Autoscol'Info

Dossier de Projet

Evan Guélard

Sommaire:

I- Introduction et présentation du projet

Pour la réalisation du projet, nous sommes partis sur l'idée d'une application et d'un site internet qui se relieraient. Nous nous sommes dit que pour cela il fallait un thème … Je pratique la conduite accompagnée, j'ai donc proposé de faire une application qui pourrait calculer le nombre de kilomètres que l'on fait lors des séances de conduite. Ces informations pourraient être envoyées au site internet que Joseph créerait et où elles seraient visibles par les utilisateurs potentiels. Nous avons voulu partir sur un projet concret et pas seulement un travail à faire dans le cadre de nos études.

Lors de nos cours d'ISN en début d'année, nous avons fait plusieurs petits projets afin de découvrir différentes facettes de la matière. Joseph et moi étant amis et dans la même classe, on s'est tout de suite dit que nous pourrions faire un projet commun, et nous avons cherché des idées en fonctions de nos envies, pour enfin arriver à ce projet.

II- Cheminement pour arriver au résultat final

Pour réaliser ce projet, de mon côté j'ai commencé par élaborer un algorithme simplifié (Annexe A), et j'ai commencé par me mettre dans l'esprit que ce serait une application et que l'utilisateur pourrait vouloir lancer l'application sans pour autant commencer la conduite. C'est pourquoi, j'ai choisi de créer un bouton «Départ». A l'enclenchement de ce bouton, l'application récupère les coordonnées GPS puis le bouton «Départ» s'efface et le bouton «Arrivée» apparait. L'utilisateur fait son temps de conduite puis appuie sur le bouton «Arrivée» ce qui reprend ses coordonnées GPS. Après je demande au programme de convertir les données obtenues en radians (elles étaient en degrés) et enfin je lui demande de faire le calcul et de donner la valeur obtenue en kilomètres.

Ensuite j'ai téléchargé le logiciel Eclipse, et j'ai créé la partie graphique de l'application (Annexe B), puis je me suis mis au code, et c'est là que mes problèmes ont commencés.
J'ai cherché les fonctions qui me permettaient de trouver les coordonnés GPS, mais il y en a beaucoup et des différentes. Sachant que je n'ai pas de téléphone sous android, pour tester mon avancée je devais utiliser l'émulateur fournis par le logiciel qui fonctionne très bien mais le problème c'est que mon ordinateur n'as pas de GPS intégré, et donc il m'est impossible de tester véritablement mon application. J'ai donc testé plusieurs fonctions pour mon code, certaines m'enlevaient du travail, d'autres m'en rajoutaient mais aucune ne me donnait véritablement de résultats convaincants.

Pendant de nombreuses séances j'ai cherché une solution, puis au bout d'un moment je me suis dit, ça ne sert à rien d'essayer par différentes manières de faire marcher la partie GPS de mon application si je ne peux pas tester. Je me suis donc dit, j'abandonne la partie GPS et j'essaie de faire une application qui calcule la distance parcourue à partir du kilométrage de la voiture, l'utilisateur devrait rentrer les chiffres correspondant aux kilomètres de la voiture au départ et à l'arrivée et donner la distance parcourue (Annexe C).

Mais les vacances sont arrivées et j'ai demandé un petit coup de main à mon beau père, qui m'a aidé à améliorer la première application, j'ai donc laissé tomber l'application avec le kilométrage pour l'application avec GPS qui est beaucoup plus intéressante.

III- Résultat final

Au final mon application fonctionne (Annexe D). Je l'ai testée lors d'un trajet en voiture, cependant le kilométrage qui était affiché n'était pas celui parcouru. Mon application calcule la distance entre 2 points toutes les 10 secondes, et s'il y a eu un virage, l'application prend en compte la ligne droite et non le virage car elle calcule une distance entre deux points à vol d'oiseau, donc s'il y a des virages, il y a une incertitude, or le résultat obtenue lors du test était une valeur plus grande que celle attendue (celle marquée sur le compteur) alors que selon la logique précédente la valeur aurait dû être en dessous. Le GPS doit manquer un peu de précision, ce qui sur une trentaine de kilomètres en rajoute un peu. Cette incertitude de GPS et cette imprécision de « code » faite « exprès » pour ne pas décharger la batterie de l'appareil trop vite donne une différence entre les kilomètres effectués et ceux affichés.

Il faut aussi savoir qu'en montagne mon application fonctionne moins bien car sachant que les coordonnés GPS sont pris toutes les 10 secondes, la formule est faite comme si on était sur un plan plat et ne prend pas en compte l'altitude, par conséquent en montagne, il y aurait une plus grande imprécision. Mais le choix de la formule a été fait sur la facilité car les formules qui seraient les plus précises demandent plus de conversions et de calculs.

Par la suite il est possible d'améliorer le design de l'application, la précision peut surement être améliorée.
Mais surtout de permettre à l'application de stocker les informations sur le site internet de Joseph quand l'utilisateur appuiera sur « Arrivée », pour le moment le bouton est sur l'application pour le design et pour me rappeler qu'il n'attend plus qu'à être utile. Je pense que je ferai la liaison finale et les finitions pendant les vacances d'été car ce détail me stresse un peu dans l'état actuel, je voudrai finir mon projet et ne pas le laisser inachevé, mais pour cela il faudra encore des heures de travail.

IV- Sources

http://www.siteduzero.com/informatique/tutoriels/votre-premiere-application-android Ce site m'a aidé à débuter dans la programmation de ce langage, comme le lien ci-dessous

http://android-pour-les-nuls.fr/tutoriaux/developpement/tuto-developpement-application-android-debutant et celui ci-dessous de même

http://android.developpez.com/cours/

https://maps.google.fr/ Pour faire des essais de formules

http://www.siteduzero.com/forum/sujet/calcul-d-une-distance-95555 Pour les distances

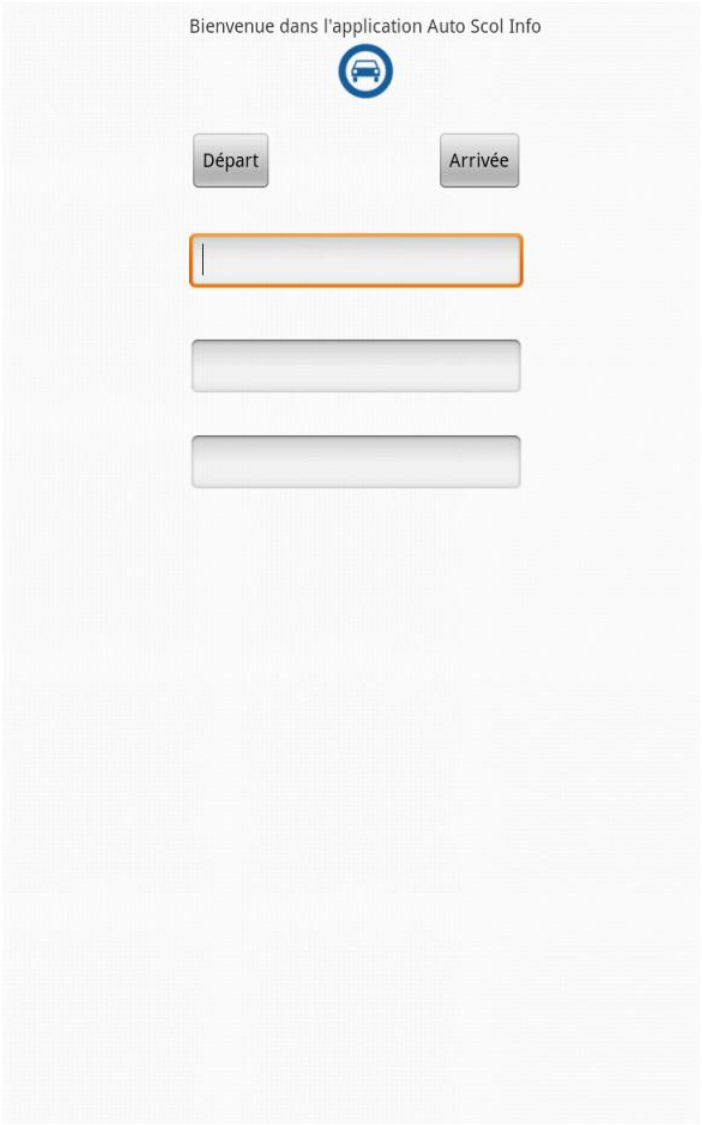
http://japan-party.net/infos-pratiques/ Logo

Annexe A : Algorithme de départ

Algorithme de base – Projet ISN

```
Début
    Afficher le bouton Départ
    Si le bouton Départ est enclenché,
        // Prendre les données GPS au point de départ 1
        Données LatDep du GPS : Latitude
        Données LonDep du GPS : Longitude
        Effacer le bouton Départ et Afficher de bouton Arrivée
    FinSi

    Si le bouton Arrivée est enclenché,
        // Prendre les données GPS au point d'arrivée 2
        Données LatArr du GPS : Latitude
        Données LongArr du GPS : Longitude
        Effacer le bouton Arrivée
    FinSi

    // Convertir les données GPS en Radians
    Calculer LatDep = LatDep*PI/180
    Calculer LonDep = LonDep*PI/180
    Calculer LatArr = LatArr*PI/180
    Calculer LonArr = LonArr*PI/180

    // Calculer la distance d entre les 2 points
    Calculer d = 6371*cos-1[sin(LatDep)*sin(LatArr)+cos(LatDep)*cos(LatArr)*cos(LonDep-LonArr)]

    Dire d // d est donné en km
Fin
```

Annexe B : Partie graphique de l'application



Annexe C : Code de l'application sans GPS

```
package com.example.auto_scol_info;

import android.os.Bundle;

public class MainActivity extends Activity {

        Button btCalcul;
        EditText kmDepart;
        EditText kmArrive;
        EditText kmDistance;

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


        kmDepart = (EditText) findViewById(R.id.kmDepart);
        kmArrive = (EditText) findViewById(R.id.kmArrive);
        kmDistance = (EditText) findViewById(R.id.kmDistance);
        btCalcul = (Button) findViewById(R.id.btCalcul);

        //on applique un écouteur d'évenement au clique sur le bouton
        btCalcul.setOnClickListener(
            new OnClickListener() {
                @Override
                public void onClick(View v) {
                    //on réupère le texte écrit dans l'EditText
                    Integer dep = Integer.parseInt(kmDepart.getText().toString());
                    Integer arr = Integer.parseInt(kmArrive.getText().toString());
                    Integer result = arr-dep;

                    ((TextView)findViewById(R.id.kmDistance)).setText("Distance parcourue: " + result.toString() + " km.");
                }
            }
        );
     }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}
```

Annexe D : Code de l'application avec GPS

```
package com.example.autoscolinfo;


import com.example.autoscolinfo.Constantes;

public class MainActivity extends Activity {
    //<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Déclaration des boutons
         Button btnDepart = (Button)findViewById(R.id.btnDepart);
         Button btnArrive = (Button)findViewById(R.id.btnArrive);

        // Déclaration des textes
        EditText distance = (EditText) findViewById(R.id.distance);
        EditText fournisseur = (EditText) findViewById(R.id.fournisseur);
        EditText coordonnees = (EditText) findViewById(R.id.coordonnees);

        // sur clique du bouton départ
         btnDepart.setOnClickListener(new View.OnClickListener()
            {
                public void onClick(View v) {

                    float[] results = null;
                    LocationListener listener;
                    double distance = 0;
                    double distancetotale = 0;

                    // Déclaration du service système pour le GPS
                    LocationManager locationManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);

                    // Déclaration des critères pour sélection le meilleur fournisseurde servivces de localisation
                    Criteria critere = new Criteria();

                    // Pour indiquer la précision voulue
                    // On peut mettre ACCURACY_FINE pour une haute précision ou ACCURACY_COARSE pour une moins bonne précision
                    critere.setAccuracy(Criteria.ACCURACY_FINE);

                    // Est-ce que le fournisseur doit être capable de donner une altitude ?
                    critere.setAltitudeRequired(true);

                    // Est-ce que le fournisseur doit être capable de donner une direction ?
                    critere.setBearingRequired(false);

                    // Est-ce que le fournisseur peut être payant ?
                    critere.setCostAllowed(false);

                    // Pour indiquer la consommation d'énergie demandée
                    // Criteria.POWER_HIGH pour une haute consommation, Criteria.POWER_MEDIUM pour une consommation moyenne et Criteria.POWER_LOW pour une basse consommation
                    critere.setPowerRequirement(Criteria.POWER_LOW);

                    // Est-ce que le fournisseur doit être capable de donner une vitesse ?
                    critere.setSpeedRequired(true);

                    // Récupération du meilleur fournisseur suivant les critères ci-dessus
                    String fournisseur = locationManager.getBestProvider(critere, true);
```

```
// Si le fournisseur n'est pas trouvé, alors on prend le fournisseur GPS par défaut
if (fournisseur == null) {
    fournisseur=LocationManager.GPS_PROVIDER;
    ((TextView)findViewById(R.id.fournisseur)).setText("L'application AutoScolInfo est connectée sur le fournisseur: " + locationManager.GPS_PROVIDER + ".");
}

// Demande de localisation toutes les 60000 ms ou tous les 150 mètres par rapport au fournisseur
locationManager.requestLocationUpdates(fournisseur, 10000, 0, new LocationListener() {

      // Déclaration des variable de coordonées
      double startLatitude=0;
      double startLongitude=0;
      double endLatitude=0;
      double endLongitude=0;
      double distance=0;

      @Override
      public void onStatusChanged(String provider, int status, Bundle extras) {
            String newStatus = "";
            switch (status) {
                case LocationProvider.OUT_OF_SERVICE:
                    newStatus = "hors service";
                    break;
                case LocationProvider.TEMPORARILY_UNAVAILABLE:
                    newStatus = "temporairement indisponible";
                    break;
                case LocationProvider.AVAILABLE:
                    newStatus = "disponible";
                    break;
            }
              ((TextView)findViewById(R.id.fournisseur)).setText("Le fournisseur " + provider + " est "+ newStatus + ".");
      }

      @Override
      public void onProviderEnabled(String provider) {
          ((TextView)findViewById(R.id.fournisseur)).setText("L'application AutoScolInfo est connectée sur le fournisseur: " + provider + ".");
      }

      @Override
      public void onProviderDisabled(String provider) {
          ((TextView)findViewById(R.id.fournisseur)).setText("L'application AutoScolInfo est déconnectée du fournisseur: " + provider + ".");
          //MainActivity.distanceBetween (startLatitude, startLongitude, endLatitude, endLongitude, results);
      }

      @Override
      public void onLocationChanged(Location location) {
          ((TextView)findViewById(R.id.coordonnees)).setText("Latitude " + location.getLatitude() + " et longitude " + location.getLongitude());
          if (startLatitude==0 && startLongitude==0) {
              // Initialisation des premières coordonnées
              startLatitude=location.getLatitude();
              startLongitude=location.getLongitude();
              ((TextView)findViewById(R.id.distance)).setText("Début de l'appli");
          } else {
              // Récupération des dernières coordonnées et mise à jour de la distance parcourue
              endLatitude=location.getLatitude();
              endLongitude=location.getLongitude();

              distance = distance + calculDistance(startLatitude,startLongitude,endLatitude,endLongitude);
              startLatitude=location.getLatitude();
```

Petite erreur ci-dessus, c'est toutes les 10 secondes (2e commentaire)

```
                  startLongitude=location.getLongitude();
                  ((TextView)findViewById(R.id.distance)).setText("Distance parcourue: "+distance+" mètres.");
              }

          }
      });
  }

  protected static double calculDistance(double startLatitude,
          double startLongitude, double endLatitude, double endLongitude) {
      // TODO Auto-generated method stub
      double distance=0;
      double x1,x2;
      double y1,y2;

      // Conversion et simplification des coordonnées en un plan 2D
      // La faible différence entre les deux poins justifie de ne pas tenir compte de la rondité de la terre
      x1=startLatitude*Constantes.perimetreTerresteMoyen/360;
      x2=endLatitude*Constantes.perimetreTerresteMoyen/360;
      y1=startLongitude*Constantes.perimetreTerresteMoyen/360;
      y2=endLongitude*Constantes.perimetreTerresteMoyen/360;
      distance=Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));

      return distance;
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
      // Inflate the menu; this adds items to the action bar if it is present.
      getMenuInflater().inflate(R.menu.main, menu);
      return true;
  }
}
```