République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Licence en Informatique

Thème

Application Smartphone programmation sur Android

Réalisé par :

- **MAKASI Tendai**
- **JALO Muhamadu Aly**

Présenté le 10 Juin 2014 devant la commission d'examination composée de MM.

- *BENAZZOUZ .M* *(Encadreur)*
- *SMAHI .I.M* *(Examinateur)*
- *MESSABIHI .M* *(Examinateur)*

Année universitaire : 2013-2014

REMERCIEMENTS

Nous tenons à remercier notre encadreur Mr M. Benazzouz pour son disponibilité et ses conseils, son orientation et surtout son aide. Il nous a toujours guide dans la bonne direction dans notre travail.

Nous remercions l'ensemble de jury qui a bien voulu examiner et évalué ce mémoire.

Enfin, nous remercions notre cher ami W. P. Phiri et toutes les personnes qui ont participé de près ou de loin à la réalisation de ce projet.

Dédicace :

A mes chers parents, à mes frères et à mes sœurs,

Malgré la distance qui nous sépare, je vaudrais partager mon succès avec eux. Que Dieu le Tout Puissant les protège ;

A toute ma famille, à tous mes amis et collègues,

Pour leur soutien et leur confort, je les dédie ce travail.

Aly.

I dedicate this work to my family

And to my brother, the late D. P. Makasi.

I'll always cherish and remember you and I'd like to thank the Lord for all the days and moments he gave us with you "General".

May your soul rest in eternal peace.

Tendai

Table des matières

Introduction

Problématique

Dans le monde actuel, la communication est la chose plus importante de nos quotidiens, pour cela en vue des difficultés des étudiants étrangers (les anglophones et lusophones) et la difficulté de quelque pourcentage des Algériens à s'exprimer et dialoguer en langue française par manque de connaissance du vocabulaires français, nous a été motivé à faire une application de Jeux des mots pour les Smartphones, car ce derniers ont de beaux jours devant eux dans le monde.

Contribution

La contribution que nous apportons c'est le courage et la force que nous donnons pour les futures étudiants qui vont vouloir faire des applications pour Smartphones en les laissant les codes ouverts de notre application et toutes les explications nécessaires pour qu'ils puissent faire des applications plus complexe et plus dynamique et plus riche, et nous espérons aider plusieurs personnes qui ont des difficulté en vocabulaire à surmonter ce problèmes grâces à notre jeux.

Chapitre 1

Programmation pour les Smartphones

I.1 Introduction :

Les Smartphones sont des appareils extrêmement sophistiqués, qui fournissent des fonctionnalités en plus de celles des téléphones mobiles classiques comme la télévision, la navigation sur le web, la consultation et l'envoi de courrier électroniques, la messagerie vocale et visuelle, etc. Dernièrement, on a aussi vu les Smartphones sophistiqués bénéficiant rapidement de la reconnaissance et synthèse vocale. Les Smartphones exécutent tous divers logiciels et applications grâce à des systèmes d'exploitation spécialement conçus pour les mobiles. Les Smartphones peuvent être personnalisés en y installant des applications additionnelles telles que des jeux ou des utilitaires grâce aux magasins d'applications en ligne (stores).

Le premier Smartphone, l'IBM Simon, fut conçu en 1992. Il a été commercialisé en août 1994. Des nouvelles sociétés spécialisées dans les Smartphones, comme Research In Motion (avec le BlackBerry) seront introduits parmi les principaux fabricants de téléphones classiques (comme Samsung, Nokia, LG) qui avaient déjà pris l'initiative de se lancer dans l'aventure. A partir de la fin de 2007, le marché des Smartphones s'étend considérablement jusqu'à dépasser en quelque années celui des téléphones mobiles classiques.

Il existe certaines contraintes pour le développement et programmation smartphone, qui ne s'appliquent pas au développement habituel. Au moment, la mémoire RAM sur les téléphones est limite à 512 Mo, ce qui implique qu'on peut lancer moins de logiciels à la fois et donc les logiciels doivent faire en sorte de réserver moins de mémoire.

Il est aussi important de prendre en compte que nos applications (programs) doivent pouvoir interagir avec un system complet sans l'interrompre. C'est-à-dire il faut respecter une certaine priorité des tâches, par exemple les systèmes permettent de recevoir des messages et des appels pendant l'utilisation d'une autre application. La variation de la taille des écrans, doit être considérée encore lors de la réalisation des applications pour les smartphones. [1]

I.2 Les Différent Systèmes d'Exploitation :

Il existe plusieurs systèmes d'exploitation spécifiques aux Smartphones. Selon l'article publié en 2013 par l'International Data Corporation (IDC) [2] les systèmes d'exploitation les plus utilises parmi les autres sont Android, iOS et Windows Phone. L'usage des systèmes Android et iOS était dominant constituent 92.3% des ventes mondialementà la fin de 2013.

Mobile plateforme	Android	iOS	Windows mobile	BlackBerry
Société	Google	Apple	Microsoft	Rim
Caractéristiques	-Code source ouvert -APIs ouvert -Minimum de 128 Mo RAM -Stockage amovible - Radio et la fonctionnalité de téléphonie		-la compatibilité avec tous les logiciels de « Windows » - multi tâche : robuste -smartdial	-Standard d'or pour courriel -Multitâches -Bien intégration avec autres plateformes -Déploiement facile -Sécurité haute

Langage de programmation	Java	C	C++,C#/VB	Java
IDE +SDK	Eclipse(Android SDK pour Windows et Mac OS) -Netbeans (Android plugin)	XCode IDE, iOS SDK	Visual tools (gratuit)	BlackBerry JDE plugin sous visual studio et Eclipse. -BlackBerry MDS studio avec BlackBerry MDS Runtime
Environnement	Windows, Linux	Mac	Windows, Mac, Linux	Windows, Mac

Tableau 1.1 : Tableau de comparaison entre les plateformes. [3]

D'autressystèmes d'exploitation existent aussi comme :

- MeeGo -développé par Intel et Nokia.
- Bada -développé par Samsung.
- WebOS -développé par Palm, puis HP.

Nombreux de ces systèmes utilisent le moteur de rendu HTML WebKit intégré dans un navigateur pour l'affichage des sites sur la toile.

Fin 2013, un nouveau système d'exploitation, base sur Ubuntu a été créé par Mark Shuttleworth. Il est déjà fonctionnel et est compatible avec les Galaxy Nexus7 et Nexus 4.

I.2.I: Android

Android était développé par la startup Android Inc. en 2003, puis racheté par Google en 2005. Pour pouvoir réalisé un système complet, ouvert et gratuit dans le monde du mobile, une coalition de 35 entreprise évoluant dans l'univers du mobile, dont Google, a été créé. Ce rassemblement se nomme l'Open Handset Alliance (OHA) et compose aujourd'hui de 80 membres. Cette alliance a pour but de développer un système open source ‹‹ c'est-à-dire dont les sources sont disponible librement sur internet ›› pour l'exploitation sur mobile, Android.

Android est à l'heure actuelle le système d'exploitation pour smartphones et tablettes le plus utilisé. Les terminaux visés par Android inclus les téléphones portables, Netbook/Smartbook, tablettes multimédia, automobile, GPS, Réfrigérateur, etc. [4]

Les Versions :

Au moment, Android est disponible en version 4.4, (Kit Kat). Les versions se succèdent rapidement et les changements qui les accompagnent sont souvent conséquents en termes de nouvelles fonctionnalités et d'améliorations.

Version Android	**Nom de la version**	**Date de release**	**Quelques évolutions**
4.4	KitKat	2014-04-14	Interface translucide,Framework pour imprimer, Framework pour la gestion des fichiers.
4.1	Jelly Bean	2012-07-09	Assistance vocale, accessibilité : mode gestuel 'Braille', WIFI-Direct service discovery, vsync timing
4.0	IceCream Sandwich	2011-10-19	WI-FI direct, Bluetooth Health Devie profile, Control over network data, Grid Layout.

3.2	Honeycomb	2011-07-15	Support des processeurs Qualcomm, Support des tablettes tactiles de 7 pouces
2.3	Gingerbread	2010-12-06	Support de la VoIP et SIP. Gestionnaire de téléchargement, support de plusieurs cameras.
2.2	Froyo	2010-05-20	Implementation de JIT, partage de connexion USB.
2.0	Eclair	2009-10-26	Bluetooth, support de plus de taille d'écran.
1.6	Donut	2009-09-15	Google navigation (GPS gratuit)
1.5	Cupcake	2009-04-30	Envoi de vidéos vers YouTube et Picasa, rotation automatique
1.1	Banana bread	2009-02-09	Support pour sauvegarder les fichiers attachent aux MMS.
1.0	Apple pie	2008-09-23	Début de l'aventure Android.

Tableau 1.2 : L'évolution de versions d'Android. [5]

L'environnement de développement est la base sur une architecture autour du noyau Linux. La plateforme Android est compose de cinq couches principales :

- Un noyau Linux qui lui confère des caractéristiques multitâches.
- Des bibliothèques graphiques, multimédias.
- La Dalvik Virtuel Machine, une machine virtuelle adaptée pour java.
- Une plateforme applicative pour la gestion des fenêtres, du contenu, de téléphonie, etc.
- Des applications.

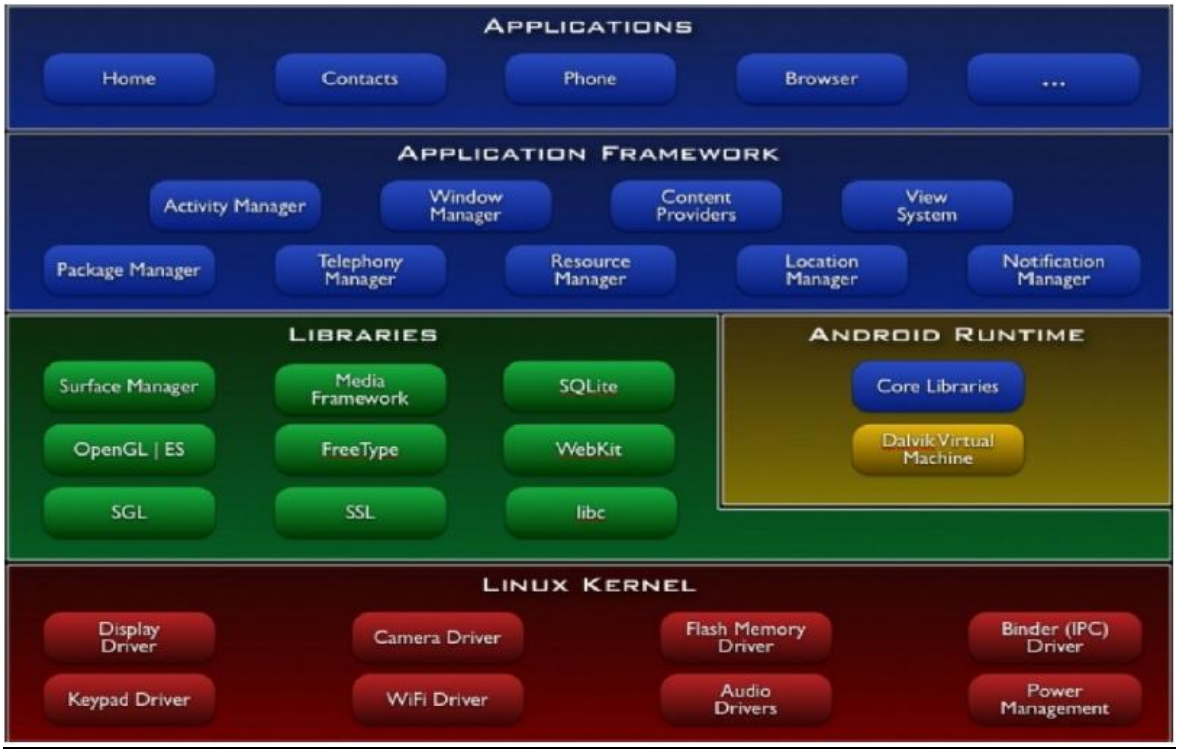

Fig. 1.1: L'Architecture de la plateforme Android. [6]

Pour pouvoir développer sur la plateforme Android, il faut accepter la licence Apache 2.0 associe lorsque vous téléchargez le développement kit. La licence autorise la modification sous forme libre ou non et permet d'en faire un usage commercial (car il est open source).

Android offre un système standard de téléchargement d'applications sur les Play Store. Les applications sont classées sur Play Store selon différents critères comme par exemple : catégorie d'âge. Le Play Store offre aussi une possibilité de rendre les applications payantes. Pour mettre une application sur Play Store, il suffit de payer $25 ce qui permet de publier autant d'applications que vous le souhaitez à vie. [7]

I.2.II : iOS

iOS était développé en 2007 par la franchise Américaine, Apple principalement pour utilisation sur l'iPhone. Il est dérivé du système d'exploitation MAC OS X qui s'exécute dans un système base sur l'UNIX. iOS est à l'heure utilisé par plusieurs devises de la franchise comme l'iPod Touch et l'iPad, l'iPad Mini et la deuxième génération d'Apple TV.

L'interface iOS est basée sur le concept de manipulation direct en utilisant les gestes multiples (multi-touchgestures). L'interaction avec l'OS se fait grâce aux différents gestes comme swipe, tap, pinch, les accéléromètres internes, etc. [8]

Les Versions :

Depuis l'apparition de la premier iOS 1.0 en 2007, Apple publie les nouvelles versions avec des améliorations chaque année qui sont toujours accessibles à travers d'Apple Store.

Version d'OS	**Date de publication**	**Quelques évolutions**
iOS 1.0	2007-03-06	La première version du SE mobile d'Apple, considère comme une version du SE du bureau d'Apple.
iOS 2.0	2008-07-11	Introduction d'un magasin d'applications tierce, l'App Store
iOS 3.0	2009-06-17	Nouvelle application, Dictaphone, permettant l'enregistrement des fichiers son.
iOS 4.0	2010-06-21	Introduction de multitâche et Facetime.
iOS 5.0	2011-06-06	Intégration d'iMessage et de Kiosque.
iOS 6.0	2012-06-11	Support de FaceTime sur les networks mobiles, nouveau privacycontrols.
iOS 7.0	2013-06-10	Ajout de Control Center, Touch ID scanneur

Tableau1.3:L'évolution d'iOS. [9]

Le développement des applications iOS nécessite un ordinateur Macintosh (Intel based), avec l'iOS SDK et ‹‹l'Apple XCode developemment tool ›› qui support la programmation orienté objet en langage C++. Les applications iOS s'exécutent sur un système base sur l'UNIX. Il suffit de s'inscrire gratuitement comme un développeur d'Apple sur le site ‹‹developer.apple.com›› pour avoir un accès total à tous les outils nécessaires pour développer sous iOS. [10]

Apple offre une plateforme pour télécharger des applications sur App Store. Pour mettre une application disponible sur l'App Store comme un développeur, il suffit d'être inscrit en payant $100 par an. [11]

1.2 III: Windows Mobile

Windows Mobile est développé par la corporation Microsoft. Alors, Windows Mobile a la capacité de fonctionnement des logiciels sur « Windows » (seulement Windows), compatible avec tous les logiciels de « Windows ».

Les outils nécessaires pour pouvoir réaliser des applications pour Windows Phone sont :

- Windows Phone Developper Tools
- Visual Studio 2010 Express for Windows Phone
- XNA Game Studio4.0
- On-Screen Windows Emulateur.

Ces outils sont disponibles sur le site ‹‹http://developer.windowsphone.com››.

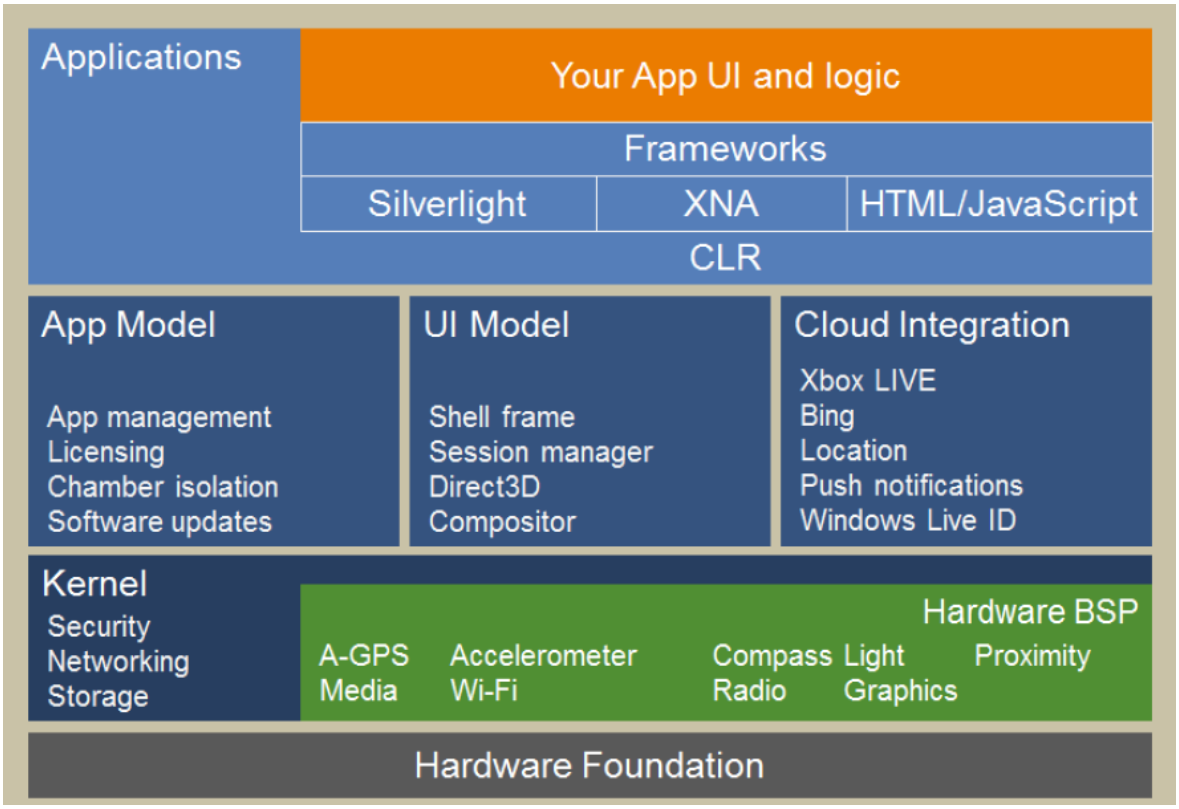

Fig. 1.2:Architecture de Windows Phone 7.

Les applications sont écrits avec langage de programmation c#. Pour rendre les applications développées sur cette plateforme installables sur un Windows Phone actuel, il faut s'inscrire sur le site ‹‹http://developer.windowsphone.com›› pour avoir la capacité de faire ça du Visual Studio. [12]

Chapitre 2

Création d'un Projet Android

II.1: Introduction :

Dans ce chapitre, nous allons décrire les étapes de création d'un simple projet Android, à savoir HelloWord et nous expliquerons la manière de le tester sur un émulateur Android.

II.2 L'architecture d'une application:

En général les applications Android ont les architectures suivantes

II.2.1 Activity :

La plupart des applications se compose plusieurs écrans. Chaque écran peut être réalisé par une activité. Si un nouvel écran s'ouvre, le système utilise une pile d'histoire pour stocker les écrans précédents et pouvoir reprendre l'état précédent ou enlever cet état.

II.2.2 Broadcast IntentReceiver :

On peut utiliser Broadcast IntentReceiver pour exécuter les réactions sur les événements extérieurs. Par exemple, l'application exécutera automatiquement si une Broadcast Receiver est éveillé (trigger).

II.2.3 Service :

Une Service est utilisé pour réaliser l'application en arrière-plan. C'est-à-dire, cette application peut marcher quand d'autre application est en train d'exécuter comme les services de lecture de musique.

II.2.4 Content Provider :

Content Provider fournit des méthodes pour stocker ou rendre de données. Il permet partager des données entre les applications via les fichiers, ou une base de données de SQLite.

II.3 Cycle de vie d'une activité (ActivityLife cycle) :

Pour développer d'une application sur Android, on doit comprendre le cycle de vie d'une activité. Le cycle de vie d'une activité est exprimé par la figure suivant (Figure Le cycle de vie d'une activité).

- L'état **Active/courant** (Running): C'est un état que l'activité marche en avant-plan. Dans ce cas, cette activité est active ou dans la course.
- L'état **Paused**(il est en pause) : Cette activité est visible mais elle n'est pas active.
- L'état **Stopped**: Cette activité n'est pas visible**.** Si une activité est complètement masquée par une autre activité, elle est arrêtée et conserve tous les états. Cependant elle n'est plus visible pour l'utilisateur, sa fenêtre est cachée et elle sera souvent tuée par le système lorsque la mémoire est nécessaire ailleurs.
- L'état **Dead** : Cette activité est terminée ou elle n'a jamais été démarrée. Si une activité est en pause ou arrêtée, le système peut supprimer l'activité de la mémoire, soit par lui demandant de se terminer, ou tout simplement tuer le processus. Quand il est affiché de nouveau à l'utilisateur, il doit être redémarré et restauré à son état antérieur.

Il existe trois boucles principales:

- La durée de vie d'une activité se passe entre le premier appel à OnCreate () par l'appel à onDestroy (). Une activité met en place tous les états globaux dans la méthode onCreate () et libère toutes les ressources restantes à onDestroy ().
- La durée de vie visible d'une activité se passe entre un appel à onStart () jusqu'à ce qu'un appel correspondant à onStop (). Dans ce temps, l'utilisateur peut voir l'activité sur l'écran, même si elle n'est pas à l'avant et à l'interaction avec l'utilisateur. Entre ces deux méthodes, les ressources qui sont nécessaires pour montrer l'activité de l'utilisateur sont conservées.
- La durée de vie d'une activité en avant-plan se passe entre un appel à onResume () jusqu'à ce qu'un appel correspondant à onPause (). Dans ce temps, l'activité est en

face de toutes les autres activités afin d'interagir avec l'utilisateur. Une activité peut souvent changer son état entre l'état de reprise et l'état en pause.

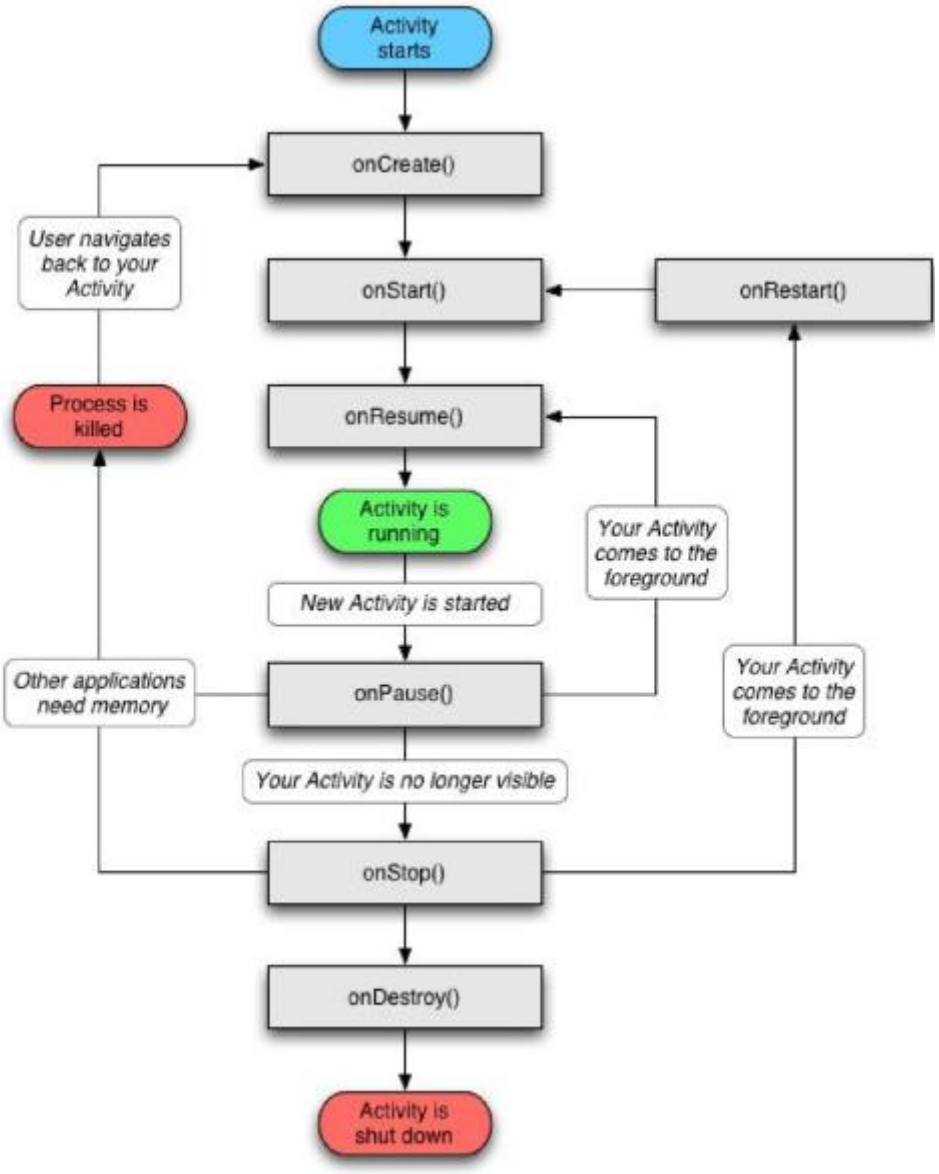

Fig. 2.1 : Cycle de vie d'une activité [13]

II.4 Création de AVD :

Afin de tester notre application, nous allons utiliser l'émulateur Android. Il faudra donc créer un Android Virtual Device (AVD). Un AVD décrit les paramètres systèmes et les composants de notre émulateur.

Pour créer un AVD:

1. Nous lançons Eclipse

2. Nous allons sous « Window> Android SDK and AVD Manager »

3. Nous sélectionnons « Virtual Device » dans le panneau à gauche

4. Nous cliquons sur « New ». La boite de dialogue « Create New AVD » apparaîtra

5. Nous tapons le nom de notre AVD, « hello_avd » par exemple

6. Nous choisissons la cible (the target). La cible est la version de la plateforme Android SDK que nous avons téléchargé.

7. Nous ignorons les autres champs pour le moment et nous cliquons sur « Create AVD »

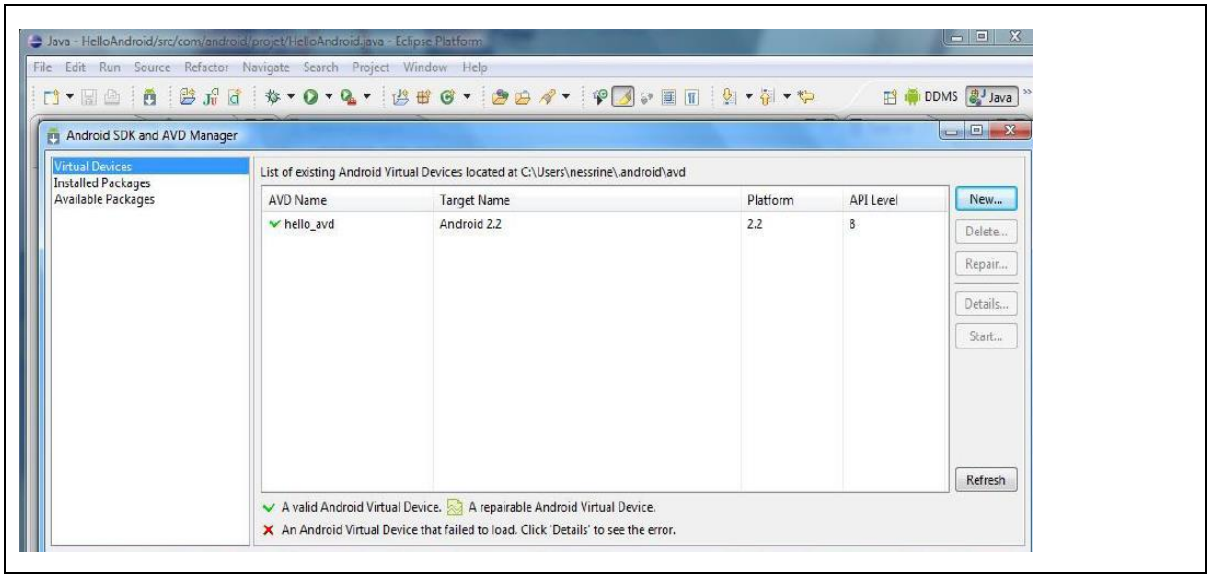

Fig. 2.2 : émulateur de l'Android:

Après avoir créé un émulateur Android, nous passons à la création du projet sous Eclipse.

II.5 Projet HelloWord :

Nous lançons Eclipse et nous allons sous **File -> New->Project** et nous sélectionnons « **Android Project** ».

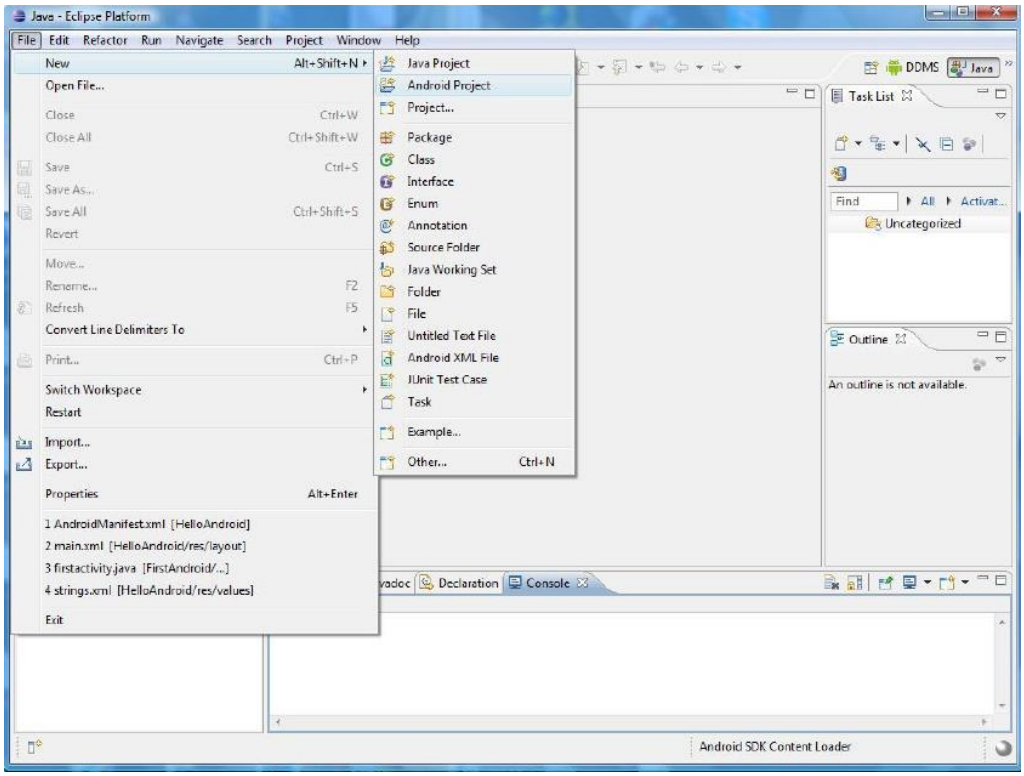

Fig. 2.3 : New project android

Nous renseignons les détails à propos de notre projet comme suit :

Project name: HelloAndroid.

Build Target: Android 2.2.

Application name: Hello, Android.

Package name: com.android.projet.

Create Activity:HelloAndroid.

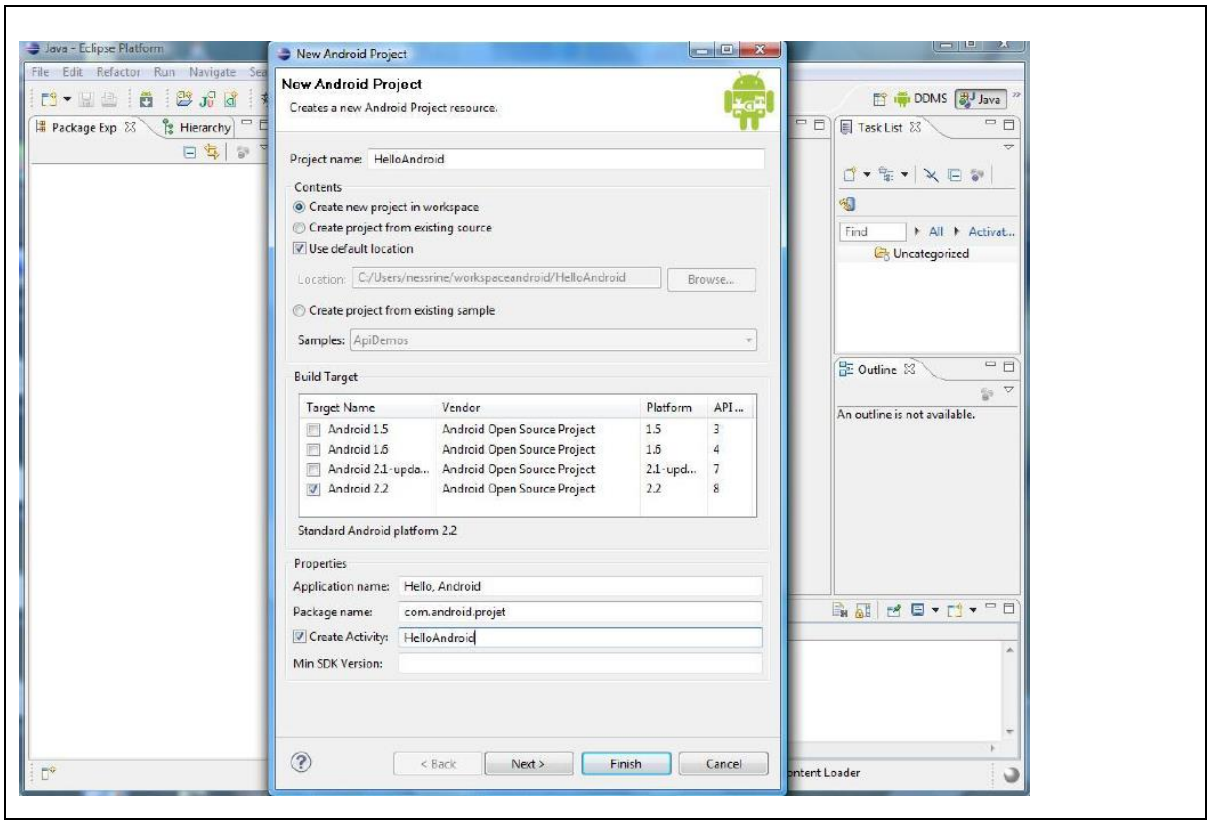

Fig. 2.4: Nom du projet

Nous cliquons sur **Finish.**

II.5.1 Explication des paramètres du projet :

Project name: C'est le nom du projet Eclipse. Tous les fichiers seront créés sous un dossier portant le même nom.

Application Name: C'est le nom de l'application tel qu'il va apparaître sur le smartphone Android.

Package Name: C'est le package namespace (suivant les mêmes règles de programmation Java) qui regroupera tout le code source qu'on va écrire. D'une manière générale, le nom du package doit être unique. Dans notre exemple, on a utilisé com.android.projet.

CreateActivity: C'est le nom du stub class qui va être généré par le plugin. Elle va être une sous-classe de la classe Activity d'Android.

II.5.2 Explication du code:

Notre projet est maintenant prêt. Examinons le code en navigant dans le Package Explorer à gauche. Nous ouvrons le fichier HelloAndroid.java situé sous HelloAndroid->src->com.android.projet qui devra ressembler à ça :

```
package com.android.projet;
import android.app.Activity;
import android.os.Bundle;

public class HelloAndroid extends Activity {

/** Called when the activity is first created. */

@Override

public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
        }
  }
```

Fig. 2.5 : Code java lors de la création

Nous notons que cette classe est basée sur la classe Activity que nous avons mentionnée précédemment. Une Activity est une entité de l'application permettant d'exécuter des actions. Une application peut avoir plusieurs Activités, mais l'utilisateur interagit avec elles une à une.

La méthode **onCreate()** sera appelée par le système Android lors du démarrage de l'application. C'est donc l'endroit idéal pour faire toutes les initialisations et préparer l'interface utilisateur. Cependant, il n'est pas obligatoire d'avoir une interface utilisateur pour chaque Activity.

Codage et Exécution du code HelloWord

Nous modifions le code comme suite

```
package com.android.projet;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloAndroid extends Activity {

/** Called when the activity is first created. */

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);
TextViewtv = new TextView(this);
tv.setText("Hello Word!");
setContentView(tv);
```

```
}
```

Fig. 2.6 : Code java avec quelques modifications

Une Interface Utilisateur Android est composée d'une hiérarchie d'objets appelés Views (Vues). Une View est un objet à dessiner, utilisé comme un élément de l'interface utilisateur. Cela peut être un bouton, une image ou tout simplement du texte comme dans notre cas. Chacun de ces objets est une sous-classe de la classe View. Et la sous-classe qui prend en charge le texte est TextView.

Nous venons de créer un TextView avec le constructeur de classe qui prend comme paramètre une instance Context Android. Un Context fournit des services comme l'accès aux ressources, l'obtention d'un accès à la base de données, etc… La classe Activity hérite du Context et comme :

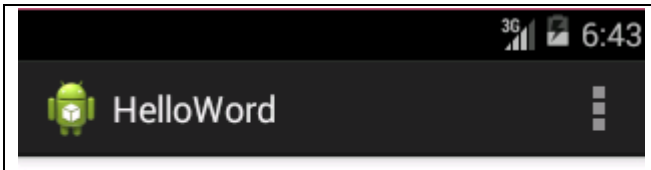

Fig. 2.7 : l'exécution du HelloWord

Chapitre 3

Conception et Implémentation de notre Application

III. 1 Objectifs :

En se basant sur notre problématique, cette application à pour objective

Améliorer notre vocabulaire linguistique et d'aider les gens d'être en mesure des formuler des mots dans un laps de temps et de tester leur niveaux de la connaissance vocabulaire en langue française en cumulant des points sur les bases de descentes aléatoires des lettres en formant des mots des longueurs différentes.

III.2 Fonction du Jeux :

Notre application fonctionne de façon suivante

Il se base sur les descentes aléatoires des lettres les joueurs, donc doivent former des mots dans un temps limité.

Dans le premier niveau de jeux, les mots doivent être de n'importe quelle longueur supérieur ou égale à 2 et les scores sont la somme de longueur de chaque mot trouve dans un laps du temps. Il aura 10 coups à jouer donc si le joueur a réussi trouver 5 ou plus de réponses correctes après les 10 coups il passera pour niveau suivant.

On a trois niveaux différents, niveau 1 dont nous venons de parler puis niveaux 2 et 3

Pour niveau 2 la longueur de mot doit être impérativement de longueur supérieur égale à 4 et pour les niveaux 3 la longueur de mot doit être de longueur supérieur à 6.

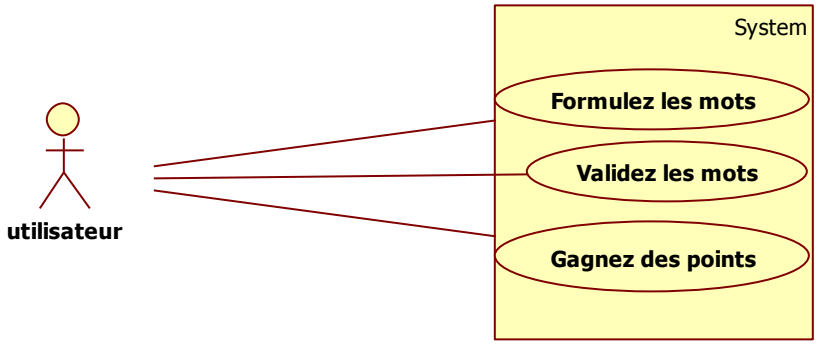

Fig. 3.1 : Cas d'utilisation

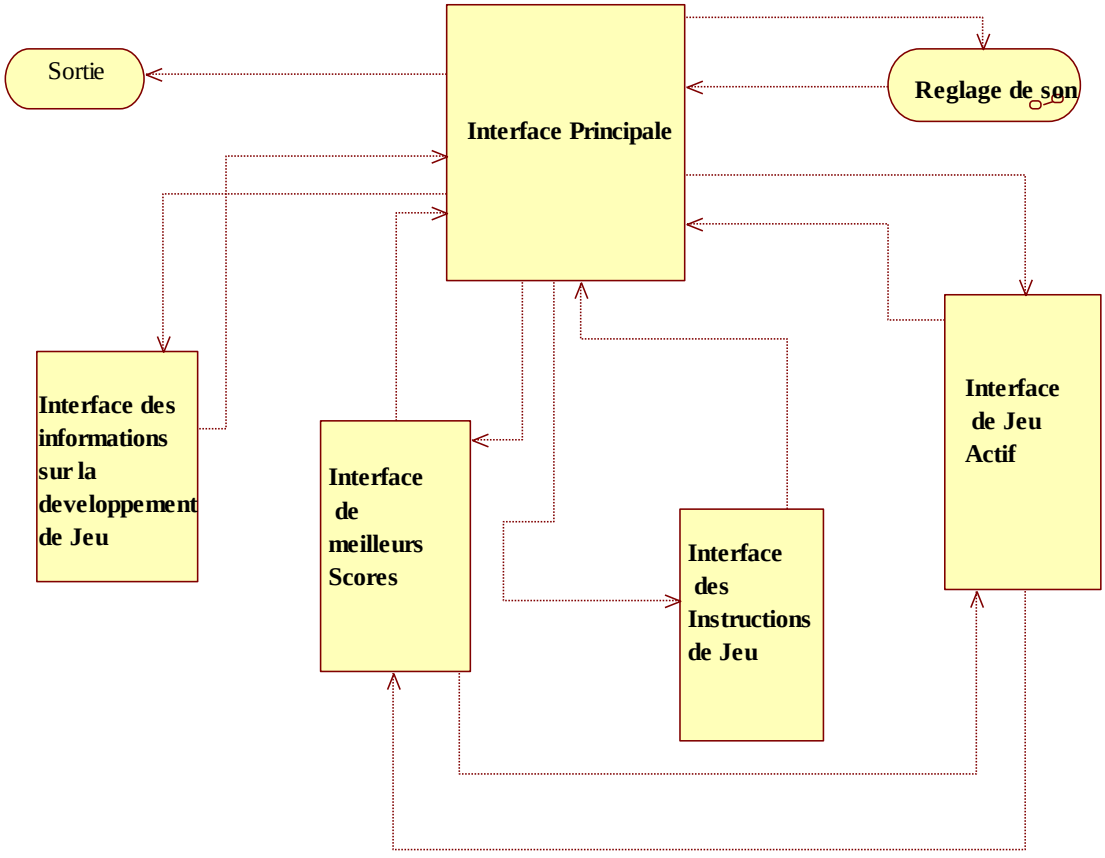

Fig. 3.2 : Illustration des Interfaces constituant notre projet

III.3 La Réalisation de Jeu :

A : Les Outils Utilisée :

- L'environnement de développement Eclipse sur Windows.
- Le SDK de l'Android
- ADT
- UML
- Adobe Photoshop
- L'émulateur - Avec des spécifications :

  Device: 3.2'' HVGA slider (ADP1) (320 × 480: mdpi) –

  Target: Android 4.4 – API Level 19

  RAM 768

  Internal Storage capacity and SD Card capacity: 5 GB

- Un dictionnaire des mots en format .txt .
- Les Images en format .png
- Sons et Musiques

B : Les Limitations :

- Notre Jeu est compatible seulement avec les smartphones qui peuvent supporter l'Android versions entre API 15 et API 19 (notre API cible).
- Notre source de vocabulaire est limite à seulement 111281 mots de longueur compris entre 2 et 10.
- On n'avait que l'émulateur pour tester le fonctionnement de notre application.

C : Les Définitions des classes :

Pour réaliser et manipule toutes les interfaces de notre jeu, on a besoin principalement de deux types des fichiers, ‹‹Java classes et les fichiers XML››.

Les fichiers XML sont charge seulement pour faire l'interface graphique, ‹‹GUI››. Mais c'est aussi possible de définir le GUI avec le codage en Java. S'il on utilise le fichier XML, on peut référencer les composants dans les fichiers .XML par la référence ‹‹@id/nom_du_component››. Les ressources de GUI se trouver dans deux répertoires principalement : ‹‹res/layout›› et ‹‹res/values››. Voici un exemple de GUI définit par un fichier XML :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bp"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="300dp"
        android:layout_height="350dp"
        android:gravity="center"
        android:text="Chargement..."
        android:textColor="#000000" />

</LinearLayout>
```

Fig. 3.3 :L'Interface définit par XML

- L'Interface Principale :

Notre écran principale sert à offrir à l'utilisateur les liens des toutes les options pour aller vers les activités différents.

Nous avons défini notre interface pour l'écran principal à travers l'Activité Layout XML fichier qu'on a nommé ‹‹activity_main.xml››. On a introduit des objets, les Buttons, TextViews, Layouts dont les actions seront manipulées par le code de la Java class associé à notre écran principal. Nous avons aussi utilisé quelques ressources pour améliorer la vue de cet écran par le référencement de leur adresse de l'emplacement. Ceci nous a permis d'implémenter l'image de background, et aussi les boutons créés en utilisant Photoshop.

Fig. 3.4 Capture de l'interface principale

La java class associée à notre écran principal s'appelle Game.java. C'est ici où on a défini les actions des objets utilisés par activity_main.xml. Ici on a implémenté les éléments

‹‹Intents›› qui nous ont permis d'accéder vers toutes les autres classes reliées à notre interface principale. Voici une partie de code java qui nous permet d'aller vers la java class Help1.java en utilisant un intent:

```
final Button b2 = (Button) findViewById(R.id.button2);
b2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent activityChangeIntent = new Intent(Game.this, Help1.class);
        Game.this.startActivity(activityChangeIntent);
    }
});
```

Fig. 3.5 : Implémentation d'un Intent

- L'Interface de Jeu (Jouer)

Notre interface de jeu actif a été défini avec du code java. Nous avons défini tous les attributs dans la java class GameView.java qui est reliée à l'Android class ‹‹View›› et implémenté aussi les méthodes d'interface Java, ‹‹Runnable››.

La classe View nous permet alors à manipuler les ressources stockées dans les dossiers. Il suffit seulement de bien spécifier le chemin vers nos ressources pour pouvoir l'implémenter sur l'interface. Voilà comment nous avons mis en place notre arrière-plan pour cet écran :

```
super.setBackgroundResource(R.raw.bp);
```

Fig. 3.6

On a pu afficher le score, chrono, le mot en cours de construction, les buttons sur notre interface de jeu actif en utilisant la fonction draw(), qui est prédéfini dans la libraire android.graphics.*.

Génération des lettres

Les lettres sont des variables objets du type Block qui est dérivée de la classe Rectangle. Les lettres sont générées aléatoirement dans la classe Block.java et elles sont affectées à chaque variable objet de type Blocks. Chaque variable objet de type Block est construite à partir de la class Rectangle des objets avec les forme carré.

Voici un diagramme des classes constituant les générations de lettres :

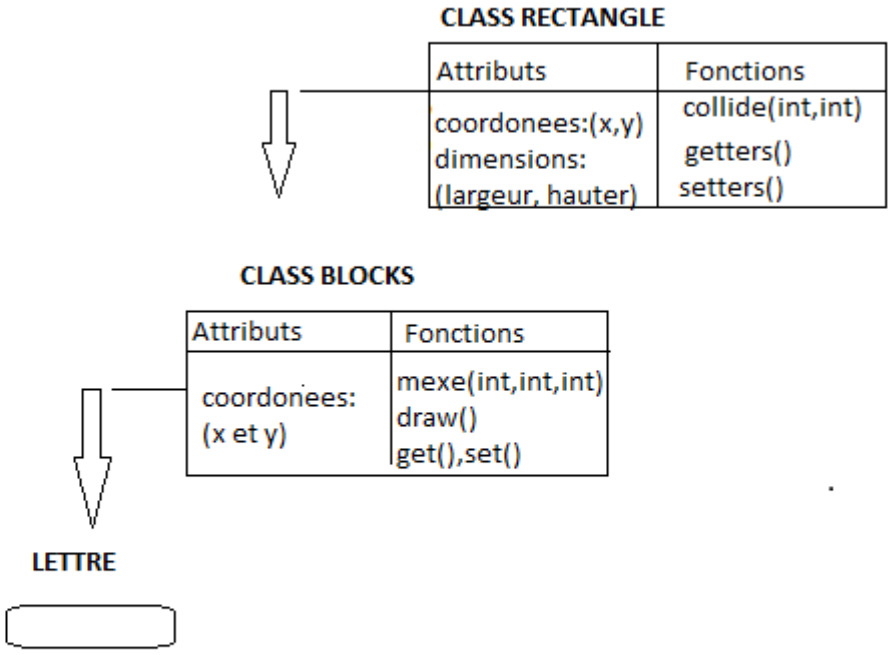

Fig. 3.7 : Génération des lettres

Avec l'implémentation d'une boucle, on a réussir à afficher les lettres sur l'interface grâce à la même fonction que avant, draw().

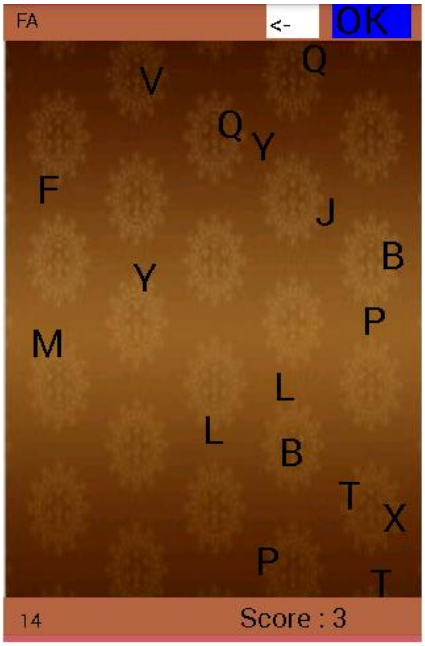

Fig. 3.8 : Capture de l'Interface de jeu (actif) :

- L'Interface d'Aide

Cet écran est mis en place par le fichier ‹‹help1.xml ›› qui déclare tous les objets.

La class Java associée à ce fichier XML, c'est-à-dire celui qui est chargée de la manipulation des objets s'appelle Help1.java.

- L'Interface de Scores

Il est définit par le fichier highscores.xml et associer au code java manipulant de la classe HighScores.java.

- L'Interface d'A propos

Il est défini par le fichier about.xml et manipuler par la code java de la classe About.java.

D : Les Algorithmes, Fonctions et leur implémentation :

Pour rendre les ‹‹Intents››, qui sont chargés du mouvement d'un écran vers un autre (plutôt d'une classe vers une autre classe) fonctionnels il suffit de les déclarer dans un fichier appelé AndroidManifest.xml. Cet ici on référence notre classe cible et donc l'action est effectuée ici.

```
<activity
    android:name="com.game.Game"
    android:configChanges="keyboard|keyboardHidden|orientation"
    android:label="@string/app_name"
    android:screenOrientation="portrait" >

    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Fig. 3.9 : Illustration de la déclaration d'une action d'un intent dans un fichier manifest

Les Fonctions

Pour augmenter l'efficacité de recherche d'existence d'un mot dans notre fichier de mots, nous avons chargé tous les mots du fichier dans une table au début du jeu, en exécutant la fonction charge dictionnaire ()

```
public static void chargerdictionnaire() throws IOException {

    InputStream f = StartScreen.res.openRawResource(R.raw.dico);
    // res est une Ressource definit dans la classe StartScreen
    int i;
    k1 = -1;
    String ip = "";
    int y = 0;
    while ((i = f.read()) != -1) {

        if (!islettre((char) i)) {
            if (y != 0)
                dico[k1] = ip;
            y = 0;
            ip = "";
        } else {
            ip += (char) i;
            y++;
        }
        if (y == 1) {
            k1++;
        }
    }
}
```

Fig. 3.10 : Fonction de chargements des mots

La recherche de l'existence du mot dans le table se fait grâce à une fonction booléenne Search().

```
public static boolean Search(String n) throws IOException {
    // verifie l'existance d'un mot n

    int i = 0;
    while (i < dico.length - 1) {
        if (dico[i].contentEquals(n))
            return true;
        i++;
    }
    return false;

}
```

Fig. 3.11 : Fonction de recherche

Cette fonction sera appelée dans un Thread diffèrent du Thread Main. Ceci va nous permettre de faire la recherche de mot pendant qu'on construit déjà un nouveau mot. Ce Thread peut aussi faire la mise à jour du Score si la fonction Search () retourne Vrai.

Pour faire la mise à jour de Chrono, nous avons introduit un autre nouveau thread. L'implémentation des threads nous permets de faire plusieurs actions et mises à jours sans affecter la vitesse ou l'exécution de notre ‹‹Thread Main››.

E : Les Libraires des Programmation utilisée :

Pour développer notre Jeu, nous avons utilisé les libraires d'Android et les libraires de Java. Voici les libraires :

-**android.content** : Il fournit des classes pour accéder et traiter des données sous Android comme : android.content.ContentProvider, android.content.Intent, etc

-**android.app**: C'est une ensemble de classes d'encapsulation d'Android en haut niveau comme : android.app.Activity, etc.

-**android.view**: Il fournit des classes qui exposent l'interface utilisateur de base et gèrent l'interaction avec l'utilisateur comme : android.view.Menu, android.view.View, etc.

-**android.widget**: Il contient des éléments d'UI comme le bouton-poussoir (android.widget.RadioButton), fileur(spinner-android.widget.spinner)

- **java.io** : Il fournit des fonctions du système d'entrée et de sortie et du système de fichiers : java.io.InputStream, etc.

-**Java.util** : Nous avons utilisé avec les classes de collection : java.util.List.

Conclusion

En vue de croissance de Smartphone au niveau mondial ces derniers années, la téléphonie mobile est sans doute le secteur le plus dynamique, le plus rentable et le plus innovant de toute l'industrie de télécommunication que fait partie de nos quotidiens ; et que plupart de ces Smartphone ont comme système exploitation l'Android. Nous étions menées à explorer ce nouveau système d'exploitation pour mobiles et à faire une application des jeux dans le cadre de notre Project Applications Smartphone Programmation sur Android.

Ce projet est donc composée de 3 parties, dans le premier nous avons fait une introduction de programmation sur Smartphone, puis nous avons parlé de la création d'un simple projet Android, HelloWord. Finalement nous avons décrit notre application et ses fonctionnalités.

Perspectives

Pour des raisons liées principalement à des insuffisances de temps, notre application est terminée avec une version 1.1 et nous envisageons de l'améliorer dans l'avenir proche en ajoutant les connectivités entres les appareils soit pour Bluetooth ou par wifi, aussi donner des possibilités aux joueur d'ajouter des effets sonores ainsi que donner le choix à l'utilisateur de changer les thèmes du background mais aussi rendre ce jeux multi-langue, et pour finir rajouter les définitions de chaque mot trouvé.

Références

[1]Wikipedia, http://en.wikipedia.org/wiki/Smartphone

[2]Framingham, Mass, Android and iOS Combine for 92.3% of All Smartphone Operating System Shipments in the First Quarter While Windows Phone Leapfrogs BlackBerry, According to IDC, 16/05/2013, http://idc.com/pressrelease.html

[3]SearchMobileComputing.com, Mobile operating systems: Which mobile device platform fits your strategy?, http://searchmobilecomputing.techtarget.com/generic/0,295582,sid40_gci1196452,00.html
[4]http://fr.wikipedia.org/wiki/Android

[5]Wikipedia, http://en.wikipedia.org/wiki/Android_version_history

[6]D. Bornstein, Android - Dalvik VM Internals, Google Inc., http://sites.google.com/site/io/dalvik-vm-internals

[7]Android Developper Site

http://developer.android.com/guide/components/fundamentals.html]

[8] Wikipedia,http://iOS.com/Wikipedia/the.free.encyclopedia.html

[9]Wikipedia, http://Apple.Inc.com/Wikipedia/the.free.encyclopedia.html

[10]Kevin McNeish,iOS App development for everyone, 2011

[11]iOS App Programming Guide,Apple.Inc, 2013

[12]Microsoft, Windows Phone 7 Guide for iPhone Application Developers, 3/25/11, rev 3

[13] librairie.immateriel.fr/fr/read_book/9782815002028/developpez-pourandroid_split_001