Intelligent Consultation Assistant

Documentation

End of term project by Antonin LENFANT and Miguel NAVARRO REINOSO

# Goal of the project

Miguel, go for it! :D

# Prerequisites

In order to be able to build and deploy the application, multiple prerequisites are needed.

## Android ADT Bundle

The [Android ADT Bundle](http://developer.android.com/sdk/index.html) is a pack of applications for Windows, Linux and OSX containing all the development tools needed for Android applications development :

* Eclipse : The most popular open-source Integrated Development Environment, and the only one officially supported by the SDK
* ADT plugin : The Android development plugin for Eclipse
* Android SDK Tools : The SDK tools, containing among others the SDK updater and the Android emulator
* Android Platform-tools : Tools used to communicate with the Android Operating System
* The latest Android platform : The latest version of the Androis Operating System
* The latest Android system image for the emulator : An image of this latest version usable in the emulator

## Github (or Git)

Install [Github for Windows](http://windows.github.com/), [OSX](http://mac.github.com/) or [Eclipse (Linux-compatible)](http://eclipse.github.com/) in order to download the sources directly from the Github repository.

You can also directly install the git package on Linux or OSX and retrieve these using the command-line.

## Android Device

Although you can test the application on the Android emulator, it is better to directly use a real Android device, such as a phone or tablet, for this purpose.

Indeed, it will not only be faster but it will also better reproduce the usability of the final application.

The Android device must at least use the version 2.2 of the Operating System.

## Android Emulator

If you don’t have an Android device at your disposal, you can use the emulator but it is best to download the Intel Android image using the SDK Updater and install the Intel HAX package located in [Android SDK Root]\extras\intel\Hardware\_Accelerated\_Execution\_Manager to enable hardware accelerated execution, which will give you greater speeds when using the emulator.

# Application Folders

When opening the application source code, you will find multiple folders containing different elements.

## Doc

Contains the various documentation files created during the application’s development, including this one.

It also contains the documentation for the previous version of the application, in the “first version” folder.

## IMCIapp

The main project of the application, it is an Eclipse project.

## IMCIappTest

The testing project for the application, it is also an Eclipse project. Its goal is to test if the applications works successfully.

## Rec-laptop

The previous version of the application, in Ruby on Rails.

## RegisterPatients

A basic application used to add new patients to the database, it was created in order to get to know Android Development.

# Building

To build the project in order to obtain the application’s Android apk binary file, do as described below:

1. Open Eclipse
2. When asked set the project folder as the Eclipse workspace
3. Open the project called IMCIapp
4. In the Project menu, uncheck Build Automatically
5. In the same menu, click on Build Project
6. You will find the build in bin/IMCIapp.apk

# Running the project

## Emulator

To run the project using the Android emulator, do the first 3 steps of the Building process, and then follow the steps below:

1. Click the green arrow in Eclipse’s toolbar (or Run menu then click Run)
2. The first time, create a new Android device using the AVD manager. Allow it at least 512mb of ram and use Android with a version superior or equal to 2.2. In order to have a faster emulator, use an Intel Android image
3. Select the virtual machine you want to run the application on
4. A window opens, wait for Android to boot, the application will then be started

## Android Device

It is better to run the project on an actual Android device. To do so, do the first 3 steps of the Building process, and then follow the steps below:

1. Go to your device’s Settings
2. If you can’t see the developer options, enable them (sometimes it is necessary to go to About Phone and then click rapidly six times on the build number, although it might be something else depending on your device)
3. Go to the developer options, and enable USB debugging
4. Plug your Android device on your computer with an USB cable, and wait for the drivers to install (download them on the manufacturer’s website if necessary)
5. Click the green arrow in Eclipse’s toolbar (or Run menu then click Run)
6. Select your device in the list of running Android devices

# Deploy the project

First, you need the packaged application in its apk format that you obtain after following the “Building” part of the documentation.

You can then send the application’s file to the target device(s) using various ways, for instance:

* Copying the file to the target device using an USB cable, and using a File Manager to open it
* Sending the file by e-mail or using the cloud (Dropbox for instance)
* Downloading the file using the target device’s internet browser

When running the file, you might have to enable the installation of applications from other sources if it wasn’t enabled on your device. A window will warn you of the problem and take you to the location where you can enable this setting, do so and open the file again.

The application should now run on your device.

# Testing the project

In order to ensure the smooth running of the project, we have developed tests that check various important parts of it and will report if everything works according to plan or if there is a problem.

To test the application yourself, you have to use the IMCIappTest project using the following steps:

1. Open Eclipse
2. When asked set the project folder as the Eclipse workspace
3. Open the project called IMCIappTest
4. Click the green arrow in Eclipse’s toolbar (or Run menu then click Run)
5. In the window that appears, choose to run the project as an Android JUnit Test
6. Wait for the tests to run
7. A new window called JUnit will open on the left, detailing every test that was run and if it was successful or not
8. In case of a failed test, clicking on it will show where the problem occurred, and the Failure Trace window below the JUnit results will display the exact stack trace