Mateusz Szuwarowski
Bartłomiej Kołodziej
Informatyka III zaocznie
25 Stycznia 2019

Aplikacja Mobilna
Skaner Kodów QR
(multifunction tool)

Aplikacja napisana przy użyciu: Android Studio 3.3. Program ten posiada odpowiednie środowisko do to łatwego oraz przyjemnego tworzenia aplikacji mobilnych.

Skaner Kodów QR powstał w oparciu o otwartą bibliotekę ZXing, która służy do przetwarzania obrazów, zaimplementowana w Javie.

Analiza kodu aplikacji Skaner Kodów QR:

Aby móc korzystać z biblioteki ZXing należy dodać do dependencies następujące implementacje:

```
-implementation 'xyz.belvi.mobilevision:barcodescanner:2.0.3'
-implementation 'me.dm7.barcodescanner:zxing:1.9.8'
```

Kolejną czynnością jest dodanie do pliku AndroidManifest.xml zezwoleń na wykorzystanie sprzętu w tym przypadku dostępu do kamery telefonu

```
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.hardware.CAMERA"/>
<uses-feature android:name="android.hardware.camera.autofocus"/>
```

Klasa MainActivity

```
package com.example.multifunction;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class MainActivity extends AppCompatActivity {
    Button barScanner;
    Button textReco;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    barScanner = (Button) findViewById(R.id.idBarcode);
    textReco = (Button) findViewById(R.id.idText);
```

```
    barScanner.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(MainActivity.this,barScanner.class));
        }
    });
    textReco.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            startActivity(new Intent(MainActivity.this,text.class));
        }
    });
    }
}
```

Tutaj są dostępne dwa przyciski z ustawionymi metodami przekierowujące do widoku barScanner oraz text. Klasa barScanner wykonuje tylko część zadania, mianowicie za jej pomocą można odczytać zakodowaną informację w kodzie Qr, gdy jest to adres www nie ma możliwości odwiedzenia strony.

```
package com.example.multifunction;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.SparseArray;
import com.google.android.gms.samples.vision.barcodereader.BarcodeCapture;
import com.google.android.gms.samples.vision.barcodereader.BarcodeGraphic;
import com.google.android.gms.vision.barcode.Barcode;
import java.util.List;
import xyz.belvi.mobilevisionbarcodescanner.BarcodeRetriever;
public class barScanner extends AppCompatActivity implements BarcodeRetriever {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bar_scanner);
        BarcodeCapture barcodeCapture =
(BarcodeCapture)getSupportFragmentManager().findFragmentById(R.id.barcode);
        barcodeCapture.setRetrieval(this);
    }
    @Override
    public void onPermissionRequestDenied() {
    }
    @Override
    public void onRetrievedFailed(String reason) {
    }
    @Override
    public void onRetrieved(final Barcode barcode) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                AlertDialog.Builder builder = new AlertDialog.Builder(barScanner.this)
                        .setTitle("Code Retrrived")
                        .setMessage(barcode.displayValue);
                builder.show();
            }
        });
    }
    @Override
    public void onRetrievedMultiple(Barcode closetToClick, List<BarcodeGraphic> barcode) {
    }
    @Override
    public void onBitmapScanned(SparseArray<Barcode> sparseArray) {
    }
}
```

Metoda onRetrived() jest odpowiedzialna za pobranie informacji zakodowanej i jej wyświetleniu na ekranie.

Klasa text jest ulepszoną wersją klasy barScanner. Umożliwia pobrany kod wyświetlić w przeglądarce o ile jest to adres www.

```
package com.example.multifunction;
import android.Manifest;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Build;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;
import com.google.zxing.Result;
import me.dm7.barcodescanner.zxing.ZXingScannerView;
import static android.Manifest.permission_group.CAMERA;
public class text extends AppCompatActivity implements ZXingScannerView.ResultHandler {
    private final static int REQ_CAMERA = 1;
    private  ZXingScannerView scannerView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        scannerView = new ZXingScannerView(this);
        setContentView(scannerView);
        if(Build.VERSION.SDK_INT >=Build.VERSION_CODES.M)
        {
            if(checkPermission())
            {
                Toast.makeText(text.this,"Grandeed",Toast.LENGTH_LONG).show();
            }else{
            }
        }
    }
    private boolean checkPermission()
    {
        return (ContextCompat.checkSelfPermission(text.this, Manifest.permission.CAMERA)==
PackageManager.PERMISSION_GRANTED);
    }
    private void requestPermission()
    {
        ActivityCompat.requestPermissions(this,new String[]{CAMERA},REQ_CAMERA);
    }
    public  void onRequestPermissionResault(int requestCode, String permission[],int
grantResult[])
    {
        switch(requestCode)
        {
            case REQ_CAMERA:
                if(grantResult.length>0)
                {
                    boolean cameraAccepted = grantResult[0] ==
PackageManager.PERMISSION_GRANTED;
                    if(cameraAccepted)
                    {
                        Toast.makeText(text.this,"Granted",Toast.LENGTH_LONG).show();
                        if(Build.VERSION.SDK_INT>= Build.VERSION_CODES.M)
                        {
```

```
                                if(shouldShowRequestPermissionRationale(CAMERA))
                                {
                                     displayAllertMessage("Need to allow perrmitions", new
DialogInterface.OnClickListener() {
                                          @Override
                                          public void onClick(DialogInterface dialog, int
which) {
                                               if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
                                                    requestPermissions(new String[]{CAMERA},
REQ_CAMERA);
                                               }
                                          }
                                     });
                                return;
                                }
                           }
                      }
                }
                break;
          }
     }
     @Override
     public void  onResume()
     {
          super.onResume();
          if(Build.VERSION.SDK_INT >=Build.VERSION_CODES.M)
          {
               if(checkPermission())
               {
                    if(scannerView == null)
                    {
                         scannerView = new ZXingScannerView(this);
                         setContentView(scannerView);
                    }
                    scannerView.setResultHandler(this);
                    scannerView.startCamera();
               }else{
                    requestPermission();
               }
          }
     }
     @Override
     public void onDestroy()
     {
          super.onDestroy();
          scannerView.stopCamera();
     }
     public void displayAllertMessage(String message, DialogInterface.OnClickListener
listener){
          new AlertDialog.Builder(text.this).setMessage(message)
                    .setPositiveButton("ok",listener)
                    .setPositiveButton("cancel",null)
                    .create()
                    .show();
     }
     @Override
     public void handleResult(Result result) {
          final String scanResult = result.getText();
          AlertDialog.Builder builder = new AlertDialog.Builder(this);
          builder.setTitle("Scan result");
          builder.setPositiveButton("cancel", new DialogInterface.OnClickListener() {
               @Override
               public void onClick(DialogInterface dialog, int which) {
                    scannerView.resumeCameraPreview(text.this);
               }
```

```
        });
        builder.setNegativeButton("Visit", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(scanResult));
                startActivity(intent);
            }
        });
        builder.setMessage(scanResult);
        AlertDialog alert = builder.create();
        alert.show();
    }
}
```

Metoda checkPermition() sprawdza uprawnienia dostępu do kamery telefonu.

Metoda onRequestResult() sprowadza się do sprawdzenia uprawnień ewentualnie wyświetlenia informacji o ich braku, bądź prośbie o udzieleniu aplikacji dostępu do kamery.

Metoda onResume() uruchamia przetwarzanie obrazu następnie przesyła obraz do metody handleResult.

Metoda handleResult() odpowiedzialna jest za wyświetlenie zawartości kodu Qr w formie komunikatu, z opcją vist, czyli przejście do przeglądarki, bądź cancel brak czynności, przejście do ponownego skanowania.

Wygląd aplikacji opisany jest w kodzie xml, activity_main.xml posiada dwa przyciski, pole tekstowe oraz zaimportowane zdjęcie.

```
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/color_orange"
    tools:context=".MainActivity">
    <Button
        android:id="@+id/idBarcode"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:background="@color/color_green"
        android:text="Barcode Scanner"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.704" />
    <Button
        android:id="@+id/idText"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
```

```
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:background="@color/colorPrimary"
        android:text="Barcode Scanner+"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.823" />
    <TextView
        android:id="@+id/textView"
        android:layout_width="220dp"
        android:layout_height="48dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="SKANER KODÓW "
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.6"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.191" />
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.377"
        app:srcCompat="@drawable/qr" />
</android.support.constraint.ConstraintLayout>
```

Wygląd activity_text zaprogramowany jest przy użyciu narzuconego wzorca biblioteki ZXing:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".text">
</android.support.constraint.ConstraintLayout>
```

To samo jest w przypadku widoku okna activity_barcode:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".barScanner">
<fragment
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:name='com.google.android.gms.samples.vision.barcodereader.BarcodeCapture'
    android:id="@+id/barcode"
    app:gvb_auto_focus = "true"
    app:gvb_code_format = "all_format"
    app:gvb_flash= "false"
    >
</fragment>
</RelativeLayout>
```