Voxeet Android
Rapport de stage

CASSAGNE Adrien

1 septembre 2011

Remerciements

Je tiens à remercier dans un premier temps, toute l'équipe pédagogique de l'université de Bordeaux 1 et les professeurs responsables de la 3ème année de licence informatique, pour avoir assuré la partie théorique de ma formation.

Je veux aussi dire ma reconnaissance aux personnes suivantes, pour l'expérience enrichissante et pleine d'intérêt qu'elles m'ont faite vivre durant ces trois mois au sein de l'entreprise Innovantic :

Monsieur Larry FORNALLAZ, président de la société Innovantic, ainsi que Benoît SENARD mon maître de stage et directeur général, pour leur accueil et la confiance qu'ils m'ont accordée dès mon arrivée dans l'entreprise.

Monsieur Ivan MEALLARES, développeur de la société Innovantic, mon tuteur, pour m'avoir intégré rapidement au sein de l'entreprise et m'avoir accordé toute sa confiance ; pour le temps qu'il m'a consacré tout au long de cette période, sachant répondre à toutes mes interrogations ; sans oublier sa participation au cheminement de ce rapport.

Messieurs Julien BESSE, Gilles BORDAS, Jean-Loïc HERVO, Jean-Marc MEMAIN, Raphaël REYNAUD, Thomas GOURGUES, Yves DELERM ainsi que l'ensemble du personnel d'Innovantic pour leur accueil sympathique et professionnel tout au long de ces trois mois.

Table des matières

1 Introduction **4**

2 Présentation de l'entreprise **5**
2.1 Présentation générale 5
2.2 Matériel et méthodes 5
2.3 Organigramme de la société 6
2.3.1 La direction 6
2.3.2 Le pôle R&D 6
2.3.3 Le pôle mobile 6
2.3.4 Le pôle applications riches 6

3 Le projet **7**
3.1 Présentation 7
3.1.1 Objectifs 7
3.1.2 Le choix des outils 8
3.1.3 Découpage du projet 8
3.2 Réalisation 10
3.2.1 Le service Voxeet Android 10
3.2.2 La liste des contacts 11
3.2.3 Le chat 12
3.2.4 Mise en place de la conférence 13

4 Conclusion **15**

Table des figures

1.1 Logo Voxeet 4

2.1 Logo Innovantic 5
2.2 Organigramme de la société 6

3.1 Capture d'écran : notifications du service 10
3.2 Capture d'écran : la liste des contacts 11
3.3 Capture d'écran : un exemple de chat avec un contact 13
3.4 Capture d'écran : un exemple de conférence 14

Chapitre 1

Introduction

Dans le cadre de la 3ème année de licence informatique de Bordeaux 1 j'ai effectué un stage d'une durée de 3 mois. J'ai choisi d'intégrer la société Innovantic à Mérignac sur le parc Innolin.

C'est pour la réalisation d'une application mobile que j'ai été recruté en tant que stagiaire.
Bien avant mon arrivée, Innovantic travaillait déjà sur la réalisation d'un projet ambitieux : Voxeet, un logiciel de conférence audio par internet avec spatialisation.

FIGURE 1.1 – Logo Voxeet

Ce projet, ayant pour objectif de se déployer rapidement et à grande échelle, M. Larry FORNALLAZ, M. Benoît SENARD et l'équipe de développement ont décidé de lancer l'application sur la majorité des plateformes (mobiles ou non). Ma directive principale était la suivante : Développer l'application Voxeet pour Android.

Pendant le stage Ivan MEALLERES, mon tuteur, m'a présenté le projet ainsi que l'existant dont il était en partie à l'origine (Application Voxeet .NET). Nous avons donc pu définir clairement les objectifs principaux du projet :

- Découvrir et apprendre les fondements de la technologie Android
- Rechercher une API (Interface de Programmation pour Application) pour la communication via le protocole XMPP (protocole de messagerie/présence) et apprendre à l'utiliser
- Réaliser le squelette de l'application avec les différentes rubriques et la logique de déplacement
- Ajouter les fonctionnalités suivantes : liste d'amis, *chat* et conférence avec le serveur Voxeet

Chapitre 2

Présentation de l'entreprise

2.1 Présentation générale

Innovantic est à l'origine une société d'édition logiciel fondée en 2005 par Larry FORNALLAZ et Benoît SENARD. Depuis elle a évolué en proposant du service. Innovantic est maintenant un société à double vocation : SSII (Société de Services en Ingénierie Informatique) et R&D (Recherche et développement). Elle est basée à Mérignac sur le parc Innolin et se compose de quinze personnes.

Les différents types de services proposés par la société sont :
- Développement d'applications riches.
- Réalisation d'applications web spécifiques.
- Consulting Java, .NET, PHP, etc.
- Développement d'applications pour mobile.

FIGURE 2.1 – Logo Innovantic

2.2 Matériel et méthodes

La plupart des postes de travail de la société utilisent Windows 7 Pro (32 ou 64bit). Cependant le pôle mobile travaille aussi sous Mac OSX pour les développements Apple (iPhone et iPad).

Les développeurs utilisent plusieurs environnements de développement : Visual Studio 2008 (environnement .NET) pour les programmes Windows ; pour les programmes Unix c'est majoritairement Eclipse qu'ils utilisent (langage Java), enfin c'est avec Xcode qu'ils développent sous iOS (Objective C).

La société est équipée d'un serveur Windows 2008 pour l'administration locale : ce dernier exécute plusieurs machines virtuelles selon les besoins des employés.

Enfin la société loue un serveur dédié qui est administré via SSH (protocole Secure Shell).

2.3 Organigramme de la société

La société se divise en 4 pôles bien distincts :
- La direction
- Le pôle recherche et développement
- Le pôle mobile (téléphone, tablette, etc.)
- Le pôle applications riches (Java, .NET)

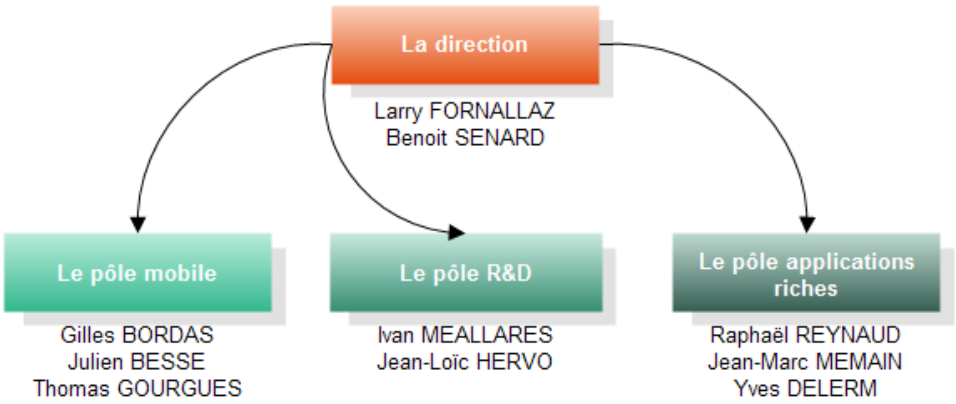

FIGURE 2.2 – Organigramme de la société

2.3.1 La direction

Larry FORNALLAZ et Benoît SENARD assurent l'administration de l'entreprise ainsi que la direction technique. Ils proposent aussi leurs compétences aux autres entreprises en tant que consultants.

2.3.2 Le pôle R&D

Il est consitué d'Ivan MEALLARES et Jean-Loïc HERVO. Tout deux travaillent actuellement au développement de Voxeet. J'ai fait partie de ce pôle tout au long de mon stage en travaillant sur Voxeet Android. J'ai aussi reçu de l'aide en provenance des autres pôles.

2.3.3 Le pôle mobile

Le pôle mobile propose du forfait ou du service sur les plateformes Android et iPhone/iPad. Julien BESSE, Gilles BORDAS et Thomas GOURGUES travaillent sur une application de domotique (iPhone/iPad).

2.3.4 Le pôle applications riches

Comme le pôle mobile, il propose du forfait ou du service suivant la demande des clients. Actuellement Jean-Marc MEMAIN, Raphaël REYNAUD et Yves DELERM travaillent sur une application de gestion pour une entreprise.

Chapitre 3

Le projet

3.1 Présentation

3.1.1 Objectifs

M. Benoît SENARD et M. Larry FORNALLAZ sont les initiateurs du projet Voxeet. Il consiste en la réalisation d'un logiciel de conférence audio, à l'image de Skype.
Voxeet se démarque de Skype par des fonctionnalités très innovantes comme la spatialisation dans une conférence, la prise en compte réverbération d'une salle, la gestion du volume sonore pour chaque personne, la restitution des ambiances sonores, etc.
La spacialisation permet de retranscrire la position des interlocuteurs dans l'espace par le son, c'est ce que notre cerveau fait au quotidien (en effet, en conditions réelles, nos deux oreilles permettent de déterminer la position d'une personne dans l'espace).
L'objectif final de Voxeet est d'obtenir un son naturel et donc proche de la réalité afin que son rendu soit confortable et reposant pour l'utilisateur.

Une version bêta du logiciel est prévue très prochainement. Innovantic souhaite la diffuser gratuitement sur internet. Afin de toucher le maximum d'utilisateurs potentiels, Innovantic a aussi prévu de lancer Voxeet sur iPhone et téléphone Android. L'objectif est d'être multiplateforme le plus rapidement possible pour que Voxeet soit utilisable partout sans aucune restriction (on remarquera que c'est aussi la politique actuelle de Skype).

C'est dans ce cadre que nous avons défini les limites de mon projet pour Android. Voici la liste de mes directives de travail (par ordre chronologique) :

- Installer et lancer l'environnement de développement Android
- Découvrir l'API Android et apprendre l'architecture de base d'un projet
- Rechercher une API Android pour communiquer avec XMPP (protocole de messagerie)
- Tester l'API XMPP pour s'assurer qu'elle correspond au besoin
- Créer le projet Voxeet pour qu'il soit compatible avec Android 2.2
- Créer la fenêtre (appelée "activité" sous Android) de login
- Afficher la liste des contacts et de leur présence
- Mettre en place un service effectuant la connexion avec le serveur XMPP
- Créer une activité de *chat* avec un contact
- Afficher la liste des *chats* en cours
- Créer une activité de conférence (très important pour la spacialisation)

– Afficher la liste des conférences

3.1.2 Le choix des outils

Le langage de programmation

La plateforme Android laisse le choix entre deux langages de programmation : le Java et le C++.
Le Java à l'avantage de profiter de l'API complète d'Android sur laquelle beaucoup d'outils sont disponibles. Le C++ est le langage natif du système ; il est ainsi beaucoup plus rapide que le Java. Par contre il ne bénéficie pas de l'API complète : l'environnement est minimaliste.
J'ai fait le choix d'utiliser le Java avec sa machine virtuelle (Dalvik) car mon application ne nécessitait pas de performances particulières.

L'environnement de développement

Google propose tous les outils nécessaires pour développer avec Eclipse. Je n'ai donc pas hésité et j'ai choisi Eclipse qui est aujourd'hui un des IDE les plus complets. J'en ai aussi profité pour installer Subversive, plugin dédié à la gestion des versions du code source, et compatible SVN.
Google propose de créer des téléphones Android virtuels pour tester les applications, mais j'ai surtout utilisé le téléphone Android que m'a fourni la société. Grâce à ce dernier et un câble USB, j'ai pu *débugger* mon application via Eclipse sans aucun problème.

La bibliothèque de communication XMPP

Afin d'assurer la communication entre le serveur Voxeet et mon application, il a fallu que je recherche une bibliothèque suivant la norme XMPP.
Pour rappel, XMPP (Extensible Messaging and Presence Protocol) est un protocole de messagerie/présence sur lequel est basé le serveur Voxeet.
Dans ma recherche, Benoît SENARD m'a indiqué une très bonne bibliothèque disponible en Java : Smack.
J'ai cherché dans cette direction et j'ai fini par trouver un portage de cette bibliothèque sous Android :
http ://florentgarin.developpez.com/tutoriel/android/client-xmpp/.
Je salue au passage Florent GARIN pour son portage de très bonne qualité.

3.1.3 Découpage du projet

Mon projet se divise en deux parties bien distinctes. La première est un service invisible pour l'utilisateur et la deuxième est l'application à proprement parler.
La question qui vient naturellement est : Pourquoi cette division est-elle vraiment nécessaire ?

Pour comprendre ce choix, il est utile de revenir sur quelques principes élémentaires d'une application Android, et en premier lieu sur le concept d'activité : une activité sous Android est un contrôleur auquel on associe une vue. Pour donner un exemple, un écran de login est géré par une activité, une liste de contacts est aussi gérée par une activité.

Revenons à notre question : par définition, toute application peut être sérialisée et mise en veille par le système d'exploitation et cela à n'importe quel moment. De plus, il existe une limitation pour passer des informations entre deux activités : ils faut que les informations soient *parcelables* (ou sérialisables, c'est le même mécanisme). Quand une activité passe la main à une autre activité, elle est automatiquement serialisée par Android.
C'est là tout le problème : Comment rester connecté au serveur Voxeet alors que notre application est suceptible d'être mise dans un état de sommeil ?
La solution est d'utiliser un service auquel notre application se connecte pour récupérer les informations dont elle a besoin. C'est ce service qui restera connecté à Voxeet et non l'application.

Le service

Dans Voxeet Android, le service récupère la liste des contacts, les messages envoyés et reçus, la liste des *chats* et des conférences.

L'application

L'application est composée de plusieurs activités :
- La connexion à Voxeet
- La liste des contacts et leur présence
- La liste des *chats* ouverts
- Un *chat* en particulier
- La liste des conférences
- Une conférence en particulier

3.2 Réalisation

3.2.1 Le service Voxeet Android

Le service assure la gestion de la connexion au serveur de Voxeet. Si l'application est fermée, le service reste ouvert : il reçoit les messages, les invitations, les connexions, etc.
Quand un évènement de ce type se produit, le service informe Android au moyen d'une notification. L'utilisateur est aussitôt averti de la nouveauté. En sélectionnant la notification, l'application Voxeet se lance sur la bonne activité (ex. : si quelqu'un veut discuter avec l'utilisateur, alors c'est la fenêtre de *chat* qui s'ouvre).

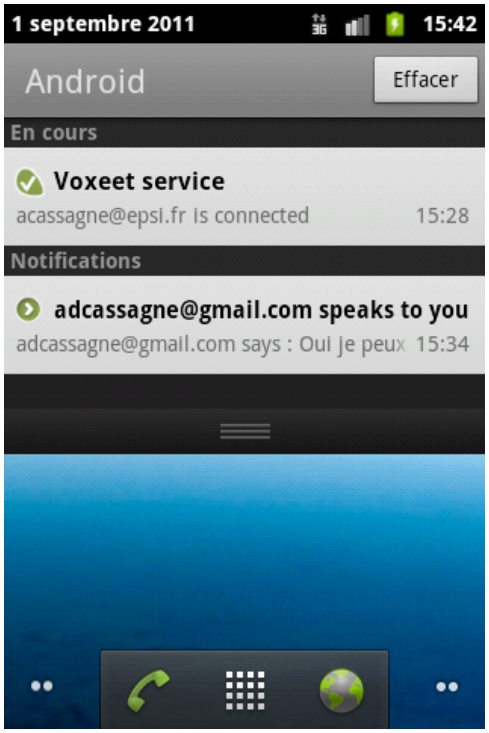

FIGURE 3.1 – Capture d'écran : notifications du service

Voici un extrait condensé de mon service sous Android. On remarquera qu'un service hérite directement de la classe "Service". Dans mon cas, il implémente aussi une multitude d'interfaces. Cela permet de simuler le comportement d'évènements. Chacune de ces interfaces est définie par Smack (bibliothèque de communication avec le serveur XMPP).

```
1  public class VoxeetService extends Service implements RosterListener,
2  ConnectionListener, ChatManagerListener, MessageListener{
3
4      ////////////////////////////////////////////////////////////////
5      // Méthodes propres au Service qu'il est conseillé de redéfinir
6
7      // Création du service
8      public void onCreate()...
```

```
9
10      // Appel au service : connexion d'un client
11      public IBinder onBind(Intent intent)...
12
13      // Déconnexion de tous les clients
14      public boolean onUnbind(Intent intent)...
15
16      // Destruction du service
17      public void onDestroy()...
18
19      ////////////////////////////////////////////////////////////
20      // Méthode de l'interface "RosterListener"
21      // => évènements liés à la liste des contacts
22
23      public void entriesAdded(Collection<String> arg0)...
24      public void entriesDeleted(Collection<String> arg0)...
25      public void entriesUpdated(Collection<String> arg0)...
26      public void presenceChanged(Presence presence)...
27
28      // C'est le même principe pour les autres interfaces
29  }
```

3.2.2 La liste des contacts

Cette activité est très importante puisqu'elle est le point de départ de l'application. C'est elle qui permet de visualiser les personnes avec qui l'utilisateur peut intéragir. Elle permet aussi de voir les présences des contacts (en ligne, hors ligne). Depuis cet écran il est possible de sélectionner un contact pour *chater* directement avec lui ou d'en sélectionner plusieurs pour créer une conférence.

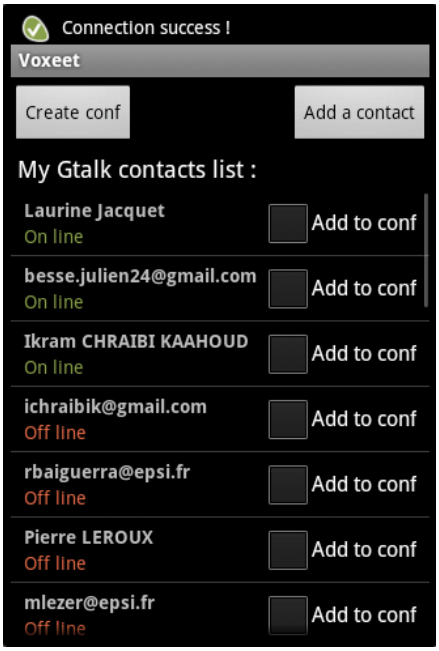

FIGURE 3.2 – Capture d'écran : la liste des contacts

Voici un extrait de l'activité qui gère la liste des contacts. Chaque activité à un cycle de vie (onStart, onStop dans l'exemple). En réalité ce cycle est plus complexe que dans l'exemple proposé mais je n'entre pas dans le détail afin de rester dans les limites du rapport. On peut voir que chaque activité se synchronise avec le service Voxeet (connexion au service lors du démarrage de l'activité, déconnexion à l'arrêt de l'activité).

```
1   public class ContactActivity extends Activity implements ServiceConnection {
2
3       //Cet attribut est la connexion entre l'activité et le service
4       private IRemoteVoxeetService serviceVoxeet;
5
6       ////////////////////////////////////////////////////////////////
7       // Méthodes propres à l'Activity qu'il est conseillé de redéfinir
8
9       // Création de l'activité
10      public void onCreate(Bundle savedInstanceState)...
11
12      // Démarrage de l'activité
13      protected void onStart(){
14          super.onStart();
15          this.bindVoxeetService(); // Connexion au service
16      }
17
18      // Arrêt de l'activité
19      protected void onStop(){
20          super.onStop();
21          this.unbindVoxeetService(); // Déconnexion au service
22      }
23
24      ////////////////////////////////////////////////////////////////
25      // Méthodes de l'interface "ServiceConnection"
26
27      public void onServiceConnected(ComponentName name, IBinder service){
28          //Quand le service est connecté on passe ici
29          this.serviceVoxeet = IRemoteVoxeetService.Stub.asInterface(service);
30      }
31
32      public void onServiceDisconnected(ComponentName name)...
33  }
```

3.2.3 Le chat

Le *chat* intègre les fonctionnalités de base, à savoir :
- La réception de messages
- l'envoi de messages
- l'affichage du nom du destinataire

Il est intéressant de décrire son architecture. L'activité "ChatActivity" s'abonne au service Voxeet pour la réception des messages. C'est de cette façon qu'elle est capable d'afficher les nouveaux messages en temps réel. En pratique, on implémente une interface à laquelle on s'abonne. Les méthodes qui correspondent à l'interface

sont ensuite appelées au niveau de l'activité.

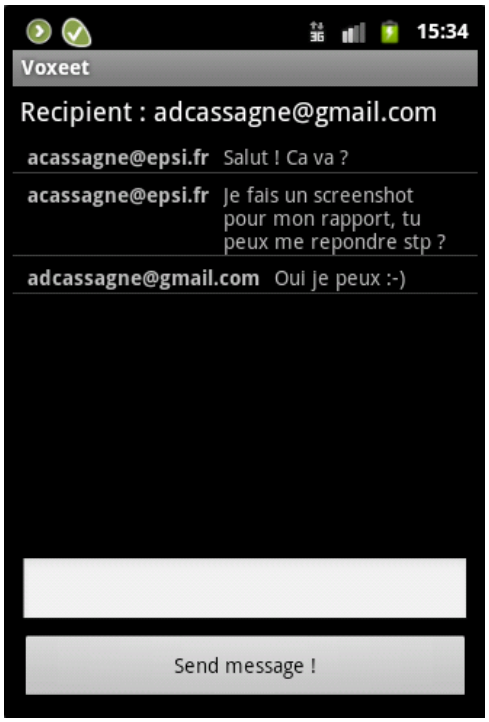

FIGURE 3.3 – Capture d'écran : un exemple de chat avec un contact

3.2.4 Mise en place de la conférence

La mise en place de la conférence est un des points les plus intéressants du projet. L'objectif final de Voxeet Android est de recréer une conférence audio avec spacialisation du son.
Cependant, ce n'est pas sur cet aspect que j'ai travaillé ; mon objectif était plutôt de créer une fenêtre où l'on pouvait sélectionner des avatars et les déplacer via l'écran tactile. Pour chaque avatar il convient de mémoriser sa position. Cela est nécessaire pour ajouter la couche audio dans le futur.
J'ai aussi rajouté la possibilité de double cliquer sur un contact : cela ouvre l'activité de *chat* avec le contact sélectionné.

Le comportement standard d'Android permet de générer automatiquement la vue associée à l'activité depuis un fichier XML (avec des widgets prédéfinis : textbar, button, listView, image, etc.). Pour réaliser la fonction conférence, j'ai dû descendre à un niveau inférieur aux précédentes activités.

Il fallait maintenant gérer une surface sur laquelle on pouvait déplacer des objets avec l'écran tactile. Pour cela Android fournit des outils mais ils sont plus rudimentaires. J'ai donc dû créer ma propre vue avec sa propore gestion des évènements (contacts avec l'écran tactile essentiellement). Il m'a fallu également gérer le renouvellement de l'image, pour cela j'ai utilisé un *thread* dédié au rafraîchissement.

Sur la capture d'écran suivante on peut voir une conférence avec six personnes. Chaque avatar est sélectionnable. Les avatars peuvent être placés les uns sur les autres.

FIGURE 3.4 – Capture d'écran : un exemple de conférence

Chapitre 4

Conclusion

Pendant une grande partie de mon stage j'ai travaillé sur l'API Android. Cela m'a appris à relire et à comprendre le code élaboré par d'autres développeurs. Malgré quelques difficultés liées à la complexité de l'environnement, j'ai pris mes repères petit à petit.
J'ai eu recours à la documentation présente majoritairement sur internet afin de comprendre les principes fondamentaux de ce système. Après quelques semaines j'ai pu mesurer tout l'intérêt de ce dernier.

Dans le cadre plus précis de l'élaboration de mon application Android, j'ai bénéficié de l'aide précieuse des développeurs qui m'entouraient. Ainsi, grâce à ce stage j'ai pu découvrir les différents aspects du travail en équipe.
Ce stage aura été pour moi l'occasion d'approfondir le langage Java. C'est une expérience très enrichissante car ce langage est complètement objet et est très répandu dans le monde de l'informatique. De plus j'ai eu l'occasion de découvrir de nouvelles bibliothèques techniques très intéressantes (Smack, API Android) et très utiles dans le monde mobile.
Enfin j'ai toujours été en contact avec plusieurs développeurs de Voxeet qui m'ont orienté. Ils m'ont permis de redéfinir mes objectifs en continu pour atteindre le résultat le plus proche possible de leurs attentes. Ce fut une expérience positive dans laquelle j'ai travaillé sereinement, grâce à la bonne ambiance qui régnait au sein de cette entreprise.

Bibliographie

[1] F. Garin, *Android*. Dunod, 2009.

[2] L. Vogel, "Tutorial très complet sur la programmation android (en anglais)," 2009-2011. [Online]. Available : http://www.vogella.de/articles/Android/article.html

[3] N. BENBOURAHLA, "Introduction à la programmation sous android," 2011. [Online]. Available : http://nbenbourahla.developpez.com/tutoriels/android/introduction-programmation-android/

[4] F. GARIN, "Créer un client xmpp android," 2009. [Online]. Available : http://florentgarin.developpez.com/tutoriel/android/client-xmpp/

[5] J. Software, "La documentation officielle de smack," 2009. [Online]. Available : http://www.igniterealtime.org/builds/smack/docs/latest/documentation/

[6] Axon, "Intent : passer d'une activity à une autre," 2010. [Online]. Available : http://www.tutomobile.fr/intent-passer-dune-activity-a-une-autre-tutoriel-android-n%C2%B011/16/07/2010/

[7] B. Nazim, "Les services sous android," 2011. [Online]. Available : http://www.tutos-android.com/service-android

[8] N. Druet, "Article complet : Utiliser les services sous android," 2010. [Online]. Available : http://blog.developpez.com/android23/p8571/android/creation-de-service/

[9] F. Mora, "Collections en java : Prise en main," 2007. [Online]. Available : http://fmora.developpez.com/tutoriel/java/collections/introduction/

[10] Google, "Création d'une notification dans la barre de status (en anglais)." [Online]. Available : http://developer.android.com/guide/topics/ui/notifiers/notifications.html

[11] KANN, "Mise en forme basique du texte du hello world !" 2010. [Online]. Available : http://www.androideur.com/mise-en-forme-basique-du-texte-du-hello-world

[12] P.-E. Mercier, "Les listview," 2010. [Online]. Available : http://www.ace-art.fr/wordpress/2010/07/21/tutoriel-android-partie-6-les-listview/

[13] B. Nazim, "Broadcast receiver sous android," 2011. [Online]. Available : http://www.tutos-android.com/broadcast-receiver-android

[14] T. Jano, "Une boucle de jeu basique (en anglais)," 2010. [Online]. Available : http://obviam.net/index.php/a-very-basic-the-game-loop-for-android/

[15] ——, "Bouger des images sur l'écran avec android (en anglais)," 2010. [Online]. Available : http://obviam.net/index.php/moving-images-on-the-screen-with-androi/

[16] Martin, "Jouer avec les graphismes sous android (en anglais)," 2009. [Online]. Available : http://www.droidnova.com/playing-with-graphics-in-android-part-i,147.html

[17] N. Chambrier, "Développer des applications pour android," 2010. [Online]. Available : http://www.clever-age.com/veille/blog/developper-des-applications-pour-android.html

[18] Google, "Activity." [Online]. Available : http://developer.android.com/reference/android/app/Activity.html

[19] Ricky81, "Introspection en java, présentation de l'api reflection," 2004. [Online]. Available : http://ricky81.developpez.com/tutoriel/java/api/reflection/

[20] tylenoly, "Comment terminer un activité avec des résultats (en anglais)," 2010. [Online]. Available : http://tylenoly.wordpress.com/2010/10/27/how-to-finish-activity-with-results/

[21] Google, "Screens support." [Online]. Available : http://developer.android.com/guide/practices/screens_support.html