Application Android: Arboretum
Rapport

Barbier Jérôme, Benyounes Radhoane, Bobo William
Fréby Rodolphe, Husson Augustin, Labat Paul

13 janvier 2014

Table des matières

1 Introduction

1.1 But de ce Rapport

Ce rapport a été rédigé pour nous permettre d'expliquer en détail nos choix de conception. Il permettra également en cas de reprise de ce projet d'avoir une connaissance approfondie de l'ensemble des fonctionnalités qui a été implémenté.

1.2 Présentation du Projet

Le projet s'intitule *Arboretum*. Il s'agit de faire une application qui permettra à l'utilisateur de faire une visite guidée du parc. Ce parc se situe à l'extrémité Nord Est du campus de Grenoble en France. L'application doit permettre à l'utilisateur de se rendre dans le parc et de faire la visite guidée. Elle doit également lui permettre de faire la visite sans y être.

La particularité de ce parc est qu'il y a une représentation à échelle réduite de notre Galaxie. On y trouve également des espèces d'arbres et de plantes peu communes. L'application doit permettre de naviguer entre les arbres et les planètes et de fournir une description détaillée pour chacun d'entre eux.

1.3 Spécification technique

- Configuration nécessaire : Android 4.1
- Un capteur NFC est nécessaire pour profiter de toutes les fonctionnalités de l'application
- Place prise sur le stockage interne : 14,42Mo pour l'application, et 3,5Mo de données téléchargées.
- L'application requiert une autorisation d'utiliser :
  - le stockage
  - votre position (via le GPS)
  - la communication réseau (c'est à dire avoir un accès Internet complet, ainsi qu'avoir le contrôle NFC)
  - les commandes matérielles (c'est-à-dire prendre des photos et des vidéos)

2 Architecture

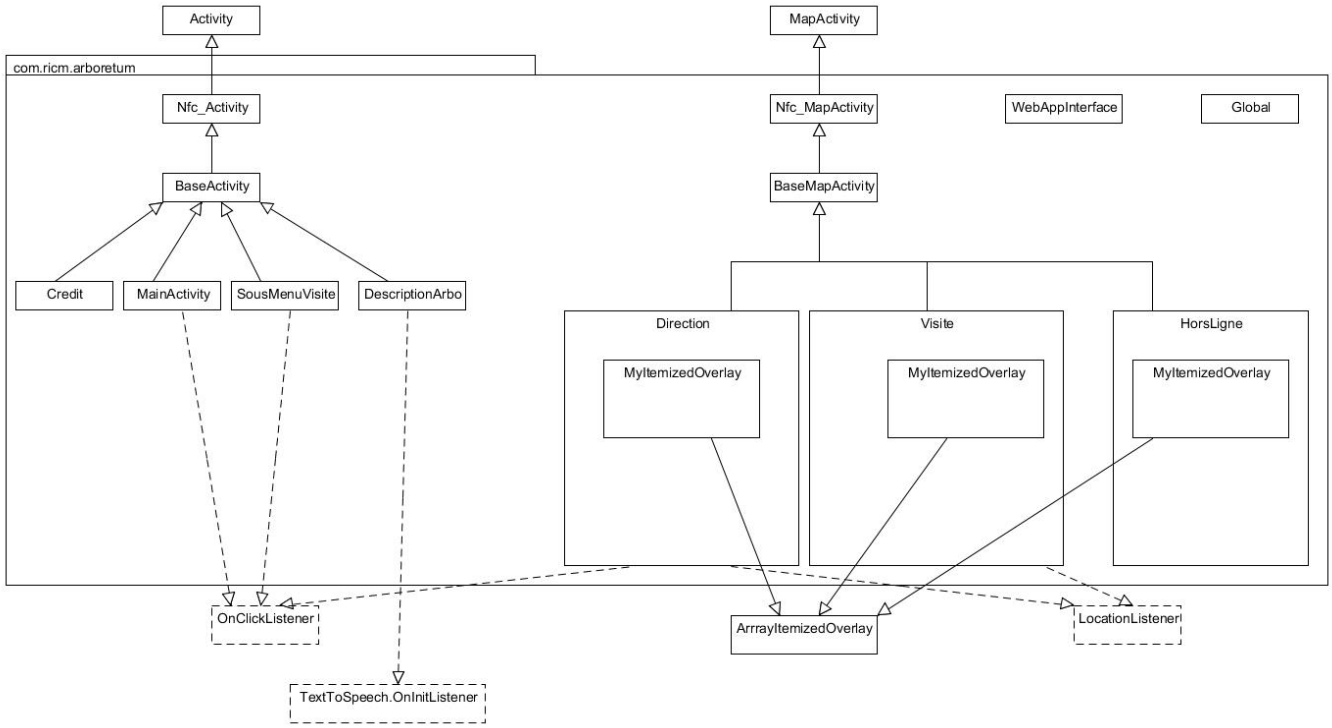

FIGURE 1 – Architecture globale

Comme on peut le voir, nous avons un package principal com.ricm.arboretum. Dans ce package on trouve deux arbres d'héritage principaux. Le premier ayant pour classe mère Activity, et l'autre ayant pour classe mère MapActivity.

Le premier arbre est l'implémentation des activités générales qui composent l'application et le second est l'implémentation de toutes les activités qui nécessitent la présence d'une map.

On remarque que certaines classes sont en dehors du package. La différence est faite si la classe a été conçue par notre équipe ou fournit par des packages externes.

3 Gestion des menus

L'application met à disposition un système de différentes *activités* pour pouvoir accéder aux fonctionnalités désirées. L'utilisateur arrive sur un menu principal lui demandant s'il souhaite se rendre à l'arboretum ou visiter celui-ci.

FIGURE 2 – Menu Principal sur un *Samsung Galaxy S3*

Dans le deuxième cas, il peut choisir entre une visite dite "en ligne" utilisant les services de localisation, ou "hors ligne". Au travers de la totalité des classes, par héritage avec *BaseActivity* ou *BaseMapActivity*, l'utilisateur possède un accès à un sous-menu paramètre, permettant d'activer ou non le son, de prendre des photos, ou bien d'accéder au crédit. Il est important de noter que les crédits sont constamment accessibles sauf lorsque l'utilisateur visionne ceux-ci.

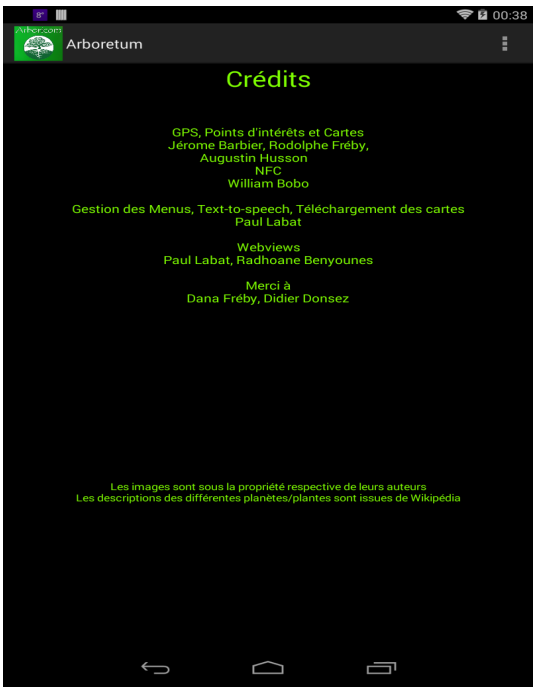

FIGURE 3 – Page des Crédits

La gestion du son est global, ainsi la désactivation de celui-ci (qui désactive le son des notifications lors de la visite en ligne) est prise en compte sur la totalité de l'application. Un autre point important réside dans le fait que bloquer le son ne bloque pas les notifications sonores de manière définitive. En effet, si un utilisateur désactive puis réactive le son, les objets devant lesquels il était déjà passé ne seront pas comptés comme ayant été dépassé, et peuvent ainsi émettre un son lors d'un passage de l'utilisateur devant le point d'intérêt en question. Il s'agit là d'un choix d'implémentation de notre part. Les photos prisent avec l'appareil possède un nommage numérique incrémentée automatiquement, et sont stockées dans */arboretum/photos/*. Une autre particularité de notre application est la création des différents répertoires nécessaires au bon fonctionnement de celle-ci, pour le téléchargement et le stockage des cartes au premier lancement, le stockage des éventuelles photos, ainsi que le stockage des fichiers HTML pour les webviews. En effet, Android permet une implémentation simple d'un système de gestion de fichiers par téléchargements.

4 Les cartes et les services de localisation

Notre application utilise deux cartes, la première étant celle de Grenoble, la seconde celle de l'arboretum. Il s'agit de fichiers *.map*. Les cartes possèdent différents points d'intérêts. Un des avantages de Mapsforge réside dans la facilité de gestion des différents marqueurs en version 0.3.0 par des objets *drawable*, gérés dans des overlays, au lieu de la version 0.3.1 permettant de créer des objets de type marker. Cependant, ces objets ne sont pas réellement à leur plein potentiel sans une optimisation externe.

En redéfinissant par une inner class java héritant de ArrayItemizedOverlay la méthode onTap(), il est possible de gérer la pression du doigt sur l'écran sur un marqueur, et d'entrainer ainsi une action associée, chose beaucoup plus difficilement réalisable avec un objet marker. Les points d'intérêts sont gérés par des GeoPoints représentant des coordonnées GPS gérées de manière statique dans la classe globale, pour être accessibles de partout.

FIGURE 4 – La liste des marqueurs choisis, de gauche à droite : pour les plantes, la position de l'arboretum, les informations, les planètes, la position de l'utilisateur

Ci-dessous, vous pouvez observer les différents choix de marqueurs qui ont été effectués :

FIGURE 5 – Carte de Grenoble avec les marqueurs de position de l'arboretum et de l'utilisateur

Il a été choisi de représenter l'utilisateur par un point plutôt qu'une flèche, du fait que nous n'avons pas réussi à implémenter la direction de celle-ci en fonction de l'orientation de la personne.

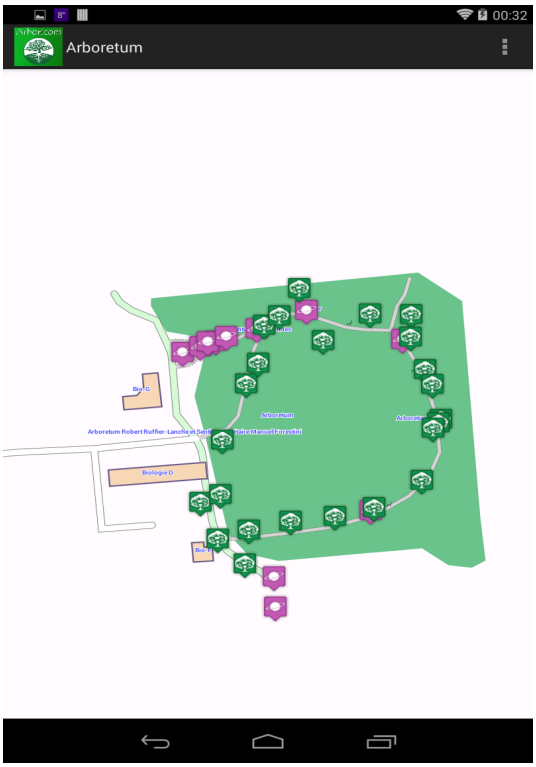

FIGURE 6 – Carte de l'Arboretum avec ses différents points d'intérêts

Pour les différents marqueurs des points d'intérêts, un clique sur ceux-ci entraine l'ouverture d'une boîte de dialogue. Il a été décider de passer celle-ci en noir, pour rendre son utilisation moins agressive à la vue. Cette boite de dialogue possède une icône représentant le type d'objet pour lequel la personne aurait besoin d'avoir plus d'informations. La personne peut au travers de cette boîte de dialogue choisir d'obtenir des informations supplémentaires sur le point d'intérêt.

Pour le bon fonctionnement des cartes avec l'affichage de la position de l'utilisateur, le GPS est l'une des composantes principales de notre application. Le GPS est utilisé à deux moments précis. Lors de la visite en ligne de l'Arboretum, et pour se rendre à celui-ci. Le GPS est complété par les services de localisation. Ils ne sont activés que lorsque leur utilisation est nécessaire. Ils sont également désactivés lors de la mise en veille de l'écran par appel de la méthode onPause(), et remis en état de marche par OnResume(). Ses services sont utilisés pour représenter la personne sur une carte.

Lors d'un passage proche d'un point d'intérêt, lors de la visite de l'arboretum en mode en ligne, un son de notification est joué si le son est activé. La détection se fait dans un carré de 3 mètres de côté centré sur le point en question. Deux sons différents sont disponibles, un pour les planètes, l'autre pour les plantes. Un problème dans l'application est à noter. La réactivation du son lors du passage proche d'un point d'intérêt entrainera un crash de l'application, par une erreur de mise en buffer du son.

Une particularité importante d'android à connaître réside dans le système de rotation de l'écran. Une rotation entraine l'utilisation de la méthode onDestroy(), et donc de la méthode onCreate(), réinitialisant l'activité. Nous avons donc décidé de bloquer la rotation de l'écran pour pallier à ce problème. Une autre solution aurait été d'utiliser des méthodes d'android permettant de sauvegarder différentes variables afin de les restaurer, mais plus compliquer à mettre en place et non important selon nous.

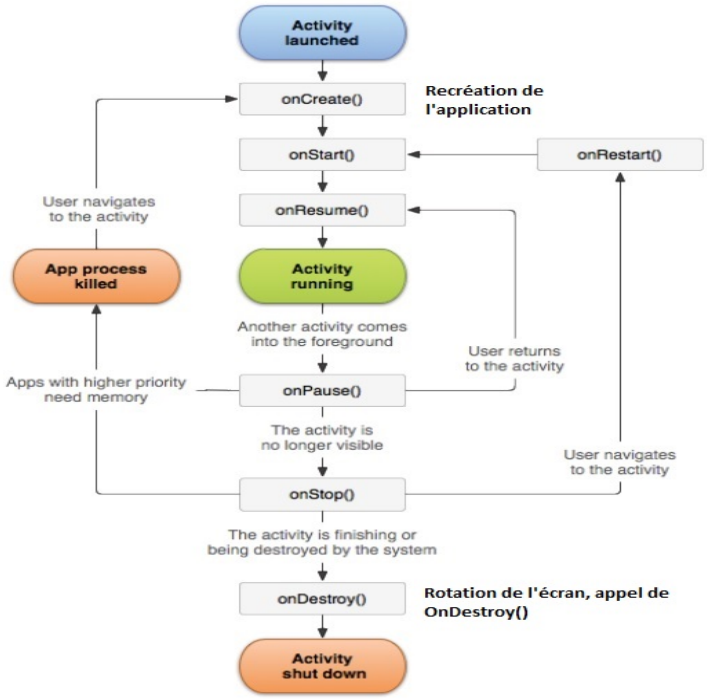

FIGURE 7 – Cycles de vie d'une application android

5 La gestion des Webviews et du Text-to-Speech

Les webviews utilisées dans notre application permettent d'afficher le contenu concernant les plantes/arbres et planètes. À chaque fois qu'un utilisateur clique sur un point d'intérêt, la webview correspondante est affichée à l'écran et est chargée dans une *activity* dédiée. Cela permet de revenir à la carte. Sans cela, les webviews seraient chargées dans la même activity que la carte, et lorsque l'utilisateur appuierait sur le bouton *retour*, il ne reviendrait pas à la carte, mais au menu de choix.

De plus, le étant autorisé dans les webviews, un bouton est implémenté dans chacune des pages web. Cela permet d'appeler une fonction JavaScript qui va à son tour appeler une fonction java. Cette dernière permet de lancer le *Text to speech* qui énoncera un texte. Le *Text to speech* est chargé au démarrage de l'application et est initialisé au lancement de la webview. La classe *WebAppInterface* sert d'interface pour utiliser la fonction java *Speakout* qui permet l'énonciation du texte. Lorsqu'un utilisateur quitte la webview, le TTS est arrêté (même s'il est en train d'être utilisé). Cela libère de la mémoire RAM.

Il y a cependant des limitations :

- Le texte est limité à 450 caractères en une seule fois
- Un nombre écrit avec des chiffres est converti en caractères alphabétiques
- Quelques problèmes d'élocution et de prononciation du texte

6 Le NFC

6.1 Lecture d'un tag

La lecture d'un tag nfc est effectuée par le service nfc. Lorsqu'il détecte un tag, il envoie un des 3 types d'intents à une des applications filtrant ces intents. Ces 3 intents sont NDEF_DISCOVERED, TECH_DISCOVERED et TAG_DISCOVERED. Vous pouvez voir comment est choisi le type d'intent envoyé dans le schéma suivant.

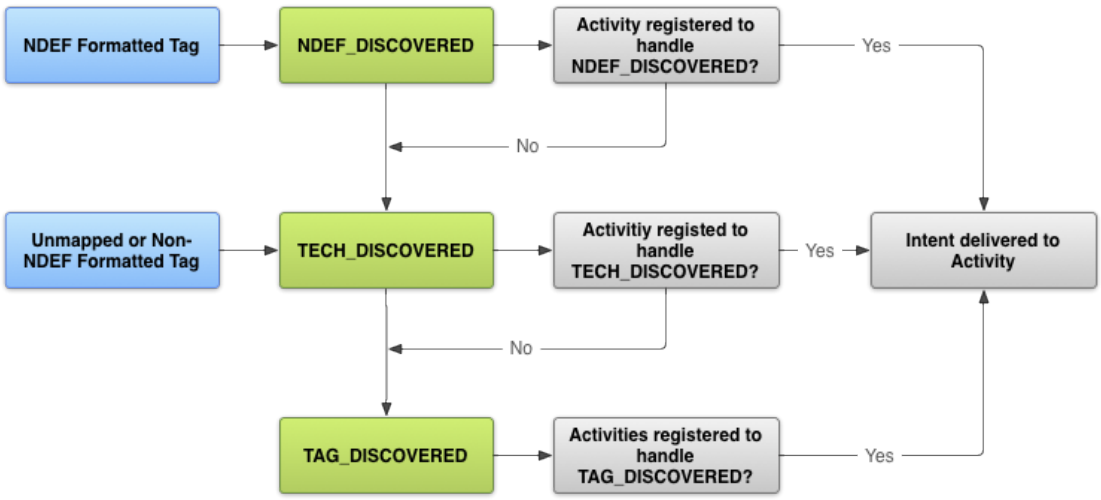

FIGURE 8 – Type d'intent

Au final, nous n'avons pas à choisir quel type d'intent nous recevons, car nos intent sont réceptionnés en foreground comme expliqué ci-après.

6.2 Écriture d'un tag

Pour écrire des tags spécifiques à notre application, plusieurs solutions étaient possibles. Nous avons choisi la plus simple qui est d'écrire dans le tag un message d'un type mime propre à notre application *application/com.ricm.arboretum*. Notre application est conçue pour lire des tags portant uniquement de ce type-là.

6.3 L'application

L'idée initiale était d'avoir une activity gérant seuls les intents émis par les tags NFC. Ce ne fut pas possible, car le service NFC ouvre l'activity dans sa task *service nfc*, on perd alors toutes les possibilités de retour dans l'application. Nous n'avons pas trouvé comment modifier ce mode d'ouverture et préféré adapter l'application à une nouvelle solution.

La seconde solution était d'intercepter tous les intents en foreground. C'est-à-dire qu'il n'est plus nécessaire d'inscrire notre activity à l'aide d'intent filter mais qu'à la place, si l'application est active au premier plan, elle interceptera les intents NFC (passant un filtre) directement. Cela pose un problème, l'application doit être ouverte pour réagir aux tags. Ce problème est en partie contourné en ouvrant l'application à l'aide d'AAR.

6.4 Android application records(AAR)

Nos tags étant prévus pour être en extérieur dans un lieu public, il arrivera que des appareils sans l'application tentent de lire nos tags. Dans les tags a été ajouté un message AAR (android application records). Si un appareil sans l'application tente de lire le tag, il sera redirigé sur la page Play store de l'application, dans le cas où l'application est déjà installée, elle sera simplement ouverte.

7 Crédits

Cette application Android a été réalisée par une équipe d'étudiants en Réseaux Informatiques et Communication Informatique composée par M. Jérôme BARBIER,Radhoane BENYOUNES, William BOBO, Rodolphe FREBY, Paul LABAT, supervisé par M. Augustin HUSSON et sous la tutelle de M. Jacques LEMORDANT.

8 Remerciements

À Dana Fréby :
- Pour avoir éffectué les relevés GPS nécessaires au positionnement des points d'interêts du parc.
- Pour avoir testé le GPS par temps couvert et dégagé.
- Pour avoir testé le son de l'application

À Didier Donzet, pour nous avoir gracieusement donné des tag NFC.