.id.button1);
        btn.setOnCreateContextMenuListener(this);
    }

    @Override
    public void onCreateContextMenu(ContextMenu menu, View view,
                                    ContextMenuInfo menuInfo)
    {
        super.onCreateContextMenu(menu, view, menuInfo);
        CreateMenu(menu);
    }
```

```
    @Override
    public boolean onContextItemSelected(MenuItem item){
        return MenuChoice(item);
    }
```

MENIJI DEMONSTRACIJA

Klikom na odgovarajući element ekrana, pokreće se konkretan meni.

Prevođenjem i pokretanjem programa, emulatorom ili mobilnim uređajem, izborom opcije Runn app (ili Shift + F10) u Android Studio IDE razvojnom okruženju, prikazuje se početni korisnički interfejs. Klikom na dugmeMENU, u gornjem desnom uglu ekrana, otvara se meni sa opcijama iz kojeg biramo stavke i pratimo akcije (sledeća slika).

Slika 1.3 Meni sa opcijama

Klikom i zadržavanjem dugmeta Klikni i zadrži otvara se kontekstni meni zajedno sa svojim opcijama. Klikom na izabranu opciju kontekstog menija dešava se definisana programska akcija (sledeća slika).

Slika 1.4 Kontekstni meni

PRIMER 1 - VEŽBANJE KREIRANJA MENIJA

Praktično zaokruživanje teorijskog znanja o menijima iz ove lekcije.

Zadatak:

Kreirati Android aplikaciju koja sadrži dva menija: standardni i kontekstni. Kontekstni meni se otvara dugim klikom na kontrolu dugme. Meni sa opcijama i kontekstni meni sadrže kao stavke predmete Fakulteta informacionih tehnologija. Izborom predmeta dobija se odgovarajuća povratna informacija Toast klasom. Definisati prečice sa tastature za pristup stavkama za nekoliko predmeta. Postaviti logo Univerziteta na pozadinu i postaviti AnalogClock i DigitalClock poglede na aktivnost za prikazivanje trenutnog vremena.

Sledećim kodom je određen GUI:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
android:background="@drawable/metpozadina">

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
```

```
    android:text="Primer rada sa menijima" />

<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Klikni i zadrži!!!" />

<AnalogClock
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
<DigitalClock
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />

</LinearLayout>
```

Klasa aktivnosti, kreirana po definisanim zahtevima, data je sledećim kodom:

```
package com.metropolitan.menidemo;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;



public class MainActivity extends AppCompatActivity {

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

        Button btn = (Button) findViewById(R.id.button1);
        btn.setOnCreateContextMenuListener(this);
    }

    @Override
    public void onCreateContextMenu(ContextMenu menu, View view,
                                    ContextMenuInfo menuInfo)
    {
        super.onCreateContextMenu(menu, view, menuInfo);
        CreateMenu(menu);
    }
```

```
    @Override
    public boolean onContextItemSelected(MenuItem item){
        return MenuChoice(item);
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        super.onCreateOptionsMenu(menu);
        CreateMenu(menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item){
        return MenuChoice(item);
    }
    private void CreateMenu(Menu menu){
        menu.setQwertyMode(true);
        MenuItem mnu1 = menu.add(0, 0, 0, "CS101");{

            mnu1.setAlphabeticShortcut('a');
            mnu1.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu2 = menu.add(0, 1, 1, "CS102");{
            mnu2.setAlphabeticShortcut('b');
            mnu2.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu3 = menu.add(0, 2, 2, "CS330");{
            mnu3.setAlphabeticShortcut('c');
            mnu3.setIcon(R.mipmap.ic_launcher);
        }
        MenuItem mnu4 = menu.add(0, 3, 3, "IT355");{
            mnu4.setAlphabeticShortcut('d');
        }
        menu.add(0, 4, 4, "IT205");
        menu.add(0, 5, 5, "MAT101");
        menu.add(0, 6, 6, "CS103");
    }
    private boolean MenuChoice(MenuItem item){
        switch (item.getItemId()) {
            case 0:
                Toast.makeText(this, "Izabrali ste predmet CS101",
                        Toast.LENGTH_LONG).show();
                return true;
            case 1:
                Toast.makeText(this, "Izabrali ste predmet CS102",
                        Toast.LENGTH_LONG).show();
                return true;
            case 2:
                Toast.makeText(this, "Izabrali ste predmet CS330",
                        Toast.LENGTH_LONG).show();
                return true;
            case 3:
                Toast.makeText(this, "Izabrali ste predmet IT355",
                        Toast.LENGTH_LONG).show();
```

```
                return true;
            case 4:
                Toast.makeText(this, "Izabrali ste predmet IT205",
                        Toast.LENGTH_LONG).show();
                return true;
            case 5:
                Toast.makeText(this, "Izabrali ste predmet MAT101",
                        Toast.LENGTH_LONG).show();
                return true;
            case 6:
                Toast.makeText(this, "Izabrali ste predmet CS103",
                        Toast.LENGTH_LONG).show();
                return true;
        }
        return false;
    }

}
```

Izgled ekrana aplikacije bi trebalo da izgleda na sledeći način:

Slika 1.5 Aplikacija sa menijem i pogledima za vreme

ZADATAK 1 - KREIRAJTE MENIJE U VAŠOJ ANDROID APLIKACIJI

Pokušajte sami

- Kreirajte vlastitu Android aplikaciju;
- Kreirajte dva menija: sa opcijama i kontekstni;
- Oba menija moraju da budu funkcionalni;
- Izborom neke stavke iz menija, korisnik dobija informaciju koja je stavka izabrana.

Poglavlje 2

Pogledi za prikazivanje slika

GALLERY I IMAGEVIEW POGLEDI

Gallery pogled predstavlja slike u formi horizontalne skrol liste, a ImageView prikazuje sliku koju je izabrao krosnik.

Kao što je već napomenuto, Android okvir predstavlja specifičnu implementaciju JAVA tehnologija namenjenu kreiranju aplikacija za mobilne uređaje. Kao i sve JAVA tehnologije Android poseduje bogatu podršku za rad sa slikama. Za Android aplikacije, kao i sve ostale JAVA aplikacije, važi da se slika čuva u Image objektu za čije prikazivanje je zadužen objekat tipa ImageView. Android podržava još neke klase za rad sa slikama.

Gallery predstavlja horizontalnu skrolujuću listu čije elemente predstavljaju slike. Ako iz liste korisnik selektuje neku sliku, ona će na ekranu biti prikazana pomoću pogleda *ImageView*. Da bi skup slika, uopšte, mogao da bude prikazan listom, on mora da bude sačuvan u posebnom folderu pod nazivom *res/drawable-*.* Džoker znak * može imati neku od sledećih vrednosti: *hdpi* (visoka rezolucija), *ldpi* (niska rezolucija), *mdpi* (srednja rezolucija) i *xhdpi* (ekstra visoka rezolucija). U konkretnom slučaju slike će biti sačuvane u opštemUrehiji folderu *res/drawable*.

Hijerarhija foldera projekta, sa folderom u kojem su sačuvane slike, koje će biti smeštene u galeriju, sa daljom mogućnošću njihovog prikazivanja uImageView objektu, prikazana je sledećom slikom. U hijerarhiji projekta, moguće je primetiti datoteku attrs.xml u kojoj se čuvaju podaci vezani za stil prikazivanja Gallery objekta (pogledati kod klase aktivnosti).

Slika 2.1 Slike za galeriju i datoteka sa stilom

GALLERY I IMAGEVIEW POGLEDI – XML DATOTEKE

Pored glavne XML datoteke, za definiciju UI, biće upotrebljena još jedna za deklarisanje stilova.

U nastavku bi trebalo napomenuti da se pogledi za rad sa slikama po načinu uključivanja u korisnički interfejs ni po čemu ne razlikuju od ostalih Android pogleda i grupa pogleda. Gallery i ImageViewpogledi biće definisani, kao deo Android korisničkog interfejsa, pomoću datoteke activity_main.xml, oggovarajućim XML elementima. Kod kojim će, na ovom nivou, pogledi biti realizovani, prikazan je sledećim listingom.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@drawable/metpozadina">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Slike Univerziteta Metropolitan" />

    <Gallery
        android:id="@+id/gallery1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />

    <ImageView
        android:id="@+id/image1"
        android:layout_width="320dp"
        android:layout_height="250dp"
        android:scaleType="fitXY" />

</LinearLayout>
```

Još jednu, pomoćnu, XML datoteku moguće je uvesti u projekat, a sa ciljem definisanja stilova prikaza. Datoteka će biti kreirana u folderu res/values, pored standardne strings.xml datoteke, i dobiće naziv attrs.xml. Kod ove datoteke je dat u formi sledećeg listinga.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="Gallery1">
        <attr name="android:galleryItemBackground" />
    </declare-styleable>
</resources>
```

Pozicija ove datoteke, u hijerarhiji projekta, je već pokazana slikom 1 ovog objekta učenja.

Korišćenje ImageView pogleda pokazano je sledećim video materijalom.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

JAVA KLASA AKTIVNOSTI ZA GALLERY I IMAGEVIEW POGLEDE

Funkcionalnost aplikacije biće realizovana JAVA klasom MainActivity.java.

Za integrisanje funkcionalnosti koje podržavaju rada sa slikama u nekoj Android aplikaciji, neophodno je dobro organizovati i kodirati glavnu klasu aktivnosti.

Funkcionalnost aplikacije biće realizovana JAVA klasom MainActivity.java. Klasa će upravljati odgovarajućim pogledima sa prikazivanje liste slika (Gallery), kaoi pojedinačih slika (ImageVew), a koristiće i *Toast* klasu za davanje informacija na ekranu o izabranoj slici iz liste. Kod MainActivity.java klase prikazan je sledećim listingom.

```
package com.metropolitan.galerijademo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Context;
import android.content.res.TypedArray;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.Gallery;
import android.widget.ImageView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    //---slike za prikazovanje---
    Integer[] imageIDs = {
            R.drawable.pic1,
            R.drawable.pic2,
            R.drawable.pic3,
            R.drawable.pic4,
            R.drawable.pic5,
            R.drawable.pic6,
            R.drawable.pic7
    };
    /** Kreira se kada se kreira aktivnost. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Gallery gallery = (Gallery) findViewById(R.id.gallery1);
        gallery.setAdapter(new ImageAdapter(this));
        gallery.setOnItemClickListener(new OnItemClickListener()
        {
            public void onItemClick(AdapterView<?> parent, View v,
                                    int position, long id)
            {
                Toast.makeText(getBaseContext(),
                        "Slika " + (position + 1) + " je izabrana",
                        Toast.LENGTH_SHORT).show();
                //---prikazuje izabrane slike---
                ImageView imageView = (ImageView) findViewById(R.id.image1);
                imageView.setImageResource(imageIDs[position]);
            }
        });
    }
```

```
    public class ImageAdapter extends BaseAdapter
    {
        Context context;
        int itemBackground;
        public ImageAdapter(Context c)
        {
            context = c;
            //---podešava stil---
            TypedArray a = obtainStyledAttributes(R.styleable.Gallery1);

            itemBackground = a.getResourceId(
                    R.styleable.Gallery1_android_galleryItemBackground, 0);
            a.recycle();
        }
        //---vraća broj slika---
        public int getCount() {
            return imageIDs.length;
        }
        //---vraća stavku---
        public Object getItem(int position) {
            return position;
        }
        //---vraća ID of stavke---
        public long getItemId(int position) {
            return position;
        }
        //---vraća ImageView pogled---
        public View getView(int position, View convertView, ViewGroup parent) {
            ImageView imageView;
            if (convertView == null) {
                imageView = new ImageView(context);
                imageView.setImageResource(imageIDs[position]);
                imageView.setScaleType(ImageView.ScaleType.FIT_XY);
                imageView.setLayoutParams(new Gallery.LayoutParams(150, 120));
            } else {
                imageView = (ImageView) convertView;
            }
            imageView.setBackgroundResource(itemBackground);
            return imageView;
        }
    }

}
```

GALLERY I IMAGEVIEW POGLEDI – FUNKCIONISANJE

Neophodno je kreirati ImageAdapter klasu koja povezuje Gallery pogled sa serijom ImageView pogleda.

Kada je projekat kreiran, prva aktivnost je bila dodavanje definicije Gallery i ImageViewpogleda u activity_main.xml datoteku, a to se može videti iz priloženog koda ove datoteke.

Lista slika, koja će biti prikazana, spakovana je u niz pod nazivom imageID(obratiti pažnju na dole priloženi deo koda).

Zatim je bilo neophodno kreirati i ImageAdapter klasu, naslednicu BaseAdapterklase, koja povezuje Gallery pogled sa serijom ImageView pogleda. *BaseAdapter* klasa povezuje *AdapterView* pogled sa izvorom podataka koji se prikazuju (videti kod priložen datotekom *MainActivity.java*).

```
//---slike za prikazivanje---
    Integer[] imageIDs = {
            R.drawable.pic1,
            R.drawable.pic2,
            R.drawable.pic3,
            R.drawable.pic4,
            R.drawable.pic5,
            R.drawable.pic6,
            R.drawable.pic7
    };
```

AdapterView pogledi su: ListView, GridView, Spinner i Gallery. Neke od naslednica BaseAdapter klase su: ListAdapter, ArrayAdapter, CursorAdapter, SpinnerAdapter itd.

Posebno, metoda getView() ove klase, vraća View na tačno određenoj poziciji. U ovom primeru biće vraćena slika, odnosno ImageView objekat.

Slike su spakovane u niz, a to znači da se po obavljenom izboru određuje i pozicija slike koja se prikazuje *ImageView* pogledom (videti sledeći kod).

```
Gallery gallery = (Gallery) findViewById(R.id.gallery1);
        gallery.setAdapter(new ImageAdapter(this));
        gallery.setOnItemClickListener(new OnItemClickListener()
        {
            public void onItemClick(AdapterView<?> parent, View v,
                                    int position, long id)
            {
                Toast.makeText(getBaseContext(),
                        "Slika " + (position + 1) + " je izabrana",
                        Toast.LENGTH_SHORT).show();
                //---prikazuje izabrane slike---
                ImageView imageView = (ImageView) findViewById(R.id.image1);
                imageView.setImageResource(imageIDs[position]);
            }
        });
    }
```

GALLERY I IMAGEVIEW POGLEDI – DEMONSTRACIJA

Prevođenjem i pokretanjem aplikacije na ekranu se pojavljuje interfejs sa dva pogleda.

Klikom na Shift + F10 (ili Run App), aplikacija se prevodi i pokreće emulatorom. Početni ekran, pri čemu nijedna slika još nije selektovana, sadrži Gallery listu i prikazan je sledećom slikom. Glavna aktivnost očekuje akciju korisnika.

Slika 2.2 Gallery pogled u glavnoj aktivnosti

Izborom neke od slika, na ekranu se pojavljuje i ImageViewpogled koji prikazuje izabranu sliku. Takođe, klasom *Toast,* na ekranu će biti prikazan i odgovarajući komentar koji je u vezi sa izabranom slikom.

Slika 2.3 Izabrana slika u ImageView pogledu

IMAGESWITCHER POGLED

Primena animacija na slikama moguća je kombinovanjem pogleda ImageSwitcher i Gallery.

U prethodnom izlaganju demonstrirana je saradnja pogleda Gallery i ImageView. Serija umanjenih slika je prikazana, a potom je jedna od njih izabrana i prikazana u većem formatu pomoću *ImageView* pogleda.

U nekim situacijama se ne traži brzo pojavljivanje neke slike već neka specifična akcija. Ta akcija može biti neka animacija koja se odnosi na prelazak sa jedne slike na drugu. Tada je neophodno koristiti ImageSwitcher pogled u kombinaciji sa pogledom Gallery.

Prethodni primer može da se modifikuje i da kvalitetno pokaže funkcionalnosti posmatranih pogleda.

Datoteka attrs.xml, iz foldera *res/value* će ostati nepromenjena, a isto će se desiti sa slikama koje su spakovane u osnovni folder *res/drawable.*

Prva modifikacija biće učinjena main.xml datoteci zamenom XML atributa <ImageView> za <ImageSwitcher>. Navedeno je prikazano sledećim kodom.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
```

```
        android:text="Slike Univerziteta Metropolitan" />

    <Gallery
        android:id="@+id/gallery1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />


    <ImageSwitcher
        android:id="@+id/switcher1"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true" />

</LinearLayout>
```

IMAGESWITCHER POGLED – MODIFIKACIJA JAVA DATOTEKE AKTIVNOSTI

Pored nasleđivanja klase Activity, klasa koja upravlja ImageSwitcher pogledom mora da implementira interfejs ViewFactory.

Klasa aktivnosti tekućeg projekta razlikuje se od klase aktivnosti iz prethodnog projekta po implementaciji interfejsa ViewFactory. Implementacijom ovog interfejsa, klasa aktivnosti projekta dobija na korišćenje još neke metode kojima je omogućena animacija slika u glavnoj aktivnosti kreirane aplikacije. Kod klase aktivnosti i u njoj ugnježdene ImageAdapter klase nalazi se u istoj datoteci i prikazan je sledećim listingom. Moguće je primetiti da su dve linije programskog koda date kao komentari. Ovim linijama moguće je promeniti tip animacije slika u glavnoj aktivnosti. Pogledom u Android dokumentaciju, za metodu setInAnimation() objekta klase ImageSwitcer i njen argument koji odgovara pozivu metodeAnimationUtils.loadAnimation(this, android.R.anim.slide_in_left), moguće je pronaći više različitih tipova animacija za slike.

```
package com.metropolitan.imageswitcherdemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Context;
import android.content.res.TypedArray;
import android.view.View;
import android.view.ViewGroup;
import android.view.ViewGroup.LayoutParams;
import android.view.animation.AnimationUtils;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.Gallery;
```

```
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher.ViewFactory;

public class MainActivity extends AppCompatActivity implements ViewFactory {

    //---slike za prikazivanje---
    Integer[] imageIDs = {
            R.drawable.pic1,
            R.drawable.pic2,
            R.drawable.pic3,
            R.drawable.pic4,
            R.drawable.pic5,
            R.drawable.pic6,
            R.drawable.pic7
    };

        private ImageSwitcher imageSwitcher;

    /** Kreira se kada se kreira aktivnost. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        imageSwitcher = (ImageSwitcher) findViewById(R.id.switcher1);
        imageSwitcher.setFactory(this);

        /*
        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_in));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_out));
        */

        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_in_left));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_out_right));


        Gallery gallery = (Gallery) findViewById(R.id.gallery1);
        gallery.setAdapter(new ImageAdapter(this));
        gallery.setOnItemClickListener(new OnItemClickListener()
        {
            public void onItemClick(AdapterView<?> parent,
                                    View v, int position, long id)
            {
                imageSwitcher.setImageResource(imageIDs[position]);
            }
        });
    }
    public View makeView(){
```

```
        ImageView imageView = new ImageView(this);
        imageView.setBackgroundColor(0xFF000000);
        imageView.setScaleType(ImageView.ScaleType.FIT_CENTER);
        imageView.setLayoutParams(new
                ImageSwitcher.LayoutParams(
                LayoutParams.FILL_PARENT,
                LayoutParams.FILL_PARENT));
        return imageView;
    }

    public class ImageAdapter extends BaseAdapter
    {
        Context context;
        int itemBackground;
        public ImageAdapter(Context c)
        {
            context = c;
            //---podešava stil---
            TypedArray a = obtainStyledAttributes(R.styleable.Gallery1);

            itemBackground = a.getResourceId(
                    R.styleable.Gallery1_android_galleryItemBackground, 0);
            a.recycle();
        }
        //---vraća broj slika---
        public int getCount() {
            return imageIDs.length;
        }
        //---vraća stavku---
        public Object getItem(int position) {
            return position;
        }
        //---vraća ID of stavke---
        public long getItemId(int position) {
            return position;
        }
        //---vraća ImageView pogled---
        public View getView(int position, View convertView, ViewGroup parent) {
            ImageView imageView;
            if (convertView == null) {
                imageView = new ImageView(context);
                imageView.setImageResource(imageIDs[position]);
                imageView.setScaleType(ImageView.ScaleType.FIT_XY);
                imageView.setLayoutParams(new Gallery.LayoutParams(150, 120));
            } else {
                imageView = (ImageView) convertView;
            }
            imageView.setBackgroundResource(itemBackground);
            return imageView;
        }
    }

}
```

IMAGESWITCHER POGLED – NASLEĐIVANJE I PAKETI

Klasa koja rukuje ImageSwitcher pogledom koristi objekte i metode brojnih klasa.

Brojni paketi su uključeni u Android projekat koji rukuje ImageSwitcher pogledom. U ovim paketima nalaze se korisne klase čije objekte i metode koristi klasa aktivnosti aplikacije za postizanje konkretnih funkcionalnosti. Takođe, kroz nasleđivanje, klasa aktivnosti aplikacije nasleđuje funkcionalnosti bazne klaseActivity (ili AppCompatActivity za aktuelne Android API verzije). Sledećim listingom su prikazani paketi i njihove klase i interfejsi, angažovani u tekućem projektu.

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Context;
import android.content.res.TypedArray;
import android.view.View;
import android.view.ViewGroup;
import android.view.ViewGroup.LayoutParams;
import android.view.animation.AnimationUtils;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.Gallery;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher.ViewFactory;
```

IMAGESWITCHER POGLED – FUNKCIONISANJE

Metodom makeView() omogućena je upotreba ImageSwitcher pogleda.

Kao što je primećeno klasa aktivnosti aplikacije ne samo da je izvedena iz osnovne Activity klase (ili AppCompatActivity za aktuelne Android API verzije) , već i implementira ViewFactory interfejs. Ovaj interfejs definiše poglede koji se koriste uporedo sa ImageSwitch pogledom. Iz ovog razloga je neophodno implementirati makeView() metodu kojom je omogućeno umetanje novog pogleda uImageSwitch pogled, a taj pogled je, konkretno, *ImageView.* Navedena programska logika realizovana je sledećim kodom izolovanim iz glavne klase aktivnosti.

```
public View makeView(){
        ImageView imageView = new ImageView(this);
        imageView.setBackgroundColor(0xFF000000);
        imageView.setScaleType(ImageView.ScaleType.FIT_CENTER);
        imageView.setLayoutParams(new
                ImageSwitcher.LayoutParams(
                LayoutParams.FILL_PARENT,
```

```
                LayoutParams.FILL_PARENT));
        return imageView;
    }
```

U metodi onCreate(), glavne klase aktivnosti, učitava se ImageSwitcher objekat i podešava animacija. U kodu, datim sledećim listingom, vidi se animacija podešena konstantama fade_in i fade_out. Ovde je moguće dalje uvoditi novine. Ako se želi da se slika pojavi sa leve strane, a nestane sa desne, prilikom izbora naredne slike, moguće je koristiti konstante za animaciju: slide_in_left *i* slide_out_right. Takođe, animacije je moguće menjati po vlastitim željama i idejama.

```
private ImageSwitcher imageSwitcher;

    /** Kreira se kada se kreira aktivnost. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        imageSwitcher = (ImageSwitcher) findViewById(R.id.switcher1);
        imageSwitcher.setFactory(this);


        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_in));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_out));

 /*
        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_in_left));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_out_right)); */


        Gallery gallery = (Gallery) findViewById(R.id.gallery1);
        gallery.setAdapter(new ImageAdapter(this));
        gallery.setOnItemClickListener(new OnItemClickListener()
        {
            public void onItemClick(AdapterView<?> parent,
                                    View v, int position, long id)
            {
                imageSwitcher.setImageResource(imageIDs[position]);
            }
        });
    }
```

GRIDVIEW POGLED

GridView pogled omogućava prikazivanje slika u matričnoj formi sa skrol opcijom.

GridView pogled je napredni Android pogled koji omogućava prikazivanje slika u dvodimenzionalnoj skrolujućoj mreži. Često se koristi u kombinaciji sa ImageView pogledom za prikazivanje serije slika u Android aplikacijama. U savremenim Android aplikacijama, često je moguće sresti različite albume slika organizovane u GridView raspored. Za demonstraciju pakovanja slika u GridView pogled moguće je uvesti izvesne korekcije u prethodni primer.

Kao i u prošlom slučaju, datoteka attrs.xml ostaće nepromenjena, ali će zato activity_main.xml imati novu definiciju kao što je prikazano kodom sa sledeće slike. U sliku je ugrađena i reprezentacija korisničkog interfejsa na ekranu.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@drawable/metpozadina">

    <GridView
        android:id="@+id/gridview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:numColumns="auto_fit"
        android:verticalSpacing="10dp"
        android:horizontalSpacing="10dp"
        android:columnWidth="90dp"
        android:stretchMode="columnWidth"
        android:gravity="center" />

</LinearLayout>
```

Pregled korišćenja GridView pogleda kao galerije slika prikazan je sledećim video materijalom.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

GRIDVIEW POGLED – JAVA KLASA AKTIVNOSTI PRIMERA

Klasa aktivnosti GridActivity je naslednica bazne klase Activity.

I u ovom slučaju, najveće modifikacije izvedene su nad JAVA klasama, klasi aktivnosti aplikacije, koja nosi naziv MainActivity, i klasiImageAdapter koja je unutrašnja klasa glavne

klase aktivnosti.. Ovde je već moguće konstatovati da je naučeno da klasa aktivnosti aplikacije nasleđuje baznu klasu *Activity* (ili AppCompatActivity za njanovije API Android verzije), a klasaImageAdapter klasu BaseAdapter. Navedene klase sa kodom koji organizuje ovaj album u GridView raspored, koji preuzima iz XML datoteke korisničkog interfejsa, date su kodom prikazanim sledećim listingom.

```
package com.metropolitan.griddemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.GridView;
import android.widget.ImageView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    //---Slike za prikazivanje---
    Integer[] imageIDs = {
            R.mipmap.pic1,
            R.mipmap.pic2,
            R.mipmap.pic3,
            R.mipmap.pic4,
            R.mipmap.pic5,
            R.mipmap.pic6,
            R.mipmap.pic7
    };
    /** Poziva se kada se aktivnost kreira. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        GridView gridView = (GridView) findViewById(R.id.gridview);
        gridView.setAdapter(new ImageAdapter(this));

        gridView.setOnItemClickListener(new OnItemClickListener(){
            public void onItemClick(AdapterView<?> parent,
                                    View v, int position, long id){
                Toast.makeText(getBaseContext(),
                        "Slika " + (position + 1) + " je kliknuta",
                        Toast.LENGTH_SHORT).show();
            }
        });

    }

    public class ImageAdapter extends BaseAdapter{
```

```
        private Context context;

        public ImageAdapter(Context c){
            context = c;
        }

        //---vraća broj slika---
        public int getCount() {
            return imageIDs.length;
        }

        //---vraća stavku---
        public Object getItem(int position) {
            return position;
        }

        //---vraća ID stavke---
        public long getItemId(int position) {
            return position;
        }

        //---vraća ImageView pogled---
        public View getView(int position, View convertView,
                            ViewGroup parent){
            ImageView imageView;
            if (convertView == null) {
                imageView = new ImageView(context);
                imageView.setLayoutParams(new
                        GridView.LayoutParams(85, 85));
                imageView.setScaleType(
                        ImageView.ScaleType.CENTER_CROP);
                imageView.setPadding(5, 5, 5, 5);
            } else {
                imageView = (ImageView) convertView;
            }
            imageView.setImageResource(imageIDs[position]);
            return imageView;
        }
    }

}
```

GRIDVIEW POGLED – FUNKCIONISANJE

Pomoću getView() metode moguće je odrediti veličinu slike i način na koji su slike međusobno razdvojene u GridView prikazu.

I u ovom slučaju, implementirana je ImageAdapter klasa koja je povezana sa GridView pogledom na način prezentvan sledećim listingom, izolovanim iz glavne klase aktivnosti.

```
gridView.setOnItemClickListener(new OnItemClickListener(){
            public void onItemClick(AdapterView<?> parent,
                                    View v, int position, long id){
                Toast.makeText(getBaseContext(),
                        "Slika " + (position + 1) + " je kliknuta",
                        Toast.LENGTH_SHORT).show();
            }
        });
```

Kada je obavljen izbor slike, *Toast* porukom šalje se informacija o selektovanoj slici na ekran uređaja. Pomoću getView()metode moguće je odrediti veličinu slike i način na koji su slike međusobno razdvojene u GridView prikazu izborom rastojanja za svaku pojedinačnu sliku (sledeći kod).

```
//---vraća ImageView pogled---
        public View getView(int position, View convertView,
                            ViewGroup parent){
            ImageView imageView;
            if (convertView == null) {
                imageView = new ImageView(context);
                imageView.setLayoutParams(new
                        GridView.LayoutParams(85, 85));
                imageView.setScaleType(
                        ImageView.ScaleType.CENTER_CROP);
                imageView.setPadding(5, 5, 5, 5);
            } else {
                imageView = (ImageView) convertView;
            }
            imageView.setImageResource(imageIDs[position]);
            return imageView;
        }
    }
```

Prikaz GridView albuma i izbor slike iz njega, sa odgovarajućom akcijom, koja je definisana kroz kreiranu aplikaciju (glavnom klasom aktivnosti), realizovan je sledećom slikom.

Slika 2.4 GridView album i akcija izbora slike

PRIMER 2 - VEŽBANJE RADA SA SLIKAMA U ANDROID APLIKACIJI

Praktično zaokruživanje teorijskog znanja o pogledima za prikazivanje slika iz ove lekcije.

Zadatak:

Kreirati aplikaciju koja koristi pogled Gallery za čuvanje niza slika, a potom, po izboru ogovarajuće slike, nju prikazuje u nekom od pogledaImageView ili ImageSwitcher.

Kreirati nekoliko slika Univeriteta Metropolitan i njegove okoline i sačuvati ih u folderu projekta res/drawable.

Sledećim kodom je prikazana GUI XML datoteka:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@drawable/metpozadina">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Slike Univerziteta Metropolitan" />

    <Gallery
        android:id="@+id/gallery1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />


    <ImageSwitcher
        android:id="@+id/switcher1"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true" />

</LinearLayout>
```

Glavna klasa aktivnosti je data sledećim kodom:

```
package com.metropolitan.imageswitcherdemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Context;
import android.content.res.TypedArray;
import android.view.View;
import android.view.ViewGroup;
import android.view.ViewGroup.LayoutParams;
import android.view.animation.AnimationUtils;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.Gallery;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher.ViewFactory;
```

```
public class MainActivity extends AppCompatActivity implements ViewFactory {

    //---slike za prikazivanje---
    Integer[] imageIDs = {
            R.drawable.pic1,
            R.drawable.pic2,
            R.drawable.pic3,
            R.drawable.pic4,
            R.drawable.pic5,
            R.drawable.pic6,
            R.drawable.pic7
    };

        private ImageSwitcher imageSwitcher;

    /** Kreira se kada se kreira aktivnost. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        imageSwitcher = (ImageSwitcher) findViewById(R.id.switcher1);
        imageSwitcher.setFactory(this);

        /*
        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_in));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.fade_out));
        */

        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_in_left));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                android.R.anim.slide_out_right));


        Gallery gallery = (Gallery) findViewById(R.id.gallery1);
        gallery.setAdapter(new ImageAdapter(this));
        gallery.setOnItemClickListener(new OnItemClickListener()
        {
            public void onItemClick(AdapterView<?> parent,
                                    View v, int position, long id)
            {
                imageSwitcher.setImageResource(imageIDs[position]);
            }
        });
    }
    public View makeView(){
        ImageView imageView = new ImageView(this);
        imageView.setBackgroundColor(0xFF000000);
        imageView.setScaleType(ImageView.ScaleType.FIT_CENTER);
        imageView.setLayoutParams(new
```

```
                ImageSwitcher.LayoutParams(
                LayoutParams.FILL_PARENT,
                LayoutParams.FILL_PARENT));
        return imageView;
    }

    public class ImageAdapter extends BaseAdapter
    {
        Context context;
        int itemBackground;
        public ImageAdapter(Context c)
        {
            context = c;
            //---podešava stil---
            TypedArray a = obtainStyledAttributes(R.styleable.Gallery1);

            itemBackground = a.getResourceId(
                    R.styleable.Gallery1_android_galleryItemBackground, 0);
            a.recycle();
        }
        //---vraća broj slika---
        public int getCount() {
            return imageIDs.length;
        }
        //---vraća stavku---
        public Object getItem(int position) {
            return position;
        }
        //---vraća ID of stavke---
        public long getItemId(int position) {
            return position;
        }
        //---vraća ImageView pogled---
        public View getView(int position, View convertView, ViewGroup parent) {
            ImageView imageView;
            if (convertView == null) {
                imageView = new ImageView(context);
                imageView.setImageResource(imageIDs[position]);
                imageView.setScaleType(ImageView.ScaleType.FIT_XY);
                imageView.setLayoutParams(new Gallery.LayoutParams(150, 120));
            } else {
                imageView = (ImageView) convertView;
            }
            imageView.setBackgroundResource(itemBackground);
            return imageView;
        }
    }

}
```

Primer izgleda aplikacije bi mogao biti sledeći:

Slika 2.5 Izgled urađene aplikacije

ZADATAK 2 - NAPRAVITE VLASTITU APLIKACIJU ALBUM

Pokušajte sami

Kreirajte novu, ili modifikujte postojeću Android aplikaciju:

- dodajte vlastite slike kao resurse;
- omogućite u galeriji prikaz vaših slika;
- omogućite da se prikaže u prvom planu slika koja je izabrana iz galerije.

Poglavlje 3

Dodatni pogledi

ANALOGCLOCK I DIGITALCLOCK POGLEDI

AnalogClock i DigitalClock pogledi omogućavaju prikazivanje trenutnog vremena na ekranu.

Android SDK, pored standardnih pogleda koji odgovaraju kontrolama korisničkog interfejsa, sadrži i dodatne poglede koji daju korisničkom interfejsu jedan lepši i bogatiji izgled. Veoma često se sreću Android aplikacije koje na različite načine prikazuju trenutno vreme u svojim glavnim aktivnostima ili njihovim fragmentima. Trenutno vreme može biti prikazano u formi analognog ili digitalnog časovnika i u daljem izlaganju će upravo biti govora o mehanizmima koji omogućavaju realizovanje navedenih funkcionalnosti.

Prvi pogled koji će biti razmatran je AnalogClock. Ovaj pogled omogućava prikazivanje trenutnog vremena, na ekranu mobilnog uređaja, odnosno u njegovoj glavnoj aktivnosti, u formi analognog časovnika sa kazaljkama za sate i minute. DigitalClock pogled prikazuje trenutno vreme u digitalnom formatu. Oba pogleda koriste sistemsko vreme preuzeto iz podešavanja mobilnog uređaja.

Za demonstraciju primera upotrebe ova dva pogleda, moguće je Kreirati nov primer čiji će GUI biti kreiran u svetlu. U activity_main.xml datoteku biće dodata dva nova elementa koji će na dva gore prethodno opisana načina prikazivati vreme u glavnoj aktivnosti aplikacije (videti sledeći kod).

Izmene u programu će biti snimljene, a potom će iz Android Studio okruženja, klikom na Run app ili (Shift + F10) aplikacija biti testirana u izabranom android emulatoru. Rezultat pokretanja programa prikazan je sledećom slikom.

```
<AnalogClock
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
<DigitalClock
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
```

Slika 3.1 AnalogClock i DigitalClock pogledi

UPOTREBA WEBVIEW POGLEDA

WebView pogled omogućava učitavanje web čitača u aktivnost.

Veoma često, u savremenim Android aplikacijama, sreće se slučaj da aplikacija u svojoj glavnoj aktivnosti učitava neki web sadržaj. Ukoliko Android aplikacija mora da ugradi specijalizovan sadržaj, poput web-a, u neku aktivnost, neophodno je da koristi poseban tip pogleda. Posebno koristan pogled, integrisan u Android SDK, jesteWebView pogled koji omogućava učitavanje web čitača u aktivnost i prikazivanje željenog web sadržaja.

Primena ovog pogleda biće, na najbolji način, demonstrirana odgovarajućim primerom. U Android Studio okruženju biće kreiran novi projekat za demonstriranje funkcionalnosti WebView pogleda, a nakon toga će se pristupiti definisanju GUI XML datoteke i odgovarajuće klase aktivnosti koja učitava korisnički interfejs iz ove datoteke.

Kod activity_main.xml datoteke prikazan je sledećim listingom.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

<WebView android:id="@+id/webview1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

</LinearLayout>
```

Da bi aktivnost bila u mogućnosti da učitava web sadržaj, neophodno je u AndroidManifest.xml datoteku dodati privilegije za pristup Internetu, koje mogu biti:

- *<uses-permission android:name="android.permission.INTERNET"/>* - za dozvolu pristupa Internetu;
- *<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>* - za proveru stanja / dostupnosti mreže;
- *<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>* - za pristup bežičnoj mreži.

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

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

```
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

UPOTREBA WEBVIEW POGLEDA – JAVA KLASA AKTIVNOSTI

WebView objektom vrši se učitavanje željenog url.

JAVA klasa aktivnosti implementira metode neophodne za učitavanje web stranica primenom WebView pogleda. Konkretno, takav zadatak može da se poveri metodi *loadUrl*() koju poziva objekat tipa WebView..Za korišćenje kontrole zumiranja, potrebno je učitati osobinu tipa WebSettings iz WebView pogleda, a zatim izvršiti metodu setBuiltInZoomControls(). Pre svega navedenog, neophodno je kreirati *WebView* objekat. Kod koji opisuje navedene akcije, dat je sledećim listingom

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

        WebView wv = (WebView) findViewById(R.id.webview1);

        WebSettings webSettings = wv.getSettings();
        webSettings.setBuiltInZoomControls(true);

        wv.loadUrl("https://www.facebook.com" +
                "/UniverzitetMetropolitan/");


    }
```

Ponekad se dešava da prilikom učitavanja stranice, koja vrši preusmeravanje na drugu web stranicu,WebView pogled inicira pokretanje Browser aplikacije, ugrađene u Android uređaj, sa ciljem učitavanja web sadržaja. Da bi se to sprečilo i omogućilo učitavanje sadržaja u okviru tekuće aktivnosti i WebWiew pogleda, neophodno je implementirati WebViewClient klasu i redefinisati metodu shouldOverrideUrlLoading(). Ova metoda će vratiti vrednost***false***. Navedena klasa je prikazana sledećim kodom.

```
private class Callback extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            return(false);
        }
    }
```

Sada je neophodno dodatno redefinisati prethodno prikazanu metodu onCreate() glavne klase aktivnosti.

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

        WebView wv = (WebView) findViewById(R.id.webview1);

        WebSettings webSettings = wv.getSettings();
        webSettings.setBuiltInZoomControls(true);

        wv.setWebViewClient(new Callback());
        wv.loadUrl("https://www.facebook.com" +
                "/UniverzitetMetropolitan/");


    }
```

UČITAVANJE WEB SADRŽAJA IZ DATOTEKE

Umesto http://... linka WebView može da prikaže sadržaj iniciran iz neke datoteke.

Ako postoji HTML datoteka, sačuvana u folderu assets projekta, koji je kreiran uroot folderu app, ona može biti učitana u WebView, primenom metodeloadURL(). Lokacija HTML datoteke, prikazana je sledećom slikom.

Slika 3.2 Lokacija assets foldera i HTML datoteke

HTML kod, datoteke Index.html iz assets foldera, dat je sledećim listingom.

```
<H1>Jednostavna HTML stranica</H1>
<body>
    <p>Prikaz malog web sadržaja!!!</p>
    <img src="http://master.rs/wp-content/uploads/2010/02/
logo_bordo_pozadina-copy1.jpg" />
</body>
```

Za realizovanje navedene funkcionalnosti, posmatrane Android aplikacije, bilo je neophodno izvršiti i određene korekcije u klasi aktivnosti. Klasa aktivnosti koja omogućava učitavanje HTML dokumenta u WebView pogled prikazana je sledećim listingom.

```
package com.metropolitan.webviewdemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {

    /**Poziva se kada se kreira aktivnosti. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        WebView wv = (WebView) findViewById(R.id.webview1);

        WebSettings webSettings = wv.getSettings();
        webSettings.setBuiltInZoomControls(true);

        wv.loadUrl("file:///android_asset/Index.html");
    }

    private class Callback extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            return(false);
        }
    }

}
```

HTML string je moguće i dinamički kreirati, u JAVA klasi aktivnosti projekta, a zatim ga učitati u WebView upotrebom metode loadDataWithBaseUrl(). Navedeno je prikazano sledećim kodom kojek je moguće ugraditi u posmatranu klasu aktivnosti.

```
final String mimeType = "text/html";
        final String encoding = "UTF-8";
        String html = "<H1>Jednostavna HTML stranica</H1>" +
        "<body>" + "<p>Prikaz malog web sadržaja!!!</p>" +
        "<img src=\"http://master.rs/wp-content/uploads/2010" +
        "/02/logo_bordo_pozadina-copy1.jpg\" />" +
         " </body>";
        wv.loadDataWithBaseURL("", html, mimeType, encoding, "");
```

UPOTREBA WEBVIEW POGLEDA - DEMONSTRACIJA

Prevođenjem programa, za tri različita slučaja učitavanja sadržaja u WebView, vrši se demonstracija postignutih rezultata

Prevođenjem programa, za tri različita slučaja učitavanja sadržaja u WebView, vrši se demonstracija postignutih rezultata. Iz razvojnog okruženja Android Studio IDE, stavljanjem pod komentare kod koji će kasnije biti testiran, klikom na Run app (ili Shift + F10) vrši se debagovanje i počinje testiranje kreiranog koda posmatrane Android aplikacije.

Prvi slučaj, koji je razmatran, jeste učitavanje zadatog http://.. linka metodom loadURL() iz klase aktivnosti projekta. Pozvani link odgovara *FaceBook* prezentaciji *Univerziteta Metropolitan u* Beogradu. Navedeno je prikazano sledećom slikom.

Slika 3.3 Učitavanje zadatog linka u WebView

Sledeća dva slučaja podrazumevaju učitavanje HTML dokumenta u *WebView*. Prvi način je statički i realizovan tako što je u folderu assets kreiran HTML dokument koji je učitan loadURL() metodom. U drugom slučaju,na dinamički način HTML dokument je predat kao string odgovarajućoj metodi u klasi aktivnosti projekta. Učitavanje HTML-a u WebView pogled prikazano je sledećom slikom.

Slika 3.4 HTML dokument učitan u WebView

DODATNI POGLEDI - VIDEO MATERIJALI

Pregled video materijala za poglede AnalogClock, DigitalClock i WebView

Objekat učenja Dodatni pogledi biće zaokružen prikazivanjem odgovarajućih video materijala koji pokrivaju problematiku pogleda koji su obrađeni u ovom objektu učenja.

Prvim video materijalom biće prezentovano kreiranje i korišćenje pogleda AnalogClock i DigitalClock

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

U daljem izlaganju, akcenat je na pokazivanju video materijala u kojima je obrađena primena WebView pogleda u Android aplikacijama.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Za kraj ovog objekta učenja, biće priložen još jedan video materijal koji obrađuje kreiranje i primenu WebView pogleda.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

PRIMER 3 - VEŽBANJE KORIŠĆENJA WEBVIEW POGLEDA

Praktično zaokruživanje teorijskog zanja o WebView pogledima iz ove lekcije.

Zadatak:

Kreirati Android aplikaciju koja u glavnu aktivnost učitava WebView pogled iz datoteke activity_main.xml. U WebView pogled učitati web stranicu Univerziteta Metropolitan. Kao pozadinu aktivnosti postaviti logo Univerziteta Metropolitan. Realizovati ugnježdenu klasu kojom se sprečava učitavanje ugrađenog web čitača za realizovanje ove aktivnosti. U Android Manifest datoteci podesiti dozvole pristupa Internetu.

Sledećim kodom data je XML GUI datoteka:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@drawable/metpozadina">

    <WebView android:id="@+id/webview1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Klasa aktivnosti data je sledećim kodom:

```
package com.metropolitan.webviewdemo;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        WebView wv = (WebView) findViewById(R.id.webview1);

        WebSettings webSettings = wv.getSettings();
        webSettings.setBuiltInZoomControls(true);

        wv.setWebViewClient(new Callback());
        wv.loadUrl("https://www.metropolitan.ac.rs");

    private class Callback extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            return(false);
        }
    }

}
```

Privilegije koje bi trebalo ugraditi u AndroidManifest.xml su sledeće:

```
<uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
```

Izgled ekrana urađene aplikacije trebalo bi da bude sledeći:

Slika 3.5 Ekran urađene aplikacije

ZADATAK 3 - KREIRAJTE SLOŽENU APLIAKCIJU SA KONTOLAMA IZ OVE LEKCIJE

Pokušajte sami!!!

Pokušajte da kreirate aplikaciju koja sadrži meni iz kojeg se bira reprodukcija slika iz galerije putem ImageView ili ImageSwither-a. Treća stavka menije neka bude WebView u koji će biti učitan veb tutorijal sa temama direktno vezanim za ovu lekciju.

Poglavlje 4

Domaći zadatak 4

ZADATAK 1

Cilj zadatka je vežbanje kreiranja i upravljanja menijima, korišćenje posebnih i pogleda za slike.

Kreirati Android aplikaciju sa sledećim karakteristikama:

- Izabrati raspored korisničkog interfejsa po slobodnom izboru;
- Dodati analalogni sat na korisnički interfejs;
- Kreirati meni sa opcijama koji će sadržavati sledeće stavke: *Univerzitet, FIT, FAM, FDU, Sledeće*;
- Klikom na stavku Univerzitet, kreira se *WebView* u kome se nalazi sajt Univerziteta; klikom na *FIT, FAM* ili *FDU* u *VebView* pogledu će biti spakovana web prezentacija odgovarajućeg fakulteta;
- Klikom na stavku *Sledeće* otvara se novi prozor sa rasporedom korisničkog interfejsa po slobodnom izboru;
- U novom prozoru postaviti digitalni časovnik i *ImageView* pogled sa logotipom našeg Univerziteta;
- Klikom i zadržavanjem slike, otvara se kontekstni meni sa ponuđenim izborom između pet predmeta;
- Klikom na stavku, na ekranu se pojavljuje poruka da ste izabrali vaš omiljeni predmet.

Zaključak

PREGLED LEKCIJE 4

U lekciji su prikazani načini prikazivanja i korišćenja specifičnih i korisnih pogleda.

U ovoj lekciji naučeno je kako se primenjuju pogledi na prikazivanje menija. Android operativni sistem podržava prikazivanje stavki u formi menija sa opcijama ili kontekstnih menija. Meni sa opcijama se poziva klikom na dugme menija, a kontekstni meni koristi neki specifičan pogled za vlastito aktiviranjem, klikom i zadržavanjem klika. Ne postoji Android aplikacija, ako se zanemare jednostavne aplikacije koje su proizvod učenja Android programiranja, koje ne sadrže neki od menija pomoću kojih korisnici biraju neku od ponuđenih opcija, a to je, pak, praćeno izvršavanjem neke programske akcije.

Nakon elaboriranja menija, akcenat je stavljen na specifične poglede za prikazivanje trenutnog vremena na ekranu mobilnog uređaja. Vreme može biti prikazano u analognom ili digitalnom formatu. Takođe, brojne Android aplikacije upravo koriste ove poglede za prikazivanje trenutnog vremena korisnicima.

Na kraju, prikazana su tri načina primene *WebView* pogleda za prikazivanje web sadržaja u nekoj aktivnosti. Na prvi način u WebView pogled je učitan web sadržaj na osnovu unapred zadatog URL - a. Nakon toga pokazano je kako se mogu u WebView pogled učitavati i HTML dokumenti. U prvom slučaju, kreiran je poseban HTML dokument, pod nazivom Index.html i koji je snimljen u podfolder projekta pod nazivom assets. U nastavku je sadržaj ovog dokumenta učitan u WebView i prikazan je odgovarajući sadržaj. U drugom slučaju, HTML kod je, kao string, na dinamički način, iz JAVA koda, prosleđen u WebView pogled na prikazivanje.

Savladavanjem ove lekcije, studenti su osposobljeni da kreiraju i koriste menije u Android aplikacijama i da koriste obrađene specifične poglede za kreiranje bogatijih i zanimljivijih korisničkih interfejsa vlastitih Android aplikacija.

LITERATURA

Za pripremanje lekcije korišćena je sledeća literatura

1. Lee W. M. 2012. *Android 4 – razvoj aplikacija*, Wiley Publishing, INC
2. http://developer.android.com/training/index.html
3. http://www.tutorialspoint.com/android/
4. http://www.vogella.com/tutorials/android.html