

Name: Avinash Kumar

CSU ID: 2739849

Android Sensor Programming

Homework #15

**AndroidManifest.xml**

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.homework15">

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

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

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>

**Activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout 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=".MainActivity"

android:orientation="horizontal">

<GridView

android:layout\_width="120dp"

android:layout\_height="match\_parent"

android:id="@+id/gridView"/>

<ImageSwitcher

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/imageSwitcher"

android:layout\_alignParentStart="true"

android:layout\_alignParentLeft="true"/>

</LinearLayout>

**MainActivity.java**

package com.example.homework15;

import androidx.appcompat.app.ActionBar;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

import android.Manifest;

import android.content.Context;

import android.content.pm.PackageManager;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.drawable.BitmapDrawable;

import android.os.AsyncTask;

import android.os.Bundle;

import android.renderscript.ScriptGroup;

import android.util.Log;

import android.view.View;

import android.view.ViewGroup;

import android.view.animation.AnimationUtils;

import android.widget.AdapterView;

import android.widget.BaseAdapter;

import android.widget.GridView;

import android.widget.ImageSwitcher;

import android.widget.ImageView;

import android.widget.Toast;

import android.widget.ViewSwitcher;

import org.w3c.dom.Document;

import java.io.IOException;

import java.io.InputStream;

import java.net.HttpURLConnection;

import java.net.URL;

import java.net.URLConnection;

import java.util.ArrayList;

import java.util.List;

import java.util.concurrent.locks.Lock;

import java.util.concurrent.locks.ReentrantLock;

public class MainActivity extends AppCompatActivity {

private final Lock aLock = new ReentrantLock();

int imageCounter = 2;

final private int REQUEST\_INTERNET = 123;

List<Bitmap> imageArray = new ArrayList<Bitmap>();

private ImageSwitcher imgSwitcher;

GridView gridView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

imgSwitcher = findViewById(R.id.imageSwitcher);

imgSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,

android.R.anim.fade\_in));

imgSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,

android.R.anim.fade\_out));

imgSwitcher.setFactory(new ViewSwitcher.ViewFactory() {

@Override

public View makeView() {

ImageView myView = new ImageView(getApplicationContext());

myView.setScaleType(ImageView.ScaleType.FIT\_CENTER);

myView.setLayoutParams(new ImageSwitcher.LayoutParams(

ActionBar.LayoutParams.WRAP\_CONTENT, ActionBar.LayoutParams.WRAP\_CONTENT));

return myView;

}

});

gridView = findViewById(R.id.gridView);

if (ContextCompat.checkSelfPermission(this, Manifest.permission.INTERNET)

!= PackageManager.PERMISSION\_GRANTED) {

ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.INTERNET}, REQUEST\_INTERNET);

} else {

ConnectURL();

}

}

private void ConnectURL() {

String imgurl = "https://www.csuohio.edu/about-csu/about-csu";

new DownloadTask().execute(imgurl);

}

@Override

public void onRequestPermissionsResult(int requestCode,

String[] permissions, int[] grantResults) {

switch (requestCode) {

case REQUEST\_INTERNET:

if (grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

ConnectURL();

} else {

Toast.makeText(MainActivity.this,

"Permission Denied", Toast.LENGTH\_SHORT).show();

}

break;

default:

super.onRequestPermissionsResult(requestCode,

permissions, grantResults);

}

}

private InputStream OpenHttpConnection(String urlString) throws IOException

{

InputStream in = null; int response = -1; URL url = new URL(urlString);

URLConnection conn = url.openConnection();

if (!(conn instanceof HttpURLConnection))

throw new IOException("Not an HTTP connection");

try{

HttpURLConnection httpConn = (HttpURLConnection) conn;

httpConn.setAllowUserInteraction(false);

httpConn.setInstanceFollowRedirects(true);

httpConn.setRequestMethod("GET");

httpConn.connect();

response = httpConn.getResponseCode();

if (response == HttpURLConnection.HTTP\_OK) {

in = httpConn.getInputStream();

}

} catch (Exception ex)

{

Log.d("Networking", ex.getLocalizedMessage()); throw new IOException("Error connecting");

}

return in;

}

private InputStream download(String URL) {

InputStream in = null;

try {

in = OpenHttpConnection(URL);

return in;

} catch (IOException e1) {

Log.d("NetworkingActivity", e1.getLocalizedMessage());

}

return null;

}

private Bitmap DownloadImage(String URL)

{

Bitmap bitmap = null;

InputStream in = download(URL);

if(in != null) {

bitmap = BitmapFactory.decodeStream(in);

try {

in.close();

} catch (IOException e1) {

Log.d("NetworkingActivity", e1.getLocalizedMessage());

}

}

return bitmap;

}

private Bitmap DownloadContent(String URL)

{

Bitmap bitmap = null;

InputStream in = download(URL);

String strDefinition = "";

if(in != null) {

Document doc = null;

try {

doc = Jsoup.connect(URL).get();

} catch (Exception e) { e.printStackTrace(); }

Elements definitionElements = doc.getElementsByTag("img");

for (int i = 0; i < definitionElements.size(); i++) {

org.jsoup.nodes.Element itemNode = definitionElements.get(i);

strDefinition = itemNode.attr("src");

if(strDefinition.contains("http"))

{

new DownloadImageTask().execute(strDefinition);

}

}

}

return bitmap;

}

private class DownloadTask extends AsyncTask<String, Void, Bitmap> {

protected Bitmap doInBackground(String... urls) {

return DownloadContent(urls[0]);

}

protected void onPostExecute(Bitmap result) {

}

}

private class DownloadImageTask extends AsyncTask<String, Void, Bitmap> {

protected Bitmap doInBackground(String... urls) {

return DownloadImage(urls[0]);

}

protected void onPostExecute(Bitmap result) {

imageArray.add(result);

aLock.lock();

imageCounter = imageCounter-1;

if(imageCounter ==0)

{

showImage();

}

aLock.unlock();

}

}

private void showImage()

{

gridView.setAdapter(new ImageAdapter(this));

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener()

{

public void onItemClick(AdapterView parent, View v, int position, long id) {

imgSwitcher.setImageDrawable(new BitmapDrawable(imageArray.get(position)));

}

});

}

public class ImageAdapter extends BaseAdapter {

private Context context;

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

public int getCount() { return imageArray.size(); }

public Object getItem(int position) { return position; }

public long getItemId(int position) { return position; }

public View getView(int position, View convertView, ViewGroup parent) {

ImageView imageView;

if (convertView == null) {

imageView = new ImageView(context);

imageView.setLayoutParams(new GridView.LayoutParams(150, 150));

imageView.setScaleType(ImageView.ScaleType.CENTER\_CROP);

imageView.setPadding(5, 5, 5, 5);

} else {

imageView = (ImageView) convertView;

}

imageView.setImageBitmap(imageArray.get(position));

return imageView;

}

}

}

**Implementation**

I have used jsoup to get the image tag from html page, it should to build the package while running on other computer.

**Screenshot**

|  |  |
| --- | --- |
|  |  |