



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ  
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ  
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF INTELLIGENT SYSTEMS

# VNÍMANIE KONTEXTU PRE MOBILNÉ TELEFÓNY

CONTEXT AWARENESS FOR MOBILE PHONES

BAKALÁŘSKÁ PRÁCE  
BACHELOR'S THESIS

AUTOR PRÁCE  
AUTHOR

LADISLAV SZÁNTO

VEDOUCÍ PRÁCE  
SUPERVISOR

Ing. JAROSLAV ROZMAN, Ph.D.

BRNO 2013

## **Abstrakt**

Táto práce má dva ciele. Prvním je pojdénávaní o context awareness dnešných mobilních telefonů a o možnostech jak by mohla byt context awareness využita v různých odvetvích biznisu. Druhý cíl je více praktický a pozestáva z implementace aplikace na jednom specifickem mobilním telefonů. Tahle aplikace by mela být schopná získavat různá data ze senzorů a ne-senzorů na mobilním telefonů. Také by mela do isté míry být schopná rozlišovat medzi jednotlivými kontextami.

## **Abstract**

This bachelor thesis has two goals. First is to talk about context awareness of nowadays mobile phones and about possibilities, how can context awareness be used in various branches of business. Second goal is more practical and it consists of implementing an application on one specific mobile phone. This application should be able to collect various data from sensors and non-sensors present on the mobile phone. It should also be able to differentiate between some different contexts.

## **Klíčová slova**

Vnímanie kontextu pre mobilné telefóny, Context Awareness, Senzory na mobile, Senzory na androide

## **Keywords**

Context Awareness for Mobile Phones , Context Awareness, Sensors on mobile phones, Sensors on android

## **Citace**

Ladislav Szánto: Context Awareness for Mobile Phones, bakalářská práce, Brno, FIT VUT v Brně, 2013

# Context Awareness for Mobile Phones

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jaroslava Rozman, Ph.D. Další informace mi poskytla Dipl.-Inform. Julia Dauwe. Uvedl sem všechny literární prameny a publikace ze kterých sem čerpal.

.....

Ladislav Szánto

July 19, 2013

© Ladislav Szánto, 2013.

*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.*

# Abbreviations

GPS - Global Positioning System

RSSI - Received signal strength indication

USB - Universal Serial Bus

CPU - Central Processing Unit

RAM - Random Access Memory

BYOD - Bring your own device



# Contents

<b>Introduction</b>	<b>1</b>
<b>1 Basics</b>	<b>4</b>
1.1 Hardware- and Software-based Sensors . . . . .	5
1.2 Non-Sensors . . . . .	9
1.3 Android programming . . . . .	14
<b>2 Related Work</b>	<b>16</b>
2.1 Context and Context Awareness . . . . .	16
2.2 Examples for Context and Context Awareness . . . . .	18
<b>3 Analysis</b>	<b>19</b>
3.1 Context Definition . . . . .	19
3.2 Collected Data . . . . .	24
3.3 Example situation . . . . .	29
<b>4 Implementation</b>	<b>32</b>
4.1 Sensors . . . . .	32
4.2 Non-sensor data . . . . .	35
<b>Conclusion</b>	<b>44</b>
<b>List of Figures</b>	<b>46</b>
<b>Listings</b>	<b>47</b>
<b>bibliography</b>	<b>48</b>

# Introduction

This thesis will be dealing with context and context awareness for mobile phones. For development and testing purposes it was decided that android platform will be used. This decision was based on the popularity of android, as can be seen in this table 0.1.

<b>Operating System</b>	<b>Q4 2012 Units</b>	<b>Q4 2012 Market Share</b>
Android	144,720	69.7%
iOS	43,457	20.9%
Symbian	2,569	1.2%
RIM	7,333	3.5%
Bada	2,684	1.3%
Microsoft	6,186	3%
Others	713	0.3%
Total	207662	100.0%

Table 0.1: Market shares of various mobile phone operating systems  
Source: Gartner (February 2013), [4]

The reference phone on which the application will be developed and tested is Samsung Galaxy S II. In the course of this thesis phrase "target mobile phone" will always refer to the mentioned Samsung Galaxy S II.

## Problem definition and motivation

The problem this thesis is dealing with can be formulated as, to what extent can a mobile phone be aware of its surroundings, what data it can collect from sensors and other non-sensors, and how can all these data be used.

Being able to use this information effectively can open up a lot of new ways how smart-phones can prove to be useful for potential customers. First the pervasive and augmented reality games can be mentioned, both of which have interest in making games that are context aware. Second group of potentially interested customers would be shopkeepers and shopping centers, as well as other businesses as airports etc. Using context awareness and creating relatively cheap infrastructure to support context awareness in their shops/places can provide customers with better accessibility to the products they want to buy, as well as allowing companies to better advertise their products. For example customer can use his mobile phone to make a list of things he wants to buy in a supermarket. Context aware application in cooperation with the right infrastructure in the supermarket will create a route that the user can take in the shop to efficiently find all the things he wants to buy. There can also be other applications using context awareness, for example museum or city guide application showing places of interest to the user and also offering navigation to them. There are also many other uses for this, for example context aware applications finding the right buses the user needs to take to get somewhere or applications that for example turn the user's mobile phone to silent mode when he is in a meeting room or studying at school.

## **Outline of the work**

**Chapter 1** Basics - this chapter deals with basic definitions of context and context awareness. It is explained what is context, how can a mobile application be context aware and what sensors are present on mobile phones in the present and what data it is possible to obtain using those sensors. There is also a basic introduction to the development of android applications.

**Chapter 2** Related work - Other definitions of context. Use of context awareness in nowadays mobile applications and examples of pervasive games.

**Chapter 3** Analysis and Design of this work's context - this chapter focuses on how to use the data one can collect. Various contexts are defined in this chapter with definite boundaries. It is explained how the application is made context aware by monitoring the data from sensors and depending on these data it can differentiate between various possible contexts.

**Chapter 4** Implementation - this chapter consists of code samples from the application, explaining the thought process behind those and discussing other possible options of their implementation.

At the end there will be a conclusion as well as an outlook on whole thesis.

# 1 Basics

This chapter deals with all the basic information needed for understanding how the application developed for this bachelor thesis works and why was it designed that way. First there will be a definition of context, then something about the sensors present on mobile phones these days (using the target phone as an example) and at the end some information about basic programming knowledge for developing android applications.

”Context is a term that is commonly used in research on the social use of language. Its use reflects the widespread realization amongst language researchers that the meanings that words or texts have for listeners or readers may be dependent on situational factors, such as the other words that surround them, the physical setting in which words are uttered, gestures and other non-linguistic signs which accompany speech, the history of the relationship between a speaker and listener, and so on.” [8] Context awareness is the ability of being able to determine the current context.

While this is not the context this thesis wants to define, it is from here that the definition of the context of this thesis is derived. Context in the way it is perceived in this thesis refers to the surroundings of a mobile phone. Not only the physical surroundings but also other factors like acceleration of the mobile phone, its rotation etc. Context awareness in this sense is then the ability to perceive the surroundings of the mobile phone which create such context and the ability to act upon this knowledge. There are two types of information that can be gathered about the surroundings of the mobile phone. They are called location and context awareness and they complement each other. The first one is all about the location of the device, like getting the GPS coordinates of the device. This can be accomplished in many different ways. On the other hand the latter is more complex and there are many things the mobile device can monitor. Nowadays mobile phones still do not allow being completely context aware. A mobile phone is not able to directly get information about all the things and people in its surrounding without external help. It is not able to recognize faces of nearby people without being pointed at them. It can not check if the user is playing with his computer at home

or if he is sleeping. This will not be possible even in the near future, but they do offer a variety of sensors thus enabling mobile phones to differentiate between number of various contexts.

First all available sensors on the target mobile phone will be shown and explained along with the data one is able to collect from them.

## 1.1 Hardware- and Software-based Sensors

Most Android devices these days have built-in sensors that measure orientation, motion and sometimes various environmental conditions [2]. These sensors provide raw data that can be used for determining surrounding environment that creates the context. Some of the sensors are hardware-based and other are software-based. Hardware-based sensors are physical components built into a handset or tablet device. They derive their data by directly measuring specific environmental properties, such as acceleration, angular change or geomagnetic field strength. Software-based sensors are not physical devices, although they mimic hardware-based sensors and are sometimes called virtual sensors or synthetic sensors. The linear acceleration sensor and the gravity sensor are examples of software-based sensors. By knowing the rotation vector of a device, accelerometer data can be separated into gravity and linear acceleration. The android platform supports three broad categories of sensors:

1. Motion sensors
2. Environmental sensors
3. Position sensors

The first category of sensors, the motion sensors, measure acceleration and rotational forces along the three main axes. In this category can be found sensors like accelerometers, gravity sensors, gyroscopes and rotational vector sensors. Second category contains sensors that measure environmental parameters, for example illumination, humidity or ambient air temperature and pressure. This category includes barometers, photometers and thermometers.

The last category consist of position sensors with which one can get the physical position of the device. Here belong sensors like orientation sensors and magnetometers. This category is the most important one for devices that aim to be context aware. One is able to get position pf the mobile phone with it and using this

information and the internet, one is able to get important information about one's surroundings and therefore context.

The motion sensor data are defined using three axes. The next picture 1.1 shows where these axes are and it also shows their orientation. As it can be seen on the picture the X axis is horizontal and goes from left to right, the Y axis is vertical and points from down to up and the Z axis points from under the device to the top of the screen.

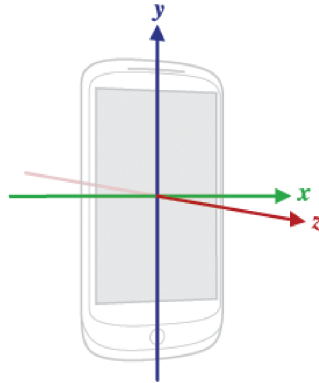


Figure 1.1: The axes of an android phone

### 1.1.1 Acceleration Sensor

In physics, acceleration is the rate at which the velocity of a body changes with time. In general, velocity and acceleration are vector quantities, with magnitude and direction, though in many cases only magnitude is considered. In this thesis, the direction of the acceleration will also be of interest. On the target phone there is accelerometer with which one can get the acceleration of the device along all three axes. Its name is K3DH Acceleration Sensor. This accelerometer measures acceleration force that is applied to the device. The values in which it is measured are meters per second squared. The values from the accelerometer also include the force of gravity on all of the axes. If the real acceleration of the device is of interest, the contribution of the force of gravity must be eliminated. For this purpose, there is also one other accelerometer on Samsung Galaxy S II called Linear Acceleration Sensor. This sensor unlike the first one is software based. This means that it does not acquire data from physical sensors. It uses the K3DH Acceleration sensor and the rotation vector to determine on which axes and to what extent is the number affected by the force of gravity and then eliminates it. This is the sensor that will

be used the most to evaluate the current acceleration of the device, excluding the force of gravity.

### 1.1.2 Magnetic Field Sensor

There is hardware based sensor capable of measuring the ambient earth's magnetic field, also called geomagnetic field. This sensor on the target phone is called AK8975 Magnetic field Sensor. Values that this sensor works in are micro Tesla. This can be useful for creating a compass. However there are a lot of magnetic fields present almost everywhere in present world, so this sensor is affected by a lot of noise making it kind of unreliable. There are also other ways to determine which way the device is moving, for example the GPS coordinates, although that is not so accurate. When it comes to how compass works, one can imagine the earth having a big magnet in it where its south pole is under earth's north pole and its north pole under earth's south pole. In reality the magnetic and real pole are a bit further away from each other but that does not really matter now. So if a compass is created that seeks the north geographic pole, it is a small magnet and its north side (the one that points north and is usually colored) is attracted to the south magnetic pole of the earth and that is where the north geographic pole is. Since the magnetic field sensor is present in the target mobile phone, it can be used to determine where the north pole is, and with that, it can be determined which way each of the directions, east, west, north and south is.

### 1.1.3 Light Sensor

Another hardware based sensor is the light sensor. This sensor provides information about the ambient light level (illumination) in SI lux units. This can be used for controlling the screen brightness depending on the light level present in the room. There is one more sensor that uses the ambient light level. This sensor is called proximity sensor and usually supports only two values and that is near and far. This sensor can be used during phone calls to prevent accidental touching of the screen. When person is making a telephone call he/she can accidentally touch the screen with his/her ear. To prevent this the proximity sensor can detect if some object is near to the display and black out the screen, disabling any touch actions during the phone call. Proximity sensor, unlike all the other sensors is interrupt based. This means that it only notifies the sensor listener when the value changes - from far to near or from near to far. This change is detected by monitoring



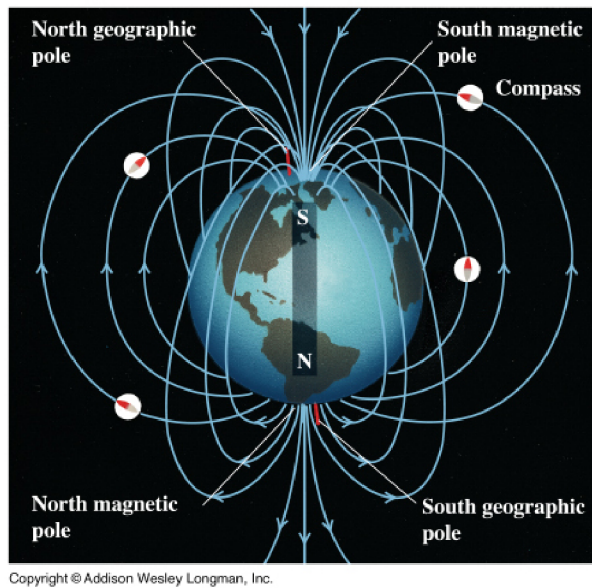


Figure 1.2: Magnetic Field of the Earth

the ambient light around the sensor. Both of these sensors use the same physical component that is present in the target mobile phone, called CM3663 Light Sensor.

### 1.1.4 Gyroscope Sensor

Gyroscope is a device for measuring or maintaining orientation. In target phone there is physical device for this called K3G Gyroscope Sensor. Gyroscope sensor measures device's rate of rotation around three physical axes. The unit used is radians per second. This can be used to detect various rotational activities like a spin or a turn. The coordinate system is the same as the one used for the acceleration sensor 1.1. Rotation values are positive in the counter-clockwise direction. That is, an observer looking from some positive location on the x,y or z axis at a device positioned on the origin would report positive rotation if the device appeared to be rotating counter clockwise.

### 1.1.5 Rotation Vector

Rotation vector can be software based or hardware based sensor. On the target phone there is a software based rotation vector. The rotation vector is a combination of an accelerometer and a magnetometer developed to determine a three-dimensional angle in which the target phone lays with respect to the Earth frame coordinates. It provides data for measuring motion or rotation detection similar to gyroscope. While gyroscope measures current rotation speed, rotation vector provides information about the current rotation state of the device. Gyroscope is a motion sensor and the rotation vector is a position sensor. The next figure 1.3 shows a coordinate system for this sensor.

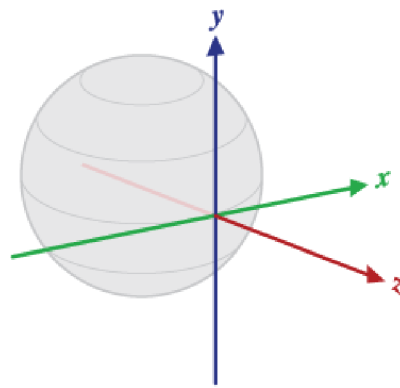


Figure 1.3: A coordinate system for the rotation vector

## 1.2 Non-Sensors

The previous section was dealing with sensors and data that can be obtained from them, those were all data about the surrounding environment and the position of a mobile phone. This section will introduce two other types of context. First one is location awareness, which is defined complementary to context awareness. Nowadays handheld devices are offering a variety of ways to get the location of a device. This thesis will deal with some of them, how they work and some advantages and disadvantages of them. GPS, WiFi and Bluetooth will be mentioned.

The other deals with sources of information about the internal state of a mobile phone. It is the state the mobile phone and its resources are in. This can sometimes be intertwined with outer context, like whether a mobile phone is plugged

into a computer or to an alternating current plug (ac). Other examples of internal state are CPU usage, ram usage, current network type or a presence and state of a SIM card.

### 1.2.1 GPS

GPS is a space-based satellite navigation system that provides location and time information in all weather conditions anywhere on or near Earth where there is an unobstructed line of sight to four or more GPS satellites. The GPS project was developed in 1973 to overcome the limitations of the previous navigation systems [1], integrating ideas from several predecessors, including a number of classified engineering design studies from the 1960s. GPS was created and realized by the U.S. Department of Defense and was originally run with 24 satellites. This system became fully operational in 1994. Since it became operational it grew widely popular and nowadays GPS receivers can be found in almost every smart phone. Here is a closer look at how the GPS works. The GPS receiver calculates its position by precisely timing the signals sent by the GPS satellites which are located high above the Earth. Each of these satellites continually transmits messages that include the time when the message was transmitted and the satellite's position at the time of message transmission. The receiver then uses the messages it received to determine the transit time of each message and computes the distance to each satellite using the speed of light. This distance can be calculated by the difference in between the times when the message was received and when it was transmitted - the time it took the message to arrive, using the speed of light as the speed that the message was traveling at and the position where the satellite was at that time. Each of these distances and satellite's location define a sphere. The receiver is on the surface of each of these spheres when the distances and the satellite' locations are correct. These distances and satellites' locations are used to compute the location of the receiver using the navigation equations. This location is then displayed, perhaps with a moving map display or a latitude and a longitude; an elevation information may be included.

The GPS provides the easiest and the quickest source of information about the user's location. No database of locations is needed, although displaying the coordinates on a map is usually very helpful. This information can be used for city guides, location aware games and in many other applications that require location awareness, but do not need it to be really precise. A good example of location awareness provided by GPS used in an Android application is Ingress. This game uses the GPS coordinates of a phone to monitor its position. There are various in-game objects the player can interact with, but he must be physically close to

the location where they are in the real world.

### 1.2.2 WiFi

Another possibility is to use WiFi networks to determine the location of a mobile phone [11]. WiFi is an abbreviation meaning Wireless Fidelity. The WiFi Alliance defines WiFi as any "wireless local area network (WLAN)" products that are based on the Institute of Electrical and Electronics Engineers' (IEEE) 802.11 standards. Since most of the modern WLANs are based on those standards, WiFi is generally used as a synonym for WLAN.

The main advantage of using WiFi to determine the location of a mobile phone is the reduced cost of deployment. WiFi are relatively cheap these days and most of the companies already have them. However, in multi-floor and dense indoor environments the performance of the system decreases because the signal reflections and dynamic network conditions may affect signal readings. Using WiFi it is possible to get an accuracy of about 3 to 5 meters.

The process of determining the location of a mobile phone is similar to the one used with bluetooth and will be explained in the following subsection. The resource that can be used for determining the distance is RSSI and the method that can be used is triangulation. There are other methods as well, this method and one more will be explained in the bluetooth section 1.2.3.

### 1.2.3 Bluetooth

The next option is using bluetooth for determining the location of a mobile phone [5]. Bluetooth is a wireless technology standard for exchanging data over short distances from fixed and mobile devices, creating personal area networks with high levels of security. It was standardized as IEEE 802.15.1, but the standard is no longer maintained.

The main advantage of using bluetooth for location awareness is the relatively cheap cost and the variable read range. Bluetooth technology is widely implemented in smart phones so the only costs are related with deploying the bluetooth terminals. The accuracy is 2 to 15 meters depending on deployment and the technique employed. First technique that is easier for implementation but does not provide such a good accuracy is just the detection of bluetooth devices. If the position of the bluetooth device is known, and it is detected that the mobile phone is in its vicinity, then the approximate location is known. The second method is

harder to implement and some mathematical computation is needed. This method uses the inquiry about nearby bluetooth devices and works with the information one can get about them, to be specific, RSSI and link quality. Both can be used to determine distance from the bluetooth device. To get the exact location of a mobile phone it needs to be in the vicinity of at least three bluetooth devices, which positions is known. Then either link quality or RSSI is processed to determine distances from each of those devices and using that information one can get the position of the mobile phone. There are different opinions about which of those one should use. For example here [7] it is stated that it is possible to determine a mobile phone's location using RSSI values and triangulation method. The way triangulation works can be seen on the next figure 1.4. Basically each

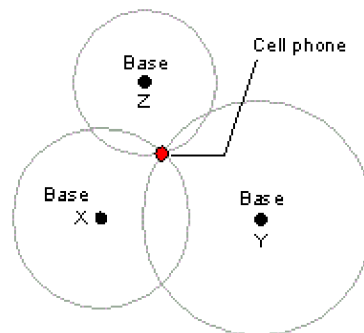


Figure 1.4: Triangulation method

device, which location is known, and the distance from it defines a circle around it, with the distance as its radius. If there are two such devices there are two intersections where the mobile phone might be. If there are three such devices in the area the exact location of the mobile phone can be calculated, plus minus a few meters. On the other hand some people argue that RSSI is not such a reliable way to determine the location. Here [5] it is stated that RSSI only shows whether the device is in a given base station power range or not. They propose the use of link quality. This value ranges from 0 to 255 and it represents the quality of the link between the two devices. The problem about using the link quality to measure distance from the device is that link quality measurements are affected by a high degree of uncertainty. Measured link quality around the device is unlikely to draw a circle like region around the device. Most of the time it is an irregular-shaped region. 1.5 Using both of these methods one might be able to get the best results. In both methods it is necessary for the place where location awareness is desired to provide a lot of stationary bluetooth devices or stations which positions is known and does not change over the course of time. This is the same when it comes to

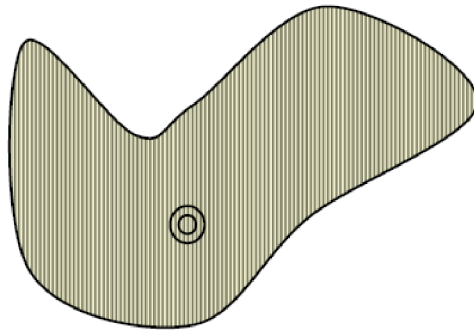


Figure 1.5: Link Quality around RF device

WiFi networks 1.2.2.

### 1.2.4 Phone status Context

Internal status can also be important for determining the activity the user is doing. For example if the user is playing something on the phone, taking photos, charging the phone or transferring data between the computer and the mobile phone. These factors will be monitored in this thesis: Sim card, its presence and current state, Network type, USB connection, CPU usage and RAM usage. All these information can provide an overview of the current state of the mobile phone. Whether it is being used or not, if it serves as a mobile phone or just a smart device and so on. First is the Sim card. Its absence can indicate that the device is not being used as a mobile phone. It is probably just used for its other features. The network type might be important for the user, to see what kind of connection he has right now. If it is good to download a lot of data at the moment of if he would be better off waiting for going to someplace with other networks. The USB connection indicates whether the phone is connected to a pc or ac. If it is connected it is possible to get some other information about the pc it is connected to. CPU usage provides an overview of the current utilization of the computation abilities. RAM usage provides information about the current RAM that is used. These two pieces of information can indicate when the user is actively using his phone. This mostly happens when the user is bored and wants to pass the time with the mobile phone. Activities such as making a call, writing an sms or writing notes or appointment into the calendar do not really indicate high usages of CPU or RAM. On the other hand, if the user is playing a game or running some other application, the levels

of CPU usage should rise above normal levels. This can be observed mostly when the user is waiting for a bus or traveling in the bus.

## **1.3 Android programming**

Following parts will be discussed in this section: Activities, Intents, Manifest, Layouts and Resources. Every android application needs at least activity, layout and manifest to work. Resources are really useful when wanting to create language independent applications and intents are used to start new activities, which are pretty common for bigger applications. By bigger applications it is meant any application that has more than just one screen. Of course more layouts can be used and switching between those, but it is common practice to use new activities for each layout so that the android back button can be used to return to the previous layout and hence the previous activity.

Activity is a single, focused thing that the application can do. Almost all activities interact with the user, so the Activity class takes care of creating a window in which user interface can be placed in with "setContentView(View)". Activities in the system are managed as an activity stack. When a new activity is started, it is placed on top of the stack, becoming the running activity and the previous activity always remain right below the new one in the stack, and will not come to the foreground again, until the new activity exits, or is paused.

On the following picture 1.6 there is a life cycle of an activity. As it can be seen there are some methods defined in the activity class. The most commonly implemented methods are onCreate() and onPause(). Intent is an object that provides runtime binding between separate components (such as two activities). It represents the application's "intent to do something". It is an abstract description of an operation to be performed. It can be used with startActivity to launch an Activity, broadcastIntent to send it to any interested BroadcastReceiver components, and startService(Intent) or bindService() to communicate with a background Service. Most often it is used to launch other activities. Every application must have an AndroidManifest.xml file (with precisely that name) in its root directory. The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's code. It contains information about the application as well as information about each activity in the application. If an application needs some special rights they need to be stated in this manifest.

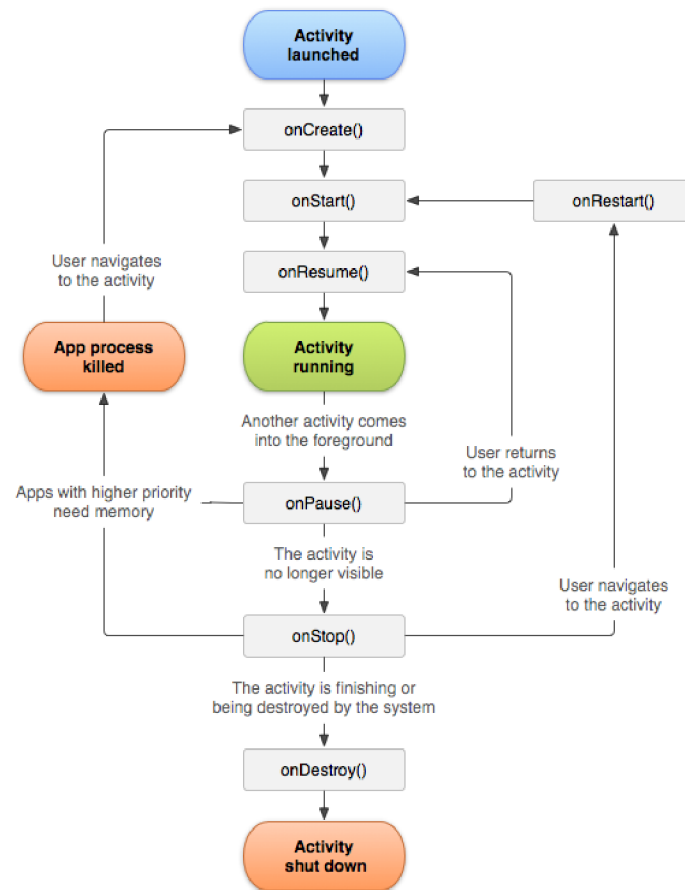


Figure 1.6: Life cycle of activity



## 2 Related Work

In this chapter other work concerning this topic will be presented. Some works and projects using context awareness and location awareness for gathering information and how they are using it will be shown. Various application and their interaction with their surroundings will be shown as well. First part starts with some other definitions of context and context awareness.

### 2.1 Context and Context Awareness

Already in the Basics chapter, there was some definition of context and how it will be perceived in this thesis. That was done to ease the understanding of the contexts of the chapter. However, in reality there is not just one definition of a context. There are a lot of definitions, some of them are similar, some are a bit further away from each other. First comes the definition of the man who introduced the term context-awareness to the world.

The term context-awareness in ubiquitous computing was introduced by Schilit [12] in year 1994 in his two articles. He states that the three most important aspects of context are :

1. Where you are
2. Who you are with
3. What resources are nearby

Then he continues by stating [12] that the context is not just about the location, because other things of interest are also mobile and changing. According to him context consists of all the information about ones surrounding like: lighting, noise level, network connectivity, communication costs, communication bandwidth and

even the social situations like: whether the user is with his manager or with a co-worker. From this, one can learn that the position is not enough when dealing with context aware applications. Context aware application must be able to get a lot of other information about the devices surrounding. It might be possible and also easier to get some of the information about physical things that are nearby, using the location and some database, informing the user about what is near that location, but some other information can only be obtained using the sensors, like for example the current noise level at that location. But the position must not be neglected as well. After all the three points he states as the most important ones, are also really important in this thesis and will be discussed further in the analysis and implementation parts.

Another definition is presented by Dey:

”Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves.” [3]

As he states after this [3], this definition makes it easier for an application developer to enumerate the context for a given application scenario, because if a piece of information can be used to characterize the situation of a participant in an interaction, then that information is context. This definition makes it easier to develop context aware applications. One should always try to get all the relevant information that he is able to obtain so he can decide on the context as precisely as possible. He also defines context-aware computing, which is shown in the following description.

”A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.” [3]

From this definition it can be seen that for an application to be context aware, it is not enough that the application can get and distinguish different contexts, it must also either display the data or act according to them. In this thesis the data will be displayed on the screen. No specific actions will be taken upon determining the current context. This thesis is focused on determining the different context types, rather than providing functionality depending on the current context.

## 2.2 Examples for Context and Context Awareness

There are a lot of example applications or situations one can use to show the different contexts and context awareness. Some of these examples will be presented in this section. There is so called BYOD policy in some companies. In these companies, their employees can bring their own device into the work. This defines some contexts that the mobile phone should be aware of. The mobile device should be able to distinguish if the user is at home or at work. The user should not be able to access the company data from anywhere else than from work. Also there might be special conditions, like for example external appointment, outside of the company. The user should be able to access the company data if needed for trips like these. Furthermore it would be good to also use contexts like, in office or in meeting room, to be able to specify where the user is, in each exact moment during the business day.

There is a game called Ingress that is able to track the position of the user, using the GPS coordinates and change the interactions, player is able to do depending on its current position. Another games also used similar systems, like using the GSM network for localization and bluetooth for detecting nearby bluetooth devices. These are all good ideas that will be used in the technical part of the thesis. Mainly the one using bluetooth for detecting nearby bluetooth devices and therefore nearby persons of interest.

There is also city tourist guide that uses context and location awareness to present a city in a new and unique way. It uses the GPS coordinates to determine where the user is, and to guide him to other sights. It also uses the motion sensors in the mobile phone to recognize, when certain gestures are being made with the phone, like a flip or a roll.

## 3 Analysis and Design of a context aware scenario for a mobile application

In this chapter data types that the application in this thesis is able to obtain will be presented and their use will be discussed. It will also be discussed about what is important and what is not for the distinguishing of context defined in this thesis. This will be the theoretical part of the bachelor thesis.

### 3.1 Context Definition

In the first instance this subsection gives the definitions for a context aware scenario for a mobile application. Here are 5 main things that will be in the center of the attention : Location - where the user is, people the user is with, resources that are nearby, time, activity that the user is doing. Each of these will have its own subsection but first, it will be explained why these 5 pieces of information are important and what they can indicate.

Location is probably the most important information. Depending on the location the activity the user is performing can be guessed, for example if he is at school, he is most probably learning or attending some lecture. To a certain extent it also defines the people the user is with and resources that might be nearby.

People the user is with are the next piece of context puzzle. Known people surrounding the user can determine the activity the user is doing. For example meeting with supervisor indicates discussing school work.

Resources that are nearby are always important. This almost always gives the impression of the activity the user is performing, if he is using those resources that is.

The time when the activity is performed is also important. Together with

people and location the activity of the user can be guessed and it can be different depending on the time factor. For example being at school with friends during the day and in the night are probably two different activities.

Activity that the user is doing will almost always be derived from the previous four information. There are some activities that can be monitored by the changes in the sensor values, for example walking, running or driving a car, but most of the time, only a guess can be made about the current activity of the user, and the first four values might help to narrow the guessing.

### **3.1.1 Location**

As it was already mentioned, location is one of the most important factors when it comes to context and location awareness. Therefore it is important to be able to track the users location most of the time, and to be as precise with it as possible. There are a lot of ways to determine users location. It can be done with the GPS, WiFi, GSM and for more precise indoor location awareness also Bluetooth. In the example application WiFi will be used most of the time, to get the location of the device. The unique IDs of WiFi networks in known places like the university will be used. This is although not a good global solution because a database of IDs of WiFi networks is needed to recognize the location. With GPS it is kind of similar, but only GPS coordinates of the place are needed and then it can be deduced what that location represents. Database of some known places is also needed, but unlike IDs of WiFi these coordinates never change. Also there are cases when WiFi network is out of commission and then it is not possible to use its ID to determine device's location, this cannot happen with the GPS coordinates. The drawback of GPS coordinates is that they do not support precise indoor location system. It is possible to do indoor location using the WiFi, but a good infrastructure of WiFi devices is needed. Using the WiFi it is possible to get a room level accuracy and that what will be presented in the example application. Better precision is usually not needed.

For the GPS localization service it is possible to call on the Google Maps application. This will provide the current latitude and longitude, as well as the map of the terrain. It is also possible to go further and track the location, sharing it with some other people using the Google Latitude application. This is location-aware mobile application that allows the user to share his location information with his friends.

For WiFi localization there are more possibilities. For the outdoor positioning a good example is Place Lab. This system relies upon existing wireless network infrastructure to provide landmarks, which can be used to determine the current

location of a device. The Place Lab software listens for wireless network base-stations, it can then look-up the coordinates of whatever networks it finds and will use triangulation to calculate its position. This system is good because it already has a wide database of known wireless networks, their unique IDs and their respective location coordinates.

To use the WiFi localization for indoor positioning, there are also applications that provide this feature. Two good examples are Meridian [9] and Navizon [10] systems. Both can be used by businesses to navigate their customers in their shops. Meridian even has its own toolkit and indoor map system for this. Other than having the WiFi transmitters an infrastructure plan of the building containing WiFi transmitters and their exact position is also needed.

Another possibility is using the bluetooth. The best indoor accuracy obtainable with bluetooth was made in a case study on positioning in a castle, with 10 base stations and the accuracy achieved was better than 0.5 meters [6]. Although Bluetooth terminals are relatively cheap, the Bluetooth will be used in other way in this thesis.

Having the location, the use of this information can be discussed. In the context defined in this thesis, there will be three example locations that will be of interest. First is home location, then university in which there will be a few places : cafeteria, mensa, first lecture classroom and second lecture classroom and last will be work. The GPS coordinates for each of these locations can be used. While all of them are indoors locations, they will all be monitored by nearby wifi networks. If the target phone comes into the range of one of the networks that is in the application database, it will be considered that the user with the phone is at one of these locations. These three locations can indicate the activity the user is doing.

When the user is at home, he can engage in a wide variety of activities. Most of the time he is not busy like at work or school so, good context aware application might for example set his phone from silent mode back to normal, when he arrives at home. When at home, it is also probable that the user will plug the phone into AC or PC to recharge it. Furthermore when the phone is connected to the PC the user might update it or move photos from the phone to the PC.

When the user is at school it is highly probably that he is at a lecture or studying. Therefore context aware application might set his mobile phone to silent mode, for the duration of the stay at school.

When the user is at work, he is probably busy, but he must still be available on his phone, in case of business calls. In this location it might be more interesting to check who the user is with. If he is with his boss, or if he is at a meeting, then it might be best to turn on the silent mode as well, but this cannot be deducted just from the location. He might be in a meeting room alone, preparing for a

meeting or just checking the room. Therefore it is a good idea to also check the surroundings for other persons, but this will be discussed in the next subsection.

There might be way more locations of interest. In a good context aware application, the user should be able to create his own locations and behavior of the phone, depending on the location. These were just some examples to show the importance and usage of the location information.

### **3.1.2 Nearby People**

In this thesis the people that are around the user at some specific time are considered to be an important part of the context. Therefore a way to identify the people that are around the user is needed. This will be heavily technology presence dependent. In this thesis bluetooth and its unique ID system is used to identify the people around the user. First a database of people of interest and IDs of their bluetooth systems in their mobile phones is needed. Then it is possible to use the bluetooth API to scan for nearby devices. When a device is found, its ID can be compared with the IDs that are present in the database. If a match is found, then the name of the person can be looked up in the database, as well as any other relevant information concerning this person. The idea of this is, that there are different categories of people the user daily deals with. For example there can be friends, co-workers, his boss, family etc. By defining these categories and putting them into the database with the bluetooth IDs it is possible to either define some actions for the mobile phone, or it can just help the mobile phone to be more aware of the situation the user is in.

There are also other ways to determine who is nearby the user. For example the Google Latitude system can be used, getting the GPS location of the user's mobile phone and the GPS location of his friend's mobile phone and then checking if they are close to each other. This would require active internet connection to function properly.

### **3.1.3 Nearby Resources**

Nearby resources might also be of interest, but the capabilities of distinguishing nearby resources using the mobile phone are not that good. At least not directly. Directly it can be checked for nearby WiFi and bluetooth devices. The PC or AC can also be recognized if the mobile phone is connected to it. Unfortunately nowadays mobile phones do not have the ability to monitor nearby surroundings

or check for non-technical items. If by resources nearby restaurants, shops etc. are meant, it might be possible to get that information. However this would require active internet connection. It is possible to use the user's location and search the internet for nearby places of interest like restaurants or galleries. There are also some programs for it, like Meridian which was mentioned before. Although this needs a good infrastructure. Using the nearby resources, guess may be made about the activity the user is doing and to some point where he is. Knowing the location of the WiFi and Bluetooth devices, they can be used to determine the location. Furthermore if the mobile phone is connected to a PC, it might be possible to communicate with the PC and determine if it is a known computer. If that is the case, it could serve as another way to determine position. The mobile phone could also monitor the activity user is doing while connected to the PC.

### **3.1.4 Time**

Getting the current time is no problem, because there is API for it. For example on android, Date() or Time() can be used to get the current date and time. As mentioned earlier, time is really important, when it comes to determining what the user is doing. Depending on the time it might differ, for example if the user is at home and it is night, the user is probably sleeping. But time can also be used in other ways. A system can be created, something like a database, that will be storing all the context that is gathered with time stamps. Now if there is a recurring event, for example the user going to the school every Wednesday and will be staying there from ten o'clock to two o'clock, context aware application might offer him to specify some special behavior for this event. It will also provide the application with some extra information about the context. Also with creating this database, the application might make all the data browsable for the user, thus creating a virtual diary with locations, people and events for the user.

### **3.1.5 Activity**

Information about the activity the user is engaged in is most of the time derived from all the other information about the context. In general, it is hard to determine the activity the user is doing. Activities like eating, studying and playing are most of the time determined by the location they take place in. They are not directly detected by the mobile phone. There are some cases when the sensors might provide some idea of the activity. In all the cases the user must have the phone on



himself. In these cases the accelerometer is used. It can be determined whether the user is walking, running or traveling by car/bus or some other form of faster traveling. This information could be greatly enhanced with on-body sensors to achieve better distinguishing of different activities carried out by the user.

## **3.2 Collected Data**

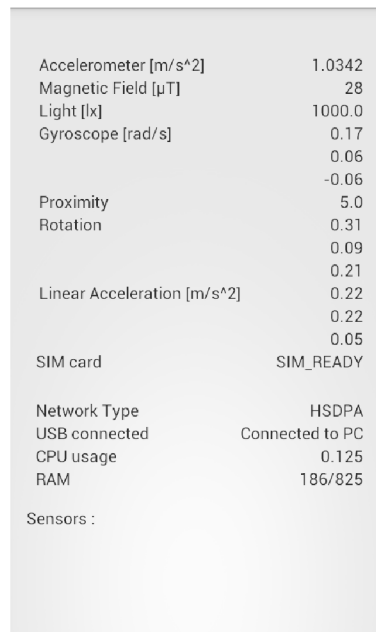
In this section, the data that can be collected using the application will be presented. Actual data will not be presented, only types of data that are monitored by the application will be shown. The following subsections start with the data that can be collected by sensors and how they can be used.

### **3.2.1 Sensor Data**

This subsection begins with a picture from the developed application showing sensors and data collected from them. On this figure 3.1 one can see sensors that are monitored and some sample data for each of them, taken at one point.

The first thing that is monitored is acceleration. The one value from accelerometer is the joint acceleration force applied on all the axes. This value is also affected by the force of gravity. The linear acceleration shows the single acceleration force applied on axis x,y and z. This is unaffected by the force of gravity. As shown on the picture the values of linear acceleration, there can be seen positive values on axis x and y that are close, but not quite equal to zero. Therefore it can be said that the mobile phone was moving when the screen-shot was taken. When the mobile phone is laying still on the table the values are almost equal to zero (plus minus 0.03). Acceleration of the phone can be used to determine if user is walking, traveling by bus/car or if he is staying still. This will be discussed later.

Next one is magnetic field sensor. As discussed this sensor and data obtained from it can be used for creating compass. There are some problems with it regarding precision and the sensor data might not provide the best possible direction. However there are other ways for creating compass and checking the direction in which the user is moving. For example the GPS can be used. On the other hand, checking the direction in which the user is moving will not tell much about the context. Well surely a guess can be made about the destination of the user depending on the place he is moving from and the direction in which he is moving, but that would require another approach as well as a database of GPS coordinates



Accelerometer [m/s <sup>2</sup> ]	1.0342
Magnetic Field [μT]	28
Light [lx]	1000.0
Gyroscope [rad/s]	0.17
	0.06
	-0.06
Proximity	5.0
Rotation	0.31
	0.09
	0.21
Linear Acceleration [m/s <sup>2</sup> ]	0.22
	0.22
	0.05
SIM card	SIM_READY
Network Type	HSDPA
USB connected	Connected to PC
CPU usage	0.125
RAM	186/825
Sensors :	

Figure 3.1: Data types collected from sensors

of places, plus it would not be accurate, if there were more places in the same direction. This guess could be misleading, but it is still a good possibility for future experiments.

The light sensor presents values of ambient light level. In theory this data can be used to determine if the user is outside or inside. However in reality there are more complications connected with it, making this very unreliable and not usable in most of the situation. First of all when the weather outside is sunny, there is a lot of light, but when it is cloudy or raining there is as much light outside as there is inside of a building. A future work in this area could be that the application could connect to the weather forecast to check the weather and behave depending on the information obtained. The seasons could also be considered, where there is more light during the summer, and the cloudy and rainy weather is more probable during the autumn. Furthermore even if this was not a problem, there is another crucial one. Most of the time, the user is walking outside with his mobile phone in his pocket. This makes the light sensor sense almost no light, regardless of the sunny weather and the application might end up determining that the user is inside, while in reality he is outside but his phone's light sensor is kept in a dark place (pocket). This mistake can be avoided using the proximity sensor. When the mobile phone is in pocket, the proximity sensor will show the value 0 and when this happens the values shown by the light sensor will not be considered.

Next comes the gyroscope and rotation vector. These two values can mainly be used for determining how the user is interacting with the phone. They can also be used to guess where the phone is. When the phone is lying still on a flat surface gyroscope should be showing 0 values on each of the x,y,z axes and the rotation vector should be 0 on x and y axes and variable value on the z axis depending on which way is the phone facing.

As it was already mentioned when speaking about the light sensor, the proximity sensor can be used to determine if something is close to the mobile phone or not. This sensor supports only values 0 and 5, 0 meaning something is close and 5 meaning nothing is close. This can be used to determine if the mobile phone is in pocket, or if the user is using the phone to make a call. There are probably no further uses for this sensor.

Presence of the sim card is next. This can be rather useful to determine the way the mobile phone is used. If there is no sim card present the mobile phone is probably used as a secondary device, maybe a calendar, a music player, or a web browser for spots with WiFi. The messages and explanations concerning them that this feature provides will be explained in the implementation chapter. For now SIM\_READY means that the sim card is present and ready to be used. It could also be possible to investigate the contacts and most called numbers to determine how the phone is used most of the time. Whether it is a mobile phone used mostly for business phone calls, if it is phone used for browsing the internet or doing some other activity, or if it is used as a camera to take pictures and upload them to the PC.

The network type might also be of interest in some kind of situations, to see what network type the phone is connected to right now etc. This information will not be used, but it is one of the information that one is able to obtain from a mobile phone, and it might be of importance to somebody else.

USB connected can provide three types of information. It says either not connected, connected to PC or connected to AC. This standalone information might provide an idea whether the user is outside or inside. It will not be taken into consideration that the user might be outside using a notebook or something similar. Furthermore it is possible to go deeper and try to get out of this information as much as possible. It can be checked what PC it is connected to, and using the information about the PC a database of known PCs and their location can be made, making a type of location awareness using just the fact of being connected to a PC.

CPU usage and RAM usage can show when the user is actively using the mobile phone. CPU usage number ranges from 0-1, with 0 being 0 percent and 1 being 100 percent of the CPU capacity being used at the moment. The RAM numbers show free MBs of the RAM at the moment slash total number of the RAM MBs

being available on the device.

This concludes the chapter of collected data from sensors and the thesis continues with other collected data.

### 3.2.2 Non-sensor data

In this subsection the other data that are accessible from the application are presented. This is mainly for the location awareness and also to check the nearby persons. There are special buttons for it with which the screens showing all the collected data from that source can be accessed. Here 3.2 are shown the representative buttons for WiFi, bluetooth and GPS. There are also buttons for some other sensor data. These buttons navigate to new screens where there are graphs plotting all the data collected from that sensor. The WiFi button leads to a new

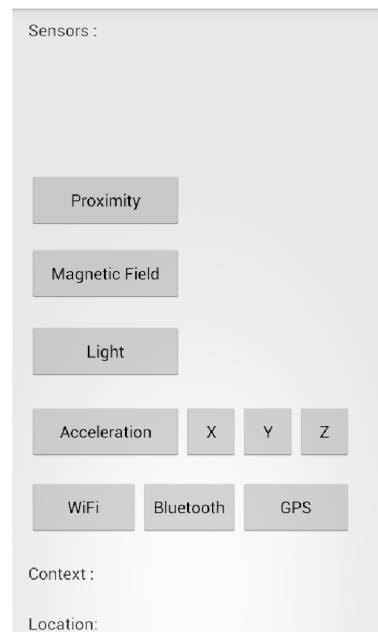


Figure 3.2: Other data that are monitored

screen where all the available networks in the area are shown. These will be used to determine the mobile phone's location if possible. It will not be shown in a graphical way, but it will be shown on the main screen in a part that will be presented soon. Note that only known WiFi and their IDs can be used to determine the mobile phone's location. In a more complex application, the user should be

able to add new networks and their locations to the database.

The bluetooth button will navigate to a new screen showing all the nearby bluetooth devices. As mentioned with the WiFi, no graphical output will be shown, but it will be possible to see the recognized bluetooth devices and the persons they belong to on the main screen. Same as with WiFi, in a more complex application, user should be able to add new bluetooth IDs to the database.

Last is the GPS button. This should navigate to the screen created with the help of google maps API. The coordinates should be presented in a graphical way on a map from the google maps database. The coordinates could also be used to determine the mobile phone's location in conjunction with the data provided by the WiFi part.

### 3.2.3 Example context awareness

With the data that have been collected and the context definition that was used, in the final part of the application the user will be presented with information about context as it is displayed on this figure 3.3. The application was made mainly for

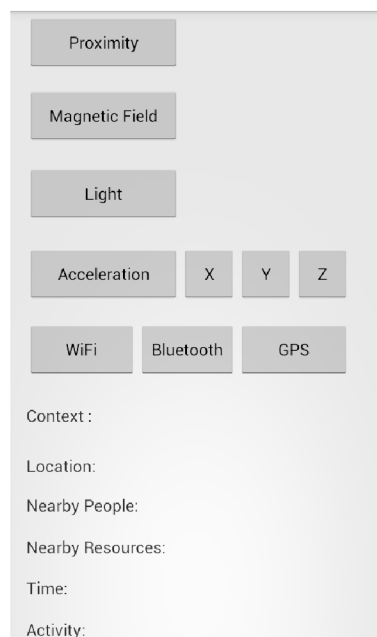


Figure 3.3: Context depending on collected data

collecting various data as a basic stepping stone for the theoretical part of this

thesis. The context distinguishing part was made as an example, not using all the techniques and resources mentioned in this thesis. It supports location awareness using WiFi, checking for nearby people using bluetooth, tracking the time and nearby resources and determines the activity depending on the location. All data that were collected can be used in numerous ways, as described in this thesis.

### **3.3 Example situation**

This is an example day of a student at the University of Siegen whose activities throughout the day are monitored by a context aware application. The application created in this thesis does not contain all the features described and proposed in this example.

During the night, the light sensor data should be really low, meaning there is no light in the room and the user is probably sleeping. When it gets closer to the morning, the light sensor data might increase by some amount in accordance to the sunlight. This might not be true, if the phone is face down on the table or if the curtains to the room are closed. During the night, the sound can be turned off, so the user is not disturbed. The sound can be turned on for example at 6 AM or when the alarm clock is set. To determine if the mobile phone is at home, first the GPS is checked for current or last known coordinates and they are compared with the home GPS coordinates. Next step is checking for known WiFi networks that are always present at home. There is also a possibility of checking for the bluetooth of the home computer. It can also be checked whether the phone is connected to the home PC.

At 6:40 the student is about to leave home. He takes the mobile phone, possibly unplugging it from PC or AC. The gyroscope changes and accelerometer changes indicate movement of the mobile phone. If the proximity changes to 0 (near) it can be deducted that the student put the mobile phone in his pocket or into a bag. This means that the light sensor cannot be used for data, until the phone is removed from the pocket/bag. The time can be checked and it can be compared with the students schedule, or calendar for any nearby events, for example if he has classes today. Match was found, the student has a class starting at 8:15.

Student is walking to the bus stop. The accelerometer data are consistent with the walking acceleration.

Student stops at the bus stop at 6:50. The GPS coordinates can be checked and compared with known bus stop locations. Match was found. Database of

buses can be checked for a bus that goes to the guessed destination - university. Another match was found and the bus comes at 6:55.

Student meets with his friend. Bluetooth device comes into range. The unique ID of the Bluetooth is in the database and it corresponds with his university friend. This reinforces the guess that he is going to the university for a lecture.

At 6:55 the bus comes and the student gets on it. Accelerometer data showed the increase in acceleration to walking and then they changed into the acceleration of a vehicle - slower acceleration.

The bus travels to the university with the student. Accelerometer shows acceleration data consistent with that of a bus. The acceleration is smaller, and there are regular bus stops. GPS signal might not be that good in a vehicle. The bus arrives at the first university building.

Student proceeds into the university. The guess is that he is going to the classroom. GPS coordinates show that the student is at the first university building. When he enters the building, GPS data cannot be used any further because of the bad indoor GPS and because of the small differences in coordinates meaning different rooms. Known WiFi are checked. Some are found indicating the cafeteria. Conclusion, student did not go to the classroom yet, he is taking food/drinks from cafeteria.

Two more friends come and join the student and they are waiting and talking in cafeteria. Guess is that the student is going to classroom after taking breakfast. Mobile phones stops showing acceleration and the WiFi in range is still the cafeteria one. Two new known Bluetooth IDs come into the range. They belong to another university students that our student spends a lot of time with. Conclusion is that the student is waiting in cafeteria with his friends.

Student goes to classroom with his friends. The acceleration changes, meaning that the student is on the move again. Guess is that he is going to classroom. Time is 8:10.

The student gets into class, he starts using notebook. Acceleration stopped, guess is that the student got into the classroom. WiFi information prove this guess. Another bluetooth device came into range, ID identified it as the students notebook. Since the class is starting soon, the sound of the mobile phone is turned off. There are no changes in acceleration, gyroscope, RAM and CPU information meaning that the student is not using his mobile phone. Redundant applications might be killed in order to conserve battery. Student goes to meet his supervisor at 9:50. The acceleration changes to the acceleration of walking. The GPS coordinates change and get closer and closer to the GPS coordinates of the 2nd university building. Since the lecture ended the sound of the mobile phone is turned back on. Student meets with his supervisor. This event is in the calendar, it is an recurring event every week. The GPS coordinates indicate the

student is at the 2nd university building. The WiFi information determine the room the student is in. It indicates the meeting room. There is a bluetooth of the students supervisor nearby. Meeting. During the meeting there are changes in the acceleration and gyroscope data. This indicates that the user is working with the phone. Proximity also shows far, meaning that the mobile phone is not in the pocket. The light sensor is showing a lot of light, meaning either the day is sunny or there is a lot of lamps in the room. There are also changes and CPU and RAM usage meaning that the student is using some applications. He is probably showing the application he created to his supervisor. Then the device goes face down on the table. The proximity shows near and rotation shows face down. This is a sign that the mobile phone will not be used soon and all applications are closed as well as the sounds are turned off.

At 12:15 the student is leaving the meeting. The sound can be turned back on. The acceleration changes to that of a walking. GPS coordinates indicate that he is getting closer to the first university building. The guess is that he is going for a lunch. When he enters the first university building there are no more updates from the GPS, the mensa is underground. The WiFi information indicate that the user is in the mensa. There is also a lot of sound nearby, which can be detected by the sound sensor.

After that, the student goes home at 13:15. The acceleration matches that of a walking. He stops at a bus stop, the database of bus stops is checked and a match is found. The buses that stop at this bus stop are checked and one should arrive at 13:20 that goes to the students home. The student gets on the bus and leaves at his home bus stop. He gets home. The GPS and WiFi information prove that he is at home. He plugs his phone into the PC. From that point on the phone is lying still on his table. When it gets closer to the night, there is less and less light. At some point the lamp in the room is turned on, but it provides less light than the sun. When the light goes to 0, the lamp was probably turned off and the student went to bed.



## 4 Implementation

This chapter focuses on the implementation part of the thesis. There are lots of code samples with explanations of the code and there is a bit of theory here and there. This should serve as a groundwork for context aware application, for either reference or usage in future applications. At the beginning this chapter will deal with implementation of the sensor data collection and their display and then it will move on to the other types of collected data. The interface of the application was shown in the previous chapter and since it was pretty basic, it will not be shown again. The manifest file will be shown as well in the end with all the permissions needed to run the application.

### 4.1 Sensors

To be able to get data from the sensors the application must have an Activity that implements `SensorEventListener`. This is done by the main activity. `SensorEventListener` defines some methods and one of this methods will be used. To be precise it has abstract methods `onAccuracyChanged()` and `onSensorChanged()`. The first one can be used if the ability to change the accuracy during the run of the application is desired. This method will not be interesting for the target application. The second one offers a way to get notifications when the values of sensors change. The created application will be heavily based on this, when it comes to the sensor data. First the main activity must be registered as a listener for sensors that are important. As this application is designed to get as much data as possible the application is registered for all the sensors available. It is done by the following code 4.1.

Listing 4.1: Registering activity as a listener for sensors

```
1  @Override
2  protected void onResume() {
3      super.onResume();
```

---

```

4  // register this class as a listener for sensors
5  sensorManager.registerListener(this,
6      sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
7      SensorManager.SENSOR_DELAY_NORMAL);
8  sensorManager.registerListener(this,
9      sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE),
10     SensorManager.SENSOR_DELAY_NORMAL);
11 sensorManager.registerListener(this,
12     sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT),
13     SensorManager.SENSOR_DELAY_NORMAL);
14 sensorManager.registerListener(this,
15     sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY),
16     SensorManager.SENSOR_DELAY_NORMAL);
17 sensorManager.registerListener(this,
18     sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD),
19     SensorManager.SENSOR_DELAY_NORMAL);
20 sensorManager.registerListener(this,
21     sensorManager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR),
22     SensorManager.SENSOR_DELAY_NORMAL);
23 sensorManager.registerListener(this,
24     sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION),
25     SensorManager.SENSOR_DELAY_NORMAL);
26 }

```

---

As it is shown on line 2. the `onResume()` method is overwritten and also the parent's `onResume()` is called on line 3. This is better than using the `onStart()` method because as it can be seen on the previous activities life cycle 1.6 figure `onResume()` is called right after `onStart()` and it is also called every time user returns to this activity.

Next the code shows how to register listeners for sensors. Specifically the application is registered as a listener for these sensor types: Accelerometer, Gyroscope, Light, Proximity, Magnetic Field, Rotation Vector, Linear Acceleration. These are the only sensors available on Samsung Galaxy S 2, the phone that is the target for the application testing.

Next piece of code 4.2 shows the implementation of `onSensorChanged()` method.

Listing 4.2: `onSensorChange()`

```

1  @Override
2  public void onSensorChanged(SensorEvent event) {
3      if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
4          getAccelerometer(event);
5      }
6      if (event.sensor.getType() == Sensor.TYPE_MAGNETIC_FIELD) {
7          getMagneticField(event);
8      }
9      if (event.sensor.getType() == Sensor.TYPE_LIGHT) {
10         getLight(event);
11     }
12     if (event.sensor.getType() == Sensor.TYPE_GYROSCOPE) {
13         if ("K3G Gyroscope Sensor".equals(event.sensor.getName()))

```

---

```

14     {
15         getGyroscope(event);
16     }
17 }
18 if (event.sensor.getType() == Sensor.TYPE_PROXIMITY) {
19     getProximity(event);
20 }
21 if (event.sensor.getType() == Sensor.TYPE_ROTATION_VECTOR) {
22     getRotation(event);
23 }
24 if (event.sensor.getType() == Sensor.TYPE_LINEAR_ACCELERATION) {
25     getLinearAcc(event);
26 }
27 }

```

---

This method is called every time when the value of any of the sensors, for which it was registered as a listener, changes. This means it needs to be determined which sensor's value did change. As one can see on line 3 this can be done using the parameter from `onSensorChange(SensorEvent event)` method. When it is determined which sensors value changed an appropriate handling method can be called, for example if the value of accelerometer changed, `getAccelerometer(event)` is called. These are methods that have been implemented manually. Before moving on to the next part, notice that on line 13. the name of the gyroscope sensor is checked. This is done because there are actually two `TYPE_GYROSCOPE` sensors on the target phone, so one of them needs to be chosen. Now comes a closer look at the implemented methods.

First comes the `getAccelerometer(event)` method.

#### Listing 4.3: `getAccelerometer()`

```

1  private void getAccelerometer(SensorEvent event) {
2      float[] values = event.values;
3      // Movement
4      float x = values[0];
5      float y = values[1];
6      float z = values[2];
7      float acelationSquareRoot = FloatMath.sqrt(x * x + y * y + z * z);
8      tx.setText(new DecimalFormat("##.####").format(accelationSquareRoot));
9      double xx = System.currentTimeMillis() - zaciatok;
10     Point p = new Point(xx, accelationSquareRoot); // Create new point
11     line4.addNewPoints(p); // Add it to our graph
12     view4.repaint();
13 }

```

---

This method is called when the accelerometer's value changes. As one can see on line 2. array of values is received. In this array the first number is new value of acceleration force applied on the x axis, second value on y axis and third value on z axis. This being accelerometer the values are affected by the force of gravity.

Therefore these values will not be of much interest, so only the total force of acceleration applied on the mobile device is displayed. This can be calculated as seen on line 7. On line 8. the output format is defined and the text is printed to a text field on the main layout. Then on lines 9 to 12 it is dealt with the graphical view. For this the AChartEngine is used, a charting library for android applications. Code for the following methods is almost identical although some do not have charts: `getMagneticField()`, `getLight()`, `getGyroscope()`, `getProximity()`, `getRotation()`, `getLinearAcc()`.

## 4.2 Non-sensor data

Getting data using the sensors was relatively easy, using the sensor manager. Next comes the need to get the other data that can be used for context awareness. The section begins with the smaller bits of data and concludes with WiFi, bluetooth and GPS.

### 4.2.1 SIM card and Network type

The sim state can be obtained using the telephony manager. It is displayed in the following code 4.4.

Listing 4.4: Getting SIM card information

```
1 TelephonyManager telMgr;
2 int simState = telMgr.getSimState();
3 switch (simState) {
4     case TelephonyManager.SIM_STATE_ABSENT:
5         t22.setText("SIM_ABSENT");
6         break;
7     case TelephonyManager.SIM_STATE_NETWORK_LOCKED:
8         t22.setText("NETWORK_LOCKED");
9         break;
10    case TelephonyManager.SIM_STATE_PIN_REQUIRED:
11        t22.setText("PIN_REQUIRED");
12        break;
13    case TelephonyManager.SIM_STATE_PUK_REQUIRED:
14        t22.setText("PUK_REQUIRED");
15        break;
16    case TelephonyManager.SIM_STATE_READY:
17        t22.setText("SIM_READY");
18        break;
19    case TelephonyManager.SIM_STATE_UNKNOWN:
20        t22.setText("UNKNOWN");
```

---

```
21         break;
22     }
```

---

This code is pretty much self-explanatory. The sim state is obtained by calling the `getSimState()` method on telephony manager. Depending on the integer result a switch is used and the output is written. Name of the constants explain what state the sim card is in, so that can be used as a text output.

Determining network type is really similar to the sim card status. The network type is obtained by this line of code 4.5.

#### Listing 4.5: Getting network type

```
1 int Network = telMgr.getNetworkType();
```

---

Then a switch is used and depending on the constant integer value the output is typed into a text field.

### 4.2.2 USB

There are more ways to determine if a mobile phone is connected to a computer. This application will be using `BatteryManager`. Here 4.6 is the code.

#### Listing 4.6: Getting USB status

```
1 IntentFilter ifilter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
2 Intent batteryStatus = this.registerReceiver(null, ifilter);
3 int chargePlug = batteryStatus.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
4 switch (chargePlug) {
5     case BatteryManager.BATTERY_PLUGGED_USB:
6         t26.setText("Connected to PC");
7         break;
8     case BatteryManager.BATTERY_PLUGGED_AC:
9         t26.setText("Connected to AC");
10        break;
11    default:
12        t26.setText("Not connected");
13        break;
14 }
```

---

On lines 1 and 2 a little trick is used to get the current battery charging status. A listener is not registered for this, just the current status is obtained. As one

can see on line 2. the target receiver is null. Now the intent batteryStatus can be used to extract data from it. On line 3. the BatterManager.EXTRA\_PLUGGED information is obtained from the intent. Depending on integer constant in this field it can be deduced whether the mobile phone is connected to a PC, an AC or neither. This information is also written into a text field on the main layout.

### 4.2.3 RAM

For getting available memory and total memory ActivityManager and MemoryInfo will be used. Here 4.7 is the code.

#### Listing 4.7: Getting RAM status

```
1 MemoryInfo mi = new MemoryInfo();
2 ActivityManager activityManager = (ActivityManager) getSystemService(
    ACTIVITY_SERVICE);
3 activityManager.getMemoryInfo(mi);
4 long availableMegs = mi.availMem / 1048576L; //
5 long totalMegs = mi.totalMem / 1048576L;
6 t30.setText(Long.toString(availableMegs)+"/"+Long.toString(totalMegs));
```

---

First a new instance of MemoryInfo class is created on the line 1. Then activity manager is received and its method getMemoryInfo() is called with the instance of MemoryInfo. After this one can access variables of the instance of MemoryInfo availMem and totalMem that was created. On lines 4 and 5 the numbers in availMem and totalMem are divided by the number 1048576. This is  $1024*1024$  and this operation means converting the numbers from bytes to megabytes. After it is all done the output is printed into a text field on the main layout.

### 4.2.4 CPU

Getting other resources was quite easy so far, because there was always an API that allowed direct access to the numbers that were of interest. That is not the case when it comes to CPU usage. Unfortunately there is no API so far to check the current CPU usage. A bit of work needs to be done with the system logs, but it is not as hard as it might seem. Here 4.8 is the code that will be used for it.

Listing 4.8: Getting current CPU usage

```

1 private float readUsage() {
2     try {
3         RandomAccessFile reader = new RandomAccessFile("/proc/stat", "r");
4         String load = reader.readLine();
5         String[] toks = load.split(" ");
6         long idle1 = Long.parseLong(toks[5]);
7         long cpu1 = Long.parseLong(toks[2]) + Long.parseLong(toks[3]) + Long.
            parseLong(toks[4])
8             + Long.parseLong(toks[6]) + Long.parseLong(toks[7]) + Long.parseLong(
                toks[8]);
9         try {
10            Thread.sleep(360);
11        } catch (Exception e) {}
12        reader.seek(0);
13        load = reader.readLine();
14        reader.close();
15        toks = load.split(" ");
16        long idle2 = Long.parseLong(toks[5]);
17        long cpu2 = Long.parseLong(toks[2]) + Long.parseLong(toks[3]) + Long.
            parseLong(toks[4])
18            + Long.parseLong(toks[6]) + Long.parseLong(toks[7]) + Long.parseLong(
                toks[8]);
19        return (float) (cpu2 - cpu1) / ((cpu2 + idle2) - (cpu1 + idle1));
20    } catch (IOException ex) {
21        ex.printStackTrace();
22    }
23    return 0;
24 }

```

As one can see on line 3. the file `/proc/stat` will be used. In this file on first line there is written "cpu" and seven numbers. First comes the explanation of what do these numbers mean. They all mean time spent on cpu by different types of processes. In this order they are : user - normal processes executing in user mode, nice - niced processes executing in user mode, system - processes executing in kernel mode, idle - cpu is free, iowait - waiting for input/output to finish, irq - servicing interrupts, softirq - servicing soft interrupts. So if this line is parsed by spaces as it is done on line 5., the first thing will be "cpu" and then the following numbers. Idle will be number 5. So the numbers on places 2,3,4,6,7,8 will be summed into cpu being used (cpu1) and number 5 as cpu not being used (idle1) (lines 6 and 7). Then the application waits for a few milliseconds and does this again getting cpu2 and idle2. Finally the current cpu usage is obtained by subtracting time cpu1 from cpu2 getting the time that cpu was used in the time the application was waiting and this number is divided by the total time that was spent while the application was waiting. This time can be obtained by adding cpu2 and idle2 together and subtracting from it the sum of cpu1 and idle1. Now the result is the cpu usage in current time on the scale from 0.00 to 1.00.



## 4.2.5 WiFi

Now come the parts that will provide important information concerning location. First comes the implementation of WiFi detection. There is a main class that manages the UI where the listener is registered and then the listener is a standalone class. First the main class is presented, and only the important part of it is shown, where the listener is registered. Here 4.9 is the code.

Listing 4.9: Registering for wifi changes

```

1 WifiManager wifi;
2 BroadcastReceiver receiver;
3 wifi = (WifiManager) getSystemService(Context.WIFI_SERVICE);
4 // List available networks
5 List<ScanResult> configs = wifi.getScanResults();
6 for (ScanResult config : configs) {
7     textStatus.append("\n\n SSID : " + config.SSID + "\n Signal strength : " + config
8         .level);
9 }
10 // Register Broadcast Receiver
11 if (receiver == null)
12     receiver = new WifiScanReceiver(this);
13 registerReceiver(receiver, new IntentFilter(WifiManager.
14     SCAN_RESULTS_AVAILABLE_ACTION));

```

First the WifiManager is obtained using the `getSystemService()` method. Then all the available networks at the time this activity is started are listed and presented on the screen. The SSID of found networks is shown as well as their signal strength. This signal strength can be used for positioning, when there are enough known WiFi access points in an area. Then on the line 12 the other class 4.10 is registered as a receiver for updates, when the scan results are available.

Listing 4.10: Wifi scan results receiver

```

1 public class WifiScanReceiver extends BroadcastReceiver {
2     WifiTester wifiDemo;
3     public WifiScanReceiver(WifiTester wifiDemo) {
4         super();
5         this.wifiDemo = wifiDemo;
6     }
7     @Override
8     public void onReceive(Context c, Intent intent) {
9         wifiDemo.textStatus.setText("");
10        List<ScanResult> configs = wifiDemo.wifi.getScanResults();
11        for (ScanResult config : configs) {
12            wifiDemo.textStatus.append("\n\n SSID : " + config.SSID + "\n Signal strength :
13                " + config.level);
14        }
15    }
16 }

```



This class does pretty similar code to the main class. It extends `BroadcastReceiver` and receives notification every time wifi scan results are available. Then the method `onReceive()` is called and the application goes through the scan results displaying important information about found wifi networks. To work with WiFi information one needs some extra permissions for the application. Here 4.11 are the two permissions that the application is using.

**Listing 4.11: Permissions for using WiFi**

```
1 <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
2 <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
```

---

First one allows applications to access information about WiFi networks and the second one allows applications to change WiFi connectivity state.

## 4.2.6 Bluetooth

As mentioned earlier, the bluetooth can be used to determine if some persons of interest are nearby or not. Bluetooth adapter will be used for this. With it one can do two types of checking for bluetooth devices. The area can be scanned for any bluetooth devices that have discovery enabled. This is to provide a general idea on what devices are nearby. However there is a possibility that the bluetooth devices have their discover-ability disabled. For that, there is another method that can be used, if the application is interested in devices that it has already communicated with. For that, querying for paired devices is really useful. This allows the application to check whether one specific device is nearby. This also bypasses the discover-ability factor. Meaning that even if the target device has discover-ability turned off, it will still be shown. For that, their unique MAC address are needed, but that is not a problem, since they are needed to determine who the person is anyway. Only the discovering new devices part is implemented, and it can be further enhanced with the querying for paired devices in the future. First the application needs to register a receiver for bluetooth notifications. The notification that will be of interest is `ACTION_FOUND`, which means that a new device was found. Here 4.12 is the receiver.

Listing 4.12: Bluetooth notification receiver

```
1 final BroadcastReceiver mReceiver = new BroadcastReceiver() {
2     @Override
3     public void onReceive(Context context, Intent intent) {
4         String action = intent.getAction();
5         // When discovery finds a device
6         if (BluetoothDevice.ACTION_FOUND.equals(action)) {
7             // Get the BluetoothDevice object from the Intent
8             BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.
9                 EXTRA_DEVICE);
10            // Show the name and address on screen
11            text.append(device.getName() + "\n" + device.getAddress());
12        }
13    };
```

Now that the receiver was defined and created, it just needs to be registered for notifications and the discovery needs to be started. That is done with the following lines 4.13.

Listing 4.13: Bluetooth scan

```
1 BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
2 IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
3 registerReceiver(mReceiver, filter);
4 text.setText("");
5 mBluetoothAdapter.startDiscovery();
```

For using bluetooth some extra permissions are also needed. Here 4.14 are the two permissions that the application is using.

Listing 4.14: Permissions for using bluetooth

```
1 <uses-permission android:name="android.permission.BLUETOOTH" />
2 <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

The first permission is needed to perform any bluetooth communication, such as requesting connection. The second one is used so a scan for nearby bluetooth devices can be performed, the first time scan for all the nearby bluetooth devices. Also if the second permission is needed, the first one must be requested as well.

## 4.2.7 GPS

The GPS will be used only to get the coordinates. The google maps API could be used, but this application was designed not to use the internet connection at all. This will be further discussed in the conclusion chapter. For GPS updates the location manager and location listener will be used. First a location listener needs to be created. That can be done with the following code 4.15.

Listing 4.15: Location listener

```

1 private class MyLocationListener implements LocationListener
2 {
3     @Override
4     public void onLocationChanged(Location loc) {
5         if (loc != null) {
6             lat.setText(Double.toString(loc.getLatitude()));
7             lon.setText(Double.toString(loc.getLongitude()));
8         }
9     }
10    @Override
11    public void onProviderEnabled(String provider) {
12    }
13    @Override
14    public void onStatusChanged(String provider, int status, Bundle extras) {
15    }
16    @Override
17    public void onProviderDisabled(String provider) {
18    }
19 }

```

As one can see only `onLocationChanged` method is used but the other three methods must be implemented as well, even if they are doing nothing in this case. The code is pretty simple, methods `getLatitude()` and `getLongitude()` are used on the location argument that was received when this method was called. Now location updates need to be requested. This can be done with this 4.16 code.

Listing 4.16: Requesting location updates

```

1 lm = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
2 locationListener = new MyLocationListener();
3 Criteria criteria = new Criteria();
4 Location location = lm.getLastKnownLocation(lm.getBestProvider(criteria, false));
5 lat.setText(Double.toString(location.getLatitude()));
6 lon.setText(Double.toString(location.getLongitude()));
7 lm.requestLocationUpdates(
8     LocationManager.GPS_PROVIDER,
9     10000,
10    10,
11    locationListener);

```

First the location manager is obtained, then an instance of the location listener is created. The last known location can be obtained with the `getLastKnownLocation()` method invoked on the location manager and then it can be displayed. After that the location updates can be requested with the following parameters. First is a provider, second is the minimal time in which the application can get these location updates in milliseconds, third is the minimal distance in meters and last one is the location listener. For this to work, some special permissions are needed as well. These two permissions are used 4.17.

**Listing 4.17: Permissions for using GPS**

```
1 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
2 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

---

The first one allows the application to get a precise location using mostly the GPS but also the WiFi and cell towers. The second one does not use the GPS, and uses only the WiFi and cell towers to get an approximate location. This can be useful when the GPS is not available.

## **Conclusion**

In the course of this thesis, most of the data one is able to get out of nowadays mobile phones were mentioned, and the focus was on showing this practically on Samsung Galaxy S II. In this thesis context was inspected from the point of view of the mobile phones and the mobile phone was made context aware to some extent in the developed application.

The reader was made familiar with the sensors and their usage on a mobile phones. It was discovered that although there are way more sensors on the mobile phone these days than there were in the past, most of these sensors help the user interact with their mobile phone better, but are not of a much use when it comes to the context awareness. On the other hand, the other way to gather data, mainly GPS, Bluetooth and WiFi play very important role in the present days. There is a lot of information they can provide and if used in conjunction with the internet, one is able to access various information about their surroundings.

In the technical part, an application that collects all of the data available from the sensors and some other non-sensor data was created. In the second part this application was improved and made context aware to some extent using the WiFi, accelerometer, time and battery status. This context awareness was made just as an example. This application was documented in the implementation part of the thesis and it was also provided with some extensive explanation of the code.

The future work in this area could follow up with either theoretical or technical improvements. The theoretical work could offer new ways to gather data and provide some ideas on how to use these, or already collected data. Some ideas are, ambient sound levels, work with the calendar and the events in it, with respect to the current time. Technical improvements include implementing the new ideas as well as the ideas already mentioned in this thesis. The GPS location awareness can be implemented along with the creation of a compass, providing better guesses about the location and the destination of the user. As it was already mentioned, a diary with places and events can be implemented, and using it the guesses about context could become way more accurate.

Great improvement for the application could be the usage of active internet

connection. This would open up a lot of new data and information the application would have access to. For example the weather forecast, requesting information about nearby places of interest, like schools, restaurants, shopping malls, as well as requesting information about nearby bus stops and the buses that frequent those.

To conclude this thesis the ability of the mobile phones to get information about their surrounding and location should be taken advantage of by large corporate organizations to make their customers life easier and to advertise and help their products sell better. Context awareness can also be taken advantage of in augmented reality and pervasive games, providing players with new and exceptional experiences. Gaming companies might take interest in this but this will probably still take some time, because as it was noted, context awareness for mobile phones is just taking its first steps towards what one day will be a part of everyday life.

However there is also another side to this topic that must be mentioned. With all these data and information that mobile phones can get in the present, one can only hope that they are not misused in any way. There would be no problem monitoring users just by using their mobile phone data. On the way to the future this amount of data will grow rapidly and one day we might find our every step being monitored, just because of the fact that we are carrying our mobile phone with us. This is not to be taken lightly.

With all that said, it will be interesting to see where context awareness concerning mobile phones and other devices will lead us in the near and far future.

# List of Figures

1.1	The axes of an android phone . . . . .	6
1.2	Magnetic Field of the Earth . . . . .	8
1.3	A coordinate system for the rotation vector . . . . .	9
1.4	Triangulation method . . . . .	12
1.5	Link Quality around RF device . . . . .	13
1.6	Life cycle of activity . . . . .	15
3.1	Data types collected from sensors . . . . .	25
3.2	Other data that are monitored . . . . .	27
3.3	Context depending on collected data . . . . .	28

# Listings

4.1	Registering activity as a listener for sensors . . . . .	32
4.2	onSensorChange() . . . . .	33
4.3	getAccelerometer() . . . . .	34
4.4	Getting SIM card information . . . . .	35
4.5	Getting network type . . . . .	36
4.6	Getting USB status . . . . .	36
4.7	Getting RAM status . . . . .	37
4.8	Getting current CPU usage . . . . .	38
4.9	Registering for wifi changes . . . . .	39
4.10	Wifi scan results receiver . . . . .	39
4.11	Permissions for using WiFi . . . . .	40
4.12	Bluetooth notification receiver . . . . .	41
4.13	Bluetooth scan . . . . .	41
4.14	Permissions for using bluetooth . . . . .	41
4.15	Location listener . . . . .	42
4.16	Requesting location updates . . . . .	42
4.17	Permissions for using GPS . . . . .	43



# Bibliography

- [1] Aeronautics and National Research Council Space Engineering Board. *The Global Positioning System: A Shared National Asset*. The National Academies Press, 1995.
- [2] Android developer. Android developer - Overview of Sensors.  
[http://developer.android.com/guide/topics/sensors/sensors\\_overview.html](http://developer.android.com/guide/topics/sensors/sensors_overview.html), 2013.  
[Online; last visited : 28.Jun.2013].
- [3] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.
- [4] Inc. USA Gartner. Gartner Says Worldwide Mobile Phone Sales Declined 1.7 Percent in 2012.  
<http://www.gartner.com/newsroom/id/2335616>, 2013.  
[Online; last visited : 4.May.2013].
- [5] Alessandro Genco. Three step bluetooth positioning. In Thomas Strang and Claudia Linnhoff-Popien, editors, *Location- and Context-Awareness*, volume 3479 of *Lecture Notes in Computer Science*, pages 52–62. Springer Berlin Heidelberg, 2005.
- [6] Alessandro Genco. Three step bluetooth positioning. In Thomas Strang and Claudia Linnhoff-Popien, editors, *Location- and Context-Awareness*, volume 3479 of *Lecture Notes in Computer Science*, pages 52–62. Springer Berlin Heidelberg, 2005.
- [7] K. Kyamakya, A. Zapater, and Z. Lue. An indoor Bluetooth-based positioning system: concept, implementation and experimental evaluation. *International*, 2003.
- [8] LabSpace. Language and literacy in a changing world.

- <http://labspace.open.ac.uk/mod/oucontent/view.php?id=445539&section=1.4>, 2013.  
[Online; last visited : 1.July.2013].
- [9] Meridian. Indoor GPS.  
<http://www.meridianapps.com/>, 2013.  
[Online; last visited : 1.July.2013].
- [10] Navizon. Accurate positioning anywhere.  
<http://www.navizon.com/>, 2013.  
[Online; last visited : 1.July.2013].
- [11] Hirokazu Satoh, Seigo Ito, and Nobuo Kawaguchi. Position estimation of wireless access point using directional antennas. In Thomas Strang and Claudia Linnhoff-Popien, editors, *Location- and Context-Awareness*, volume 3479 of *Lecture Notes in Computer Science*, pages 144–156. Springer Berlin Heidelberg, 2005.
- [12] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pages 85–90. IEEE, 1994.

# Attachement A

CD with source codes as well as an apk file of the application is attached with this thesis. The apk file can be installed on any android phone with software version 4.0 and higher, but was designed to work on Samsung Galaxy S II.