A Report for

**SUMMER INTERNSHIP ON CORE JAVA WITH ANDROID**

*Submitted in the fulfillment of the requirements of the award of the degree of* ***Bachelor of Technology*** *in*

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING By

B.MOUNIKA ROSHAN- 1210314011

##### Under the esteemed guidance of Mr. NITISH, **Undergone at QUESTEASES SOLUTIONS,HYDERABAD**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM University, Gandhi Nagar,

Rushikonda, Visakhapatnam

Andhra Pradesh, India-530045

**CERTIFICATE**

This is to certify that the summer intern report entitled **CORE JAVA WITH ANDROID** is the bonafide report work done by **B.MOUNIKA ROSHAN(1210314011),** during the academic year 2016-17 for the partialfulfillment of the requirements for the Bachelor of Technology in Computer Science and Engineering from GITAM University, Rushikonda, Vishakhapatnam, AP. Has successfully completed summer intern program in GITAM University, Visakhapatnam

|  |
| --- |
| Head of Department |
| **Mr.shoukath ali khan** |

**ACKNOWLEDGEMENT**

I express my deep sense of gratitude to Mr.Nithish, for providing me his valuable guidance and sharp vision to undertake this one month industrial training. Training in an organization like ͞HP͟ which is fuelled by the so much zest & energy, ͞teaming͟ up to form a formidable force, was in itself a true learning experience which is going to help me immensely in my career. There is no substitute to ͞Teamwork͟. This is one of the many lessons I learnt during my training in ͞HP͟.I hereby convey my thanks to all those whom have rendered their valuable help, support and guidance during our training period.

I would also like to express our sincere gratitude to our Head of Department for having granted permission for the industrial experience. I would like to thank Mr.shoukath ali khan for granting me the permission to do project as a Trainee in this esteemed organization and for providing us all the

facilities.



**COMPANY PROFILE**

The Hewlett-Packard Company (commonly referred to as HP) was an

American multinational information technology company headquartered in Palo Alto, California. It developed and provided a wide variety of hardware components as well as software and related services to consumers, small- and medium-sized businesses (SMBs) and large enterprises, including customers in the government, health and education sectors.

HP Enterprise Services is the global business and technology services subsidiary of the HP Enterprise, (now Hewlett Packard Enterprise), strategic business unit. It was formed by the combination of HP's legacy services consulting and outsourcing business and the integration of acquired Electronic Data Systems, which had defined the outsourcing business when it was established in 1962 by H.

Ross Perot.

**Abstract**

In real life, forgetting the things is very common human tendency..sometimes we may kept the things in our premises and we may forget about it.Same thing can be happened with our mobiles.We may forget/lose our phone in our near premises.We may think that,we can just make call to that phone and find the location.but if the phone is in silent mode what we can do? how can we find the mobile location?

# In that situation the “RINGMODE” is very helpful. In this project,just by sending a single message to our phone we are changing our phone mode to ringing or silent or vibrate whatever we want.

# In the existing system there is no service like changing our phone upon receiving security code as a message previously. If any user unexpectedly forget his phone in his near premises and its in silent mode. As of now in android mobiles there is no such system which changes our phone mode to whtever we want i.e.silent or ring or vibrate.if user wants to change his mode,only manually he can change the mode of the phonewhich is not possible when phone is missing.

Summer Intern Report

**RingMode ON ANDROID**



By

**B.MOUNIKA ROSHAN – 121031401**

**Contents**

**1. Introduction**

**1.1. PROBLEM DEFINITION**

**1.2 EXISTING SYSTEM**

**1.3 PROPOSED SYSTEM**

**1.4 ADVANTAGES**

**1.5 MODULES**

**1.5.1 Setting the customized Code**

**1.5.2 Changing the phone mode**

**1.6 SCOPE OF PROJECT**

**2. LITERATURE SURVEY**

**2.1. JAVA**

**2.2. ANDROID**

**2.2.1 About Native code**

**2.2.2 Creating an android project**

**2.2.3 To create an AVD with the AVD manager**

**2.2.4 To Create or Modify a Launch Configuration**

**2.3. SQLITE**

**2.3.1 Design**

**2.3.2 Features**

**3. SYSTEM ANALYSIS**

**3.1 SOFTWARE REQUIREMENTS**

**3.2 HARDWARE REQUIREMENTS**

### **3.3 FEASIBILITY STUDY**

**3.3.1 Economic Feasibility**

**3.3.2 Operational Feasibility**

**3.3.3** **Technical Feasibility**

**4. SYSTEM DESIGN**

**4.1 ARCHITECTURE DIAGRAM**

**4.2 DATA FLOW DIAGRAMS**

**4.3 UNIFIED MODELING LANGUAGE (UML)**

**5. SYSTEM CODING & IMPLEMENTATION**

**5.1 Sample Code**

**5.2 ringmode tasklist**

**6. TESTING AND VALIDATION**

**6.1 Unit Testing**

**6.2 Integration Testing**

**6.3 Validation Testing**

**6.4 System Testing**

**6.5 Test cases**

**7. CONCLUSIONS**

**8. REFERENCES**

**RINGMODE**

**1.Introduction**

* 1. **PROBLEM DEFINITION:**

One of the **fastest growing industries now a day** is mobile industry. There are many competitors in this area who are doing research and development on new platforms & user experience. One such technology is Android from Google which is supported for many manufactured phones. These phones are described as next Generation mobiles [As described by Google]. In real life, forgetting the things is very common human tendency..sometimes we may kept the things in our premises and we may forget about it.Same thing can be happened with our mobiles.We may forget/lose our phone in our near premises.We may think that,we can just make call to that phone and find the location.but if the phone is in silent mode what we can do? how can we find the mobile location?

In that situation this project is very helpful.In this project,just by sending a single message to our phone we are changing our phone mode to ringing or silent or vibrate whatever we want.

**1.2 Existing System:**

Till now there is no such application for android mobiles to change mode by sending a single message. So it is difficult to find the location of your phone incase if we lost when it is in silent mode.

**1.3 Proposed System:**

To overcome the drawback in the existing System we are going to develop this project which makes the mobile set back to ringing mode.

**1.4 Advantages:**

* Giving the customized codes for each mode (Silent, Ring, Vibrate)
* Using that customized codes, changing our phone mode to silent, ring or vibrate.

**1.5 Modules:**

1. Setting the customized Codes.
2. Changing the phone mode
3. UI Module.

**1.5.1 Setting the customized Code:**

In this module, we need to set code for each mode i.e. silent, ring and vibrate. By default values will be silent for silent mode, ring for Ring mode and vibrate for vibrate mode. User can also change these values.

**1.5.2 Changing the phone mode:**

Whatever the codes you have given, just send that code as message to your phone. Then your phone will be changed to that particular mode what u have sent.

**1.6 SCOPE OF PROJECT**

One of the fastest growing industries is mobile industry. There are many competitors in this area who are doing research and development on new platforms & user experience. One such technology is Android from Google which is supported for Google phones. These phones are described as next Generation mobiles [As described by Google]. Developing application for such mobile phones using the open source android SDK is quite interesting. This makes the application call history quite easy, efficient, flexible and economic.

**2. LITERATURE SURVEY**

**2.1. JAVA**

Java  is a general-purpose [computer programming language](https://en.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere)" (WORA), meaning that [compiled](https://en.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to [bytecode](https://en.wikipedia.org/wiki/Java_bytecode) that can run on any [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) regardless of [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture). As of 2016, Java is one of the most [popular programming languages in use](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity), particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by [James Gosling](https://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems)(which has since been [acquired by Oracle Corporation](https://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](https://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B), but it has fewer [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) facilities than either of them.

The original and [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) Java [compilers](https://en.wikipedia.org/wiki/Compiler), virtual machines, and [class libraries](https://en.wikipedia.org/wiki/Library_(computing)) were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the [Java Community Process](https://en.wikipedia.org/wiki/Java_Community_Process), Sun [relicensed](https://en.wikipedia.org/wiki/Software_relicensing) most of its Java technologies under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License). Others have also developed alternative implementations of these Sun technologies, such as the [GNU Compiler for Java](https://en.wikipedia.org/wiki/GNU_Compiler_for_Java) (bytecode compiler), [GNU Classpath](https://en.wikipedia.org/wiki/GNU_Classpath) (standard libraries), and [IcedTea](https://en.wikipedia.org/wiki/IcedTea)-Web (browser plugin for applets).

The latest version is [Java 8](https://en.wikipedia.org/wiki/Java_version_history) which is the only version currently supported for free by Oracle, although earlier versions are supported both by Oracle and other companies on a commercial basis.

**2.2 ANDROID**

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming **Application framework** enabling reuse and replacement of components **Dalvik virtual machine** optimized for mobile devices **Integrated browser** based on the open source [WebKit](http://webkit.org/) engine **Optimized graphics** powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional) **SQLite** for structured data storage **Media support** for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF) **GSM Telephony** (hardware dependent) **Bluetooth, EDGE, 3G, and WiFi** (hardware dependent) **Camera, GPS, compass, and accelerometer** (hardware dependent) **Rich development environment** including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE

**2.2.1 About Native code**:

Libraries written in C and other languages can be compiled to ARM native code and installed, but the Native Development Kit is not yet officially supported by Google. Native classes can be called from Java code running under the Dalvik VM using the System. Load Library call, which is part of the standard Android Java classes.

**2.2.2 Creating an android project**

The ADT plug-in provides a New Project Wizard that you can use to quickly create a new Android project (or a project from existing code). To create a new project:

* Select File > New > Project.
* Select Android > Android Project, and click Next.
* Select the contents for the project:
* Enter a Project Name. This will be the name of the folder where your project is created.
* Under Contents, select Create new project in workspace. Select your project workspace location.
* Under Target, select an Android target to be used as the project's Build Target. The Build Target specifies which Android platform you'd like your application built against.
* Unless you know that you'll be using new APIs introduced in the latest SDK, you should select a target with the lowest platform version possible, such as Android 1.1.
* Under Properties, fill in all necessary fields :

Enter an Application name. This is the human-readable title for your application — the name that will appear on the Android device.

1. Enter a Package name. This is the package namespace (following the same rules as for packages in the Java programming language) where all your source code will reside.
2. Select Create Activity (optional, of course, but common) and enter a name for your main Activity class.
3. Enter a Min SDK Version. This is an integer that indicates the minimum API Level required to properly run your application. Entering this here automatically sets the min Sdk Version attribute in the [<uses-sdk>](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html) of your Android Manifest file. If you're unsure of the appropriate API Level to use, copy the API Level listed for the Build Target you selected in the Target tab.
4. Click Finish.

**2.2.3 To create an AVD with the AVD manager:**

* Select Window > Android SDK and AVD Manager, or click the Android SDK and AVD Manager icon (a black device) in the Eclipse toolbar.
* In the Virtual Devices panel, you'll see a list of existing AVDs. Click New to create a new AVD.
* Fill in the details for the AVD.
* Give it a name, a platform target, an SD card image (optional), and a skin (HVGA is default).
* Click Create AVD.

When you first run a project as an Android Application, ADT will automatically create a run configuration. The default run configuration will launch the default project Activity and use automatic target mode for device selection (with no preferred AVD).

**2.2.4 To Create or Modify a Launch Configuration**

Follow these steps as appropriate for your Eclipse version:

* Open the run configuration manager.
* In Eclipse 3.3 ,select **Run > Open Run Dialog** (or Open Debug Dialog)
* In Eclipse 3.4 (Ganymede), select **Run > Run Configurations** (or Debug Configurations)
* Expand the Android Application item and create a new configuration or open an existing one

**2.3 SQLITE**

**2.3.1 DESIGN:**

Unlike client-server database management systems, the SQLite engine is not a standalone process with which the application program communicates. Instead, the SQLite [library](http://en.wikipedia.org/wiki/Library_%28computing%29) is linked in and thus becomes an integral part of the application program. The library can also be called dynamically. The application program uses SQLite's functionality through simple [function calls](http://en.wikipedia.org/wiki/Subroutine), which reduces [latency](http://en.wikipedia.org/wiki/Latency_%28engineering%29) in database access as function calls within a single process are more efficient than [inter-process communication](http://en.wikipedia.org/wiki/Inter-process_communication).

An Android application should be fast. Well, it's probably more accurate to say that it should be *efficient*. That is, it should execute as efficiently as possible in the mobile device environment, with its limited computing power and data storage, smaller screen, and constrained battery life.

As you develop your application, keep in mind that, while the application may perform well enough in your emulator, running on your dual-core development computer, it will not perform that well when run a mobile device — even the most powerful mobile device can't match the capabilities of a typical desktop system. For that reason, you should strive to write efficient code, to ensure the best possible performance on a variety of mobile devices.

Generally speaking, writing fast or efficient code means keeping memory allocations to a minimum, writing tight code, and avoiding certain language and programming idioms that can subtly cripple performance. In object-oriented terms, most of this work takes place at the *method* level, on the order of actual lines of code, loops, and so on.

**2.3.2 FEATURES:**

* [Transactions](https://www.sqlite.org/transactional.html) are atomic, consistent, isolated, and durable (ACID) even after system crashes and power failures.
* [Zero-configuration](https://www.sqlite.org/zeroconf.html) - no setup or administration needed.
* [Full-featured SQL](https://www.sqlite.org/fullsql.html) implementation with advanced capabilities like [partial indexes](https://www.sqlite.org/partialindex.html), [indexes on expressions](https://www.sqlite.org/expridx.html), [JSON](https://www.sqlite.org/json1.html), and [common table expressions](https://www.sqlite.org/lang_with.html). ([Omitted features](https://www.sqlite.org/omitted.html))
* A complete database is stored in a [single cross-platform disk file](https://www.sqlite.org/onefile.html). Great for use as an [application file format](https://www.sqlite.org/appfileformat.html).
* Supports terabyte-sized databases and gigabyte-sized strings and blobs. (See [limits.html](https://www.sqlite.org/limits.html).)
* Small code [footprint](https://www.sqlite.org/footprint.html): less than 500KiB fully configured or much less with optional features omitted.
* Simple, easy to use [API](https://www.sqlite.org/cintro.html).
* Written in ANSI-C. [TCL bindings](https://www.sqlite.org/tclsqlite.html) included. Bindings for dozens of other languages available separately.
* Well-commented source code with [100% branch test coverage](https://www.sqlite.org/testing.html#coverage).
* Available as a [single ANSI-C source-code file](https://www.sqlite.org/amalgamation.html) that is [easy to compile](https://www.sqlite.org/howtocompile.html) and hence is easy to add into a larger project.
* [Self-contained](https://www.sqlite.org/selfcontained.html): no external dependencies.
* Cross-platform: Android, \*BSD, iOS, Linux, Mac, Solaris, VxWorks, and Windows (Win32, WinCE, WinRT) are supported out of the box. Easy to port to other systems.
* Sources are in the [public domain](https://www.sqlite.org/copyright.html). Use for any purpose.
* Comes with a standalone [command-line interface](https://www.sqlite.org/cli.html) (CLI) client that can be used to administer SQLite database

**3.SYSTEM ANALYSIS**

**3.1 SOFTWARE REQUIREMENTS**

* Java
* Android SDK 1.5 or later
* Eclipse Ganymede IDE
* Operating System can be Windows XP, LINUX and Mac etc.

**3.2 HARDWARE REQUIREMENTS**

1. PROCESSOR : (min) P2 Processor

2. RAM : (min) 128

3. HARD DISK : 100 MB

### **3.3 FEASIBILITY STUDY**

An important outcome of the preliminary investigation is the determination that system requested is feasible. This is to identify the objectives of a new system. Before solving a problem one must know what the problem is. The study is carried out by a small group of people who are familiar with system analysis and design process. Fact finding techniques are used to gather the required information.

The three major areas consider while determining the feasibility of the project are

1. Economic Feasibility
2. Operational Feasibility
3. Technical Feasibility

**3.3.1 Economic Feasibility**

Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system.

* + 1. **Operational Feasibility**

Proposed projects are beneficial only if they can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project:

Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for change, there may be resistance.

Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about a more operational and useful systems.

**3.3.3** **Technical Feasibility**

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many-detailed design of the system, making it difficult to access issues like performance, costs on (on account of the kind of technology to be deployed) etc.

A number of issues have to be considered while doing a technical analysis.

i) Understand the different technologies involved in the proposed system: ii) Find out whether the organization currently possesses the required technologies:

1. **SYSTEM DESIGN**

**4.1 ARCHITECTURE DIAGRAM**

Architecture diagram is a [diagram](http://en.wikipedia.org/wiki/Diagram) of a [system](http://en.wikipedia.org/wiki/System), in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. The block diagram is typically used for a higher level, less detailed description aimed more at understanding the overall concepts and less at understanding the details of implementation.

Fig 4.1 Architecture of Call History

Call History

User

Android SDK

SQLite

Java program

Get call logs

A Call History user for who the application looks like an user interface actually consists of a database called as SQLite that comes along with Android SDK and need no other installation. This is the database that is used to store and retrieve information. This is an application that is developed in java and hence all its features apply here as well such as platform independence, data hiding, portable etc.

**4.2 DATA FLOW DIAGRAMS**

A **data-flow diagram** (**DFD**) is a graphical representation of the "flow" of data through an [information system](http://en.wikipedia.org/wiki/Information_system). DFDs can also be used for the [visualization](http://en.wikipedia.org/wiki/Data_visualization) of [data processing](http://en.wikipedia.org/wiki/Data_processing) (structured design).

On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process. The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

`A DFD is also known as a “bubble chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design.

So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

**4.2.1 DFD Symbols**

In the DFD, there are four symbols

1. A Square defines a source or destination of system data.
2. An arrow identifies data flow. It is the pipeline through which the information flows.
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store ,data at rest or a temporary repository of data

Process that transforms data fow

Source or destination of data

Data flow

Data store

**4.3 UNIFIED MODELING LANGUAGE (UML)**

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language for

* Visualizing
* Specifying
* Constructing
* Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

**Usecase diagram**:



**Class Diagram:**



**Object Diagram:**



**Sequence diagram for setting security codes:**



**Sequence diagram for changing mode:**



**StateChart Diagram:**



Activity Diagram:



Deployement Diagram:



**5. SYSTEM CODING & IMPLEMENTATION**

FirstSplash.xml

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout

xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"*

android:background=*"@drawable/finalconfirm1"*

android:id=*"@+id/lll"* >

</LinearLayout>

SplashUI.java

package com.mode;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.view.ViewGroup;

public class SplashUI extends Activity {

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.firstsplash);

Thread runnerlog=new Thread()

{

public void run()

{

try

{

int logoTimer=0;

while(logoTimer<2000)

{

sleep(100);

logoTimer=logoTimer+100;

}

//startActivity(new Intent("com.mode.HowTo"));

Intent i=new Intent(SplashUI.this,HowTo.class);

startActivity(i);

}catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}finally

{

finish();

}

}

};

runnerlog.start();

}

@Override

protected void onDestroy() {

// TODO Auto-generated method stub

super.onDestroy();

unbindDrawables(findViewById(R.id.lll));

System.gc();

}

private void unbindDrawables(View view) {

if (view.getBackground() != null) {

view.getBackground().setCallback(null);

}

if (view instanceof ViewGroup) {

for (int i = 0; i < ((ViewGroup) view).getChildCount(); i++) {

unbindDrawables(((ViewGroup) view).getChildAt(i));

}

((ViewGroup) view).removeAllViews();

}

}

}

Howto.xml

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<ScrollView

xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:id=*"@+id/scrollView1"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:background=*"@drawable/thk2"*

*>*

<LinearLayout

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"*

android:layout\_marginLeft=*"6px"*

android:layout\_marginRight=*"6px"*

>

<TextView android:text=*"How to use this app"*

android:id=*"@+id/tView121"*

android:textColor=*"#ffffff"*

android:textStyle=*"bold"*

android:textSize=*"20px"*

android:layout\_gravity=*"center"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginTop=*"6px"*

android:layout\_marginBottom=*"15px"*

></TextView>

<TextView android:text=*"@string/s1"*

android:id=*"@+id/tView1"*

android:textStyle=*"bold"*

android:textSize=*"15px"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"10px"*

android:layout\_marginRight=*"10px"*></TextView>

<TextView android:text=*"@string/howtotext1"*

android:id=*"@+id/tView1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"10px"*

android:layout\_marginRight=*"10px"*></TextView>

<View

android:layout\_marginTop=*"6px"*

android:layout\_marginBottom=*"6px"*

android:layout\_marginLeft=*"10px"*

android:layout\_marginRight=*"10px"*

android:background=*"#C2DFFF"*

android:layout\_height=*"1dip"*

android:layout\_width=*"fill\_parent"*

/>

<TextView android:text=*"@string/s3i"*

android:id=*"@+id/tView1"*

android:textStyle=*"bold"*

android:textSize=*"15px"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"10px"*

android:layout\_marginRight=*"10px"*></TextView>

<TextView android:text=*"@string/s3id"*

android:id=*"@+id/tView1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"10px"*

android:layout\_marginRight=*"10px"*></TextView>

<Button android:layout\_width=*"fill\_parent"*

android:id=*"@+id/startb"* android:text=*"Start App"*

android:layout\_marginTop=*"10px"*

android:layout\_marginBottom=*"10px"*

android:layout\_height=*"50px"*></Button>

</LinearLayout>

</ScrollView>

HowTo.java

package com.mode;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.Window;

import android.widget.Button;

public class HowTo extends Activity {

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

//requestWindowFeature(Window.FEATURE\_CUSTOM\_TITLE);

setContentView(R.layout.howto);

Button b=(Button)findViewById(R.id.startb);

b.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

// TODO Auto-generated method stub

Intent it=new Intent(HowTo.this,main.class);

startActivity(it);

}

});

}

}

Main.xml

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<TableLayout android:id=*"@+id/TableLayout01"*

android:background=*"@drawable/bc"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

xmlns:android=*"http://schemas.android.com/apk/res/android"*>

<TableRow android:id=*"@+id/TableRow01"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<TextView android:text=*"SwitchTo Silent"*

android:id=*"@+id/uname"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

</TextView>

<EditText android:text=*""*

android:id=*"@+id/silent"*

android:layout\_width=*"180dip"*

android:layout\_height=*"wrap\_content"*

>

</EditText>

</TableRow>

<TableRow android:id=*"@+id/TableRow02"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<TextView android:text=*"SwitchTo Ring"*

android:id=*"@+id/pass"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

</TextView>

<EditText android:text=*""*

android:id=*"@+id/ring"*

android:layout\_width=*"180dip"*

android:layout\_height=*"wrap\_content"*>

</EditText>

</TableRow>

<TableRow android:id=*"@+id/TableRow03"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<TextView android:text=*"SwitchTo Vibrate"*

android:id=*"@+id/vibr"*

android:layout\_width=*"180dip"*

android:layout\_height=*"wrap\_content"*>

</TextView>

<EditText android:text=*""*

android:id=*"@+id/vibrate"*

android:layout\_width=*"180dip"*

android:layout\_height=*"wrap\_content"*>

</EditText>

</TableRow>

<TableRow android:id=*"@+id/TableRow06"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<AbsoluteLayout

android:id=*"@+id/AbsoluteLayout01"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

<Button android:text=*"Save"*

android:layout\_x=*"120px"*

android:layout\_y=*"30px"*

android:id=*"@+id/Save"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*>

</Button>

</AbsoluteLayout>

</TableRow>

</TableLayout>

**Main.java**

package com.mode;

import java.io.IOException;

import java.util.ArrayList;

import android.app.Activity;

import android.content.Context;

import android.database.SQLException;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

public class main extends Activity { Dbhandler myDbHelper;

SQLiteDatabase Mydatabase;

ArrayList<String> aa;

EditText silent,ring,vibrate;

Button save;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

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

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

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

save=(Button)findViewById(R.id.Save);

this.myDbHelper=new Dbhandler(this);

FetchingData();

Mydatabase=myDbHelper.getReadableDatabase();

aa=this.myDbHelper.getModes(Mydatabase);

System.out.println("values of Db "+aa);

String s1= aa.toString();

s1=s1.substring(1, s1.length()-1);

//s1=s1.replace("", newChar)

String[] s=s1.split(",");

silent.setText(s[0]);

ring.setText(s[1]);

vibrate.setText(s[2]);

save.setOnClickListener(new OnClickListener(){

@Override

public void onClick(View arg0) {

// TODO Auto-generated method stub

String stxt=silent.getText().toString();

String rtxt=ring.getText().toString();

String vtxt=vibrate.getText().toString();

if(stxt.equals("") || rtxt.equals("") || vtxt.equals(""))

{

Toast.makeText(getApplicationContext(), "Please fill all fields", 60).show();

}

else

{

/\*Context context = getApplicationContext();

CharSequence error = "Please enter a track name" +stxt;

int duration = Toast.LENGTH\_LONG;

Toast toast = Toast.makeText(context, error, duration);

toast.show();\*/

myDbHelper=new Dbhandler(main.this);

FetchingData();

Mydatabase=myDbHelper.getReadableDatabase();

Mydatabase.execSQL("update mode set text='"+stxt+"' where Mode='silent'");

System.out.println("^^^^^^^^^^^^^^^ inserted value is "+stxt);

Mydatabase.execSQL("update mode set text='"+rtxt+"' where Mode='ring'");

Mydatabase.execSQL("update mode set text='"+vtxt+"' where Mode='vibrate'"); Toast.makeText(getApplicationContext(), stxt, 70).show();

}

}

});

}

private void FetchingData()

// TODO Auto-generated method stub

try {

myDbHelper.onCreateDataBase();

} catch (IOException ioe) {

throw new Error("Unable to create database");

}

try {

myDbHelper.openDataBase();

Mydatabase = myDbHelper.getWritableDatabase();

System.out.println("executed");

}catch(SQLException sqle){

throw sqle;

}

// TODO Auto-generated method stub

}

}

Broadcast.java

package com.mode;

import java.io.IOException;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.database.Cursor;

import android.database.SQLException;

import android.database.sqlite.SQLiteDatabase;

import android.media.AudioManager;

import android.os.Bundle;

import android.telephony.SmsMessage;

import android.util.Log;

import android.widget.Toast;

public class Broadcast extends BroadcastReceiver {

// private ITelephony telephonyService;

private static final String SMS\_RECEIVED = "android.provider.Telephony.SMS\_RECEIVED";

private static final String TAG = "SMSBroadcastReceiver";

Dbhandler myDbHelper;

private AudioManager maudio;

SQLiteDatabase Mydatabase;

SQLiteDatabase db;

Context context = null;

@Override

public void onReceive(Context context, Intent intent) {

// TODO Auto-generated method stub

Log.i(TAG, "Intent recieved: " + intent.getAction()); maudio=(AudioManager)context.getSystemService(context.AUDIO\_SERVICE);

if (intent.getAction().equals(SMS\_RECEIVED)) {

Bundle bundle = intent.getExtras();

if (bundle != null) {

Log.i(TAG, "Message recieved: ");

Object[] pdus = (Object[])bundle.get("pdus");

final SmsMessage[] messages = new SmsMessage[pdus.length];

for (int i = 0; i < pdus.length; i++) {

messages[i] = SmsMessage.createFromPdu((byte[])pdus[i]);

System.out.println("message is.."+messages[i]);

Log.i(TAG, "Message recieved: " + messages[0].getMessageBody());

String mode=messages[0].getMessageBody();

System.out.println("message body is.."+mode);

String m\_mode=getMode(mode,context);

System.out.println("mode is...."+m\_mode);

changemode(m\_mode);

}

if (messages.length > -1) {

Log.i(TAG, "Message recieved: " + messages[0].getMessageBody());

}

}

}

else

{

Log.i(TAG, "in else... ");

}

}

private void changemode(String m\_mode) {

// TODO Auto-generated method stub

if(m\_mode.equals("silent"))

{

System.out.println("The phone state is changing to silent mode");

// int n=maudio.getRingerMode();

// System.out.println("The phone state is changing to silent mode "+n);

maudio.setRingerMode(AudioManager.RINGER\_MODE\_SILENT);

Log.i(TAG, "Changed to Silent");

}

else if(m\_mode.equals("ring"))

{

System.out.println("The phone state is changing to ring mode");

maudio.setRingerMode(AudioManager.RINGER\_MODE\_NORMAL);

Log.i(TAG, "Changed to Ring ");

}

else if(m\_mode.equals("vibrate"))

{

System.out.println("The phone state is changing to vibrate mode");

maudio.setRingerMode(AudioManager.RINGER\_MODE\_VIBRATE);

Log.i(TAG, "Changed to Vibrate");

}

else if(m\_mode.equals("") || m\_mode==null)

{

Log.i(TAG, "Other message Recieved...");

// Toast.makeText(context,"Other message Recieved...",Toast.LENGTH\_LONG).show();

}

}

private String getMode(String mode,Context context) {

// TODO Auto-generated method stub

String m="";

this.myDbHelper=new Dbhandler(context);

FetchingData();

System.out.println("inside getmodde");

Mydatabase=myDbHelper.getReadableDatabase();

Cursor c=Mydatabase.rawQuery("SELECT Mode FROM mode where text='"+mode+"' ", null);

System.out.println("cursor is.."+c);

c.moveToFirst();

if(c!=null){

int i=c.getColumnIndex("Mode");

System.out.println("column index is.."+i);

m =c.getString(i).toString();

System.out.println("mode in getmode.."+m);

}

return m;

}

private void FetchingData() {

// TODO Auto-generated method stub

try {

myDbHelper.onCreateDataBase();

} catch (IOException ioe) {

throw new Error("Unable to create database");

}

try {

myDbHelper.openDataBase();

db = myDbHelper.getReadableDatabase();

System.out.println("executed");

}catch(SQLException sqle){

throw sqle;

}

// TODO Auto-generated method stu

}

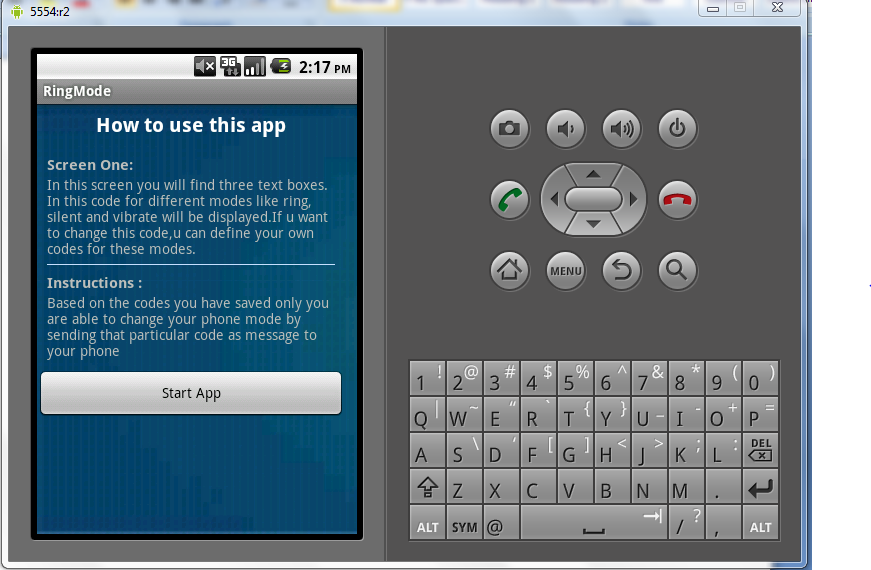
}

**5.2 RingMode TaskList :**

**Task:1** Create a splash screen with an image as follows..

****

**Task:2**  Create a screen as follws with instructions about how to use application and with a button called startapp.



**Task:3**

In the background create SQLite databse manually and insert values as silent ,ring and vibrate.Design the screen as follows and retrive the values from database and append those values to below textboxes.and also write code for database updations.



**Task:4**

Do u sample on Broadcast receivers..

**6. TESTING AND VALIDATION**

The development of software systems involves of a series of production activities where opportunities for injection of human fallibilities are enormous. Errors may begin to occur at the very inception of the process where the objectives may be erroneously or imperfectly specified, as well as in later design and development stages. Because of human inability to perform and communicate with perfection, software development is accompanied by a quality assurance activity.

# **TESTING TECHNIQUES**

Testing is the process of executing a program with the intention of finding errors. The various test strategies used for testing the software are as follows.

**6.1 Unit Testing**

Unit testing focuses on verification effort on the smallest unit of the software design module. The main goal is to make sure that every source statement and logic path has been executed correctly at least once. The output of this stage is the source code.

**6.2 Integration Testing**

In Integration testing, we find errors that have occurred during the integration. After testing each module, which is then integrated into subsystems and then to form the entire system on which integration testing is performed. The goal of testing is to detect the design errors, while focusing on the testing the interconnection between modules.

## **6.3 Validation Testing**

This testing concentrates on confirming that the software is error-free in all respects. All the specified validations are verified and the software is subjected to hard-core testing. It also aims at determining the degree of deviation that exists in the software designed from the specification; they are listed out and are corrected.

**6.4 System Testing**

In this testing, the system is tested for the errors after coupling all the modules together The system is tested against the specified requirements to see if all the requirements are met and the system performs as specified by the requirements.

**7. CONCLUSION**

This application is used for changing phone mode. This application is build in Google mobiles using Android SDK. It is a tool developed for android platform, which is used to search various books and their related information within the mobile. This is an advantage when compared to existing system because a single mobile piece is enough for deploying the application .As this is a mobile application one can easily search for required information. One can search for books whenever one wants to without waiting for some system. This makes this application efficient, convenient and easy to use along with providing maximum user satisfaction which is the key aspect for any developer.

**8.REFERENCES**

1. Herbert Schildt.2008 ,”Java Complete Reference”, Tata McGraw-Hill ,

7th Edition, pp.177-180 .

1. Grady Booch, James Rambaugh.1998,“UnifiedModeling Language User Guide”, Addison Wesley Publishing, chapter 8-31.

**WEBSITES:**  **REFERRED URLS:**

www.android.com http://developer.android.com/index.html

www.google.com http://en.wikipedia.org/wiki/SQLite