

Md.Abdullah-Al Mamun

# Know Finnish: Android Application to Learn Finnish Language

Metropolia University of Applied Sciences

Bachelor of Engineering

Information technology

Bachelor's Thesis

31 October 2017

Author Title	Md. Abdullah-AI Mamun Know Finnish: Android Application to Learn Finnish Language.
Number of Pages Date	45 pages + 7 appendices 31 October 2017
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Professional Major	Software Engineering
Instructors	Tero Nurminen, PhD, Principal Lecturer
<p>The purpose of this thesis is to develop an Android Application for learning the Finnish language quickly. A large number of foreign people are living in Finland. The main objective of this application is to help those people who want to communicate with Finnish people or want to develop their language skills.</p> <p>The application was developed on Android studio tools. The app was developed using API level 16 or above (highest 26) because lower API level uses more Android devices. This application was developed using Android Studio IDE. Java was the main programming language used to develop this application.</p> <p>The home screen of the application developed in this thesis contains a logo, a motto, and six clickable image views for the features provided. Some of the important features include word bank, vocabulary, true false questions and word game. When the app was tested with real users, the results were rather promising.</p> <p>Gradually this application will be improved and features such as pronunciation and sound effects will be added. In future, it will be uploaded to the Google Play Store.</p>	
Keywords	Android, User Interface, Java, Dictionary, Android Game, Finnish language, Quiz.

## Contents

### List of Abbreviations

1	Introduction	1
1.1	Purpose of the Thesis	2
2	Liturature Review	4
3	Application Development Environment	7
3.1	Overview of Android System	7
3.2	Android System Architechture	8
3.2.1	Application Layer	9
3.2.2	Framework Layer	11
3.2.3	Application Libraries	11
3.2.4	Android Runtime	12
3.2.5	The Linux Kernel Layer	13
3.3	Integrated Development Environment (IDE)	13
3.3.1	Manifests File	13
3.3.2	Resources File	14
3.3.3	Java File	14
3.3.4	Gradle file	15
3.4	Installation of Android studio	15
3.4.1	Creating new project	16
3.4.2	Creating and running the app in emulator	18
3.4.3	Running the app on Android device	18
4	Know Finnish app development techniques	20
4.1	Manifest File	21
4.2	Application design and control	22
4.2.1	Home Screen	24
4.2.2	Own Dictionary	28
4.2.3	Own Dictionary Controller	30
4.2.4	Verify You!	32
4.2.5	Verify You! Controller classes	32
4.2.6	Vocabulary	34
4.2.7	Vocabulary Controller	34
4.2.8	Word game	35

4.2.9	Word Game Controller	36
4.2.10	TFQ	37
4.2.11	TFQ Controlling	37
4.2.12	Word Bank	39
4.2.13	Word Bank Controller	40
5	Evaluation of the App	41
6	Conclusion	43
6.1	Limitations and Future work	43
	References	44
	Appendices	
	Appendix 1: Own Dictionary Controller classes	
	Appendix 2: Verify You! Controller classes	
	Appendix 3: Vocabulary Controller classes	
	Appendix 4: Word Game Controller classes	
	Appendix 5: TFQ Controller classes	
	Appendix 6: Word Bank Controller classes	
	Appendix 7: Online Evaluation Form	

## List of Abbreviations

IDE	Integrated Development Environment
ADT	Android Development Tools
GUI	Graphical The user Interface
ADT	Android Development Tools
SDK	Software Development Kit
ADT	Android Development Tools
JVM	Java Virtual Machine
API	Application Programming Interface
TFQ	True False Questions
AVD	Android Virtual Device

## 1 Introduction

This study shows how an android app can be used as a tool of self-training in day to day life. The app demonstrates various aspects of Finnish language like practical vocabulary, easy grammar, and pictorial representation in a playful and friendly manner to the users.

Since the innovation of the telephone in the late eighteenth century, the field of broadcast communications has had various real advances in coding, data communication technologies and so forth. Mobile phones were boned after the communication technologies improvement and they have improved rapidly. These 21<sup>st</sup>-century portable phones enabled clients to make telephone calls from anyplace, as long as they are in the radio transmission of their base stations and ready to keep up a dynamic connection (generally alluded to as cells). Be that as it may, these cell phones were unequipped to do anything else besides making telephone calls and putting away the client's contact list. Situation has changed nowadays because of advancement of mobile technology. However, the situation has changed now because of the rapid development and new inventions of mobile and network technology.

Mobile application developments are thought to be one of the quickest developing sectors in the Information Technology industry. Clients appreciate easy usable, easy understandable user interface. Most of the users do not like hard and complex designs of the application. Therefore, mobile apps display a more prominent interface for cooperation with business frameworks than utilizing web applications by the means of Web Browser.

With the vast development of the World Wide Web (WWW), the popularity of the internet has increased drastically and the use of the mobile gadgets decreased at that time. As the assortment of such needs expanded, the new type of smart mobile gadgets was required that had capacities to run altered/particular applications to satisfy client needs. These new gadgets were called smartphones. With the time, the cell phone developed to be a little PC on which the cell phone was additionally only one of numerous different applications. They highlighted music players, camera, photograph/video surfing programs, diversions, and so forth, as other conceivable applications. These smart phones

consume less power and energy. It's a kind of computer that has less weight and exceedingly effective operating system to run the majority of a client's applications.

Numerous proficient arrangements were contrived after some time by organizations, for example, RIM, Palm, and so on. Nonetheless, just two arrangements remained over the pack because of their effectiveness and simplicity of utilization: Google's Android Operating System and Apple's iOS running on iPhones. "**Know Finnish**" application is based on Android mobile technology.

Android, which was founded on the open source Linux operating system can keep running on an assortment of gadgets. Accordingly, various OEM organizations pick Android for their gadgets, along these lines boosting its piece of the overall industry. The Android working framework, as well as any application composed for it, should be tried on an assortment of mobile gadgets before being discharged [2].

Also, the Android platform is publicly released, and bolsters significantly more open improvement platforms and capacity to interface/run third-party tools to upgrade the usefulness of applications. Apple, in turn, forces extremely strict application improvement rules and screening methodology to enable any application to be made accessible generally.

The devices utilized for any application advancement on iOS are entirely controlled by Apple. Likewise, the testing and troubleshooting apparatuses accessible on Android through its IDE are extremely developed compared to iOS's Xcode.

## 1.1 Purpose of the Thesis

The main objective of this thesis is to build an Android application to learn the Finnish language. The target people are foreign people living in Finland as well as people who have recently arrived in Finland. Their target is to learn the Finnish language to help them better integrate to the society for study, work and social purposes.

The name of the application is "**Know Finnish**". The motto of the app is "**Know Finnish, Develop Yourself**". The user interface of the app has been designed using Photoshop and Adobe Illustrator CS6. Java is used as core programming language to build the Know Finnish Application.



## 2 Liturature Review

Android, an open source, Linux kernel-based mobile operating system developed by the giant tech company Google Inc. Android was revealed by Google on 2007 [5]. According to Morrill, Dan 2008, the first commercial Android device was released by Google in September 2008. Figure 1 exhibits the first official Android logo. It gives the idea about the look of nascent period of the android application.



Figure 1. First official Android logo (2007-2014). Source: <http://www.Android.com>

This thesis converses about Android Application development technologies and how to apply them to various research problems. The official Android website describes this Android platform as follows: “Android is a software stack for mobile devices that includes an operating system, middleware, and key applications” [2]. Android supplies a few basic applications including calendar, maps, contacts, browser, email, SMS and so on [2].

Google play services have played a vital role in the Android market from 2012 [3].Google play services can be used by the application developer to unveil their application for free, while they can upload their application by asking price as well. The user can download various kinds of applications according to their needs from Google play Service [3].

Nowadays many portable mobile devices use the android operating system, software and some other applications. The most popular programming language is Java to develop an Android application. Other languages can be used on Android such as C#, Kotlin (from Jetbeans), C++, PhoneGap technology (uses HTML5, JavaScript, CSS), python. However, Android is a Java programming language dependent platform [2]. Java is the main programming language used in this thesis.

Android application advancement basics have been depicted by many authors, which incorporate setting up the Android environment, manifest file (the main XML file), supporting XMLfiles for design purposes and activity file (Java file). Jackson (2017) de-

scribes "three noteworthy parts of Android improvement conditions: Java, Eclipse, Android" and gives guidelines on the most proficient method for downloading and installing important files to build up this environment [4]. According to Felker (2011) the following components are necessary to build up an Android application environment:

- An Integrated Development Environment (ie. Android Studio, Eclipse)
- A software development kit
- Java SE development kit
- Android development tools (plugin)

The means given by these two creators are standard. They show up in many books composed of Android advancement and are likewise introduced on the official Android site.

During May 2013, Google Inc. announced at the I/O developer's event that Android Studio is the IDE for Android and is planned as the other option to Eclipse. Nowadays Android Studio is the most popular integrated development environment (IDE) for developing Android applications. During the writing of the thesis, Android studio released its 3.0 version for Windows. Android studio 3.0 is the most recent version of Android Studio and it was released in October 2017. Android Studio 3.0 was used to program the Know Finnish Android application.

Four essential segments of Android applications are the following:

- Activity
- Service
- Broadcast Receiver
- Content Provider

Each and every Android application contains at least one of the components from the above four components [6]. Since Activity "shows user Interface and reacts to the framework and client started occasions" [6], it is utilized every now and again for Android applications. AndroidManifest.xml is the base file of an Android application that contains direction of activities [7].

Application perspectives are displayed through XML formats by Activities and pass through Intents. Clear comprehension of these ideas and the Java language is essential to begin actualizing the development strategies utilized as a part of Android applications.

### 3 Application Development Environment

This chapter provides the necessary information about the overview of the Android system, required development tools, architecture of the system, basic environment, activity lifecycle, UI elements among others.

#### 3.1 Overview of Android System

Android is a bundle of software that incorporates a Linux based Android operating system, softwares and some other applications. It is intended to keep running on touch screen smartphones like PDAs, mobile devices, and tablet PCs. Android Inc. was established in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Afterward, Android Inc. was gained by Google in 2005. It is presently possessed by Google in conjunction with Open Handset Alliance. This Alliance is a gathering of 86 equipment, programming and media transmission organizations creating open gauges for cell phones or mobile devices.

Android Software Development Kit (SDK) is used to create an Android application and Java is used as the main programming language. The Java code is platform independent gathered into bytecode (.class records). The Java Virtual Machine compatible .class files converted into Dalvik Executable files (.dex files) by The Dalvik interpreter. It is used for runtime interpretation [9].

The Android SDK gives the vital Application Programming Interface (APIs) and what's more, Android developer apparatuses for building, testing and troubleshooting applications in Android. One such tool is the Android Developer Tool (ADT) plugin for Eclipse IDE. Eclipse IDE was used by the developer for long time. Afterwards, Eclipse was replaced by a new IDE called Android Studio. During 2013 giant tech company Google Inc announced about the Android studio as Android application developing IDE [9].

An Android application comprises of at least one of the accompanying segments:

- Activities
- Services
- Content Providers
- Broadcast Receivers

Every segment has a particular task and adds to the general application conduct. Every one of the segments utilized as a part of the application and the gadget highlights are proclaimed in the Manifest file. The Android application is additionally made out of Resources that are separate from the source code. These assets like the pictures, sound documents, and so forth add to the viewable presentation of the application.

In the ensuing subsections, the user interface design, the system architecture, features of the operating system, the framework of the Android application will be explained.

### 3.2 Android System Architecture

Android Operating System (OS) can be considered as a product stack comprising of different layers, where each layer is a gathering of a few distinct segments. The layers are:

- The Linux Kernel (bundle of drivers for networking, hardware, file access and internal communication)
- Native Libraries, Daemons, and Services
- Framework Services and Libraries
- Applications

Each layer in the system architecture gives diverse administrations to the layer above it. Figure 2 [8] demonstrates the diagram of Android Architecture. Each layer will be discussed in detail.

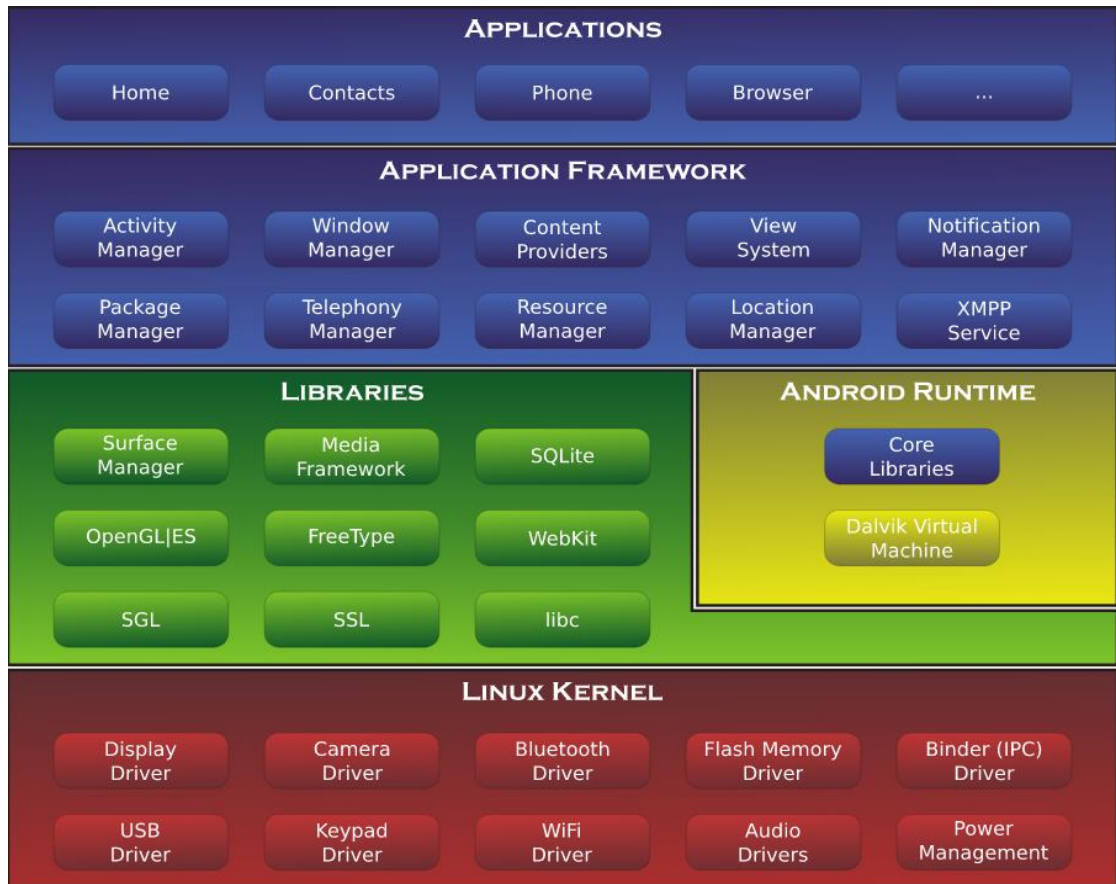


Figure 2. Android System Architecture. Source: [https://elinux.org/Android\\_Architecture](https://elinux.org/Android_Architecture)

Each layer of android system architecture (figure 2) provides clear idea about the functions of different components of the layers.

### 3.2.1 Application Layer

Application layer is the base layer of the Android Architecture. Android gives a set of base applications, for example:

- Web Browser
- Contact Manager
- Email Client
- Calendar
- SMS Program
- Maps.

Client characterized applications can likewise connect with these base applications.

Java programming language is used to write all of the applications. The Android SDK apparatuses gather all information and assets into an Android bundle which is a document with the .apk file. This record is considered as one application and the .apk files are installed on mobile devices.

After the installation process, the Android OS finds every application as an alternate client along these lines making the Android OS a multi-client Linux framework. The framework doles out the user ID to the application, sets authorizations to every one of the records that the application needs so just the client ID doled out to the application can get to them. Each application keeps running on its own particular Linux process, each procedure has its own virtual machine, so an application's code runs autonomously from different applications [10].

As a matter of course, every application just approaches its own parts and cannot get to any piece of the framework without consent of the system, therefore making an exceptionally secure condition. Notwithstanding, it is feasible for an application to get to different applications and framework administrations. Two applications share data between them by having a similar client ID. An application can ask for consent to get to gadget's information like contacts, camera and so forth which will be conceded according to the application installation time. Android applications comprise of at least one application segments [10].

**Application Components** shape the basic building pieces of an application and everyone of them is unmistakable in the way it adds to the general conduct of the application. There are four sorts of segments. They are following:

- **An Activity** is a solitary screen that the client sees and cooperates inside an application. Every action runs freely and one application can begin a movement of another application [14].
- **Content Providers** give an approach to share information over applications. Utilizing content suppliers, information can be recovered, made and changed in the application [14].
- **Services** do not have a UI. It keeps running in the foundation to perform long-running operations or to take a shot at remote procedures. A case of administration is playing music out of sight while the client is getting to an alternate application [11].

- **Broadcast Receivers** react to any communicated declarations. Many communications are made by the framework; however, they can likewise be started by the application. They are proposed to do next to no work and are along these lines considered as "entryways" to different parts [10].

### 3.2.2 Framework Layer

Application Framework is a layer that the Application Layer specifically associates with. It is a toolbox that empowers the reuse of segments: an application can uncover its capacities and what's more; some other application may then make utilization of those capacities. This layer provides authorize security limitations on the capacities. Designers subsequently have full access to these segments and can utilize them or supplant them and execute them in their own particular manner [12].

Essential Components and the features of the components of the application Framework [12] are the following:

- **Activity Manager** manages the lifecycle of uses and gives a typical route back stack.
- **Content Providers** enables applications to get to information from different applications or to share their own information.
- **Telephony Manager** manages the voice approach the gadget. Voice brings in the application are gotten to utilizing the communication administrator.
- **Location Manager** manages the area data on the gadget utilizing Global Situating System (GPS) or a mobile phone tower.
- **Resource Manager** manages the assets that are not some portion of the code like designs, picture documents and so on.
- **View System** helps in working on an application, including content boxes, catches and so forth.
- **Notification Manager** enables all applications to show any client cautions on the status bar of the gadget.

### 3.2.3 Application Libraries

The layer beneath the Android Application Framework is the Android's local libraries. This layer uncovered the diverse capacities through the Application Framework. This layer can be managed as a game plan of headings that enables the contraption to manage different sorts of data. For instance, the playback and recording of any sound,



video, and picture positions is upheld by the media system library. C/C++ is used as the programming language for those cases [13].

The essential local libraries [12] are:

- **The Surface manager** handles the off-screen buffering for the window administrator from the structure layer. Off-screen buffering suggests that illustrations can't be drawn specifically onto the screen yet are put into an off-screen cushion. There it is joined with different illustrations to frame the last screen which the client can see. The straightforwardness of windows is accomplished on account of this off-screen support.
- **The OpenGL** is another library. The GL means Graphic Library. It is mostly utilized for rendering the substance of 2D or 3D illustrations to the screen.
- **Media Framework** library underpins diverse media codecs permitting the recording and playback of different media designs (sound, video, picture, and so on.)
- **FreeType** is for the most part utilized for the ending of bitmap and vector textual style.
- **SSL** stands for Secure Sockets Layer. It is a cryptographic convention utilized for secure correspondence over the web.
- **SQLite** is lightweight social database mottor utilized for the information stockpiling purposes in Android.
- **WebKit** library gives instruments to perusing the site pages. It has an HTML renderer, a JavaScript Engine, and a treat chief. These devices encourage the showing of any web content inside an application.
- **libc** is a standard C Library tuned for installed Linux based gadgets.

#### 3.2.4 Android Runtime

The location of Android runtime is the layer where the local libraries are located. It contains two libraries:

- **The Dalvik Virtual Machine** bolsters various virtual machine forms per gadget, therefore, giving high productivity in low asset conditions. The Dalvik VM through the creation of numerous procedures likewise guarantees security, segregation of one application from another [12].
- **Core Java Libraries** are composed of Java Programming Language. These libraries give the vast majority of the usefu functionalities fundamental for building the applications. Some of these functionalities incorporate information structures, realistic help, access of the network, file and so forth [12].

### 3.2.5 The Linux Kernel Layer

The essential layer on which every other layer assembled is the Linux Kernel layer. It employs the Linux 2.6 Operating System to play out the greater part of the key framework administrations like process administration, memory administration, organizing, security settings, networking, record administration, and so on. It is this layer that cooperates with the equipment and incorporates the greater part of the basic equipment drivers.

Drivers are programming programs that control and speak with the hardware. For instance, for any gadget that has Bluetooth equipment in it, the part would contain a Bluetooth card that speaks to the Bluetooth driver. This layer additionally acts as a deliberation layer between the equipment and whatever remains of the product layers [12].

### 3.3 Integrated Development Environment (IDE)

This Android application was made utilizing Android studio 3.0 for Windows. It is the most recent official Integrated Development Environment (IDE) by Google. It is open source accessible under the Apache permit.

The reason the android studio is chosen over other like Eclipse is on account of, android studio as of now has ADT Plugin instruments incorporated into it while Eclipse does not. All up-to-date variants of the Android studio can be gotten from the official site. Android Studio consists of file sections described in the following subsections.

#### 3.3.1 Manifests File

AndroidManifest.xml is an XML file (eXtensible Markup Language) in the root directory. It contains various important information about Android application. Many important features of the manifest file [10] are given below:

1. Java Package naming is an important feature of Manifest file. Package name is used as unique identifier for Android applications.

2. The user permission declaration has been conducted using manifest. It helps to interact with other application. It declares the user permission for example Internet access. Write only or read-only permissions and some other information can be declared in this file.
3. Application components such as activities, services, broadcast receivers and content providers have been declared using manifest. These components identify the launching environment of an Android app.
4. It sets out suitable API for application and minimum API level has been selected by it.
5. Libraries such as Google Map Library have been declared in the manifest file.
6. Various features such as Camera, Screen touch, Bluetooth etc. services are conducted in Manifest.

### 3.3.2 Resources File

The visual presentation of an Android application is conducted by this part. It is not the main coding part. The resource file includes Audio files, Video files, images, menus, styles, colours, animations, layouts and so on. Layout files, colors, menus etc. are written in the XML file. The user interface is designed by using sets of those resources. On the other hand, the characteristics of app, screen sizes, screen configuration are written and maintained using the XML file [15]. The resource file is situated in /res folder in the root directory. /res folder contains following resources:

- **Drawable Folder** contains Image files, different XML files are placed in this folder.
- **Layout file** consists of XML based files. Basically the user interface has been designed in this section.
- **Raw file contains** different types of raw files are placed in this folder for example Audio file, video file, database file and so on.
- **Other files** such as menu, mipmap, values are also part of resource file. These are also essential to UI design of an application.

### 3.3.3 Java File

The package file contains the Java file package. It defines interfaces and classes for the Java virtual Machine. Java file contains Java classes, enumeration classes and singleton classes and interface and annotation types. Java programming languages

are used to define classes, objects, functionalities, methods to run the app. Basically, the app is conducted by this section. We can call it as the brain of the app.

### 3.3.4 Gradle file

Gradle is a JVM based tool. It is a kind of a building system that combines all resource files, .Java or .xml file into one and convert into an APK file. It has a significant role in building systems and making a compressed file from group of files. The Groovy code is used to automate the tasks in `build.gradle` file in the Android studio project file. The New build system makes reusable code and helps good IDE integration [10].

### 3.4 Installation of Android studio

An Android application development process starts with the installation of IDE. For Android studio establishment, Java improvement pack (JDK) rendition eight is required if not introduced some time recently. When it is prepared, it explores the official Android development page at <https://developer.Android.com/studio/index.html> in figure 3 [16].

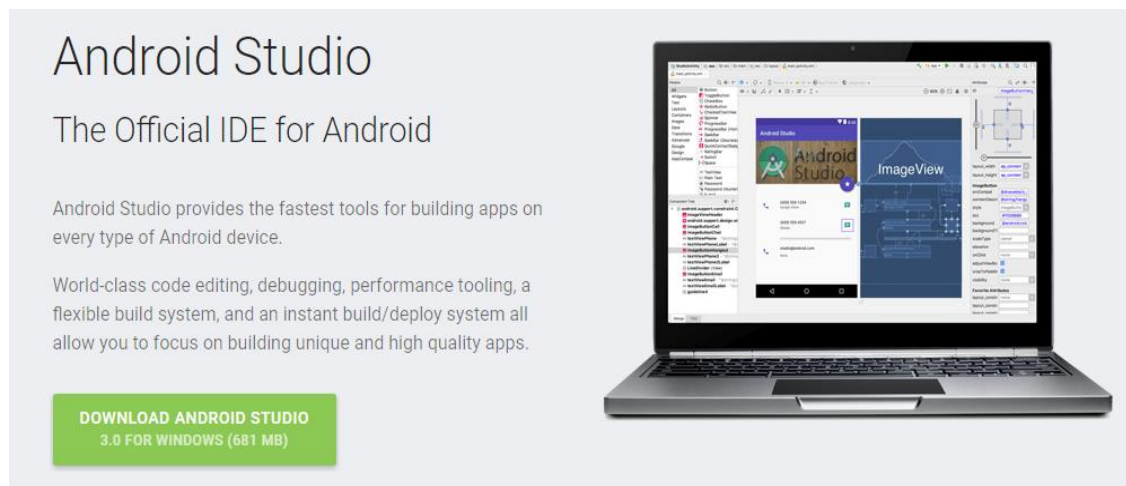
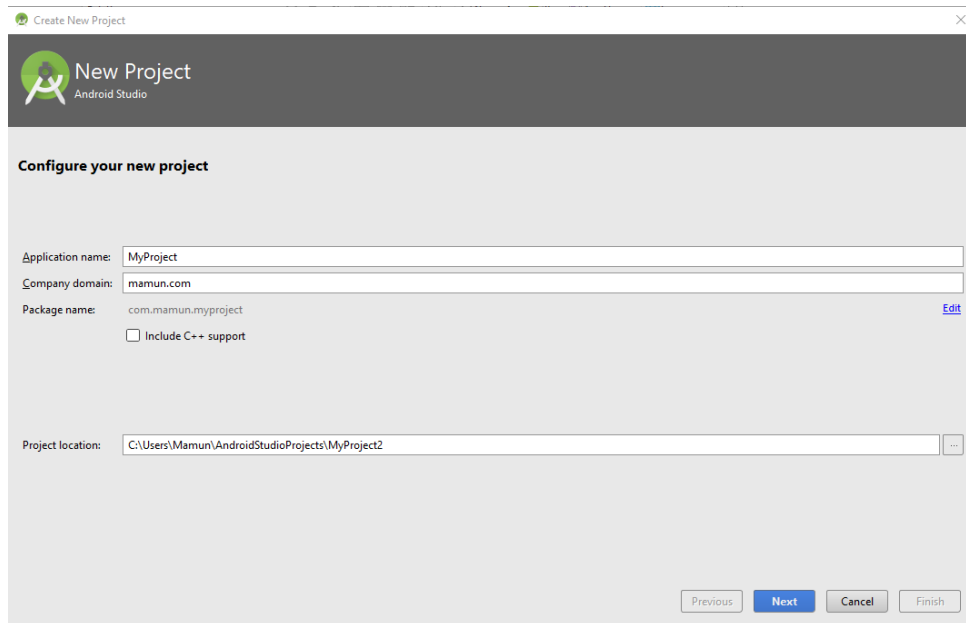


Figure 3. Android Studio Installation. Source: <https://developer.Android.com/studio/index.html>

To begin with, the .exe file is downloaded from the connection above and afterward the executable document is dispatched. New project file is created after finishing all the required steps.

### 3.4.1 Creating new project

In order to create a new project (figure 4) in android studio the following steps should be done, File → New → New Project.



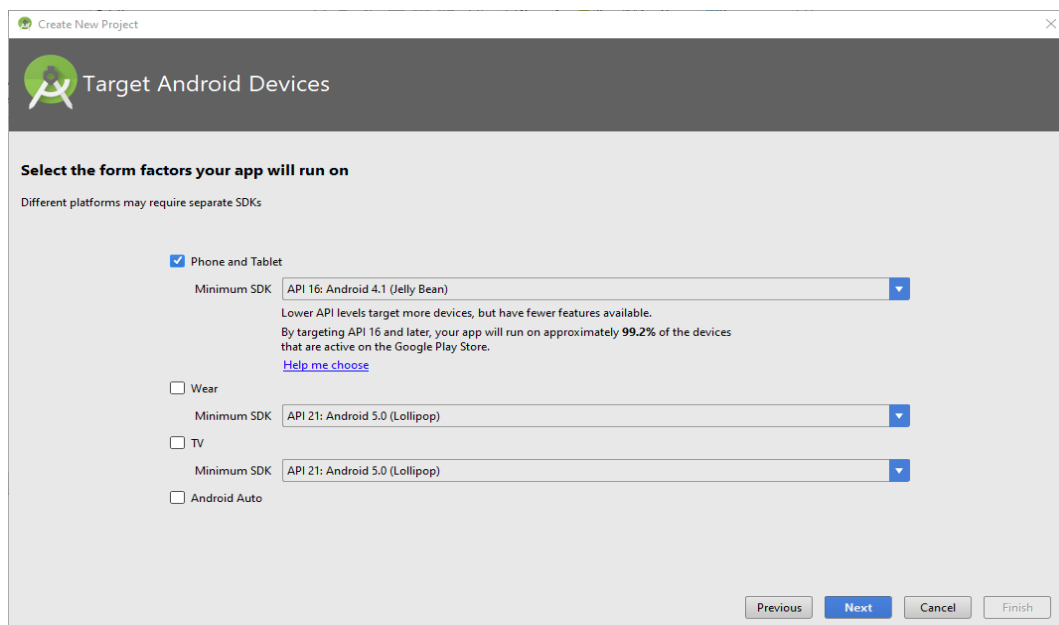
The screenshot shows the 'Create New Project' dialog in Android Studio. The dialog is titled 'New Project' and 'Android Studio'. It contains the following fields and options:

- Application name:** MyProject
- Company domain:** mamun.com
- Package name:** com.mamun.myproject (with an [Edit](#) link)
- Include C++ support:**
- Project location:** C:\Users\Mamun\AndroidStudioProjects\MyProject2

At the bottom of the dialog, there are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'.

Figure 4. New Android project and naming the project

Application name and package name have to be inserted (figure 4). Package name will be created automatically for example here com.mamun.myproject.



The screenshot shows the 'Target Android Devices' dialog in Android Studio. The dialog is titled 'Target Android Devices' and 'Android Studio'. It contains the following options and settings:

- Select the form factors your app will run on**  
Different platforms may require separate SDKs.
- Phone and Tablet**  
Minimum SDK: API 16: Android 4.1 (Jelly Bean)  
Lower API levels target more devices, but have fewer features available.  
By targeting API 16 and later, your app will run on approximately 99.2% of the devices that are active on the Google Play Store.  
[Help me choose](#)
- Wear**  
Minimum SDK: API 21: Android 5.0 (Lollipop)
- TV**  
Minimum SDK: API 21: Android 5.0 (Lollipop)
- Android Auto**

At the bottom of the dialog, there are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'.

Figure 5. Selecting form factor to run the app

In this step, minimum SDK version has to be selected (figure 5). It is wise to select API level 16 or above because lower API level use more devices. Approximately 99.2% Android devices are using apps targeting API 16 or later.

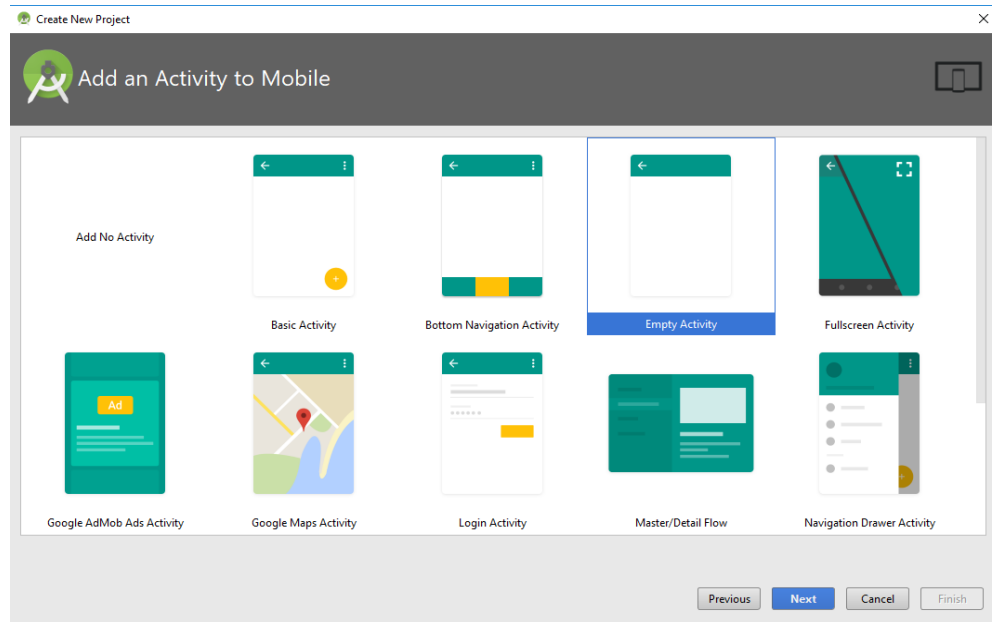


Figure 6. Default Activity

In the next step, the activity state can be chosen and renamed it. An empty activity (figure 6) have been used in this app. Now activity and its layout file have to be named. It can be kept it as MainActivity as shown in figure 7. After that, Finish is clicked.

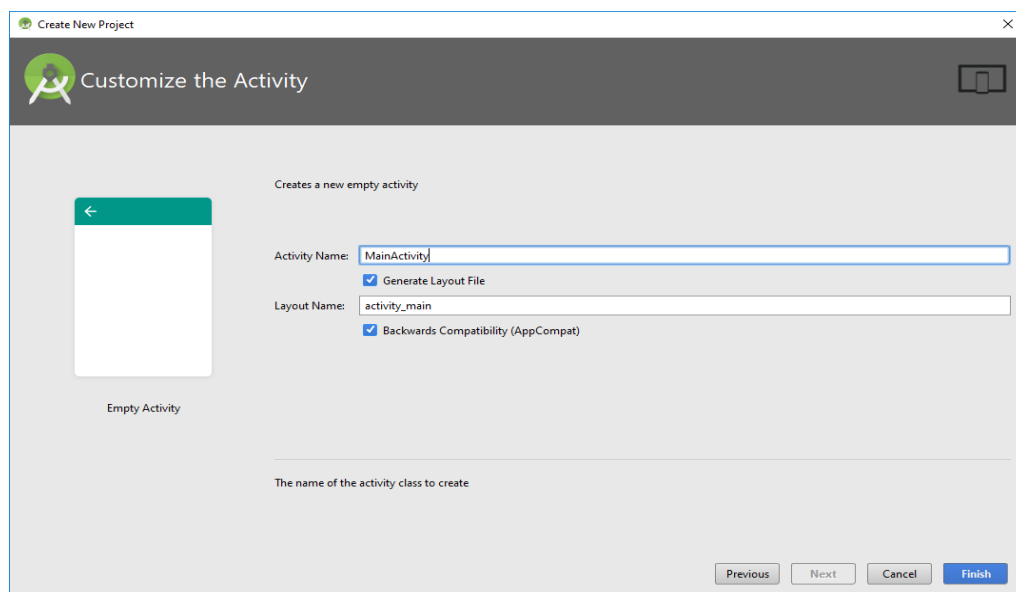


Figure 7. Customizing the Activity

### 3.4.2 Creating and running the app in emulator

Once the project is successfully created, by default a Hello World! view is created. Now an emulator can be created and the app can be run on the emulator.

The emulator takes a long time to start; once it is started it should not close it. Every time the developer makes changes to the code, the App is updated at the same time on emulator. Before application is run on an emulator, an Android Virtual Device (AVD) definition has to be made. An AVD definition indicates the attributes of an Android mobile, tablet, Android Wear, or Android TV gadget that are needed to mimic in the Android Emulator. An AVD Definition has been made by the following steps: Dispatch the Android Virtual Device Manager by choosing Tools→ Android→ AVD Manager, or by tapping the AVD Manager symbol in the toolbar. Then it needs to click on Create Virtual Device in the virtual device screen. In the Select Hardware screen, select a device, for example, Pixel, and after that snap next. In the System Image screen, click Download for one of the suggested framework pictures. Consent to the terms to finish the download. After the download is finished, select the framework picture from the run-down and snap next. On the following screen, leave all the arrangement settings as they are and click Finish. Back in the the Virtual Devices screen, select the gadget that was made and snap launch this AVD in the emulator. While the emulator begins up, close the Android Virtual Device Manager window and come back to the venture to run the application: Once the emulator is booted up, tap the application module in the Project window and afterward select Run →Run (or snap Run in the toolbar). In the Select Deployment Target window, select the emulator and snap OK. Android Studio introduces the application on the emulator and begins it.

### 3.4.3 Running the app on Android device

In order to install and run the app in a real Android-powered device, the following steps should be followed. The device and the development machine have to be plugged in with a USB cable. An appropriate USB driver might be needed in case of Windows for the device.

A mobile device has to be connected to the development machine with a USB cable. If development machine is Windows might need to install the USB driver on a mobile device.

USB debugging can be done by Settings → Developer options. Run the application from Android Studio as follow:

In Android Studio, tap the application module in the Project window and after that select Run → Run (or snap Run in the toolbar). In the Select Deployment Target window, select your gadget, and snap OK. Android Studio introduces the application on your associated mobile device and begins it.



## 4 Know Finnish app development techniques

This chapter provides the necessary information about the Android application development processes, design techniques, and core programming. Development techniques mainly consist of the following three parts:

- Manifest file
- Graphical The user Interface (GUI)
- Application Controller

Now the discussion will be about the user Interface of this application, how it works and what the development techniques of this application are.

The project file contains the MainActivity.Java file section. It was created during project setup under src(source)→package name (com.mamun.myproject)→ MainActivity.Java (figure 8).

Activity\_main.xml will be located under res (Resources) →layout→ activity\_main.xml. Double-click is required on activity\_main.xml (figure 8) to open it.

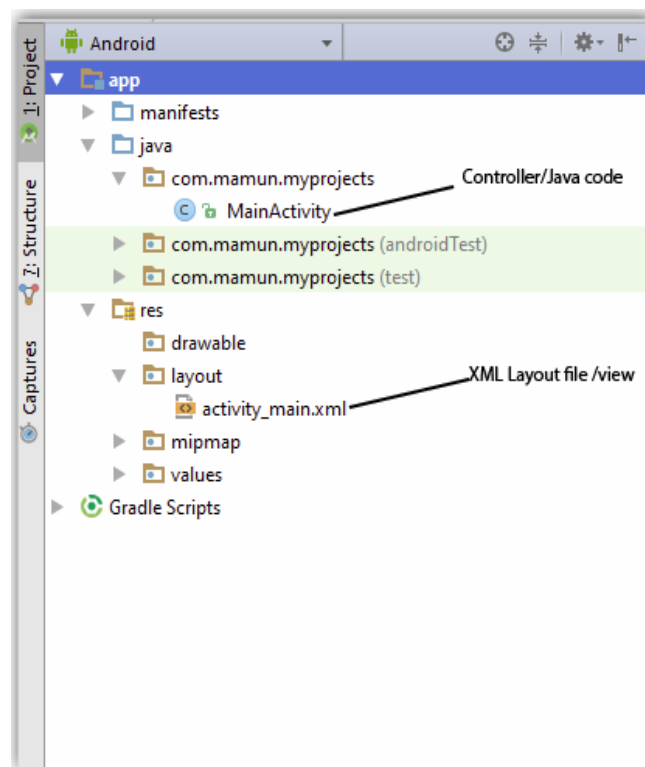


Figure 8. Location of Java and XML file

## 4.1 Manifest File

In this manifest file, the package name (com.mamun.myproject) is included that is the unique identifier for Android application. It declares the user permission, for example, Internet access. Write only or read-only permissions are also declared in the manifest file. Other features are also conducted in Manifest (listing 1).

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:Android="http://schemas.Android.com/apk/res/Android"
    package="com.mamun.myproject">
    <uses-permission Android:name="Android.permission.READ_EXTERNAL_STORAGE"/>
    <uses-permission Android:name="Android.permission.WRITE_EXTERNAL_STORAGE"
/>
    <uses-permission Android:name="Android.permission.INTERNET" />
    <application
        Android:allowBackup="true"
        Android:hardwareAccelerated="false"
        Android:largeHeap="true"
        Android:icon="@mipmap/ic_launcher"
        Android:roundIcon="@mipmap/ic_launcher_round"
        Android:supportsRtl="true"
        Android:label="@string/app_name"
        Android:theme="@style/AppTheme">
        <activity Android:name=".MainActivity">
            <intent-filter>
                <action Android:name="Android.intent.action.MAIN"/>
                <category Android:name="Android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <activity Android:name=".Dictionary_Page2"></activity>
        <activity Android:name=".Add_Word"></activity>
        <activity Android:name=".Word_List"></activity>
        <activity Android:name=".Quiz_Easy1"></activity>
        <activity Android:name=".Listmaking"></activity>
        <activity Android:name=".SentencePuzzle"></activity>
        <activity Android:name=".WelcomeActivity"></activity>
        <activity Android:name=".WordBankActivity"></activity>
        <activity
            Android:name=".GameActivity"
            Android:label="@string/title_activity_game" >
        </activity>
        <activity
            Android:name=".webview"
            Android:theme="@Android:style/Theme.NoTitleBar" />
        <activity
            Android:name=".DictionaryListActivity">
        </activity>
        <activity
            Android:name=".WordDefinitionDetailActivity">
        </activity>
    </application>
</manifest>
```

Listing 1. Android Manifest File of Know Finnish Android Application.

## 4.2 Application design and control

The user interface of the application is how the user's will see this application. The user will easily understand the user interface of the application. They can easily understand the functionality of the application.

The User Interface of the application consists of the following clickable image view and buttons (figure 9). After clicking the image view and buttons the user will go to next activity to achieve their goal.

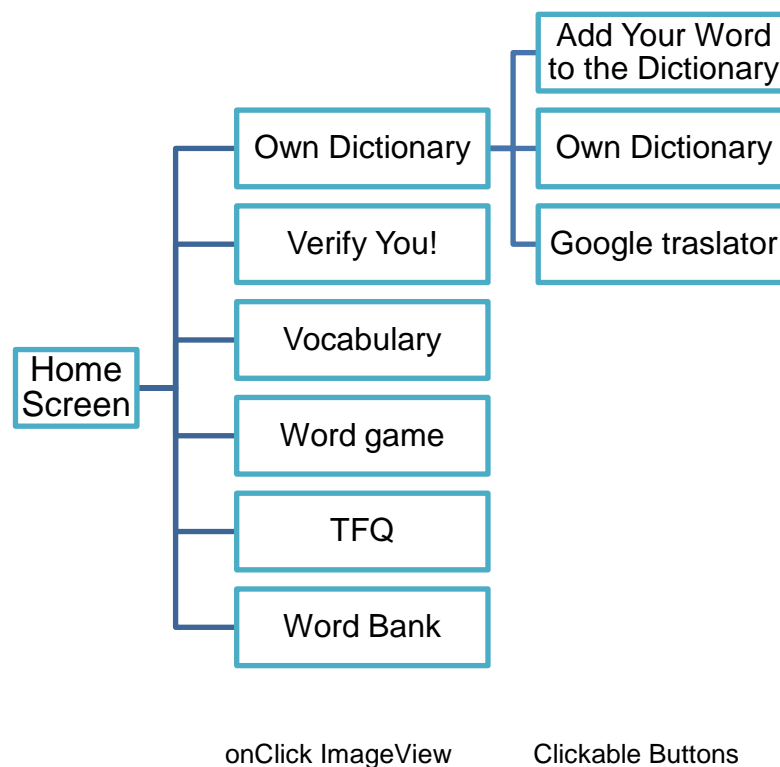


Figure 9. Know Finnish Graphical UI at a glance

There are two views available for layout file graphical layout (Design) and another one is XML view (Text). Switching to XML view takes place by clicking the tab at the bottom of Pallete Panel. the code for "Hello World!" text can be seen that will appear on the screen in the XML format (listing 2).

```

<TextView
    Android:layout_width="wrap_content"
    Android:layout_height="wrap_content"
    Android:text="@string/hello_world"
    tools:context=".MainActivity" />

```

Listing 2. Hello World! Code in xml view.

Now this file needs to be edited to create a view of our MainActivity (i.e. figure 8). This application requires many screens like the main Screen. It will start by adding ImageView buttons to view which will take us to next activity.

Now ImageView and add an id name "button" can be added by deleting the code for TextView of with Hello World (listing 3). This ImageView will act as a button. When the user clicks on it, another activity will open. The following code have been added to the XML file to do so.

```

<ImageView
    Android:id="@+id/button"
    Android:layout_width="140dp"
    Android:layout_height="140sp"
    Android:layout_gravity="left"
    Android:layout_marginLeft="41dp"
    Android:layout_marginStart="41dp"
    app:srcCompat="@drawable/owndictionary"
    tools:ignore="ContentDescription,RtlHardcoded"
    Android:layout_above="@+id/game"
    Android:layout_alignParentLeft="true"
    Android:layout_alignParentStart="true"/>

```

Listing 3. Adding ImageView using XML code.

Another important code which need to be added is the background setup. After preparing a suitable background picture, `Android:background="@drawable/back1"` is required to be coded inside main relative layout inside activity\_mail.xml file.

Similarly, different color can be added by adding hash color code or similar formulas. `Android:background="#FFFFFF"`. This code will make the background white.

#### 4.2.1 Home Screen

The main features of this application are summarized in Table 1. These features will be shown on the home screen of the application.

Table 1. Main features of Know Finnish Application

Features	Brief Description
Own Dictionary	The user can easily update their words in this dictionary and much more.
Verify You!	The user can verify their Finnish knowledge by using this feature. In this feature, the user has to choose one correct answer from three. Their performance will be assessed by the score. Basically, questions will come from their own dictionary so that they will verify whether they learned properly or not.
Vocabulary	Vocabulary part is arranged with a large number of words using images.
Word game	The user will know the words easily during playing this game. Words appear on the screen with the touch with the touch of the user's finger on screen and words with the image will act like animation.
TFQ	True or false questions. The questions will be either about words or sentences.
Word Bank	This is basically English to Finnish Dictionary. This dictionary contains approximately 2000 words.

The home screen of the application contains a logo, a motto, and six clickable image views. Figure 10 shows the screenshot of the home screen of the application.



Figure 10. Home page of the application

The user will get good knowledge about the Finnish language by using this application. The logo has been designed using Adobe Illustrator CS6.

The motto of the application is “Know Finnish, Develop Yourself”. We believe that the user will develop their Finnish learning skills by using this application.

A logo has been placed at the top of the page from drawable folder under ImageView. The motto of the application has been set up using a text view and two views with red background (figure 11).



Figure 11. Logo and Motto of the app

The homepage controller consists of several onClick ImageView so that user can click on it. Every ImageView as a unique ID which was declared in the layout file and retrieved from MainActivity.Java (listing 4) and performs the following steps.

- Creating the ImageView objects in scope of class.
- Initializing and adding reference to corresponding ImageView widget in the view file in onCreate method.
- Adding onClickListener. Navigation between screens of an app (activities) and parameters/data is usually transferred using intents.

```

package com.mamun.myproject;

import Android.content.Intent;
import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.view.View;
import Android.view.Window;
import Android.widget.ImageView;

public class MainActivity extends AppCompatActivity {
    ImageView btn1, btn2, btn4, sentence, gameact, wordbnk ;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this.requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activity_main);

        btn1 = (ImageView) findViewById(R.id.button);
        btn2 = (ImageView) findViewById(R.id.button2);

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, Dictionary_Page2.class));
            }
        });
        btn2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(MainActivity.this, Quiz_Easy1.class));
            }
        });
    }
}

```

```

    }
    });

    btn4 = (ImageView) findViewById(R.id.button4);

    btn4.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MainActivity.this, Listmaking.class));
        }
    });

    sentence = (ImageView) findViewById(R.id.sentenceButton);

    sentence.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MainActivity.this, SentencePuzzle.class));
        }
    });

    gameact = (ImageView) findViewById(R.id.game);

    gameact.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
        }
    });

    wordbnk = (ImageView) findViewById(R.id.wordbankbutton);

    wordbnk.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MainActivity.this, WordBankActivity.class));
        }
    });
}
}

```

Listing 4. MainActivity class to go to next activities using Intent.



Before having a look into the navigation between Screens (Activities), first a new Activity must be created and added to the application Manifest file. The process of adding the controller class will given below :

- Right-Click on the package name (com.mamun.myproject) and go to New→Class (Package name is same as set up during the creation of the the project).
- Using CamelCase convention and adding the name of the Class. In Java classes, the first letter should always be capital e.g. Listmaking.**class**.
- In superclass adding: Android app activity, this will import and inherit the required classes for this Java file to be an Android activity.

#### 4.2.2 Own Dictionary

It is a personal dictionary. Once the user clicks on the “Own Dictionary” view (figure 12), it will move to the next activity (figure 13). The user can add their words in this dictionary (figure 14). It will save their words to the database and the user can retrieve the words for further use (figure 16). Google translator button shown in figure 13 will direct the next activity (figure 17). It is basically a web view activity. The user can translate any words.

The Own Dictionary Screen consists of 3 buttons and one search box.

- Add your words to the Dictionary (button): After clicking on this button the user will go to new screen. The user can add the word to the Own Dictionary. The ultimate goal of this section is to store the word for further use.
- Own Dictionary (button)
- Google Translator (button)
- Word search from Own Dictionary is a search box. The user can find the word in the Own Dictionary.



Figure 12. Home Screen



Figure 13. Own Dictionary next activity Screen

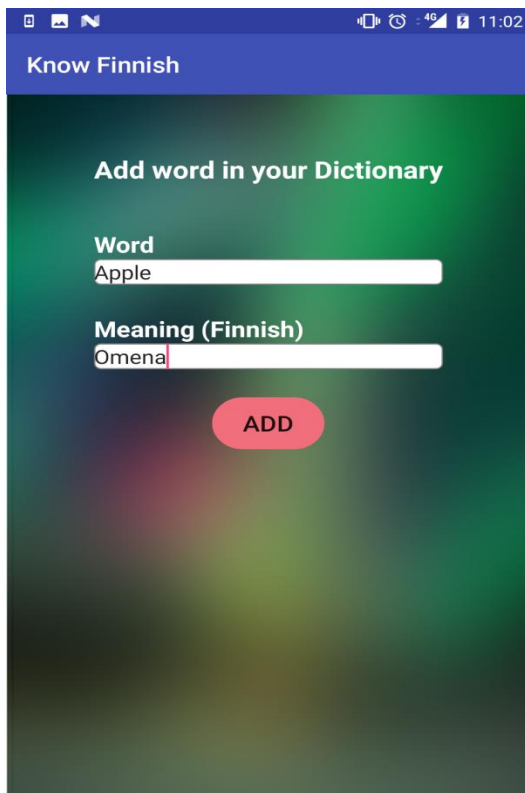


Figure 14. Add word to your dictionary

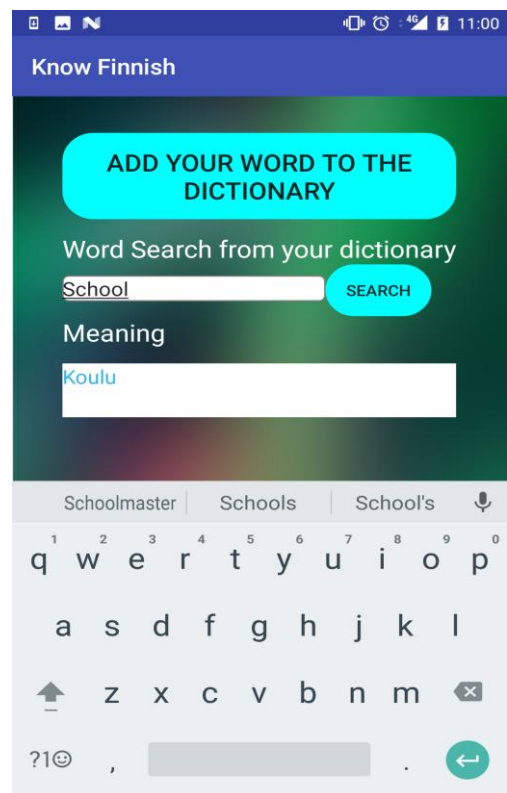


Figure 15. Word search from Own Dictionary

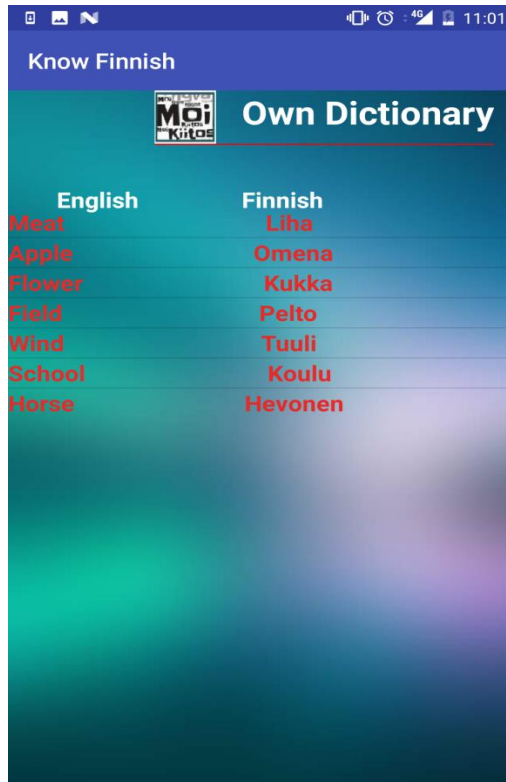


Figure 16. Own Dictionary Screen

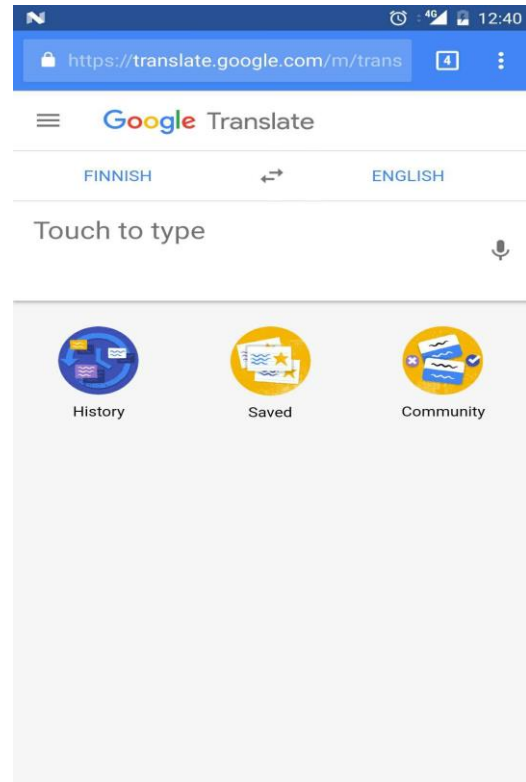


Figure 17. Google Translator

#### 4.2.3 Own Dictionary Controller

When the user clicks on “Add your words to the Dictionary” button, it navigates to the new activity class called `Add_Word.class`. The new intent is used to navigate from home screen to the Own Dictionary page (listing 5).

```
add_word.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(Dictionary_Page2.this, Add_Word.class));
    }
});
```

Listing 5. Example of passing data from dictionary page to add word page using Intent.

After putting the word by the user and its meaning and clicking on “ADD” button, words will be saved on Own dictionary. Words and its meaning have been added with the help of `VideoDetails` class and `DatabaseHelper` class. Word and method will be added by `getText()` and `toString()` method as well as with help of `DatabaseHelp` class (listing 6).

After putting the word by the user in the editText box and clicking on the search button the meaning of that word will appear on the same screen. The word will be searched using DatabaseHelper class from the Own Dictionary.

```
DatabaseHelper db;
db = new DatabaseHelper(Dictionary_Page2.this);
search_word=db.getVideofromTable(edit_search.getText().toString(), DatabaseHelper.TABLE_VIDEOS);
db.closeDB();
edit_show.setText(search_word.meaning);
```

Listing 6. Database helper class for word search function from the dictionary.

The main objective of the Own Dictionary is to make a list of words that the user will add. It is controlled by Own Dictionary controller. Words and their meanings will be added as String arrayAdapter. CustomListAdapterDialog class is used to make the word list (listing 7).

```
CustomListAdapterDialog clad = new CustomListAdapterDialog(Word_List.this, word);
lv.setAdapter(clad);
}
private class CustomListAdapterDialog extends ArrayAdapter<String> {
private Context context;
LayoutInflater inflater;
public CustomListAdapterDialog(Context con, ArrayList values) {
super(con, R.layout.word_list_style, values);
this.context = con;
inflater = (LayoutInflater) context
.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
}
}
```

Listing 7. CustomListAdapterDialog class to make wordlist.

Google translator has been loaded from the URL(<https://translate.google.com/>) in Webview class. The main objective of this translator is to find a new word that is not included in the dictionary part of this application. The following code is used to get the URL (listing 8) [17].

```
webView.getSettings().setJavaScriptEnabled(true);
webView.loadUrl("https://translate.google.com/");
```

Listing 8. Google translator using loadURL.

Full source code of Own Dictionary controller classes are found in Appendix 1, listing 12 - 17.

#### 4.2.4 Verify You!

The “Verify You!” screen (figure 19) will appear when the user presses the “Verify You!” image view on the Home Screen (figure 18). The user will get the questions and possible answers. The user will get points for right answers. Those questions and possible answers come from their own dictionary. The user can verify them using those quizzes.



Figure 18. Home Screen

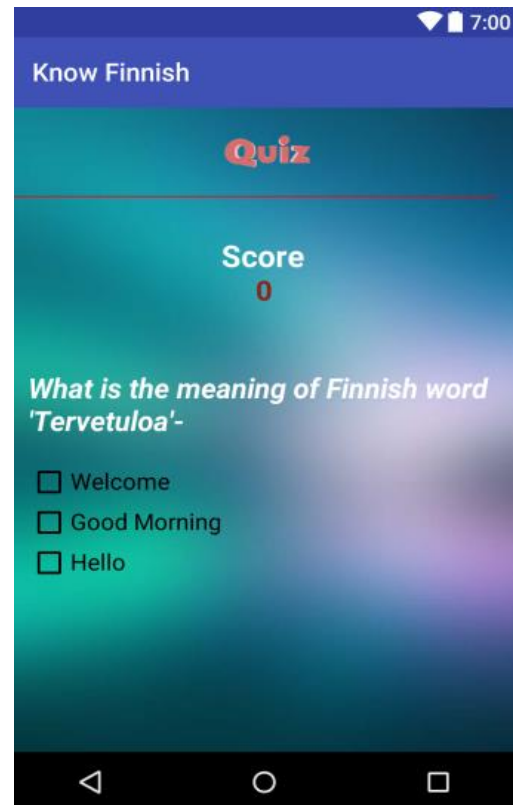


Figure 19. Verify You! Screen

#### 4.2.5 Verify You! Controller classes

“Verify You!” section is designed using random questions and answers from the Own Dictionary. Three different functionalities have been set up in this section. Questions, possible answers, and score. Questions and possible answers have been randomly taken from the Own dictionary. The user stored the words in the Own Dictionary. Full source code of “Verify You!” classes are found in Appendix 2, listing 18.

Quiz\_Easy1 class is the main class to call questions, answers, and score of Verify You! section. The questions and answers have been set by random() method is Database-

Helper class of the database file (listing 16). For loop is used to make the array lists inside random() method. Three check boxes have been made according to possible answers and shuffled with possible answers. Scores has been counted inside get\_quistion() method. Random() method also called inside get\_quistion() method (listing 9). Finally, get\_quistion() method is called inside the onCreate method.

```

public void random() {
    list = new ArrayList<>();
    for (int j=0; j<3; j++) {
        list.add(new Integer(j));
    }
    Collections.shuffle(list);
    String[] data={search_word1.word,search_word2.word,search_word3.word};
    checkBox1.setText(data[list.get(0)]);
    checkBox2.setText(data[list.get(1)]);
    checkBox3.setText(data[list.get(2)]);
    checkBox1.setChecked(false);
    checkBox2.setChecked(false);
    checkBox3.setChecked(false);
    checkBox1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(checkBox1.getText().toString().equals(search_word1.word)){
                point++;
            }else {
                Toast.makeText(getApplicationContext(),"Error",Toast.LENGTH_LONG).show();
            }
            get_quistion();
        }
    });
}

```

Listing 9. random() method to get the questions from array list .

#### 4.2.6 Vocabulary

The user can develop their knowledge using the vocabulary section. This part is arranged with a large number of words with images (figure 21).



Figure 20. Home Screen



Figure 21. Vocabulary Screen

#### 4.2.7 Vocabulary Controller

After clicking on the vocabulary image view on the Home screen (figure 20), the user will get a list of words with images.

Listmaking class and CustomListView class are two classes that result list of words with image in the vocabulary part. Numbers of Finnish words, sentences, and suitable images has been declared as array in Listmaking class. CustomListView acts as adapter and holds the formula so that the new adaptor in Listmaking connects those words and images and display on the screen (listing 10). Here the array adapter acts like a connector or bridge [18]. Full source code of vocabulary controller class is found in Appendix 3, listing 19.

```

public class Listmaking extends AppCompatActivity {

    ListView list;
    String[] finnishwords = {"Man-
go", "Vesimeloni", "Banaani", "Viinirypäleitä", "Kiivi", "Avokado" };
    String[] desc = {"Tämä on mango", " Tämä on Vesimeloni", "Tämä on banaani", "
Tämä on viinirypäleitä", "Tämä on kiivi", "Tämä on Avokado" };
    Integer[] imageid =
    {R.drawable.mango, R.drawable.watermelon, R.drawable.banana, R.drawable.grapes, R.
drawable.kiwi, R.drawable.avocado };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.wordlist);

        list= (ListView) findViewById(R.id.listview12);
        CustomListView adater = new Custom-
ListView(Listmaking.this, finnishwords, desc, imageid);

        list.setAdapter(adater);

    }
}

```

Listing 10. Adapter setting to get the words from the list.

#### 4.2.8 Word game

This game is designed to amaze the user. The user can learn words while playing this game. Large numbers of common words with images have been presented to the user. After pressing the fruit image button as shown in figure 22, the game will start. When the user touches the screen different types of words will appear randomly (figure 23). Those words have been designed nicely using Adobe Illustrator CS6.



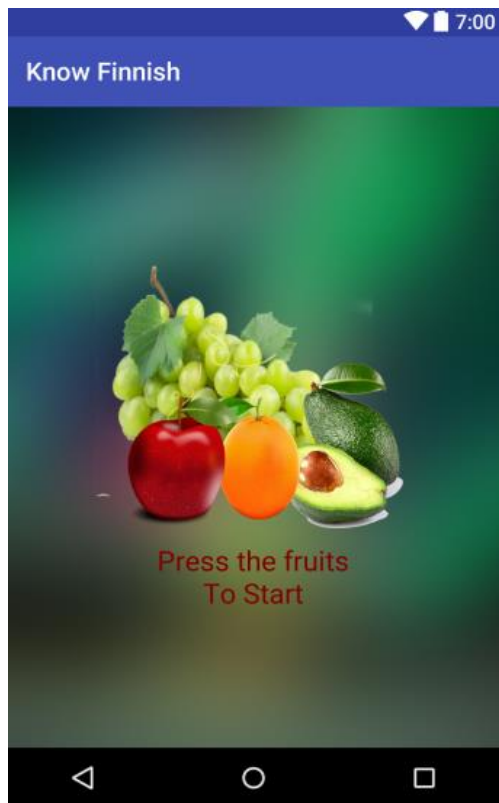


Figure 22. Game start button activity screen

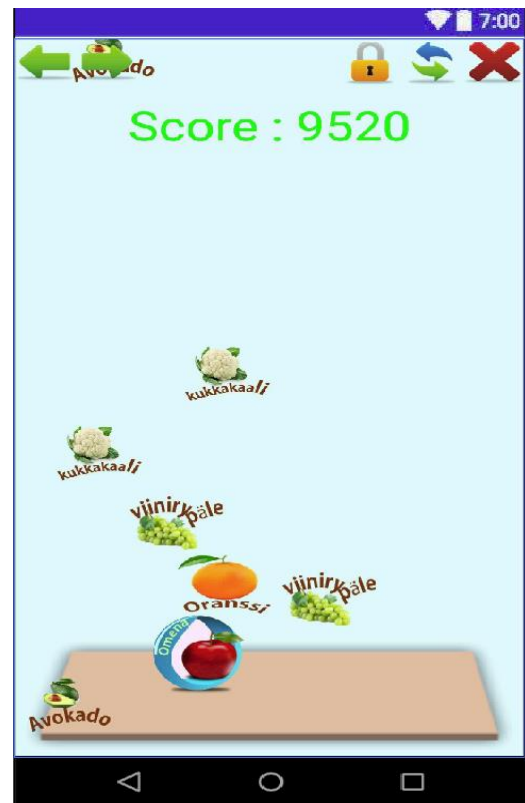


Figure 23. Word Game Screen

#### 4.2.9 Word Game Controller

A new intent has to be added to move from WelcomeActivity class to GameActivity class. When the user presses on the fruit it will change its transparency using setAlpha.

SurfaceView has been created in the GameView class. A drawing Thread is necessary to pass the GameView and context. It will act like drawing canvas. The display will be updated by using updateDisplay() method. The initializeDisplay() method will help to understand the display dimensions of the display window. The main target here is to make full-screen display. All possible words will be shown in Robot class. In this class, the array list is included so that it will make a new array list. Words will be displayed and showed on the screen according to giveResizedRobot method. The random() method is used to bring the words randomly on touchPoint. The accelerometer is used to bounce the words. In GameActivity class, SensorManager method and SensorEventListener have been added and used to make this state so that it can bounce the words.

VelocityTracker is used to set the velocity of the falling words. Furthermore, there are a few more functions that have been added for example game pause screen making, restart button function, stop button function and so forth [20]. Full source code of game activity controller classes is found in Appendix 4, listing 20 - 27.

#### 4.2.10 TFQ

TFQ is short for True-False Questions. Finnish words and sentences will appear in correct and incorrect form. The user has to choose the correct answer so that they will get points.



Figure 24. Home Screen

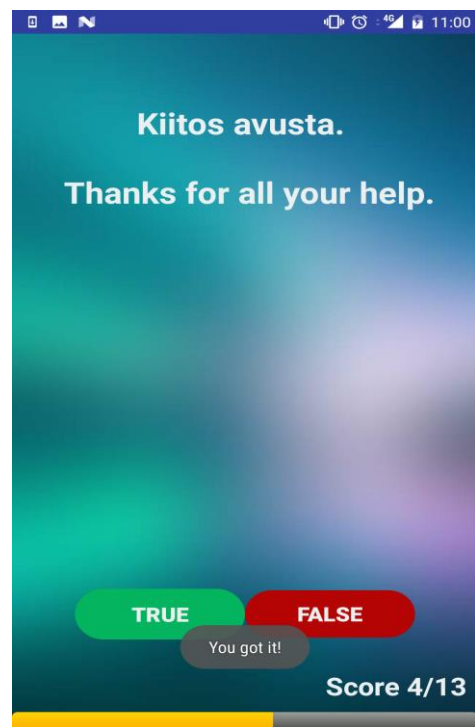


Figure 25. TFQ Screen

#### 4.2.11 TFQ Controlling

The list of True False Questions is arranged in a private ArrayList. The updateQuestion() method is used to update questions from the array list and the checkAnswer() method checks the answers that have been chosen by the user (listing 11). Another functionality of the checkAnswer() method is to show the toast whether right or wrong has been chosen by the user. If-else statement is used to show the toast.

The score has been set up in `updateQuestion()` method and called from `onCreate` method, so that the score key and progress bar progresses with the update of the questions and answers given by the user.

```

        mTrueButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                checkAnswer(true);
                updateQuestion();
            }
        });

        mFalseButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                checkAnswer(false);
                updateQuestion();
            }
        });
    }

    private void updateQuestion() {
        mIndex = (mIndex + 1) % mQuestionBank.length;
        if (mIndex == 0) {
            AlertDialog.Builder alert = new AlertDialog.Builder(this);
            alert.setCancelable(false);
            alert.setMessage(" You scored " + mScore + " points! ");
            alert.setPositiveButton("Close ", new DialogInterface-
face.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    finish();
                }
            });
            alert.show();
        }

        mQuestion = mQuestionBank[mIndex].getmQuestionID();
        mQuestionTextView.setText(mQuestion);
        mProgressBar.incrementProgressBy(PROGRESS_BAR_INCREMENT);
        mScoreTextView.setText("Score " + mScore + "/" + mQuestionBank.length);
    }

    private void checkAnswer(boolean the userSelection) {

        boolean correctAnswer = mQuestionBank[mIndex].ismAnswer();

        if (the userSelection == correctAnswer) {
            Toast.makeText(getApplicationContext(), R.string.correct_toast,
Toast.LENGTH_SHORT).show();
            mScore = mScore + 1;
        } else{
            Toast.makeText(getApplicationContext(), R.string.incorrect_toast,
Toast.LENGTH_SHORT).show();
        }
    }

    public void onSaveInstanceState(Bundle outState){

```

```

super.onSaveInstanceState(outState);
    outState.putInt ("ScoreKey", mScore);
    outState.putInt ("IndexKey", mIndex);
}
}

```

Listing 11. updateQuestion() method is used to update the question from Question Bank.

The TrueFalse class is used for the get and set method. The questions have been updated using the get and set method that is called from SentencePuzzle Class. Full source code of TFQ controlling activity classes are found in Appendix 5, listing 28-29.

#### 4.2.12 Word Bank

Word bank is an English to Finnish dictionary. The user can find their expected words from this word bank. Three different classes have been used to make this word bank.



Figure 26. Home Screen



Figure 27. WordbankActivity

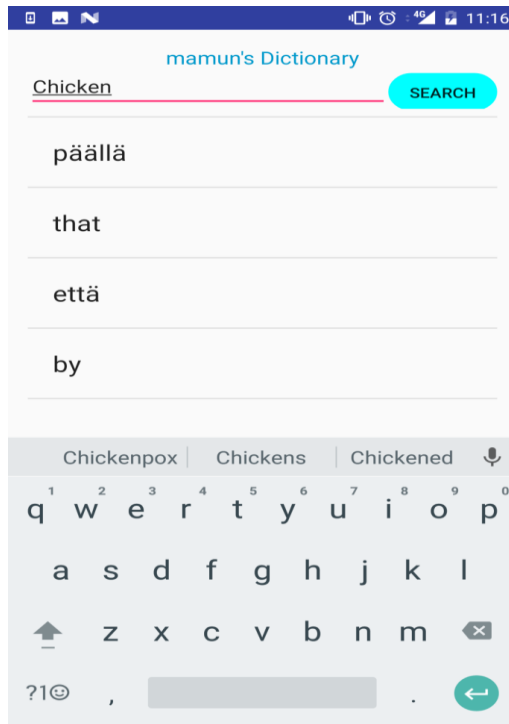


Figure 28. Word Search Activity

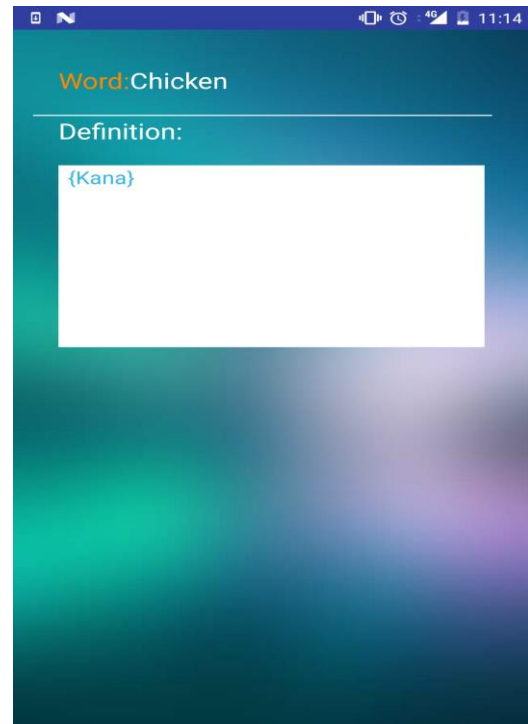


Figure 29. Word definition Screen Activity

#### 4.2.13 Word Bank Controller

In the first screen, the user has to put their name in an edit text box. The next screen will appear with that name and search option from the dictionary.

Functionalities of the WordbankActivity class are started by passing the username as getString method and putString method. WordDefinition class and WordDefinitionDetails class are used to define words and their meaning. When the user searches the word, WordDefinition class passes the word from the dictionary database that is stored in the raw file. StringBuilder object defines to match the word and pass the meaning from the database. Full source code of word bank controller classes is found in Appendix 6, listing 30-36.

## 5 Evaluation of the App

An online evaluation form was designed to evaluate the “Know Finnish” app. The main goal of the evaluation is to get feedback from real people who want to learn Finnish. The “Know Finnish” app was sent to the participants and they were invited to fill up the form. The form was distributed to the participants using sharing media i.e facebook, email and google+. More than 50 people were participating within a few days. They gave their evaluation and comments about the app. The evaluation form is found in Appendix 7, figure 31.

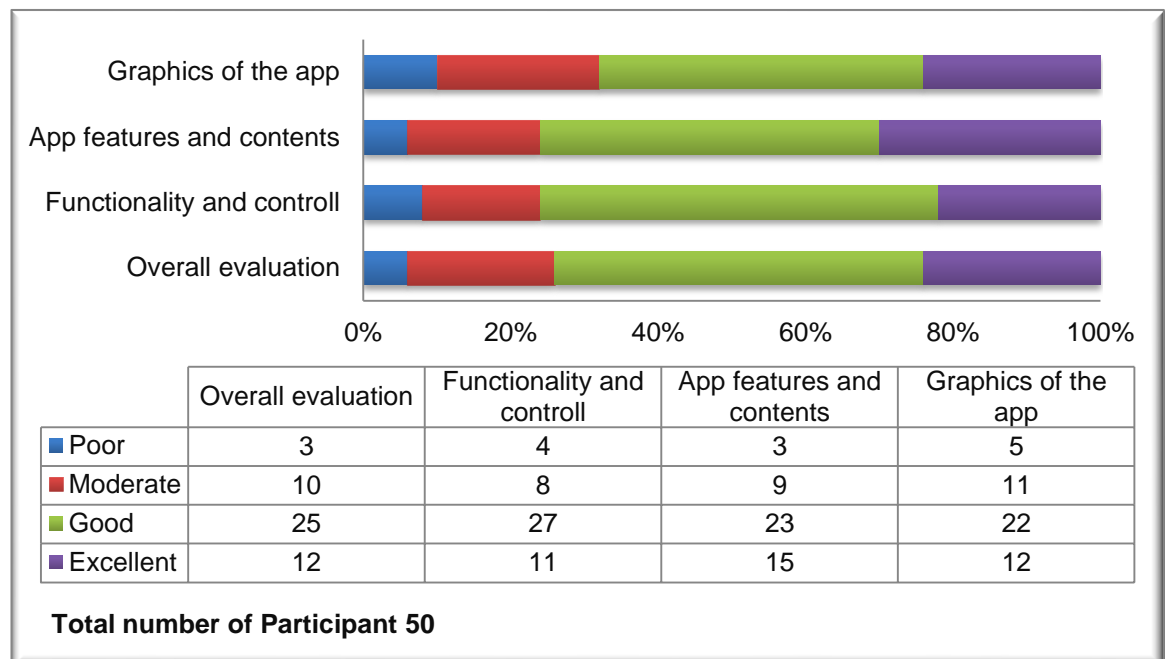


Figure 30. Evaluation of the Know Finnish the app

Source: <https://form.jotformeu.com/73125684817362>

The total number of participants to evaluate the app is about 50 people. Most of the people (25 people) think that the overall features of “Know Finnish” App are good. However, 12 people think that this is an excellent app to learn Finnish. On the other hand, 3 people reflected that it is a poor quality app to learn Finnish. Moreover, 10 people found it as a moderate level app.

22 people think that this is a good app because the graphics of the app is good. Nonetheless, on the basis of graphics, 12 individuals surmise this is an excellent application to learn Finnish. Then again, 5 individuals mirrored that low-quality graphics is used in

this app. On the other hand, 11 people reflected that moderate quality graphics are used in this app.

15 people feel that the app features and contents of the app are really excellent for those who want to learn Finnish. App features and contents are in poor level depicted by 3 people among 50. However, 23 people believe that app features and contents are at a good level. On the other hand, 9 people think that app features and contents are at a moderate level.

The majority of the participants (27 out of 50 people) think that functionality and control of the application are at a good level to learn Finnish. However, 4 people mirrored that functionality and control is not tvery good. On the other hand, 11 people found functionality and control excellent in this app. Again, 8 people feel that they are at a moderate level.

## 6 Conclusion

The main objective of this thesis was to build an application for people who want to learn Finnish. In order to achieve this target, the “Know Finnish” app was developed using API level 16 or above (highest 26) because lower API level uses more Android devices. This application was developed using Android Studio IDE because the majority of the plugin or required classes are built-in in the system. An assortment of mobile devices, for example, Samsung Galaxy Note III, HTC, OnePlus 5 were tested for the usefulness of the application. A completely tested and utilitarian Android application for the target people was produced in this thesis.

The user will gain many benefits from the Android application to learn Finnish. They will be able to save their words in the Own dictionary and assess themselves for that word later on. There is a word game included in the application. The user will get good Finnish knowledge when playing that game. Moreover, they will get their required words from the dictionary section. The vocabulary part will improve their knowledge about common Finnish words with images.

### 6.1 Limitations and Future work

The “**Know Finnish**” application will help to improve the user’s Finnish knowledge. However, this app is the first version, so that it has some limitations. It will be updated for a future version. Possible future plans are the following:

- The user interface and designs of the application will be improved in future.
- More words and sentences will be added to the dictionary.
- Different types of word/sentence puzzles will be added.
- Pronunciation of the word / sentences will be added in future.
- Sound effects will be added in future.



## References

1. Announcing the Android 1.0 SDK, release 1. [Online]  
URL: <https://Android-developers.googleblog.com/>. Retrieved: August 5, 2017.
2. What is Android? [Online]  
URL: <http://developer.Android.com/guide/basics/what-is-Android.html>.  
Retrieved: July 17, 2017
3. Google Play replaces Android Market, consolidates Google's media marketplaces.[Online]  
URL:<http://www.geekwire.com/google-playreplaces-Android-market-consolidates-googles-media-marketplaces>. Retrieved: September 12, 2017.
4. Jackson, W. (2017). Android Apps for Absolute Beginners. New York, NY.
5. Meier, R. (May 2012). Professional Android 4 Application Development, 4<sup>th</sup>- edition. Indianapolis, IN. Wiley Publishing, Inc.
6. Ableson, W.F., Sen, R. and King, C. (2016). Android in Action. Stamford, CT. Manning Publications Co.
7. Murphy, M. (2016). Beginning Android. New York, NY. Apress.
8. Android Architecture. [Online]  
URL: [http://elinux.org/Android\\_Architecture](http://elinux.org/Android_Architecture) Retrieved on September 18, 2017
9. Allen.G. Beginning Android 4. Apress, Lexington, KY, 2016.
10. Android Developers. Application Fundamentals. [Online]  
URL:<http://developer.Android.com/guide/components/fundamentals.html>.  
Retrieved: October 24, 2017.
11. An Android Developers Services. [Online]  
URL: <http://developer.Android.com/guide/components/services.html>  
Retrieved: May 21, 2017.
12. Android Introduction. [Online]  
URL: <http://ecee.colorado.edu/ecen3000/lecture/introduction.pdf>.

Retrieved: August 8, 2017.

13. Android App Market. Android Architecture - The Key Concepts of Android OS. [Online]

URL: <http://www.Android-app-market.com/Android-architecture.html>.

Retrieved: June 11, 2017.

14. Android Developers Activity. [Online]

URL: <http://developer.Android.com/reference/Android/app/Activity.html>.

Retrieved: September 28, 2017.

15. Android Developers. UI Overview. [Online]

URL: <http://developer.Android.com/guide/topics/ui/overview.html>.

Retrieved: August 19, 2017

16. Android Studio Installation. [Online]

URL: <https://developer.Android.com/studio/index.html> .

Retrieved on July 18, 2017.

17. Extending the WebView. [Online]

URL: [http://groups.google.com/group/Androiddevelopers/browse\\_thread/thread/f14e899b953f333f#](http://groups.google.com/group/Androiddevelopers/browse_thread/thread/f14e899b953f333f#) . Retrieved: October 18, 2017.

18. ListView Backgrounds: An Optimization. [Online]

URL: <http://tool.oschina.net/uploads/apidocs/Android/resources/articles/listview-backgrounds.html>. Retrieved: July 15, 2017.

19. Android Studio [Online]

URL: <https://developer.Android.com/> .

Retrieved on July 18, 2017.

20. Online form builder

URL: <https://www.jotform.com/>.

Retrieved on October 20, 2017.

21. Zechner, M. (2017). Beginning Android Games. New York, NY. Apress.

## Appendices: Source code of the Know Finnish Application (Appendix 1-6)

### Appendix 1 Own Dictionary Controller classes

```

package com.mamun.myproject;

import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.view.View;
import Android.widget.Button;
import Android.widget.EditText;

import Database.DatabaseHelper;
import Database.VideoDetails;

public class Add_Word extends AppCompatActivity {
    Button btn1;
    EditText word, meaning;
    VideoDetails add=null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.addword);
        add=new VideoDetails();
        btn1 = (Button) findViewById(R.id.button7);
        word = (EditText) findViewById(R.id.editText3);
        meaning = (EditText) findViewById(R.id.editText4);
        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                add.word=word.getText().toString();
                add.meaning=meaning.getText().toString();
                DatabaseHelper db;
                db = new DatabaseHelper(Add_Word.this);
                db.createVideo(add, DatabaseHelper.TABLE_VIDEOS);
                db.closeDB();
            }
        });
    }
}

```

Listing 12. Add\_Word class to go to next activities using Intent.

```

package com.mamun.myproject;

import Android.content.Context;
import Android.content.Intent;
import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.view.View;
import Android.widget.Button;
import Android.widget.EditText;
import Android.widget.TextView;
import Database.DatabaseHelper;
import Database.VideoDetails;

public class Dictionary_Page2 extends AppCompatActivity {
    Button word_list,add_word,search, button8;
    EditText edit_search;
    TextView edit_show;
    VideoDetails search_word;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        final Context context = this;
        setContentView(R.layout.dictionarypage1);
        search_word=new VideoDetails();
        word_list=(Button) findViewById(R.id.button5);
        add_word=(Button) findViewById(R.id.button6);
        search=(Button) findViewById(R.id.button13);
        edit_search=(EditText) findViewById(R.id.editText);
        edit_show=(TextView) findViewById(R.id.showword);

        word_list.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(Dictionary_Page2.this,
Word_List.class));
            }
        });
        add_word.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new In-
tent(Dictionary_Page2.this,Add_Word.class));
            }
        });

        search.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                DatabaseHelper db;
                db = new DatabaseHelper(Dictionary_Page2.this);

search_word=db.getVideofromTable(edit_search.getText().toString(),DatabaseHelp
er.TABLE_VIDEOS);
                db.closeDB();
                edit_show.setText(search_word.meaning);
            }
        });

        button8=(Button) findViewById(R.id.button8);
        button8.setOnClickListener(new View.OnClickListener() {

            @Override

```

```

        public void onClick(View arg0) {
            Intent intent = new Intent(context, webview.class);
            startActivity(intent);
        }
    });
}
}

```

Listing 13. Dictionary\_Page2 class to search word from dictionary.

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.os.Bundle;
import Android.webkit.WebView;

/**
 * Created by Mamun on 4/27/2016.
 */
public class webview extends Activity{
    private WebView webView;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.webview1);

        webView = (WebView) findViewById(R.id.webView1);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl("https://translate.google.com/");
    }

}

```

Listing 14. WebView class for Google Translator.

```

package com.mamun.myproject;

import Android.content.Context;
import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.view.LayoutInflater;
import Android.view.View;
import Android.view.ViewGroup;
import Android.widget.ArrayAdapter;
import Android.widget.ListView;
import Android.widget.TextView;

import Java.util.ArrayList;

import Database.DatabaseHelper;
import Database.VideoDetails;

public class Word_List extends AppCompatActivity {
    DatabaseHelper db;
    VideoDetails word_list = null;
}

```

```

ArrayList<VideoDetails> word;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.wordlists);
    word_list = new VideoDetails();
    db = new DatabaseHelper(Word_List.this);
    word = db.getAllVideoDetailsFromTable(DatabaseHelper.TABLE_VIDEOS);
    db.closeDB();
    ListView lv = (ListView) findViewById(R.id.listView);

    // Change MyActivity.this and myListOfItems to your own values
    CustomListAdapterDialog clad = new CustomListAdapterDialog(
Dialog(Word_List.this, word);

    lv.setAdapter(clad);
}

private class CustomListAdapterDialog extends ArrayAdapter<String> {

    private Context context;
    LayoutInflater inflater;

    public CustomListAdapterDialog(Context con, ArrayList values) {
        super(con, R.layout.word_list_style, values);
        this.context = con;
        inflater = (LayoutInflater) context
            .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent)
{
        ViewHolderDialog holder = null;

        if (convertView == null) {
            holder = new ViewHolderDialog();
            convertView = inflater.inflate(R.layout.word_list_style,
null);

            holder.english = (TextView) convertView
                .findViewById(R.id.english);
            holder.finnish = (TextView) convertView
                .findViewById(R.id.finnish);

            convertView.setTag(holder);
        } else {
            holder = (ViewHolderDialog) convertView.getTag();
        }
        holder.english.setText(word.get(position).word);
        holder.finnish.setText(word.get(position).meaning);

        return convertView;
    }
}

```

```

    }

    class ViewHolderDialog {

        TextView english, finnish;
    }
}

```

Listing 15. Word\_List class.

```

package Database;

import Android.content.ContentValues;
import Android.content.Context;
import Android.database.Cursor;
import Android.database.sqlite.SQLiteDatabase;
import Android.database.sqlite.SQLiteOpenHelper;
import Android.os.Environment;
import Android.util.Log;

import Java.io.File;
import Java.util.ArrayList;
import Java.util.List;

public class DatabaseHelper extends SQLiteOpenHelper {

    // Logcat tag
    private static final String LOG = "DatabaseHelper";

    // Database Version
    private static final int DATABASE_VERSION = 1;

    // Database Name
    private static final String DATABASE_NAME = "quize_app";

    // Table Names
    public static final String TABLE_VIDEOS = "quiz_app_table";

    private static final String KEY_VIDEO_ID = "english";
    private static final String KEY_meaning = "finnish";
    // Tag table create statement
    private static final String CREATE_TABLE_VIDEOS = "CREATE TABLE " + TA-
BLE_VIDEOS
        + "(" + KEY_VIDEO_ID + " TEXT," + KEY_meaning + " TEXT" + ")";

    public DatabaseHelper(Context context) {
        super(context, Environment.getExternalStorageDirectory()
            + File.separator + "Mamun"
            + File.separator + DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {

        // creating required tables

```

```

        db.execSQL(CREATE_TABLE_VIDEOS);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // on upgrade drop older tables
        db.execSQL("DROP TABLE IF EXISTS " + CREATE_TABLE_VIDEOS);
        // create new tables
        onCreate(db);
    }

    // ----- "todos" table methods -----//

    /*
     * Creating a todo
     */
    public long createVideo(VideoDetails video, String tableName) {
        SQLiteDatabase db = this.getWritableDatabase();

        ContentValues values = new ContentValues();
        values.put(KEY_VIDEO_ID, video.word);
        values.put(KEY_meaning, video.meaning);
        // insert row
        long todo_id = db.insert(tableName, null, values);

        return todo_id;
    }

    public void createVideos(List<VideoDetails> videos, String tableName) {
        for (VideoDetails videoDetail : videos) {
            createVideo(videoDetail, tableName);
        }
    }

    public ArrayList<VideoDetails> getAllVideoDetailsFromTable(String tableName) {
        SQLiteDatabase db = this.getReadableDatabase();
        /*Cursor ti = db.rawQuery("PRAGMA table_info(videos_table_mosarraaf)",
        null);
        if ( ti.moveToFirst() ) {
            do {
                Log.e("Column Name: ", "col: " + ti.getString(1));
            } while (ti.moveToNext());
        }*/
        ArrayList<VideoDetails> videos = new ArrayList<>();
        String selectQuery;
        selectQuery = "SELECT * FROM " + tableName;

        Log.e(LOG, selectQuery);

        //SQLiteDatabase db = this.getReadableDatabase();
        Cursor c = db.rawQuery(selectQuery, null);

        // looping through all rows and adding to list
        if (c.moveToFirst()) {
            do {
                VideoDetails videoDetail = new VideoDetails();
                videoDetail.meaning =
                c.getString((c.getColumnIndex(KEY_meaning)));
                videoDetail.word =
                c.getString((c.getColumnIndex(KEY_VIDEO_ID)));
                // adding to todo list
                videos.add(videoDetail);
            } while (c.moveToNext());
        }
    }

```



```

    }

    return videos;
}
/*
 * get single todo
 */
public VideoDetails getVideofromTable(String video_id, String tableName) {
    SQLiteDatabase db = this.getReadableDatabase();

    String selectQuery = "SELECT * FROM " + tableName + " WHERE "
        + KEY_VIDEO_ID + " = " + video_id;
    Cursor c = db.query(tableName, null, "english=?", new String[] { video_id }, null, null, null);
    Log.e(LOG, selectQuery);

    /* Cursor c = db.rawQuery(selectQuery, null);
 */
    if (c != null)
        c.moveToFirst();

    VideoDetails videoDetail = new VideoDetails();
    videoDetail.meaning = c.getString((c.getColumnIndex(KEY_meaning)));
    videoDetail.word = c.getString((c.getColumnIndex(KEY_VIDEO_ID)));
    return videoDetail;
}

public List<String> getAllmeaningsFromTable(String tableName) {
    List<String> meanings = new ArrayList<String>();
    String selectQuery = "SELECT " + KEY_meaning + " FROM " + tableName;

    Log.e(LOG, selectQuery);

    SQLiteDatabase db = this.getReadableDatabase();
    Cursor c = db.rawQuery(selectQuery, null);

    // looping through all rows and adding to list
    if (c.moveToFirst()) {
        do {
            // adding to todo list
            meanings.add(c.getString((c.getColumnIndex(KEY_meaning))));
        } while (c.moveToNext());
    }

    return meanings;
}

/*
 * getting todo count
 */
public int getCountForTable(String tableName) {
    String countQuery = "SELECT * FROM " + tableName;
    SQLiteDatabase db = this.getReadableDatabase();
    db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery(countQuery, null);

    int count = cursor.getCount();
    cursor.close();

    // return count
    return count;
}

```

```

/*
 * Updating a todo
 */
public int updateVideoToTable (VideoDetails
                               video, String tableName) {
    SQLiteDatabase db = this.getWritableDatabase();

    ContentValues values = new ContentValues();
    values.put(KEY_VIDEO_ID, video.word);
    values.put(KEY_meaning, video.meaning);
    // updating row
    return db.update(tableName, values, KEY_VIDEO_ID + " = ?",
                    new String[]{String.valueOf(video.word)});
}

/*
 * Deleting a todo
 */
public void deleteVideoFromTable (String word, String tableName) {
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(tableName, KEY_VIDEO_ID + " = ?",
             new String[]{String.valueOf(word)});
}

public void deleteAllVideos (String tableName) {
    SQLiteDatabase db = this.getWritableDatabase();
    db.delete(tableName, null, null);
}

public void closeDB() {
    SQLiteDatabase db = this.getReadableDatabase();
    if (db != null && db.isOpen())
        db.close();
}
}

```

Listing 16. DatabaseHelper class.

```

package Database;

public class VideoDetails {
    public String word;
    public String meaning;
}

```

Listing 17. VideoDetails class.

## Appendix 2 Verify You! Controller classes

```

package com.mamun.myproject;

import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.view.View;
import Android.widget.CheckBox;
import Android.widget.TextView;
import Android.widget.Toast;

import Java.util.ArrayList;
import Java.util.Collections;

import Database.DatabaseHelper;
import Database.VideoDetails;

public class Quiz_Easy1 extends AppCompatActivity {
    DatabaseHelper db;
    VideoDetails search_word1, search_word2, search_word3;
    TextView txt_question, score ;
    CheckBox checkBox1, checkBox2, checkBox3;

    public static int point=0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.easy1);
        point=0;
        txt_question=(TextView) findViewById(R.id.textView16);
        checkBox1=(CheckBox) findViewById(R.id.checkBox);
        checkBox2=(CheckBox) findViewById(R.id.checkBox2);
        checkBox3=(CheckBox) findViewById(R.id.checkBox3);
        score=(TextView) findViewById(R.id.textView4);
        get_uestion();
    }

    void get_uestion() {
        score.setText(point+"");
        db = new DatabaseHelper(Quiz_Easy1.this);
        search_word1=new VideoDetails();
        search_word2=new VideoDetails();
        search_word3=new VideoDetails();
        list2 = new ArrayList<>();
        for (int j=0; j<db.getCountForTable(DatabaseHelper.TABLE_VIDEOS); j++)
        {
            list2.add(new Integer(j));
        }
        Collections.shuffle(list2);

        search_word1=db.getAllVideoDetailsFromTable(DatabaseHelper.TABLE_VIDEOS).get(list2.get(0));

        search_word2=db.getAllVideoDetailsFromTable(DatabaseHelper.TABLE_VIDEOS).get(list2.get(1));

        search_word3=db.getAllVideoDetailsFromTable(DatabaseHelper.TABLE_VIDEOS).get(list2.get(2));

        db.closeDB();
        txt_question.setText(" What is the meaning of Finnish word \" +

```

```

search_word1.meaning + "\'-";
    random();
}

ArrayList<Integer> list2;

public void random() {

    list = new ArrayList<>();
    for (int j=0; j<3; j++) {
        list.add(new Integer(j));
    }
    Collections.shuffle(list);
    String[] data={search_word1.word,search_word2.word,search_word3.word};
    checkBox1.setText(data[list.get(0)]);
    checkBox2.setText(data[list.get(1)]);
    checkBox3.setText(data[list.get(2)]);
    checkBox1.setChecked(false);
    checkBox2.setChecked(false);
    checkBox3.setChecked(false);
    checkBox1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(checkBox1.getText().toString().equals(search_word1.word)){
                point++;
            }else {

Toast.makeText(getApplicationContext(),"Error",Toast.LENGTH_LONG).show();
            }
            get_uestion();
        }
    });
    checkBox2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(checkBox2.getText().toString().equals(search_word1.word)){
                point++;
            }else {

Toast.makeText(getApplicationContext(),"Error",Toast.LENGTH_LONG).show();
            }
            get_uestion();
        }
    });
    checkBox3.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if (checkBox3.getText().toString().equals(search_word1.word))
            {
                point++;
            } else {
                Toast.makeText(getApplicationContext(), "Error",
Toast.LENGTH_LONG).show();
            }
            get_uestion();
        }
    });
}
ArrayList<Integer> list;
}

```

Listing 18. Quiz\_Easy1 class for verify you! Screen

## Appendix 3 Vocabulary Controller classes

```

package com.mamun.myproject;

import Android.os.Bundle;
import Android.support.v7.app.AppCompatActivity;
import Android.widget.ListView;

/**
 * Created by Mamun on 9/19/2017.
 */

public class Listmaking extends AppCompatActivity {

    ListView list;
    String[] finnishwords = {"Man-
go", "Vesimeloni", "Banaani", "Viinirypäleitä", "Kiivi", "Avokado" };
    String[] desc = {"Tämä on mango", " Tämä on Vesimeloni", "Tämä on banaani", "
Tämä on viinirypäleitä", "Tämä on kiivi", "Tämä on Avokado" };
    Integer[] imageid =
{R.drawable.mango, R.drawable.watermelon, R.drawable.banana, R.drawable.grapes, R.
drawable.kiwi, R.drawable.avocado };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.wordlist);

        list= (ListView) findViewById(R.id.listview12);
        CustomListView adater = new Custom-
ListView(Listmaking.this, finnishwords, desc, imageid);

        list.setAdapter(adater);

    }

}

```

Listing 19. Listmaking class to create listview for Vocabulary section

## Appendix 4 Word Game Controller classes

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.content.Intent;
import Android.os.Bundle;
import Android.view.Menu;
import Android.view.MotionEvent;
import Android.view.View;
import Android.view.View.OnTouchListener;
import Android.view.Window;
import Android.view.WindowManager;
import Android.widget.Button;

import com.mamun.myproject.GameActivity;

public class WelcomeActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, Win-
dowManager.LayoutParams.FLAG_FULLSCREEN);

        setContentView(R.layout.activity_welcome);
        final Button startButton=(Button) findViewById(R.id.AndroidButton);

        startButton.setOnTouchListener(new OnTouchListener() {

            @Override
            public boolean onTouch(View v, MotionEvent event) {
                switch (event.getAction()) {
                    case MotionEvent.ACTION_DOWN:
                        startButton.getBackground().setAlpha(100);

                        break;
                    case MotionEvent.ACTION_UP:
                        Intent intent=new Intent(WelcomeActivity.this, GameAc-
tivity.class);

                        startActivity(intent);
                        startButton.getBackground().setAlpha(255);

                        break;
                    case MotionEvent.ACTION_MOVE:

                        break;

                    default:
                        break;
                }
                return false;
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {

```

```

        getMenuInflater().inflate(R.menu.welcome, menu);
        return true;
    }
}

```

Listing 20. WelcomeActivity class

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

import Android.graphics.Bitmap;
import Android.graphics.BitmapFactory;
import Android.graphics.Point;

public class Dock {

    Bitmap dockBitmap;
    int dockWidth,dockHeight;
    int leftmostpoint, rightmostPoint;

    Point topleftPoint=new Point(0, 0),bottomCenterPoint;
    DrawingThread drawingThread;

    boolean movingLeftFlag=false;
    boolean movingRightFlag=false;

    public Dock(DrawingThread drawingThread, int bitmapId) {
        this.drawingThread=drawingThread;
        Bitmap tempBit-
map=BitmapFactory.decodeResource(drawingThread.context.getResources(), bit-
mapId);
        tempBitmap=Bitmap.createScaledBitmap(tempBitmap, draw-
ingThread.displayX, draw-
ingThread.displayX*tempBitmap.getHeight()/tempBitmap.getWidth(), true);

        dockBitmap=tempBitmap;
        dockHeight=dockBitmap.getHeight();
        dockWidth=dockBitmap.getWidth();

        bottomCenterPoint=new Point((int)drawingThread.displayX/2,
(int)drawingThread.displayY);
        topleftPoint.y=bottomCenterPoint.y-dockHeight;

        updateInfo();

    }

    private void updateInfo() {
        leftmostpoint=bottomCenterPoint.x-dockWidth/2;
        rightmostPoint=bottomCenterPoint.x+dockWidth/2;

        topleftPoint.x=leftmostpoint;
    }
}

```

```

}

public void moveDockToLeft() {
    bottomCenterPoint.x-=4;
    updateInfo();
}

public void moveDockToRight() {
    bottomCenterPoint.x+=4;
    updateInfo();
}

public void startMovingLeft() {
    Thread thread=new Thread(){
        @Override
        public void run() {
            movingLeftFlag=true;

            while (movingLeftFlag) {
                moveDockToLeft();
                try {
                    sleep(10);
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
    };
    thread.start();
}

public void stopMovingLeft() {
    movingLeftFlag=false;
}

public void startMovingRight() {
    Thread thread=new Thread(){
        @Override
        public void run() {
            movingRightFlag=true;

            while (movingRightFlag) {
                moveDockToRight();
                try {
                    sleep(10);
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
    };
    thread.start();
}

public void stopMovingRight() {
    movingRightFlag=false;
}
}

```

Listing 21. Dock class



```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

import Android.content.Context;
import Android.graphics.Bitmap;
import Android.graphics.BitmapFactory;
import Android.graphics.Canvas;
import Android.graphics.Color;
import Android.graphics.Paint;
import Android.graphics.Paint.Align;
import Android.graphics.Point;
import Android.view.Display;
import Android.view.WindowManager;

import Java.util.ArrayList;

public class DrawingThread extends Thread {

    private Canvas canvas;
    GameView gameView;
    Context context;

    boolean threadFlag=false;
    boolean touchedFlag=false;
    boolean pauseFlag=false;

    Bitmap backgroundImage;
    int displayX,displayY;

    int maxScore=0;

    ArrayList<Robot> allRobots;
    ArrayList<Bitmap> allPossibleRobots;

    com.mamun.myproject.AnimationThread animationThread;
    ScoreCounterThread scoreCounterThread;

    Paint scorePaint;

    com.mamun.myproject.Dock dock;

    public DrawingThread( GameView gameView, Context context) {
        super();
        this.gameView = gameView;
        this.context = context;

        initializeAll();
    }

    private void initializeAll() {
        WindowManager windowManager=(WindowManager) context.getSystemService(Context.WINDOW_SERVICE);
        Display defaultDisplay=windowManager.getDefaultDisplay();

        Point displayDimension=new Point();
        defaultDisplay.getSize(displayDimension);

```

```

        displayX=displayDimension.x;
        displayY=displayDimension.y;

        backgroundImage=BitmapFactory.decodeResource(context.getResources(),
R.drawable.background);
        backgroundImage=Bitmap.createScaledBitmap(backgroundImage, displayX,
displayY, true);

        initializeAllPossibleRobots();
        scoreCounterThread=new ScoreCounterThread(this);

        dock=new com.mamun.myproject.Dock(this, R.drawable.dock);
        scorePaint=new Paint();
        scorePaint.setColor(Color.WHITE);
        scorePaint.setTextAlign(Align.CENTER);
        scorePaint.setTextSize(displayX/10);

    }

    private void initializeAllPossibleRobots() {
        allRobots=new ArrayList<Robot>();
        allPossibleRobots=new ArrayList<Bitmap>();

        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.robot1));
        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.robot2));
        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.robot3));
        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.robot4));
        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.robot5));
        allPossibleRobots.add(giveResizedRobotBitmap(R.drawable.fruits));

    }

    private Bitmap giveResizedRobotBitmap(int resourceID) {
        Bitmap tempBitmap=BitmapFactory.decodeResource(context.getResources(),
resourceID);
        tempBitmap=Bitmap.createScaledBitmap(tempBitmap, displayX/5, (tempBit-
map.getHeight()/tempBitmap.getWidth())*(displayX/5), true);

        return tempBitmap;
    }

    @Override
    public void run() {
        threadFlag=true;
        animationThread=new com.mamun.myproject.AnimationThread(this);
        animationThread.start();
        scoreCounterThread.start();

        while (threadFlag) {

            canvas=gameView.surfaceHolder.lockCanvas();

            try {
                synchronized (gameView.surfaceHolder) {

                    updateDisplay();

                }
            } catch (Exception e) {

```

```

        // TODO Auto-generated catch block
        e.printStackTrace();
    } finally{
        if (canvas!=null) {
            gameView.surfaceHolder.unlockCanvasAndPost(canvas);
        }
    }

    try {
        Thread.sleep(10);
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

}

scoreCounterThread.stopThread();
animationThread.stopThread();

}

private void updateDisplay() {
    canvas.drawBitmap(backgroundBitmap, 0, 0, null);
    drawDock();

    for (int i = 0; i < allRobots.size(); i++) {
        Robot tempRobot=allRobots.get(i);

        canvas.drawBitmap(tempRobot.robotBitmap, tempRobot.centerX-
(tempRobot.width/2), tempRobot.centerY-(tempRobot.height/2),
tempRobot.robotPaint);
    }

    if (pauseFlag) {
        pauseStateDraw();
    }

    drawScore();

    //drawSensorData();
}

private void pauseStateDraw() {
    Paint paint=new Paint();
    paint.setColor(Color.WHITE);
    paint.setTextSize(100);
    paint.setAlpha(170);
    paint.setTextAlign(Align.CENTER);
    canvas.drawARGB(170, 0, 0, 0);
    canvas.drawText("PAUSED",displayX/2, displayY/2, paint);
}

public void stopThread() {
    threadFlag=false;
}

```

```

        private void drawSensorData() {
            Paint paint=new Paint();
            paint.setColor(Color.GREEN);
            paint.setTextSize(displayX/10);

            canvas.drawText("X-Axis :"+ GameActivity.getgX(), 0, displayY/3,
paint);
            canvas.drawText("Y-Axis :"+GameActivity.getgY(), 0, displayY/3+(displayX/5), paint);
        }

        private void drawDock() {
            canvas.drawBitmap(dock.dockBitmap, dock.topleftPoint.x,
dock.topleftPoint.y, null);
        }

        private void drawScore() {
            if (maxScore>1000) {
                scorePaint.setColor(Color.GREEN);

                if (maxScore>10000) {

                    scorePaint.setColor(Color.CYAN);
                    if (maxScore>100000) {
                        scorePaint.setColor(Color.RED);
                    }
                }
            }

            canvas.drawText("Score : "+maxScore, displayX/2, displayY/7 , score-
Paint);
        }
    }
}

```

Listing 22. DrawingThread class

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

import Android.app.Activity;
import Android.app.AlertDialog;
import Android.content.DialogInterface;
import Android.content.pm.ActivityInfo;
import Android.hardware.Sensor;
import Android.hardware.SensorEvent;
import Android.hardware.SensorEventListener;
import Android.hardware.SensorManager;
import Android.os.Bundle;
import Android.view.Menu;
import Android.view.MotionEvent;
import Android.view.View;
import Android.view.View.OnTouchListener;
import Android.view.Window;
import Android.view.WindowManager;

```

```

import Android.view.WindowManager.LayoutParams;
import Android.widget.Button;
import Android.widget.Toast;

public class GameActivity extends Activity {

    GameView gameView;
    SensorManager sensorManager;
    SensorEventListener sensorEventListener;
    Sensor accelerometerSensor;
    private static float gX, gY;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, Win-
dowManager.LayoutParams.FLAG_FULLSCREEN);
        getWindow().addFlags(LayoutParams.FLAG_KEEP_SCREEN_ON);
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);

//    gameView=new GameView(this);

        initializeSensors();

        setContentView(R.layout.activity_game);

        gameView=(GameView) findViewById(R.id.myGameView);

        initializeButtons();

    }

    private void initializeButtons() {
        final Button moveLeftButton=(Button) findViewById(R.id.leftButton);

        moveLeftButton.setOnTouchListener(new OnTouchListener() {

            @Override
            public boolean onTouch(View v, MotionEvent event) {

                switch (event.getAction()) {
                    case MotionEvent.ACTION_DOWN:
                        gameView.drawingThread.dock.startMovingLeft();
                        moveLeftButton.getBackground().setAlpha(100);
                        break;

                    case MotionEvent.ACTION_UP:
                        gameView.drawingThread.dock.stopMovingLeft();
                        moveLeftButton.getBackground().setAlpha(255);
                        break;

                    default:
                        break;
                }

                return false;
            }
        });

        final Button moveRightButton=(Button) findViewById(R.id.rightButton);

```

```

moveRightButton.setOnTouchListener(new OnTouchListener() {

    @Override
    public boolean onTouch(View v, MotionEvent event) {

        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
                gameView.drawingThread.dock.startMovingRight();
                moveRightButton.getBackground().setAlpha(100);
                break;

            case MotionEvent.ACTION_UP:
                gameView.drawingThread.dock.stopMovingRight();
                moveRightButton.getBackground().setAlpha(255);
                break;

            default:
                break;
        }

        return false;
    }
});
}

public static float getgX() {
    return gX;
}

public static float getgY() {
    return gY;
}

private void initializeSensors() {
    sensorManager=(SensorManager) getSystemService(SENSOR_SERVICE);
    sensorEventListener=new SensorEventListener() {

        @Override
        public void onSensorChanged(SensorEvent event) {
            if (event.sensor.getType()==Sensor.TYPE_ACCELEROMETER) {
                gX=-event.values[0];
                gY=event.values[1];

                if (gY<0) {
                    stopUsingSensors();
                }
            }
        }
    };

    gameView.drawingThread.scoreCounterThread.stopThread();

    AlertDialog.Builder alertBuilder= new AlertDialog.Builder(GameActivity.this);
    alertBuilder.setTitle("Coution!");
    alertBuilder.setIcon(R.drawable.warning);
    alertBuilder.setMessage("Do not shake or hold your
phone upside down!!!");

    alertBuilder.setPositiveButton("RESTART", new DialogInterface.OnClickListener() {

```

```

        @Override
        public void onClick(DialogInterface dialog, int
which) {
            restartGame(null);
        }
    });

    alertBuilder.setNegativeButton("QUIT", new Di-
alogInterface.OnClickListener() {

        @Override
        public void onClick(DialogInterface dialog, int
which) {
            stopGame(null);

        }
    });

    alertBuilder.show();

    }
}

@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {
}
};
accelerometerSen-
sor=sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

startUsingSensors();
}

private void startUsingSensors() {
    sensorManager.registerListener(sensorEventListener, accelerometerSen-
sor, SensorManager.SENSOR_DELAY_NORMAL);
}

private void stopUsingSensors() {
    sensorManager.unregisterListener(sensorEventListener);
}

@Override
protected void onPause() {
    stopUsingSensors();
    super.onPause();
}

@Override
protected void onResume() {
    startUsingSensors();
    super.onResume();
}

@Override
protected void onStop() {

```

```

        stopUsingSensors();
        super.onStop();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.game, menu);
        return true;
    }

    public void pauseGame(View view) {
        if (gameView.drawingThread.pauseFlag==false) {
            gameView.drawingThread.animationThread.stopThread();
            gameView.drawingThread.pauseFlag=true;
            view.setBackgroundResource(R.drawable.unlock);
        }else {
            gameView.drawingThread.animationThread=new
com.mamun.myproject.AnimationThread(gameView.drawingThread);
            gameView.drawingThread.animationThread.start();
            view.setBackgroundResource(R.drawable.lock);
            gameView.drawingThread.pauseFlag=false;
        }
    }

    public void restartGame(View view) {
        stopUsingSensors();
        startUsingSensors();

        gameView.drawingThread.stopThread();
        gameView.drawingThread=new com.mamun.myproject.DrawingThread(gameView,
this);
        gameView.drawingThread.start();

        Toast.makeText(this, "Game Restarted", Toast.LENGTH_SHORT).show();
    }

    public void stopGame(View view) {
        this.finish();
    }
}

```

Listing 23. GameActivity class

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

public class AnimationThread extends Thread {

    private boolean flag=false;
    float gravityX,gravityY;
    float timeConstant=0.3f;
}

```



```

float retardationRatio=-0.7f;
int width,height;
int left,right,top,bottom;

DrawingThread drawingThread;

public AnimationThread(DrawingThread drawingThread) {
    super();
    this.drawingThread = drawingThread;
    updateDimensions();
}

private void updateDimensions() {
    width=drawingThread.allPossibleRobots.get(0).getWidth();
    height=drawingThread.allPossibleRobots.get(0).getHeight();

    left=width/2;
    top=height/2;
    right=drawingThread.displayX-(width/2);
    bottom=drawingThread.displayY-(height/2);
}

@Override
public void run() {
    flag=true;
    while (flag) {
        updateAllPositions();

        try {
            sleep(10);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }
}

private void updateAllPositions() {
    gravityX= GameActivity.getgX();
    gravityY=GameActivity.getgY();

    if (drawingThread.touchedFlag) {
        for (int i = 0; i < drawingThread.allRobots.size()-1; i++) {
            updateRobotsPosition(drawingThread.allRobots.get(i),i);
        }
    }else {
        for (int i = 0; i < drawingThread.allRobots.size(); i++) {
            updateRobotsPosition(drawingThread.allRobots.get(i),i);
        }
    }
}

private void updateRobotsPosition(Robot robot, int position) {
    ro-
    bot.centerX+=(robot.velocityX*timeConstant+0.5*gravityX*timeConstant*timeConst
ant);
    robot.velocityX+=gravityX*timeConstant;

    ro-
    bot.centerY+=(robot.velocityY*timeConstant+0.5*gravityY*timeConstant*timeConst

```

```

ant);
    robot.velocityY+=gravityY*timeConstant;

    constrainPosition(robot,position);

}

private void constrainPosition(Robot robot,int position) {
    //for x axis

    if (robot.centerX<left) {
        robot.centerX=left;
        robot.velocityX*=retardationRatio;

    }else if (robot.centerX>right) {
        robot.centerX=right;
        robot.velocityX*=retardationRatio;

    }

    //for y axis

    // if (robot.centerY<top) {
    //     robot.centerY=top;
    //     robot.velocityY*=retardationRatio;
    // }
    // }else

    if (robot.centerY>bottom) {
        if (isRobotOutsideDock(robot)) {
            robot.robotFelldown=true;
            if (robot.centerY>bottom+height) {
                drawingThread.allRobots.remove(position);
            }
        }

        if (robot.robotFelldown==false) {
            robot.centerY=bottom;
            robot.velocityY*=retardationRatio;
        }
    }

}

public void stopThread() {
    flag=false;
}

private boolean isRobotOutsideDock(Robot robot) {
    if ((robot.centerX+(width/2))<drawingThread.dock.leftmostpoint|| (robot.centerX-
(width/2)>drawingThread.dock.rightmostPoint)) {
        return true;
    }

    return false;
}
}

```

Listing 24. AnimationThread class

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

import Android.content.Context;
import Android.graphics.Point;
import Android.util.AttributeSet;
import Android.view.MotionEvent;
import Android.view.SurfaceHolder;
import Android.view.SurfaceHolder.Callback;
import Android.view.SurfaceView;
import Android.view.VelocityTracker;

import Java.util.Random;

public class GameView extends SurfaceView implements Callback{
    Context context;
    SurfaceHolder surfaceHolder;
    com.mamun.myproject.DrawingThread drawingThread;
    VelocityTracker velocityTracker;

    public GameView(Context context) {
        super(context);
        this.context=context;

        surfaceHolder=getHolder();
        surfaceHolder.addCallback(this);

        drawingThread=new DrawingThread( this, context);
        velocityTracker=VelocityTracker.obtain();
    }

    public GameView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
        this.context=context;

        surfaceHolder=getHolder();
        surfaceHolder.addCallback(this);

        drawingThread=new DrawingThread( this, context);
        velocityTracker=VelocityTracker.obtain();
    }

    public GameView(Context context, AttributeSet attrs) {
        super(context, attrs);
        this.context=context;

        surfaceHolder=getHolder();
        surfaceHolder.addCallback(this);

        drawingThread=new DrawingThread( this, context);
        velocityTracker=VelocityTracker.obtain();
    }
}

```

```

@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width,
                           int height) {
    // TODO Auto-generated method stub
}

@Override
public void surfaceCreated(SurfaceHolder holder) {
    try {
        drawingThread.start();
    } catch (Exception e) {
        // TODO Auto-generated catch block
        restartTread();
    }
}

private void restartTread() {
    drawingThread.stopThread();
    drawingThread=null;
    drawingThread=new DrawingThread(this, context);
    drawingThread.start();

    // TODO Auto-generated method stub
}

@Override
public void surfaceDestroyed(SurfaceHolder holder) {
    drawingThread.stopThread();
}

@Override
public boolean onTouchEvent(MotionEvent event) {
    if (drawingThread.pauseFlag==true) {
        return true;
    }

    Point touchPoint=new Point();
    touchPoint=new Point((int)event.getX(), (int)event.getY());
    Random random=new Random();

    switch (event.getAction()) {
        case MotionEvent.ACTION_DOWN:
            drawingThread.touchedFlag=true;
            drawingThread.allRobots.add(new Ro-
bot(drawingThread.allPossibleRobots.get(random.nextInt(5)), touchPoint));

            break;
        case MotionEvent.ACTION_UP:
            velocityTracker.computeCurrentVelocity(40);
            drawingThread.allRobots.get(drawingThread.allRobots.size()-
1).setVelocity(velocityTracker);
            drawingThread.touchedFlag=false;
            break;
        case MotionEvent.ACTION_MOVE:
            velocityTracker.addMovement(event);
            drawingThread.allRobots.get(drawingThread.allRobots.size()-
1).setCenter(touchPoint);

            break;
    }
}

```

```

        default:
            break;
    }

    return true;
}
}

```

Listing 25. GameView class

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/14/2017.
 */

import Android.graphics.Bitmap;
import Android.graphics.Paint;
import Android.graphics.Point;
import Android.view.VelocityTracker;

public class Robot {
    float centerX, centerY;
    int height, width;
    float velocityX, velocityY;

    Bitmap robotBitmap;
    Paint robotPaint;

    boolean robotFellDown=false;

    public Robot(Bitmap bitmap) {
        robotBitmap=bitmap;
        centerX=centerY=0;
        height=robotBitmap.getHeight();
        width=robotBitmap.getWidth();
        robotPaint=new Paint();
        velocityX=velocityY=0;
    }

    public Robot(Bitmap bitmap, int cX, int cY) {
        this(bitmap);
        centerX=cX;
        centerY=cY;
    }

    public Robot(Bitmap bitmap, Point center) {
        this(bitmap, center.x, center.y);
    }

    public void setCenter(Point centerPoint) {
        centerX=centerPoint.x;
        centerY=centerPoint.y;
    }
}

```

```

    public void setVelocity(VelocityTracker velocityTracker) {
        velocityX=velocityTracker.getXVelocity();
        velocityY=velocityTracker.getYVelocity();
    }
}

```

Listing 26. Robot class

```

package com.mamun.myproject;

import com.mamun.myproject.DrawingThread;
import com.mamun.myproject.Robot;

/**
 * Created by Mamun on 9/14/2017.
 */

public class ScoreCounterThread extends Thread {

    DrawingThread drawingThread;
    boolean threadRunningFlag=false;

    public ScoreCounterThread(DrawingThread drawingThread) {
        this.drawingThread=drawingThread;
    }

    @Override
    public void run() {
        threadRunningFlag=true;
        while (threadRunningFlag) {

            float tempMax=0;

            for (Robot robot: drawingThread.allRobots) {
                if (robot.centerY<tempMax) {
                    tempMax=robot.centerY;
                }
            }

            drawingThread.maxScore= (int) (drawingThread.maxScore>(-tempMax)?
drawingThread.maxScore: (-tempMax));

            try {
                sleep(100);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }

    public void stopThread() {

```

```
        threadRunningFlag=false;  
    }  
}
```

Listing 27. ScoreCounterThread class

## Appendix 5 TFQ Controller classes

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.app.AlertDialog;
import Android.content.DialogInterface;
import Android.os.Bundle;
import Android.view.View;
import Android.widget.Button;
import Android.widget.ProgressBar;
import Android.widget.TextView;
import Android.widget.Toast;

/**
 * Created by Mamun on 9/20/2017.
 */

public class SentencePuzzle extends Activity{
    // TODO: Declare member variables here:
    Button mTrueButton, mFalseButton;
    TextView mQuestionTextView;
    int mIndex;
    int mQuestion;
    TextView mScoreTextView;
    ProgressBar mProgressBar;
    int mScore;

    private TrueFalse[] mQuestionBank = new TrueFalse[] {
        new TrueFalse(R.string.question_1, false),
        new TrueFalse(R.string.question_2, true),
        new TrueFalse(R.string.question_3, true),
        new TrueFalse(R.string.question_4, true),
        new TrueFalse(R.string.question_5, true),
        new TrueFalse(R.string.question_6, false),
        new TrueFalse(R.string.question_7, true),
        new TrueFalse(R.string.question_8, false),
        new TrueFalse(R.string.question_9, true),
        new TrueFalse(R.string.question_10, true),
        new TrueFalse(R.string.question_11, false),
        new TrueFalse(R.string.question_12, false),
        new TrueFalse(R.string.question_13, true)
    };

    final int PROGRESS_BAR_INCREMENT =(int ) Math.ceil(100.0 / mQuestion-
Bank.length) ;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.sentencepuzzle);

        if(savedInstanceState != null){
            mScore = savedInstanceState.getInt("ScoreKey");
            mIndex =savedInstanceState.getInt("IndexKey");
        } else {
            mScore = 1;
            mIndex = 1;

```



```

    }

    mTrueButton = (Button) findViewById(R.id.true_button);
    mFalseButton = (Button) findViewById(R.id.false_button);
    mQuestionTextView = (TextView) findViewById(R.id.question_text_view);
    mScoreTextView = (TextView) findViewById(R.id.score);
    mProgressBar = (ProgressBar) findViewById(R.id.progress_bar);

    mQuestion = mQuestionBank[mIndex].getmQuestionID();
    mQuestionTextView.setText(mQuestion);
    mScoreTextView.setText ("Score " + mScore + "/"
+mQuestionBank.length);

    mTrueButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            checkAnswer(true);
            updateQuestion();
        }
    });

    mFalseButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            checkAnswer(false);
            updateQuestion();
        }
    });
}
// TODO: Methods

private void updateQuestion() {
    mIndex = (mIndex + 1) % mQuestionBank.length;

    if (mIndex == 0) {
        AlertDialog.Builder alert = new AlertDialog.Builder(this);
        alert.setCancelable(false);
        alert.setMessage(" You scored " + mScore + " points! ");
        alert.setPositiveButton("Close ", new DialogInterface-
face.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                finish();
            }
        });
        alert.show();
    }

    mQuestion = mQuestionBank[mIndex].getmQuestionID();
    mQuestionTextView.setText(mQuestion);
    mProgressBar.incrementProgressBy(PROGRESS_BAR_INCREMENT);
    mScoreTextView.setText ("Score " + mScore + "/" +mQuestionBank.length);
}

private void checkAnswer(boolean the userSelection) {

    boolean correctAnswer = mQuestionBank[mIndex].ismAnswer();

    if (the userSelection == correctAnswer) {
        Toast.makeText(getApplicationContext(),R.string.correct_toast,

```

```

Toast.LENGTH_SHORT).show();
        mScore = mScore + 1;
    } else{
        Toast.makeText(getApplicationContext(),R.string.incorrect_toast,
Toast.LENGTH_SHORT).show();
    }
}

    public void onSaveInstanceState(Bundle outState){
        super.onSaveInstanceState(outState);
        outState.putInt ("ScoreKey", mScore);
        outState.putInt ("IndexKey", mIndex);
    }
}

```

Listing 28. SentencePuzzle class to update True false Question

```

package com.mamun.myproject;

/**
 * Created by Mamun on 9/20/2017.
 */

public class TrueFalse { public int getmQuestionID() {
    return mQuestionID;
}

    public void setmQuestionID(int mQuestionID) {
        this.mQuestionID = mQuestionID;
    }

    public boolean ismAnswer() {
        return mAnswer;
    }

    public void setmAnswer(boolean mAnswer) {
        this.mAnswer = mAnswer;
    }

    private int mQuestionID;
    private boolean mAnswer;

    public TrueFalse(int questionResourceID, boolean trueOrFalse){
        mQuestionID = questionResourceID;
        mAnswer = trueOrFalse;
    }
}

```

Listing 29. TrueFalse class

## Appendix 6 Word Bank Controller classes

```

package com.mamun.myproject;

/**
 * Created by Mamun on 10/20/2017.
 */

import Android.app.Activity;
import Android.content.Intent;
import Android.content.SharedPreferences;
import Android.os.Bundle;
import Android.view.Menu;
import Android.view.View;
import Android.view.View.OnClickListener;
import Android.widget.Button;
import Android.widget.EditText;

public class WordBankActivity extends Activity {

    final static String SHARED_NAME_STRING="sharedp";
    final static String THE_USER_NAME_STRING="the user";

    Button button;
    EditText editText;

    SharedPreferences sharedPreferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.wordbank_main);
        editText=(EditText) findViewById(R.id.the_usernameEditText);
        button=(Button) findViewById(R.id.enterButton);

        sharedPreferences=getSharedPreferences(SHARED_NAME_STRING,
        MODE_PRIVATE);
        String the_usernameString=sharedPreferences.getString(THE_US-
        ER_NAME_STRING, "");

        editText.setText(the_usernameString);

        button.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                String string=editText.getText().toString();
                Intent intent=new Intent(WordBankActivity.this, Diction-
                aryListActivity.class);
                intent.putExtra("the user", string);

                SharedPreferences.Editor editor=sharedPreferences.edit();
                editor.putString(THE_USER_NAME_STRING, string);
                editor.commit();

                startActivity(intent);
            }
        });
    }
}

```

```

        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}

```

Listing 30. WordBankActivity class

```

package com.mamun.myproject;

import java.util.ArrayList;

public class WordDefinition {
    String word, definition;

    public WordDefinition(String word, ArrayList<String> alldefinition) {
        this.word=word;

        StringBuilder stringBuilder=new StringBuilder();
        for (String string : alldefinition) {
            stringBuilder.append(string);
        }
        this.definition=stringBuilder.toString();
    }

    public WordDefinition(String word, String alldefinition) {
        this.word=word;
        this.definition=alldefinition;
    }
}

```

Listing 31. WordDefinition class

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.os.Bundle;
import Android.util.Log;
import Android.view.Menu;
import Android.widget.TextView;

public class WordDefinitionDetailActivity extends Activity {

    TextView wordTextView;
    TextView definitionTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_word_definition_detail);
        wordTextView=(TextView) findViewById(R.id.wordTextView);
        definitionTextView=(TextView) findViewById(R.id.definitionTextView);

        Log.d("DICTIONARY", "third activity started");

        wordTextView.setText(getIntent().getStringExtra("word"));
        definitionTextView.setText(getIntent().getStringExtra("definition"));
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.word_definition_detail, menu);
        return true;
    }
}

```

Listing 32. WordDefinitionDetailActivity class

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.view.LayoutInflater;
import Android.view.View;
import Android.view.ViewGroup;
import Android.widget.AdapterView;
import Android.widget.AdapterView.OnItemClickListener;
import Android.widget.AdapterView.OnItemSelectedListener;
import Android.widget.ImageView;
import Android.widget.TextView;

/**
 * Created by Mamun on 9/19/2017.
 */

public class CustomListView extends ArrayAdapter<String>{

    private final String[] finnishwords;
    private final String[] desc;
    private final Integer[] imageid;
    private final Activity context;
}

```

```

    public CustomListView(Activity context, String[] finnishwords,String[]
desc, Integer[] imageid) {

        super(context, R.layout.listviewshow, finnishwords);

        this.context = context;
        this.finnishwords = finnishwords;
        this.desc = desc;
        this.imageid = imageid;

    }

    @Override
    public View getView(int position, View view ,ViewGroup parent) {

        LayoutInflater inflater = context.getLayoutInflater();
        View rowView = inflater.inflate(R.layout.listviewshow, null, true);
        TextView tv1 = (TextView) rowView.findViewById(R.id.tvword);
        TextView tv2 = (TextView) rowView.findViewById(R.id.tvdescription);

        ImageView imageView = (ImageView)
rowView.findViewById(R.id.imageViewa);
        tv1.setText(finnishwords[position]);
        tv2.setText(desc[position]);

        imageView.setImageResource(imageid[position]);
        return rowView;
    }
}

```

Listing 33. CustomListView class

```

package com.mamun.myproject;

import Android.content.ContentValues;
import Android.content.Context;
import Android.database.Cursor;
import Android.database.sqlite.SQLiteDatabase;
import Android.database.sqlite.SQLiteDatabase.CursorFactory;
import Android.database.sqlite.SQLiteOpenHelper;

import Java.util.ArrayList;

public class DictionaryDatabaseHelper extends SQLiteOpenHelper {

    final static String DICTIONARY_DATABASE="dictionary";
    final static String ITEM_ID_COLUMN="id";
    final static String WORD_COLUMN="word";
    final static String DEFINITION_COLUMN="definition";

    final static String CREATE_DATABASE_QUERY="CREATE TABLE
"+DICTIONARY_DATABASE+" ( "+
        ITEM_ID_COLUMN+" INTEGER PRIMARY KEY AUTOINCREMENT, "+

```

```

WORD_COLUMN+" TEXT , "+
DEFINITION_COLUMN+" TEXT)";

final static String ON_UPGRADE_QUERY="DROP TABLE "+DICTIONARY_DATABASE;

Context context;

public DictionaryDatabaseHelper(Context context, String name,
                                CursorFactory factory, int version) {
    super(context, DICTIONARY_DATABASE, factory, version);
    this.context=context;
}

@Override
public void onCreate(SQLiteDatabase database) {
    database.execSQL(CREATE_DATABASE_QUERY);
}

@Override
public void onUpgrade(SQLiteDatabase database, int oldVersion, int newVer-
sion) {
    database.execSQL(ON_UPGRADE_QUERY);
    onCreate(database);
}

public long insertData(WordDefinition wordDefinition) {
    SQLiteDatabase database=this.getWritableDatabase();
    ContentValues values=new ContentValues();

    values.put(WORD_COLUMN, wordDefinition.word);
    values.put(DEFINITION_COLUMN, wordDefinition.definition);

    return database.insert(DICTIONARY_DATABASE, null, values);
}

public long updateData(WordDefinition wordDefinition) {
    SQLiteDatabase database=this.getWritableDatabase();
    ContentValues values=new ContentValues();

    values.put(WORD_COLUMN, wordDefinition.word);
    values.put(DEFINITION_COLUMN, wordDefinition.definition);

    return database.update(DICTIONARY_DATABASE, values, WORD_COLUMN+"=?",
new String[]{wordDefinition.word});
}

public void deleteData(WordDefinition wordDefinition) {
    SQLiteDatabase database=this.getWritableDatabase();
    String queryString="DELETE FROM "+DICTIONARY_DATABASE+" WHERE
"+WORD_COLUMN+" = '"+wordDefinition.word+"'";

    database.execSQL(queryString);
}

public ArrayList<WordDefinition> getAllWords() {
    ArrayList<WordDefinition> arrayList=new ArrayList<WordDefinition>();
    SQLiteDatabase database=this.getReadableDatabase();

```

```

String selectAllQueryString="SELECT * FROM "+DICTIONARY_DATABASE;
Cursor cursor=database.rawQuery(selectAllQueryString, null);

    if (cursor.moveToFirst()) {
        do {
            WordDefinition wordDefinition=new WordDefini-
tion(cursor.getString(cursor.getColumnIndex(WORD_COLUMN)), cur-
sor.getString(cursor.getColumnIndex(DEFINITION_COLUMN)));
            arrayList.add(wordDefinition);
        } while (cursor.moveToNext());
    }
    return arrayList;
}

public WordDefinition getWordDefinition(String word) {
    SQLiteDatabase database=this.getReadableDatabase();
    WordDefinition wordDefinition=null;

    String selectQueryString="SELECT * FROM "+DICTIONARY_DATABASE+ " WHERE
"+WORD_COLUMN+" = '"+word+" '";
    Cursor cursor=database.rawQuery(selectQueryString, null);

    if (cursor.moveToFirst()) {
        wordDefinition=new WordDefini-
tion(cursor.getString(cursor.getColumnIndex(WORD_COLUMN)), cur-
sor.getString(cursor.getColumnIndex(DEFINITION_COLUMN)));
    }

    return wordDefinition;
}

public WordDefinition getWordDefinition(long id) {
    SQLiteDatabase database=this.getReadableDatabase();
    WordDefinition wordDefinition=null;

    String selectQueryString="SELECT * FROM "+DICTIONARY_DATABASE+ " WHERE
"+ITEM_ID_COLUMN+" = '"+id+" '";
    Cursor cursor=database.rawQuery(selectQueryString, null);

    if (cursor.moveToFirst()) {
        wordDefinition=new WordDefini-
tion(cursor.getString(cursor.getColumnIndex(WORD_COLUMN)), cur-
sor.getString(cursor.getColumnIndex(DEFINITION_COLUMN)));
    }

    return wordDefinition;
}

public void initializeDatabaseFortheFirstTime (ArrayList<WordDefinition>
wordDefinitions) {
    SQLiteDatabase database=this.getWritableDatabase();
    database.execSQL("BEGIN");

    ContentValues contentValues=new ContentValues();

    for (WordDefinition wordDefinition : wordDefinitions) {
        contentValues.put(WORD_COLUMN, wordDefinition.word);
        contentValues.put(DEFINITION_COLUMN, wordDefinition.definition);
        database.insert(DICTIONARY_DATABASE, null, contentValues);
    }
}

```



```

        database.execSQL("COMMIT");
    }
}

```

Listing 34. DictionaryDatabasehelper class

```

package com.mamun.myproject;

import Android.app.Activity;
import Android.content.Intent;
import Android.content.SharedPreferences;
import Android.os.Bundle;
import Android.util.Log;
import Android.view.View;
import Android.view.View.OnClickListener;
import Android.view.ViewGroup;
import Android.widget.AdapterView;
import Android.widget.AdapterView.OnItemClickListener;
import Android.widget.BaseAdapter;
import Android.widget.Button;
import Android.widget.EditText;
import Android.widget.ListView;
import Android.widget.TextView;
import Android.widget.Toast;

import Java.io.BufferedReader;
import Java.io.InputStream;
import Java.io.InputStreamReader;
import Java.util.ArrayList;

public class DictionaryListActivity extends Activity {

    TextView the userTextView;
    EditText searchEditText;
    Button searchButton;
    ListView dictionaryListView;

    String logTagString="DICTIONARY";
    ArrayList<WordDefinition> allWordDefinitions=new Ar-
rayList<WordDefinition>();

    DictionaryDatabaseHelper myDictionaryDatabaseHelper;
    SharedPreferences sharedPreferences;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_dictionary_list);

        Log.d("DICTIONARY", "second activity started");

        the userTextView=(TextView) findViewById(R.id.personTextView);
        the userTextView.setText(getIntent().getStringExtra(WordBankActivity.THE
USER_NAME_STRING)+"'s Dictionary");

        searchEditText=(EditText) findViewById(R.id.searchEditText);

```

```

        searchButton=(Button) findViewById(R.id.searchButton);
        dictionaryListView=(ListView) findViewById(R.id.dictionaryListView);

        myDictionaryDatabaseHelper=new DictionaryDatabaseHelper(this, "Dictionary-
ary", null, 1);
        sharedPrefer-
ences=getSharedPreferences(WordBankActivity.SHARED_NAME_STRING, MODE_PRIVATE);

        boolean initialized=sharedPreferences.getBoolean("initialized", false);

        if (initialized==false) {
            //Log.d(logTagString, "initializing for the first time");
            initializeDatabase();
            SharedPreferences.Editor editor=sharedPreferences.edit();
            editor.putBoolean("initialized", true);
            editor.commit();

        }else {
            Log.d(logTagString, "db already initialized");
        }

        allWordDefinitions=myDictionaryDatabaseHelper.getAllWords();

        dictionaryListView.setAdapter(new BaseAdapter() {

            @Override
            public View getView(int position, View view, ViewGroup arg2) {
                if (view==null) {
                    view=getLayoutInflater().inflate(R.layout.list_item, null);
                }
                TextView textView=(TextView)
view.findViewById(R.id.ListItemTextView);
                textView.setText(allWordDefinitions.get(position).word);

                return view;
            }

            @Override
            public long getItemId(int arg0) {
                // TODO Auto-generated method stub
                return 0;
            }

            @Override
            public Object getItem(int arg0) {
                // TODO Auto-generated method stub
                return null;
            }

            @Override
            public int getCount() {
                // TODO Auto-generated method stub
                return allWordDefinitions.size();
            }
        });

        dictionaryListView.setOnItemClickListener(new OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> arg0, View view, int position,
                long arg3) {
                Intent intent =new Intent(DictionaryListActivity.this, WordDefini-
tionDetailActivity.class);

```

```

        intent.putExtra("word", allWordDefinitions.get(position).word);
        intent.putExtra("definition", allWordDefinitions.get(position).definition);

        startActivity(intent);
    }
});

searchButton.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {
        String string=searchEditText.getText().toString();

        WordDefinition wordDefinition=myDictionaryDatabaseHelper.getWordDefinition(string);

        if (wordDefinition==null) {
            Toast.makeText(DictionaryListActivity.this, "Word not found",
            Toast.LENGTH_LONG).show();
        }else {
            Intent intent =new Intent(DictionaryListActivity.this, WordDefinitionDetailActivity.class);
            intent.putExtra("word", wordDefinition.word);
            intent.putExtra("definition", wordDefinition.definition);

            startActivity(intent);
        }
    }
});

}

private void initializeDatabase() {
    InputStream inputStream=getResources().openRawResource(R.raw.dictionary);
    BufferedReader bufferedReader=new BufferedReader(new InputStreamReader(inputStream));
    DictionaryLoader.loadData(bufferedReader, myDictionaryDatabaseHelper);
}

}
}

```

Listing 35. DictionaryListActivity class

```
package com.mamun.myproject;
```

```

import Java.io.BufferedReader;
import Java.io.IOException;
import Java.util.ArrayList;

public class DictionaryLoader {

    public static void loadData(BufferedReader bufferedReader, DictionaryData-
baseHelper dictionaryDatabaseHelper) {

        ArrayList<WordDefinition> allWords=new ArrayList<WordDefinition>();

        try {

            BufferedReader fileReader=bufferedReader;

            try {

                int c=17;
                c=fileReader.read();
                while (c!=(-1)) {

                    StringBuilder stringBuilder=new StringBuilder();

                    while ((char)c!='\n'&& c!=-1) {
                        try {
                            stringBuilder.append((char)c);
                        } catch (Exception e) {
                            // TODO Auto-generated catch block
                            System.out.println(stringBuilder.length());
                            //e.printStackTrace();
                        }
                        c= fileReader.read();
                        if (c==-1) {
                            return;
                        }
                    }

                    String wordString=stringBuilder.toString();

                    ArrayList<String> definition=new ArrayList<String>();
                    while (c=='\n' || c=='\t') {
                        c= fileReader.read();
                        if (c=='\n' || c=='\t' || c=='\r') {
                            StringBuilder stringBuilder2=new StringBuilder();
                            while (c!='\n') {
                                stringBuilder2.append((char)c);
                                c=fileReader.read();
                            }
                            String definitionString=stringBuilder2.toString();
                            definition.add(definitionString);
                        } else {
                            break;
                        }
                    }

                    wordString=wordString.trim();
                    //Logger.log("word Loaded: "+(++counter)+" :"+wordString);
                    allWords.add(new WordDefinition(wordString, definition));
                }
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}

```

```
    }  
  
    try {  
        dictionaryDatabaseHelper.initializeDatabaseFortheFirstTime(allWords);  
        fileReader.close();  
    } catch (IOException e) {  
        // TODO Auto-generated catch block  
        e.printStackTrace();  
    }  
    } catch (Exception e) {  
        // TODO Auto-generated catch block  
        e.printStackTrace();  
    }  
    }  
}
```

Listing 36. DictionaryLoader class

## Appendix 7 Online Evaluation Form

# Online evaluation form

Please rate the Know Finnish app based on its criteria....

**Know  
Finnish**

An Android App to Learn Finnish Language Easily.

**E-mail**

**Evaluation criteria \***

	Poor	Moderate	Good	Excellent
Graphics of the app	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
App features and contents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Functionality and control	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Instructor responded questions	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Overall criteria of the app	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

**Your comment about the app**

Figure 31. Evaluation form of the Know Finnish the app,

Source: <https://form.jotformeu.com/73125684817362>