Vidzemes Augstskola

**Inženierzinātņu Fakultāte**

mobilo tehnoloģiju risinājumi

grupa 02

Lielais Praktiskais Darbs

InspirationApp

Valmiera, 2018

SATURS

[1 Ievads 4](#_Toc530578357)

[2 InspirationApp 4](#_Toc530578358)

[2.1 Prototips 4](#_Toc530578359)

[2.2 MainActivity.java 5](#_Toc530578360)

[2.3 Fragment\_main.xml 5](#_Toc530578361)

[2.4 Cloud vision 6](#_Toc530578362)

[PIELIKUMS 1 7](#_Toc530578363)

[PIELIKUMS 2 9](#_Toc530578364)

|  |  |  |  |
| --- | --- | --- | --- |
| Dokumenta vēsture | | | |
| Versija | Statuss / Izmaiņas | Datums | Autors |
| 1.0 | Pirmā versija | 19.01.2018 | Toms Amsons |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Kontakti un atbildīgais (-ie) | | | |
| Vārds, Uzvārds | Struktūrvienība | Amats | Kontaktinformācija (e-pasts) |
| Toms Amsons | Grupa #2 | Dalībnieks | toms.amsons@va.lv |
| Ģirts Ķesteris | Grupa #2 | Dalībnieks | girts.kesteris@va.lv |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Ievads

“InspirationApp” ir lietotne, kas paredzēta ikdienas lietotājam, lai padarītu tā dienu labāku, un iedvesmotu to turpināt iesāktos darbus.

Lietotne attēlo lietotājam iedvesmojošus citātus uz skaistām un nomierinošām bildēm, iedvesmojošas mūzikas pavadījumā. Teksti un attēli tiek ielādēti pēc nejaušības principa no datubāzes.

# InspirationApp

## Prototips



Atverot lietotni, atveras galvenais logs, kurā lietotājs var veikt visas darbības (lasīt citātus, aplūkot attēlus, un klausīties pavadošo mūziku). Lietotājs izmantojot *swipe* funkciju var pārslēgties starp citātiem.

Citāti tiek ielādēti no datubāzes un tiek saglabāti lokāli. Pie lietotnes palaišanas notiek pārbaude, vai nav pievienoti jauni citāti.

## MainActivity.java

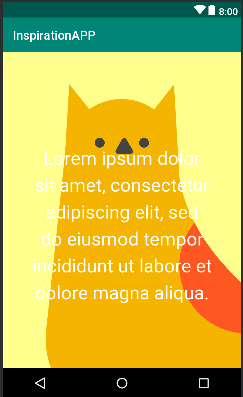


Failā MainActivity.java tiek izveidots savienojums ar serveri, kas lejupielādē citātus no servera, ja citāti jau ir lejupielādēti, tad notiek pārbaude vai nav pievienoti jauni citāti. Tie tiek attēloti uz ekrāna.

Šobrīd fona bildes tiek uzglabātas lokāli uz ierīces, bet nākotnē šos attēlus varētu pārcelt uz servera un attēlu ielāde notiktu līdzīgi, kā tekstam.

Fona mūzika atskaņošana tiek palaista no šī faila. Fona mūzika sāk skanēt, kad lietotne ir palaista.

## Fragment\_main.xml

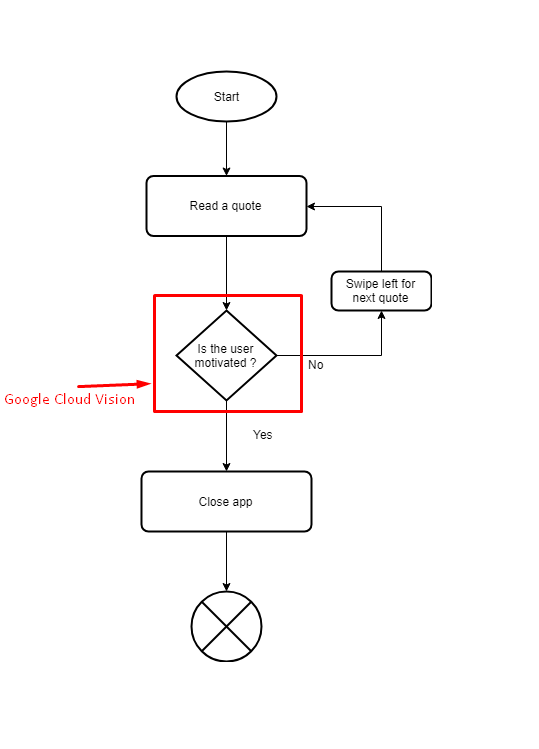


Šis fails veido lietotnes dizaina paraugu. Šeit tiek definēti teksta parametri (fonts, lielums, krāsa, atrašanās vieta), attēla atrašanās vieta. Šiem parametriem tiek piešķirts ID, ko tālāk apstrādā no MainActivity.java faila.

## Cloud vision

Projektā bija paredzēts ieviest Google Cloud vision risinājumu, kas veiktu lietotāja sejas atpazīšanu. Šī funkcionalitāte tiktu izmantota, lai noteiktu vai lietotājam uzlabojas garastāvoklis vai nē.

Google Cloud Vision risinājums spēj noteikt cilvēka sejas stāvokļus (emocijas) un arī to, vai cilvēks valkā kādu galvas aksesuāru (cepuri, brilles u.c.).



Šo risinājumu projektā neizdevās ieviest, jo lietotne tiek veidota vairākām iterācijām, un šī brīža prototips attēlo lietotnes pamata funkcionalitāti.

# PIELIKUMS 1

MainActivity.java

public class MainActivity extends AppCompatActivity {  
 private static final String *TAG* = "MainActivity";  
 private static final String *QUOTE\_CACHE\_FILENAME* = "qoutes.cache";  
  
 private static final String *QUOTES\_JSON\_DATA\_URI* = "https://gist.githubusercontent.com/Sacristan/3cdc5db13184df250349467e7a568e28/raw/inspirational\_quotes.json";  
 private static final int *PICTURE\_COUNT* = 24;  
 private SectionsPagerAdapter mSectionsPagerAdapter;  
 private ViewPager mViewPager;  
  
 private static List<String> *quotesList* = new ArrayList<String>();  
 private MediaPlayer mediaPlayer;  
  
 private static FirebaseAnalytics *mFirebaseAnalytics*;  
  
 ProgressDialog pd = null;  
  
 private static int *currentPageCounter* = 0;  
 private static int *prevPicId* = -1;  
 private static int *prevQuouteId* = -1;  
  
 private static void logAnalyticsEvent(String event){  
 Bundle bundle = new Bundle();  
 bundle.putString(FirebaseAnalytics.Param.*ITEM\_NAME*, event);  
 *mFirebaseAnalytics*.logEvent(FirebaseAnalytics.Event.*SELECT\_CONTENT*, bundle);  
 }  
  
 private void startInspirationalExperience(){  
 mSectionsPagerAdapter = new SectionsPagerAdapter(getSupportFragmentManager());  
  
 mViewPager = (ViewPager) findViewById(R.id.*container*);  
 mViewPager.setAdapter(mSectionsPagerAdapter);  
 }  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 playAudio(R.raw.*inspirational\_background*);  
 fetchQuoutes();  
  
 *mFirebaseAnalytics* = FirebaseAnalytics.*getInstance*(this);  
 *logAnalyticsEvent*("launch\_application");  
 }  
  
 @Override  
 protected void onDestroy() {  
 super.onDestroy();  
 *quotesList* = null;  
 *mFirebaseAnalytics* = null;  
 *currentPageCounter* = 0;  
  
 *logAnalyticsEvent*("exit\_application");  
 }  
  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.*menu\_main*, menu);  
 return true;  
 }  
  
 @Override  
 public boolean onOptionsItemSelected(MenuItem item) {  
 int id = item.getItemId();  
  
 if (id == R.id.*action\_settings*) {  
 return true;  
 }  
  
 return super.onOptionsItemSelected(item);  
 }  
  
 public static class InpirationalContentFragment extends Fragment {  
 private static final String *ARG\_SECTION\_NUMBER* = "section\_number";  
  
 public InpirationalContentFragment() {  
 }  
  
 public static InpirationalContentFragment newInstance(int sectionNumber) {  
 InpirationalContentFragment fragment = new InpirationalContentFragment();  
 Bundle args = new Bundle();  
 args.putInt(*ARG\_SECTION\_NUMBER*, sectionNumber);  
 fragment.setArguments(args);  
 return fragment;  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 View rootView = inflater.inflate(R.layout.*fragment\_main*, container, false);  
  
 ImageView imageView = (ImageView) rootView.findViewById(R.id.*inspirationalBackground*);  
  
 *prevPicId* = *getRandomId*(*PICTURE\_COUNT*, *prevPicId*) + 1;  
  
 String imgName = "img\_" + *prevPicId*;  
 int id = getResources().getIdentifier(imgName, "drawable", this.getContext().getPackageName());  
 imageView.setImageResource(id);  
  
 TextView textView = (TextView) rootView.findViewById(R.id.*inspirationalText*);  
  
 *prevQuouteId* = *getRandomId*(*quotesList*.size(), *prevQuouteId*);  
 String inspirationalText = *quotesList*.get(*prevQuouteId*);  
  
 textView.setText(inspirationalText);  
  
 Log.*i*(*TAG*, "GENERATED IMGID: " + *prevPicId*+ " QUOUTEID: "+ *prevQuouteId*);  
  
 *currentPageCounter*++;  
 *logAnalyticsEvent*("look\_page\_"+*currentPageCounter*);  
  
 return rootView;  
 }  
 }  
  
 public class SectionsPagerAdapter extends FragmentPagerAdapter {  
  
 public SectionsPagerAdapter(FragmentManager fm) {  
 super(fm);  
 }  
  
 @Override  
 public Fragment getItem(int position) {  
 return InpirationalContentFragment.*newInstance*(position + 1);  
 }  
  
 @Override  
 public int getCount() {  
 return Integer.*MAX\_VALUE*; //Gl, with that  
 }  
 }  
  
 private static int getRandomId(int barrierValue, int prevValue){  
 Random rand = new Random();  
  
 int rnd = rand.nextInt(barrierValue);  
  
 while(rnd==prevValue){  
 rnd = rand.nextInt(barrierValue);  
 }  
  
 return rnd;  
 }  
  
 private void playAudio(int id){  
 mediaPlayer = MediaPlayer.*create*(getApplicationContext(), id);  
  
 if (!mediaPlayer.isPlaying())  
 {  
 mediaPlayer.start();  
 mediaPlayer.setLooping(true);  
 }  
 }  
  
 private void fetchQuoutes(){  
 new JsonTask().execute(*QUOTES\_JSON\_DATA\_URI*);  
 }  
  
 private void parseQuotesJSON(String rawJSON){  
 final String ParseQuotesTag = "ParseQuotesJSON";  
  
 JSONObject jObject = null;  
 JSONObject jCacheObject = loadJSONFromCache();  
  
 boolean emptyData = rawJSON == "";  
  
 try {  
  
 if(emptyData) {  
 Log.*i*(ParseQuotesTag, "Received empty JSON data - trying to load from cache...");  
 jObject = jCacheObject;  
 }  
 else{  
 jObject = new JSONObject(rawJSON);  
 int version = jObject.getInt("version");  
  
 if (jCacheObject != null) {  
 int cacheVersion = jCacheObject.getInt("version");  
  
 if (version == cacheVersion) {  
 Log.*i*(ParseQuotesTag, "EXT and CACHE JSON versions the same. Nothing to do here!");  
 }  
 else{  
 Log.*i*(ParseQuotesTag, "EXT and CACHE JSON versions differ. Creating cache file...");  
 createCacheFile(rawJSON);  
 }  
 }  
 else{  
 Log.*i*(ParseQuotesTag, "No cache JSON present - creating one...");  
 createCacheFile(rawJSON);  
 }  
 }  
  
 if(jObject!=null) {  
 JSONArray jArray = jObject.getJSONArray("quotes");  
 populateQuoutes(jArray);  
 }  
  
 } catch (JSONException e) {  
 e.printStackTrace();  
 }  
  
 }  
  
 private JSONObject loadJSONFromCache(){  
 File directory = getApplicationContext().getFilesDir();  
 File file = new File(directory, *QUOTE\_CACHE\_FILENAME*);  
  
 if(!file.exists()) return null;  
  
 try {  
 BufferedReader br = new BufferedReader(new FileReader(file));  
 String line;  
  
 line = br.lines().collect(Collectors.*joining*());  
 Log.*i*(*TAG*, "READ CACHE JSON: "+line);  
  
 br.close();  
 try {  
 JSONObject jObject = new JSONObject(line);  
 return jObject;  
 }  
  
 catch (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
 catch (IOException e) {  
 e.printStackTrace();  
  
 }  
 return null;  
 }  
  
 private void createCacheFile(String data){  
 try {  
  
// File file = File.createTempFile(QUOTE\_CACHE\_FILENAME, null, getApplicationContext().getFilesDir());  
 File file = new File(getApplicationContext().getFilesDir(), *QUOTE\_CACHE\_FILENAME*);  
  
 FileOutputStream fOut = new FileOutputStream(file);  
 OutputStreamWriter myOutWriter = new OutputStreamWriter(fOut);  
 myOutWriter.append(data);  
  
 myOutWriter.close();  
  
 fOut.flush();  
 fOut.close();  
 }  
 catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void populateQuoutes(JSONArray jArray){  
 for (int i=0; i < jArray.length(); i++){  
 try {  
 String qoute = jArray.getString(i);  
 *quotesList*.add(qoute);  
 } catch (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 protected class JsonTask extends AsyncTask<String, String, String> {  
  
 protected void onPreExecute() {  
 super.onPreExecute();  
  
 pd = new ProgressDialog(MainActivity.this);  
 pd.setMessage("Please wait...");  
 pd.setCancelable(false);  
 pd.show();  
 }  
  
 protected String doInBackground(String... params) {  
 HttpURLConnection connection = null;  
 BufferedReader reader = null;  
  
 try {  
 URL url = new URL(params[0]);  
 connection = (HttpURLConnection) url.openConnection();  
 connection.connect();  
  
 InputStream stream = connection.getInputStream();  
  
 reader = new BufferedReader(new InputStreamReader(stream));  
  
 StringBuffer buffer = new StringBuffer();  
 String line = "";  
  
 while ((line = reader.readLine()) != null) {  
 buffer.append(line+"\n");  
 Log.d("Response: ", "> " + line);  
  
 }  
  
 return buffer.toString();  
  
  
 } catch (MalformedURLException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 if (connection != null) {  
 connection.disconnect();  
 }  
 try {  
 if (reader != null) {  
 reader.close();  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 return null;  
 }  
  
 @Override  
 protected void onPostExecute(String result) {  
 super.onPostExecute(result);  
  
 parseQuotesJSON(result);  
  
 if (pd.isShowing()){  
 pd.dismiss();  
 }  
  
 startInspirationalExperience();  
  
 }  
 }  
}

# PIELIKUMS 2

Fragment\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/constraintLayout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity$InpirationalContentFragment">  
  
 <TextView  
 android:id="@+id/section\_label"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="@dimen/activity\_horizontal\_margin"  
 android:layout\_marginTop="@dimen/activity\_vertical\_margin"  
 android:layout\_marginEnd="@dimen/activity\_horizontal\_margin"  
 android:layout\_marginBottom="@dimen/activity\_vertical\_margin"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toTopOf="@+id/constraintLayout"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintTop\_creator="1" />  
  
 <ImageView  
 android:id="@+id/inspirationalBackground"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:contentDescription="@string/contentDescription"  
 android:scaleType="centerCrop"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintStart\_toEndOf="@+id/section\_label"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0"  
 tools:src="@tools:sample/avatars[3]" />  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginTop="150dp"  
 android:layout\_marginEnd="20dp"  
 android:layout\_marginBottom="100dp"  
 android:gravity="center"  
 android:orientation="vertical"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent">  
  
 <TextView  
 android:id="@+id/inspirationalText"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:lineSpacingExtra="8sp"  
 android:shadowColor="#000000"  
 android:shadowDx="0.0"  
 android:shadowDy="0.0"  
 android:shadowRadius="15.0"  
 android:text="Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."  
 android:textAlignment="center"  
 android:textColor="#ffffff"  
 android:textSize="30sp"  
 tools:layout\_editor\_absoluteX="101dp"  
 tools:layout\_editor\_absoluteY="24dp" />  
 </LinearLayout>  
  
</android.support.constraint.ConstraintLayout>