

Pact 2016-2017

**Rapport de tests PAN4 :**

**Rhythm Run**



Saousan KADDAMI

Haozhe SUN

**Introduction :**

L’objectif du module est d’effectuer des tests qui englobent des tests de fonctionnalités dans des modules spécifiés, dont le nombre est limité, et des tests globaux qui mettent ensemble plusieurs modules à la fois.

Les tests du PAN4 reprennent les tests qui ont échoué au PAN3 et rajoutent des tests qui étaient de base prévus pour le PAN4. Ci-joint le premier plan de Tests qui était prévu au PAN3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N° | Objectif du test | Modules testÉs | EntrÉe | Sortie | Testeur | ÉCHEANCE |
| I | Tests globaux | | | | | |
| I-1 | Tester qu'une secousse du téléphone affiche la fréquence sur l'application Android | Audio tempo /Android | Accélération du téléphone | Valeurs (double) du tempo sur l'activité | Haozhe | **PAN4** |
| I-2 | Tester qu'après accélération du téléphone, on peut obtenir la lecture de la bonne musique | Récupération des données/Audio/Android | Accélération du téléphone | Lecture de la musique sur ActivityRun | Haozhe | **PAN4** |
| I-3 | Test de la vérification du mot de passe/ identifiant dans le login | Android/ Client serveur | String (mdp+id) sur loginActivity | Connexion / String message d'erreur | Haozhe | PAN3 ✓ |
| I-4 | Tester que le rythme cardiaque est bien synchronisé sur l'application | Récupération des données/Android | Rythme cardiaque humain | Valeurs (double) du rythme sur l'activité | Saousan | **PAN4** |
| II | Tests de fonctionnalités | | | | | |
| II-1 | Tester la réponse d'Android pour afficher l'historique | Android (lecture de mémoire) | Onclick sur HistoricButton | Activité de la listview des historiques | Haozhe | PAN3✓ |
| II-2 | Tester que le module audio peut adapter une musique à un tempo en entrée | Audio | i/ Tempo(double)  ii/Chemin absolu (string) | Chemin absolu (String) | Saousan | **PAN4** |
|  |  |  |  |  |  |  |
| II-3 | Tester qu'un tempo+chemin absolu donne la lecture sur Android de la musique adaptée | Audio/Android | i/ Tempo(double)  ii/Chemin absolu (string) | Lecture de la musique sur ActivityRun | Haozhe | **PAN4** |
|  |  |  |  |  |  |  |
| II-4 | Tests spécifiques aux capteurs cardiaques | Récupération des données/Android | Données relatives aux activités cardiaques | Résultats vus à l’écran | Saousan | **PAN4** |
| II-5 | Tests fiabilité du podomètre | Récupération des données/Android | Un ensemble d'accélérations | Un ensemble des images par podomètre | Saousan | PAN3✓ |
| II-6 | Test fiabilité d'étirement | Audio/Android | Couples (tempo, chemin absolu) | Lectures en fonction des couples | Haozhe | **PAN4** |
| II-7 | Test fiabilité du login | Récupération des données/Android | Couples(id,mdp) | Résultats en fonction des entrées | Haozhe | PAN3✓ |
| II-8 | Test fiabilité historique | Android (lecture de mémoire) | Runs successifs | Listview des historiques | Haozhe | PAN3✓ |
| II-9 | Test Bluetooth | Android (lecture de mémoire) | Essai connexion Bluetooth | Connexion réussie affichée sur l’écran | Saousan | **PAN4** |

Le plan de tests adopté pour le PAN4 est le suivant :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N° | Objectif du test | Modules testÉs | EntrÉe | Sortie | Testeur | ÉCHEANCE |
| I | Tests globaux | | | | | |
| I-1 | Tester qu'une secousse du téléphone affiche la fréquence sur l'application Android | Audio tempo /Android | Accélération du téléphone | Valeurs (double) du tempo sur l'activité | Haozhe/Saousan | **PAN4** |
| I-2 | Tester qu'après accélération du téléphone, on peut obtenir la lecture de la bonne musique | Récupération des données/Audio/Android | Accélération du téléphone | Lecture de la musique sur ActivityRun | Haozhe | **PAN4** |
| I-3 | Tester que le rythme cardiaque est bien synchronisé sur l'application | Récupération des données/Android | Rythme cardiaque humain | Valeurs (double) du rythme sur l'activité | Saousan | **PAN4** |
| II | Tests de fonctionnalités | | | | | |
| II-1 | Tester qu'un tempo+chemin absolu donne la lecture sur Android de la musique adaptée | Audio/Android | i/ Tempo(double)  ii/Chemin absolu (string) | Lecture de la musique sur ActivityRun | Haozhe | **PAN4** |
|  |  |  |  |  |  |  |
| II-2 | Tests spécifiques aux capteurs cardiaques | Récupération des données/Android | Données relatives aux activités cardiaques | Résultats vus à l’écran | Saousan | **PAN4** |
| II-3 | Test fiabilité d'étirement | Audio/Android | Couples (tempo, chemin absolu) | Lectures en fonction des couples | Haozhe | **PAN4** |
| II-4 | Test Bluetooth | Android (lecture de mémoire) | Essai connexion Bluetooth | Connexion réussie affichée sur l’écran | Saousan | **PAN4** |

**TESTS GLOBAUX :**

**I-1 Test secousse tempo sur l’application :**

**Modules testés :** Audio tempo /Android

**Objectif :** Tester qu'une secousse du téléphone affiche la fréquence sur l'application Android

**Date :** 23.04.2017 de 21h40 – 22h10

Audio

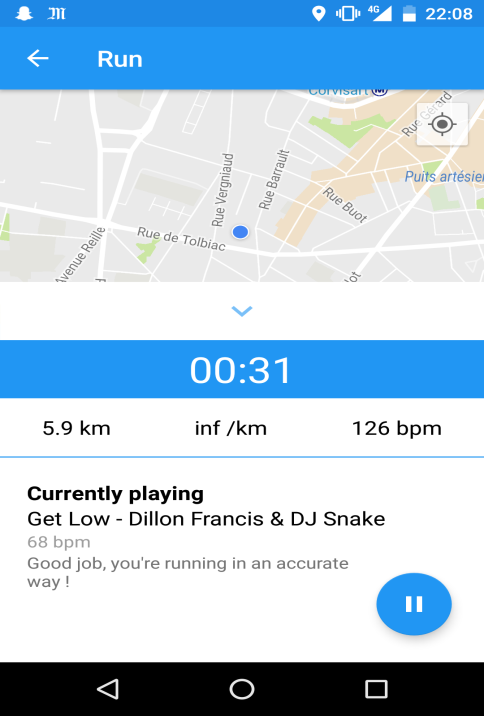
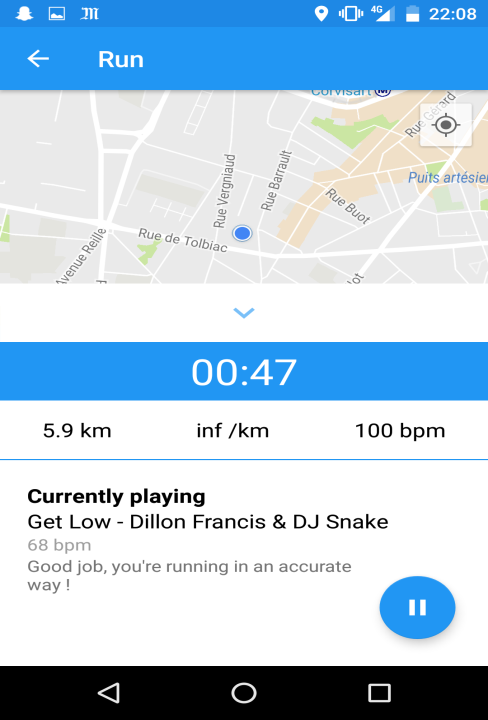
Android

Secousse chemin absolu Ecouter musique

**Entrée  :** accélération du téléphone

**Sortie  :** Valeur du tempo sur l’activité

**Résultats :**



Nous avons effectué trois essais avec 3 rythmes de secousse différents. Nous avons bien constaté qu’avec différents rythmes de secousse, le tempo, soit le bpm, change. En effet, le bpm varie entre 100 et 200.

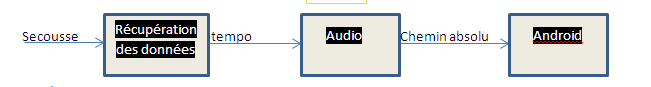
Cependant, nous avons constatons que le détecteur est un petit peu trop sensible, même si nous posons le téléphone sur la table, le tempo varie parfois quand même avec une petite amplitude, quand nous tapons la table avec un minimum d’effort. En plus, nous avons constaté un petit retard par rapport à notre secousse et le changement tempo. Ceci peut venir du fait que physiquement, il faut un minimum de temps pour calculer un changement de rythme. En pratique, cela ne posera pas de problème pour les coureurs puisque les coureurs ne vont jamais accélérer tout un coup ou s’arrête brutalement.

**I-2 Test secousse musique sur l’application :**

**Modules testés :** Audio tempo /Android

**Objectif :** Tester qu'une secousse du téléphone affiche la fréquence sur l'application Android

**Date :** 23.04.2017 de 21h40 – 22h10

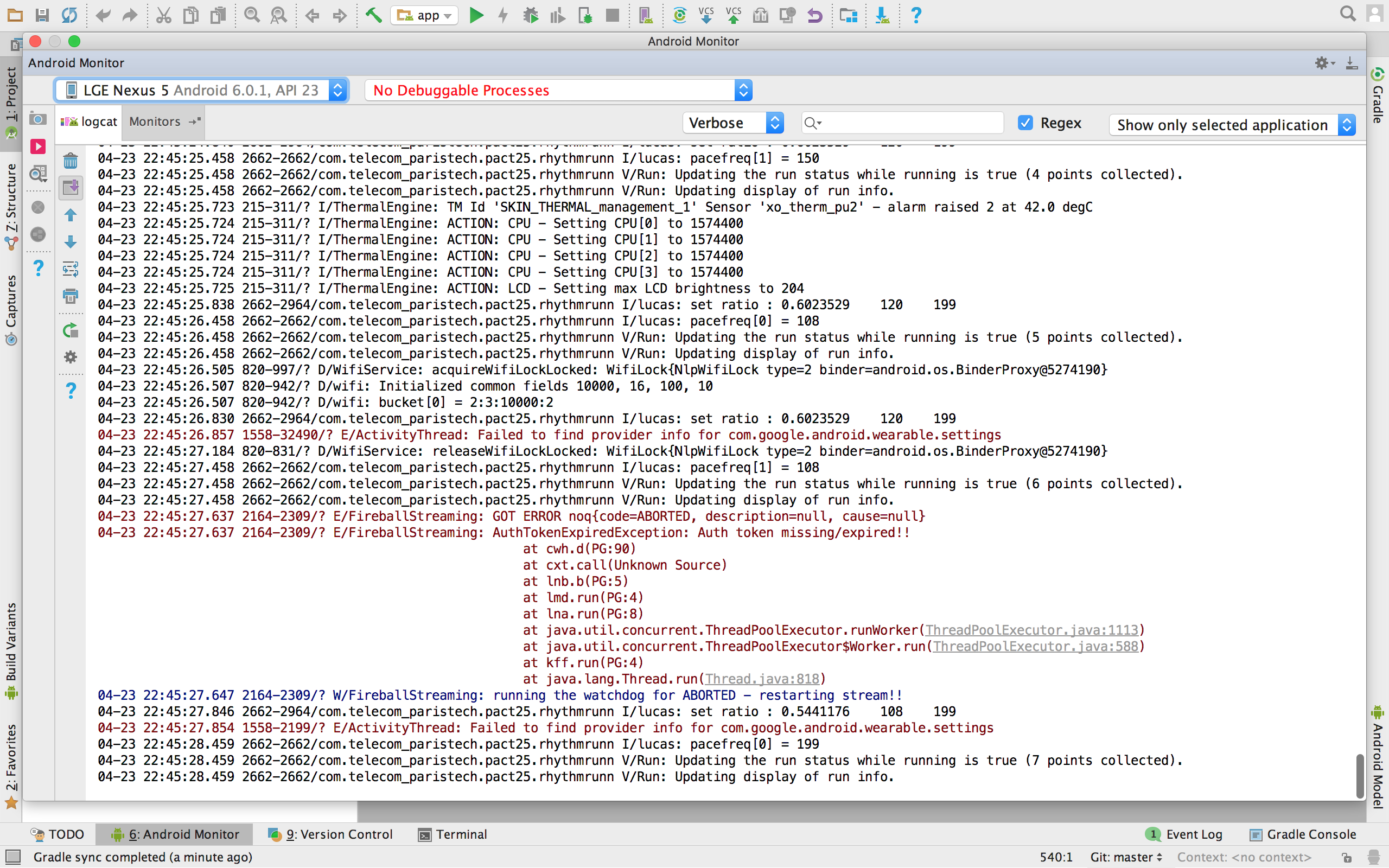


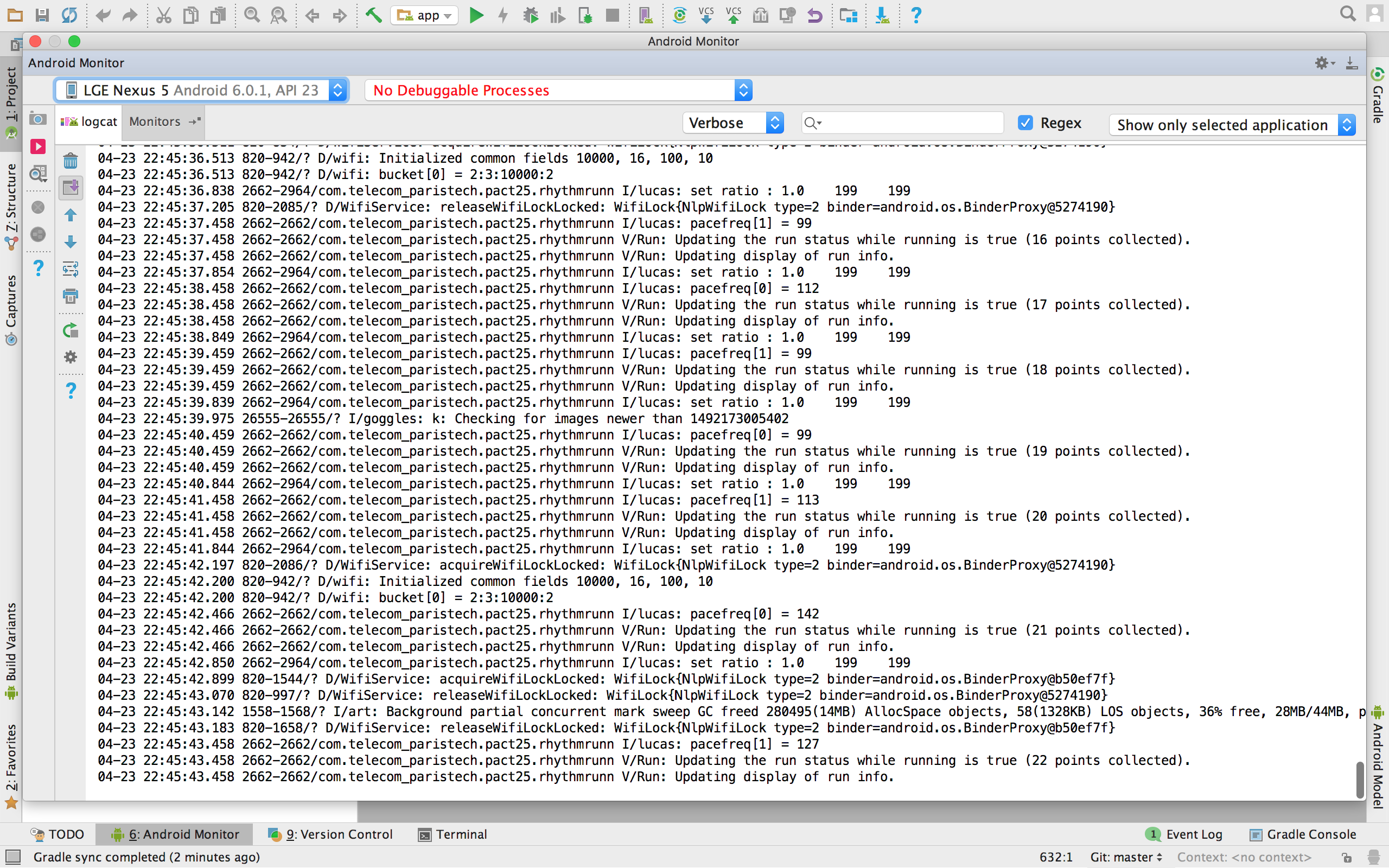
**Entrée  :** Accélération du téléphone

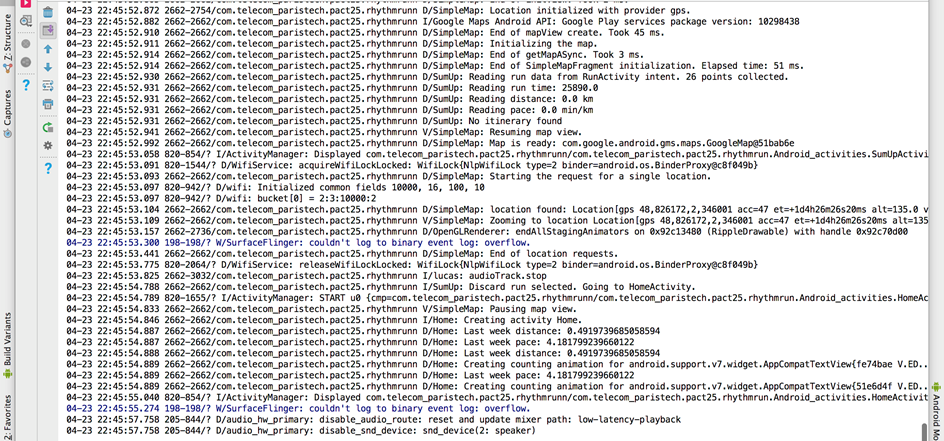
**Sortie  :** Lecture de la musique sur l’activité

**Résultats :**

Nous avons capturé les logs au lancement de la course, au courant de la course et à l’arrêt de la course :

Au lancement de la course :

Au cours de la course :

A l’arrêt de la course :

Nous avons notre appli n’a pas de bug à priori. Si nous nous mettons en mode free, nous écoutons que la musique s’adapte réellement au changement de bpm, c’est-à-dire notre secousse. La lecture de la musique est bien correcte.

Bien entendu, l’adaptation de la musique subit, elle aussi, un léger retard par rapport à l’action humaine.

**I-3 Test de synchronisation du rythme cardiaque sur l’application :**

**Modules testés :** Récupération des données /Android

**Objectif :** Tester que le rythme cardiaque est bien synchronisé sur l'application

**Date :** Samedi 22/04/17 à 12 h

Récupération des données

Android

Rythme Run Connexion Valeur sur appli

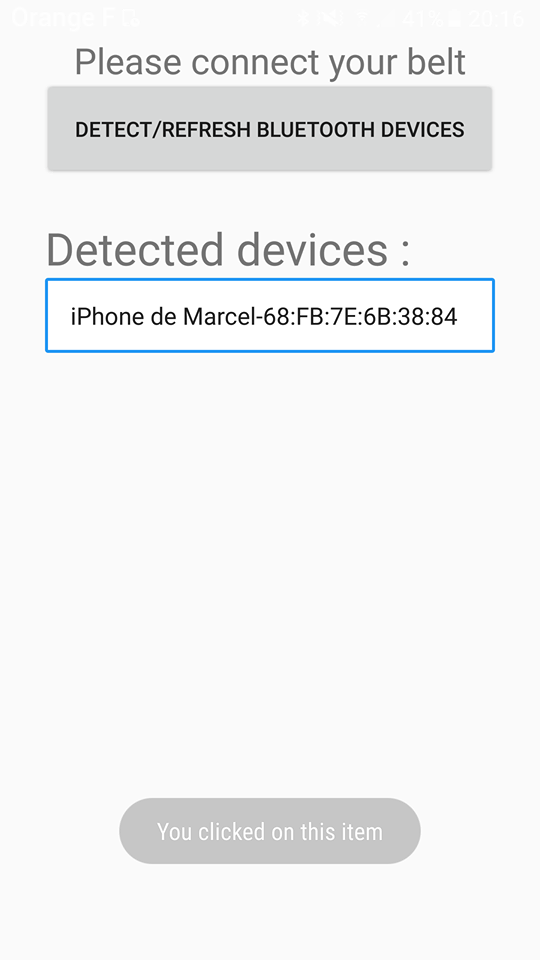
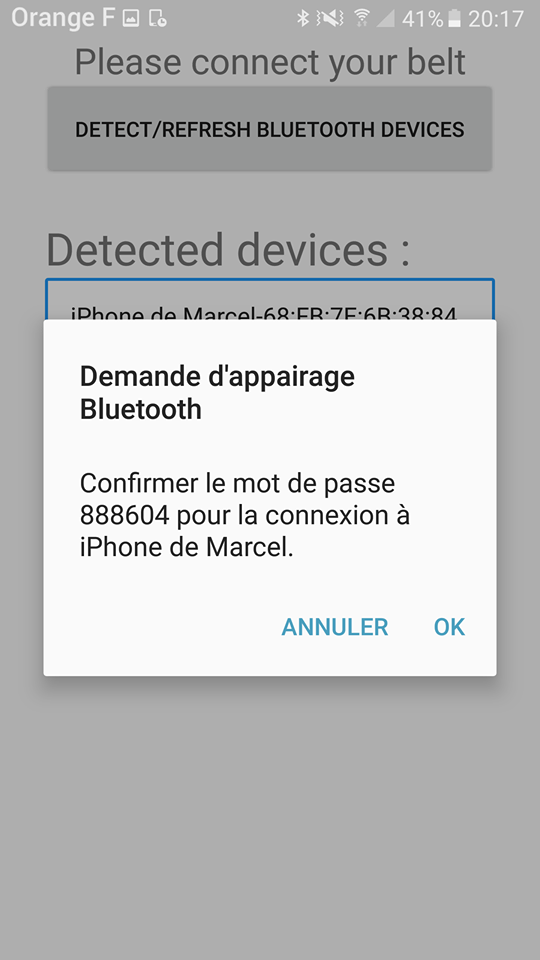
**Entrée  :** Rythme cardiaque humain

**Sortie  :** Valeurs (double) du rythme sur l'activité

**Résultats :**

**I-Connexion Bluetooth :**

La connexion bluetooth pour détecter la ceinture cardiaque est établie : l’application peut se connecter sur bluetooth et détecter les appareils disponibles. Ci-joints deux captures d’écran pour démontrer que cette détection a bien lieu sans problèmes :

**II-Code de coaching à partir du rythme cardiaque :**

Une classe java a bien été implémentée sur AndroidStudio pour jouer le rôle du coach sportif qui a deux fonctions :

1-En cas de rythme au repos : la mesure du rythme cardiaque permet de suivre le développement du rythme au repos qui informe sur l’état d’amélioration cardiovasculaire chez le coureur : un rythme qui diminue serait la preuve à long terme d’une amélioration de l’adaptation cardiovasculaire du corps du coureur. Ci-joint la fonction qui en traite.

**public** String messageOneMinuteAfterRun()  
 {  
 **if**(getFC()-getFrequencyAfterRun()<20)  
 {  
 **return "Still in the normal performance. Pay greater efforts to be an athlete. Good luck!"**;  
 }  
 **else** {  
 **if**(getFC()-getFrequencyAfterRun()<30)  
 {  
 **return "Good runner, congratulations! You're about to be an athlete."**;  
 }  
 **else**{  
 **return "Congratulations! You're an athlete."**;  
 }  
 }  
 }  
  
  
 2-En cas de course : le coach s’assure que le rythme cardiaque du coureur est inférieur à la fréquence critique de course : cette alerte permettrait d’améliorer le rythme du coureur.

**public** String messageDuringRun()  
 {  
 **if** (goodHeartBeatFrequency())  
 {  
 **return "Good job, you're running in an accurate way !"**;  
 }  
  
 **else** {  
 **return "High heartBeat. You'd rather run a bit lower to enhance your endurance! "**;  
  
 }  
  
 }

**III-Connexion avec la ceinture cardiaque :**

Le test de connexion avec la ceinture cardiaque a par contre échoué, à cause d’une inadaptation des bibliothèques avec le processeur de la carte arduino de la ceinture cardiaque.

Code testé :

include <SoftwareSerial.h>

int bluetoothTx = 2; // TX-O pin of bluetooth mate, Arduino D2

int bluetoothRx = 3; // RX-I pin of bluetooth mate, Arduino D3

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup()

{

Serial.begin(9600); // Begin the serial monitor at 9600bps

bluetooth.begin(115200); // The Bluetooth Mate defaults to 115200bps

bluetooth.print("$"); // Print three times individually

bluetooth.print("$");

bluetooth.print("$"); // Enter command mode

delay(100); // Short delay, wait for the Mate to send back CMD

bluetooth.println("U,9600,N"); // Temporarily Change the baudrate to 9600, no parity

// 115200 can be too fast at times for NewSoftSerial to relay the data reliably

bluetooth.begin(9600); // Start bluetooth serial at 9600

}

void loop()

{

if(bluetooth.available()) // If the bluetooth sent any characters

{

Serial.print((char)bluetooth.read());

}

if(Serial.available()) // If stuff was typed in the serial monitor

{

bluetooth.print((char)Serial.read());

}

}

Message d’erreur :

Librairie uniquement compatible avec des processeurs de plus de 50MHz.

Problème  :

Problèmes de compatibilité au niveau des bibliothèques à importer, ça n’était pas compatibles avec la carte et son processeur.

**TESTS DE FONCTIONALITES :**

**II-1 Test tempo sur Android :**

**Modules testés :** Audio Android

**Objectif :** Tester que les musiques s’adaptent aux rythmes différents

**Date :** Samedi 23/04/17 21h00-21h40

Android

Audio

Tempo+chemin Chemin Lecture

**Entrée  :** Tempo + chemin absolu

**Sortie  :** Lecture sur Android

**Résultats :**

Après la discussion avec les responsables du module Android. Nous avons constaté que le mode de fonctionnement n’est pas comme ça, c’est-à-dire que nous ne créons pas de nouveau fichier au courant de la course. Ce que l’application fait, c’est lire la musique originale avec un rythme qui change, lire la musique au fur et à mesure et stocker les données dans une table (buffer). Donc il n’y a pas un chemin absolu qui intervient. Ce test n’a plus de sens.

**II-2 Test spécifique de la ceinture cardiaque :**

**Modules testés :** Android (lecture de mémoire)

**Objectif :** Tester la détection de réseau bluetooth et la connexion aux réseaux disponibles

**Date :** Samedi 22/04/17 12h-13h

Android

Récupération des données

Valeur rythme Affichage appli

**Entrée  :** Essai connexion Bluetooth

**Sortie  :** Connexion réussie affichée sur l’écran

**Résultats :**

Ce test a bien réussi. On peut visualiser sur l’écran tous les réseaux disponibles, en l’occurrence celui de la ceinture cardiaque à laquelle on veut connecter notre Bluetooth. On joint ci-dessous une capture d’écran.

**II-3 Test fiabilité d’étirement :**

**Modules testés :** Tous les modules confondus

**Objectif :** Tester que les musiques s’adaptent aux rythmes différents

**Date :** 23/04/17 22h-23h

Android

Audio

Plusieurs rythmes Chemins différents musiques

**Entrée  :** Essai connexion Bluetooth

**Sortie  :** Connexion réussie affichée sur l’écran

**Résultats :**

Nous avons essayé de courir à petite distance avec l’application en accélérant ou bien marche tout doucement. Nous constatons que le changement du rythme de la musique est bien fonctionnel, malgré le petit décalage temporel et la sensibilité un peu trop forte. Globalement, la fonction audio est bien fiable.

**II-4 Test Bluetooth :**

**Modules testés :** Récupération des données /Android

**Objectif :** Tester que les données relatives au rythme cardiaque sont bien affichées sur l’application Android.

**Date :** Samedi 22/04/17 à 12 h- Dimanche 23/04/17 à 16h

Récupération des données

Android

Valeur rythme Affichage appli

**Entrée  :** Données relatives aux activités cardiaques

**Sortie  :** Résultats vus à l’écran

**Résultats :**

L’implémentation de la valeur du rythme cardiaque sur l’application donne une valeur aléatoire, ce qui prouve que pour le module Android, il suffit de remplacer la valeur aléatoire par la valeur exacte du rythme cardiaque.

Au niveau de la ceinture cardiaque, les résultats ont échoué à cause d’une incompatibilité des bibliothèques avec le processeur de la carte Arduino.

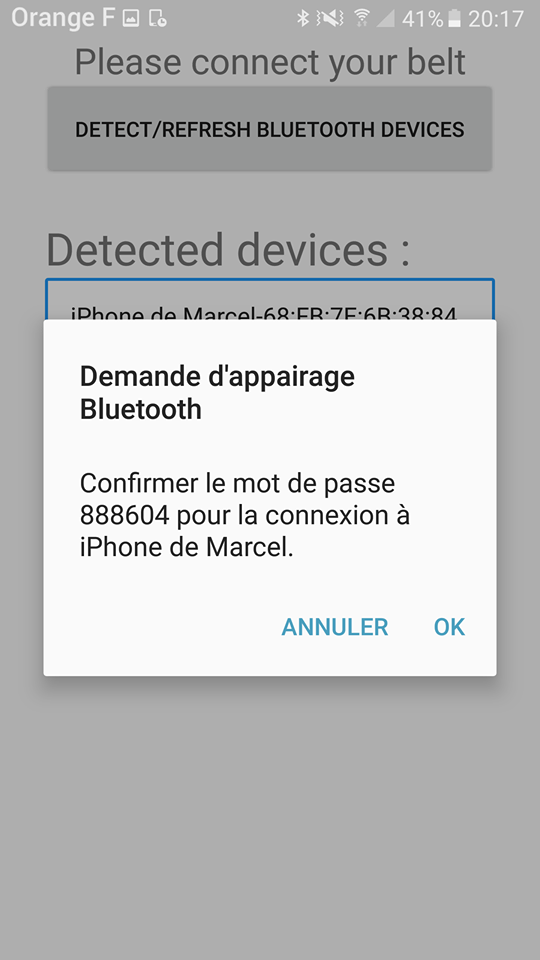
Message d’erreur :

Librairie uniquement compatible avec des processeurs de plus de 50MHz.

Problème  :

Problèmes de compatibilité au niveau des bibliothèques à importer, ça n’était pas compatibles avec la carte et son processeur.

Détails de ce test de fonctionnalités explicités dans le test I-3 , partie résultats III.



Les détails de ce test sont explicités au I-3, I-Bluetooth.