

**KAUNAS UNIVERSITY OF TECHNOLOGY**

**FACULTY OF INFORMATICS**

**COMPUTER DEPARTMENT**

### App Development for Smart Mobile Systems 3rd Labaratory work

**Assignment completed by:**

IFF 6/8 group student

Tadas Laurinaitis

**Assignment evaluated by**:

Prof. Rytis Maskeliūnas

### Table of contents:

Contents

[App Development for Smart Mobile Systems 3rd Labaratory work 1](#_Toc532760456)

[Table of contents: 2](#_Toc532760457)

[Tasks: 3](#_Toc532760458)

[Text describing how each task was solved: 3](#_Toc532760459)

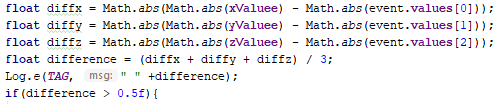
[Literature list: 9](#_Toc532760460)

[Source code of my tasks: 9](#_Toc532760461)

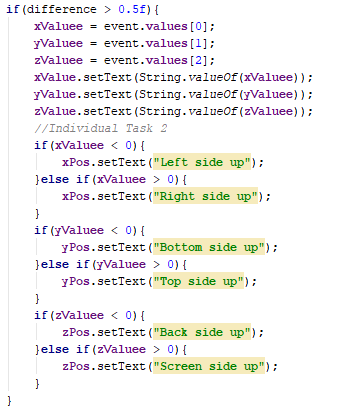
### Tasks:

### Text describing how each task was solved:

Task #1: I calculated the difference between old and new XYZ values and before changing them I would check if the difference is atleast 0.5 to avoid registering slight changes



Task #2: For this task I wrote a simple if statement which would check values of each axis and would display the position of device besides them

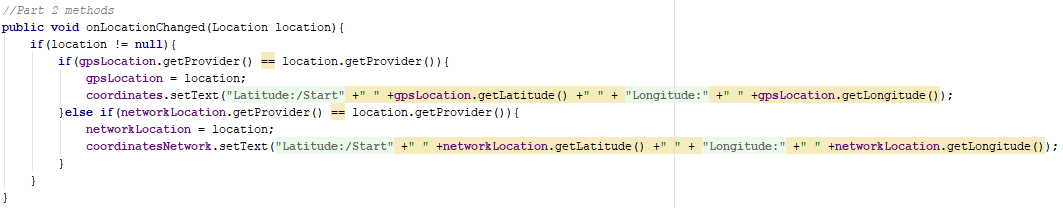


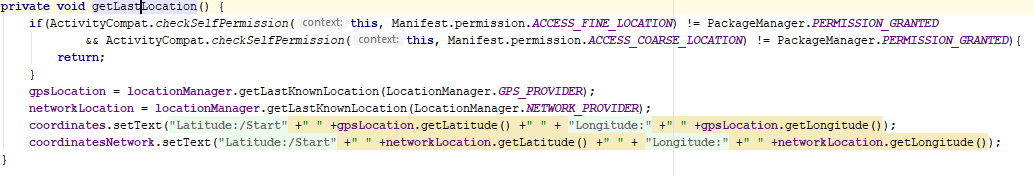
Task #3: I created a separate activity which displays compass showing north based on sensor data. The picture below shows that I get degree values based on the values of X axis and the method I use for the rotation of the image.





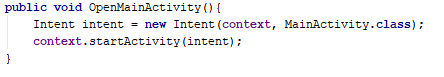
Task #4: The task was solved by receiving data from both GPS provider and Network provider and converting that data to coordinates.



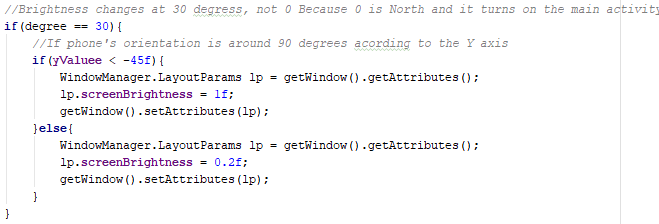


Task #5: The task was solved by writing a simple if statement which checks if the compass degrees are equal to 0 (North). If the statement is true, then it opens mainActivity which contains the camera. The only thing that doesn’t work is automatic photo capture and showing.





Task #6: The task was solved by using an if statement which checks if the value of compass degrees is equal to 30 (using 30 instead of 0 because 0 is the value of North and it would automatically bring me to mainActivity) and then checks the orientation of the device (if the device is laying flat it sets the brightness to 0% and if the device is oriented at around 90 degrees based on Y axis, then the brightness is set to Maximum).



Task #7: The task was solved by using an if statement which checks if the value of compass degrees is equal to 180 (South) and then checks the orientation of the device (if the device is oriented at around 90 degrees based on Y axis) and sends SOS code in flashes (…---…). For this part I used method which I created for my project work which uses older camera API to access its flash.



### Literature list:

<https://www.androidcode.ninja/android-compass-code-example/>

<https://stackoverflow.com/questions/6068803/how-to-turn-on-front-flash-light-programmatically-in-android>

### Source code of my tasks:

**package** hehehe.destroyer.lab3mobileapps;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.hardware.Camera;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.os.Bundle;  
**import** android.provider.Settings;  
**import** android.util.Log;  
**import** android.view.WindowManager;  
**import** android.view.animation.Animation;  
**import** android.view.animation.RotateAnimation;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
**import** hehehe.destroyer.lab3mobileapps.R;  
  
**public class** CompassActivity **extends** Activity **implements** SensorEventListener {  
  
 **private** Context **context** = **this**;  
 **private static final** String ***TAG*** = **"Compass: "**;  
 *// define the display assembly compass picture* **private** ImageView **image**;  
 *// record the compass picture angle turned* **private float currentDegree** = 0f;  
 **private float lastDegree** = 0f;  
 *// device sensor manager* **private** SensorManager **mSensorManager**;  
 TextView **tvHeading**;  
  
 *//Part 5 BRIGHTNESS* **private int brightness**;  
 **private** SensorManager **senSensorManager**;  
 **private** Sensor **senAccelerometer**;  
 **private boolean InformationObtained**;  
  
 *//Part 6 SOS* **public static** Camera *cam* = **null**;  
  
 *//Part 1 and 2 Individual* **private float xValuee**;  
 **private float yValuee**;  
 **private float zValuee**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***compass\_activity***);  
  
 *// our compass image* **image** = (ImageView) findViewById(R.id.***imageViewCompass***);  
  
 *// TextView that will tell the user what degree is he heading* **tvHeading** = (TextView) findViewById(R.id.***tvHeading***);  
  
 *// initialize your android device sensor capabilities* **mSensorManager** = (SensorManager) getSystemService(***SENSOR\_SERVICE***);  
 */\*  
 try{  
 Settings.System.putInt(getContentResolver(),  
 Settings.System.SCREEN\_BRIGHTNESS\_MODE,  
 Settings.System.SCREEN\_BRIGHTNESS\_MODE\_MANUAL);  
  
 brightness = System.g(getContentResolver(),  
 Settings.System.SCREEN\_BRIGHTNESS);  
 }  
 catch(Settings.SettingNotFoundException e){  
 Log.e("Error", "Cannot access system brightness");  
 e.printStackTrace();  
 }\*/* **senSensorManager** = (SensorManager) getSystemService(Context.***SENSOR\_SERVICE***);  
 **senAccelerometer** = **senSensorManager**.getDefaultSensor(Sensor.***TYPE\_ACCELEROMETER***);  
 **xValuee** = 0;  
 **yValuee** = 0;  
 **zValuee** = 0;  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
  
 *// for the system's orientation sensor registered listeners* **mSensorManager**.registerListener(**this**, **mSensorManager**.getDefaultSensor(Sensor.***TYPE\_ORIENTATION***),  
 SensorManager.***SENSOR\_DELAY\_GAME***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
  
 *// to stop the listener and save battery* **mSensorManager**.unregisterListener(**this**);  
 }  
 **public void** OpenMainActivity(){  
 Intent intent = **new** Intent(**context**, MainActivity.**class**);  
 **context**.startActivity(intent);  
 }  
  
 @Override  
 **public void** onSensorChanged(SensorEvent event) {  
 Sensor mySensor = event.**sensor**;  
 *// get the angle around the z-axis rotated* **float** degree = Math.*round*(event.**values**[0]);  
  
 **if**(Math.*abs*(**lastDegree** - degree) >= 1){  
  
 Log.*e*(***TAG***, **" "** +degree);  
 **tvHeading**.setText(**"Heading: "** + Float.*toString*(degree) + **" degrees"**);  
 *// create a rotation animation (reverse turn degree degrees)* RotateAnimation ra = **new** RotateAnimation(  
 **currentDegree**,  
 -degree,  
 Animation.***RELATIVE\_TO\_SELF***, 0.5f,  
 Animation.***RELATIVE\_TO\_SELF***,  
 0.5f);  
 *// how long the animation will take place* ra.setDuration(210);  
 *// set the animation after the end of the reservation status* ra.setFillAfter(**true**);  
 *// Start the animation* **image**.startAnimation(ra);  
 **currentDegree** = -degree;  
 *//Individual Part 5 Brightness  
 //xValuee = event.values[0];* **yValuee** = event.**values**[1];  
 *//zValuee = event.values[2];* Log.*e*(***TAG***, **" "** +degree +**"Y: "** +**yValuee**);  
 *//If phone is pointing north, turn on the camera activity* **if**(degree == 0) {  
 OpenMainActivity();  
 }  
 *//Brightness changes at 30 degress, not 0 Because 0 is North and it turns on the main activity* **if**(degree == 30){  
 *//If phone's orientation is around 90 degrees acording to the Y axis* **if**(**yValuee** < -45f){  
 WindowManager.LayoutParams lp = getWindow().getAttributes();  
 lp.**screenBrightness** = 1f;  
 getWindow().setAttributes(lp);  
 }**else**{  
 WindowManager.LayoutParams lp = getWindow().getAttributes();  
 lp.**screenBrightness** = 0.2f;  
 getWindow().setAttributes(lp);  
 }  
 }  
 *//Individual work Part 6 SOS flashes* **if**(degree == 180){  
 **if**(**yValuee** < -45f){  
 String sos = **"...---..."**;  
 **char**[] array = sos.toCharArray();  
 *cam* = Camera.*open*();  
 Camera.Parameters p = *cam*.getParameters();  
 **for**(**int** i = 0; i < array.**length**; i++) {  
 *//Signalu tipai* **if** (array[i] == **'.'**) {  
 p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_TORCH***);  
 *cam*.setParameters(p);  
 *cam*.startPreview();  
 **try** {  
 Thread.*sleep*(500);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 } **else if**(array[i] == **'-'**){  
 p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_TORCH***);  
 *cam*.setParameters(p);  
 *cam*.startPreview();  
 **try** {  
 Thread.*sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }**else if**(array[i] == **' '**){  
 p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_OFF***);  
 *cam*.setParameters(p);  
 *cam*.startPreview();  
 **try** {  
 Thread.*sleep*(500);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }**else if**(array[i] == **'/'**){  
 p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_TORCH***);  
 *cam*.setParameters(p);  
 *cam*.startPreview();  
 **try** {  
 Thread.*sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 *//Tarpai tarp signalu* p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_OFF***);  
 *cam*.setParameters(p);  
 **try** {  
 Thread.*sleep*(300);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 *//Pabaigus darba isjungiam* p.setFlashMode(Camera.Parameters.***FLASH\_MODE\_OFF***);  
 *cam*.setParameters(p);  
 }  
 }  
 }  
 **lastDegree** = degree;  
 }  
  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** accuracy) {  
 *// not in use* }  
}

**package** hehehe.destroyer.lab3mobileapps;  
  
**import** android.Manifest;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.graphics.Camera;  
**import** android.graphics.ImageFormat;  
**import** android.graphics.SurfaceTexture;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.hardware.camera2.CameraAccessException;  
**import** android.hardware.camera2.CameraCaptureSession;  
**import** android.hardware.camera2.CameraCharacteristics;  
**import** android.hardware.camera2.CameraDevice;  
**import** android.hardware.camera2.CameraManager;  
**import** android.hardware.camera2.CameraMetadata;  
**import** android.hardware.camera2.CaptureRequest;  
**import** android.hardware.camera2.TotalCaptureResult;  
**import** android.hardware.camera2.params.StreamConfigurationMap;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.media.Image;  
**import** android.media.ImageReader;  
**import** android.os.Environment;  
**import** android.os.Handler;  
**import** android.os.HandlerThread;  
**import** android.support.annotation.NonNull;  
**import** android.support.annotation.Nullable;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.util.Size;  
**import** android.util.SparseArray;  
**import** android.util.SparseIntArray;  
**import** android.view.Surface;  
**import** android.view.TextureView;  
**import** android.view.View;  
**import** android.view.animation.Animation;  
**import** android.view.animation.RotateAnimation;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
**import** android.widget.Button;  
**import** android.widget.Toast;  
  
**import** java.io.File;  
**import** java.io.FileNotFoundException;  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
**import** java.io.OutputStream;  
**import** java.nio.ByteBuffer;  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
**import** java.util.List;  
  
**public class** MainActivity **extends** AppCompatActivity **implements** SensorEventListener, LocationListener{  
 *//Part 1 - Accelerometer* **private** SensorManager **senSensorManager**;  
 **private** Sensor **senAccelerometer**;  
 **private** Button **startAndStop**;  
 **private** TextView **xValue**;  
 **private** TextView **yValue**;  
 **private** TextView **zValue**;  
 **private boolean InformationObtained**;  
  
 *//Part 1 and 2 Individual* **private** TextView **xPos**;  
 **private** TextView **yPos**;  
 **private** TextView **zPos**;  
 **private float xValuee**;  
 **private float yValuee**;  
 **private float zValuee**;  
  
 *//Compass* **private** Context **context** = **this**;  
 **private** Button **compassButton**;  
  
 *//Part 2 - GPS* **private** TextView **coordinates**;  
 **private** TextView **coordinatesNetwork**;  
 **private** LocationManager **locationManager**;  
 **private** Location **gpsLocation**;  
 **private** Location **networkLocation**;  
  
 *//Part 3 - Camera* **private static final** String ***TAG*** = **"AndroidCameraApi"**;  
 **private** Button **takePictureButton**;  
 **private** TextureView **textureView**;  
 **private static final** SparseIntArray ***ORIENTATIONS*** = **new** SparseIntArray();  
 **static**{  
 ***ORIENTATIONS***.append(Surface.***ROTATION\_0***, 90);  
 ***ORIENTATIONS***.append(Surface.***ROTATION\_90***, 0);  
 ***ORIENTATIONS***.append(Surface.***ROTATION\_180***, 270);  
 ***ORIENTATIONS***.append(Surface.***ROTATION\_270***, 180);  
 }  
 **private** String **cameraId**;  
 **protected** CameraDevice **cameraDevice**;  
 **protected** CameraCaptureSession **cameraCaptureSessions**;  
 **protected** CaptureRequest.Builder **captureRequestBuilder**;  
 **private** Size **imageDimension**;  
 **private** ImageReader **imageReader**;  
 **private** File **file**;  
 **private static final int *REQUEST\_CAMERA\_PERMISION*** = 200;  
 **private** Handler **mBackgroundHandler**;  
 **private** HandlerThread **mBackgroundThread**;  
  
 @Override  
 **protected void** onCreate(@Nullable Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***main\_activity***);  
 *//Part 1 and 2* **InformationObtained** = **false**;  
 **startAndStop** = (Button) findViewById(R.id.***start\_and\_stop***);  
 **startAndStop**.setOnClickListener(**StartAndStopButtonListener**);  
 **xValue** = (TextView) findViewById(R.id.***x\_value***);  
 **yValue** = (TextView) findViewById(R.id.***y\_value***);  
 **zValue** = (TextView) findViewById(R.id.***z\_value***);  
 **xPos** = (TextView) findViewById(R.id.***x\_pos***);  
 **yPos** = (TextView) findViewById(R.id.***y\_pos***);  
 **zPos** = (TextView) findViewById(R.id.***z\_pos***);  
  
 **coordinates** = (TextView) findViewById(R.id.***coordinates***);  
 **coordinatesNetwork** = (TextView) findViewById(R.id.***coordinates\_network***);  
 **senSensorManager** = (SensorManager) getSystemService(Context.***SENSOR\_SERVICE***);  
 **senAccelerometer** = **senSensorManager**.getDefaultSensor(Sensor.***TYPE\_ACCELEROMETER***);  
 **locationManager** = (LocationManager) getSystemService(Context.***LOCATION\_SERVICE***);  
 getLastLocation();  
  
 **xValuee** = 0;  
 **yValuee** = 0;  
 **zValuee** = 0;  
  
 *//Compass button* **compassButton** = (Button) findViewById(R.id.***compass***);  
 **compassButton**.setOnClickListener(**compassListener**);  
  
 *//Part 3* **textureView** = (TextureView) findViewById(R.id.***textureView***);  
 **assert textureView** != **null**;  
 **textureView**.setSurfaceTextureListener(**textureListener**);  
 **takePictureButton** = (Button) findViewById(R.id.***take\_photo***);  
 **assert takePictureButton** != **null**;  
 **takePictureButton**.setOnClickListener(**new** View.OnClickListener(){  
 @Override  
 **public void** onClick(View v){  
 takePicture();  
 }  
 });  
 }  
 View.OnClickListener **StartAndStopButtonListener** = **new** View.OnClickListener(){  
 @Override  
 **public void** onClick(View v){  
  
 **if**(**senAccelerometer** == **null**){  
 Toast.*makeText*(MainActivity.**this**, getString(R.string.***no\_sensor***), Toast.***LENGTH\_LONG***).show();  
 **return**;  
 }  
  
 **if**(**InformationObtained**){  
 **startAndStop**.setText(getString(R.string.***start***));  
 **senSensorManager**.unregisterListener(MainActivity.**this**, **senAccelerometer**);  
 **InformationObtained** = **false**;  
 } **else**{  
 **senSensorManager**.registerListener(MainActivity.**this**, **senAccelerometer**, SensorManager.***SENSOR\_DELAY\_NORMAL***);  
 **startAndStop**.setText(getString(R.string.***stop***));  
 **InformationObtained** = **true**;  
 }  
 }  
 };  
 **public void** runCompass(){  
 Intent intent = **new** Intent(**context**, CompassActivity.**class**);  
 **context**.startActivity(intent);  
 }  
 View.OnClickListener **compassListener** = **new** View.OnClickListener(){  
 @Override  
 **public void** onClick(View v){  
 runCompass();  
 }  
 };  
 @Override  
 **public void** onSensorChanged(SensorEvent event){  
  
 Sensor mySensor = event.**sensor**;  
  
 **if**(mySensor.getType() == Sensor.***TYPE\_ACCELEROMETER***){  
 *//For accuracy (Individual Task 1)* **float** diffx = Math.*abs*(Math.*abs*(**xValuee**) - Math.*abs*(event.**values**[0]));  
 **float** diffy = Math.*abs*(Math.*abs*(**yValuee**) - Math.*abs*(event.**values**[1]));  
 **float** diffz = Math.*abs*(Math.*abs*(**zValuee**) - Math.*abs*(event.**values**[2]));  
 **float** difference = (diffx + diffy + diffz) / 3;  
 Log.*e*(***TAG***, **" "** +difference);  
 **if**(difference > 0.5f){  
 **xValuee** = event.**values**[0];  
 **yValuee** = event.**values**[1];  
 **zValuee** = event.**values**[2];  
 **xValue**.setText(String.*valueOf*(**xValuee**));  
 **yValue**.setText(String.*valueOf*(**yValuee**));  
 **zValue**.setText(String.*valueOf*(**zValuee**));  
 *//Individual Task 2* **if**(**xValuee** < 0){  
 **xPos**.setText(**"Left side up"**);  
 }**else if**(**xValuee** > 0){  
 **xPos**.setText(**"Right side up"**);  
 }  
 **if**(**yValuee** < 0){  
 **yPos**.setText(**"Bottom side up"**);  
 }**else if**(**yValuee** > 0){  
 **yPos**.setText(**"Top side up"**);  
 }  
 **if**(**zValuee** < 0){  
 **zPos**.setText(**"Back side up"**);  
 }**else if**(**zValuee** > 0){  
 **zPos**.setText(**"Screen side up"**);  
 }  
 }  
 }  
 }  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** accuracy){  
 }  
 @Override  
 **protected void** onPause(){  
 **super**.onPause();  
 **if**(**senAccelerometer** != **null**)  
 **senSensorManager**.unregisterListener(MainActivity.**this**, **senAccelerometer**);  
  
 **if**(ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***){  
 **return**;  
 }  
 **this**.**locationManager**.removeUpdates(**this**);  
 stopBackgroundThread();  
  
 }  
  
 @Override  
 **protected void** onResume(){  
 **super**.onResume();  
  
 **if**(**senAccelerometer** != **null** && **InformationObtained**)  
 **senSensorManager**.registerListener(MainActivity.**this**, **senAccelerometer**, SensorManager.***SENSOR\_DELAY\_NORMAL***);  
  
 **if**(ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***){  
 **return**;  
 }  
 **this**.**locationManager**.requestLocationUpdates(LocationManager.***GPS\_PROVIDER***, 400, 1, **this**);  
 startBackgroundThread();  
 **if**(**textureView**.isAvailable()){  
 openCamera();  
 }**else**{  
 **textureView**.setSurfaceTextureListener(**textureListener**);  
 }  
 }  
  
 *//Part 2 methods* **public void** onLocationChanged(Location location){  
 **if**(location != **null**){  
 **if**(**gpsLocation**.getProvider() == location.getProvider()){  
 **gpsLocation** = location;  
 **coordinates**.setText(getString(R.string.***latitude\_text***) +**" "** +**gpsLocation**.getLatitude() +**" "** + getString(R.string.***longitude\_text***) +**" "** +**gpsLocation**.getLongitude());  
 }**else if**(**networkLocation**.getProvider() == location.getProvider()){  
 **networkLocation** = location;  
 **coordinatesNetwork**.setText(getString(R.string.***latitude\_text***) +**" "** +**networkLocation**.getLatitude() +**" "** + getString(R.string.***longitude\_text***) +**" "** +**networkLocation**.getLongitude());  
 }  
 }  
 }  
 @Override  
 **public void** onStatusChanged(String provider, **int** status, Bundle extras){  
 }  
  
 @Override  
 **public void** onProviderEnabled(String provider){  
 }  
  
 @Override  
 **public void** onProviderDisabled(String provider){  
 }  
 **private void** getLastLocation() {  
 **if**(ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***){  
 **return**;  
 }  
 **gpsLocation** = **locationManager**.getLastKnownLocation(LocationManager.***GPS\_PROVIDER***);  
 **networkLocation** = **locationManager**.getLastKnownLocation(LocationManager.***NETWORK\_PROVIDER***);  
 **coordinates**.setText(getString(R.string.***latitude\_text***) +**" "** +**gpsLocation**.getLatitude() +**" "** + getString(R.string.***longitude\_text***) +**" "** +**gpsLocation**.getLongitude());  
 **coordinatesNetwork**.setText(getString(R.string.***latitude\_text***) +**" "** +**networkLocation**.getLatitude() +**" "** + getString(R.string.***longitude\_text***) +**" "** +**networkLocation**.getLongitude());  
 }  
  
  
  
  
  
  
  
 *//Part 3 methods CAMERA STUFF BELOW* TextureView.SurfaceTextureListener **textureListener** = **new** TextureView.SurfaceTextureListener() {  
 @Override  
 **public void** onSurfaceTextureAvailable(SurfaceTexture surface, **int** width, **int** height) {  
 openCamera();  
 }  
 @Override  
 **public void** onSurfaceTextureSizeChanged(SurfaceTexture surface, **int** width, **int** height) {  
  
 }  
 @Override  
 **public boolean** onSurfaceTextureDestroyed(SurfaceTexture surface) {  
 **return false**;  
 }  
 @Override  
 **public void** onSurfaceTextureUpdated(SurfaceTexture surface) {  
 }  
 };  
 **private final** CameraDevice.StateCallback **stateCallback** = **new** CameraDevice.StateCallback() {  
 @Override  
 **public void** onOpened(CameraDevice camera) {  
 Log.*e*(***TAG***, **"onOpened"**);  
 **cameraDevice** = camera;  
 createCameraPreview();  
 }  
  
 @Override  
 **public void** onDisconnected(@NonNull CameraDevice camera) {  
 **cameraDevice**.close();  
 }  
  
 @Override  
 **public void** onError(@NonNull CameraDevice camera, **int** error) {  
 **cameraDevice**.close();  
 **cameraDevice** = **null**;  
 }  
 };  
  
 **final** CameraCaptureSession.CaptureCallback **captureCallbackListener** = **new** CameraCaptureSession.CaptureCallback() {  
 @Override  
 **public void** onCaptureCompleted(CameraCaptureSession session, CaptureRequest request, TotalCaptureResult result) {  
 **super**.onCaptureCompleted(session, request, result);  
 Toast.*makeText*(MainActivity.**this**, **"Saved:"** + **file**, Toast.***LENGTH\_SHORT***).show();  
 createCameraPreview();  
 }  
 };  
  
 **protected void** startBackgroundThread(){  
 **mBackgroundThread** = **new** HandlerThread(**"Camera Background"**);  
 **mBackgroundThread**.start();  
 **mBackgroundHandler** = **new** Handler(**mBackgroundHandler**.getLooper());  
 }  
  
 **protected void** stopBackgroundThread(){  
 **mBackgroundThread**.quitSafely();  
 **try**{  
 **mBackgroundThread**.join();  
 **mBackgroundThread** = **null**;  
 **mBackgroundHandler** = **null**;  
 }**catch** (InterruptedException e){  
 e.printStackTrace();  
 }  
 }  
  
 **protected void** takePicture(){  
 **if**(**null** == **cameraDevice**){  
 Log.*e*(***TAG***, **"cameraDevice is null"**);  
 **return**;  
 }  
  
 CameraManager manager = (CameraManager) getSystemService(Context.***CAMERA\_SERVICE***);  
  
 **try**{  
 CameraCharacteristics characteristics = manager.getCameraCharacteristics(**cameraDevice**.getId());  
 Size[] jpegSizes = **null**;  
 **if**(characteristics != **null**){  
 jpegSizes = characteristics.get(CameraCharacteristics.***SCALER\_STREAM\_CONFIGURATION\_MAP***).getOutputSizes(ImageFormat.***JPEG***);  
 }  
 **int** width = 640;  
 **int** height = 480;  
 **if**(jpegSizes != **null** && 0 < jpegSizes.**length**){  
 width = jpegSizes[0].getWidth();  
 height = jpegSizes[0].getHeight();  
 }  
 ImageReader reader = ImageReader.*newInstance*(width, height, ImageFormat.***JPEG***, 1);  
 List<Surface> outputSurfaces = **new** ArrayList<Surface>(2);  
 outputSurfaces.add(reader.getSurface());  
 outputSurfaces.add(**new** Surface(**textureView**.getSurfaceTexture()));  
 **final** CaptureRequest.Builder captureBuilder = **cameraDevice**.createCaptureRequest(CameraDevice.***TEMPLATE\_STILL\_CAPTURE***);  
 captureBuilder.addTarget(reader.getSurface());  
 *//Overall mode of 3A* captureBuilder.set(CaptureRequest.***CONTROL\_MODE***, CameraMetadata.***CONTROL\_MODE\_AUTO***);  
 *//Orientation* **int** rotation = getWindowManager().getDefaultDisplay().getRotation();  
 captureBuilder.set(CaptureRequest.***JPEG\_ORIENTATION***, ***ORIENTATIONS***.get(rotation));  
 *//Output file* **final** File file = **new** File(Environment.*getExternalStorageDirectory*()+**"/pic.jpg"**);  
  
 ImageReader.OnImageAvailableListener readerListener = **new** ImageReader.OnImageAvailableListener() {  
 @Override  
 **public void** onImageAvailable(ImageReader reader) {  
 Image image = **null**;  
 **try**{  
 image = reader.acquireLatestImage();  
 ByteBuffer buffer = image.getPlanes()[0].getBuffer();  
 **byte**[] bytes = **new byte**[buffer.capacity()];  
 buffer.get(bytes);  
 save(bytes);  
 } **catch**(FileNotFoundException e){  
 e.printStackTrace();  
 }**catch** (IOException e){  
 e.printStackTrace();  
 }**finally** {  
 **if**(image != **null**){  
 image.close();  
 }  
 }  
 }  
 **private void** save(**byte**[] bytes) **throws** IOException{  
 OutputStream output = **null**;  
 **try**{  
 *//save to file* output = **new** FileOutputStream(file);  
 output.write(bytes);  
 }**finally** {  
 **if**(**null** != output){  
 output.close();  
 }  
 }  
 }  
 };  
  
 reader.setOnImageAvailableListener(readerListener, **mBackgroundHandler**);  
  
 *//This callback is ivoked when a request triggers a capture to start, and when the capture is complete.* **final** CameraCaptureSession.CaptureCallback captureListener = **new** CameraCaptureSession.CaptureCallback() {  
 @Override  
 **public void** onCaptureCompleted(CameraCaptureSession session, CaptureRequest request, TotalCaptureResult result) {  
 **super**.onCaptureCompleted(session, request, result);  
 Toast.*makeText*(MainActivity.**this**, **"Saved:"** + file, Toast.***LENGTH\_SHORT***).show();  
 createCameraPreview();  
 }  
 };  
  
 **cameraDevice**.createCaptureSession(outputSurfaces, **new** CameraCaptureSession.StateCallback(){  
 @Override  
 **public void** onConfigured(CameraCaptureSession session){  
 **try**{  
 session.capture(captureBuilder.build(), captureListener, **mBackgroundHandler**);  
 } **catch** (CameraAccessException e){  
 e.printStackTrace();  
 }  
 }  
 @Override  
 **public void** onConfigureFailed(CameraCaptureSession session){  
  
 }  
 }, **mBackgroundHandler**);  
 }**catch** (CameraAccessException e){  
 e.printStackTrace();  
 }  
 }  
 **protected void** createCameraPreview(){  
 **try**{  
 SurfaceTexture texture = **textureView**.getSurfaceTexture();  
 **assert** texture != **null**;  
 texture.setDefaultBufferSize(**imageDimension**.getWidth(), **imageDimension**.getHeight());  
 Surface surface = **new** Surface(texture);  
 **captureRequestBuilder** = **cameraDevice**.createCaptureRequest(CameraDevice.***TEMPLATE\_PREVIEW***);  
 **captureRequestBuilder**.addTarget(surface);  
 **cameraDevice**.createCaptureSession(Arrays.*asList*(surface), **new** CameraCaptureSession.StateCallback() {  
 @Override  
 **public void** onConfigured(@NonNull CameraCaptureSession cameraCaptureSession) {  
 **if**(**null** == **cameraDevice**){  
 **return**;  
 }  
  
 **cameraCaptureSessions** = cameraCaptureSession;  
 updatePreview();  
 }  
  
 @Override  
 **public void** onConfigureFailed(@NonNull CameraCaptureSession session) {  
 Toast.*makeText*(MainActivity.**this**, **"Configuration change"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }, **null**);  
 } **catch** (CameraAccessException e){  
 e.printStackTrace();  
 }  
 }  
  
 **private void** openCamera(){  
 CameraManager manager = (CameraManager) getSystemService(Context.***CAMERA\_SERVICE***);  
 Log.*e*(***TAG***, **"is camera open"**);  
  
 **try**{  
 **cameraId** = manager.getCameraIdList()[0];  
 CameraCharacteristics characteristics = manager.getCameraCharacteristics(**cameraId**);  
 StreamConfigurationMap map = characteristics.get(CameraCharacteristics.***SCALER\_STREAM\_CONFIGURATION\_MAP***);  
 **assert** map != **null**;  
 **imageDimension** = map.getOutputSizes(SurfaceTexture.**class**)[0];  
  
 **if**(ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***CAMERA***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***) != PackageManager.***PERMISSION\_GRANTED***){  
 ActivityCompat.*requestPermissions*(MainActivity.**this**,  
 **new** String[]{Manifest.permission.***CAMERA***, Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***}, ***REQUEST\_CAMERA\_PERMISION***);  
 **return**;  
 }  
 manager.openCamera(**cameraId**, **stateCallback**, **null**);  
 }**catch** (CameraAccessException e){  
 e.printStackTrace();  
 }  
 Log.*e*(***TAG***, **"open camer X"**);  
 }  
  
 **protected void** updatePreview(){  
 **if**(**null** == **cameraDevice**){  
 Log.*e*(***TAG***, **"updatePreview error, return"**);  
 }  
 **captureRequestBuilder**.set(CaptureRequest.***CONTROL\_MODE***, CameraMetadata.***CONTROL\_MODE\_AUTO***);  
 **try**{  
 **cameraCaptureSessions**.setRepeatingRequest(**captureRequestBuilder**.build(), **null**, **mBackgroundHandler**);  
 }**catch**(CameraAccessException e){  
 e.printStackTrace();  
 }  
 }  
  
 **private void** closeCamera(){  
 **if**(**null** != **cameraDevice**){  
 **cameraDevice**.close();  
 **cameraDevice** = **null**;  
 }  
 **if**(**null** != **imageReader**){  
 **imageReader**.close();  
 **imageReader** = **null**;  
 }  
 }  
  
 @Override  
 **public void** onRequestPermissionsResult(**int** requestCode, @NonNull String[] permissions, @NonNull **int**[] grantResults){  
 **if**(requestCode == ***REQUEST\_CAMERA\_PERMISION***){  
 **if**(grantResults[0] == PackageManager.***PERMISSION\_DENIED***){  
 Toast.*makeText*(MainActivity.**this**, **"You cant use this app without granting permission"**, Toast.***LENGTH\_SHORT***).show();  
 finish();  
 }  
 }  
 }  
}

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**TableLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**TableRow  
 android:padding="10dp"**>  
 <**TextView  
 android:text="@string/x"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingEnd="5dp"  
 android:paddingRight="5dp"** />  
  
 <**TextView  
 android:id="@+id/x\_value"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 <**TextView  
 android:id="@+id/x\_pos"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**TableRow**>  
  
 <**TableRow  
 android:padding="10dp"**>  
  
 <**TextView  
 android:text="@string/y"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingEnd="5dp"  
 android:paddingRight="5dp"** />  
  
 <**TextView  
 android:id="@+id/y\_value"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 <**TextView  
 android:id="@+id/y\_pos"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**TableRow**>  
  
 <**TableRow  
 android:padding="10dp"**>  
  
 <**TextView  
 android:text="@string/z"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingEnd="5dp"  
 android:paddingRight="5dp"** />  
  
 <**TextView  
 android:id="@+id/z\_value"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 <**TextView  
 android:id="@+id/z\_pos"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**TableRow**>  
  
 </**TableLayout**>  
  
 <**Button  
 android:id="@+id/start\_and\_stop"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/start"**/>  
  
 <**Button  
 android:id="@+id/compass"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/compass"**/>  
  
 <**TableLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**TableRow  
 android:padding="10dp"**>  
  
 <**TextView  
 android:text="@string/Coordinates\_text"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingEnd="5dp"  
 android:paddingRight="5dp"** />  
  
 <**TextView  
 android:id="@+id/coordinates"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**TableRow**>  
  
 <**TableRow  
 android:padding="10dp"**>  
  
 <**TextView  
 android:text="@string/CoordinatesNET\_text"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingEnd="5dp"  
 android:paddingRight="5dp"** />  
  
 <**TextView  
 android:id="@+id/coordinates\_network"  
 android:text="-"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**TableRow**>  
  
 </**TableLayout**>  
  
 <**Button  
 android:id="@+id/take\_photo"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Take Photo"**/>  
  
 <**FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**TextureView  
 android:id="@+id/textureView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** />  
  
 </**FrameLayout**>  
  
  
  
</**LinearLayout**>

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#fff"** >  
  
 <**TextView  
 android:id="@+id/tvHeading"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginBottom="40dp"  
 android:layout\_marginTop="20dp"  
 android:text="Heading: 0.0"** />  
  
 <**ImageView  
 android:id="@+id/imageViewCompass"  
 android:layout\_width="150dp"  
 android:layout\_height="150dp"  
 android:layout\_margin="50dp"  
 android:src="@drawable/compass"** />  
  
</**RelativeLayout**>