Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

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

Développement d'une application mobile sous Android (Jeu éducatif : Des Chiffres et Des Lettres)

Réalisé par :

- **KAZI AOUEL Abdessamad & IMINE Youcef**
- **HEDDI Hichem & KADDOUR Sidi-Mohammed**

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

- *M.BNAZZOUZ MOURTADA* *(Encadreur)*
- *M.SMAHI Ismail* *(Examinateur)*
- *M.MESSABIHI Mohammed* *(Examinateur)*

Année universitaire : 2013-2014

Remerciements

Nous tenons tout d'abord à remercier M.BENAZOUZ Mourtada, notre encadreur de nous

avoir proposé ce sujet, de nous avoir aidé et encadré tout au long du projet.

Nous remercions aussi tout le personnel du département d'informatique ayant assisté à la

présentation du projet, et de nous avoir encourages.

Nous remercions particulièrement chaque étudiant du groupe, d'avoir participé et

contribué à atteindre notre objectif.

TABLE DES MATIERES

INTRODUCTION

Dans le cadre d'une initiation au développement mobile en générale, et la programmation sous Android en particulier, et en appliquant les connaissances requises durant notre cursus de licence comme la conception, réalisation, modélisation et bien sûr programmation, nous avons créé notre première application Android.

De nos jours les gens sont très attachés à leurs Smartphone, en utilisant plusieurs types d'applications (communication, divertissement, enseignement, multimédia et vidéo, photographie, cuisine,…) et également plusieurs types de jeux (action, arcade, sport, carte, course, réflexion, simulation, éducatif,…) mais le problème est que la majorité des jeux sont destinés aux enfants ayant moins de 10 ans…

Application Android ? Quel genre d'application ?

Il y a plusieurs idées, plusieurs domaines, différents publics à atteindre, pour cela nous avons choisi un jeu éducatif simple et célèbre, un jeu très populaire que son étoile brille pendant des années, et oui c'est le fameux jeu des chiffres et des lettres diffuser sur la chaine TV France 3.

Alors nous avons pensé à créer notre première application sous le système Android, qui sera la version mobile de ce jeu.

Fans de ce jeu ? Alors le voilà entre vos mains, vous pouvez y jouer à n'importe quel moment. Cette application et pour vous.

CHAPITRE1 : DEVELOPPEMENT MOBILE

I. Introduction :

Une application mobile est un logiciel applicatif développé pour être installé sur un appareil électronique mobile, tel qu'un assistant personnel, un téléphone portable , un « smartphone », ou un baladeur numérique.

Une telle application peut être installée sur l'appareil dès la conception de celui-ci ou bien, si l'appareil le permet, téléchargée par l'utilisateur par le biais d'une boutique en ligne, telle que Google Play, l'App Store ou encore le Windows Phone Store. Une partie des applications disponibles sont gratuites tandis que d'autres sont payantes. [1]

Les applications mobiles ont été initialement proposées pour la productivité et la récupération d'information, comprenant courrier électronique, calendrier électronique, contacts, marché boursier et informations météorologiques. Toutefois, la demande du public et la disponibilité d'outils de développement ont conduit à une expansion rapide dans d'autres domaines, comme les jeux mobiles, les automatismes industriels, les GPS et services basés sur la localisation, les opérations bancaires, les suivis des commandes, l'achat de billets, ou encore des applications médicales mobiles.

Le nombre de téléchargements d'applications mobiles est actuellement en forte hausse. Cette tendance va de pair avec la vente des smartphones, elle aussi en forte augmentation : + 74 % en un an. [1]

Il existe plusieurs systèmes d'exploitation mobiles (OS) dont les plus répandus sont les suivants :

- Android (Google) qui anime un grand nombre de smartphones tels que Samsung, HTC, LG, Motorola…
- iOS (Apple) utilisé sur iPhone et iPad.
- Blackberry OS.
- Windows Phone (Microsoft).
- Symbian (Nokia).
- Bada (Samsung).

Bien sûr il y a une très grande concurrence entre ces derniers…

II. Un peu de statistiques …

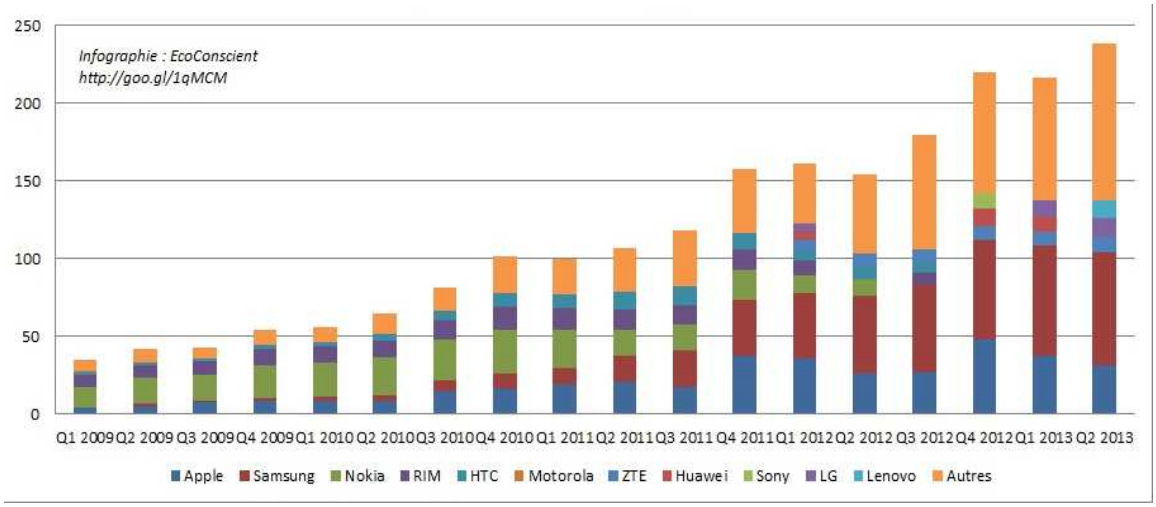


Figure 1 : Evolution des ventes de Smartphone dans le monde en millions d'unités (Infographie EcoConscient : http://goo.gl/1qMCM)

III. Résultats statistiques :

Les taux de réussites sont globalement bons pour les 3 OS mobiles, les utilisateurs arrivent donc sans trop de difficultés à leurs fins.

Le taux de réussite des néophytes est plus faible que pour les utilisateurs réguliers : la prise en main n'est pas immédiate.

Tous OS confondus, les utilisateurs réguliers mettent 40% de temps en moins que les utilisateurs néophytes à exécuter une tâche. Cela montre qu'il existe une courbe d'apprentissage importante.

Sur **Windows Phone** les utilisateurs réussissent davantage à effectuer les tâches et avec le nombre d'erreurs le plus faible, ce système propose la plus grande facilité d'usage pour les néophytes.

La prise en main par les néophytes est plutôt bonne pour iOS.

Les utilisateurs font plus d'erreurs sur iOS que sur les autres OS.

Android est l'OS le plus difficile à utiliser pour les néophytes avec un taux de réussite de (seulement) 42% et beaucoup d'abandons de tâches. [1]

IV. Analyse des expertes :

Windows Phone propose une interface très homogène qui facilite la prise en main pour les néophytes.

iOS et Android faillissent par manque d'homogénéité.

Il y a davantage de **flexibilité** (raccourcis, personnalisation) sur Android et Windows Phone que sur iOS.

Android propose beaucoup de feedbacks (messages de confirmation…) aux utilisateurs qui sont informés en temps réel de la prise en compte de leurs actions.

Alors qu'Android et Windows Phone proposent des gestuelles variées, **iOS a moins développé les interactions tactiles**.

La taille des éléments des interfaces des 3 OS est satisfaisante pour une utilisation tactile confortable.

Le guidage est globalement faible sur les 3 OS mobiles.

iOS propose les meilleurs formulaires alors qu'ils sont fastidieux à remplir sur Windows Phone et source d'erreurs.

Les utilisateurs sont parfois perdus sur Android car il y a trop d'informations affichées à l'écran.

Tous OS confondus, **les néophytes ont des difficultés à trouver les paramètres** du téléphone : sonnerie, luminosité, wifi…

Android permet d'accéder plus **rapidement** aux paramètres du téléphone.

A chaque OS mobile correspond une technologie et un « store » où les applications peuvent être téléchargées, de manière gratuite ou payante :

- App Store pour les applications iPhone et iPad (500 000 apps),
- Play Sotre (300 000 apps),

- Blackberry App World (40 000 apps),
- Marketplace de Windows (30 000 apps),
- Samsung Apps Store (12 000 apps) et l'Ovi de Nokia (85 000 apps). [1]

V. Point de vue des utilisateurs :

- Android, arrivé en 2008, a choisi une autre direction pour son interface. L'interface Android est visuellement chargée et apporte une latitude plus importante à l'utilisateur pour personnaliser son interface comme par exemple la possibilité de remplacer les applications natives. Cependant, l'OS Android reste difficile à prendre en main. Nos tests ont montré que les utilisateurs néophytes sont parfois perdus et abandonnent leur tâche.

- iOS d'Apple est le premier OS pour téléphones tactiles qui a véritablement lancé la vague des smartphones. Apparu sur le marché en 2007, il a innové dans un domaine qui n'était pas encore développé. Cependant, iOS n'a pas beaucoup évolué depuis sa création.
iOS se caractérise par une interface peu chargée, qui permet l'exécution de tâches rapidement pour les utilisateurs réguliers, mais qui laisse peu de place à la personnalisation. Dans de nombreux cas observés, le positionnement de mêmes éléments diffère d'une application à l'autre. Notre étude montre que les utilisateurs sont fréquemment ralentis par ce manque d'homogénéité qui entraîne un grand nombre d'erreurs.

- Windows Phone, challenger du marché apparu en 2010, a su s'inspirer de ses concurrents et de son expérience passée dans le domaine mobile tout en innovant. Proposant une interface simple et épurée ainsi que de multiples possibilités de personnalisation, il permet d'arriver rapidement à l'exécution d'une tâche ce qui a particulièrement séduit les utilisateurs. C'est d'ailleurs Windows Phone qui se trouve être l'OS mobile le plus adapté aux utilisateurs néophytes. A noter que certaines fonctionnalités demeurent absentes et que les formulaires de saisie actuels posent de réelles difficultés d'usage. [2]

CHAPITRE 2 : PRESENTATION D'ANDROID ET SON ENVIRONNEMENT DE DEVELOPPEMENT

I. Présentation d'Android :

Une combinaison de trois composants :

- Un système d'exploitation open source pour terminaux mobiles.
- Une plateforme de développement open source pour créer des applications mobiles.
- Terminaux, particulièrement téléphones mobiles, qui exécutent le système d'exploitation Android et les applications mobiles conçues pour ce système.

II. Présentation d'Android en détail :

- Un système d'exploitation basé sur le noyau Linux qui fournit l'interface bas niveau avec le HW, la gestion de la mémoire, le contrôle des processus, le tout optimisé pour les terminaux mobiles.
- Un ensemble de bibliothèques Open Source pour le développement d'applications incluant SQLite , WebKit , OpenGL , et la gestion des média.
- Android, incluant la VM Dalvik et les librairies principales qui fournissent ces fonctionnalités.
- C'est aussi un framework applicatif exposant les services systèmes à la couche application y compris " window manager, location manager, content providers, telephony, sensors ".
- Un framework d'interface utilisateur pour installer et lancer les applications. Un software développement kit utilisé pour créer des applications, incluant des outils, plugins et documentation.
- Un software développement kit utilisé pour créer des applications, incluant des outils, plugins et documentation

III. Architecture de l'OS Android :

Android se base sur un noyau Linux 2.6. Le SDK Android possède une bibliothèque de librairie, de plusieurs classes java de base pour plusieurs types d'application (exemple : OpenGL pour la 3D, SSL pour les protocoles de sécurité, etc.).

Une application Android se repose sur un Framework qui facilite l'utilisation des classes de base et sert d'interface entre les "Librairies" et les applications.

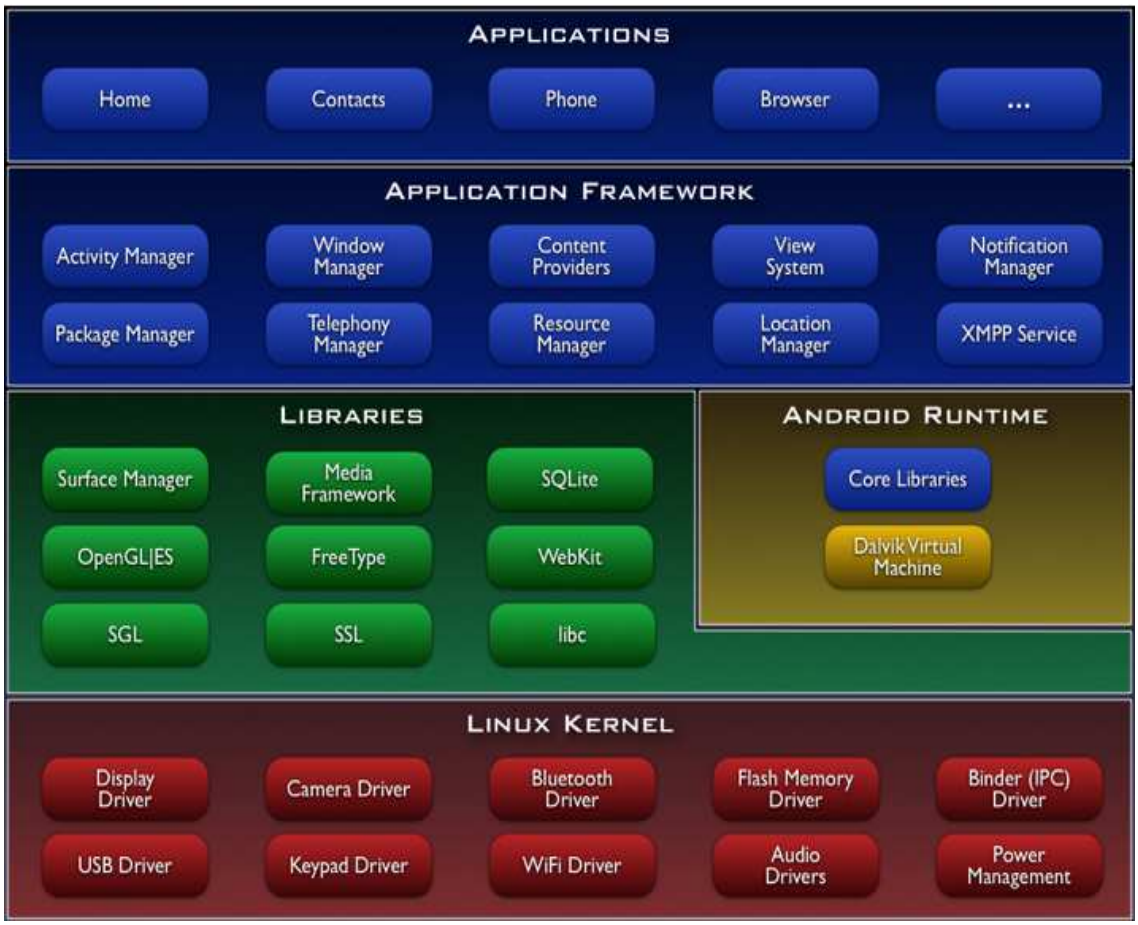


Figure 2 : Architecture de la plateforme Android

- **La couche Linux Kernel :**

Figure 3 : La couche Linux Kernel

Android se base sur le noyau Linux 2.6 pour ses services système de base tels que la sécurité, la gestion de la mémoire, la gestion des processus, le modèle de pilote. Ce qui permet une meilleure gestion des caractéristiques des appareils mobiles.

Le noyau Linux offre un système de gestion de mémoire et de processus stable et performant avec un modèle de sécurité robuste.

- **La couche Android Runtime :**

Figure 4 : La couche Android Runtime

Un ensemble de bibliothèques de base inclut par le système d'exploitation Android.

Chaque application Android s'exécute dans son propre processus avec sa propre instance dans la machine virtuelle « Dalvik ». Cette dernière a été faite de sorte qu'un dispositif peut fonctionner plusieurs tâches simultanément de manière efficace.

- **La couche Libraries :**

Figure 5 : La couche Libraires

Android inclut un ensemble des librairies utilisées telles que : OpenGL ES, SQ Life, Web Kit, Media Framework. Ces bibliothèques sont accessibles à travers la couche application d'Android.

- **La couche Application Framework :**

Figure 6 : La couche Application Framework

Elle Contient les applications de base de l'Os Android et permet aux développeurs une utilisation complète de ces APIs pour développer des applications avancées.

- **La couche Application :**

Figure 7 : La couche Application

Android est livré avec un ensemble d'applications de base, dont un client de messagerie, un programme des SMS, un calendrier, un navigateur, la liste des contacts.

Toutes les applications sont écrites en utilisant le langage de programmation Java.

IV. Le SDK Android :

Le Software Developement Kit Android est un ensemble de fichiers d'aide et d'exemples. On y trouve aussi des utilitaires pour la mise au point et les tests.

- **Les fonctions du SDK :**

  - Accès au Hardware, y compris Camera, GPS, et Accéléromètre.
  - Base de données SQLite.
  - Données et dépôt de données partagées et communication inter application par échange de messages.
  - Ecran d'accueil riche par l'utilisation des Widgets, Live Folders, and Live Wallpaper Support Média très riche et graphiques 2D/3D : Rendus graphiques par HW optimisé pour la mobilité, incluant une librairie " path - based " pour les rendus 2D et le support pour les graphiques 3D utilisant OpenGL ES 2.0

- Accès au HW Wifi et libraries pour l'utilisation du Bluetooth pour le transfert peer - to-peer.
- Technologies réseau GSM, EDGE, et 3G pour la téléphonie ou le transfert de données, permettant de placer des appels téléphoniques, des SMS, et d'envoyer et de recevoir des données en utilisant les réseaux de données mobiles.
- API pour l'utilisation des capteurs HW y compris les accéléromètres et le compas.

V. L'AVD Android :

- Android Virtual Device est un dispositif mobile virtuel qui s'exécute sur l'ordinateur et permet de développer et de tester des applications Android sans l'aide d'un dispositif physique.
- L'émulateur Android imite toutes les fonctionnalités matérielles et logicielles d'un dispositif mobile typique, tel que la lecture des fichiers audio et vidéo, stockage des données, sauf qu'il ne peut pas passer des appels réels.

Figure 8 : Exemple d'émulateur (Genymotion)

VI. Elément de base dans une application Android :

- Classes Activity pour chaque écran
- Fichier layout (en général un fichier pour chaque Activité)
- Intent pour pouvoir enchainer les activités et échanger des variables.
- Fichier Manifest pour toute l'application :
    - Les différentes activités de l'application
    - Les droits de l'application
    - La version …

VII. Contraintes liées au développement sous Android :

Différentes contraintes sont à prendre en compte lors du développement dans cet environnement mobile :

A- Il faut pouvoir interagir avec un système complet sans l'interrompre. Android fait des choses pendant que votre application est utilisée, il reçoit des SMS et des appels, entre autres.

B- . Il faut respecter une certaine priorité dans l'exécution des tâches.

C- Il faudra exploiter tous les outils fournis afin de débusquer les portions de code qui nécessitent des optimisations.

D- La taille de l'écran est réduite, et il existe par ailleurs plusieurs tailles et résolutions différentes.

E- L'interface graphique doit s'adapter à toutes les tailles et toutes les résolutions, ou il aura des risques de laisser de côté un bon nombre d'utilisateurs.

F- Les interfaces tactiles sont peu pratiques en cas d'utilisation avec un stylet et/ou peu précises en cas d'utilisation avec les doigts, d'où des contraintes liées à la programmation événementielle plus rigides.

G- Enfin, en plus d'avoir une variété au niveau de la taille de l'écran, on a aussi une variété au niveau de la langue, des composants matériels présents et des versions d'Android. Il y a une variabilité entre chaque téléphone et même parfois entre certains téléphones identiques c'est un travail en plus à prendre en compte.

VIII. Composants d'une application Android :

Les composants majeurs suivants sont en quelques sortes les briques élémentaires des applications :

A- Activités : qui sont la couche de présentation de votre application

B- Services : les composants qui tournent en arrière-plan.

C- Content providers : Sources de données partageables.

D- Intens : Framework de communication inter-applications.

E- Broadcast receivers : Consommateurs des messages diffusés par les intents.

IX. L'environnement de développement ECLIPSE :

1. Définition :

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la Fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation des logiciels, englobant les activités de programmation (notamment environnement de développement intégré et frameworks).

Son EDI, partie intégrante du projet, vise notamment à supporter tout langage de programmation à l'instar de Microsoft Visual Studio.

Bien qu'Eclipse ait d'abord été conçu uniquement pour produire des environnements de développement, les utilisateurs et contributeurs se sont rapidement mis à réutiliser ses briques logicielles pour des applications clientes classiques. Cela a conduit à une extension du périmètre initial d'Eclipse à toute production de logiciel : c'est l'apparition du framework Eclipse RCP en 2004.

Figurant parmi les grandes réussites de l'Open source, Eclipse est devenu un standard du marché des logiciels de développement, intégré par de grands éditeurs logiciels et services.

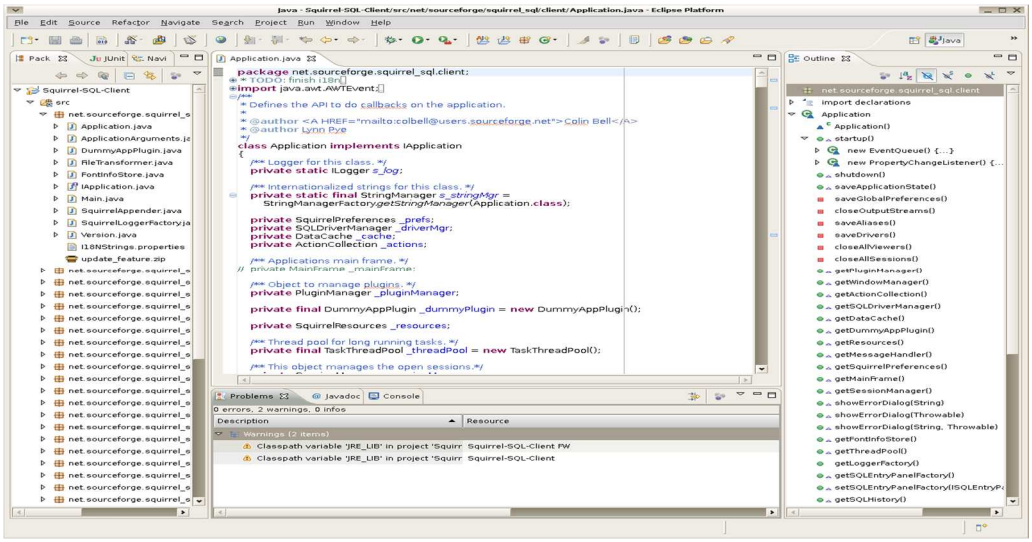


Figure 9 : Capture d'écran de l'IDE eclipse

2. Le plugin de développement d'Eclipse : ADT

Figure 10 : Capture d'écran de l'ADT

Un projet basé sur le plugin ADT(Android Developer Tools) est décomposé de la manière suivante:

•src/: les sources Java du projet

•libs/: bibliothèques tierces

•res/:

•res/drawable: ressources images

•res/layout: description des IHMs en XML

•res/values: chaines de caractères et dimensions

•gen/: les ressources auto générées par ADT

•assets/: ressources brutes (raw bytes)

•bin/:

•bin/classes: les classes compilées en .class

•bin/classes.dex: exécutable pour la JVM Dalvik

•bin/myapp.zip: les ressources de l'application

•bin/myapp.apk: application empaquetée avec ses ressource et prête pour le déploiement

CHAPITRE3 : L'APPLICATION (JEU DES CHIFFRES ET DES LETTRES)

I. Présentation de l'application :

Présentation générale :

Comme déjà précisé, des Chiffres et des Lettres est un jeu éducatif qui a pour but l'amélioration du niveau linguistique des gens, et aussi augmenter les capacités de calculs mentale.

Ce jeu est composé de deux parties :

A- Partie chiffres :

Dans cette partie le joueur aura 6 nombres générés aléatoirement (selon certains critères de difficulté : facile, moyen ou difficile), et un résultat auquel il doit arriver dans un temps personnalisé (60s, 80s, 100s) en utilisant les quatre opérations arithmétiques (+,-, /,*), sachant qu'un nombre ne pourra être utilisé qu'une seule fois, si le joueur peut aboutir au résultat donné il aura le BON COMPTE et un score de 100%, sinon il aura un COMPTE APPROCHANT et un score relative. De plus les meilleures suites d'opérations lui seront affichées.

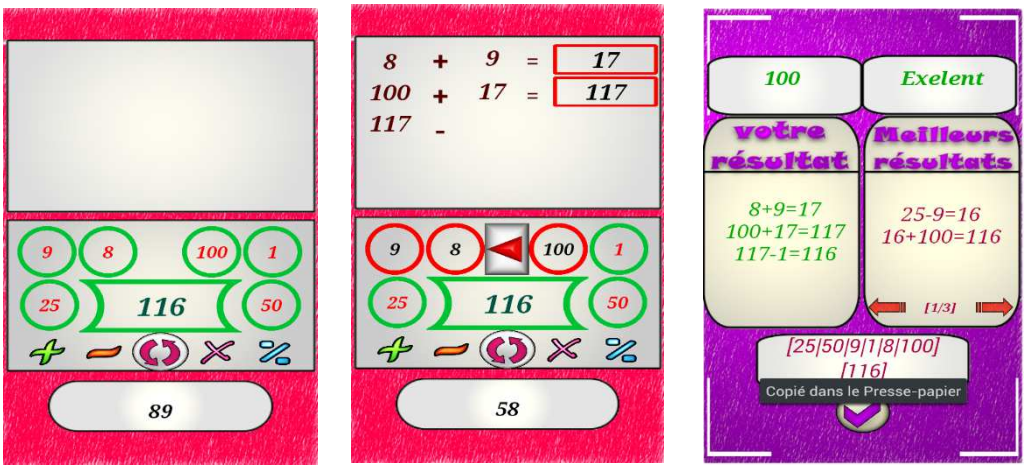


Figure 11 : Capture d'écran d'une partie chiffres

B- Partie lettres :

Dans cette parie le joueur aura le choix de générer soit une consonne ou une voyelle, jusqu'à l'obtention de 9 lettres, ensuite il doit trouver un mot d'une taille minimale de 3 lettres (chaque lettre sera utilisé une seule foi), et plus le mot est long le score est grand. Et il aura aussi une liste de tous les mots pouvant être trouvés avec les 9 lettres générer.

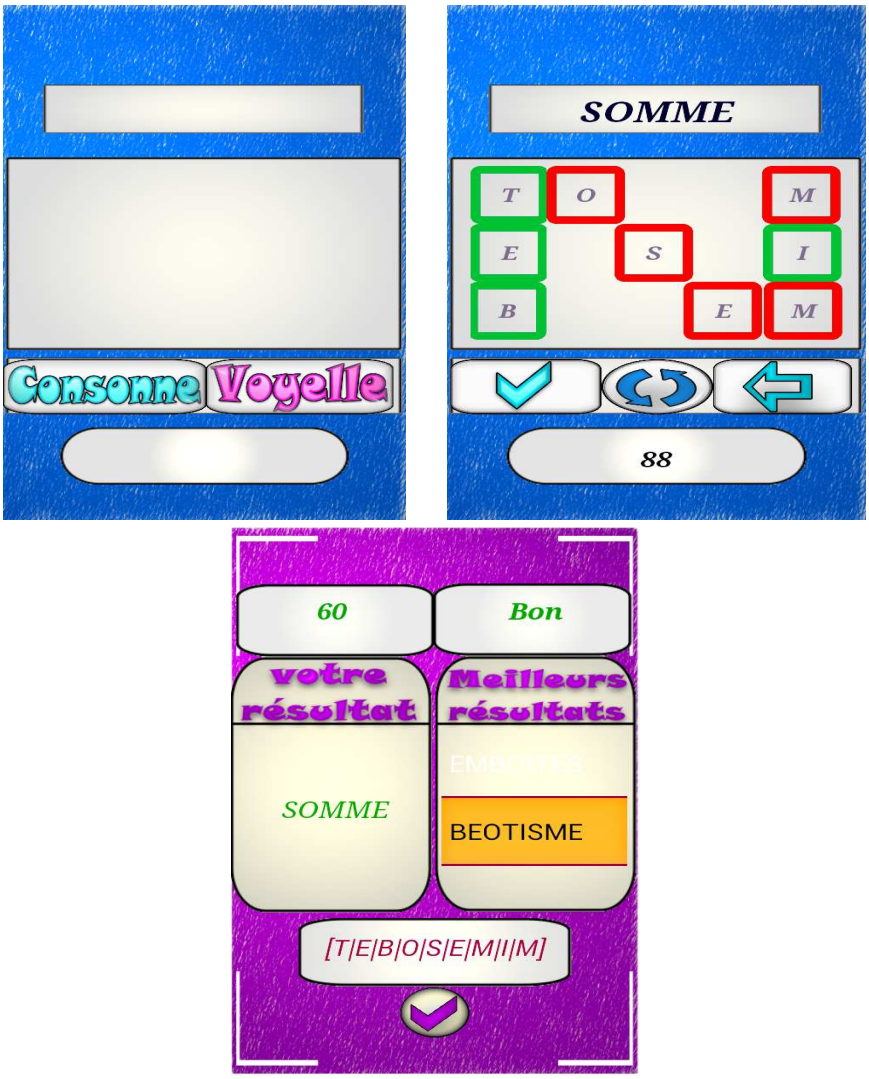


Figure 12 : Capture d'écran d'une partie lettres

1- Présentation des autres détails :

Dans le cadre d'une amélioration du jeu et de l'interface utilisateur, nous avons ajouté une partie MULTIJOUEURS qui permettront à 2 joueurs de ce défié via WIFI.

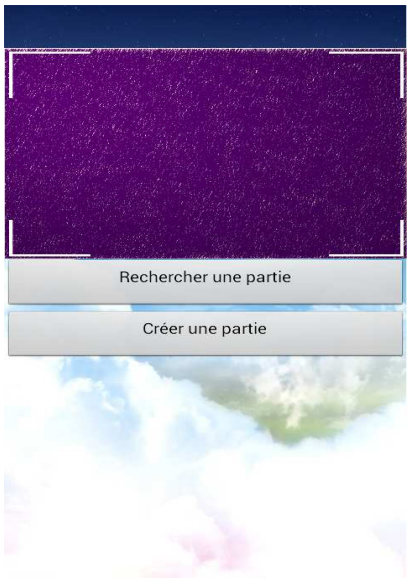


Figure 13 : Capture d'écran de la partie multi-joueurs

Aussi une partie spéciale pour les différents réglages (paramètres) :

- Options du jeu :
  - Difficulté : facile, moyen, difficile
  - Durée du chrono : 100, 80, 60
  - Nombre d'itération : 3, 6, 9
- Options du son :
  - Son du fond
  - Effet sonores
  - Volume
- Meilleurs scores

Figure 14 : Capture d'écran des paramètres

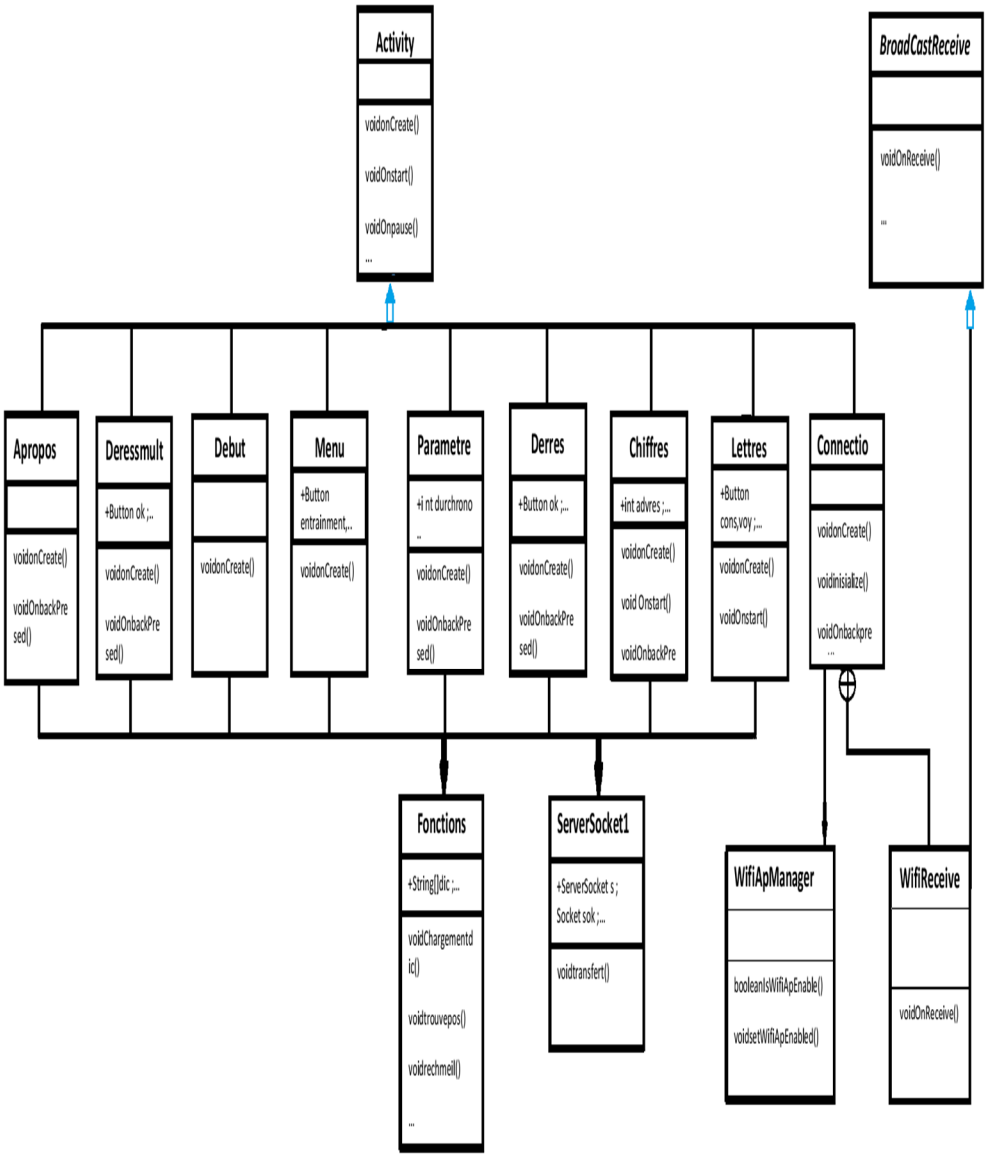
De plus il y a la partie à propos qui contient l'explication du jeu, ainsi qu'une option pour permettre aux utilisateurs de notre application de nous contacter et nous envoyer des feedback via notre adresse gmail(hikk.android@gmail.com).

Figure 15 : Capture d'écran de la partie à propos

II. CONCEPTION ET REALISATION :

1- Diagramme de classes :

La structure générale du projet est décrite par le model UML suivant :



1- Implémentation :

L'application est un jeu éducatif comporte deux parties :

A. La partie Chiffres :

Cette partie du jeu consiste à générer 6 chiffres et un résultat d'une façon aléatoire, puis on donne la main au joueur pour essayer de trouver ou d'approcher le résultat pendant une durée de temps défini dans un chronomètre, lorsque le temps s'achève l'application doit afficher le meilleur résultat trouvé par le joueur, les meilleurs opérations possibles, un score et une émotion.

Pour atteindre ces objectifs, il était nécessaire de créer différentes méthodes :

a- La génération des Chiffres :

int gentab (int tab [], int difficultés) : est une méthode qui génère les 6 éléments entier et un résultat de 3 chiffres aléatoirement.

Nb : la génération dépend du niveau du jeu (facile, normal ou difficile).

```
public static int gentab(int tab[],int difficulte){
        int a[]={1,2,3,4,5,6,7,8,9,10,25,50,75,100};
        switch(difficulte){
        case 3:{
        int k=(int)(Math.random()*5.99);
        for (int ii=0;ii<6;ii++){
                if(ii==k)tab[ii]=a[(int)(4.99*Math.random()+9)];
                else
do{tab[ii]=a[(int)(8.99*Math.random())];}while(((tab[ii]<=9)&&(existdeux(tab,ii,tab[ii])))||((tab[ii]>9)&&(exist(tab,ii,tab[ii]))));
            }
        return (int)(Math.random()*500+500);}
        case 2:{
                 int  k=(int)(Math.random()*5.99);
                int gre =k;
                do{gre=(int)(Math.random()*5.99);}while(k==gre);

                for (int ii=0;ii<6;ii++){
                        if((ii==k)||(ii==gre))do{tab[ii]=a[(int)(4.99*Math.random()+9)];}while(exist(tab,ii,tab[ii]));
                        else
do{tab[ii]=a[(int)(8.99*Math.random())];}while(((tab[ii]<=9)&&(existdeux(tab,ii,tab[ii])))||((tab[ii]>9)&&(exist(tab,ii,tab[ii]))));
                }
                return (int)(Math.random()*900+100);

        }
        case 1:{
                 int  k=(int)(Math.random()*5.99);
                int gre =k;
                do{gre=(int)(Math.random()*5.99);}while(k==gre);
                int gre1=gre;
                do{gre1=(int)(Math.random()*5.99);}while((k==gre1)||(gre==gre1));
                for (int ii=0;ii<6;ii++){
                        if((ii==k)||(ii==gre)||(ii==gre1))do{tab[ii]=a[(int)(4.99*Math.random()+9)];}while(exist(tab,ii,tab[ii]));
                        else
do{tab[ii]=a[(int)(8.99*Math.random())];}while(((tab[ii]<=9)&&(existdeux(tab,ii,tab[ii])))||((tab[ii]>9)&&(exist(tab,ii,tab[ii]))));
                }
                return gennorex(tab);

        }


        }

        return -1;
   }
```

b- Recherche des meilleures suites d'opérations :

Problème : la difficulté qui se posait dans ce point était de créer un algorithme capable de trouver toutes les opérations possibles entre 6 chiffres et en même temps rapide, avec la moindre complexité possible.

Solution adoptée :

La méthode String trouveposib (int l [], int nor, int x []) : le principe de cette méthode est de ranger une combinaison de 6 chiffres dans un tableau, puis faire toutes les opérations possibles entre les deux premiers éléments du tableau, le résultat sera ranger avec le reste des éléments du tableau précédent dans un nouveau tableau de longueur 5, l'opération se répète jusqu'à l'obtention d'un tableau qui contient 3 éléments où on fait toutes les opérations possibles entre ses éléments, la meilleure suite d'opérations sera stockée dans une chaine de caractère.

L'algorithme choisira toujours les suites d'opérations possédant :

1- Le minimum d'opérations.

2- Le résultat le plus proche à ce qu'on doit trouver.

Lors des calculs d'opérations, on a ajouté des conditions pour éviter les divisions non entières ; les résultats nuls et négatifs ; les produit et les divisions avec 1. Ainsi que des conditions pour éviter la répétition des suites d'opération.

Schéma illustratif montrant le déroulement de l'algorithme :

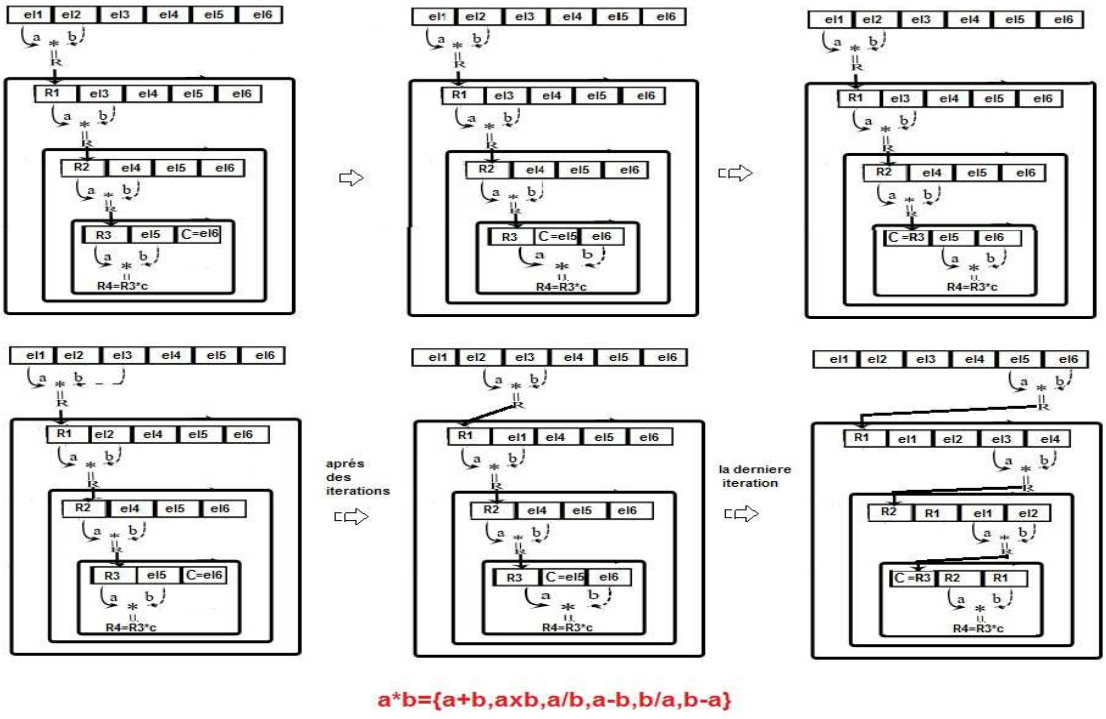


Figure 16 : Schéma illustratif démontrant le déroulement de l'algorithme

B. Partie Lettres :

Dans cette partie le joueur doit choisir 9 lettres, puis il va essayer de former un mot avec ces lettres pendant une durée définie dans un chronomètre.

Les problèmes rencontrés :

- La recherche des meilleurs mots formés à partir d'un ensemble de 9 lettres en utilisant un dictionnaire.
- Préparer la liste des meilleurs mots.
- Vérifier si le mot entré par le joueur existe dans le dictionnaire.

Solutions :

Les problèmes cités ci-dessus nous ont conduits à créer les méthodes suivantes :

- La méthode « chargerdictionnaire () » : le fonctionnement de cette méthode est simple, elle charge les mots qui se trouver dans dictionnaire (fichier) dans un tableau de chaines de caractères.

Cette méthode nous a permet d'éviter les opérations d'entrer/sortie (lecture du fichier dictionnaire) à chaque fois qu'on a besoin de rechercher un mot, et d'exploiter l'accès rapide à la RAM là où le tableau est stocké.

```
public static void chargerdictionnaire() throws IOException {
     //Création de l'objet File

     InputStream f = Debut.resi.openRawResource(R.raw.dico);

     int i;k1=-1;
     String ip = "";
     int y = 0;
      while((i=f.read())!=-1)
      {

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

      }
}
```

- La méthode « élimine () » : cette méthode et comme son nom l'indique élimine tous les éléments du tableau qui contient la liste des mots contenant une lettre ou plus n'appartenant pas à l'ensemble des lettres choisies par le cpu.

Remarque : avant le lancement de la partie lettres le cpu choisit aléatoirement 14 lettres (7 consonnes et 7 voyelles), le joueur va choisir neuf lettres parmi les quatorze choisis par le cpu.

```
public static void elemine(String pass){
    int k=0;
    for(int i=0;i<111280;i++)if(content1(dic[i],pass)){dicel[k]=dic[i];/*dictrietmp[k]=dictrie[i];*/k++;}
    g=k;
}
```

- La méthode « rechmeil2 (char m []) » : cette méthode retourne les meilleurs résultats qu'on peut trouver avec un ensemble de 9 lettres, dans un temps très rapide.

Le principe de cette méthode consiste à utiliser la rapidité du cpu dans le parcours des tableaux où on calcule pour chaque élément du tableau le nombre des lettres appartenant à l'ensemble choisi par le joueur et à l'élément au même temps, tout en sauvegardant les mots contenant le nombre maximal des lettres.

```
public static String Rechmeil2(char m[]) throws IOException {//retourner la meilleure resultat
     int h=g-1;
     int meilleur = 0;
     String meil_mot="";
     int longeur=9;
     int nbr_mot=0;
     while(h>=0){

         String s=dicel[h];

         int cpt=0;
          for(int i=0;i<9;i++){
                                                  if(s.contains(String.valueOf(m[i])))
                                                  {
                                                  cpt++;
                                                  s=s.replaceFirst(String.valueOf(m[i]),"1");
                                                  }

                                              }
     if(dicel[h].length()!=longeur)if(nbr_mot!=0)return meil_mot;
                                else longeur=dicel[h].length();
     if(cpt==s.length())if(cpt>=meilleur){meil_mot=meil_mot+dicel[h]+'\n';meilleur=cpt;nbr_mot++;}
     h--;
     }
     return meil_mot;
}
```

- La méthode « Rand (String n) » : cette méthode est responsable de vérifier si le mot formé est correcte.

```
public static boolean Rand(String n) throws IOException{
  //verifie l'existance d'un mot n
        //@SuppressWarnings("deprecation")

            // TODO Auto-generated method stub
            int i=0;
          while(i<g-1){
        if(dicel[i].contentEquals(n)) return true;
                i++;
        }
            return false;

        }
```

On peut aussi citer d'autres méthodes telle que :

gencons () : qui génère une consonne.

```
public static char gencons(){
    //char cons[]={'B','C','D','F','G','H','J','K','L','M','N','P','K','R','S','T','V','W','X','Z'};

    int i= (int)(Math.random()*468890);
    if((i>=0)&&(i<14502)) return 'B';
    if((i>=14502)&&(i<44367)) return 'C';
    if((i>=44367)&&(i<65549)) return 'D';
    if((i>=65549)&&(i<78368)) return 'F';
    if((i>=78368)&&(i<95082)) return 'G';
    if((i>=95082)&&(i<105352)) return 'H';
    if((i>=105352)&&(i<107536)) return 'J';
    if((i>=107536)&&(i<108328)) return 'K';
    if((i>=108328)&&(i<143743)) return 'L';
    if((i>=143743)&&(i<166190)) return 'M';
    if((i>=166190)&&(i<218940)) return 'N';
    if((i>=218940)&&(i<240503)) return 'P';
    if((i>=240503)&&(i<241295)) return 'K';
    if((i>=241295)&&(i<313821)) return 'R';
    if((i>=313821)&&(i<387494)) return 'S';
    if((i>=387494)&&(i<444313)) return 'T';
    if((i>=444313)&&(i<455640)) return 'V';
    if((i>=455640)&&(i<455818)) return 'W';
    if((i>=455818)&&(i<459159)) return 'X';
    if((i>=459159)&&(i<468890)) return 'Z';
    return 'D';
    }
```

- genvoy () : qui génère une voyelle.

```
public static char genvoy(){
   // char voy[]={'A','E','I','O','U','Y'};

   int i=(int)(Math.random()*380615);
   if((i>=0)&&(i<90061)) return 'A';
   if((i>=90061)&&(i<220051)) return 'E';
   if((i>=220051)&&(i<293412)) return 'I';
   if((i>=293412)&&(i<340042)) return 'O';
   if((i>=340042)&&(i<377064)) return 'U';
   if((i>=377064)&&(i<380615)) return 'Y';
   return 'A';
  }
```

Remarque : Ces 2 méthodes génèrent les lettres en se basant sur des statistiques effectuées sur les fréquences d'apparitions des lettres dans le dictionnaire.

C. *PARTIE MULTIJOUEURS*

L'application avait comme but la réalisation du coté multi-joueurs qui donne la possibilité de créer des parties comportant 2 joueurs, en exploitant la technologie WIFI.

Le système Android donne la possibilité de la réalisation de tel but, en permettant la manipulation de cette technologie à travers des services telle que le Wifi Manager.

1- Implémentation :

La partie multi-joueurs a été mise en point à l'aide de plusieurs outils telle que :

a- Le service Wifi Manager : le Wifi Manager est un service fournit par le système Android et qui permet la manipulation du Wifi en fournissant des informations sur son état dans l'appareil (en donnant la possibilité de changer cet état ou même plus en définissant un appareil comme étant un point d'accès Wifi), ce service peut fournir aussi des informations sur n'importe quelle connexion établie sur l'appareille (informations concernant les réseaux disponible).

b- Les Sockets : représentent l'outil principal d'une communication entre les membres d'un réseau ou tout échange de données s'effectue à travers ces objets.

c- BroadCastReceiver : représente un récepteur d'Intent pendant l'exécution d'une application telle que les Intents correspondant à l'échange d'état du Wifi et la réception des listes des réseaux détectés…

2- Réalisation :

Les défis étaient :

- Etablir une connexion entre 2 appareilles.
- Réussir l'échange des informations et la synchronisation entre 2 appareils lors du déroulement de la partie multi-joueurs.

- Etablissement de la connexion :

Pour établir une connexion un des deux joueurs doit transformer son appareil a un point d'accès Wifi, l'autre doit chercher le réseau créé par son adversaire et se joindre.

- Création d'un point d'accès Wifi : Le service Wifi Manager possède des méthodes cachées permettant la création, la gestion et la manipulation des points d'accès Wifi.

La difficulté dans ce point était la récupération puis l'exploitation des méthodes cachées, cela s'est réaliser en définissant des méthodes permettant de telles taches :

- **getHiddenMethods () : qui permet la récupération des méthodes cachées permettant la manipulation du point d'accès Wifi.**

```
private void getHiddenMethods() {
    try {
        setWifiApEnabledMethod = wifiMan.getClass().getMethod("setWifiApEnabled", WifiConfiguration.class, boolean.class);
        isWifiApEnabledMethod = wifiMan.getClass().getMethod("isWifiApEnabled");
    } catch (NoSuchMethodException e) {
        e.printStackTrace();
    }
}
```

- **setWifiApEnabled (Configuration conf, boolean enabled) : qui permet la création et la mise en marche d'un point d'accès wifi en faisant appel à la méthode cache SetWifiApEnabled dans la classe Wifi Manager.**

```
public boolean setWifiApEnabled(WifiConfiguration conf, boolean enabled) {
    try {
        return (Boolean) setWifiApEnabledMethod.invoke(wifiMan, conf, true);
    } catch (Exception e) {
        return false;
    }
}
```

- La connexion à un point d'accès Wifi : Le service Wifi Manager donne aussi la possibilité de lancer un scan afin de trouver les réseaux actifs et même de se connecter.

  Remarque : la récupération de toutes informations détectées par le service Wifi Manager et effectuer à l'aide du BroadCastReceiver.

- **getSystemService (Context c) : permet la récupération de n'importe quelle service disponible sur le système Android.**

```
mainWifi = (WifiManager) getSystemService(Context.WIFI_SERVICE);
```

- **registerReceiver (BroadCastReceiver receiver, IntentFilter i) : permet la registration d'un objet BroadCastReceiver afin d'être capable de récupérer des informations concernant le IntentFilter déclarer si dessous.**

```
receiverWifi = new WifiReceiver();

registerReceiver(receiverWifi, new IntentFilter(
        WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
```

On peut aussi citer d'autres méthodes fournies par le service Wifi Manager telle que :

```
mainWifi.setWifiEnabled(true);
```

Qui permet de changer l'état du Wifi.

```
mainWifi.startScan();
```

Qui permet de lancer un scan des réseaux actifs.

- **enableNetwork (int id, boolean b) : Qui permet d'activer un réseau.**

```
mainWifi.enableNetwork(arg2, false);
```

```
mainWifi.reassociate();
```

Qui permet de réassocier la connexion a un réseau.

```
Chiffres.ip=intToIp(mainWifi.getDhcpInfo().serverAddress);
```

Qui permet d'avoir l'adresse IP du serveur.

- L'échange des données :

L'échange des données s'effectue en utilisant les Sockets, où on distingue deux côtés :

a- Coté serveur : ou on crée un objet Socket Server, et on se met à l'écoute afin de recevoir les données provenant de l'adversaire.

```
Thread a=new Thread(new Runnable() {

    @Override
    public void run() {
        // TODO Auto-generated method stub
        try {
            if(m!=null) m.close();
             m=new ServerSocket(8580);
            sok=m.accept();
            lire();
            Chiffres.ip=sok.getInetAddress().getHostAddress();
            Message msg=hand.obtainMessage();
            msg.obj="b";
            hand.sendMessage(msg);
            serveur=1;
            Chiffres.z=1;
            client=false;
            m.close();
            //sok.close();
           } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }
});     a.start();          //}
```

b- Coté client : ou on crée un objet Socket, et on envoi des données à l'adversaire.

```
try{
sok=new Socket(Chiffres.ip,8580);
os=new ObjectOutputStream(sok.getOutputStream());
//int tab[]=new int[11];
int m;
m=Fonctions.gentab(tab1,Parametres.defficulté);
tab1[6]=m;
String s=tantostr(tab1);
os.flush();
os.reset();
os.writeObject(s);
os.flush();
client=true;
serveur=0;
Message msg=hand.obtainMessage();
msg.obj="b";
//msg.obj=tab1;
hand.sendMessage(msg);
}catch (UnknownHostException e) {
// TODO Auto-generated catch block
e.printStackTrace();
Toast.makeText(getApplicationContext(), "aucun apareille en ecout", Toast.LENGTH_SHORT).show();
```

L'utilisation des Sockets se diffère entre selon les besoins où on trouve que la partie Lettres consiste à faire une alternation de l'objet Server Socket à chaque fois qu'une des deux appareilles veut recevoir une lettre en laissant l'envoie à l'autre (l'objet Socket), cette opération se déroule a tour de rôle jusqu'à avoir 9 lettres. Par contre la partie Chiffres demande qu'un appareil soit à l'écoute et l'autre génère et envoie les chiffres.

- **La sauvegarde des paramètres du jeu :**

L'application donne la possibilité de sauvegarder plusieurs informations concernant la difficulté du jeu, les caractéristiques des parties, le réglage des effets sonores et la sauvegarde des meilleurs scores.

La réalisation de telle possibilité a nécessité la création d'un fichier qui contient les informations définis ci-dessus.

Ce fichier sera créé dès le premier lancement de l'application en exploitant :

openFileOutput (nom fichier, mode d'ouverture) : cette méthode permet d'ouvrir un fichier en écriture si il existe déjà et dans le cas contraire, elle crée un fichier dans la mémoire interne.

Le système Android fournit plusieurs modes d'ouverture : MODE_PRIVATE, MODE_APPEND, MODE_WORLD_READABLE, MODE_MULTI_PROCESS, MODE_MULTI_PROCESS.

OpenFileInput (nom fichier) : cette méthode permet d'ouvrir un fichier en lecture.

- **Comment crée le fichier paramètres une seule fois ?**

La méthode openFileInput (nom fichier) génère une exception(FileNotFoundExeption) dans le cas où le fichier n'existe pas dans la mémoire interne et logiquement cette exception va se déclencher lors du premier lancement de l'application, La création du fichier va se dérouler exactement à ce moment en exploitant la méthode openFileOutput (nom fichier, mode d'ouverture) cite ci-dessus.

```
try {
    in=openFileInput(fich);
Fonctions.chargparam();
Fonctions.chargtabsco();
} catch (FileNotFoundException e1) {
    // TODO Auto-generated catch block
    try {
        out=openFileOutput(fich, MODE_PRIVATE);
        String nom="1\ntrue\ntrue\n100\n100\n3\nscore
        out.write(nom.getBytes());
    }catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

- **Le chargement, modification et sauvegarde des paramètres :**

Les différentes opérations effectuées sur le fichier paramètres se réalisent en exploitant les méthodes suivantes :

La méthode « chargparam () » : cette méthode lit le fichier paramètre situé dans la mémoire interne et stocke le contenu dans une chaine de caractères, ensuite elle affecte chaque paramètre a sa variable correspondante.

```
public static void chargparam(){
        int value;
        String lu="";
        try {
            while((value = Debut.in.read()) != -1) {
                // On écrit dans le fichier le caractère lu
                lu=lu+(char)value;
              }
        Debut.tab=lu.split("\n");
        Parametres.defficulté=toint(Debut.tab[0]);
            if(Debut.tab[1].contentEquals("true"))Parametres.son=true;
            else Parametres.son=false;
            if(Debut.tab[2].contentEquals("true"))Parametres.effect=true;
            else Parametres.effect=false;
            Parametres.volume=toint(Debut.tab[3]);
            Parametres.dur_chr=toint(Debut.tab[4])+1;
            Parametres.nbr_iterations=toint(Debut.tab[5]);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

- La methode « chargtabsco() » : après le chargement des parametres cette methode affecte les meilleurs scores et les noms des joueurs correspondants dans deux tableaux(un pour les noms et l'autre pour les scores correspondants).

```
public static void chargtabsco(){
    for(int i=0;i<5;i++){Parametres.nom[i]="";Parametres.score[i]=0;}
    for(int i=7;i<12;i++){
        for(int j=0,ind=0;j<Debut.tab[i].length();j++){
            if(Debut.tab[i].charAt(j)!=':'&& ind==0) Parametres.nom[i-7]=Parametres.nom[i-7]+Debut.tab[i].charAt(j);
            else {ind=1;
            int val=0;
            switch (Debut.tab[i].charAt(j)){
            case '0':val=0;break;
            case '1':val=1;break;
            case '2':val=2;break;
            case '3':val=3;break;
            case '4':val=4;break;
            case '5':val=5;break;
            case '6':val=6;break;
            case '7':val=7;break;
            case '8':val=8;break;
            case '9':val=9;break;
            default:val=0;break;

            }
            Parametres.score[i-7]=Parametres.score[i-7]*10+val;}
        }
    }
}
```

- La methode « modif() » :permet la modification des parametres.

```
public static void modif() throws IOException{
     Debut.tab[0]=String.valueOf(Parametres.defficulté);
     Debut.tab[1]=String.valueOf(Parametres.son);
     Debut.tab[2]=String.valueOf(Parametres.effect);
     Debut.tab[3]=String.valueOf(Parametres.volume);
     Debut.tab[4]=String.valueOf(Parametres.dur_chr-1);
     Debut.tab[5]=String.valueOf(Parametres.nbr_iterations);
     Debut.tab[6]="score:";
     text="";
     for(int i=0;i<12;i++){
    if(i>=7) {Debut.tab[i]=Parametres.nom[i-7]+":"+String.valueOf(Parametres.score[i-7]);}
    text=text+Debut.tab[i]+"\n";
     }

     }
```

Apres l'appel de la methode modif(), la sauvegarde des parametres s'effectue en utilisant la methode write(byte []b) située dans la classe OutputStream.

- **L'interface** :

Une interface n'est pas une image statique mais un ensemble de composants graphiques qui peuvent être des boutons, du texte ou des groupements d'autres composants graphiques.

Android offre aux développeurs la possibilité de créer une interface avec deux manières différentes :

- Programmatique : dans le code d'une activité en faisant appel aux classes correspondantes (Button, TextView, Layout,..).

- Déclarative : en utilisant le langage de structuration XML.

Dans notre application les interfaces sont créées par la méthode déclarative vu la facilité de manipulation des composants graphiques.

➢ **Gestion d'interfaces :**

Durant une partie, l'application demande plusieurs interactions tel que : l'affichage des données, l'état des buttons (cliquable ou pas), l'état du chrono (visible ou pas)… etc.

Du coup il fallait tout synchroniser en utilisant des objets tels que les Threads et les Handlers, de plus de nouvelles méthodes qu'on a implémentées.

- La méthode affsux() : qui gère les effets sonores et l'affichage successif des chiffres ou des lettres .

```
final Handler affsux= new Handler(){
        @Override
        public void handleMessage(Message msg1) {
            if(cpt1<6)
            {n[cpt1].setVisibility(View.VISIBLE);n[cpt1].setClickable(false);n[cpt1].setText(String.valueOf(tab[cpt1]));}

            else{for(int i=0;i<6;i++)n[i].setClickable(true);
                                nor.setVisibility(View.VISIBLE);
                                    if(cpt1==12)m=m1;
                                    else m=Fonctions.gennor();
                                    nor.setText(String.valueOf(m));
                                    cpt1++;
                                        lesnums.setText("["+tab[0]+'|'+tab[1]+'|'+tab[2]+'|'+tab[3]+'|'+tab[4]+'|'+tab[5]+"]\n["+m+']');}

            if(cpt1<6){
            switch(tab[cpt1]){
            case 1:fx=MediaPlayer.create(Chiffres.this,R.raw.un);if(Parametres.son)fx.start();break;
            case 2:fx=MediaPlayer.create(Chiffres.this,R.raw.deux);if(Parametres.son)fx.start();break;
            case 3:fx=MediaPlayer.create(Chiffres.this,R.raw.trois);if(Parametres.son)fx.start();break;
            case 4:fx=MediaPlayer.create(Chiffres.this,R.raw.quatre);if(Parametres.son)fx.start();break;
            case 5:fx=MediaPlayer.create(Chiffres.this,R.raw.cinq);if(Parametres.son)fx.start();break;
            case 6:fx=MediaPlayer.create(Chiffres.this,R.raw.six);if(Parametres.son)fx.start();break;
            case 7:fx=MediaPlayer.create(Chiffres.this,R.raw.sept);if(Parametres.son)fx.start();break;
            case 8:fx=MediaPlayer.create(Chiffres.this,R.raw.huit);if(Parametres.son)fx.start();break;
            case 9:fx=MediaPlayer.create(Chiffres.this,R.raw.neuf);if(Parametres.son)fx.start();break;
            case 10:fx=MediaPlayer.create(Chiffres.this,R.raw.dix);if(Parametres.son)fx.start();break;
            case 25:fx=MediaPlayer.create(Chiffres.this,R.raw.vingtecinq);if(Parametres.son)fx.start();break;
            case 50:fx=MediaPlayer.create(Chiffres.this,R.raw.cinquant);if(Parametres.son)fx.start();break;
            case 75:fx=MediaPlayer.create(Chiffres.this,R.raw.soicentquanz);if(Parametres.son)fx.start();break;
            case 100:fx=MediaPlayer.create(Chiffres.this,R.raw.cent);if(Parametres.son)fx.start();break;
            }
            cpt1++;}
```

- La méthode « ecrire () » : est la méthode responsable de l'affichage des données lors du déroulement de la partie des chiffres ou des lettres.

```
public void ecrire() {

            if(nbrtxt%4==0){derniere_button_utiliser[nbrtxt/2]=cpt1;
                            t[nbrtxt].setText(n[cpt1].getText());
                            elem1=tab[cpt1];
                            retour.setVisibility(View.VISIBLE);
                            for(int i=0;i<11;i++){n[i].setClickable(false);}
                            add.setClickable(true);mult.setClickable(true);div.setClickable(true);sup.setClickable(true);nbrtxt++;}
            else if((nbrtxt-1)%4==0){t[nbrtxt].setText(num);
                                        add.setClickable(false);
                                        mult.setClickable(false);
                                        div.setClickable(false);
                                        sup.setClickable(false);
                                              for(int i=0;i<11;i++){
                                                    if(n[i].getTextColors()!=compteur.getTextColors())
                                                                                {n[i].setClickable(true);}
                                                                          }nbrtxt++;}
                else if((nbrtxt-2)%4==0){derniere_button_utiliser[(nbrtxt-2)/2+1]=cpt1;t[nbrtxt].setText(n[cpt1].getText());elem2=tab[cpt1];

                int resultat = 0;
                switch(t[nbrtxt-1].getText().charAt(0)){
                case '+':resultat=elem1+elem2;n[(nbrtxt-2)/4+6].setText(String.valueOf(resultat));
                            tab[(nbrtxt-2)/4+6]=resultat;
                            n[(nbrtxt-2)/4+6].setVisibility(View.VISIBLE);
                            t[nbrtxt+1].setVisibility(View.VISIBLE);
                            nbrtxt+=2;break;
                case '-':if(elem1-elem2<0){
                                Toast.makeText(Chiffres.this,"Operation Incorect:\nresultat negative",500).show();
                                fx=MediaPlayer.create(Chiffres.this,R.raw.erreur);
                                if(Parametres.son)fx.start();
                                nbrtxt+=2;retour();
                                break;}
                                                                    else {resultat=elem1-elem2;tab[(nbrtxt-2)/4+6]=resultat;
                                                                    n[(nbrtxt-2)/4+6].setText(String.valueOf(resultat));
                                                                    tab[(nbrtxt-2)/4+6]=resultat;
                                                                    n[(nbrtxt-2)/4+6].setVisibility(View.VISIBLE);
                                                                    t[nbrtxt+1].setVisibility(View.VISIBLE);
                                                                    nbrtxt+=2;}break;
                case 'x':resultat=elem1*elem2;n[(nbrtxt-2)/4+6].setText(String.valueOf(resultat));
                tab[(nbrtxt-2)/4+6]=resultat;
                n[(nbrtxt-2)/4+6].setVisibility(View.VISIBLE);
                t[nbrtxt+1].setVisibility(View.VISIBLE);
                nbrtxt+=2;break;
                case '/':if(elem1%elem2!=0){Toast.makeText(Chiffres.this,"Operation Incorrect:\nresultat non entier ",500).show();
fx=MediaPlayer.create(Chiffres.this,R.raw.erreur);
if(Parametres.son)fx.start();
nbrtxt+=2;retour();
break;}
                                                                    else {resultat=elem1/elem2;tab[(nbrtxt-2)/4+6]=resultat;
n[(nbrtxt-2)/4+6].setText(String.valueOf(resultat));
```

- L'implémentation de ces méthodes n'était pas suffisante il a fallu utiliser en parallèle d'autres objets tel que les Threads et les Handler afin de synchroniser et détecter toute interaction sur l'interface et la traiter directement.

```
threadafsux = new Thread(new Runnable() {
    public void run() {
    try {
        Thread.sleep(1000);
    for (int i = 0; enexec && i < 6; ++i) {

    Message msg1 = affsux.obtainMessage();
    affsux.sendMessage(msg1);
    Thread.sleep(1500);


    }
    for(int i=0;enexec && i<7;i++){
        Message msg1 = affsux.obtainMessage();
        affsux.sendMessage(msg1);
        Thread.sleep(250);
    }
    } catch (Throwable t) { }
    }
    });
enexec=true;
threadafsux.start();
```

```
threadchrono= new Thread(new Runnable() {
      public void run() {
      try {

          threadafsux.join();
      for (int i = 0; enexec && i <=Parametres.dur_chr+1 ; ++i) {
      // Mise en pause d'une seconde
      Thread.sleep(1000);
      // Création d'un message vide
      Message msg = chrono.obtainMessage();
      // Envoi du message au handler
      chrono.sendMessage(msg);
      }
      } catch (Throwable t) { }
      }
      });
      enexec = true;
      // Lancement du thread d'arrière-plan
      threadchrono.start();
```

```
final Handler affsux= new Handler(){
@Override
public void handleMessage(Message msg1) {
if(cpt1<6){
n[cpt1].setVisibility(View.VISIBLE);n[cpt1].setClickable(false);n[cpt1].setText(St
ring.valueOf(tab[cpt1]));
}
else{
      for(int i=0;i<6;i++)n[i].setClickable(true);
       nor.setVisibility(View.VISIBLE);
       if(cpt1==12)m=m1;
       else m=Fonctions.gennor();
       nor.setText(String.valueOf(m));
       cpt1++;
       lesnums.setText("["+tab[0]+'|'+tab[1]+'|'+tab[2]+'|'+tab[3]+'|'+tab[4]+'|'
        +tab[5]+"]\n["+m+']');}
```

CONCLUSION :

Après plusieurs mois d'efforts, notre application a été publiée sur deux stores algériens, il s'agit de « condor store » de condor, et de « Ostore » d'Ooredoo.

Ainsi les utilisateurs pourront nous contacter via notre adresse Gmail : hikk.android@gmail.com

La réalisation de ce projet nous a permis de s'organisé, d'apprendre à travailler en groupe, d'apprendre un nouveau langage programmation (Java), prendre des idées sur le fonctionnement d'un réseau, apprendre le langage de structuration XML et l'utilisation de langage de modélisation UML…

Comme tout projet évolutif et qui nécessite continuellement des améliorations, nous proposons des perspectives comme :

- Publier l'application sur Play store.
- Ajouter une option permettant de jouer en ligne en gardant le score sauvegardé à l'aide d'un serveur.
- Améliorer les parties Multi-joueurs en augmentant le nombre d'adversaires. (plus que 2)
- Créer une application similaire en Arabe et en Anglais.
- Créer d'autres application sous Android, surtout pour enrichir le contenue « MADE IN ALGERIA » dans ce domaine.
- Approfondir encore plus nos connaissances dans le domaine de développement mobile, notamment avec les autres systèmes (Android, Windows phone, IOS…).

REFERENCES ET SITES WEB :

- Damien Guignard, Julien Chable, Emmanuel Robles, Avec la contribution de Nicolas Sorel et Vanessa Conchodon : Programmation Android De la conception au déploiement avec le SDK Google Android 2, EYROLLES (www.editions-eyrolles.com).
- Mouhammed DIOP, Présentation-Android, software developper, mouhammed.diop@ucad.edu.sn
- [1]http://www.lesmobilizers.com/developpement-application-mobile.html
- [2]http://www.ergonomie-interface.com/mobile-tactile-nomade/test-comparatif systemes-ios-android-windows-phone/
- http://fr.wikipedia.org/wiki/Syst%C3%A8me_d'exploitation_mobile
- http://android.developpez.com/cours/?page=Debutant#Debut
- http://www.tontonfred.net/blog/?p=2534
- http://www.ergonomie-interface.com/mobile-tactile-nomade/test-comparatif-systemes-ios-android-windows-phone/
- http://fr.wikipedia.org/wiki/Eclipse_(projet)
- www.objis.com – formation Android
- https://www.youtube.com/channel/UCJbPGzawDH1njbqV-D5HqKw (Android Application Developement Tutorials)

LISTE DES FIGURES :