INF8405 – INFORMATIQUE MOBILE

TRAVAIL PRATIQUE 1

APPLICATION DE RECHERCHE DE CONNEXION WI-FI

SOUMIS À :

MIKAELA **NGAMBOÉ**

PAR :

1773922, ÉTIENNE **ASSELIN**

1744784, VINCENT **RODIER**

DATE : 1 MARS 2018

POLYTECHNIQUE MONTRÉAL

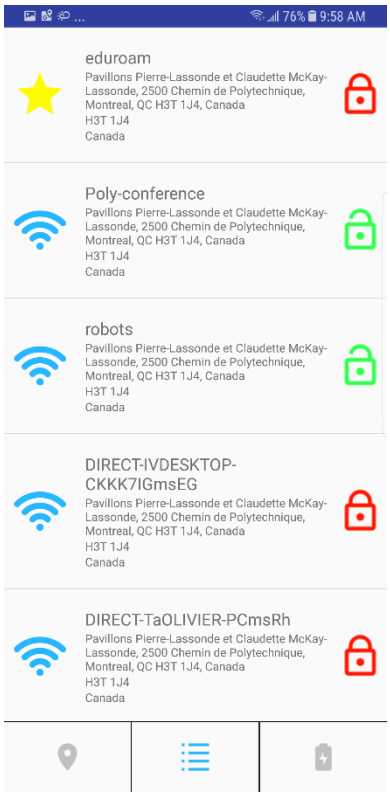
Table des matières

Introduction

Dans le cadre de ce travail pratique nous avons conçu une application pour les appareils roulant sur le system d'exploitation Android 4.3.1 ou supérieur (API 18). L'application développée permet de rechercher les connexions Wi-Fi et de les situer sur une carte fournie par Google. En plus d'identifier l'emplacement des bornes d'accès, nous avons fournis une option pour partager et naviguer vers cette position. Finalement, nous avons développé une fonctionnalité qui permet de calculer la consommation de la pile lors de l'utilisation de l'application. Le développement de cette application nous a permis de se familiariser avec les différents capteurs associés aux appareils mobiles.

Présentation Technique

Navigation dans l'application



Il y a plusieurs manières pour naviguer dans l'application. Premièrement, il y a le ruban de navigation qui se trouve dans le bas et qui reste statique lors des changements de vue. Chacun des choix est représenté par une icône représentative de la section respective. Une seconde façon de naviguer dans l'application est avec des glissements de doigts sur le bord de l'écran. Cette façon de naviguer dans l'application est très fluide et naturelle pour l'utilisateur.

La navigation des vues dans notre application est faite à l'aide d'un *ViewPager* d'Android. La vue du *ViewPager* ainsi que la vue de la barre de navigation se trouve chacune dans un fragment différent. L'activité principale possède deux *FrameLayout*, le premier est d'une hauteur de 75dp et se trouve toujours aligner avec le bas de la vue parente, tandis que le deuxième *FrameLayout* se trouve au-dessus du premier et occupe l'espace restant. Les deux *FrameLayout* servent donc de conteneur pour les fragments du *ViewPager* et du fragment du ruban de navigation.

La cohérence entre le *ViewPager* et le ruban de navigation se fait essentiellement avec des évènements du *ViewPager*. Lorsque le focus est fait sur une nouvelle vue à l'intérieur du *ViewPager*, celui-ci demande à l'activité d'avoir l'instance du ruban de navigation et appel une fonction pour changer la couleur des icônes.

Lorsqu'un usager clique sur un réseau Wifi afin d'avoir une vue plus approfondie des informations concernant ce réseau, une nouvelle activité est ouverte et une information unique du réseau à observer sont envoyées à la nouvelle activité. La nouvelle activité fait ensuite une recherche dans la base de données locale afin d'identifier d'avoir accès à toutes les informations nécessaires à l'affichage.

Géolocalisation

La géolocalisation est permise grâce à la puce GPS intégrée à même les appareils mobiles. Cependant, avant de pouvoir utiliser les fonctionnalités associées à la géolocalisation il faut demander les permissions nécessaires à l'administrateur de l'appareil. Une fois ces permissions accordées, nous pouvons utiliser les fonctionnalités l'API de Google Map afin de situer l'appareil sur la carte. Pour ménager la pile, nous calculons la position de l'appareil dans un intervalle de temps précis. Dans le cas où l'utilisateur désactive la géolocalisation de son appareil, nous gérons l'exception et réévaluons la position dans le prochain appel à la méthode lors du prochain intervalle. En plus d'avoir la longitude et latitude grâce à la puce GPS, l'API de Google nous permet d'avoir l'adresse associée à cette localisation. Pour se faire, il faut avoir des données internet.

Balises Wi-Fi

Les balises Wifi sont découvertes grâce à une fonction qui est appelée toutes les 10 secondes et qui fait un scan Wifi. Le temps entre les appels à cette fonction est arbitraire, mais pas totalement puisqu'elle minimise la possibilité de ne pas voir un réseau tout en n'étant pas exécuter continuellement.

Les résultats des balises Wifi reçu lors du scan sont comparés avec les valeurs déjà connues dans la base de données locale de l'appareil. Si le réseau est nouveau, une nouvelle entrée est ajoutée à la base de données sinon, les informations sont tout simplement rafraichies.

Enregistrement des données

Nous avons choisi d'utiliser la plateforme *Realm.io* comme solution afin d'ajouter de la persistance locale à notre application. L'utilisation de la plateforme *Realm* est très similaire à l'utilisation d'une base de données. La base de données *Realm* permet de stocker des objets qui héritent de *RealmObject*, donc nous avons tout simplement enregistré chaque réseau différent comme un objet qui hérite de cette classe et ainsi nous avons toute l'information nécessaire concernant les réseaux Wifi.

Quelques informations sont aussi enregistrées localement sur le téléphone concernant la consommation d'énergie. Comme peu de valeurs doivent conserver, nous avons décidé de les enregistrer à l'aide des *SharedPreference* d'Android. Ainsi une classe nommée *SharedPreferenceManager* implémente quelques méthodes statiques qui permettent d'enregistrer et de récupérer les différentes valeurs enregistrées dans un fichier local.

Consommation de la pile

La consommation de la batterie est assez simple dans notre application. Nous calculons le temps total durant lequel l'application fonctionne en notant l'heure exacte de l'ouverture de l'application et en y soustrayant la valeur de l'heure actuelle. L'application surveille le pourcentage de batterie perdu pendant cette période de temps. Lorsque l'utilisateur décide de quitter l'application, ces données sont enregistrées tel que décrit dans la section ci-dessus à l'aide des méthodes fournies par la classe *SharedPreferenceManager*.

Tous se qui attrait aux calculs de temps et de pourcentage de batterie se font dans une classe nommée *EnergyManager* qui implémente le patron singleton. Les différentes activités de l'application communiquent donc à l'instance *EnergyManager* le moment opportun pour partir et arrêter le calcul du temps et les vues n'ont qu'à questionner cette instance afin d'avoir les informations pertinentes concernant la consommation d'énergie de l'application.

Difficultés Rencontrées

Notre équipe est constituée de deux étudiants dont un avait un peu d'expérience dans la création d'application Android. Le fait qu'un des membres de l'équipe avait de l'expérience a fait en sorte que la plus grosse difficulté dans la création de cette application a été l'interaction avec les instruments du téléphone. Nous avons eu plus particulièrement de la difficulté avec le capteur qui nous permettait de faire la lecture des différents beacon Wifi. Il a fallu faire plusieurs recherches et essayer plusieurs façons de faire différentes avant d'arriver à un résultat satisfaisant.

Une autre difficulté rencontrée lors de ce TP a été de tester tous les différents points de défaillance. En effet, notre application fonctionnait très bien dans les conditions dans des conditions optimales, mais il nous a été difficile de vérifier le bon fonctionnement de l'application considérant le fait que l'usager peut activer ou désactivé le Wifi ainsi que la position à n'importe quel moment.

Critiques et Suggestions

Ce travail pratique peut définitivement être intimidant pour une équipe ne possédant aucune expérience dans le développement d'une application Android. Malgré cela, nous pensons que le temps donné afin d'accomplir cette tâche est raisonnable, peu importe le niveau d'expérience. De plus, le fait de développer avec Android plutôt qu'IOS est très apprécié puisqu'il est beaucoup plus accessible de développer avec Android (pas besoin d'un MAC).

Pour les suggestions, nous pensons qu'il serait bien d'interagir avec plus que juste les capteurs de l'appareil. Intégrer seulement un capteur de l'appareil est suffisant à notre avis si l'on ajoute aussi une partie web au travail ou même avoir une portion plus interactive avec l'application. Par exemple, une application qui utilise le gyroscope de l'appareil afin de déplacer un marqueur dans la carte Google.

Pour conclure, le travail à faire dans ce TP est très intéressant et formateur. Il intègre plusieurs notions utiles dans le développement d'une application Android et permet de rapidement comprendre la base de la création d'application.