Assignment 1

Start:	04 October 2010
End:	15 October 2010

Objectives

The aim of this assignment is to familiarise yourself with the Android development process. You will start by exploring app development in conjunction with the Sensor API. After that you will go on to develop an application to "secure" an Android device against unauthorised usage. When the device is armed, movements should be registered. After a certain amount of time, the phone should react to the movements by raising an alarm (e.g., by playing a sound file or sending a silent notification). You should create an `Activity` to control the sensitivity of the alarm and a `Service` to deal with the output from the sensors. Your program must use the Android 2.2 API and run on the HTC Desire devices supplied by the department.

Besides implementing the application, you are required to produce a document describing your work including architecture, design choices and any problems you may have encountered (**one page max**). More details are available in the *Deliverables* section.

1 Sensing with Android

Every Android application must provide an `Activity` to serve as an interface for user-interaction. In this part of the exercise you should enable the user to access sensors and actuators through a simple user interface.

What to do

- Download and install the Android SDK from `http://developer.android.com/sdk`. You should follow the guidelines outlined in the slides and/or the Android website and install at least the components necessary to develop for the Android 2.2 SDK.
- Create a new Android project called `VS_Assignment_1_1`. Create a first `Activity` called `SensorsMain` and set the package name to `ch.ethz.inf.vs.android.g<group_no>.<ethz_login>.a1`.
- In this `Activity`, design a UI to list all available sensors on the HTC Desire. The sensors should be contained in a `ListView` which automatically resizes with different input sizes. **Hint:** You can retrieve an array of all available sensors by calling the `getSensorList(Sensor.TYPE_ALL)` method on a `SensorManager` object.
- Now create a second `Activity` called `SensorsDetail`. When the user highlights a sensor in the `ListView`, `SensorsDetail` should be started through an `Intent`. The `Intent` should carry information about the sensor. In `SensorsDetail` you should then have another `ListView`, displaying the readings for this particular sensor.
- Finally add a `Button` below the `ListView` in `SensorsMain`. This `Button` should start another `Activity` called `ActuatorsMain`. You should implement `ActuatorsMain` with two buttons and add capabilities to play a sound file and to activate the vibration. For the latter you should offer the user a `SeekBar` to control the duration.

2 The Anti-Theft Alarm

As described in the introduction, the goal of this assignment is to develop an application that provides some sort of protection against unauthorised usage. An `Activity` should control settings such as the timeout after which an alarm is raised. Please note that the current version of Android 2.2 on the HTC Desire cannot read sensor values when in *standby* mode. For now, we will ignore this limitation.

What to do

- Create a new project called `VS_Assignment_1_2` with an `Activity` called `AntiTheftMain`. The package name should still be `ch.ethz.inf.vs.android.g<group_no>.<ethz_login>.a1`.
- The `Activity` will need some means to start and stop the background process running the alarm logic. We suggest you to use a `Button` to toggle the state of the alarm.
- Create a `Service` called `AntiTheftService`. It should run in the background and host the alarm logic. Your `Service` should post an *ongoing* notification which cannot be cleared by the user. This notification should only disappear when the `Service` is shut down. **Hint:** `Notification.FLAG_ONGOING_EVENT` and `Notification.FLAG_NO_CLEAR` may be worth a look. This notification is to enable resuming of the `AntiTheftMain` activity that monitors the state of the `Service`.
- Design and implement the sensor logic needed to trigger the alarm. This should all be done inside the `Service`. Which sensor you use for this is up to you, but we suggest you to use the *accelerometer* or the *orientation* sensor. Your logic should recognise a *deliberate* movement (which we will arbitrarily define as a significant change in sensor readings for a period $\Delta_m > 5sec$). Accidental movements ($\Delta_m < 5sec$) should not cause an alarm.
- The user should have a certain period of time (Δ_t) during which she can still disarm the device. This should be done through a notification in the notification bar. You should enable the user to set Δ_t directly in the `Activity`. This information could be provided by a `SeekBar` for example and will have to be propagated from the `Activity` to the `Service`.
- When Δ_t has elapsed, the phone should ring an alarm (i.e., play a sound file). The user should still be able to disarm the device and stop the alarm using the notification mentioned above.

3 Enhancements (assessed)

As noted in the previous exercise, the current implementation does not provide any security when the device is in *standby* mode. This is somewhat a Catch-22 since we would need to lock the device to provide adequate security against unauthorised users disabling the alarm. One possible solution is to disable standby while the service is running and to *mock up* a separate login screen. Another problem is that the alarm sound can be easily suppressed by connecting headphones. Both problems could be solved by utilising the GPS chip on the device. GPS data is collected even in standby and it could also be used to provide the owner an idea of the location of the device by the means of a silent alarm (i.e., a text or email message). Additionally, a `BroadcastReceiver` could be installed to deal with a headset being plugged in. In this final section of the assignment, we would like to see you tackle one of these problems and come up with a creative solution.

However, you may regard the above as suggestions, only. We are interested in any possible enhancements you may come up with! Since this last part is open-ended, any work done here is especially useful to those seeking the highest grade.

Deliverables

The following two deliverables have to be submitted by **09:00am on October 15**:

1. **report.[txt,doc,docx,pdf]** As part of this assignment, you should produce a single page report. Please talk about the design and implementation of **tasks 2 and 3** and motivate any choices you have made during the process. You can include code snipplets to explain particular ideas and we encourage you to highlight bits you are especially proud of (or don't like at all). Feel free to also compare the Android platform to other devices such as Symbian or iOS if you have previously gained experience in those. If you have provided a solution for the final part of this assignment, we expect you to introduce your enhancements and evaluate their usefulness.

2. **code.zip** You should create a zip file containing the two eclipse projects (`VS_Assignment_1_1` and `VS_Assignment_1_2`) created in this assignment. The projects should have been thoroughly tested on the HTC Desire and the Android emulator. The enhancements should be contained in `VS_Assignment_1_2` and be clearly marked in the code. Please use UTF-8 encoding for your documents and avoid special characters like umlauts if you can.

Marks

The distribution of marks is as follows (partial solutions will be marked individually):

- First task and report: 4.0
- First task, Second task and report: 5.0
- All three tasks and report: up to 6.0

The report is necessary to archieve a pass grade. You don't necessarily have to finish the second task in order to get a 4.0 on this assignment, but you should at least report on problems encountered along the way (i.e., if things do not work, tell us about the challenges you have encountered). *Code that does not compile will result in a deduction of marks!*

Submission

Report and code must be uploaded through:

`https://www.vs.inf.ethz.ch/edu/vs/submissions/`

The submission script will not allow you to submit any part of this exercise after the deadline. However, you can re-submit as many times as you like until **09:00am on October 15**.