Rapport du projet

Ámchemmere Mohamed , Jamai Issam, M1 informatique

23 novembre 2019

Résumé

Dans le cadre de l'UE obligatoire développement mobiles, il est demandé de concevoir un jeu de notre choix sur la plateforme Android et iOS. Dans ce cadre, nous avons travaillé en binôme pour réussir à coder au mieux le projet. A cet effet, nous avons combiné notre temps, volonté et motivation, ainsi que nos capacités en informatique pour pouvoir réaliser au mieux ce projet qui nous tenait à cœur. Nous avons dû surmonter et pallier avec nos erreurs et faiblesses pour concevoir et répondre aux cahier des charges ainsi que pour offrir aux utilisateurs le jeu le plus attractifs et distrayants possible. A terme de quelques semaines de travail, nous avons donc réussi à mettre en place ce projet, qui répond au mieux aux demandes du cahier des charges.

1 Introduction

Le Pedometer, en accord avec les cahiers des charges proposés par le professeur de développement mobiles, a donc été réalisé sur Android et sur iOS. Nous avons fait en sorte que nos jeux soient semblables en tout point de vue. Nous nous attellerons donc à vous présenter les différentes ficelles de sa conceptions que ce soit, d'abord, à travers la version Android puis, dans un second temps, celle d'iOS.

2 Description de l'application

Pedometer est une application qui enregistre le nombre de pas que vous effectuez lorsque vous marchez ou même lorsque vous allez courir ; elle affiche celui-ci avec le nombre de calories que vous avez brûlées, la distance parcourue, le temps de marche ainsi que la vitesse par heure.Ainsi que dans cette application il y a un accés direct à la caméra ce qui permet de prendre des photos quand vous atteignez un objectif (de marcher 1500 pas par exemple)comme des des souvenirs, en plus il permet aussi d'accéder à la Map et du coup vous pouver savoir votre localisation quand vous le désirez . **Pedometer** une application très utile :

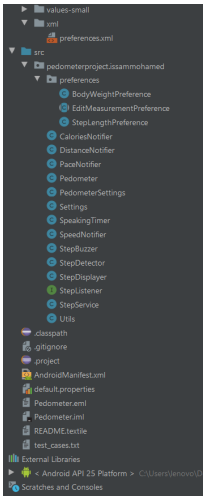
- • Si vous voulez vérifier le nombre de vos pas.
- •Si vous voulez essayer d'utiliser un podomètre.
- •Si vous voulez faire un régime.
- •Si vous faites des marches ou des balades.
- • Si vous faites du jogging.
- • Si vous voulez utiliser un podomètre réputé.
- • Si vous voulez accéder à la caméra quand vous voulez
- • Si vous voulez savoir votre localisation spontanément.
- • Si vous marchez beaucoup pour votre travail.
- • Si vous voulez un podomètre facile à utiliser.
- • Si vous voulez marcher davantage.
- • Si vous voulez prendre l'habitude de marcher.
- • Si vous ne vous sentez pas en forme.
- • Si vous souhaitez utiliser un podomètre avec un thème que vous aimez.
- • Si vous voulez utiliser votre smartphone au lieu de transporter un podomètre portatif

3 Architecture générale :

3.1 Version Android

Cette partie du rapport sera consacrée a présenter l'architecture générale de la version Android de notre projet. La version Android a été réalisé sur Android Studio en langage Java.

3.1.1 Architecture du projet



3.1.2 Classe Pedometre

C'est dans cette classe qu'on peut trouver la majorité des fonctionalitées de l'application. En effet, la vue liée à cette classe affiche le nombre de pas,les kilomètres, les pas/minutes ... et aussi le bouton pour acceder à la map, un bouton pour prendre une photo et un troisième pour jouer un morceau de musique.

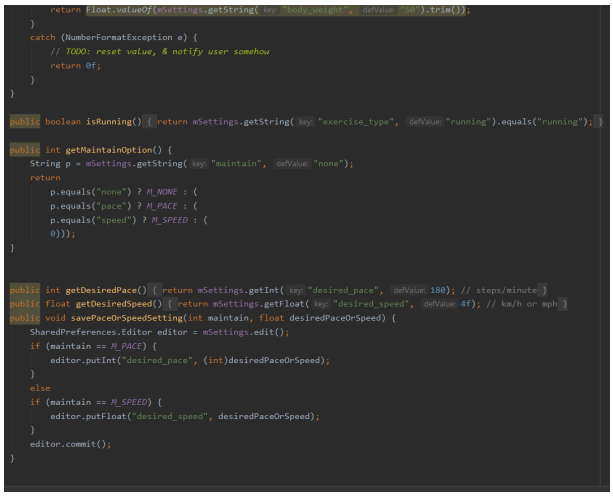
Voici une partie du code de cette classe :

```
@Override
public void onCreate(Bundle savedInstanceState) {
    Log.i(TAG,  msg: "[ACTIVITY] onCreate");
    super.onCreate(savedInstanceState);
    mStepValue = 0;
    mPaceValue = 0;
    setContentView(R.layout.main);
    mUtils = Utils.getInstance();
    Button button2 = (Button) findViewById(R.id.camera);
    button2.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            openCamera();
        }
    });
          final MediaPlayer mp = MediaPlayer.create( context: this, R.raw.song);
          Button play_button = (Button)this.findViewById(R.id.play);
          play_button.setOnClickListener(new View.OnClickListener() {
              public void onClick(View v) {
                  mp.start();
              }
              });
}
@Override
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
    switch (requestCode) {
        case PERMISSION_CODE:{
            if(grantResults.length>0 && grantResults[0] ==
                    PackageManager.PERMISSION_GRANTED){
                openCamera();
            }
            else {
                Toast.makeText( context: this, text: "Permission denied ...",Toast.LENGTH_SHORT).show();
            }
        }
```

3.1.3 Classe PedomettreSettings

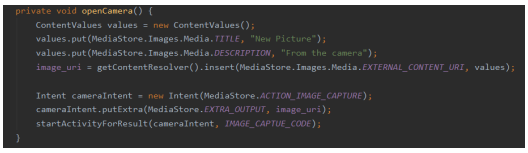
Cette classe prend en charge tout ce qui est paramètre de l'application comme la sensibilité, les unités, le type d'exercice etc

Voici une partie du code de cette classe :



3.1.4 La caméra

Pour que la caméra, notre application peut accéder à l'aide d'un bouton à la caméra du téléphone et prendre une photo qui sera ensuite enregistrer dans la galerie et tout ça à partir du code ci-dessous :



3.1.5 L'audio

On a inclu un morceau de musique dans l'application qui se déclanche via un bouton à l'aide du code ci-dessous :

3.1.6 l'internationalisation

Notre application fonctionne en 2 languages : Anglais et Français. Voici le fichier XML pour l'internationalisation :

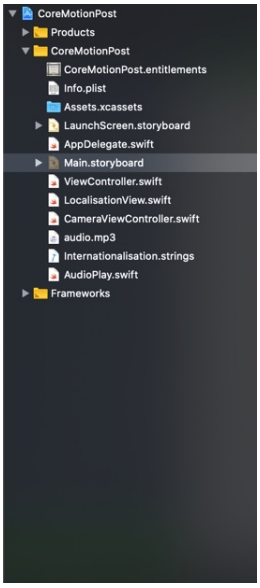
Key	Resource Folder	Untranslatable	Default Value	French (fr)
app_name	res		Pedometer	Pedometer
notification_subtitle	res		Counting your steps	Compter vos pas
started	res		Pedometer started.	Pedometer a commencé.
stopped	res		Pedometer stopped.	Pedometer s'est arrêté.
steps	res		steps	Pas
kilometers	res		kilometers	kilomètres
miles	res		miles	Miles
steps_per_minute	res		steps / minute	Pas / minute
kilometers_per_hour	res		kilometers / hour	kilomètres / heure
miles_per_hour	res		miles / hour	Miles / heure
calories_burned	res		calories burned	Calories brûlées
centimeters	res		centimeters	centimeters
inches	res		inches	Inches
kilograms	res		kilograms	kilogrammes
pounds	res		pounds	Livers
desired_pace	res		Desired pace:	Rythme souhaité:
desired_speed	res		Desired speed:	Rapidité souhaitée:
pause	res		Pause	Pause
resume	res		Resume	Reprendre
reset	res		Reset	Réinitialiser
settings	res		Settings	Paramètres
activity_settings	res		Pedometer Settings	Pedometer Paramètres

3.2 Version Ios

Cette partie-ci est destinée à la présentation de l'architecture générale de la version iOS de notre projet. La version iOS a été réalisé sur Xcode en langage Swift avec SpriteKit .

3.2.1 Architecture générale.swift

Dans l'image suivante on va vous montrer les différéntes classes qu'on a utilisé dans notre application , et après on va essayer expliquer le fonctionnement de chaque classe :



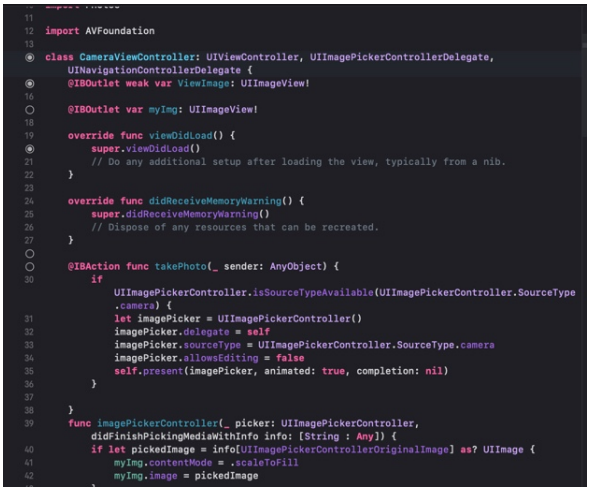
3.2.2 LocalisationView.swift

Afin de répondre aux demandes du cahier des charges, l'intégration de la géo-localisation était nécessaire. Le LocalisationView.swift permet à l'application d'accéder à l'Apple Maps. Dans un premier temps, il fallait d'abord faire appel à la classe MKMapView. Cette classe nous permet d'afficher la map sur l'écran. Dans un second temps, il fallait déterminer la position actuelle du joueur. Pour cela, il fallait faire appel à la classe CCLocationManager. On a ajouté une épingle sur la position actuelle de l'utilisateur avec en légende les coordonnées géographiques Et voici une image qui met l'accent sur un éxtrait de la classe **ViewLocalisation** :

```
class LocalisationView: UIViewController, CLLocationManagerDelegate, MKMapViewDelegate,
    UITextFieldDelegate {

    @IBOutlet weak var annotationTextField: UITextField!
    @IBOutlet weak var worldView: MKMapView!

    let locationManager = CLLocationManager()
    var currentLocation: CLLocation?

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        if CLLocationManager.locationServicesEnabled() {
            locationManager.delegate = self
            locationManager.desiredAccuracy = kCLLocationAccuracyHundredMeters

            locationManager.requestWhenInUseAuthorization()

            worldView.showsUserLocation = true
        }
        else {
            let alert = UIAlertController(title: NSLocalizedString("error", comment:
                "Error"),
                                          message: NSLocalizedString("location not enabled",
                                              comment: "Error"),
                                          preferredStyle: .alert)
            alert.addAction(UIAlertAction(title: "OK", style: .default, handler: nil))
            self.present(alert, animated: true, completion: nil)
        }
    }

    @IBAction func whereAmI(_ sender: Any) {

            print("current location is nil")
            let alert = UIAlertController(title: NSLocalizedString("error", comment:
```

3.2.3 CameraViewController.swift

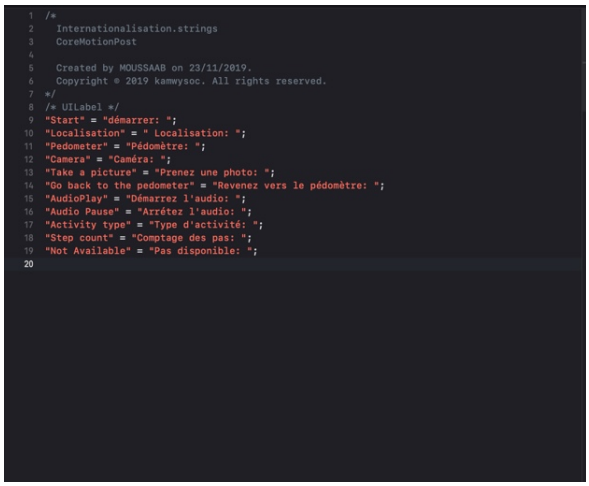
Le CameraViewController.swift va nous permettre d'accéder à l'appareil photo, de prendre une photo et de la sauvegarder. Pour cela, on utilise la classe UIImagePickerController. Voici une image qui montre un éxtrait de cette classe :



```swift
import AVFoundation

class CameraViewController: UIViewController, UIImagePickerControllerDelegate,
    UINavigationControllerDelegate {
    @IBOutlet weak var ViewImage: UIImageView!

    @IBOutlet var myImg: UIImageView!

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    @IBAction func takePhoto(_ sender: AnyObject) {
        if
            UIImagePickerController.isSourceTypeAvailable(UIImagePickerController.SourceType
            .camera) {
            let imagePicker = UIImagePickerController()
            imagePicker.delegate = self
            imagePicker.sourceType = UIImagePickerController.SourceType.camera
            imagePicker.allowsEditing = false
            self.present(imagePicker, animated: true, completion: nil)
        }

    }
    func imagePickerController(_ picker: UIImagePickerController,
        didFinishPickingMediaWithInfo info: [String : Any]) {
        if let pickedImage = info[UIImagePickerControllerOriginalImage] as? UIImage {
            myImg.contentMode = .scaleToFill
            myImg.image = pickedImage
```

3.2.4 Internationalisation.swift

Le processus d'internationalisation consiste à effectuer les modifications côté code et interface afin que le support multilingue soit assuré dans votre application. Il s'agit d'utiliser pour cela les APIs d'Apple pour faire en sorte que l'application ait un rendu adapté à la région ou bien le pays où l'application sera utilisée.ÒEt c'est ce que vous pouvez constatez dans l'image ci-dessous qui présente la classe **Internationalisation** :



```
/*
  Internationalisation.strings
  CoreMotionPost

  Created by MOUSSAAB on 23/11/2019.
  Copyright © 2019 kamwysoc. All rights reserved.
*/
/* UILabel */
"Start" = "démarrer: ";
"Localisation" = " Localisation: ";
"Pedometer" = "Pédomètre: ";
"Camera" = "Caméra: ";
"Take a picture" = "Prenez une photo: ";
"Go back to the pedometer" = "Revenez vers le pédomètre: ";
"AudioPlay" = "Démarrez l'audio: ";
"Audio Pause" = "Arrêtez l'audio: ";
"Activity type" = "Type d'activité: ";
"Step count" = "Comptage des pas: ";
"Not Available" = "Pas disponible: ";
```

3.2.5 AudioPlay.swift

On a essayé aussi d'intégrer la musique à notre application , qui s'appelle audio avec l'extension mp3 comme le montre cette image :

```swift
import AVFoundation


class AudioPlay : UIViewController {

    var audioPlayer = AVAudioPlayer()

    override func viewDidLoad() {
        super.viewDidLoad()

        let sound = Bundle.main.path(forResource: "audio", ofType: "mp3")
        do{
            audioPlayer = try AVAudioPlayer(contentsOf: URL(fileURLWithPath: sound!))

        }catch{
            print(error)
        }
    }
    @IBAction func audioplayButtonPressed(_ sender: Any) {
        audioPlayer.play()

    }

    @IBAction func audiopauseButtonPressed(_ sender: Any) {
        audioPlayer.play()

    }

}

```

3.2.6 ViewController.swift

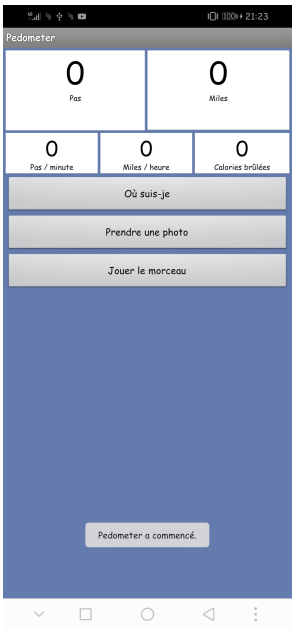
C'est la classe qui regroupe les boutons et c'est elle qui présente la vue principale de l'application où il y a l'affichage des pas comptées et le type d'activité(Marche ou La course ...) et voilà une image qui vous permettrez d'avoir une idée claire c'est le fonctionnement de la classe :

```swift
class ViewController: UIViewController {

    private let activityManager = CMMotionActivityManager()
    private let pedometer = CMPedometer()
    private var shouldStartUpdating: Bool = false
    private var startDate: Date? = nil

    @IBOutlet weak var startButton: UIButton!
    @IBOutlet weak var stepsCountLabel: UILabel!
    @IBOutlet weak var activityTypeLabel: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()
        startButton.addTarget(self, action: #selector(didTapStartButton), for:
            .touchUpInside)
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
        guard let startDate = startDate else { return }
        updateStepsCountLabelUsing(startDate: startDate)
    }

    @objc private func didTapStartButton() {
        shouldStartUpdating = !shouldStartUpdating
        shouldStartUpdating ? (onStart()) : (onStop())
    }
}


extension ViewController {
    private func onStart() {
        startButton.setTitle("Stop", for: .normal)
        startDate = Date()
        checkAuthorizationStatus()
        startUpdating()
    }

    private func onStop() {
        startButton.setTitle("Start", for: .normal)
        startDate = nil
        stopUpdating()
    }

    private func startUpdating() {
```

4 Les interfaces

4.1 Android

L'application possède 2 vues :
- La vue principale :



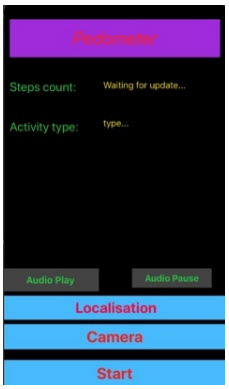
Cette vue contient les informations des pas, des kilomètres, des pas/minute ... ainsi que le bouton pour ouvrir la map, un bouton pour prendre une photo et l'enregistrer dans la galerie et finalement un bouton qui fait jouer un morceau de musique.
- La vue des paramètres :

Dans cette vue, on trouve plusieurs menus que l'utilisateur de l'application peut changer comme la sensiblité, les unités ... ainsi qu'il peut aussi saisir des informations comme le poids et la longueur du pas qui vont aider à avoir des résultats plus précis.

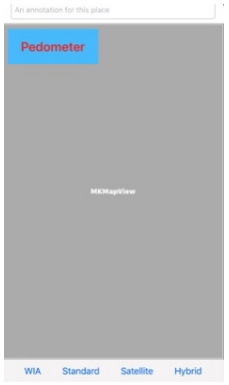
4.2 Ios

Dans cette partie , on essaye de vous montrer les vues finales de l'application , en utilisant les images ci-dessous :

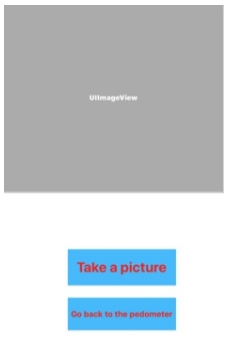
L'interface principale de l'application :



L'interface qui présente la map et qui localise l'utilisateur :



L'interface où l'utilisateur peut prendre des photos :



5 Conclusion

En conclusion, ce projet fut une belle épreuve qui nous a permis d'acquérir d'avantage d'expérience dans le développement d'applications pour mobiles. L'année de L3 nous a permis d'emmagasiner de nombreuses connaissances.Cette année de Master, par contre, nous a permis d'améliorer nos compétences à travers la pratique. Notre savoir a été mis en application en situation réelle. Sur le plan personnel, il a été très fructueux de travailler en équipe. Nous avons pu créer une véritable cohésion et mettre en place un dialogue. Ce projet nous a permis de confronter nos idées, développer notre esprit d'équipe, d'améliorer nos capacités rédactionnelles et de mettre de côté nos différents et idées antagonistes afin de pouvoir créer très bonne application.

Références

— Youtube
— Openclassroom
— Androidstudio
— SupInfo