AGENDA 8

LOCALIZAÇÃO

GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO MOBILE II

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

São Paulo – SP, 2020

MERGULHANDO NO TEMA...

GPS (Global Positioning System)

Os sistemas de localização já fazem parte da nossa rotina através dos aplicativos para smartphone. Muitas vezes não percebemos que utilizamos, e muitas pessoas desconhecem, mas utilizam os serviços proporcionados pelo GPS.

Um exemplo é quando enviamos nossa localização através de um aplicativo de mensagens para nosso amigo. Para que essa localização seja enviada, primeiramente nosso dispositivo efetuou uma busca e localizou as coordenadas geográficas que este aparelho se encontra. E para localizar essas coordenadas, o dispositivo utiliza o sistema de GPS, que são informações obtidas através de satélites e processadas para estipular uma determinada localização.

O sistema de GPS é controlado pelo Governo Norte Americano, e é gratuito para utilização da população mundial. Essa utilização se tornou popular graças aos aparelhos smartphones, que introduziram este recurso, e hoje é um hardware básico em qualquer aparelho.

Localização no Android

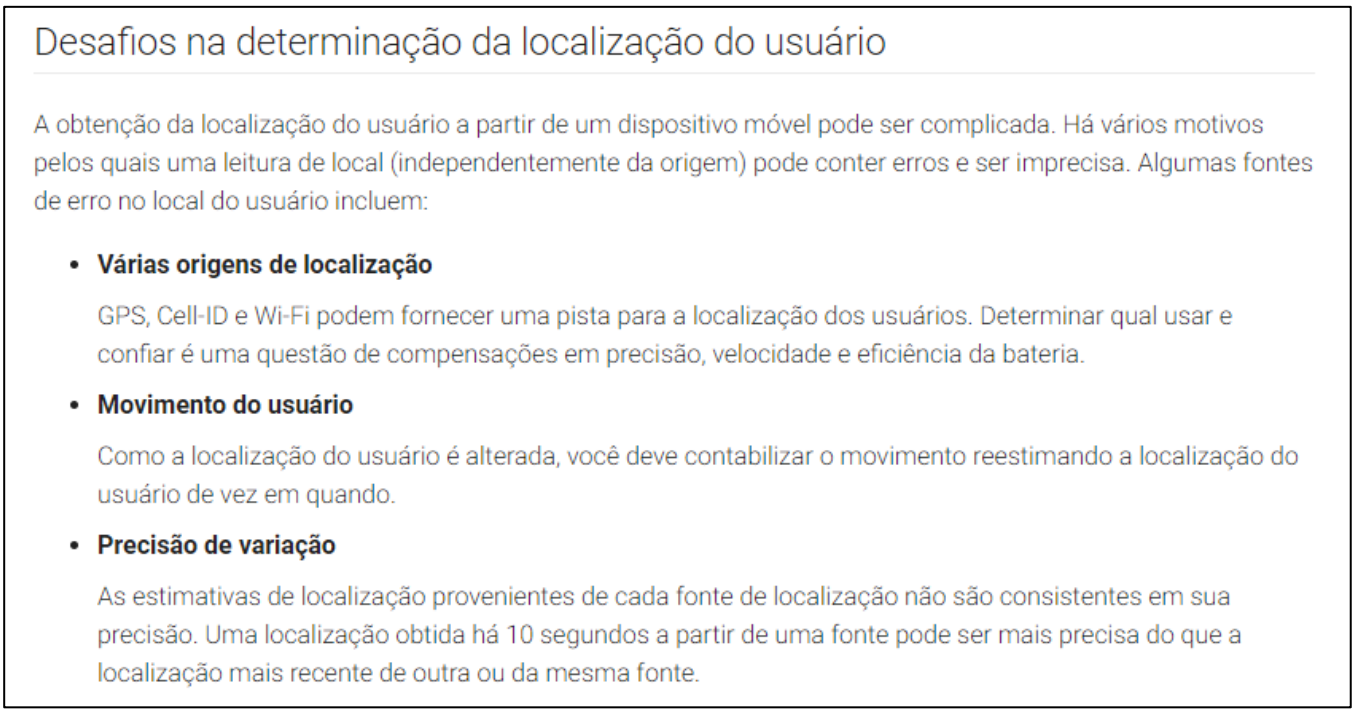
Segundo o site oficial de desenvolvimento do Android, disponível em: https://developer.android.com/guide/topics/location/strategies. “Saber onde o usuário está permite que seu aplicativo seja mais inteligente e forneça melhores informações ao usuário. Ao desenvolver um aplicativo com reconhecimento de local para Android, você pode utilizar o provedor de localização de rede do GPS e do Android para adquirir o local do usuário.

Embora o GPS seja mais preciso, ele só funciona ao ar livre, consome rapidamente a energia da bateria e não retorna à localização tão rapidamente quanto os usuários desejam. O provedor de localização de rede do Android determina a localização do usuário usando sinais de torre de celular e Wi-Fi, fornecendo informações de localização de maneira interna e externa, responde mais rapidamente e usa menos energia da bateria. Para obter a localização do usuário em seu aplicativo, você pode usar o GPS e o Network Location Provider, ou apenas um".

Quando o projeto requerer uma localização mais precisa do aparelho, é fundamental utilizar o GPS, em conjunto com o provedor do Android ou sozinho. Uma vez que o sistema de localização que não utiliza GPS, pode prover uma localização com uma margem de erro maior, ou seja, pode atribuir uma localização com um erro de quilômetros da realidade do dispositivo.

Quando utilizamos um aplicativo que contém as funções de localização, não imaginamos alguns desafios que o sistema necessita vencer para entregar uma posição mais real possível daquela determinada virtualmente.

A Figura 1 mostra alguns aspectos que podem influenciar na determinação de uma determinada localização geográfica para um aparelho smartphone.



Desafios na determinação da localização do usuário

A obtenção da localização do usuário a partir de um dispositivo móvel pode ser complicada. Há vários motivos pelos quais uma leitura de local (independentemente da origem) pode conter erros e ser imprecisa. Algumas fontes de erro no local do usuário incluem:

- **Várias origens de localização**

  GPS, Cell-ID e Wi-Fi podem fornecer uma pista para a localização dos usuários. Determinar qual usar e confiar é uma questão de compensações em precisão, velocidade e eficiência da bateria.

- **Movimento do usuário**

  Como a localização do usuário é alterada, você deve contabilizar o movimento reestimando a localização do usuário de vez em quando.

- **Precisão de variação**

  As estimativas de localização provenientes de cada fonte de localização não são consistentes em sua precisão. Uma localização obtida há 10 segundos a partir de uma fonte pode ser mais precisa do que a localização mais recente de outra ou da mesma fonte.

Figura 1 – Desafios na determinação de localização do usuário. Disponível em: https://developer.android.com/guide/topics/location/strategies.

Obter a localização através da classe LocationManager

O sistema Android possui classes para facilitar o desenvolvimento de aplicativos. Alguns recursos mais complexos contam com a assessoria delas para que o desenvolvedor ganhe tempo na produção dos projetos.

A ***LocationManager*** foi desenvolvida para que o aplicativo comunique com o sistema de localização do dispositivo, seja através do GPS ou pelo provedor de localizações.

A classe pode ser utilizada para solicitar em determinados intervalos de tempo a localização do aparelho, ou também pode ser programada para que quando o dispositivo entre em uma determinada área geográfica chame uma aplicação ou serviço.

Em nosso projeto vamos utilizar o método **requestLocationUpdates()**, ele é responsável por atualizar a localização toda vez que o dispositivo muda sua localização geográfica.

```
locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,  minTime: 0,  minDistance: 0, locationListener);
```

Figura 2 – RequestLocationUpdats().

A Figura 2, mostra o método **requestLocationUpdates()**, ele possui 4 parâmetros básicos. O primeiro é responsável por indicar qual sistema de localização do dispositivo vamos utilizar, no caso vamos trabalhar com o GPS do aparelho. O segundo parâmetro indica qual intervalo de tempo vamos solicitar a atualização da localização. O terceiro fornece ao método a distância mínima que ele deve se basear para atualizar a localização. Como informamos o valor zero, a cada mínima alteração, a atualização é chamada. A quarta e última, é onde informamos o LocationListener, fundamental para o funcionamento do código.

LocationListener

O **LocationListener** recebe as notificações da alteração da localização geográfica do aparelho, informadas pelo **LocationManager** através do **requestLocationUpdates().**

Ele trabalha com alguns métodos públicos como mostra a Figura 3, retirada do site do Android Studio, disponível em: https://developer.android.com/reference/android/location/LocationListener

onLocationChanged

added in API level 1

```
public abstract void onLocationChanged (Location location)
```

Called when the location has changed.

There are no restrictions on the use of the supplied Location object.

Parameters	
`location`	**`Location`**: The new location, as a Location object.

onProviderDisabled

adicionado no nível de API 1

```
public abstract void onProviderDisabled ( provedor de cadeia )
```

Chamado quando o provedor é desativado pelo usuário. Se requestLocationUpdates for chamado em um provedor já desativado, esse método será chamado imediatamente.

Parâmetros	
`provider`	**`String`**: o nome do provedor de localização associado a esta atualização.

onProviderEnabled

adicionado no nível de API 1

```
public abstract void onProviderEnabled ( provedor de cadeia )
```

Chamado quando o provedor é ativado pelo usuário.

Parâmetros	
`provider`	**`String`**: o nome do provedor de localização associado a esta atualização.

onStatusChanged

adicionado no nível de API 1

```
public abstract void onStatusChanged ( provedor de cadeia de caracteres ,
                status int
                Extras do pacote )
```

Chamado quando o status do provedor é alterado. Esse método é chamado quando um provedor não pode buscar um local ou se o provedor tiver se tornado disponível recentemente após um período de indisponibilidade.

Parâmetros	
`provider`	**`String`**: o nome do provedor de localização associado a esta atualização.
`status`	**`int`**: se o provedor estiver fora de serviço e não se espera que isso mude em um futuro próximo; se o provedor estiver temporariamente indisponível, mas espera-se que esteja disponível em breve; e se o provedor estiver disponível no momento.`LocationProvider.OUT_OF_SERVICE`LocationProvider.`TEMPORARILY_UNAVAILABLE`LocationProvider.`AVAILABLE`
`extras`	**`Bundle`**: um Pacote opcional que conterá variáveis de status específicas do provedor. Um número de pares de chave / valor comuns para o pacote de extras está listado abaixo. Os provedores que usam qualquer uma das chaves dessa lista devem fornecer o valor correspondente, conforme descrito abaixo. • satélites - o número de satélites usados para derivar a correção

Figura 3 – Métodos do LocationListener.

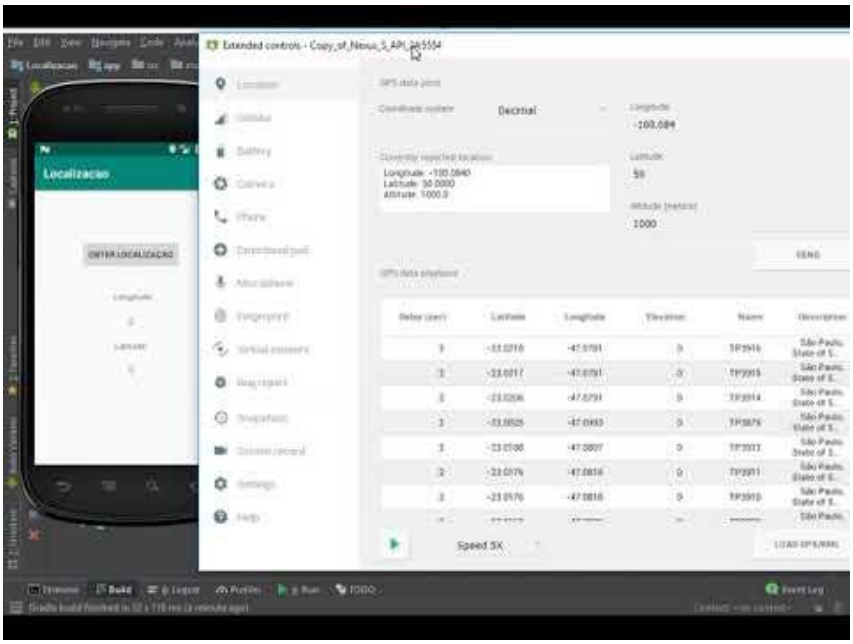
Permissões de uso para a localização do aparelho

Já vimos anteriormente que alguns recursos do aparelho são bloqueados para o aplicativo. Desta forma quando surge a necessidade de utilização, o aplicativo tem que solicitar ao usuário o seu consentimento para que ele tenha acesso.

O sistema de localização do aparelho vem bloqueado para os aplicativos. Vamos informar no arquivo **Manifest.xml** a nossa intenção de utilizar esse hardware do aparelho. Para isso, temos que criar uma rotina na nossa classe Java para solicitar, na primeira execução do aplicativo, a autorização do usuário para que nosso app receba as informações referentes a localização do dispositivo mobile.

Vamos acompanhar no vídeo a seguir o início do desenvolvimento do nosso projeto "**Localizacao**" que utiliza a localização do aparelho Android.

Projeto "Localizacao"



Verifique os códigos do arquivo "**AndroidManifest.xml**" e a classe Java do projeto "**MainActiviry.java**".

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.guilherme.testegps3" >


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

    <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>
```

Figura 4 – AndroidManifest.xml.

```
package com.example.guilherme.localizacao;

import android.Manifest;
import android.content.Context;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    Button btnGpsProg;
    TextView txtLatitudeProg, txtLongitudeProg;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        txtLatitudeProg = (TextView) findViewById(R.id.txtLatitude);
```

```
        txtLongitudeProg = (TextView) findViewById(R.id.txtLongitude);

        btnGpsProg = (Button) findViewById(R.id.btnGps);
        btnGpsProg.setOnClickListener(new Button.OnClickListener() {
            public void onClick(View v) {
                solicitarPermissao();
            }
        });
    }

    private void solicitarPermissao() {

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
                && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED)
        {

            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 1);
        }
        else
            ativarServicoGPS();
    }

    @Override
    public void onRequestPermissionsResult(int requestCode,
                                           String permissions[], int[]
grantResults) {
        switch (requestCode) {
            case 1: {
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    ativarServicoGPS();
                } else {
                    Toast.makeText(this, "Sem conexão com o GPS!",
Toast.LENGTH_LONG).show();
                }
                return;
            }
        }
    }

    public void ativarServicoGPS(){
        try {
            LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);

            LocationListener locationListener = new LocationListener() {
                public void onLocationChanged(Location location)
{escreverLocalizacao(location);}

                public void onStatusChanged(String provider, int status, Bundle
extras) { }

                public void onProviderEnabled(String provider) { }

                public void onProviderDisabled(String provider) { }
            };
            locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0,
```

```
0, locationListener);
        }catch(SecurityException ex){
            Toast.makeText(this, ex.getMessage(), Toast.LENGTH_LONG).show();
        }
    }

    public void escreverLocalizacao(Location location)
    {
        Double longitude = location.getLongitude();
        Double latitude = location.getLatitude();

        txtLongitudeProg.setText(longitude.toString());
        txtLatitudeProg.setText(latitude.toString());
    }
}
```

Figura 5 – MainActiviry.java.

Fontes Imagéticas:

Imagens: freepik.com, Pixabay , Arquivo do GEEaD