Soutenance Développement Mobile : Magic Tiles

Kamarouzamane Combo & Hajanirina Randimbisoa

M1 Informatique Université de la Réunion

12 novembre 2018

Sommaire

1. Introduction
2. Présentation du jeu
3. Android
4. iOS
5. Démonstration
6. Conclusion

Introduction

Objectif

- Créer une application mobile sur Android et iOS

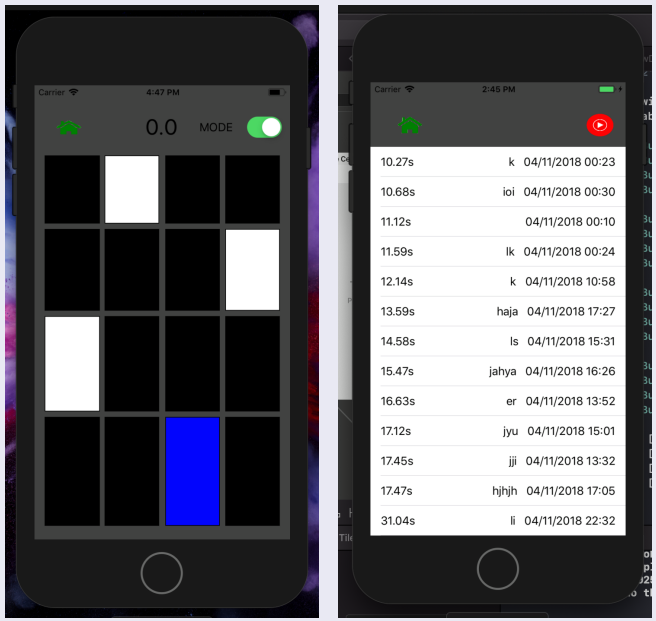
Contraintes

- Changement de configuration
- Géolocalisation
- Utiliser un capteur
- Utilisation d' au moins un geste courant non-trivial
- Ajouter du son

Présentation du jeu

Principe

- Le jeu Magic Tiles est un jeu de modélisation et de simulation des touches de piano



Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.
- *ResultatNeg* : Elle s'affiche en cas de défaite, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer ou d'accéder à l'écran d'accueil.

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.
- *ResultatNeg* : Elle s'affiche en cas de défaite, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer ou d'accéder à l'écran d'accueil.
- *ResultatPos* : Elle s'affiche en cas de victoire, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer, d'enregistrer son score ou d'accéder à l'écran d'accueil. on récupère les données transmises par l'activité Game dans l'intent et on les retransmet ensuite pour l'activité Enregistrer si on appuie sur ce bouton.

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.
- *ResultatNeg* : Elle s'affiche en cas de défaite, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer ou d'accéder à l'écran d'accueil.
- *ResultatPos* : Elle s'affiche en cas de victoire, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer, d'enregistrer son score ou d'accéder à l'écran d'accueil. on récupère les données transmises par l'activité Game dans l'intent et on les retransmet ensuite pour l'activité Enregistrer si on appuie sur ce bouton.
- *Enregistrer* : Dans cette activité, on dispose de trois boutons, l'un pour valider son enregistrement, un autre pour prendre une photo et un autre pour annuler (pour revenir au menu). Le joueur a la possibilité d'écrire son nom, mais ne doit pas dépasser une quinzaine de caractères, et doit en entrer au moins trois. Les données seront également transmises à l'activité ListeScores.

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.
- *ResultatNeg* : Elle s'affiche en cas de défaite, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer ou d'accéder à l'écran d'accueil.
- *ResultatPos* : Elle s'affiche en cas de victoire, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer, d'enregistrer son score ou d'accéder à l'écran d'accueil. on récupère les données transmises par l'activité Game dans l'intent et on les retransmet ensuite pour l'activité Enregistrer si on appuie sur ce bouton.
- *Enregistrer* : Dans cette activité, on dispose de trois boutons, l'un pour valider son enregistrement, un autre pour prendre une photo et un autre pour annuler (pour revenir au menu). Le joueur a la possibilité d'écrire son nom, mais ne doit pas dépasser une quinzaine de caractères, et doit en entrer au moins trois. Les données seront également transmises à l'activité ListeScores.
- *ListeScore* : Dans cette activité, nous allons créer les scores, les afficher et les enregistrer dans une base de données (via les classes Score, ScoresAdapter et MyDBAdapter)

Les Activités

Il y a 7 *activités* :

- *Accueil* : L'écran d'accueil, apparaît au lancement de l'application.
- *Game* : Elle permet de lancer le jeu.
- *ResultatNeg* : Elle s'affiche en cas de défaite, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer ou d'accéder à l'écran d'accueil.
- *ResultatPos* : Elle s'affiche en cas de victoire, on affche le meilleur score s'il y en un. On donne ensuite la possibilité au joueur de rejouer, d'enregistrer son score ou d'accéder à l'écran d'accueil. on récupère les données transmises par l'activité Game dans l'intent et on les retransmet ensuite pour l'activité Enregistrer si on appuie sur ce bouton.
- *Enregistrer* : Dans cette activité, on dispose de trois boutons, l'un pour valider son enregistrement, un autre pour prendre une photo et un autre pour annuler (pour revenir au menu). Le joueur a la possibilité d'écrire son nom, mais ne doit pas dépasser une quinzaine de caractères, et doit en entrer au moins trois. Les données seront également transmises à l'activité ListeScores.
- *ListeScore* : Dans cette activité, nous allons créer les scores, les afficher et les enregistrer dans une base de données (via les classes Score, ScoresAdapter et MyDBAdapter)
- *Maps* : Cette activité permet à l'application de récupérer la position du joueur et l'afficher sur une carte.

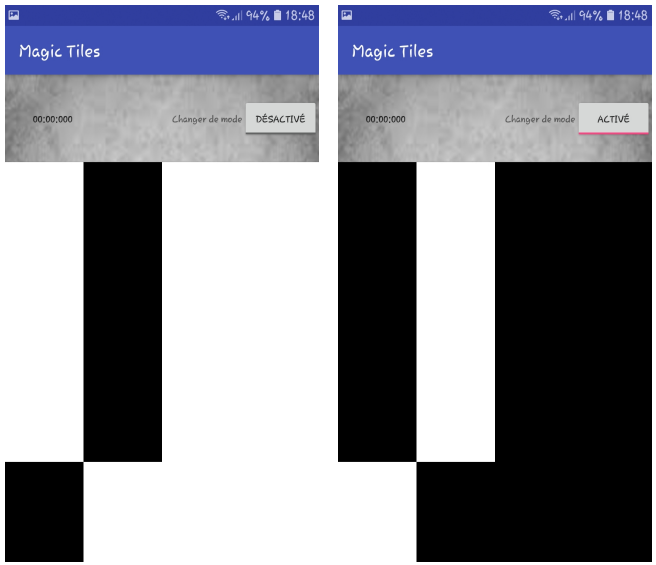
Android

Activity Game

L'objectif ici était de pouvoir ajouter une couleur aux boutons pour simuler des touches de piano. Ainsi, dès que cette activité est lancée, nous générons un nombre aléatoire compris entre 1 et 4 pour chacune des lignes afin de déterminer quel bouton sera noir. Les autres seront dès lors coloriés en blanc .

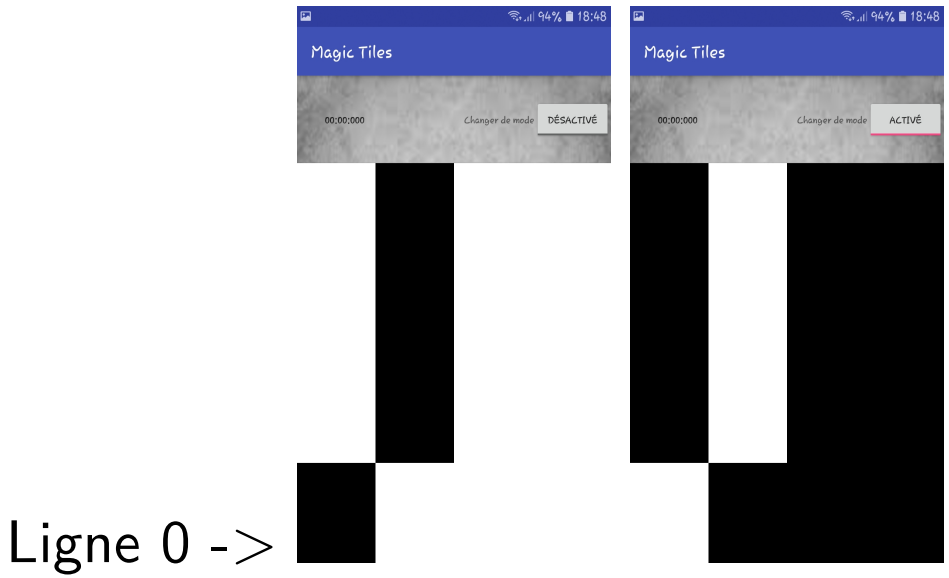
Activity Game

L'objectif ici était de pouvoir ajouter une couleur aux boutons pour simuler des touches de piano. Ainsi, dès que cette activité est lancée, nous générons un nombre aléatoire compris entre 1 et 4 pour chacune des lignes afin de déterminer quel bouton sera noir. Les autres seront dès lors coloriés en blanc .



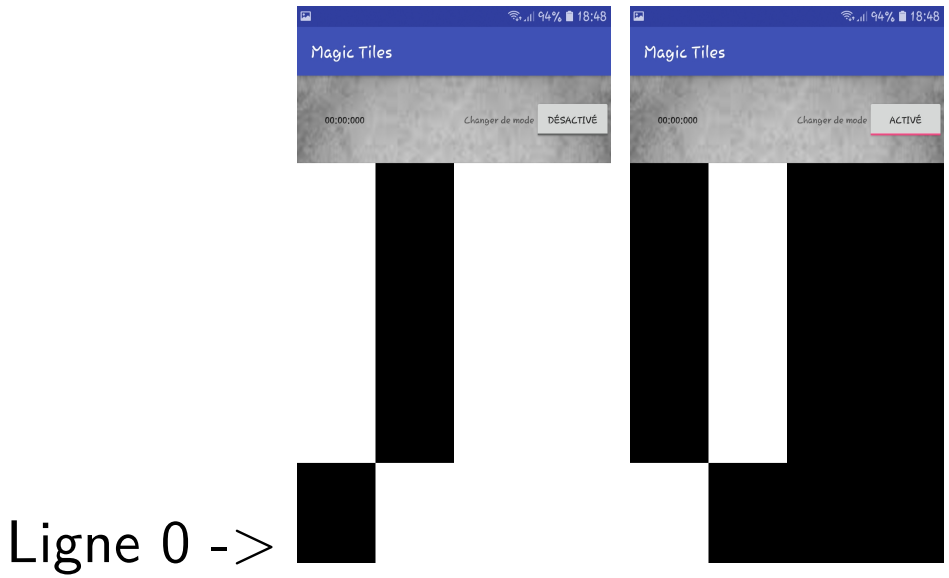
Activity Game

Une seule ligne de boutons est cliquable - nous l'appelleront " Ligne 0 " -, celle qui se situe tout en bas de l'écran, le reste des boutons sont désactivé. Nous n'avons donc ajouté des écouteurs qu'aux boutons de la ligne 0.



Activity Game

Une seule ligne de boutons est cliquable - nous l'appelleront " Ligne 0 " -, celle qui se situe tout en bas de l'écran, le reste des boutons sont désactivé. Nous n'avons donc ajouté des écouteurs qu'aux boutons de la ligne 0.



Activity Game

Lorsqu'on appuie pour la premier fois sur l'un des boutons de la ligne 0, le chronomètre sera démarré et le choix du mode ne sera dès lors plus accessible.

Changement de configuration

La méthode onSaveInstanceState(Bundle) est appelée sur l'activité avant sa destruction et la méthode onCreate(Bundle) est appelée sur la nouvelle activité

Changement de configuration

Android

Chemin

MagicTiles/
- src/
- res/
  - drawable/
  - drawable-hdpi/
  - layout/
  - layout-land/
  - values/
    - strings.xml
  - values-en/
    - strings.xml
  - values-es/
    - strings.xml

Android

Son

```
case04.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(!start){
            start = true;
            chrono.setBase(SystemClock.elapsedRealtime());
            chrono.start();
            onOff.setVisibility(View.INVISIBLE);
            mode.setVisibility(View.INVISIBLE);
        }
        if (ligne0 == 4) {
            if(onOff.isChecked()) { changementLigne(Color.WHITE, Color.BLACK); }
            else{ changementLigne(Color.BLACK, Color.WHITE); }
            if(nbtuiles <= 4){
                if(onOff.isChecked()){ finDefilement(nbtuiles, Color.BLACK); }
                else{ finDefilement(nbtuiles, Color.WHITE); }
            }
            nbtuiles--;
        }
        else{
            case04.setBackgroundColor(Color.RED);
            playGameOver();
            chrono.stop();
            Intent intent = new Intent(Game.this, ResultatNeg.class);
            startActivity(intent);
            finish();
        }
    }
});
```

Son

```
public void playGameOver(){
    stopGameSong();
    gameOverSong = MediaPlayer.create(this, R.raw.game_over);
    gameOverSong.setOnCompletionListener(
            new MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    stopGameSong();
                }
            });
    gameOverSong.start();
}

public void winSong(){
     stopGameSong();
    gameEndSong = MediaPlayer.create(this, R.raw.win);
    gameEndSong.setOnCompletionListener(
            new MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    stopGameSong();
                }
            });
    gameEndSong.start();
}

public void stopGameSong(){
    if(gameOverSong != null){
        gameOverSong.release();
        gameOverSong = null;
    }
    else if(gameEndSong != null){
        gameEndSong.release();
```

Création de l'application :

- Quand on crée un nouveau projet, plusieurs options nous sont offertes. Le premier sert à effectuer des tests (Playgrounds) de code swift pour déboguer sans démarrer de véritable projet. Le deuxième permet de créer un nouveau véritable projet. Le troisième sert si vous avez un repository à votre dispotition sur un git, il est alors possible de le récupérer ainsi.

Storyboard et programmation : MVC

La structure de projet iOS est basée sur le modèle MVC (Modèle Vue Controlleur)

- Modèle : Les données de l'applications
- Vue : La partie visuelle de l'application, c'est en quelque sorte l'IHM
- Controlleur : Contrôleur, c'est la partie logique de notre application, elle va permettre d'interagir entre les différentes vues

Storyboard et programmation

- Accueil : contenant les fonctionnalités de l'application
- AnimationButton.swift : création d'une animation sur les bouttons
- AppDelegate.swift : gère les événements du cycle de vie d'une application
- Camera.swift : l'appareil photo
- Game.swift : présenter à l'aide d'un boutton play en rouge
- Highscores.swift : historique des scores sauvegardés
- Localisation.swift : permet de faire une géolocalisation
- Save.swift : sauvegarde des temps mis pour le jeux
- ViewController.swift : corps de l'application

Storyboard et programmation

Pour en conclure avec cela, les fichiers AppDelegate.swift et Main.swift sont donc les code sources des contrôleurs de notre application. Les fichiers .storyboard sont les vues de l'application (LaunchScreen pour l'écran de démarrage d'une application, et Main pour la première vue de l'application)

Explication d'autre méthodes

- ViewDidLoad : appelée automatiquement lorsque la vue se charge
- didRecieveMemoryWarning :appelée lorsque l'iPhone est surchargé en mémoire vive (RAM), c'est donc pour permettre une bonne optimisation et donc éviter que l'application ne crash.
- @NSManaged : permet à Xcode de faire le lien avec l'attribut de notre entité
- CoreData : permet d'utiliser les classes qui s'y rapportent

Démonstration

Conclusion

Conclusion

Ce projet a fait l'objet d'une expérience à la fois intéressante et enrichissante, qui nous a permis d'améliorer nos connaissances et nos compétences dans le domaine du développement d'applications mobile.

The End