

**Android Application to test hearing impairments in adults**

by

**Junaid Malik**

This report is submitted in partial fulfilment of the requirements of the B.Sc. Honours Degree in Networking Applications & Services, of the Dublin Institute of Technology

June 2nd, 2015

Supervisor: **Dr Ray Lynch**

School of Electrical and Electronic Engineering

Year Coordinator:

Joseph Kellegher DT080B/4 Report

**Declaration**

I, the undersigned, declare that this report is entirely my own written work, except where otherwise accredited, and that it has not been submitted for a degree or award to any other university of institution.

**Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Contents

[Abstract 7](#_Toc420770639)

[Chapter 1 Introduction 8](#_Toc420770640)

[1.1 The Audiogram 11](#_Toc420770641)

[1.2 Pure Tone Audiometry 11](#_Toc420770642)

[1.3 Procedure for Pure tone audiometry 12](#_Toc420770643)

[Chapter 3 Purpose of the Project 14](#_Toc420770644)

[3.2 Goal of the Project 14](#_Toc420770645)

[3.3 Motivation 14](#_Toc420770646)

[3.4 Considerations 14](#_Toc420770647)

[3.5 Measurement 14](#_Toc420770648)

[Chapter 4 Setting up the development environment 15](#_Toc420770649)

[4.1 Development tools 15](#_Toc420770650)

[4.1 Implementing Source Control with Git 15](#_Toc420770651)

[4.2 Add github repository to source tree 18](#_Toc420770652)

[4.3 Android Activities 23](#_Toc420770653)

[4.4 Android life-cycle 23](#_Toc420770654)

[Chapter 5 Project Management 26](#_Toc420770655)

[Chapter 6 Requirements Analysis 27](#_Toc420770656)

[6.1 Use Case Diagrams 27](#_Toc420770657)

[Chapter 7 System Design 28](#_Toc420770658)

[7.2 UX/UI Design 30](#_Toc420770659)

[Chapter 8 System Implementation 32](#_Toc420770660)

[8.1 Implementing the activities 32](#_Toc420770661)

[8.2 Implementing Frequency Generator 32](#_Toc420770662)

[8.3 Implementing Audiogram with Androidplot library 32](#_Toc420770663)

[8.4 Implementing Fragments 32](#_Toc420770664)

[8.5 Implementing Rate this app plug in 32](#_Toc420770665)

[Chapter 9 Testing 33](#_Toc420770666)

[Test Structure 35](#_Toc420770667)

[Testing preconditions 36](#_Toc420770668)

[Appendices 38](#_Toc420770669)

[Appendix A Java code for Dijkstra’s Algorithm 38](#_Toc420770670)

[Appendix B User Story Mapping 39](#_Toc420770671)

[Appendix C 39](#_Toc420770672)

[Chapter 11 Bibliography 40](#_Toc420770673)

**Tables**

**Table 1‑1 Grade of impairment** 8

**Table 2‑1 Noise levels during audiometric testing: Accordance with ANSI S3.1-1991.** 12

**Figures**

[Figure 1 Diagram of ear. 10](#_Toc420770674)

[Figure 2 Typical Audiogram 11](#_Toc420770675)

[Figure 3 Typical activity flow of hearing test 13](#_Toc420770676)

[Figure 4 Git- Hub Client Application 16](#_Toc420770677)

[Figure 5 Git-Hub Bash module 16](#_Toc420770678)

[Figure 6 Checkout style using unix style 17](#_Toc420770679)

[Figure 7 Initial commit of project to git. 17](#_Toc420770680)

[Figure 8 Sprint cycle 26](#_Toc420770681)

[Figure 9 Use-Case Diagram of main menu 27](#_Toc420770682)

[Figure 10 A professional audiometer used in clinics 28](#_Toc420770683)

[Figure 11 the architecture of the apk. 28](#_Toc420770684)

[Figure 12 Quick layout design created in UX 30](#_Toc420770685)

[Figure 13 Initial design of the main menu 31](#_Toc420770686)

Acknowledgments

I would like to thank Dr Ray Lynch, my project supervisor who thought me Mobile Development providing a very important understanding of application design and architecture and also helped me and gave me guidance during implementation of the project.

Secondly I would like to thank Paula Kelly for introducing me to Java, providing me with excellent knowledge that has been crucial not only for this project but has helped me have a greater overall understanding of Software Design.

Lastly I would like to thank the School of Electronics and Communications Engineering who gave me a chance to enter this course and to receive study from great professionals.

Also would like to thank my family and friends who encouraged me and supported in my effort to develop this project.

# Abstract

Hearing is an important ability to human beings in which we perceive sounds. Sound waves occur over a wide spectrum of frequencies. The human ear is sensitive to a frequency band within that spectrum that is expressed in decibels (dB). Frequencies capable of being heard by humans are called audio or sonic. The range is typically considered to be between 20 Hz and 20,000 Hz (Hertz). Frequencies higher than audio are referred to as ultrasonic, while frequencies below audio are referred to as infrasonic. A loss of the ability to hear sound frequencies in the normal range that is specified is called hearing impairment.

# Chapter 1 Introduction

Hearing tests provides a means to diagnose an individual’s hearing impairment or deficiencies. ISO 8253-1:2010 is an international standard for PTA (Pure Tone Audiometry). This standard specifies procedures and requirements for PTA air conduction and bone conduction. Pure tone audiometry is the measurement of an individual’s hearing across a range of test frequencies using a standardized test method that specifies the procedure for determining the threshold, the range of test frequencies and presentation levels, and the way thresholds are presented in a graphical manner and including the symbols user to depict objects. We use PTA to evaluate possible hearing losses and to determine the type of hearing loss that an individual may have.

**Table 1‑1 Grade of impairment**

|  |  |  |
| --- | --- | --- |
| **Grade of impairment** | **Audiometric ISO value (500, 1000, 2000, 4000)** | **Impairment Description** |
| 0 (No impairment) | 25 dBHL | No or very slight hearing problems |
| 1 (Slight impairment) | 26-40 dBHL | Able to hear and repeat spoken words in normal voice |
| 2 (Moderate impairment | 41-60 dBHL | Able to hear but raised voice |
| 3 (Sever impairment) | 61-80 dBHL | Able to hear some words shouted |
| 4 (Profound impairment) | 81 dBHL | Unable to hear and understand a shouted voice |

The above table shows grade of impairment, level of decibels and the impairment description corresponding to each grade. [7]

Hearing loss can be defined as the amount a person’s hearing level changes as a result of some adverse influence. This means that some structure or function of the ear that is crucial to hearing has been damaged.

There are three main forms of hearing loss; Sensor neural affecting the cochlea, Conductive hearing loss affecting the ear canal and mixed hearing loss which is a combination of both sensor neural and conductive hearing loss.

The World Health Organization (WHO) defines “disabling hearing impairment in adults as a permanent unaided hearing threshold level (average for frequencies 0.5, 1, 2, 4 kHz) for the better ear of 41 dB or greater (WHO, 2001).2 In children under 15 years of age, disabling hearing impairment is defined as permanent unaided hearing threshold level (average for frequencies 0.5, 1, 2, 4 kHz) for the better ear of 31 dB or greater.” [7]

The auditory pathway includes the external ear, the middle ear and the inner ear, followed by the auditory nerve ending up in the auditory centres in the auditory cortex.

* The external ear consists of the pinna, ear canal and eardrum. Sound travels down the ear canal, through the eardrum and causing it to move or vibrate.
* The middle ear is a space behind the eardrum that contains three small bones called ossicles*.* This string of tiny bones is connected to the eardrum at one end and to the oval window at the other end which connects to the inner ear. Vibrations from the eardrum cause the ossicles to vibrate which, in turn, creates movement of the fluid in the inner ear.
* Movement of the fluid in the inner ear, or cochlea, causes changes in tiny structures called hair cells*.* This movement of the hair cells sends electric signals from the inner ear up the auditory nerve to the brain. [7]

The brain then interprets these electrical signals as sound. Figure 1 - shows the different areas of the ear described below. [11]

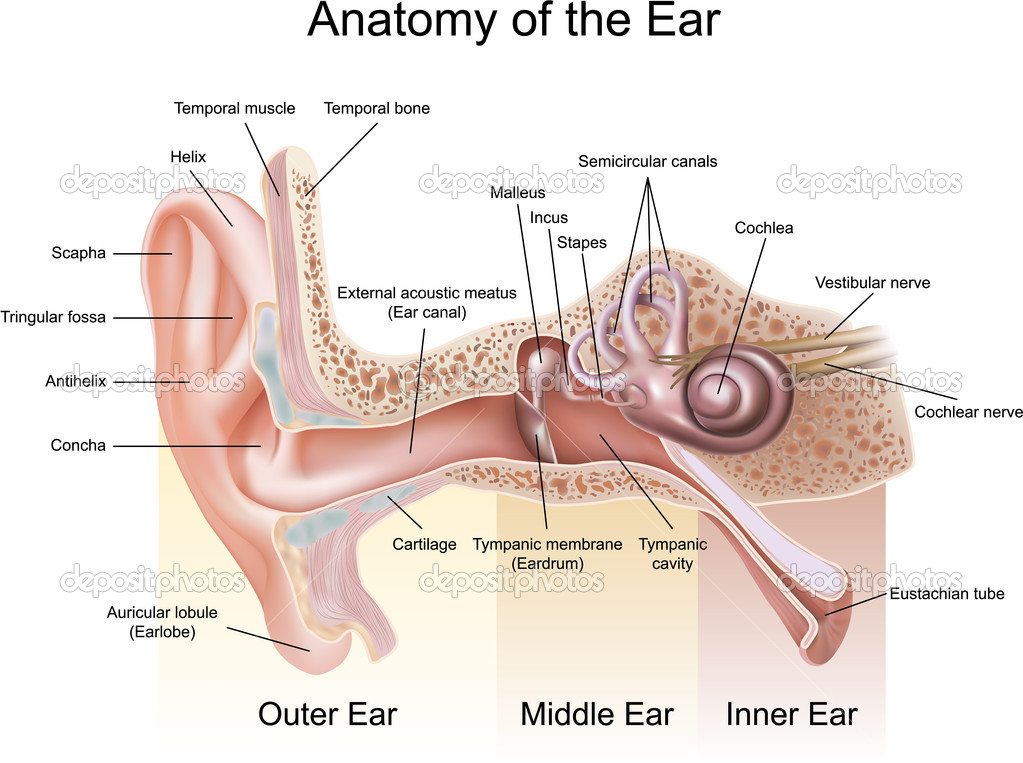


Figure 1 Diagram of ear.

From the above diagram we can see the middle ear this is where most conductive hearing loss occurs this is treated by medical treatment or through surgery. Sensor neural hearing loss occurs in the inner ear as the nerves become damaged and do not properly transmit their signals to the brain. This happens become of aging of the inner ear cells. Sensor neural hearing loss is the most common type hearing loss among adults. It is not often medically or surgically treatable. This type of hearing loss is can be treated with the help of hearing aids. When hearing loss is conductive and Sensor neural this is called mixed hearing loss.

## 1.1 How we hear sound?

Sound waves enter the outer ear and through the ear canal. The eardrum vibrates from incoming sound waves and sends those vibrations to bones in middle malleus, incus and stapes. The sound vibrations go from the air to fluid vibrations in cochlea of the inner ear. Once fluid inside cochlea vibrates waves form along the basilar membrane. When hair cells move, the bending of the hair cells causes chemicals to create an electrical signal. The auditory nerve than carries this signal to the brain which then is translated into a sound that we hear and understand. [15]

## 1.2 The Audiogram

The audiogram plots decibel (dB) values on the Y-Axis on a audiogram. If a loss of hearing is present then the graph will have elevated points with increased dB signal where the person is having difficulty hearing the tone. The more severe the hearing loss the more the dB values on the graph will going downwards. On the X-Axis we have the plotting of the frequency pure tones in hertz (Hz). An Audiogram usually plots sound on the left lowest 125Hz and increasing to 8000 KHz.

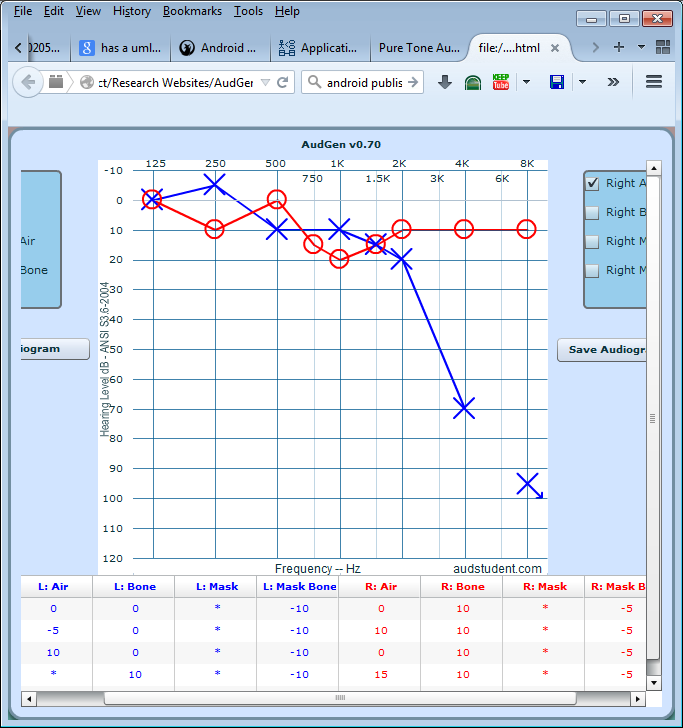


Figure 2 Typical Audiogram

## 1.3 Pure Tone Audiometry

Hearing thresholds of humans can be found by using behavioural hearing tests or physiological tests. An audiogram can be obtained using a behavioural hearing test called Audiometry. For humans the test involves different tones being played at a specific frequencies (pitch) and intensity (loudness). When this test is done with a doctor, the person hears the sound they raise their hand or press a button so that the tester knows that they have heard the sound. The lowest sound the person hears is recorded. The entire auditory system from ear canal to auditory cortex is tested. When a test is decided a method of selecting the correct method has to be chosen. There is three main methods manual, self-recording and microprocessor. I decided to choose manual audiometry method for the design of the application as it is most practical as the user decides the test environment. However this does not make the application design the most scientifically accurate. In the manual audiometry the frequency and intensity are controlled by the user. When hearing is tested the frequencies of the range of 125Hz to 8000 Hz are usually tested. One of the most important things in audiometry is the environment in which the subject is undertaking the test. There are different standards depending on the region which require different calibration.

**Table 2‑1 Noise levels during audiometric testing: Accordance with ANSI S3.1-1991.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Octave-Band Center Frequency | 500 Hz | 1000 Hz | 2000 Hz | 4000 Hz | 8000 Hz |
| ANSI S3.1-1991 (Rounded to the nearest whole decibel) | 22 | 30 | 34 | 42 | 45 |
| OSHA Table D-2 | 27 | 30 | 32 | 42 | 45 |
| OSHA Table D-1 | 49 | 40 | 47 | 57 | 62 |

## 1.4 Best Procedure for Pure tone audiometry

When beginning a test it is best to start with the right ear but the user decides what ear he wants checked. A pulsing tone that goes on and off is usually recommended; in the design of the application the user presses a button to hear the tone. The optimal test environment is quiet and free from anything that could distract user. Sometimes it is useful to wear noise-reducing earphones for best performance but regular earphones can also be used.

## 1.5 Advantage and Disadvantage of manual self-testing

The advantage of self-testing is that it can be done regularly and quickly as no specialised equipment is required. On the other hand the disadvantage is that with manual testing a user can make mistakes, press wrong button or test same ear twice. This can introduce errors into the audiogram which can make it useless from a scientific view. There are many things that can go wrong with accurate results example; too much background noise, earphones not placed properly and tester bias when a person cannot hear.



Figure 3 Typical activity flow of hearing test

From the above a typical hearing test consists of the following choices. A user selects if we can hear tone or cannot as the option is selected the result is recorded on a graph, in this case an Audiogram. Audiograms help us to explain audiometric readings.

# Chapter 3 Purpose of the Project

The purpose of this project was to develop an application that would help identify common hearing impairment by using an audiogram which would picture how a person hears. The audiogram is used to describe the hearing of a person across different frequencies. It can be used as a tool to determine amount of damage done or determine the cause.

## 3.2 Goal of the Project

The goal was to develop and application to determine hearing impairments in adults.

## 3.3 Motivation

An easy and simple application that will allow users to quickly do hearing tests for an assessment, where they get results by reading the audiogram.

## 3.4 Considerations

The problem of hearing impairment is an increasing problem across most populations of the world and is something that people should be aware of and by taking hearing tests they can be made more aware of their own level of hearing and how to best keep it healthy. In the EU it is estimated that more than 55 million people are having hearing impairments and the costs in the EU is estimated to 160 billion Euros per year [10]. According to a study, a mild hearing loss costs 2000 euros per individual each year, a moderate costs 6,600 and severe 11,000 euros.[9]

From the above studies we can see that it is a problem that needs to be carefully analysed and more research needs to be done.

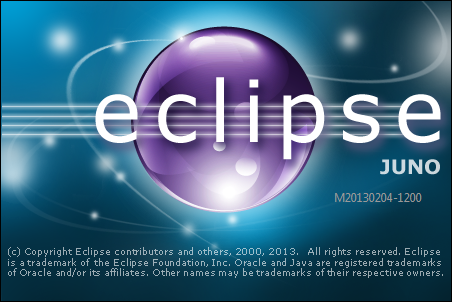
## 3.5 Measurement

The advantage brought to users through this application is that it provides free and quick self-test ability. Where-as many Android applications or web applications require connection to the internet using databases, this current project does all processing locally on the users device.

# Chapter 4 Setting up the development environment

## 4.1 Development tools

The integrated development environment (IDE) I will be using is Eclipse Juno, which is used to develop Android Applications using Java to develop the application. For the project management aspect of the project I will be using MS Visio for UML diagrams and UI Wire-framing Tool Pencil for drawing design and test user interfaces. The test tones can be generated using the android Tone Generator class. To comply with ISO 8253 -1:2010 I made be making use of an open source graph plotting library called AndroidPlot to generate the audiogram [5].



There are many different versions of Eclipse and a newer Android Studio was released, however I continued using Eclipse Juno as it is the same as Android Studio which mainly has a graphical and performance upgrade for newer PC’s. However a switch to Android Studio is simple as all that is required is importing the project into Android Studio.

## 4.1 Implementing Source Control with Git

Version control is one of the most important aspects of a software project. It’s an essential part of development where we can process the management of several files and different revisions of those files. The version control system provides us with a central hub where the code is located, it maintains a history of the work on the project, central place for backup and making separate copies of the project is not required and it also allows reversibility, meaning we can reverse changes [12]. For this project I learned to use Git-Hub where I created my own repository and licensed my project under the Apache license. In order to implement source control into the project a number of steps had to be taken. First the installation of the git-hub client for windows was required.



Figure 4 Git- Hub Client Application

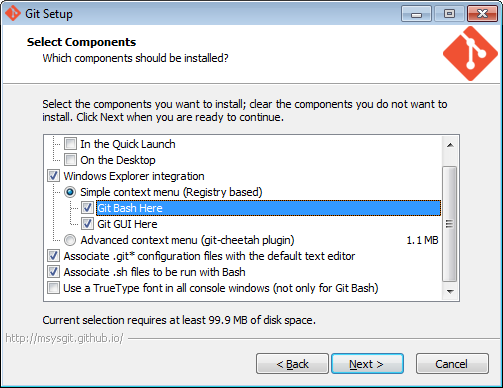


Figure 5 Git-Hub Bash module

I chose the Bash module as I am familiar with Bash scripting and chose the associated extensions for git files.

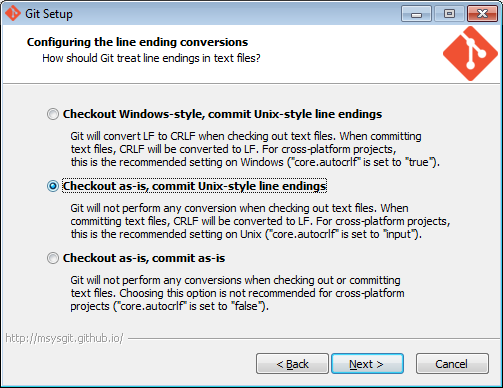


Figure 6 Checkout style using Unix style

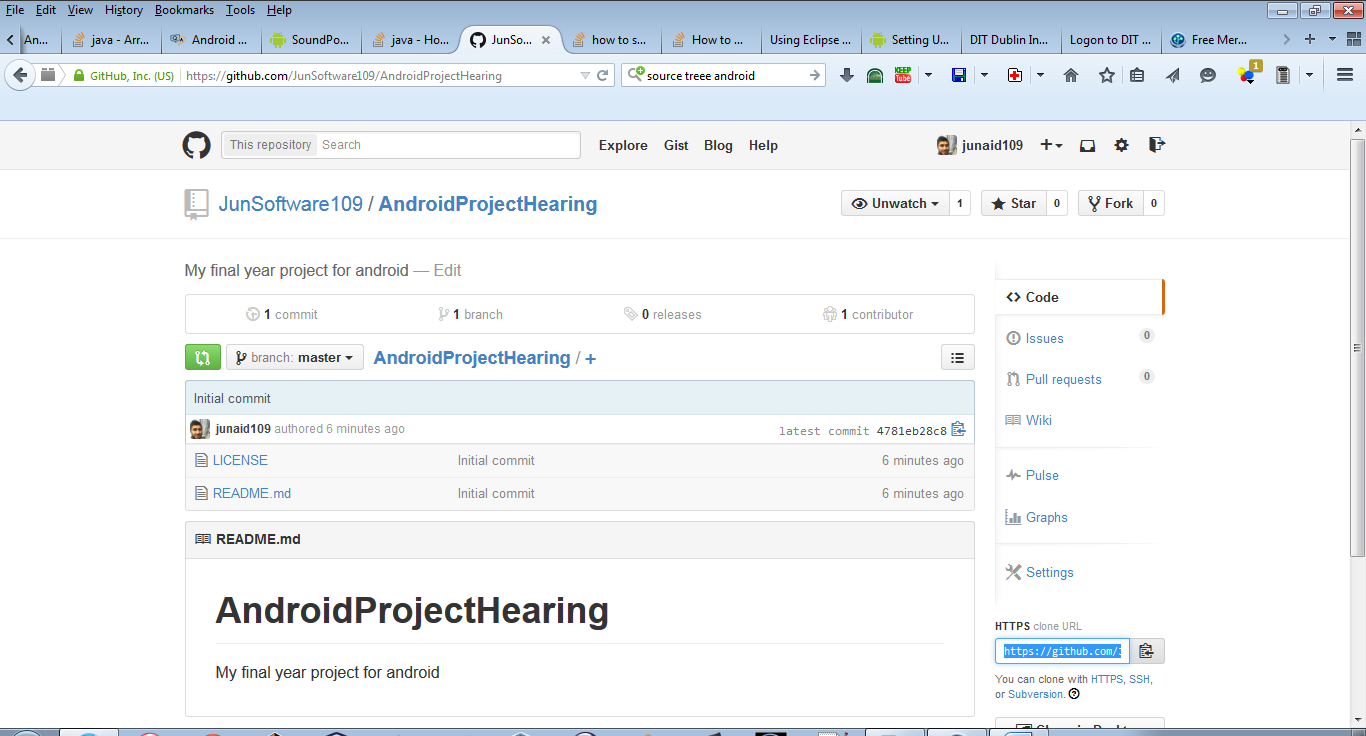


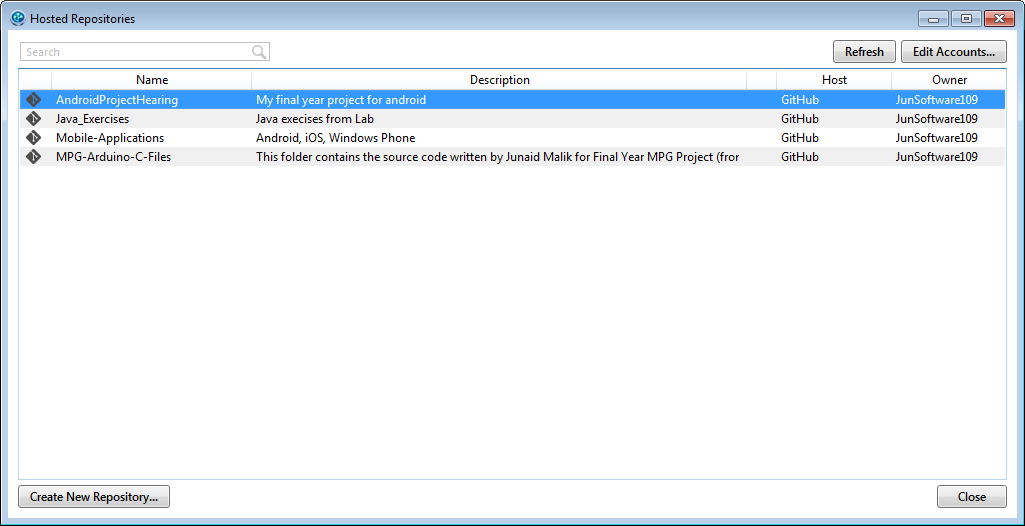
Figure 7 Initial commit of project to git.

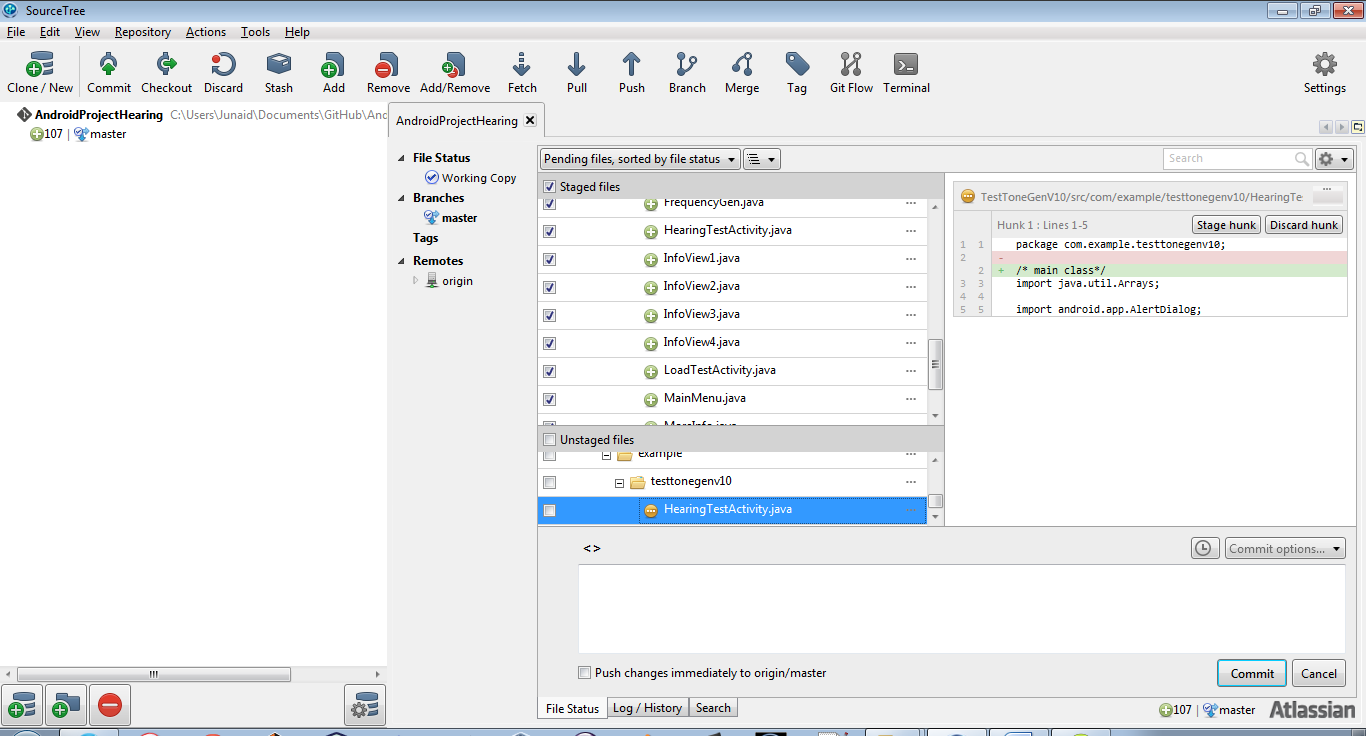
The repository is created along with license.

## 4.2 Using Source Tree Version Control System and adding git-hub

I used Source Tree by Atlassian for version control which is a very powerful VCS. These stores all the project files onto Git including source code files, documentation, build files configuration files, assets and libraries. Once I

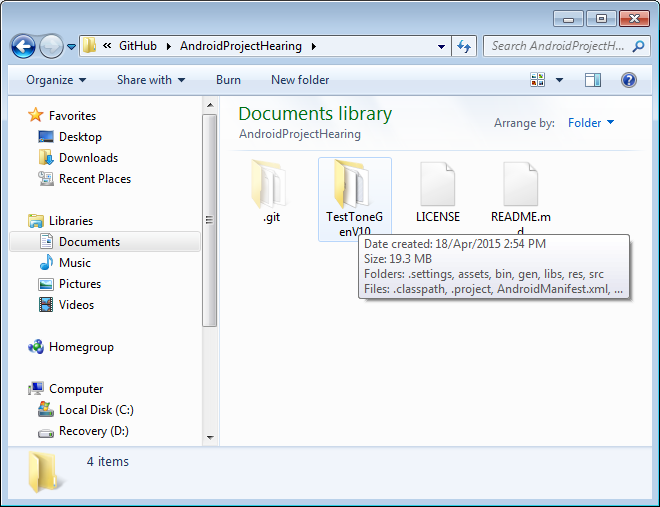
SourceTree Logo[16]



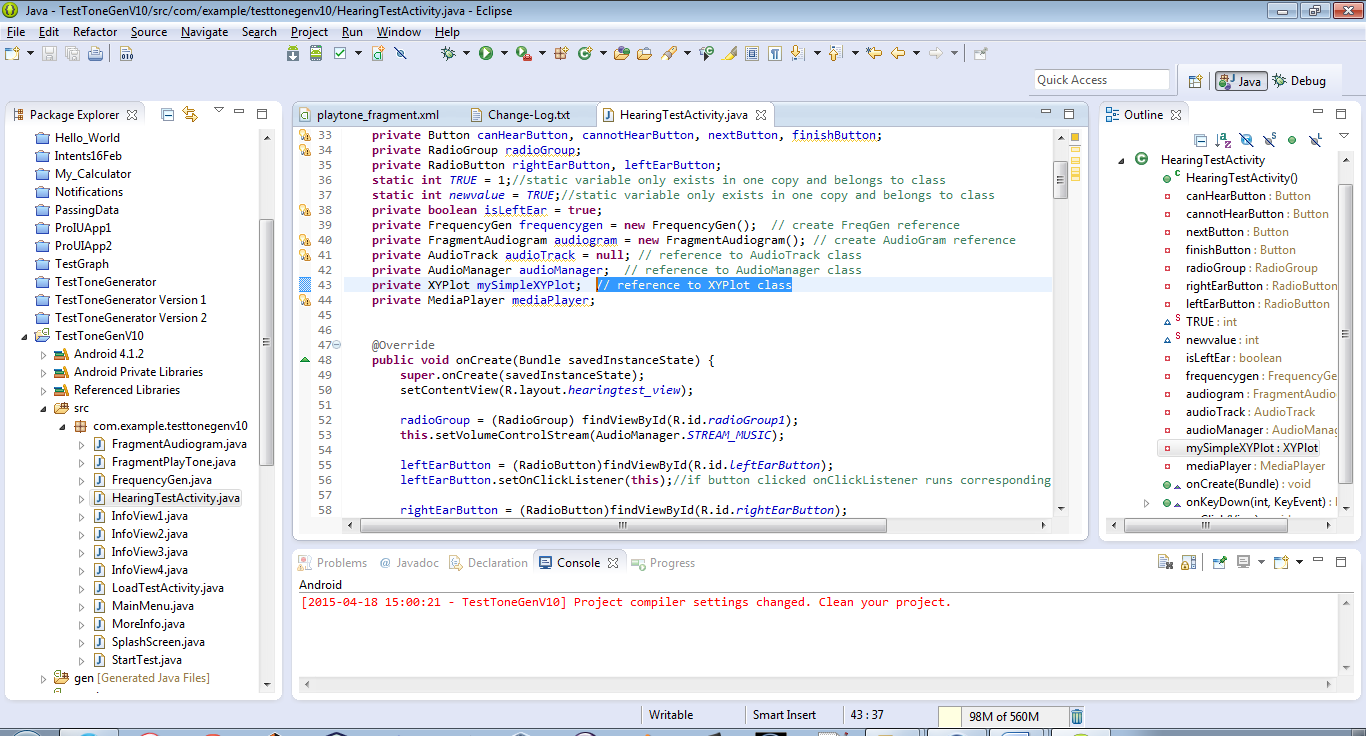
Once we have installed source tree, we link our git hub account

count and repository,

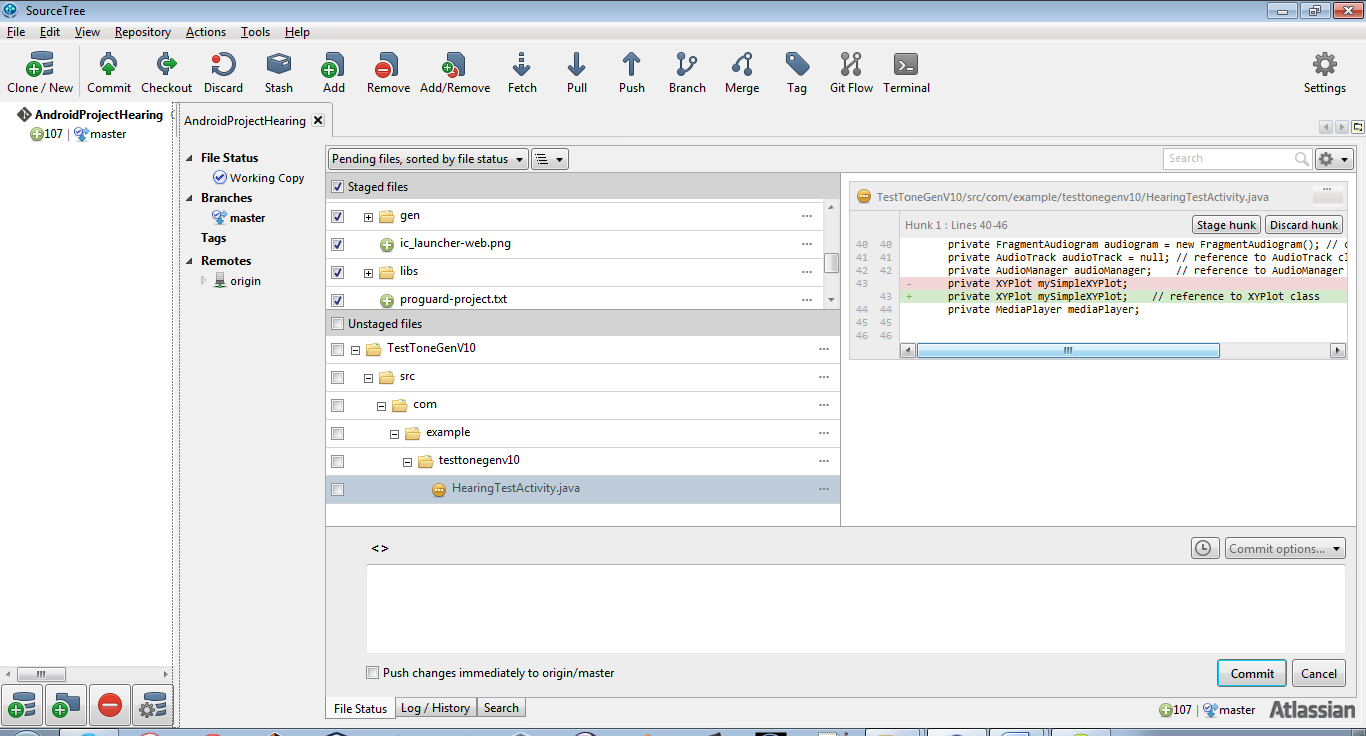
To implement source control to Eclipse we make a working backup copy of the project to the repository created locally for github



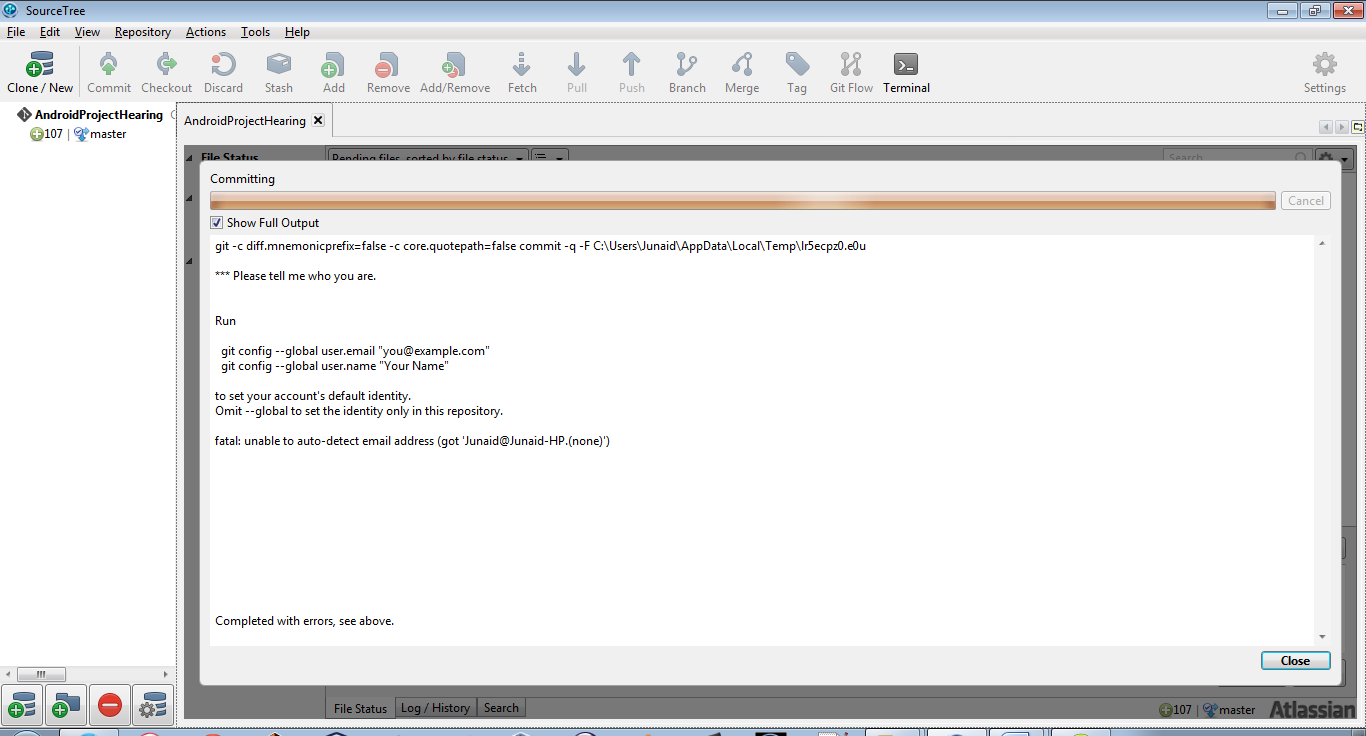
This now adds the project to source tree



We add some comments to HearingTestActivity class.

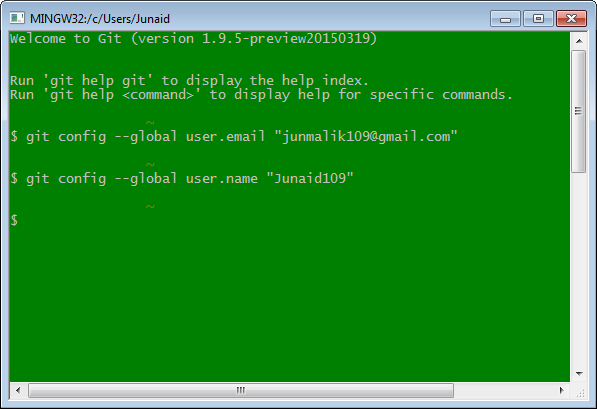


Any changes made to code are saved in source tree and we can see from the above that highlighted code in green is what we added,

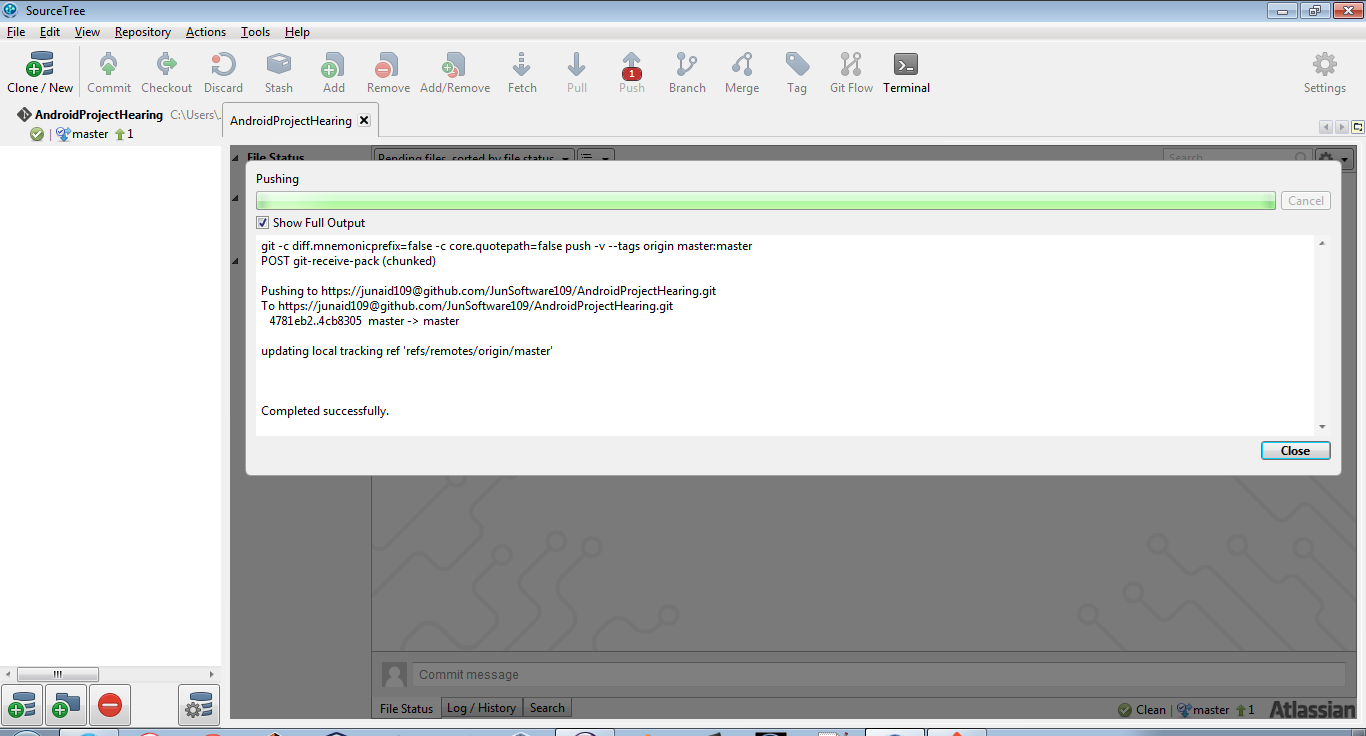


Once happy with changes made we can commit changes made.

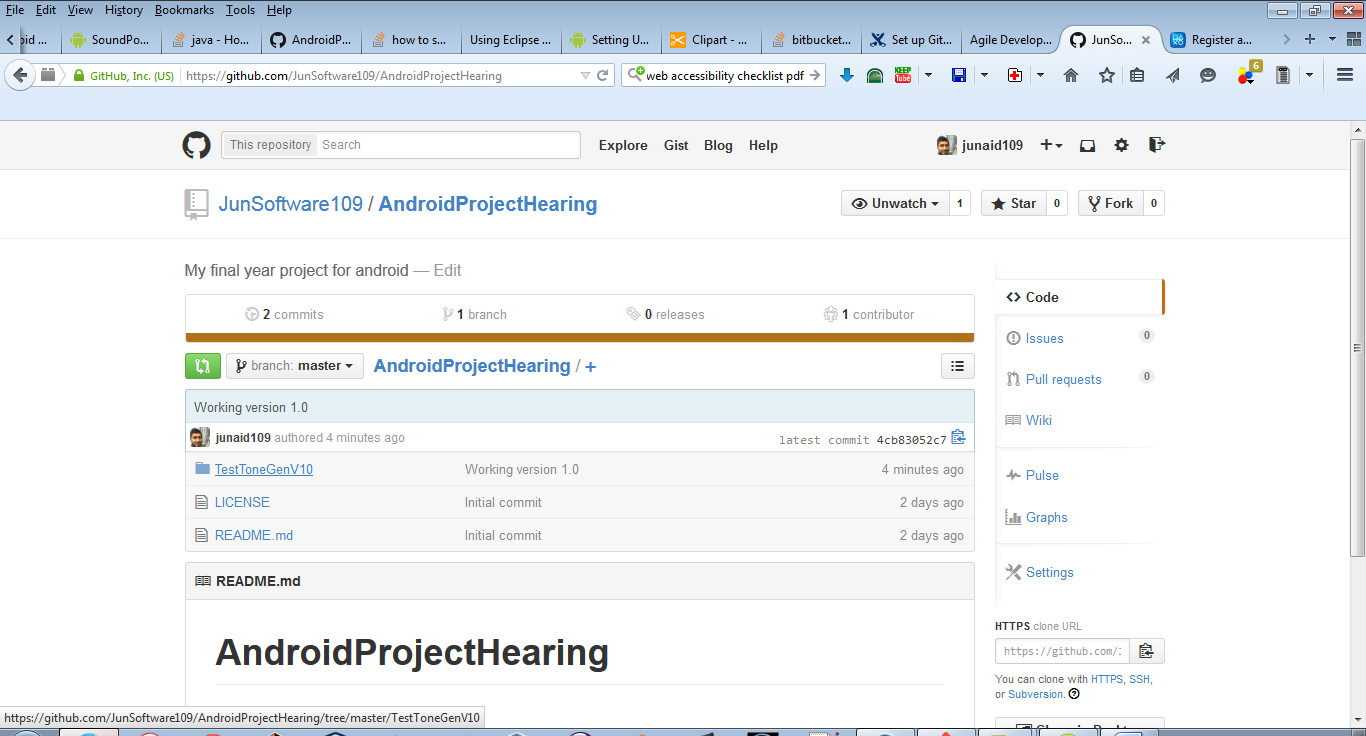
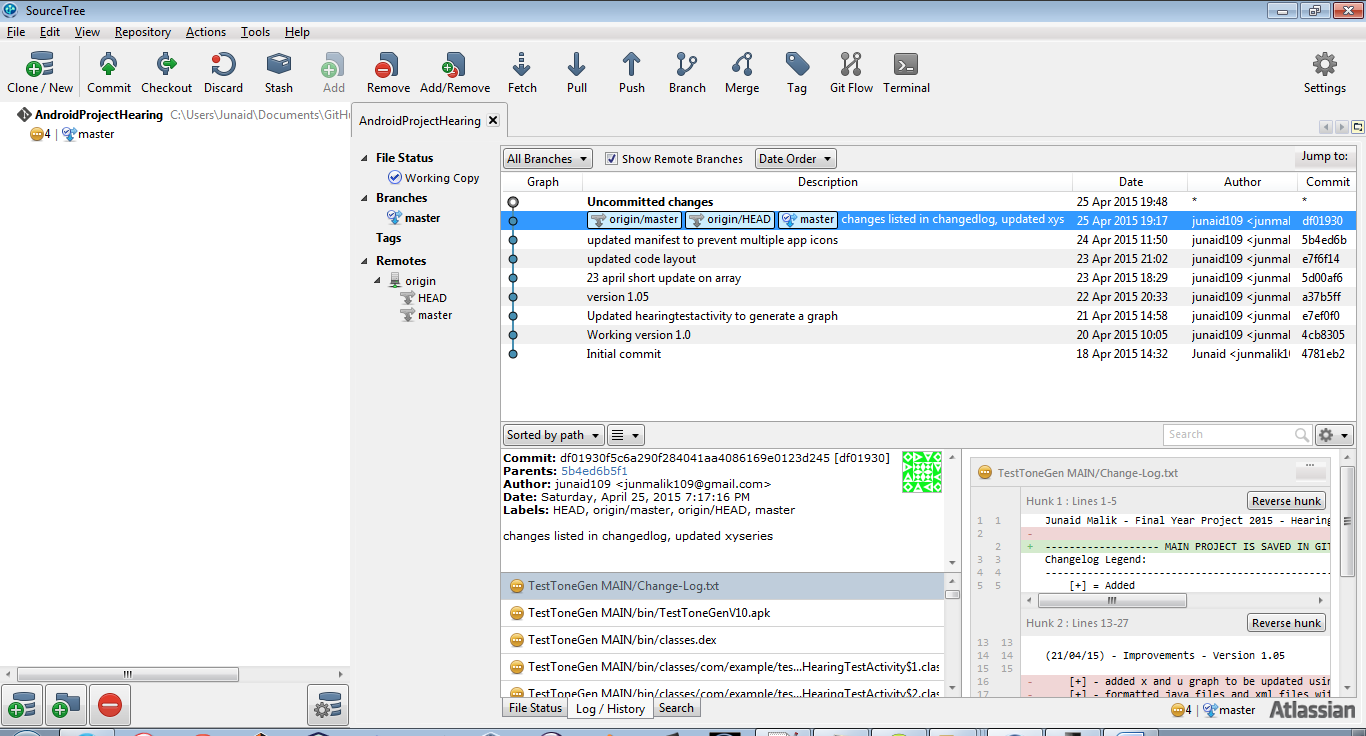
To commit changes we have to install Git for windows this adds the command line tools to run git.



Once I set the git config commands in git bash shell we can commit any changes made in source tree



The files are then added to the main git repository.



## 4.3 Android Activities

An activity represents a single screen that the user sees on the device. An application usually consists of multiple activities. Activities are the most observable part of the application. In Android, you can be looking at an activity of one application, but shortly after you could start another activity in a completely separate application. For example, if you are in the Calendar application and you decide to call a friend, you would be launching the activity to bring up the phone application in the Calendar application.

## 4.4 Android life-cycle

Activity can be expensive on the device CPU and RAM. It can sometime involve creating a new Linux process, allocating memory for all the new objects, inflating the objects from XML layouts, and setting up the screen. In Android the activity life cycle is managed by the Activity Manager.

Activity Manager is responsible for creating, destroying, and managing activities. For example, when the user starts an application for the first time, the Activity Manager will create its activity and put it onto the screen. Later, when the user switches screens, the Activity Manager will move that previous activity to a holding place. This way, if the user wants to go back to an older activity, it can be started more quickly. Older activities that the user hasn’t used in a while will be destroyed in order to free more space for the currently active one. This mechanism is designed to help improve the speed of the user interface and thus improve the overall user experience.

Programming for Android is conceptually different than programming for some other environments. In Android, you find yourself responding more to certain changes in the state of your application rather than driving that change yourself. It is a managed, container-based environment similar to programming for Java applets or servlets. So, when it comes to an activity life cycle, you don’t get to say what state the activity is in, but you have plenty of opportunity to say what happens during the transitions from state to state. The figure below shows the states that an activity can go through. [13]



Managing the lifecycle of your activities by implementing call-back methods is crucial to developing a strong and flexible application. The lifecycle of an activity is directly affected by its association with other activities, its task and back stack.

Regarding writing callback methods, we are interested essentially in the three Activity states:

Resumed

The activity is in the foreground of the screen and has user focus. (This state is also sometimes referred to as "running".)

Paused

Another activity is in the foreground and has focus, but this one is still visible. That is, another activity is visible on top of this one and that activity is partially transparent or doesn't cover the entire screen. A paused activity is completely alive (the Activity object is retained in memory, it maintains all state and member information, and remains attached to the window manager), but can be killed by the system in extremely low memory situations.

Stopped

The activity is completely obscured by another activity (the activity is now in the "background"). A stopped activity is also still alive (the Activity object is retained in memory, it maintains all state and member information, but is not attached to the window manager). However, it is no longer visible to the user and it can be killed by the system when memory is needed elsewhere.

# Chapter 5 Project Management

In this project I chose an iterative approach towards development. With the UX design, UI design and software development stages all iterative pieces of each sprint cycle. Bit by bit each feature was added. With an iterative approach I was able to focus on each small piece and create a working bit. My aim was to have a working piece of software by every week, this further enabled clarity and gave me quick feedback by demonstrating the build to peers or friends.



Figure 8 Sprint cycle

Each piece of the development cycle was broken down into 1 to 2 weekly sprints which consisted of adding functionality to each part of the project. I used user story mapping which helped draw out the project, this acted as a visual guide to the project, which solidified my understanding of the requirements and if any changes t occurred were easy foreseeable and changeable.

# Chapter 6 Requirements Analysis

## 6.1 Use Case Diagrams



Figure 9 Use-Case Diagram of main menu

# Chapter 7 System Design



Figure 10 A professional audiometer used in clinics

The above is a professional audiometer used in clinics it is priced at around €1650. [14]

The classes can be modelled by observing the mechanism of the audiometer. It contains a DSP unit which produces the pure tone so we create a frequency generator class, the audiometer may produce an audiogram with separate software on a pc but we can create a class that will either hold the information for audiogram or render it onto the users screen. The main activity is the Hearing Test Activity this class contains main behaviour of the overall system, this is where user selects which ear to test, frequency to select and



Figure 11 Simple class diagram

The diagram below demonstrates the architecture of the android package file which consists of the classes that are in the Dalvik Executable format (DEX). Dalvik is virtual machine in Android operating system. Programs are written in Java and compiled into byte code for JVM which is then translated into Dalvik byte code and stored in .dex or .odex (Optimized Dalvik).



Figure 12 the architecture of the apk.

## 7.2 UX/UI Design

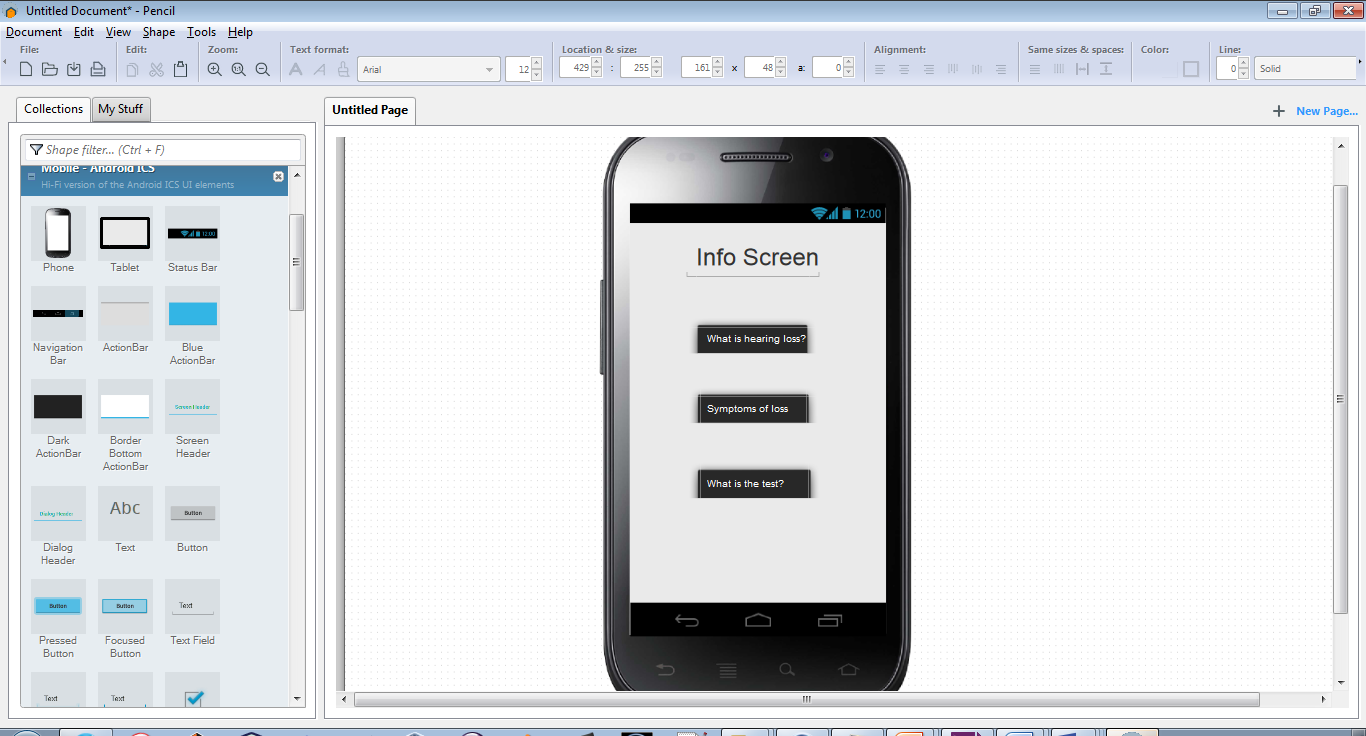


Figure 13 Quick layout design created in UX

I used pencil UI designer to make quick designs, to visualise the look and feel of the application, which I later than adapted into the design of the layout in XML.

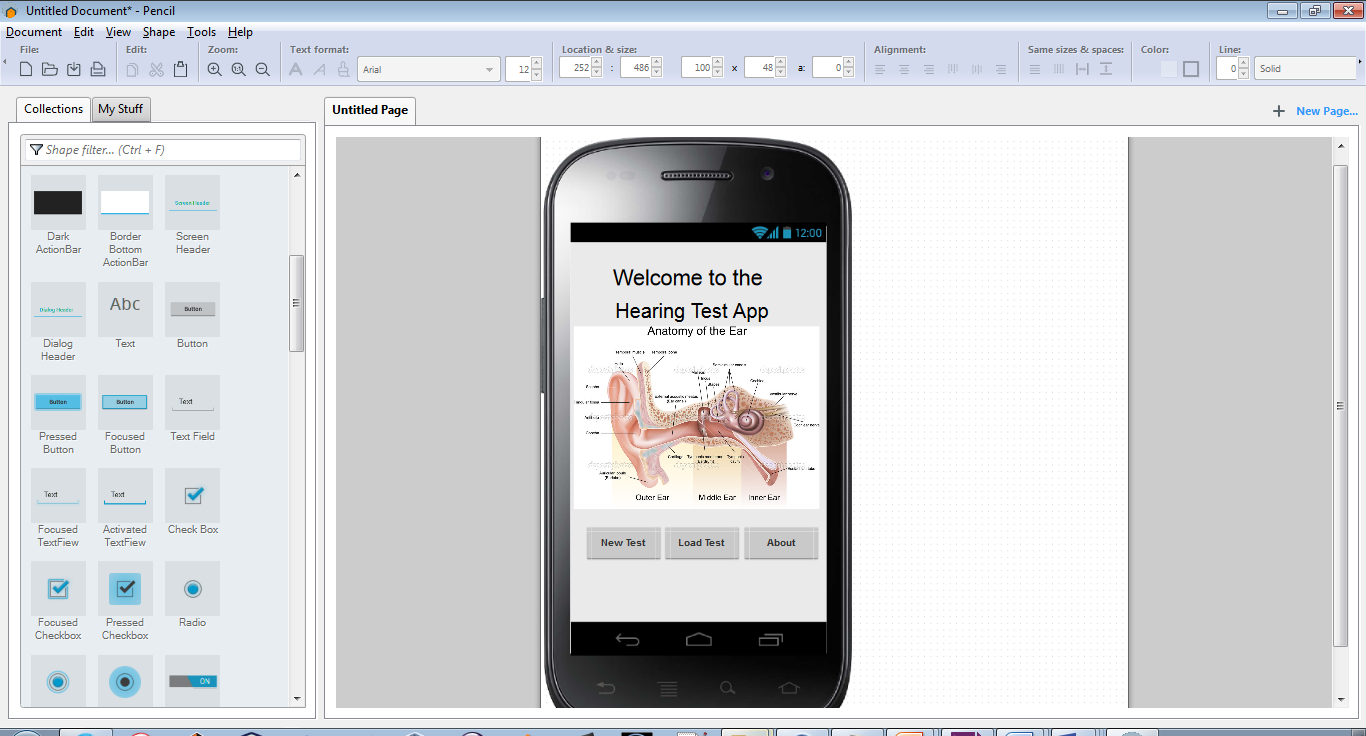


Figure 14 Initial design of the main menu

# Chapter 8 System Implementation

## 8.1 Implementing the activities



## 8.2 Implementing Frequency Generator

## 8.3 Implementing Audiogram with Androidplot library

## 8.4 Implementing Fragments

## 8.5 Implementing Rate this app plug in

I implemented a simple rate this app plugin written by Keisuke Kobayashi. It is an open source library and was available through git-hub. I decided to implement this feature to add to the applications user experience. As this plugin provides the user with a simple prompt alert box which provides a layer of interactivity to rate the application on the Google Play Store.

# Chapter 9 Testing

To create a test project in Eclipse with ADT:

1. In Eclipse, select File > New > Other. This opens the Select a Wizard dialog.

2. In the dialog, in the Wizards drop-down list, find the entry for Android, then click the toggle to the left. Select Android Test Project, then at the bottom of the dialog click Next. The New Android Test Project wizard appears.

3. Next to Test Project Name, enter a name for the project. You may use any name, but you may want to associate the name with the project name for the application under test. One way to do this is to take the application's project name, append the string "Test" to it, and then use this as the test package project name.

The name becomes part of the suggested project path, but you can change this in the next step.

4. In the Content panel, examine the suggested path to the project. If Use default location is set, then the wizard will suggest a path that is a concatenation of the workspace path and the project name you entered. For example, if your workspace path is /usr/local/workspace and your project name is MyTestApp, then the wizard will suggest /usr/local/workspace/MyTestApp. To enter your own choice for a path, unselect Use default location, then enter or browse to the path where you want your project.

To learn more about choosing the location of test projects, please read Testing Fundamentals.

5. In the Test Target panel, set An Existing Android Project, click Browse, then select your Android application from the list. You now see that the wizard has completed the Test Target Package, Application Name, and Package Name fields for you (the latter two are in the Properties panel).

6. In the Build Target panel, select the Android SDK platform that the application under test uses.

7. Click Finish to complete the wizard. If Finish is disabled, look for error messages at the top of the wizard dialog, and then fix any problems.

The testing framework has these key features:

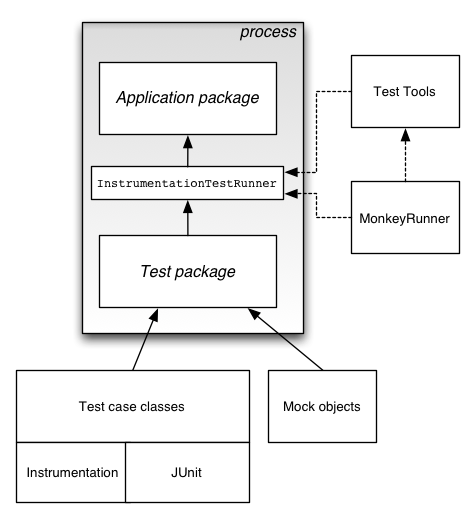
• Android test suites are based on JUnit. You can use plain JUnit to test a class that doesn't call the Android API, or Android's JUnit extensions to test Android components. If you're new to Android testing, you can start with general-purpose test case classes such as AndroidTestCase and then go on to use more sophisticated classes.

• The Android JUnit extensions provide component-specific test case classes. These classes provide helper methods for creating mock objects and methods that help you control the lifecycle of a component.

• Test suites are contained in test packages that are similar to main application packages, so you don't need to learn a new set of tools or techniques for designing and building tests.

• The SDK tools for building and tests are available in Eclipse with ADT, and also in command-line form for use with other IDEs. These tools get information from the project of the application under test and use this information to automatically create the build files, manifest file, and directory structure for the test package.

• The SDK also provides monkeyrunner, an API for testing devices with Python programs, and UI/Application Exerciser Monkey, a command-line tool for stress-testing UIs by sending pseudo-random events to a device.



## Test Structure

Android's build and test tools assume that test projects are organized into a standard structure of tests, test case classes, test packages, and test projects.

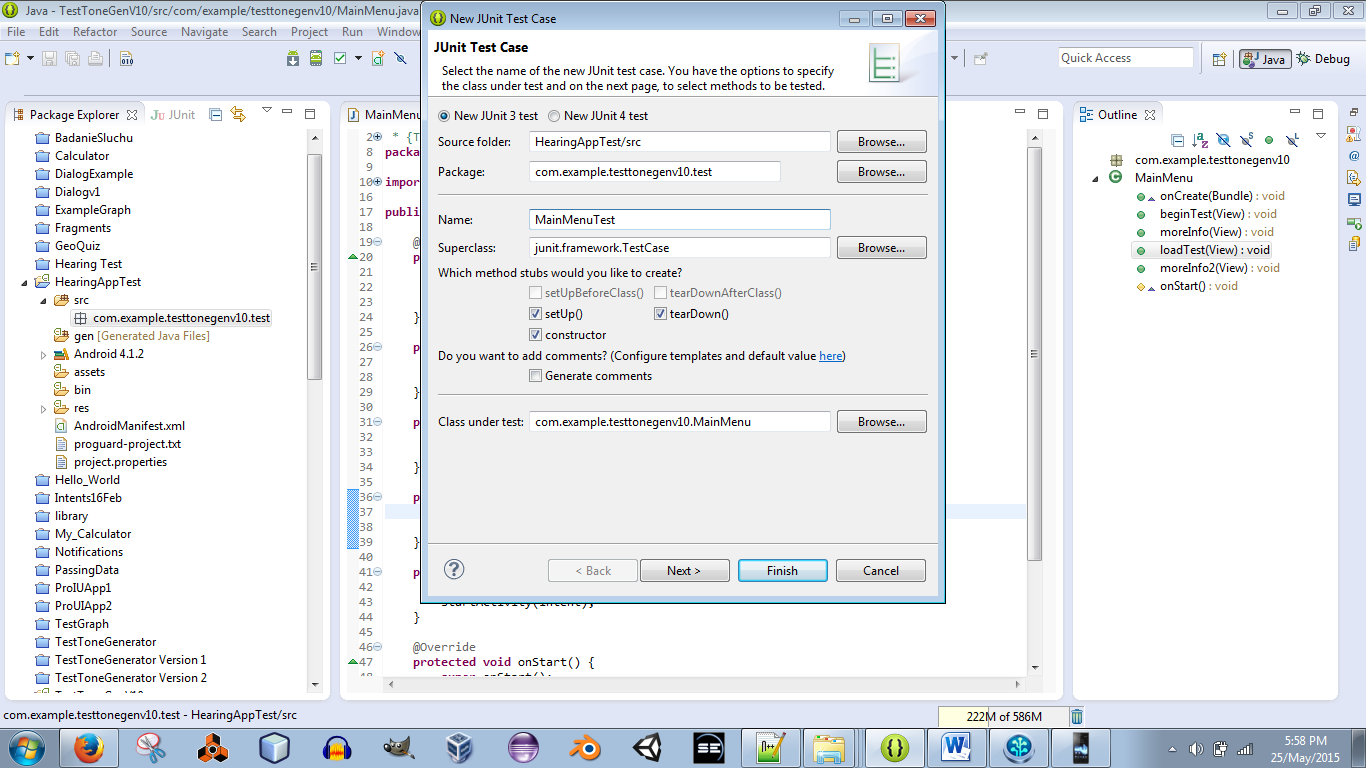
Android testing is based on JUnit. In general, a JUnit test is a method whose statements test a part of the application under test. You organize test methods into classes called test cases (or test suites). Each test is an isolated test of an individual module in the application under test. Each class is a container for related test methods, although it often provides helper methods as well.

In JUnit, you build one or more test source files into a class file. Similarly, in Android you use the SDK's build tools to build one or more test source files into class files in an Android test package. In JUnit, you use a test runner to execute test classes. In Android, you use test tools to load the test package and the application under test, and the tools then execute an Android-specific test runner.

| **Test area** | **Description** |
| --- | --- |
| Activity life cycle events | You should test if you activity handles the Android life cycle events correctly. You should also test if the configuration change events are handled well and if instance state of your user interface components is restored. |
| File system and database operations | Write and read access from and to the file system should be tested including the handling of databases. |
| Different device configurations | You should also test if your application behaves well on different device configurations. |

### Testing preconditions

It is good practice in Android testing to have one method called testPreconditions() which tests the pre-conditions for all other tests. If this method fails, you know immediately that the assumptions for the other tests have been violated.

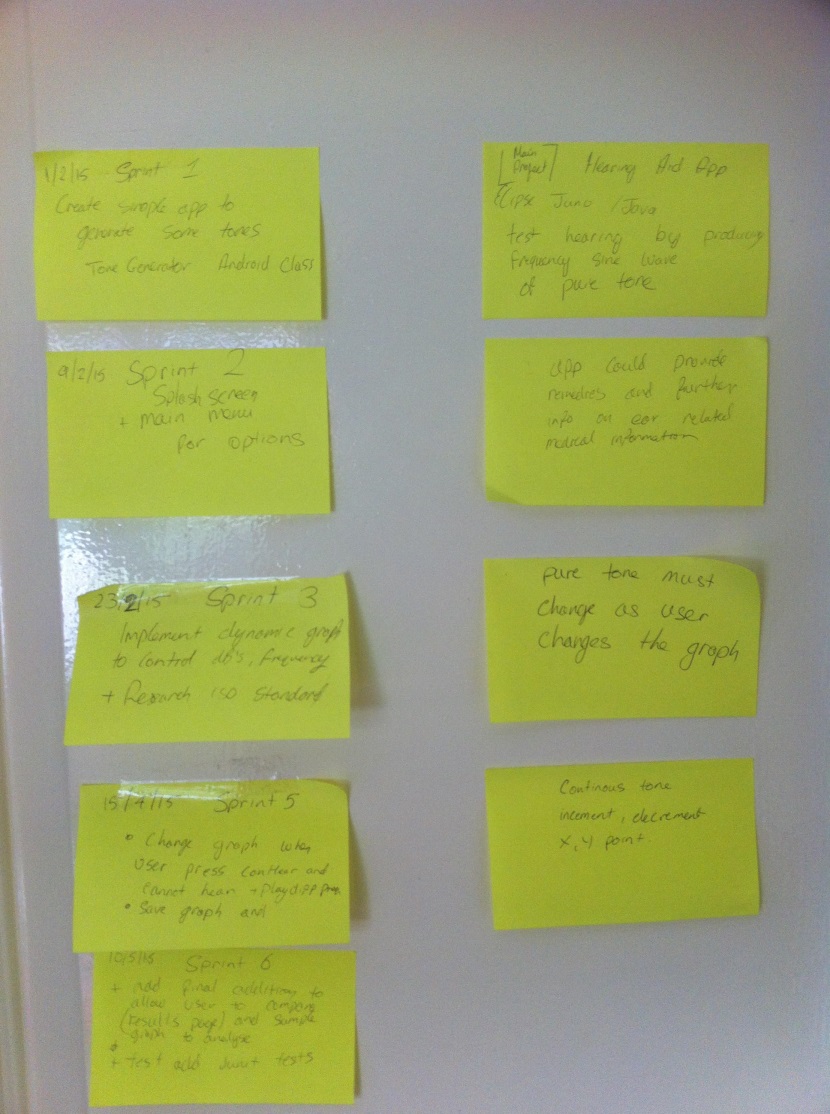


# Appendices

## Appendix A **Java code for Dijkstra’s Algorithm**

|  |
| --- |
| This code was written by Vladimir Sutskever (Sutskever). It best to use monospaced font for source code (e.g. Courier New ).  public class Dijkstra {  private Graph graph;  //priority queue stores all of the nodes, reachable from the start node  //The queue is sorted by the node.distance  private GraphNodePriorityQueue priorityQ = new GraphNodePriorityQueue();  private Hashtable <GraphNode,Integer> distance =  new Hashtable<GraphNode, Integer>();  //1. Get the list of all nodes in the graph  //2. Initialize distance vector to infinity  //3. Edge Cost function  public Dijkstra (Graph g){  this.graph = g;  this.graph.getStartNode().setDistance(0);  this.priorityQ.add(this.graph.getAllNodes());  }  // Actual algorithm  public void go(){  while (this.priorityQ.hasMore()){  GraphNode n = this.priorityQ.remove();  n.print();  for (Edge e: n.getOutGoingEdges()){  GraphNode adjNode = e.getNode();  Integer newPossiblePathCost = e.getCost()+n.getDistance();  if (newPossiblePathCost < adjNode.getDistance()){  adjNode.setDistance(newPossiblePathCost);  this.priorityQ.updateGraphNodeDistance(adjNode);  System.out.println("current node "+ (e.getNode()).getVal());  this.priorityQ.PrintContents();  }  }  }  } |

## Appendix B User Story Mapping



## Appendix C

# Chapter 11 Bibliography

[1] Pure tone air and bone conduction threshold audiometry with and without masking - British Society of Audiology. 2015. Pure tone air and bone conduction threshold audiometry with and without masking - British Society of Audiology*.* [ONLINE] Available at: <http://www.thebsa.org.uk/resources/pure-tone-air-bone-conduction-threshold-audiometry-without-masking/>. [Accessed 20 May 2015]

[2] Suzanne Purdy Head of Speech Science University of Auckland, Guideline for diagnosing occupational noise induced hearing loss Part 3: Audiometric standards Available at: <http://www.acc.co.nz/PRD_EXT_CSMP/groups/external_communications/documents/reference_tools/wpc091007.pdf>.

[Accessed 20 May 2015]

[3] Hearing Tests (Audiometry) | Health | Patient.co.uk. 2015. Hearing Tests (Audiometry) | Health | Patient.co.uk*.* [ONLINE] Available at: <http://www.patient.co.uk/health/hearing-tests-audiometry>. [Accessed 20 May 2015]

[4] Types of Tests Used to Evaluate Hearing in Children and Adults. [ONLINE] Available at: <http://www.asha.org/public/hearing/Types-of-Tests-Used-to-Evaluate-Hearing/>.

[Accessed 20 May 2015]

[5] Androidplot — Bitbucket. [ONLINE] Available at: <https://bitbucket.org/androidplot/androidplot>. [Accessed 20 May 2015]

[6] Marble Mice: Generate And Play A Tone In Android. [ONLINE] Available at: <http://www.marblemice.blogspot.ie/2010/04/generate-and-play-tone-in-android.html>.

[Accessed 26 May 2015]

[7] Dr Béatrice Duthey, Ph.D., 20th February2013, Priority Medicines for Europe and the World, “A Public Health Approach to Innovation", Update on 2004 Background Paper, 6.21 Hearing Loss

Available at: <http://www.who.int/medicines/areas/priority_medicines/BP6_21Hearing.pdf>.

[Accessed 26 May 2015]

[8] Application Development for Android Available at: <http://www.who.int/medicines/areas/priority_medicines/BP6_21Hearing.pdf>.

[Accessed 28 May 2015]

[9] Prevalence of age-related hearing loss in Europe: a review - Springer. [ONLINE] Available at: <http://link.springer.com/article/10.1007%2Fs00405-011-1597-8>.

[Accessed 28 May 2015]

[10] Evaluation of the Social and Economic Costs of Hearing Impairment by Shield (2006 - Google Scholar. 2015. Available at: https://scholar.google.com/scholar?q=Evaluation+of+the+Social+and+Economic+Costs+of+Hearing+Impairment%E2%80%9D+by+Shield+

[Accessed 28 May 2015]

[11] Human Ear Diagram Available at: <http://www.rudyard.org/wpcontent/uploads/2013/09/human-ear-diagram.jpg>.

[Accessed 28 May 2015]

[12] Pete Goodliffe, 2014. Becoming a Better Programmer: A Handbook for People Who Care About Code. 1 Edition. O'Reilly Media.

[13] Marko Gargenta, 2011. Learning Android. 1 Edition. O'Reilly Media.

[14] Portable Digital Audiometer Supplier Hearing Aid Professional Physician's Office <http://www.alibaba.com/product-detail/Portable-Digital-Audiometer-Supplier-Hearing-Aid_1233690528/showimage.html>.

[Accessed 30 May 2015]

[15] Noise-induced Hearing <https://www.nidcd.nih.gov/staticresources/health/hearing/NIDCD-Noise-Induced-Hearing-Loss.pdf>.

[Accessed 30 May 2015]

[16] Git client for Mac and Windows | Atlassian. [ONLINE] Available at: <https://www.atlassian.com/software/sourcetree/overview?_mid=e0734c30de09bb01fe8ee8625dc35ca4&gclid=CIeZgeqR6sUCFYJZ2wodDFIAxg>.

[Accessed 30 May 2015]