

Android Cycling Trainer



Lechaire Thomas – FIN2
ETML - Lausanne
102 heures
M. Patrick Chenaux

Table des matières

1	Spécifications.....	3
1.1	Spécifications de départ et documentation associée.....	3
1.1.1	Objectifs et portée du projet	3
1.1.2	Fonctionnalités requises (du point de vue de l'utilisateur)	3
2	Planification.....	4
2.1	Planning(s) initial	4
3	Analyse.....	6
3.1	Faisabilité	6
3.1.1	Les compétences à acquérir :.....	6
3.1.2	Matériel/Logiciels	6
3.1.3	Recherches d'informations/documentation	6
3.1.4	Gestion du temps.....	6
3.1.5	Choix des logiciels.....	7
3.2	Document d'analyse et conception	9
3.3	Conception des tests	18
3.4	Planning détaillé.....	18
4	Réalisation	18
4.1	Dossier de réalisation.....	18
4.2	Modifications	23
5	Tests.....	23
5.1	Dossier des tests	23
6	Conclusion	23
6.1	Bilan des fonctionnalités demandées.....	23
6.2	Bilan de la planification.....	23
6.3	Bilan personnel	24
7	Divers	24
7.1	Journal de travail de chaque participant	24
7.2	Bibliographie.....	24
7.3	Webographie.....	24
8	Annexes.....	24

1 Spécifications

1.1 Spécifications de départ et documentation associée

1.1.1 Objectifs et portée du projet

- Analyser attentivement le cahier des charges.
- Proposer une interface graphique simple et intuitive.
- Réaliser l'application sous Android.
- Tester l'application.

1.1.2 Fonctionnalités requises (du point de vue de l'utilisateur)

- L'utilisateur doit pouvoir ...:
- Créer de façon simple une séance d'entraînement par exemple :
- Entraînement route « force » : 6 sprints 8''/52'' + 6 sprints 12''/48'' + 5 x 1' + 3 x 3' + 1 x 5' + 3 x 3' + 6 sprints 10''/50''.
- Entraînement VTT « côtes » : 2 tours de 3 côtes de 5' une doublée (7 côtes) : 4^e et 5^e en 15/15 et 6^e et 7^e en 20/20.
- Entraînement route « rythme » : 30' échauffement avec 10 sprints de 6 à 8'', 10' 20/20 côte 160-165 bpm, 15' vitesse 130 bpm, 3' 30/30 côte à bloc 180 bpm, 15' vitesse 130bpm, 20' contre-la-montre 165 bpm souplesse, 15' vitesse 130 bpm, 20' contre-la-montre 165 bpm braquet + gros, 20' retour au calme.
- Afficher un résumé de la séance d'entraînement, cela permet d'avoir un aperçu global de l'entraînement.
- Démarrer la séquence d'entraînement, chaque partie de la séquence doit être affichée de façon claire, au moyen de couleurs et contrôles graphiques adéquats, barres de défilement, boutons, clignotements etc.
- Mettre en pause la séquence d'entraînement et la redémarrer au moment souhaité.
- Sauvegarder une séance d'entraînement créée.
- Charger une séance d'entraînement précédemment sauvegardée.
- Insérer sa fréquence cardiaque de repos et maximale, elle pourra être utilisée pour travailler ou afficher une certaine zone cardiaque.
- Calculer un indice de récupération sur le vélo pour une date précise :
- $F_{cmax} - FC$ 1'30'' après effort, cette valeur augmente lorsque la forme physique augmente, afficher l'historique des valeurs enregistrées.

2 Planification

2.1 Planning(s) initial

- Dates de réalisation : du lundi 11.05.2015 au lundi 08.06.2015
- Horaire de travail :
Lundi 08h00-12h15 13h10-15h40 Pentecôte le 25 mai
Mardi 08h00-11h25 13h10-16h35
Mercredi - 13h10-16h35
Jeudi 08h00-11h25 12h20-15h40 Ascension le 14 mai
Vendredi 08h00-11h25 12h20-15h40 Pont de l'Asc. le 15 mai

- Nombres d'heures : 102

Tâches - objectifs	Nb 1/4 heure	Semaine 1
Absence - Imprévus	72	
Planification	0	
Analyse	24	
Documentation	0	
Maquette de réalisation	21	
Programmation	0	
Tests	21	
Debugging	0	
Rédaction Rapport et Journal de travail	21	
Zone Tampon	0	
Total planifié	456	
Total réalisé	0	

2.1 Semaine 1

Tâches - objectifs	Semaine 2
Absence - Imprévus	
Planification	
Analyse	
Documentation	
Maquette de réalisation	
Programmation	
Tests	
Debugging	
Rédaction Rapport et Journal de travail	
Zone Tampon	
Total planifié	
Total réalisé	

2.2 Semaine 2

Tâches - objectifs	Semaine 3																											
Absence - Imprévu																												
Planification																												
Analyse																												
Documentation																												
Maquette de réalisation																												
Programmation																												
Tests																												
Debugging																												
Rédaction Rapport et Journal de travail																												
Zone Tampon																												
Total planifié																												
Total réalisé																												

2.3 Semaine 3

Tâches - objectifs	Semaine 4																											
Absence - Imprévu																												
Planification																												
Analyse																												
Documentation																												
Maquette de réalisation																												
Programmation																												
Tests																												
Debugging																												
Rédaction Rapport et Journal de travail																												
Zone Tampon																												
Total planifié																												
Total réalisé																												

2.4 Semaine 4

Tâches - objectifs																												
Absence - Imprévu																												
Planification																												
Analyse																												
Documentation																												
Maquette de réalisation																												
Programmation																												
Tests																												
Debugging																												
Rédaction Rapport et Journal de travail																												
Zone Tampon																												
Total planifié																												
Total réalisé																												

2.5 Semaine 5

Le lien vers le fichier de planification initiale est disponible en annexe

3 Analyse

3.1 Faisabilité

- Dans le cadre de ce projet, une analyse est nécessaire afin de pouvoir anticiper les problèmes qui pourraient survenir durant la réalisation ou la programmation. Il faut vérifier certains points comme :
 - les compétences à acquérir ou approfondir
 - le matériel nécessaire ou à exploiter
 - les recherches d'informations particulières
 - la gestion du temps de travail
 - sélection des logiciels
 - les difficultés potentielles et les solutions envisagées

3.1.1 Les compétences à acquérir :

- Compétences en java et POO
- Compétences en programmation Android
- Bases en XML
- Compétences en Sql ou Base de données
- Connaissances de l'environnement Android Studio

3.1.2 Matériel/Logiciels

- Un Ordinateur PC
- Microsoft Office
- Android Studio
- Virtual Box
- Gimp ou InkScape pour le traitement d'image
- Navigation Internet

3.1.3 Recherches d'informations/documentation

- Recherches sur les bases de données Android
- Documentation sur l'environnement Android Studio
- Recherches sur les solutions pour les Tests ou Debugging
- Recherches sur les interfaces Android
- Documentation de Google pour le développement Android

3.1.4 Gestion du temps

La gestion du temps se fait grâce à la planification Initiale ainsi qu'au journal de bord qui permettent de suivre un fil rouge. La gestion du temps peut facilement devenir problématique. Il est donc important de respecter au maximum les délais fixé dans la planification (initiale et détaillée) afin de mener le projet à son terme.

3.1.5 Choix des logiciels

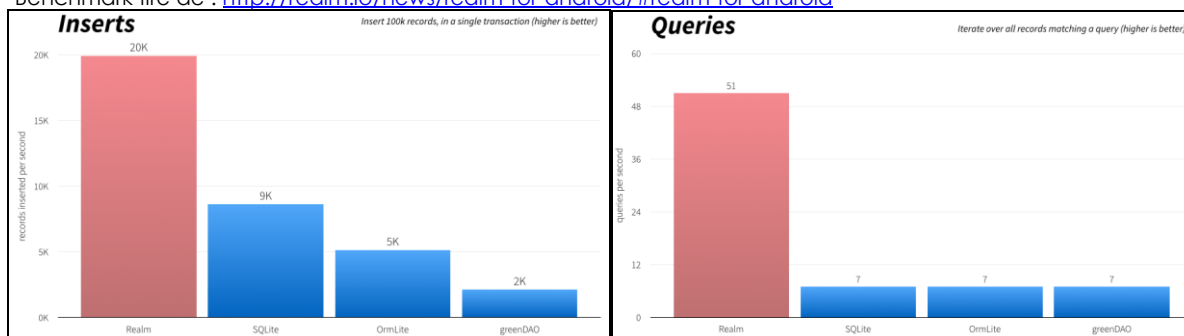
Concernant le logiciel de développement. Deux grands logiciels peuvent être envisagés. Eclipse projet d'Eclipse Foundation qui possède un plugin (ADT) qui s'intègre très facilement à Eclipse et qui permet de développer sous Android, ainsi que Android Studio qui est développé par Google et qui se base sur IntelliJ IDEA (autre logiciel de développement).

Note: If you have been using Eclipse with ADT, be aware that [Android Studio](#) is now the official IDE for Android, so you should migrate to Android Studio to receive all the latest IDE updates. For help moving projects, see [Migrating to Android Studio](#).

ADT n'étant plus mise à jour et Google recommandant son produit pour obtenir les dernières mises à jour, Android Studio a donc été choisi pour développer le projet.

Pour la partie base de données, SQL Lite est la solution de base implémentée pour Android. Cependant les recherches ont mises en avant un grand nombre de développeurs ayant des problèmes et principalement des problèmes liés à la vitesse d'exécution des requêtes ainsi qu'à la difficulté d'implémentation des bases de données avec SQL Lite. Comme solutions, beaucoup de réponses parlaient et mettaient en avant Realm qui une librairie permettant la création de base de données sur mobile. Pourquoi choisir Realm plutôt que SQL Lite ou un ORM (cf [Lexique](#)) comme greenDAO ? Premièrement, car Realm est facile à implémenter, il utilise son propre mécanisme de persistance (cf [Lexique](#)) et est capable de sauvegarde n'importe quel objet implémenté dans le code. Deuxièmement, car Realm est multiplateformes. Un fichier *.realm contenant une base de données peut très facilement être importée dans un autre projet que ce soit un projet Android ou IOS. Et enfin, car il est, d'après le benchmark trouvé, plus rapide qu'un ORM ou qu'une base de données SQL Lite. Realm est capable d'insérer 2000 enregistrements à la seconde contre 900 pour SQL Lite et 500 pour OrmLite et exécuter 51 requêtes secondes contre 7 pour les autres plateformes même s'il faut bien sûr prendre du recul sur ces résultats, car ils ont été effectués par les créateurs de Realm et non par un tiers.

Benchmark tiré de : <http://realm.io/news/realm-for-android/#realm-for-android>



3.1 Requêtes avec Realm (51 par sec)

3.2 Insertions avec Realm (2k par sec)

Pour la compilation du projet, un Samsung S3 version 4.3 a été utilisé ainsi que GenyMotion pour émuler des autres terminaux. L'émulateur Android permet l'installation de nouveau terminal, mais la mise en place est longue et complexe pour un résultat et une rapidité très moyenne. GenyMotion est une solution rapide et facile à mettre en place puisqu'un plugin est directement téléchargeable depuis Android Studio. Il s'intègre dans la barre des tâches et propose une liste de plus de plus de 80 devices dans 7 versions Android différentes. Pour les installer rien de plus simple car se sont de simples Machines Virtuelles gérées grâce à VirtualBox. Le téléchargement de la machine virtuelle suffit à avoir un émulateur prêt à être lancé. Un gain de temps énorme pour développer et tester son application sur un grand nombre de devices et ainsi augmenter la compatibilité, sans mettre en avant la possibilité de prendre des screenshots, de simuler des positions GPS, d'installer des applications par Drag and Drop ou encore de simuler les capteurs comme l'accéléromètre.

	Genymotion	Physical device
Deploy a 30 MB App	7s	21s
Start a device	15s	40s

3.3 Comparaison GenyMotion

Concernant le reste des logiciels utilisés, ils sont les suivants : Microsoft Office 2016 en preview pour la gestion de la partie administrative et rapport du projet, Gimp et Inkscape pour tout ce qui est la partie design et retouche (principalement pour les interfaces), Chrome pour la navigation et eduncanet, qui est la plateforme de l'ETML pour la

correspondance. Tous ces outils sont des outils gratuits et ils ont été choisis pour cela.

3.2 Document d'analyse et conception

3.2.1 Maquette graphique



Dans cette activité (cf [Lexique](#)), chaque entraînement affichera le nom choisi pour l'entraînement, l'icône du vélo selon le type d'entraînement et la date du jour de création de l'entraînement.

Deux Type d'entraînements :

- VTT (en Bleu) (1)
- Vélo de route (en rouge) (2)

(6) Le menu paramètre permet d'ajouter les fréquences cardiaques de repos et maximum.

Voir image 3.6

(3) Un geste (Swipe cf. [Lexique](#)) vers la gauche laissera apparaître un bouton permettant d'effacer l'entraînement.

(4) Chaque entraînement (ligne) est cliquable. Le click ouvre l'entraînement et laisse apparaître les séquences contenu dans chaque entraînement.

Voir Image 3.7 Eléments entraînement

3.4 Activité entraînements



3.5 Boîte de dialogue

(5) Le Bouton « Ajouter Entraînement » ouvre une boîte de dialogue permettant l'ajout d'un entraînement.

Voir : Image 3.5 Boîte de dialogue.

La boîte de dialogue apparaît lorsqu'on ajoute un entraînement.

- Une Zone pour ajouter le nom de l'entraînement
- 2 boutons pour définir le type d'entraînement VTT ou Route.
- 2 boutons pour valider ou annuler.

Fréquence Cardiaque au repos:
Fc Repos

Fréquence Cardiaque max:
FC max

VALIDER ANNULER

3.6 Insertion des fréquences cardiaques

Au clic sur le bouton paramètres de l'activité entraînement (Image 3.4) une boîte de dialogue s'ouvre et permet à l'utilisateur d'introduire dans la base de données sa fréquence cardiaque de repos et sa fréquence cardiaque maximum (qui sera utilisé pour calculer un indice de récupération)

Entraînement 1 1) +

Échauffement	RPM: 90	10:30'
Remarques: 3-4 sprints 10"	BPM: 155	
Sprint	RPM: 190	05:00'
Plateaux: 52 X 17-19	BPM: 180	
Récupération	RPM: 70	01:00'
	BPM: 90	
Effacer cet élément? 2) DELETE		
Contre-la-montre – à bloc	RPM: ...	15:00'
	BPM: ...	
3) DÉMARRER		

3.7 Eléments entraînements

(1) Un bouton pour ajouter un élément à l'entraînement.

L'action sur ce bouton ouvre l'activité détails.

Voir : Image 3.8 Détails activité

Chaque ligne représente un élément de l'entraînement, comme des sprints, de la récupération, un contre-la-montre. L'élément indique à l'utilisateur le temps de la séquence, les [bpm](#) et les [rpm](#).

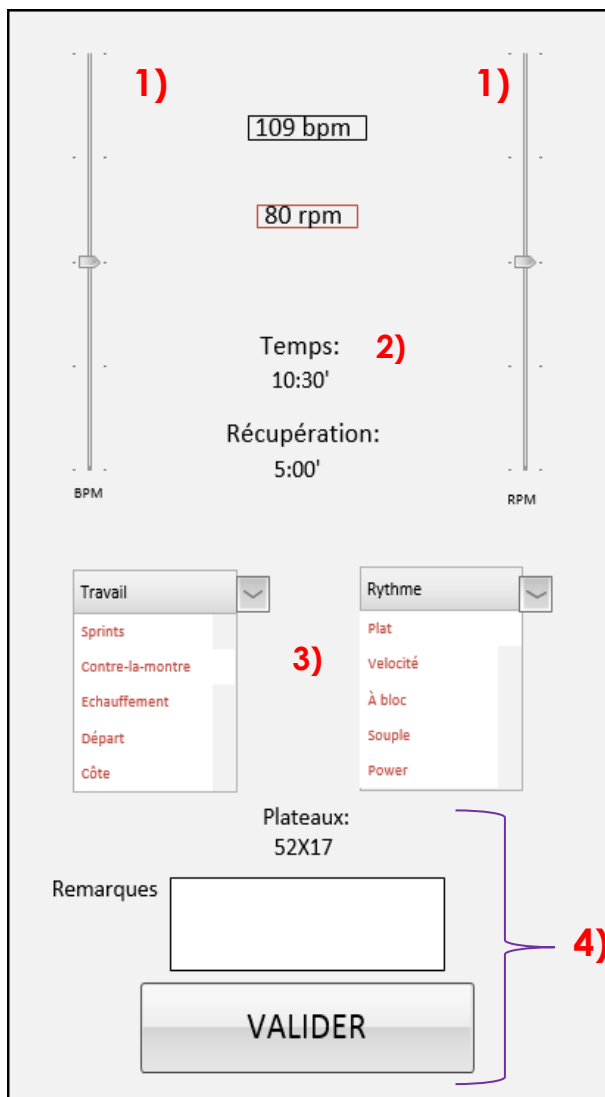
Pour les termes techniques voir Annexes : [Lexique](#)

(2) Comme pour l'entraînement. Un geste (Swipe) vers la gauche laissera apparaître un bouton permettant d'effacer un élément.

Chaque élément (ligne) est aussi cliquable permettant de modifier les valeurs de l'élément choisi.

(3) Le bouton démarrer ici, lance l'activité Timer qui démarre l'entraînement de vélo.

Voir : image 3.9 Timer.

**3.8 Détails activité**

Les détails de l'activité sont accessibles soit en ajoutant un élément soit en cliquant sur un élément de l'activité précédent (3.7) permettant ainsi la modification.

Dans cette activité il est possible de choisir plusieurs informations.

Les deux barres verticales permettent de choisir les bpm et les rpm (1) et mettent à jour automatiquement les valeurs dans les deux TextView (cf [Lexique](#)) prévues à cet effet.

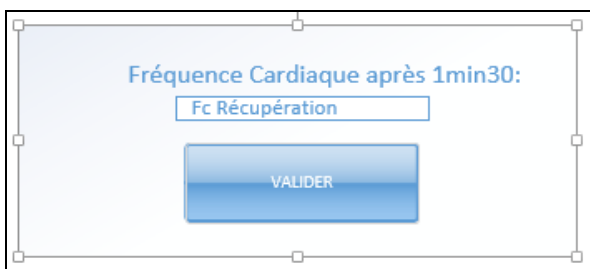
(2) Le temps et la récupération sont validés par des TimePicker (cf [Lexique](#)), présent dans Android. Le choix d'afficher le TimePicker directement ou dans une boîte de dialogue (cf [Lexique](#)), est encore à faire.

(3) Deux listes déroulantes ou Spinners seront implémentés. Le premier pour choisir le type de travail à effectuer, comme sprint, contre-la-montre ou côtes et le second pour choisir le rythme de la séquence (à bloc, Vélocité etc...)

(4) Un élément pour choisir les plateaux avant et arrière, une boîte de texte pour ajouter des remarques personnelles et un bouton « valider » pour enregistrer la séquence et l'ajouter à l'entraînement.



3.9 Timer



3.10 Récupération

Tous les éléments présents ici (1) proviennent de la séquence en cours et ont été entrés par l'utilisateur. L'affichage permet à l'utilisateur de savoir ce qu'il doit faire durant la séquence. Le type de travail, le rythme, les bpm et rpm, les plateaux les remarques et même la séquence suivant seront affichés à l'écran.

Cette activité permet de lancer les chronomètres et le temps des séquences (2) et de l'entraînement (3).

Le point 2 représente donc le temps de la séquence. Il a été défini par l'utilisateur dans l'activité 3.8 (Détails activité) et le point 3 représente lui l'entraînement au complet.

(4) Une série de contrôles permettant d'arrêter, de mettre sur pause ou de stopper l'entraînement.

En fin d'entraînement lorsque ce dernier est terminé, une boîte de dialogue permettra à l'utilisateur de rentrer sa fréquence cardiaque après 1m30 de récupération. Grâce à cette fréquence et la fréquence maximum entrée dans les paramètres, un indice de récupération (pour l'entraînement effectué) pourra être calculé.

3.2.2 Développement

Les recherches et la documentation sur internet ont permis de trouver des solutions rapides pour la base de données, la gestion du Swipe lors de l'effacement ou encore le choix du timing.

- Méthodologie :

La première étape consistera à découper le travail en tâches plus petites. La méthode la plus efficace est de travailler par activité. Création d'une activité, création de son interface, puis création des méthodes et des fonctionnalités requises pour cette activité et ainsi de suite. Ce choix est motivé par le fait qu'il y a une hiérarchie dans les activités et qu'il n'est pas possible de naviguer dans une application comme sur un site internet.

- Fonctionnalités à développer :

Pour activité illustrée à l'image 3.4 qui représente la première activité avec laquelle l'utilisateur peut interagir. L'utilisateur doit pouvoir ajouter un entraînement grâce au bouton « Ajouter Entraînement », lors de l'ajout d'un entraînement un nom et le type de l'entraînement sera demandé à l'utilisateur au moyen d'une boîte de dialogue lui permettant de remplir ces informations. Il doit pouvoir supprimer l'entraînement lors du swipe ou voir les éléments présents dans l'entraînement en cliquant sur la ligne de l'entraînement. Chaque ajout d'un nouvel entraînement entraînera automatiquement la sauvegarde de celui-ci dans la base de données.

L'activité de l'image 3.7 (activité éléments d'entraînement) représente les séquences d'un entraînement. La même structure d'activité en ligne est utilisée pour afficher les éléments que celle présente dans l'activité précédente (image 3.4), le swipe pour effacer un élément est présent lui aussi, cependant pour ajouter une séquence il faut utiliser la croix bleu ou bouton « ajouter » situé en haut à droite dans l'Action Bar (cf [Lexique](#)). Le clic sur cet élément de l'interface lance l'activité Détails permettant d'ajouter une séquence à notre entraînement. Le bouton démarrer l'entraînement situé en bas de l'activité permet quant à lui de lancer l'activité Timer et donc de démarrer l'entraînement. La modification d'une séquence enregistrée se fait par clic sur la ligne que l'utilisateur souhaite modifier. S'ouvre alors l'activité Détails avec les données de notre séquence.

L'activité Détails présentée à l'image 3.8 est l'activité qui s'ouvre si l'on souhaite ajouter une séquence à l'entraînement. Il est possible aussi possible d'ouvrir cette activité si l'utilisateur désire modifier une des séquences de son entraînement. Ajouter une séquence ouvre l'activité détails avec des valeurs vides. Modifier une séquence ouvre la même activité avec les valeurs de la séquence que l'on souhaite modifier. Il faudra, lors de la validation, vérifier qu'il s'agit bien de la

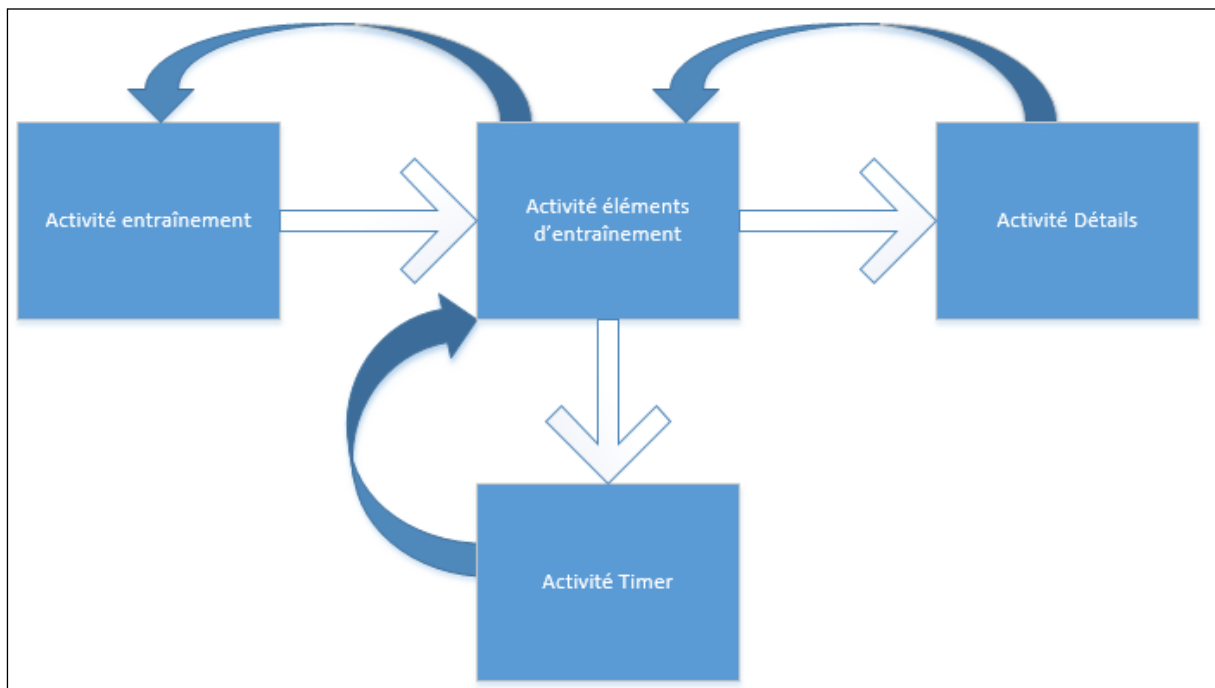
modification d'une séquence déjà existante afin de ne pas rajouter une nouvelle séquence.

La dernière activité est l'activité timer présentée par l'image 3.9. C'est l'activité principale de l'application, car c'est cette activité qui permet à l'utilisateur (cycliste) d'effectuer son entraînement. Quand l'utilisateur lance son entraînement, l'activité Timer prend alors le temps total de l'entraînement, ainsi que le temps et les informations issues de la première séquence de l'entraînement. Elle affiche alors les remarques, plateaux, Bpm, Rpm et toutes les autres informations présentes dans la séquence et démarre les timers du temps total de l'entraînement et du temps de la séquence. Une fois la séquence terminée, les données et le temps de la séquence suivante sont affichées mettant à jour les informations déjà présentes dans l'activité. Le timer de l'entraînement continue de tourner normalement, mais le timer de la nouvelle séquence se met à jour avec le temps correspondant et redémarre à zéro. La partie de gestion des timers est quelque chose de compliqué qu'il faudra bien gérer afin de ne pas laisser des timers tourner alors que l'application n'est plus active.

Les paramètres de l'application permettront d'ouvrir une boîte de dialogue. Cette boîte est nécessaire pour l'enregistrement de deux valeurs dans la base de données. La fréquence cardiaque de repos et la fréquence cardiaque maximale.

A la fin de l'entraînement, une boîte de dialogue s'ouvre et permet à l'utilisateur d'entrer sa fréquence de récupération (Fréquence obtenue 1min30 '' après la fin de l'entraînement). Un indice est calculé à partir de cette fréquence et de la fréquence max entrée dans les paramètres.

3.2.3 Hiérarchie des activités/vues

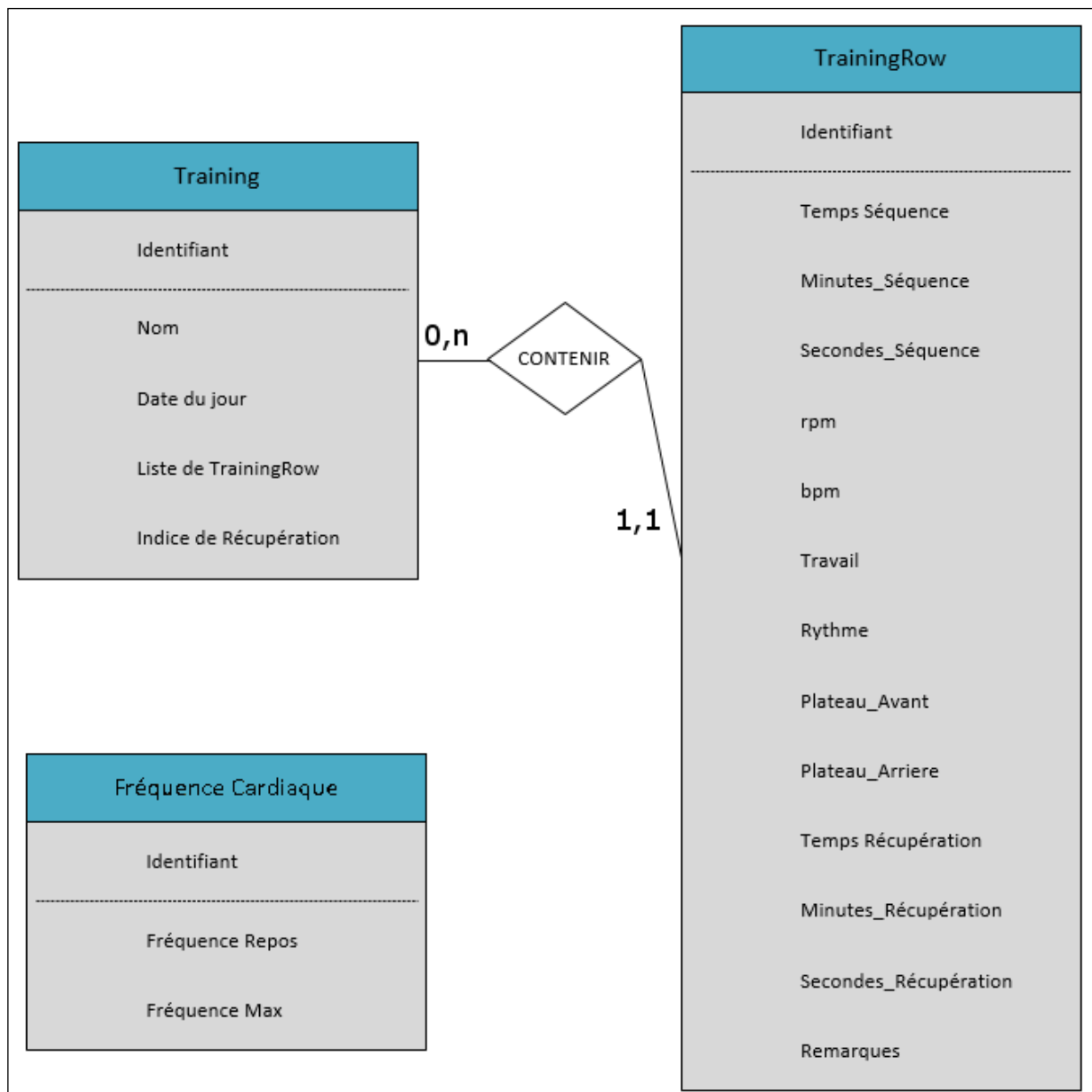


3.11 Hiérarchie des activités

La navigation ne s'effectuant pas comme sur un site internet, l'image ci-dessus représente les activités qui peuvent être appelées ou déclenchées par l'activité en cours. Le schéma permet de voir qu'il est, par exemple, impossible pour l'activité entraînement de déclencher (ouvrir) l'activité timer ou détails. Chaque activité a une fonction retour, qui, sur Android est gérée par le bouton « retour » présent sur chaque téléphone. L'activité Timer aura cependant un retour vers l'activité éléments d'entraînement qui mettra fin à l'entraînement en cours entraînant la perte de la progression déjà effectuée par l'utilisateur.

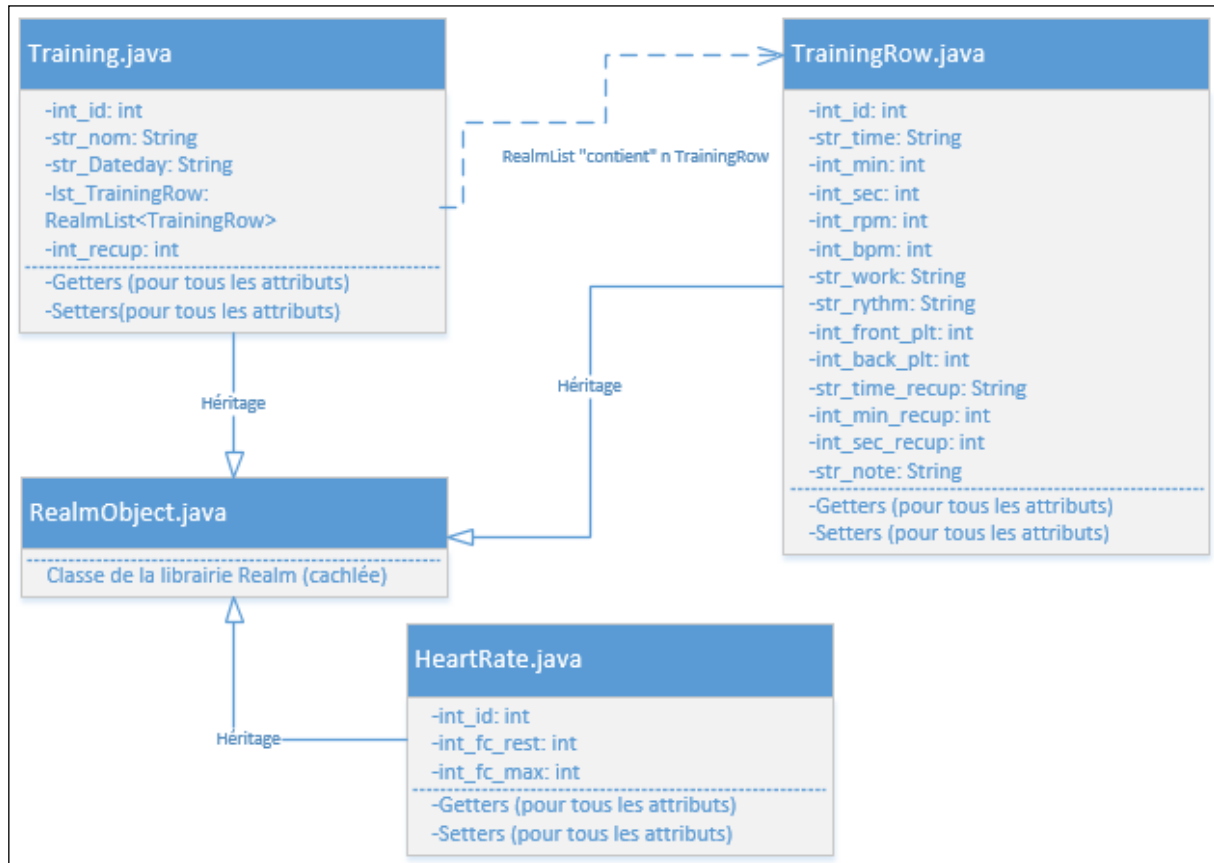
3.2.4 Base de Données

Realm sera utilisé pour la base de données. Cette utilitaire/librairie crée à partir d'une classe Java une table base sur la classe automatiquement. Cette partie n'étant pas modifiable, seul le MLD est présenté ici : Une seule base de données avec 3 tables. Une table pour les entraînements (Training) une table pour les séquences (TrainingRow) et une table pour les fréquences Cardiaques. La table fréquence cardiaque n'a pas de lien avec les deux autres. La table Training a un lien avec la table TrainingRow, car Training peut contenir 0 ou plusieurs TrainingRow. TrainingRow appartient à un et un seul entraînement.



3.12 MCD de la base de données

Cependant, la base de données étant gérée grâce aux classes, un model UML de Design de classes permettra de compenser l'absence de MLD, MPD



3.13 Design de classes UML

Le design de classes UML, remplace le MLD, MPD. Trois classes java héritent de la super classe RealmObject provenant de la librairie Realm. Realm s'occupe de convertir les classes qui héritent RealmObject.java. Cet héritage permet de faire le lien entre la classe créée et la sauvegarde des données dans la base.

Training.java gère les entraînements. TrainingRow.java gère les séquences dans un entraînement. Pour représenter un lien 0 à plusieurs (0 à n) avec Realm, il suffit d'instancier un tableau/liste (RealmList) contenant des objets.

Dans l'UML, la classe Training.java (entraînements) possède une RealmList (lst_TrainingRow) d'objets TrainingRow (séquence), au même titre qu'un entraînement possède 0 ou plusieurs séquences.

3.3 Conception des tests

La partie des tests est complexe et beaucoup de fonctionnalités devront être testées sur le moment, car la hiérarchie des vues sous-entend que si une fonctionnalité n'est pas implémentée correctement, celles dépendantes de la première ne pourront pas non plus fonctionner. Le temps imparti en fin de planification pour les tests finaux devra être utilisé correctement, mais il est réservé pour les tests finaux à savoir le comportement général de l'application.

Aucune méthodologie spécifique n'est prévue. Chaque bouton a une action spécifique, chaque partie de l'application doit se comporter comme décrit dans la partie analyse graphique et fonctionnalité. Si l'une des fonctionnalités ne s'exécute pas correctement lors d'une vérification (compilation de l'application pour vérifier le comportement de l'application), cela pourra être assimilé à un bug qu'il faudra résoudre. De par le fait que ce genre de contrôle fait partie du développement de l'application, elle s'intègre dans la planification sous l'élément « programmation ». En effet, si une fonctionnalité de clic, d'ouverture de nouvelle activité ne fonctionne pas, il faut résoudre le problème avant d'avancer, voilà pourquoi il n'y a pas de temps réservé aux tests avant la dernière semaine du projet.

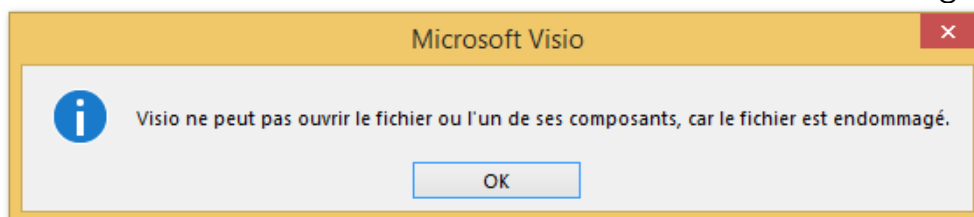
3.4 Planning détaillé

Selon les indications du chef de projet, le lien vers la planification détaillée se trouve en annexe.

4 Réalisation

4.1 Dossier de réalisation

- Office Visio étant absent de l'environnement de travail il a été nécessaire de trouver une Machine Virtuelle ayant la suite office installée dessus. La création des schémas s'est faite à l'intérieur de l'environnement virtuel. Un problème est survenu lors de la reprise du travail, le lendemain, car le fichier vision avait été endommagé.



Une partie du travail a donc dû être refaite et environ dix quarts d'heures ont été utilisés pour refaire les schémas et interfaces.

- Installation de Android Studio. Version 24.2
Téléchargement de L'IDE (cf [Lexique](#)) & du SDK (cf [Lexique](#))

Fichier *.exe. Installation Basique.

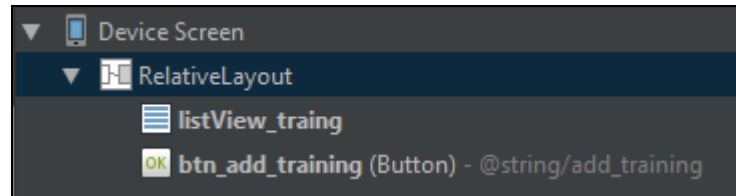
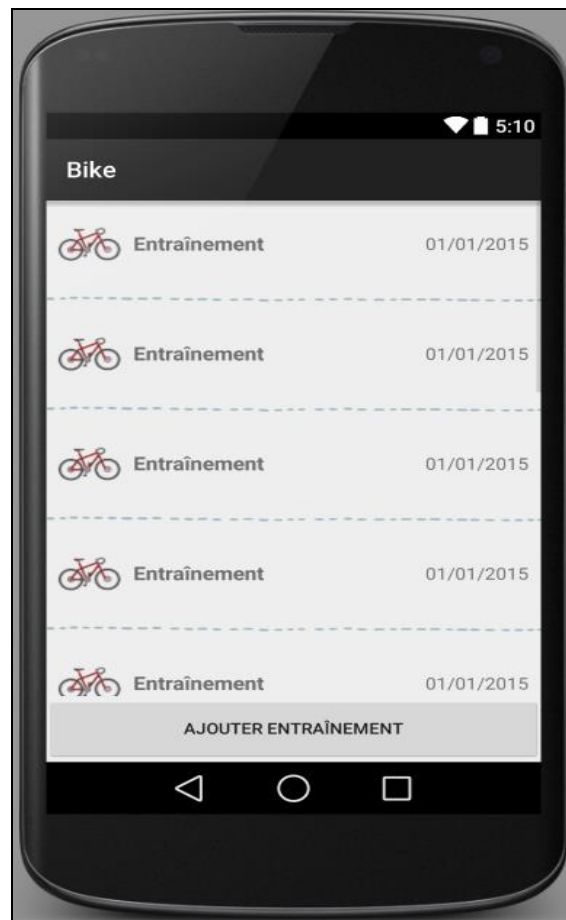
- Création d'un nouveau projet sous Android Studio.
- Installation des libraires pour débiter le développement.
Pour l'installation des libraires dans Android Studio il suffit d'ajouter une ligne dans le fichier build.gradle de l'application.
Exemple : compile 'io.realm:realm-android:0.80.1'

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    compile 'com.android.support:appcompat-v7:22.0.0'  
    //android Realm Library  
    compile 'io.realm:realm-android:0.80.1'  
    /* android library for swipe gesture */  
    compile 'com.android.support:recyclerview-v7:21.0.0'  
    compile 'com.android.support:support-v4:22.+'  
    compile "com.daimajia.swipelayout:library:1.2.0@aar"  
    /* android Yoyo Library */  
    compile 'com.nineoldandroids:library:2.4.0'  
    compile 'com.daimajia.easing:library:1.0.1@aar'  
    compile 'com.daimajia.androidanimations:library:1.1.3@aar'  
    //android Better Picker Library  
    compile ("com.doomonafireball.betterpickers:library:1.6.0") {  
        exclude group: 'com.android.support', module: 'support-v4'  
    }  
}
```

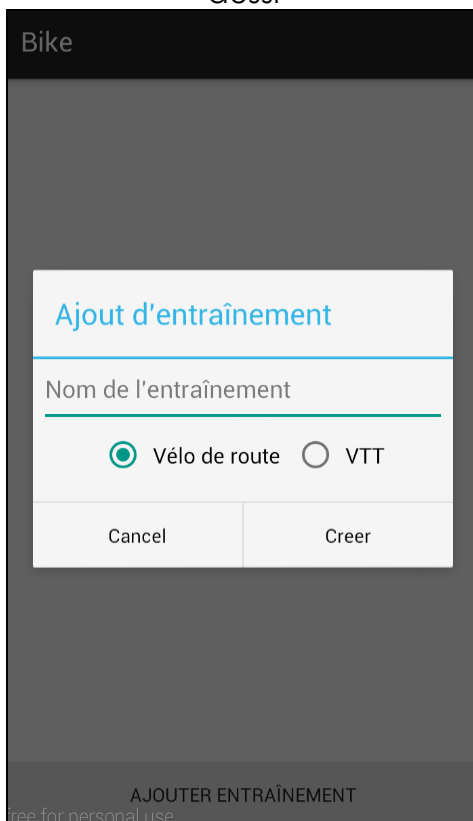
4.1 Installation des librairies

- Design des premières interfaces dans Android Studio.

- Création du layout pour la classe Training_activity.java. Un container pour le layout de type « RelativeLayout » contenant une liste de type « ListView » pour afficher les entraînements au fur et à mesure. Création d'un Bouton qui va lancer la boîte de dialogue.

**4.2 Hiérarchie des éléments de l'activité****4.3 Interface Training_activity.java**

- La boîte de dialogue pour entrer un entraînement est implémenté aussi



Bike

Ajout d'entraînement

Nom de l'entraînement

☒ Vélo de route ☐ VTT

Cancel Creer

AJOUTER ENTRAÎNEMENT

free for personal use

4.4 Boîte de dialogue pour l'ajout d'entraînement

-



- Les 3 interfaces principales sont terminées.
- Implémentation des éléments de l'Interface Utilisateur (UI)
- Ce paragraphe permet de reproduire ou reprendre le projet par un tiers.
Pour chaque étape, il faut décrire sa mise en œuvre. Typiquement:
 - Versions des outils logiciels utilisés (OS, applications, pilotes, librairies, etc.)
 - Configurations spéciales des outils (Equipements, PC, machines, outillage, etc.)
 - Code source des éléments logiciels développés.
 - Modèle physique d'une base de données.
 - Arbres des documents produits.
- Il faut décrire le parcours de réalisation et justifier les choix.

4.2 Modifications

- Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.
 - Date, raison, description, etc.

5 Tests

5.1 Dossier des tests

- On dresse le bilan des tests effectués (qui, quand, avec quelles données...) sous forme de procédure (tableau).
- Si des tests prévus dans la stratégie n'ont pas pu être effectués:
 - raison, décisions, etc.
- Liste des bugs répertoriés avec la date de découverte et leur état:
 - Corrigé, date de correction, corrigé par, etc.

6 Conclusion

6.1 Bilan des fonctionnalités demandées

- Il s'agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.
- Si ce n'est pas le cas, mesuré en « % » ou en « temps supplémentaire » le travail qu'il reste à accomplir pour terminer le tout.

6.2 Bilan de la planification

- Distinguer et expliquer les tâches qui ont généré des retards ou de l'avance dans la gestion du projet.

6.3 Bilan personnel

- Si c'était à refaire:
 - Qu'est-ce qu'il faudrait garder ? Les plus et les moins ?
 - Qu'est-ce qu'il faudrait gérer, réaliser ou traiter différemment ?
- Qu'est-ce que ce projet m'a appris ?
- Suite à donner, améliorations souhaitables, ...
- Remerciements, signature, etc.

7 Divers

7.1 Journal de travail de chaque participant

- Date, activité (décrit afin de reproduire le cheminement du projet), durée.

7.2 Bibliographie

- Références des livres utilisés durant le projet.

7.3 Webographie

- <http://Wikipedia.fr>
- <http://developer.android.com/>

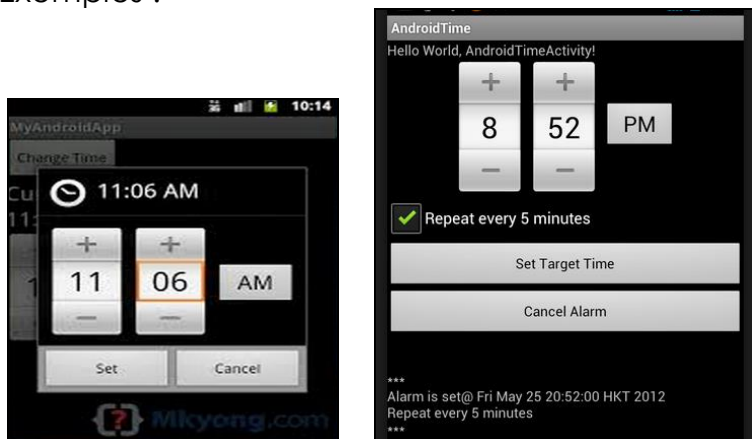
8 Annexes

- Listing du code source (partiel ou, plus rarement complet)
- Guide(s) d'utilisation et/ou guide de l'administrateur
- Etat ou « dump » de la configuration des équipements (routeur, switch, robot, etc.).
- Extraits de catalogue, documentation de fabricant, etc.
- Photocopies diverses, etc.
- Planification initiale.
- Planification détaillée.

8.1 Lexique

- **ORM** : Un ORM (Object-relational mapping) ou en français mapping objet-relationnel est une technique de programmation qui crée l'illusion d'une base de données orientée objet à partir d'une base de données relationnelle. (Wikipedia)
- **Persistence** : La persistance en programmation informatique se réfère au mécanisme responsable de la sauvegarde et de la restauration des données.
- **Activité** : Une activité est une fenêtre, un élément visuel ou plus précisément l'Interface graphique de l'application. Elle est parfois appelée UI pour User Interface (Interface Utilisateurs) ou encore Vue (View)

- **Swipe** : est le nom d'un geste utilisé sur les téléphones portables et qui consiste à faire glisser son doigt de droite à gauche ou de gauche à droite.
- **TextView** : Une TextView est un élément présent dans Android et qui permet de recevoir du texte à l'intérieur. Il n'est pas modifiable et pourrait être assimilé à une étiquette ou un label.
- **Bpm** : Battements par minutes. Concerne le rythme Cardiaque
- **Rpm**, Rotation par minute, révolution par minute, tour par minute. Concerne le cycliste et plus précisément les tours de pédales effectuées.
- **TimePicker** : un timePicker est un élément présent dans Android qui permet d'ouvrir un pop-up afin de nous permettre de choisir le temps. Le temps peut être une date une heure ou les deux.
Exemples :



8.1 TimePicker intégré dans l'activité 8.2 TimePicker avec boîte de dialogue

- **Boîte de dialogue** : Une boîte de dialogue est une sorte de pop-up qui apparaît sur l'écran de l'utilisateur et qui permet d'interagir avec. Ci-dessus un exemple d'une boîte de dialogue contenant un TimePicker
- **Action Bar** : L'Action Bar est la barre de l'application dans laquelle le titre de l'application ou le logo est affiché. Elle est modifiable et il est donc possible d'ajouter un menu ou des boutons spécifiques.
- **IDE** : Environnement de développement. L'IDE correspond au logiciel permettant de développer notre application.
- **SDK** : pour SoftWare Development Kit, kit de développement logiciel. Contient les compilateurs et tous les éléments nécessaires au développement (spécifique au langage et à la plateforme sur laquelle l'application est développée)