**2,3,4**

if (ActivityCompat.*checkSelfPermission*(this, Manifest.permission.*POST\_NOTIFICATIONS*) != PackageManager.*PERMISSION\_GRANTED*) {  
 ActivityCompat.*requestPermissions*(MainActivity.this, new String[]{Manifest.permission.*POST\_NOTIFICATIONS*}, PackageManager.*PERMISSION\_GRANTED*);  
 return;  
}

@Override  
public void onRequestPermissionsResult(int requestCode, String[] permission, int[] grand) {  
 super.onRequestPermissionsResult(requestCode, permission, grand);  
 if (grand.length > 0) {  
 IntentFilter filter = new IntentFilter(Intent.*ACTION\_TIMEZONE\_CHANGED*);  
 registerReceiver(new timeZone(), filter);  
 }  
}

public class timeZone extends BroadcastReceiver {  
 private static final String *CHANNEL\_ID* = "TimeZoneChannel";  
 private static final int *NOTIFICATION\_ID* = 1;  
  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 if (Intent.*ACTION\_TIMEZONE\_CHANGED*.equals(intent.getAction())) {  
 NotificationManager notificationManager = (NotificationManager) context.getSystemService(Context.*NOTIFICATION\_SERVICE*);  
  
 NotificationChannel channel = new NotificationChannel(*CHANNEL\_ID*, "Time Change Channel", NotificationManager.*IMPORTANCE\_DEFAULT*);  
 channel.setDescription("Channel for time changes");  
 channel.enableLights(true);  
 channel.setLightColor(Color.*RED*);  
 notificationManager.createNotificationChannel(channel);  
  
 Notification.Builder builder = new Notification.Builder(context, *CHANNEL\_ID*)  
 .setContentTitle("Time Zone Changed")  
 .setContentText("Часовой пояс был изменен")  
 .setSmallIcon(android.R.drawable.*ic\_dialog\_alert*);  
  
 notificationManager.notify(*NOTIFICATION\_ID*, builder.build());  
 }  
 }  
}

<uses-permission android:name="android.permission.POST\_NOTIFICATIONS" />  
<uses-permission android:name="android.permission.TIME\_ZONE\_CHANGED" />

<receiver android:name=".timeZone"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.TIMEZONE\_CHANGED"/>  
 </intent-filter>  
</receiver>

**1**

boolean isAirplaneModeOn = intent.getBooleanExtra("state", false);

if (android.os.Build.VERSION.*SDK\_INT* >= android.os.Build.VERSION\_CODES.*O*) {  
 NotificationChannel channel = new NotificationChannel(*CHANNEL\_ID*, "Airplane Mode Channel", NotificationManager.*IMPORTANCE\_DEFAULT*);  
 notificationManager.createNotificationChannel(channel);  
}  
  
Notification.Builder builder = new Notification.Builder(context, *CHANNEL\_ID*)  
 .setContentTitle("Airplane Mode")  
 .setContentText(isAirplaneModeOn ? "Включен" : "Выключен")  
 .setSmallIcon(android.R.drawable.*ic\_dialog\_info*);  
  
notificationManager.notify(*NOTIFICATION\_ID*, builder.build());

**6**

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION"/>  
<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>

public class MainActivity extends AppCompatActivity {  
  
 TextView textView;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 textView = findViewById(R.id.*text1*);

if (ActivityCompat.*checkSelfPermission*(this, android.Manifest.permission.*ACCESS\_FINE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED* && ActivityCompat.*checkSelfPermission*(this, android.Manifest.permission.*ACCESS\_COARSE\_LOCATION*) != PackageManager.*PERMISSION\_GRANTED*) {  
 ActivityCompat.*requestPermissions*(MainActivity.this,  
 new String[]{Manifest.permission.*ACCESS\_FINE\_LOCATION*}, PackageManager.*PERMISSION\_GRANTED*);  
 return;  
 }  
  
 }  
 @SuppressLint("MissingPermission")  
 @Override  
 public void onRequestPermissionsResult(int requestCode, String[] permission, int[] grand) {  
 if (grand.length > 0) {  
 LocationManager location = (LocationManager) getSystemService(*LOCATION\_SERVICE*);  
 location.getCurrentLocation(LocationManager.*GPS\_PROVIDER*,null, getApplication().getMainExecutor(), new Consumer<Location>() {  
 @Override  
 public void accept(Location location) {  
 textView.setText(String.*valueOf*( "Широта: " + location.getLatitude() + "\nДолгота: " + location.getLongitude()));  
 }  
 });  
  
 }  
 super.onRequestPermissionsResult(requestCode, permission, grand);  
 }  
}

**5**

public class MainActivity extends AppCompatActivity {  
 private ImageView photoImageView;  
 private Button getPhotoButton;  
 private ActivityResultLauncher<Intent> resultLauncher = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(), new ActivityResultCallback<ActivityResult>() {  
 @Override  
 public void onActivityResult(ActivityResult result) {  
 if(result.getResultCode() == *RESULT\_OK*){  
 Intent data = result.getData();  
 Bitmap photo = (Bitmap) data.getExtras().get("data");  
 photoImageView.setImageBitmap(photo);  
 }  
 }  
 });  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 photoImageView = findViewById(R.id.*imageView*);  
 getPhotoButton = findViewById(R.id.*button*);  
  
 getPhotoButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 OpenCamera();  
 }  
 });  
 }  
  
 private void OpenCamera(){  
 Intent cameraIntent = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 resultLauncher.launch(cameraIntent);  
 }  
}