L'une des caractéristiques essentielles des téléphones mobiles est leur portabilité, et il n'est donc pas surprenant que quelques-unes des fonctionnalités les plus séduisantes d'Android soient les services permettant de déterminer, de contextualiser et de cartographier des positions géographiques.

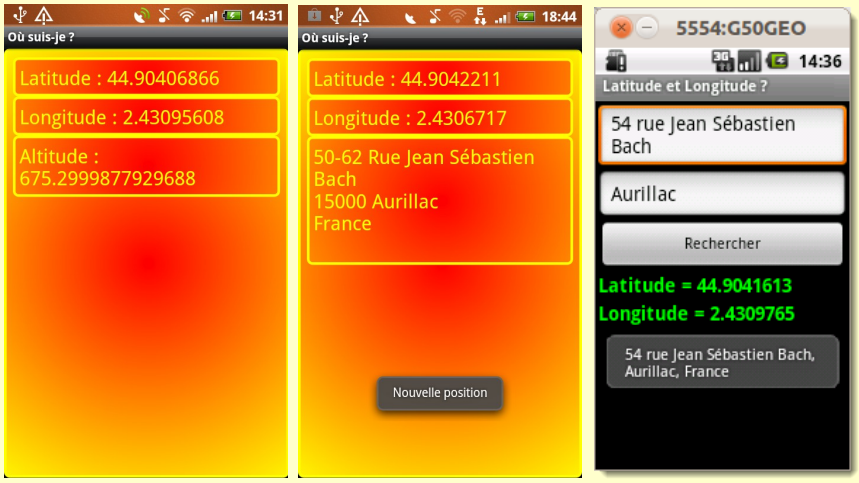
Nous couvrirons également dans ce chapitre les services de géolocalisation, qui permettent de déterminer la position courante de l'appareil. Ils incluent le GPS et la technologie Google de localisation cellulaire.

- Les cartes et les services de géolocalisation utilisent la **latitude** et la **longitude** pour identifier les positions géographiques, mais la plupart des utilisateurs raisonnent plutôt en terme d'adresses. Android fournit un **Geocoder** qui supporte les **géocodages** avant et inverse. Grâce à lui, vous pourrez convertir **latitudes** et **longitudes** en adresses et inversement.

*Utilisés conjointement, la **cartographie**, le **géocodage** et les **services de géolocalisation** fournissent une puissante boîte à outils pour incorporer la mobilité native de votre téléphone à vos applications. Les services de géolocalisation d'Android sont donc divisés en deux grandes parties :*

- **Les API qui gèrent les plans** (dans l'espace de noms ***com.google.android.maps***) ;
- **Les API qui gèrent la localisation** à proprement parler (dans l'espace de noms ***android.location***).

Nous allons eu cours de cette étude voir comment déterminer la position d'un appareil sur Android.



UTILISER LES SERVICES DE GÉOLOCALISATION

Les services de géolocalisation sont un terme générique désignant les différentes technologies utilisées pour déterminer la position courante d'un appareil. Les deux principaux éléments sont les suivants :

- **Location Manager** : Fournit des points d'entrée vers les services de géolocalisation.
- **Location Providers** : (fournisseurs de position) : Chacun d'eux représente une technologie de localisation de la position de l'appareil.

*A l'aide du **Location Manager** vous pouvez :*

- Obtenir une position courante.
- Suivre des déplacements.
- Déclencher des alertes de proximité en détectant les mouvements dans une zone spécifique.
- Trouver les **Location Providers** disponibles.

Lorsqu'il s'agit de déterminer la position courante de l'appareil, plusieurs problèmes peuvent être rencontrés :

- Tous les appareils ne disposent pas tous du même matériel de géolocalisation (certains n'ont pas de récepteur GPS) ;
- Les conditions d'utilisation du téléphone peuvent rendre inutilisable une méthode de localisation (cas de fonctionnalités GPS dans un tunnel, par exemple).

SÉLECTIONNER UN FOURNISSEUR DE POSITION - LOCATION PROVIDER

Android offre plusieurs moyens de localisation au travers d'une liste de fournisseurs de positions : selon les conditions du moment ou vos propres critères, Android se chargera de sélectionner le plus apte à donner la position de l'appareil. De manière générale, nous distinguons deux types de fournisseurs naturels :

- Le fournisseur basé sur la technologie GPS, de type **LocationManager.GPS_PROVIDER**. C'est le plus précis des deux, mais c'est également le plus consommateur en terme de batterie.
- Le fournisseur qui se repère grâce aux antennes des opérateurs mobiles et aux points d'accès WI-FI, de type **LocationManager.NETWORK_PROVIDER**.

```
// Pour obtenir une instance d'un provider spécifique, appelez la méthode getProvider() en lui passant son nom :
String fournisseur = LocationManager.GPS_PROVIDER;
LocationProvider gpsProvider = sysLocalisation.getProvider(fournisseur);
```

OBTENIR LA LISTE DES FOURNISSEURS DE POSITION

Le but des services de géolocalisation est de déterminer la position de l'appareil. L'accès à ces services est géré par le système au travers de la méthode ***getSystemService()***. La classe des fournisseurs de position, **LocationProvider**, est une classe abstraite : chacune des différentes implémentations (correspondant aux différents moyens de localisation à notre disposition) fournit l'accès aux informations de localisation d'une manière qui lui est propre.

```
// Le code suivant permet d'obtenir la liste de tous les fournisseurs :
LocationManager sysLocalisation = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
List<String> fournisseurs = sysLocalisation.getAllProviders();
```

A un instant donné, tous les outils de géolocalisation de l'appareil ne sont peut-être pas disponibles : l'utilisateur peut en effet décider de désactiver certains moyens (par exemple, le GPS peut être désactivé pour économiser les batteries).

- Une autre alternative permet ainsi d'obtenir la liste de tous les providers disponibles sur l'appareil en appelant cette fois-ci la méthode ***getProviders()*** en spécifiant un booléen pour indiquer si vous désirez toute la liste des fournisseurs ou uniquement ceux qui sont activés :

```
// Le code suivant permet d'obtenir la liste de tous les fournisseurs activés :
LocationManager sysLocalisation = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
boolean actifsSeulement = true;
List<String> fournisseurs = sysLocalisation.getProviders(actifsSeulement);
```

*Comme nous l'avons vu plus haut, si vous souhaitez obtenir un fournisseur particulier, appelez plutôt la méthode **getProvider()** en spécifiant le type de fournisseur à l'aide des constantes prédéfinies **LocationManager.GPS_PROVIDER** ou **LocationManager.NETWORK_PROVIDER** :*

```
LocationManager sysLocalisation = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
String fournisseur = LocationManager.GPS_PROVIDER;
LocationProvider gpsProvider = sysLocalisation.getProvider(fournisseur);
```

Comme la plupart des fonctionnalités sous Android, pour utiliser ces quelques lignes de code, il ne faut pas oublier de déclarer les permissions appropriées dans la section <manifest> du fichier de configuration de l'application.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" ...>
    <application android:label="@string/app_name" >
                ....
    </application>
    // L'utilisation du fournisseur de type GPS est lié à la déclaration de permission suivante :
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    // L'utilisation du fournisseur réseau dépend quant à lui de la permission :
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
</manifest>
```

*Je rappelle encore une fois que le but des services de géolocalisation est de déterminer la position de l'appareil. Une seule ligne de code suffit pour cela. Il faut cependant choisir auparavant un fournisseur de position. Une fois cette étape réalisée, nous utilisons la méthode **getLastKnownLocation()** issue de la classe **LocationManager**, avec comme paramètre le fournisseur de position choisi. Le retour de cette méthode est une instance d'un objet **Location** si une position a été calculée, sinon son résultat vaut **null** :*

```
LocationManager sysLocalisation = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
String fournisseur = LocationManager.GPS_PROVIDER;
Location localisation = sysLocalisation.getLastKnownLocation(fournisseur);
```

*L'objet de type **Location** renvoyé par la méthode **getLastKnownLocation()** inclut toutes les informations de position disponibles données par le fournisseur : latitude, longitude, cap, altitude, vitesse et heure à laquelle la position a été déterminée. Toutes ces propriétés sont accessibles via les méthodes **getXxx()** de l'objet **Location**. Je vous propose maintenant de prendre un premier projet de géolocalisation qui permet de récupérer la latitude, la longitude et l'altitude de votre mobile au démarrage de votre mobile.*

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="fr.btsiris.localisation"  android:versionCode="1"  android:versionName="1.0">
    <application android:label="GPS" >
        <activity android:name="Geolocalisation" android:label="Où suis-je ?">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
</manifest>
```

res/drawable/fond.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:shape="rectangle" >
          <stroke android:width="3dp" android:color="#FFFF00" />
          <corners android:radius="5dp" />
          <gradient android:startColor="#FF0000"
                    android:endColor="#FFFF00"
                    android:type="radial"
                    android:gradientRadius="300"  />
</shape>
```

res/layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="vertical"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:padding="10dp"
      android:background="@drawable/fond" >
      <TextView
            android:id="@+id/latitude"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textSize="22dp"
            android:textColor="#FFFF00"
            android:padding="7dp"
            android:background="@drawable/fond"  />
      <TextView
            android:id="@+id/longitude"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="#FFFF00"
            android:padding="7dp"
            android:textSize="22dp"
            android:background="@drawable/fond"  />
      <TextView
            android:id="@+id/altitude"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="#FFFF00"
            android:padding="7dp"
            android:textSize="22dp"
            android:background="@drawable/fond"  />
</LinearLayout>
```

Ici nous utilisons le GPS intégré, mais si vous ne possédez pas de capteur GPS dans votre mobile, prévoyez le type de fournisseur ***LocationManager.NETWORK_PROVIDER***.

fr.btsiris.localisation.Geolocalisation.java

```
package fr.btsiris.localisation;

import android.app.Activity;
import android.content.Context;
import android.location.*;
import android.os.Bundle;
import android.widget.TextView;

public class Geolocalisation extends Activity {
     private TextView latitude, longitude, altitude;

      @Override
      public void onCreate(Bundle savedInstanceState)  {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
            latitude = (TextView) findViewById(R.id.latitude);
            longitude = (TextView) findViewById(R.id.longitude);
            altitude = (TextView) findViewById(R.id.altitude);
      }
```

```
    @Override
    public void onStart()  {
        super.onStart();
        LocationManager localisations = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
        Location position = localisations.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        if (position!=null) {
            latitude.setText("Latitude : "+position.getLatitude());
            longitude.setText("Longitude : "+position.getLongitude());
            altitude.setText("Altitude : "+position.getAltitude());
        }
        else latitude.setText("Position  non déterminée ");
    }

}
```

✗ SUIVRE LES DÉPLACEMENTS

La plupart des applications de localisation doivent réagir aux déplacements de l'utilisateur. Se contenter de sonder le **LocationManager** ne suffira pas à forcer l'obtention des nouvelles mises à jour des **Location Providers**. Détecter le changement de position relève de deux problématiques : recevoir des mises à jour de sa position et détecter le mouvement.

*Ces deux problématiques de changement se résolvent par l'utilisation de la même méthode du **LocationManager**. Il s'agit de la méthode **requestLocationUpdates()** dont voici les paramètres utiles :*

```
LocationManager sysLocalisation = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
String fournisseur = LocationManager.GPS_PROVIDER;
sysLocalisation.requestLocationUpdates(fournisseur, temps, distance, écouteurLocalisation);
```

- ✗ Le premier paramètre est le fournisseur de position souhaité.
- ✗ Ensuite, le deuxième paramètre indique le temps entre deux mises à jour exprimé en millisecondes. Attention à ne pas mettre de valeur trop faibles, qui accaparerait les ressources du téléphone et viderait votre batterie. La documentation du SDK recommande une valeur minimale de **60 000 ms**.
- ✗ Le troisième paramètre précise la distance correspond au nombre de mètres qui doivent être parcourus avant de recevoir une nouvelle position. Si elle est supérieure à zéro, la mise à jour s'effectuera uniquement une fois la distance parcourue.
- ✗ Enfin, le dernier paramètre est l'écouteur (une implémentation de l'interface **LocationListener**) qui recevra les diverses notifications. Cet écouteur propose quatre méthodes que vous devez redéfinir suivant les circonstances :

```
class ChangementPosition implements LocationListener {
    public void onLocationChanged(Location position) {  // Met à jour l'application en fonction de la nouvelle position.    }
    public void onStatusChanged(String fournisseur, int statut, Bundle extras) {  // Met à jour l'application si le status du matériel a changé.  }
    public void onProviderEnabled(String fournisseur) {  // Met à jour l'application lorsque le fournisseur est activé.  }
    public void onProviderDisabled(String fournisseur) {  // Met à jour l'application lorsque le fournisseur est désactivé.  }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="fr.btsiris.localisation"  android:versionCode="1"  android:versionName="1.0">
      <application android:label="GPS" >
            <activity android:name="Geolocalisation" android:label="Où suis-je ?">
                  <intent-filter>
                        <action android:name="android.intent.action.MAIN" />
                        <category android:name="android.intent.category.LAUNCHER" />
                  </intent-filter>
            </activity>
      </application>
      <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
      <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
</manifest>
```

fr.btsiris.localisation.Geolocalisation.java

```
package fr.btsiris.localisation;

import android.app.Activity;
import android.content.Context;
import android.location.*;
```

```
import android.os.Bundle;
import android.widget.*;

public class Geolocalisation extends Activity {
    private TextView latitude, longitude, altitude;
    private LocationManager localisations;
    private String fournisseur;
    private ChangementPosition changements = new ChangementPosition();

     @Override
     public void onCreate(Bundle savedInstanceState)  {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.main);
          latitude = (TextView) findViewById(R.id.latitude);
          longitude = (TextView) findViewById(R.id.longitude);
           altitude = (TextView) findViewById(R.id.altitude);
          localisations = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
          miseAJourDeLaPosition(localisations.getLastKnownLocation(LocationManager.GPS_PROVIDER));
     }

     @Override
     protected void onStart() {
          super.onStart();
          localisations.requestLocationUpdates(LocationManager.GPS_PROVIDER, 2000, 2, changements);
          localisations.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 5000, 5, changements);
     }

     @Override
     protected void onStop() {
         super.onStop();
         localisations.removeUpdates(changements);
     }

     private void miseAJourDeLaPosition(Location position) {
          if (position!=null) {
               latitude.setText("Latitude : "+position.getLatitude());
               longitude.setText("Longitude : "+position.getLongitude());
               altitude.setText("Altitude : "+position.getAltitude());
          }
          else latitude.setText("Position  non déterminée ");
          Toast.makeText(Geolocalisation.this, "Nouvelle position", Toast.LENGTH_SHORT).show();
     }

     class ChangementPosition implements LocationListener {
           public void onLocationChanged(Location position) { miseAJourDeLaPosition(position);  }
           public void onStatusChanged(String fournisseur, int statut, Bundle extras) { }
           public void onProviderEnabled(String fournisseur) { }
           public void onProviderDisabled(String fournisseur) { }
     }
}
```

x CONVERSION D'ADRESSES ET D'ENDROITS - GÉOCODAGE

Le géocodage permet de déterminer des coordonnées en latitude et longitude à partir d'une adresse ou d'une description d'un endroit. A l'inverse, le géocodage permet de retrouver un lieu en fonction de ses coordonnées. Ces fonctionnalités sont liées à l'API Google.

*Ainsi, le géocodage, au travers de la classe **Geocoder**, permet de traduire automatiquement des adresses en coordonnées géographiques et inversement. Cela permet de connaître les positions utilisées par les services de géolocalisation et les activités géographiques. Les recherches de géocodage ont lieu sur le serveur et vos applications devront donc avoir la permission de se connecter à Internet :*

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" ...>
    <application android:label="@string/app_name" >
                ....
    </application>
    //  L'utilisation du fournisseur de type GPS est lié à la déclaration de permission suivante :
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    //  L'utilisation du géocodage dépend quant à lui de la permission :
    <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

*La classe **Geocoder** donne accès à deux fonctions de géocodage :*

- **Géocodage avant** : Détermine la latitude et la longitude d'une adresse.
- **Géocodage inverse** : Détermine l'adresse en fonction de la latitude et de la longitude.

*Le résultat de ces appels sont contextualisés par le biais d'une locale (utilisée pour définir votre emplacement et votre langue habituels). L'extrait qui suit montre comment la mettre en œuvre lors de la création du **Geocoder**. Si vous ne la spécifiez pas, celle de votre appareil sera utilisée par défaut.*

```
Geocoder geocodage = new Geocoder(this, Locale.FRANCE);

Geocoder geocodage = new Geocoder(this, Locale.getDefault());
```

*Les deux fonctions de géocodage renvoient une liste d'objets **Address**. Chacun peut contenir plusieurs résultats en fonction d'une limite que vous spécifiez lors de l'appel. Chaque objet **Address** est renseigné avec tous les détails que le géocoder a pu déterminer. Cela peut inclure la latitude, la longitude, un numéro de téléphone et des détails d'adresse de granularité de plus en plus fine allant du pays au numéro de la voie.*

GÉOCODAGE INVERSE

Le géocodage inverse renvoie des adresses en fonctions de lieux spécifiés par leur latitude et leur longitude. Il permet de reconnaître les positions renvoyées par les services de géolocalisation.

*Pour effectuer une recherche inversée, passez la latitude et la longitude à la méthode **getFromLocation()** du **Geocoder**. Elle renverra une liste des correspondances possibles d'adresses. Si le **Geocoder** ne peut déterminer aucune adresse pour les coordonnées spécifiées, il renverra la valeur **null**.*

- Voici un exemple ci-dessous que permet de spécifier l'adresse complète de la localité où nous nous trouvons en plus des coordonnées géographiques :

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="fr.btsiris.localisation"  android:versionCode="1"  android:versionName="1.0">
    <application android:label="GPS" >
        <activity android:name="Geolocalisation" android:label="Où suis-je ?">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

res/layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="10dp"
    android:background="@drawable/fond" >
    <TextView
        android:id="@+id/latitude"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="22dp"
        android:textColor="#FFFF00"
        android:padding="7dp"
        android:background="@drawable/fond"  />
    <TextView
        android:id="@+id/longitude"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="#FFFF00"  android:padding="7dp"
        android:textSize="22dp"  android:background="@drawable/fond"  />
    <TextView
        android:id="@+id/adresse"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="#FFFF00"  android:padding="7dp"
        android:textSize="22dp"  android:background="@drawable/fond"  />
</LinearLayout>
```

fr.btsiris.localisation.Geolocalisation.java


```
package fr.btsiris.localisation;

import android.app.Activity;
import android.content.Context;
import android.location.*;
import android.os.Bundle;
import android.widget.*;
import java.io.IOException;
import java.util.*;

public class Geolocalisation extends Activity {
    private TextView latitude, longitude, adresse;
    private LocationManager localisations;
    private ChangementPosition changements = new ChangementPosition();
    private Geocoder geocodage;

    @Override
     public void onCreate(Bundle savedInstanceState)  {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.main);
          latitude = (TextView) findViewById(R.id.latitude);
          longitude = (TextView) findViewById(R.id.longitude);
          adresse =  (TextView) findViewById(R.id.adresse);
          geocodage = new Geocoder(this, Locale.FRANCE);
          localisations = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
          miseAJourDeLaPosition(localisations.getLastKnownLocation(LocationManager.GPS_PROVIDER));
     }

     @Override
     protected void onStart() {
          super.onStart();
          localisations.requestLocationUpdates(LocationManager.GPS_PROVIDER, 2000, 2, changements);
          localisations.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 5000, 5, changements);
     }

     @Override
     protected void onStop() {
         super.onStop();
         localisations.removeUpdates(changements);
     }

     private void miseAJourDeLaPosition(Location position) {
         if (position!=null) {
             double lat = position.getLatitude();
             double lng = position.getLongitude();
             latitude.setText("Latitude : "+lat);
             longitude.setText("Longitude : "+lng);
             try {
                 List<Address> adresses = geocodage.getFromLocation(lat, lng, 1);
                 StringBuilder texte = new StringBuilder();
                 if (adresses.size() > 0) {
                     Address une = adresses.get(0);
                     texte.append(une.getAddressLine(0)).append("\n");
                     texte.append(une.getPostalCode()).append(" ").append(une.getLocality()).append("\n");
                     texte.append(une.getCountryName()).append("\n");
                     adresse.setText(texte.toString());
                 }
             }
             catch (IOException ex) { }
         }
         else latitude.setText("Position  non déterminée ");
         Toast.makeText(Geolocalisation.this, "Nouvelle position", Toast.LENGTH_SHORT).show();
     }

     class ChangementPosition implements LocationListener {
          public void onLocationChanged(Location position) { miseAJourDeLaPosition(position);  }
          public void onStatusChanged(String fournisseur, int statut, Bundle extras) { }
          public void onProviderEnabled(String fournisseur) { }
          public void onProviderDisabled(String fournisseur) { }
     }
}
```

GÉOCODAGE AVANT

Le géocodage avant (ou tout simplement géocodage) détermine cette fois-ci les coordonnées géographique d'un lieu à partir d'un texte décrivant un endroit. Le format des adresses, noms de stations ou codes postaux dépend évidemment de la zone géographique concernée.

Ce que nous appelons lieu valide varie effectivement en fonction de la zone géographique. Il s'agira en général d'une adresse normale de granularité variable (depuis le pays jusqu'au numéro dans la voie), d'un code postal, d'une gare, d'un monument, d'un hôpital ou, de façon générale, de tout ce qui peut être utilisé dans une recherche sur Google Maps.

- Pour effectuer un géocodage, appelez la méthode ***getFromLocationName()*** sur une instance de **Geocoder**. Passez-lui le lieu dont vous souhaitez obtenir les coordonnées ainsi que le nombre maximal de résultats :

```
Geocoder geocodage = new Geocoder(this, Locale.FRANCE);

List<Address> positions = geocodage.getFromLocationName(adresse, 1);
```

La liste renvoyée peut inclure plusieurs correspondances possibles pour le lieu indiqué. Chaque résultat inclura une latitude et une longitude ainsi que toutes les informations d'adresse disponibles. Ceci est utile pour confirmer que la position a été correctement résolue ainsi que pour fournir des informations spécifiques lors de recherches de monuments.

- Comme pour le géocodage inverse, **null** est renvoyé si aucune correspondance n'est trouvée. De même, la disponibilité, l'exactitude et la granularité des résultats dépendent entièrement des données disponibles dans la base.
- Lorsque vous effectuez des recherches inversées, l'objet **Locale** spécifié lors de la création du **Geocoder** est particulièrement important. La **Locale** fournit le contexte géographique pour l'interprétation des résultats de recherche car un nom de lieu peut exister à plusieurs endroits. Lorsque c'est possible, sélectionnez une **Locale** régionale pour éviter les ambiguïtés.

Voici un autre exemple qui permet de retrouver la latitude et la longitude à partir d'une adresse :

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="fr.btsiris.localisation"
        android:versionCode="1"
        android:versionName="1.0">
    <application android:label="Localisation" >
        <activity android:name="Localisation" android:label="Latitude et Longitude ?">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

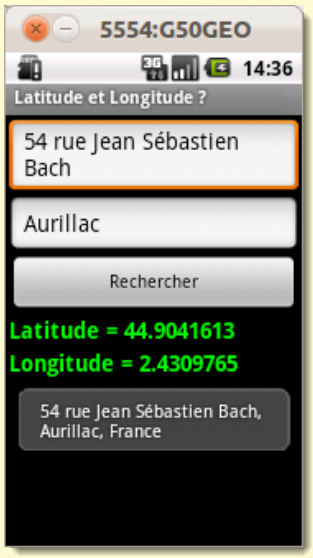
res/layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="3px">
  <EditText
    android:id="@+id/rue"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Rue" />
  <EditText
    android:id="@+id/ville"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Ville" />
  <Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Rechercher"
    android:onClick="localiser"/>
  <TextView
    android:id="@+id/latitude"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textStyle="bold"
    android:textColor="#00FF00"
```

```
        android:textSize="18sp" />
    <TextView
        android:id="@+id/longitude"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:textColor="#00FF00"
        android:textSize="18sp" />
</LinearLayout>
```

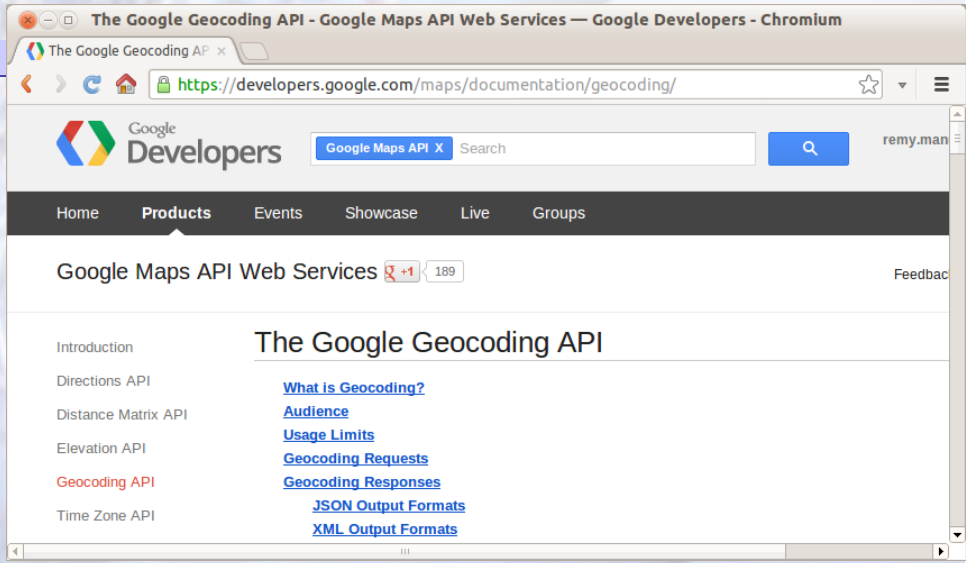
fr.btsiris.localisation.Localisation.java

```
package fr.btsiris.localisation;

import android.app.Activity;
import android.location.*;
import android.os.Bundle;
import android.view.View;
import android.widget.*;
import java.io.IOException;
import java.util.*;

public class Localisation extends Activity {
    private TextView latitude, longitude;
    private EditText rue, ville;
    private Geocoder geocodage;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        latitude = (TextView) findViewById(R.id.latitude);
        longitude = (TextView) findViewById(R.id.longitude);
        rue =  (EditText) findViewById(R.id.rue);
        ville =  (EditText) findViewById(R.id.ville);
        geocodage = new Geocoder(this, Locale.FRANCE);
    }

    public void localiser(View vue) {
        String adresse = rue.getText().toString() + ", " + ville.getText().toString() + ", France";
        try {
            List<Address> positions = geocodage.getFromLocationName(adresse, 1);
            if (positions.size() > 0) {
                Toast.makeText(this, adresse, Toast.LENGTH_LONG).show();
                Address position = positions.get(0);
                latitude.setText("Latitude = "+position.getLatitude());
                longitude.setText("Longitude = "+position.getLongitude());
            }
        }
        catch (IOException ex) { Toast.makeText(this, "Non localisée", Toast.LENGTH_LONG).show(); }
    }
}
```



x SERVICE DE GÉOLOCALISATION

Lorsque vous utilisez les services de géolocalisation, il s'agit, en réalité de services web que nous pouvons atteindre au travers d'un simple navigateur.



maps.googleapis.com/maps/api/geocode/json?address=54 rue Jean Sébastien BACH,Aurillac,France&sensor=false - Chromium

maps.googleapis.com/maps/api/geocode/json?address=54%20rue%20Jean%20Sébastien%20BACH,Aurillac,France&sensor=false


```
      "formatted_address" : "54 Rue Jean Sébastien Bach, 15000 Aurillac, France",
      "geometry" : {
         "bounds" : {
            "northeast" : {
               "lat" : 44.90416390,
               "lng" : 2.43098580
            },
            "southwest" : {
               "lat" : 44.90415450,
               "lng" : 2.4309730
            }
         },
         "location" : {
            "lat" : 44.90416390,
            "lng" : 2.4309730
         },
         "location_type" : "RANGE_INTERPOLATED",
         "viewport" : {
            "northeast" : {
               "lat" : 44.90550818029150,
               "lng" : 2.432328380291502
            },
            "southwest" : {
               "lat" : 44.90281021970850,
               "lng" : 2.429630419708498
            }
         }
      },
      "types" : [ "street_address" ]
   }
],
"status" : "OK"
}
```


maps.googleapis.com/maps/api/geocode/json?latlng=44.90415450,2.4309730&sensor=false - Chromium

maps.googleapis.com/maps/api/geocode/json?latlng=44.90415450,2.4309730&sensor=false


```
   "address_components" : [
      {
         "long_name" : "50-62",
         "short_name" : "50-62",
         "types" : [ "street_number" ]
      },
      {
         "long_name" : "Rue Jean Sébastien Bach",
         "short_name" : "Rue Jean Sébastien Bach",
         "types" : [ "route" ]
      },
      {
         "long_name" : "Aurillac",
         "short_name" : "Aurillac",
         "types" : [ "locality", "political" ]
      },
      {
         "long_name" : "Cantal",
         "short_name" : "15",
         "types" : [ "administrative_area_level_2", "political" ]
      },
      {
         "long_name" : "Auvergne",
         "short_name" : "Auvergne",
         "types" : [ "administrative_area_level_1", "political" ]
      },
      {
         "long_name" : "France",
         "short_name" : "FR",
         "types" : [ "country", "political" ]
      },
      {
         "long_name" : "15000",
         "short_name" : "15000",
         "types" : [ "postal_code" ]
      }
   ],
```