POLYTECHNIQUE MONTRÉAL LE GÉNIE EN PREMIÈRE CLASSE	**École Polytechnique de Montréal** **Département Génie Informatique et Génie Logiciel**

Travail Pratique N°2 : Programmation des capteurs sous Android

1 Informations générales

Session	Hiver 2015
Date de début	Vendredi 20 février 2015
Date de remise	Jeudi 12 mars 2015 23 :55
Laboratoire	L4708
Pondération	15%
Nombre d'étudiants	3 (les mêmes groupes de laboratoire précédent)
Directives particulières	1. Tout rapport sera pénalisé de **3** points s'il est soumis par une équipe dont la taille est différente de celle définie sans l'approbation préalable du chargé de laboratoire 2. Rapport à rendre en format PDF ou Word. 3. **Soumission** du rapport et des travaux réalisés par **moodle uniquement** (http://moodle.polymtl.ca). 4. Tout retard de soumission du rapport et des travaux réalisés sera pénalisé de **3** points par jour de retard.
Chargé de laboratoire	Mauricio Mendoza (Mauricio.Mendoza-Medellin@polymtl.ca)

2 Connaissances préalables.

- Programmation Android (activities, intents, broadcast receivers, services, bases de données etc.).
- Configuration matérielle des capteurs sous Android.
- Langage XML.
- Notions de bases des réseaux sans fil Wifi, GPS, réseaux cellulaires.

3 Environnement et outils nécessaires.

- Environnement de développement Intégré (IDE) Android Studio ou Éclipse + SDK Android +Plug-in
- Éclipse d'Android (ADT).
- Tablette Android ou Terminaux mobiles Android.
- Google Maps API

4 Éléments de contexte

Les téléphones intelligents possèdent comme caractéristique les capteurs, lesquels leurs permettent dans un écosystème d'être polyvalents. Les capteurs font référence à la capacité de relever selon un paramètre, des mesures sur l'appareil et son environnement

externe. Sans ces capteurs, un périphérique Android se réduirait probablement à un simple équipement mobile, doté d'un écran d'une taille relativement petite et il nécessiterait très peu d'interactions avec l'environnement externe.

Mais cette caractéristique a un prix dans la consommation d'énergie où beaucoup d'études ont été réalisées afin d'optimiser sa consommation et les sources disponibles comme ce sont les batteries.

Les capteurs continueront d'être une partie importante de la plate-forme Android. Mieux, leur nombre se verra accroitre et ils seront de meilleure qualité, conformément aux améliorations des spécifications matérielles desdits dispositifs.

Le nombre de solutions que des capteurs utilisent croît d'une manière exponentielle également, que la quantité de capteurs disponibles dans les dispositifs Android et les attentes des utilisateurs quant à leur utilisation pour une meilleure expérience se verront aussi accroitre. Par conséquent, la conception de nouvelles applications faisant utilisation des capteurs est une compétence essentielle que tout programmeur Android devrait maîtriser. Cette conception implique une compréhension des capacités de détection d'un appareil Android, la sélection des capteurs adéquats à utiliser dans une application, la mise en œuvre convenable de ceux-ci en vue de procéder à l'acquisition, la gestion énergétique pertinente et l'interprétation correcte des données.

5 Objectifs du laboratoire

Sensibiliser l'étudiant aux limitations existantes dans les terminaux mobiles à l'égard de la consommation d'énergie, de la capacité de traitement et de l'utilisation de la mémoire. Aussi, familiariser l'étudiant au développement de services de géolocalisation pour des terminaux mobiles Android. De manière spécifique, au terme de ce laboratoire, il s'agira pour l'étudiant de :

- Déterminer la position d'un périphérique Android en utilisant une variété de capteurs incluant le GPS et le réseau cellulaire (Capteurs de localisation).
- Acquérir des informations propres au périphérique mobile et à son environnement : l'orientation, l'accélération, la rotation, la collecte d'images visuelles (Capteurs physiques).
- Développer une application qui utilise les services réseaux notamment le GPS, le réseau cellulaire, le Wifi à des fins de géolocalisation.
- Identifier, mesurer et contrôler les différents composants qui peuvent épuiser la batterie dans l'usage de l'application développée.

6 Présentation de l'application

Au cours de ce travail pratique, vous aurez à concevoir et réaliser une application permettant de retracer sur une carte Google le trajet effectué par un utilisateur muni d'un périphérique Android.

Considérant un parcours prédéfini (point de départ et point d'arrivée connus), plusieurs points de marquage seront identifiés sur la carte afin de matérialiser le trajet parcouru. Deux modes de localisation seront mis en œuvre : la localisation par GPS, ainsi que la localisation par le réseau cellulaire. À chaque point de marquage, des données contextuelles propres à l'environnement du périphérique seront relevées.

À la fin du parcours, la visualisation de la carte permettra de consulter tous les

points de marquage ainsi que les différentes informations relevées

7 Requis

Au lancement de l'application, les paramètres d'exécution de l'application doivent être définis. L'utilisateur devra spécifier le mode de localisation ainsi que la fréquence de mise à jour de la position courante du périphérique Android. Cette fréquence de mise à jour permettra de relever différents points de marquage afin de retracer l'itinéraire parcouru. De manière plus détaillée, les requis fonctionnels de l'application se décrivent comme suit :

7.1 Requis fonctionnels

La sélection d'options au démarrage de l'application. L'utilisateur peut à cette phase :

- Sélectionner le mode de localisation ainsi que la fréquence de mise à jour de la position du périphérique mobile.
- Définir le zoom initial minimal de la carte Google.
- Consulter les 3 derniers trajets déjà parcourus ainsi que les informations qui y sont rattachées.
- Quitter l'application (prévoir une boite de dialogue de confirmation afin de prévenir les erreurs de manipulation).

Pour tous les deux modes de localisation (GPS ou réseau cellulaire) :

- Définir le point de départ et d'arrivée du trajet (Numéro et Rue, exemple : 2500 chemin de polytechnique), et les marquer ensuite sur une carte Google.
- Relever le niveau initial de la batterie (*Niv_init_batt*) avant de lancer la procédure de localisation
- Activer la procédure de localisation du périphérique mobile (GPS ou réseau cellulaire selon le mode de localisation choisi.).
- Pour chaque point de marquage vous devez identifier :
    - Ses *coordonnées* (**Coord**) indiquant la longitude, la latitude, ainsi que l'altitude et *l'instant de marquage* (**Im**).
    - La *direction de déplacement* (**Dir_dep**) par rapport au point de marquage précédent. Il s'agit de préciser l'orientation : Nord, Sud, Est, Ouest ou une combinaison (sud-ouest, nord-est.. etc.)
    - La *distance relative parcourue* (**Drp**) depuis le dernier point de marquage ainsi que la *vitesse moyenne* (**Vm**) de déplacement.
    - La *distance totale* (**Dt**) parcourue depuis le début du trajet.
    - Le *mode de localisation* (**Mod_loc**) : vous devez préciser si la localisation a été faite par le GPS ou le Réseau cellulaire.
    - Émettre une alerte et capturer une image d'identification de l'environnement immédiat du point de marquage.
    - Relever le *niveau courant de la batterie* (**Niv_batt**) lors du marquage.
- Placer sur la carte Google le point de marquage identifié. Les informations relevées sur le point de marquage sont affichées dans une fenêtre contextuelle en cliquant sur celui-ci. (Vous pouvez marquer au fur et à mesure les points relevés sur la carte Google ou marquer l'ensemble des points relevés au moment de visualiser la carte)
- Arrêter la procédure de localisation. (**NB** : L'utilisateur doit pouvoir arrêter la procédure de localisation à tout moment pendant l'exécution de l'application).

- Relever le *niveau final de la batterie* (**Niv_fin_batt**) à l'arrêt de la procédure de localisation.
- Afficher sur la carte Google l'itinéraire suivi par le périphérique mobile entre le point départ et le point d'arrivée en incluant les points de marquage identifiés.
- Sauvegarder toutes les informations relevées dans une base de données SQLite.
- Quitter l'application à tout moment (prévoir une boite de dialogue de confirmation afin de prévenir les erreurs de manipulation).

Pour le mode localisation par GPS, une fois que votre point de marquage a été relevé sur l'itinéraire de parcours :

- Émettre une alerte de proximité uniquement si au moins un point d'accès Wifi (PA) se trouve à proximité du point de marquage.
- Pour le **PA ayant le niveau de signal le plus élevé**, vous devez relever le BSSID, le SSID ainsi que la qualité du signal RSS.
- Associer les informations relevées sur le PA aux informations contextuelles du point de marquage (les afficher dans la fenêtre contextuelle, *PA_Wifi (SSID, RSS, BSSID)*).

NB : On ne vous demande pas de marquer le point d'accès Wifi sur la carte Google.

Pour la localisation par le réseau cellulaire, une fois que votre point de marquage a été relevé sur l'itinéraire de parcours, vous devez identifier pour la station de base courante les paramètres suivants :

- Le type de réseau cellulaire (**Type_R**) : GSM, CDMA ou UMTS, etc.
- Le **MCC** (Mobile Country Code).
- Le **MNC** (Mobile Network Code) de l'opérateur du réseau cellulaire, ainsi que son nom.
- L'identifiant de la cellule (**Cell_ID**) du point d'attache (station de base) courant du périphérique mobile
- Le **LAC** (Location Area Code) qui identifie le code de la zone de localisation.
- Le niveau du signal reçu de la station de base (**Niv_sig_sb**).
- La longitude et la latitude de la station de base (**long_sb, lat_sb**).
- Marquer la station de base sur la même carte Google (différencier ce marquage des points de marquage sur l'itinéraire de déplacement).
- A l'arrêt de la procédure de localisation, l'application doit afficher en plus de la carte Google et les informations relevées le nombre de stations de bases (**Nbr_sb**) identifiées entre le point de départ et le point d'arrivée.

7.2 Requis non fonctionnels

Les exigences non fonctionnelles attendues sont les suivantes :

- L'application sera développée pour une plate-forme Android. À des fins de compatibilité, la version de la plate-forme à considérer est Android 4.1 (Jelly Bean). Toutes les versions récentes d'Android seront acceptées.
- L'application doit être utilisable sur des terminaux mobiles offrant des services de GPS, ayant accès à Internet et répondant aux requis indiqués ci-dessus sur la plate-forme. Les tablettes et téléphones mobiles intelligents (smartphones) sont les terminaux ciblés.
- L'application doit être facile d'utilisation. Un soin doit être accordé à l'interface en général.
- La production d'un code lisible et suffisamment documenté (commentaires du code) est souhaitée.
- Le développement doit se faire avec l'IDE Android Studio ou avec l'IDE Éclipse.

8 Livrables

Outre la présentation de l'application (démonstration), les livrables à fournir à la fin du TP seront regroupés dans une archive (ZIP ou RAR) dont le nom est formé des matricules des membres de l'équipe, séparés par le caractère « trait de soulignement » c'est-à-dire ('_'). L'archive contiendra un rapport de TP au format PDF et le code source de l'application. Le non-respect des consignes est sujet à pénalité.

8.1 Rapport de TP

Votre rapport doit contenir :

- **Une page de présentation** (indispensable) faisant mention des éléments suivants : Nom et logo de l'école ; le sigle et le titre du cours ; la session ; le libellé du TP ; les noms, prénoms et matricules des membres de l'équipe; la mention « Soumis à : *nom et prénoms du chargé de laboratoire* » ; la date de soumission.
- **Une introduction** qui mentionne le contexte dans lequel le travail a été réalisé ainsi qu'une brève description de la méthodologie (aspects techniques).
- Une section dédiée à la **méthodologie** (aspects techniques) de réalisation du TP. Cette section doit mentionner :
    - Une indication de la répartition des travaux au sein de l'équipe.
    - Une indication des différentes phases de réalisation du TP.
    - La structure de l'application (classes, packages, méthodes/variables publiques privées, composantes utilisées). Expliquer autant que faire se peut, les choix effectués. Un diagramme de classes.
    - La séquence d'exécution. (Il est suggéré d'inclure un diagramme à cet effet).
    - Les difficultés rencontrées ainsi que les décisions qui ont été prises pour les résoudre.
    - **Répondre aux questions suivantes :**
        - Comparez la précision des deux modes de localisation quant à la réalisation de l'itinéraire
        - Comparez la consommation énergétique des deux modes de localisation
- Une section dédiée aux **tests d'exécution** (inclure des captures d'écran) pour valider les fonctions réalisées.
- Une conclusion et **les travaux futurs.** Cette section doit mentionner :
    - Un résumé de l'expérience acquise après la réalisation du projet.
    - Des améliorations susceptibles d'être apportées à votre projet si un délai supplémentaire devait être accordé.
    - Vos critiques et suggestions pour l'amélioration du contenu.

8.2 Code source

Afin de faciliter la correction, vous devrez remettre un fichier .zip
Pour Éclipse :

Les étapes sont les suivantes : Clic droit sur le projet > Export... > General > Archive File > Next > S'assurer que le projet est bien coché ainsi que tous les dossiers et fichiers> Donner un nom au fichier archive avec précision du chemin complet d'accès en remplissant le champ « To archive file » > Cocher l'option « Save in Zip format) > Finish.

Pour AndroidStudio:

Fermer votre Project : File > Close Project, après, allez au dossier /AndroidStudioProjects comprimer en format zip et copier le ficher.

9 Grille d'évaluation

Rubriques	Points
Présentation (Démonstration + réponses aux questions)	4
Évaluation complémentaire de l'exécutable : (fonctionnalité, ergonomie, portabilité, performance).	4
Évaluation du code source (implémentation de l'architecture, logique de programmation, documentation du code).	6
Rapport de TP (fond, forme).	6
Total	20

10 Liens utiles

- https://developers.google.com/maps/documentation/android/
- http://developer.android.com/guide/topics/sensors/index.html
- http://www.tutos-android.com/integration-google-map-android
- http://www.tutos-android.com/geolocalisation-android
- http://developer.android.com/guide/topics/location/strategies.html
- http://developer.android.com/reference/android/location/LocationManager.html#GPS_PROVIDER
- http://developer.android.com/reference/android/location/LocationManager.html#NETWORK_PROVIDER
- http://www.andygup.net/how-accurate-is-android-gps-part-1-understanding-location-data/
- http://developer.android.com/reference/android/telephony/gsm/GsmCellLocation.html
- http://opencellid.org/
- http://developer.android.com/reference/android/net/wifi/WifiManager.html
- http://developer.android.com/reference/android/net/wifi/ScanResult.html
- http://android-er.blogspot.ca/2010/09/detect-battery-info.html