**Auto Call Answering Application for Android**

****

**logo_android**

**Clever Call Handler**



**State University of Bangladesh**

**CSE-400**

Submitted in partial fulfillment of the requirement for the award of

**Bachelor of**

**Computer Science & Engineering**

**Submitted By**

Elius Ahmed UG02-28-11-011

Mehrab Sharmin Tora UG01-23-10-017

**Under the Guidance Of**

**Supervisor Co-supervisor**

Dr. Bilkis Jamal Ferdosi Debbrata Kumar Saha

Head, Dept. of CSE Lecturer, Dept. of CSE

State University of Bangladesh State University of Bangladesh

**Department of Computer Science and Engineering**

Acknowledgement

I am very grateful to Almighty Allah that finally I have successfully finished the work in time. I am indebted to my beloved mother and my idol, honorable teacher Ishtiaque Hossain to help me bountifully in my critical situations.

I would like to express my special appreciation and thanks to my respected teacher Dr. Bilkis Jamal Ferdosi for her excellent guidance & respected teacher Debbrata Kumar Saha for his dedicated assistance to complete this project. Their relentless mentoring and encouragement at last made me possible to successfully completion of my project. They have been a tremendous mentor for me. I would like to thank them for encouraging me. Their advice on both project as well as on my career have been priceless.

I am also grateful to my team member Elius Ahmed for supporting me to reach the goal.

Mehrab Sharmin Tora

Appendix

**Chapter1: Clever Call Handler**

* 1. *Introduction to Clever Call Handler 8*
  2. *Motivation Behind the work 8*
  3. *Previous Work related to this Application 9*
  4. *Issues with the previous work 9*
  5. *Achievement of This Project 10*
  6. *Infliction 10*

**Chapter2: Foundation**

*2.1 Equipment 11*

*2.2 Introduction to Android Application 11*

*2.3 Application Framework 12*

*2.4 Applications 13*

*2.5 Development Toolkit 20 17*

**Chapter3:** Design and User Interface

*3.1 Flow Chart for the application working procedure 19*

*3.2 Designing Activity 20*

*3.3 Application & Notification icon 20*

*3.4 User Interface: 21*

**Chapter4: Result & Discussion 33**

*4.1 Introduction: 22*

*4.2 Application Information & Result 22*

*4.3 Snapshot of outputs: 22*

*4.3.1 Choosing Application form the Apps: 23*

*4.3.2 Application Preference Window 24*

*4.3.3 Notification Appears After Enabling Application 25*

*4.3.4 Delayer Option Changing Dialog Box 26*

*4.3.5 Contacts Chooser Dialog Box 27*

*4.3.6 Bluetooth Option 28*

*4.3.7 Speaker Option 29*

*4.3.8 Choosing Off-Hook mode 30*

*4.3.9 Select Service to Edit/Cancel from Notification Status Bar 31*

**Chapter3: Coding and Manipulating**

*5.1 Files and Folders 32*

*5.2 aidl file 33*

*5.3 CCHReceiver.java 34*

*5.4 CCHTaskerReceiver.java 34*

*5.5 CCHBootReceiver.java 34*

*5.6 CCHNotifier.java 35*

*5.7 CCHPreferenceActivity.java 35*

*5.8 CCHIntentService.java 35*

*5.9 BluetoothHeadset.java 36*

*5.10 Manifest file 36*

*5.11 bin 37*

**References: 38**

Abstract

Modern hand held devices such as smart phones have become increasingly powerful in recent years. Dramatic breakthroughs in processing power along with the number of extra features included in these devices have opened the doors to a wide range of commercial possibilities. As mobile devices become more like PCs they will come to replace objects we tend to carry around such as checkbooks, credit cards, cameras, planners, mp3 players, etc. In short, we will be using them to accomplish our daily tasks. One application that falls into this category is the Auto Answering Phone Call Application developed for the Android Smart Phones.

The prime objective of “Clever Call Handler” is to answer the phone call without using hand, while busy with other works. User will pre-set the application to accept the call. One can choose the caller to receive, can set a time delayer and choose Speaker/ Bluetooth through which the call will be received.

**Chapter 1**

**Clever Call Handler**

**1.1 Introduction to Clever Call Handler:**

The purpose of our project is to increase the features and efficiency of the Auto Answering Application, which will be able to meet the user need. In Auto Answering Application our goal was to enhance the Application with some more features like accept the call with Bluetooth, delay before answering the call, will not answer when a call is already in progress, recording the conversation, reject a call by texting etc. But, some features call rejecting, conversation recording couldn’t be added. So, this application now works for the other features.

**1.2 Motivation behind the work:**

My grandfather is the cause to motivate me in this sector. He is sick and can’t attend the call timely and also doesn’t see well. Thus I’ve decided to create such an Application, which would select the callers, can receive call in time while ringing. At last I’ve succeed to create such an application that will help him. Other persons, who have the problem like him, can be benefited from this application. Another reason behind this work is that, I am interested to work in this sector in future also.

**1.3** **Previous Work related to this Application:**

Auto Answering Application is a very simple android application to answer the phone automatically when it rings. Can be limited to just contacts or starred contacts. Also has a speakerphone option. It is a Google Project. There are already a number of project related to this work, but with a lot of problems. In the next section the problems will be discussed.

**1.4 Bug with the previous work:**

Right now the issues of Google Project related to this wok “Auto Answering Application” are:

* Wired Headset Detection Support
* Selectable Timeout
* Notification Icon in BT headset mode
* Audio goes to phone speaker rather than connected BT headset
* Is this app still active?
* Spanish Translation
* Auto launch App on open
* Not Working Htc evo2.3
* Crash on Htc sensation
* Stops on Droid X2
* Quit working with droid 2.3.4 update
* Establish conference call for arriving new call
* Force Speaker doesn’t work on Samsung S2

|  |
| --- |
| * Detect headset Plug-in * Play recorded sound |
| **1.5 Achievement of This Project:** |
|  |
|  |

Our approach was to reduce these issues and add some more options like auto rejecting and recording voice call. But there are some problems with the auto rejecting call. Google has reduced the option to auto reject call in the Application Framework level. It can be only rooted in the kernel level because of synchronization problem and must need user interaction to reject the call. For this reason we couldn’t to implement this part of work. Another disability is the recording of call; we could not cover and add this feature because of time shortage.

Finally the project “Clever Call Handler” can work with other features well and good with the new addition of Bluetooth, time delayer, and Off-Hook mode.

**1.6 Infliction:**

The functionalities of our “Clever Call Handler” is a user friendly Application. It can be used in several sectors such as the appliance of this Application is as the Car Kit. It can be also used while busy with household works or also with official work. It can be very useful for the disabled persons who can’t use their hands or doesn’t see well. Especially for the old persons, it will be very useful. Sometimes it is very risky to receive call while passing the road. So this can be used to become alert and out of danger from the snatchers. It will also be helpful for that user, who has to attend call very frequently. It is irritating to attend the call manually when it is very frequent. So, it will be a beneficial sector.

**Chapter 2**

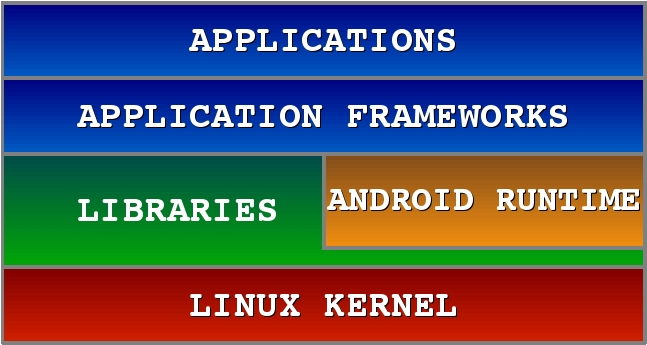
**Foundation**

**2.1 Equipment:**

The Project is developed in Java Programming Language by using the Eclipse Ganymede Integrated Development Environment (IDE). We use the Android Software Development Kit (SDK) which includes a variety of custom tools that help us develop mobile applications on the Android platform. The most important of these are the Android Emulator and the Android Development Tools (ADT) plug-in for Eclipse.

**2.2 Introduction to Android Application**

Android is an S/W stack for mobile devices developed and managed by OHA.A free S/W under Apache License.



Java

C/C++

Kernel

*Fig: Application Software Stack*

**Kernel:** It is Hardware Abstraction Layer, Linux 2.6 Kernel, and Unix C APIs are available.

**Libraries [C/C++] :** Surface Manager composing windows on the screen are having SGL 2D Graphics, OpenGL|ES 3D Library, Media FW, Audio/Video Codec, FreeType Font rendering, WebKit Browser Engine, System C libraries, SQLite, OpenSSL.

**Android Runtime:** It is the DalvikVM. It is Optimized for low memory requirements. Designed to allow multiple VM instances to run at once. Every program runs in its own process / own VM instance. Relies on the underlying OS for process isolation, memory management and threading support. Operates on DEX files.

**2.3 Application Framework:**



Enabling and simplifying the reuse of components Developers have full access to the same framework APIs used by the core applications. Users are allowed to replace components.

|  |  |
| --- | --- |
| **Feature** | **Role** |
| View System | Used to build an application, including lists, grids, text boxes, buttons, and embedded web browser |
| Content Provider | Enabling applications to access data from other  applications or to share their own data |
| Resource Manager | Providing access to non-code resources (localized string, graphics, and layout files) |
| Notification Manager | Enabling all applications to display customer alerts in the status bar |
| Activity Manager | Managing the lifecycle of applications and providing a common navigation back stack |

**Table: Some resource manager and their job**

**2.4 Applications:**

An application carries Activity, Service, Broadcast receiver, Content Provider, Intent, Android Manifest. Good separation (and corresponding security) from other applications: Each application runs in its own process, each process has its own separate VM, each application is assigned a unique Linux user ID – by default files of that application are only visible to that application (can be explicitly exported).

**Activities (The View):**

Each activity has a default window to draw in (although it may prompt for dialogs or notifications). The content of the window is a view or a group of views (derived from View or ViewGroup). Example of views: buttons, text fields, scroll bars, menu items, check boxes, etc. View(Group) made visible via Activity.setContentView() method.

**Services:**

It does not have a visual interface. Runs in the background indefinitely. Examples: Network Downloads, Playing Music, TCP/UDP Server etc. It can be bind to an existing service and controlled its operation.

**Broadcast receiver:**

Receive and react to broadcast announcements. Extend the class BroadcastReceiver. Examples of broadcasts: Low battery, power connected, shutdown, timezone changed, etc. Other applications can initiate broadcasts.

**Content provider:**

It makes some of the application data available to other applications. It is the only way to transfer data between applications in Android (no shared files, shared memory, pipes, etc.) Extends the class ContentProvider; Other applications use a ContentResolver object to access the data provided via a ContentProvider.

**Intent:**

Intent is an Intent object with a message content. Activities, services and broadcast receivers are started by intents. ContentProviders are started by ContentResolvers: An activity is started by Context.startActivity(Intent intent) or Activity.startActivityForResult(Intent intent, int RequestCode). A service is started by Context.startService(Intent service). An application can initiate a broadcast by using an Intent in any of Context.sendBroadcast(Intent intent), Context.sendOrderedBroadcast() or Context.sendStickyBroadcast().

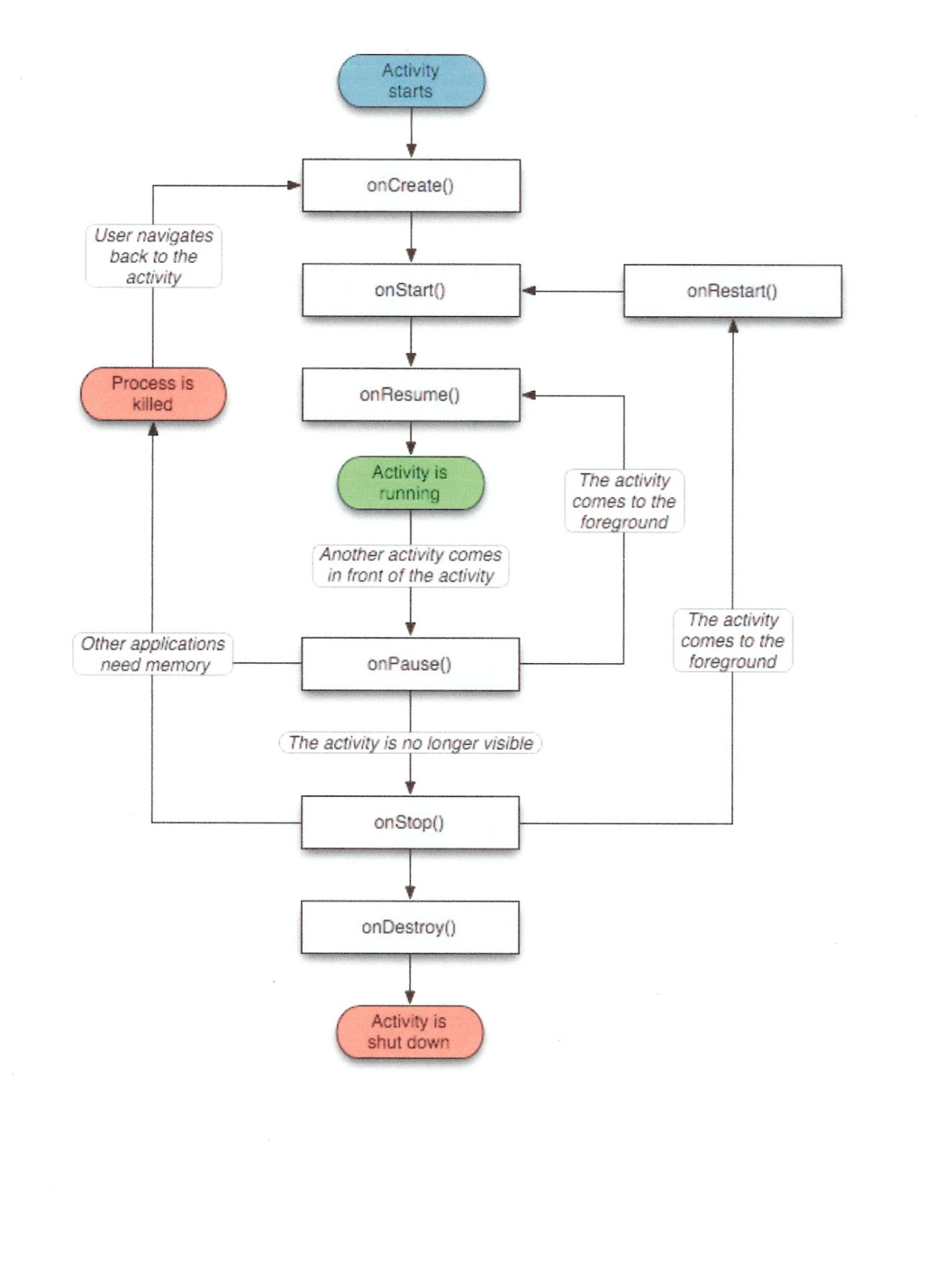
**Manifest:**

Its main purpose in life is to declare the components to the system.

**Activity Life Cycle:**

When this activity has the focus and when you click on the HOME button, the onPause() and onStop() callbacks are invoked.

Now, if you start the application again, the onCreate(), onStart() and onResume() callbacks are invoked.

****

**Fig: Activity Life Cycle**

**Shutting Down Components:**

* Activities**:** Can terminate itself via finish(), Can terminate other activities it started via finishActivity().
* Services**:** Can terminate via stopSelf() or Context.stopService();
* Content Providers: Are only active when responding to ContentResolvers.
* Broadcast Receivers: Are only active when responding to broadcasts

**2.5 Development Toolkit:**

**Android SDK:**

The Android SDK includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator (based on QEMU), documentation, sample code, and tutorials. Currently supported development platforms include x86- architecture computers running Linux (any modern desktop Linux distribution), Mac OS X 10.4.8 or later, Windows XP or Vista. Requirements also include Java Development Kit, Apache Ant, and Python 2.2 or later. The officially supported integrated development environment (IDE) is Eclipse (3.2 or later) using the Android Development Tools (ADT) Plugin, though developers may use any text editor to edit Java and XML files then use command line tools to create, build and debug Android applications.

**Eclipse:**

In computer programming, Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Written mostly in Java, Eclipse can be used to develop applications in Java.

**ADT Plugin**

The Android Development Tools (ADT) plugin for Eclipse adds powerful extensions to the Eclipse integrated development environment. It allows you to create and debug Android applications easier and faster.

**Emulator**

The Android SDK includes a mobile device emulator - a virtual mobile device that runs on your computer. The emulator lets you prototype, develop, and test Android applications without using a physical device. The Android emulator mimics all of the typical hardware and software features of a typical mobile device, except that it can place actual phone calls. It provides a variety of navigation and control keys, which you can "press" using your mouse or keyboard to generate events for your application. It also provides a screen in which your application is displayed, together with any other Android applications running.

**Chapter 3**

**Design and User Interface**

**3.1 Flow Chart for the application working procedure :**

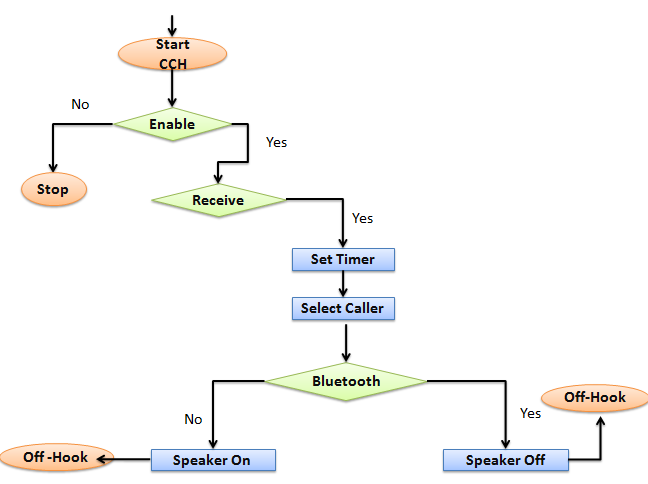


Fig: Flow Chart of the working procedure

At first the user has to set the preference “Enable Answering” to start the actions. Then other settings will have to set like, after enabling the application the user should choose the caller, whether he wants to pick call from all the callers, or only from contacts or starred list. Then user should set the time delayer within which s/he wants to pick up the call. Then user will have to Choose the Bluetooth/Speaker option, though which s/he wants to receive call.

**3.2Designing Activity:**

Preference Screen Activity is used to make the Application faster and light to load. Activity run in the background, whenever an incoming call is detected, the Broadcast receiver immediately notify the receiver and start the service at once if the activity is enabled. If the Loudspeaker need to turn on, the Bluetooth should be disabled. If the Bluetooth is enabled, the Speaker will be disabled. The call will not be received in Off-Hook mode, if “Don’t Answer if in Call” is selected. Otherwise, another call will be received also.

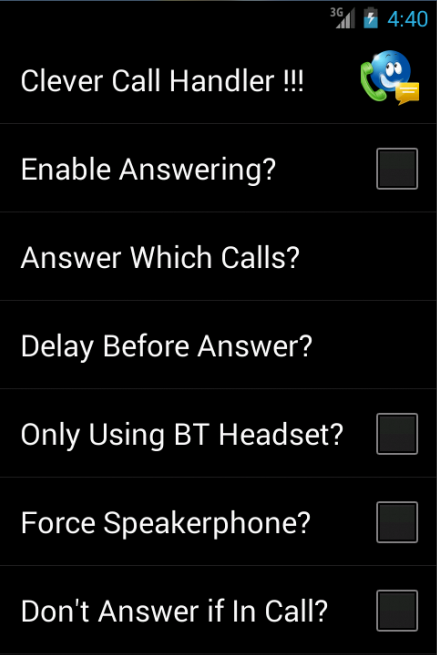
**3.3 Application Launcher & Notification icon :**

****

Application Notification on the status bar and the launcher Icon will be this.

**3.4 User Interface:**

This is actually a preference screen, which is used for editing settings, options. Since the application will mainly run on the background, the simpler and light way of preference activity has chosen.



**Chapter 4**

**Result & Discussion**

**4.1 Introduction:**

This chapter highlights the results of the project & the snapshots for each of the activities. Each snapshot describes every single step of the Clever Call Handler Application.

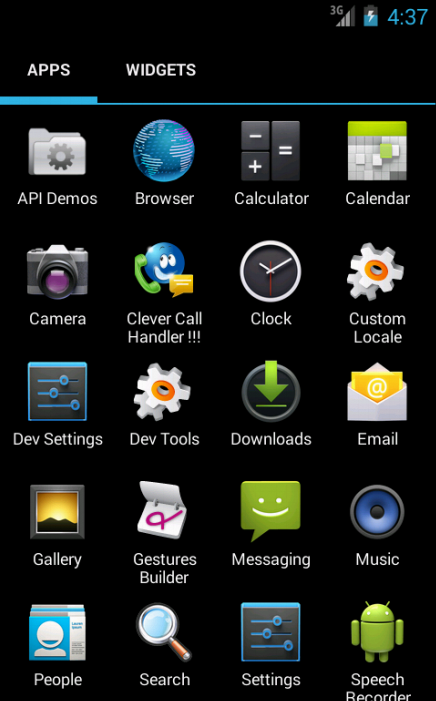
**4.2 Application Information & Result:**

The total size of the Application is 253.2KB.All options are tested and working properly.It should work more efficiently than the previous Applications on GooglePlay. Supported from Gingerbread 2.3 to JellyBean 4.2.

**4.3 Snapshot of outputs:**

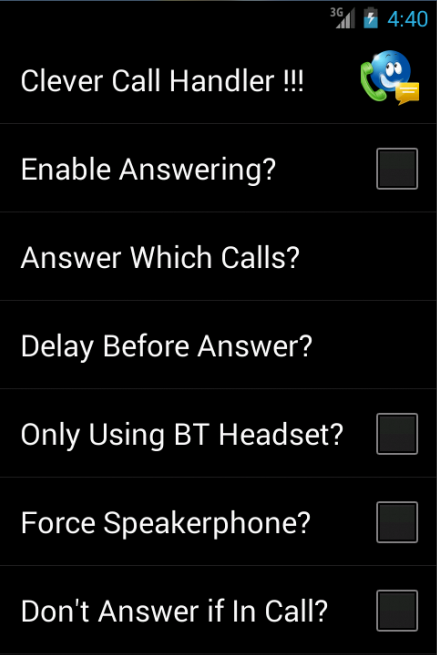
Snapshots are taken from the Emulator. The code was run in the Virtual Device. The snapshots are taken from Emulator.

**4.3.1 Choosing Application from the Apps:**

****

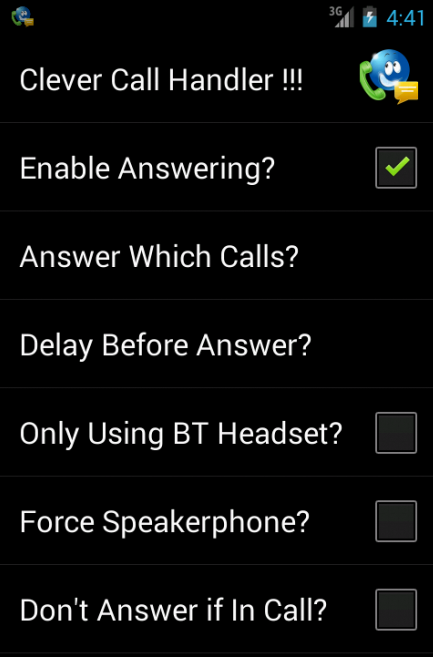
This is the Launcher Icon. If user clicks on it, a preference window will appear where one can choose the options that he/she wants.

**4.3.2 Application Preference Window:**

****

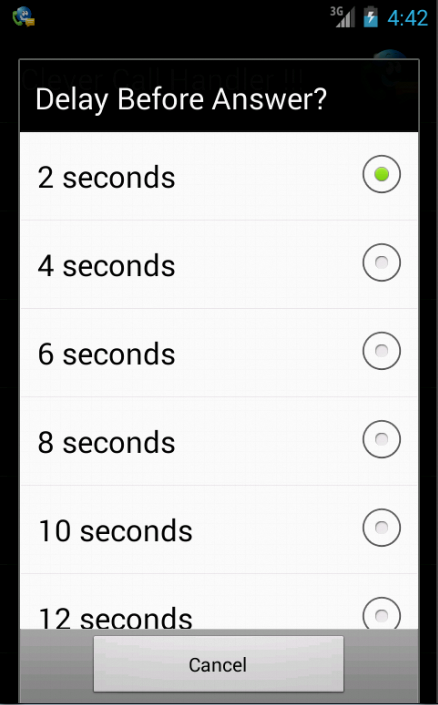
Four Preference checkbox and two list preferences are on the screen. User may edit or select some options from here.

**4.3.3 Notification Appears After Enabling Application:**

****

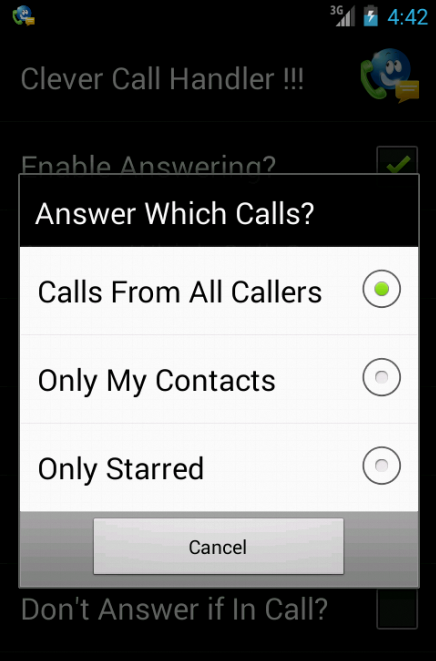
When the “Enable Answering” box will be chosen, the service to detect call will be started on the background. And a notification icon will be appeared on the status bar.

**4.3.4 Delayer Option Changing Dialog Box:**

****

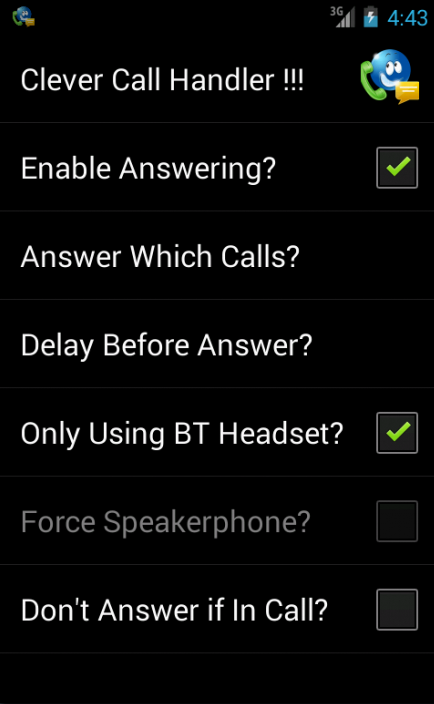
This option is to set the delay timing to receive the call. If the call doesn’t ring for the sudden given time, the call will not be received.

**4.3.5 Contacts Chooser Dialog Box:**

****

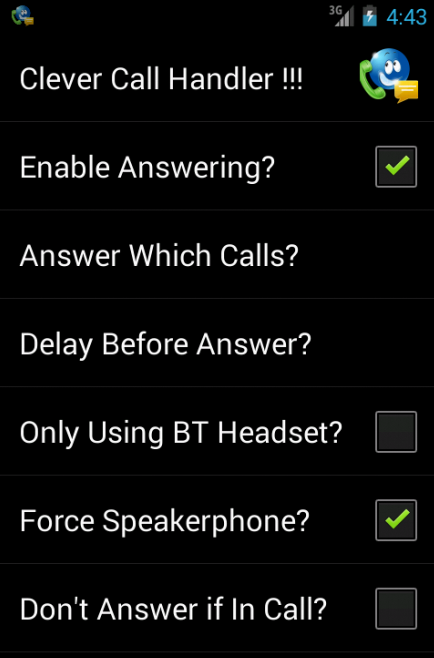
Contacts may be starred, only contacts or all. The user can set that which calls he wants to receive.

**4.3.6 Bluetooth Option:**

****

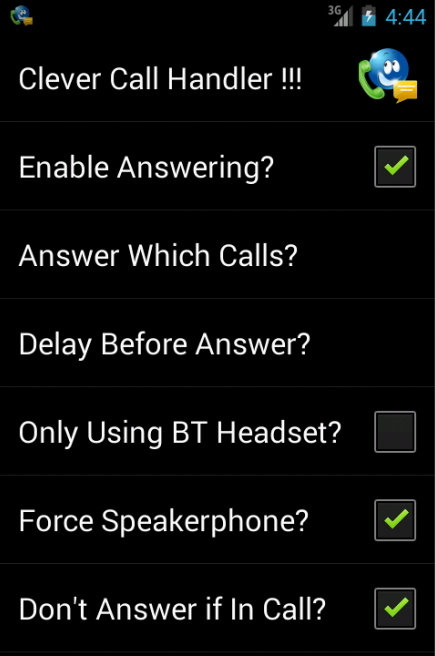
**Choosing Bluetooth option will disable the speakerphone option. The call will be received with Bluetooth Device.**

**4.3.7 Speaker Option:**

****

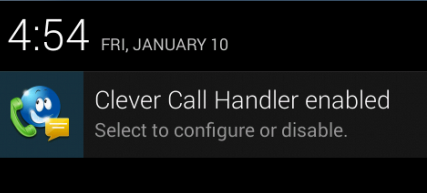
Speaker Choosing will work if Bluetooth is disabled only.

**4.3.8 Choosing Off-Hook mode :**

****

If user is already attending a call, it is good to be alert that if any third party call is arrived, that shouldn’t be received by keeping hold to another call. So this option is good to prefer.

**4.3.9 Select Service to Edit/Cancel from Notification StatusBar:**



Application will be running on the Background. When user chooses the notification from notification bar, the preference activity will appear to edit settings.

## Chapter 5

**Coding and Manipulating**

In this Chapter we will briefly discuss about the coding part. All the files, their works, how they work will be described here.

**5.1 Files and Folders:**

Src:

android.bluetooth

IbluetoothHeadset.aidl

com.android.internal.telephony

ITelephony.aidl

tora.clevercallhandler

BluetoothHeadset.java

CCHBootReceiver.java

CCHIntentService.java

CCHNotifier.java

CCHPreferenceActivity.java

CCHReceiver.java

CCHTaskerReceiver.java

Gen:

android.bluetooth

IbluetoothHeadset.java

com.android.internal.telephony

Itelephony.java

tora.clevercallhandler

BuildConfig.java

Manifest.java

R.java

Bin:

aidl

res:

AndroidManifest.xml

classes.dex

resources.ap\_

ToraCleverCallHandler.apk

Res:

drawable:

ic\_launcher.png

values:

arrays.xml

dimens.xml

strings.xml

styles.xml

xml:

preferences.xml

AndroidManifest.xml

**5.2 aidl file:**

We have two aidl file in the package

**android.bluetooth**

**IbluetoothHeadset.aidl**

**com.android.internal.telephony**

**ITelephony.aidl**

The package name and the aidl file name should have the same name as given here to get work in the system. Actually they are interface file to interface with Bluetooth and Telephony service.

**5.3 CCHReceiver.java**

* Extends BroadcastReceiver Class
* Load preferences
* Detects phone state
* Checks the phone is ringing and CCH enabled, then checks for seond call restriction.
* If all is true, then, loads ModeInCall from AudioManager. if in call mode, then returns.
* Checks for contact restriction.
* Then start service for CCHIntentService.

**5.4 CCHTaskerReceiver**

* extends BroadcastReceiver
* Rechecks data validity for preferences
* then update notification by CCHNotifier.class

**5.5 CCHBootReceiver.java**

* Extends broadcast receiver
* In the on receive method, call CCHNotifier.class and update notification.

**5.6 CCHNotifier.java**

* Loads Preferences, get NOTIFICATIONSERVICE.
* in the updateNotification method, ckecks for enable or disable CCH.
* If enable then, goes to PreferenceActivity, intent to turn on autoanswer, and create notification.
* else, cancel the notifier .

**5.7 CCHPreferenceActivity.java**

* The helper class of CCHNotifier extends PreferenceActivity implements OnSharedPreferenceChangeListener
* The method onSharedPreferenceChanged uses the boolean key enable.
* Keep track with the preferences,
* If enable is true, then update the notifier class to be enabled, else destroy.

**5.8 CCHIntentService.java**

* Extends Intent Service
* Load Preferences,
* If Bluetooth is enabled, receive call with Bluetooth.
* Let the phone ring for a set delay.
* if speaker on, then answer call with speaker

**5.9 BluetoothHeadset.java**

* Prompts for connecting Bluetooth Headset
* Connects with Bluetooth headset
* Handles all the action related to Bluetooth Service

**5.10 Manifest file**

When the application will be installed, it asks for some permission. Permissions needed for this application are:

* Sdk Version=10 to 18
* MODIFY\_PHONE\_STATE
* READ\_PHONE\_STATE
* BLUETOOTH"
* MODIFY\_AUDIO\_SETTINGS
* READ\_CONTACTS
* RECEIVE\_BOOT\_COMPLETED
* USE\_PLUGIN

Reeceiver are: CCHTasker receiver, CCHBootreceiver and CCHReceiver and Service: Intent service.

**5.11** **bin:**

Contains the .apk file. Android application package(APK) is the format used to distribute and install application software and middleware onto [Google](http://en.wikipedia.org/wiki/Google)'s Android operating system.An APK file contains all of that program's code (such as [.dex](http://en.wikipedia.org/wiki/.dex) files), resources, assets, certificates, and manifest file. As is the case with many file formats, APK files can have any name needed, provided that the file name ends in ".apk".APK files are [zip format](http://en.wikipedia.org/wiki/Zip_%28file_format%29) packages based on the [JAR file format](http://en.wikipedia.org/wiki/JAR_%28file_format%29), with .apk as the [filename extension](http://en.wikipedia.org/wiki/Filename_extension).

**References:**

[1]<http://code.google.com/p/auto-answer/>

[2]<http://developer.android.com/index.html>

[3]<https://github.com/search?p=10&q=android&ref=cmdform&type=Repositories>

[4]<http://learnandroideasily.blogspot.com/2013/09/preferenceactivity-in-android-example.html>

[5]<http://www.codeproject.com/Articles/113831/An-Advanced-Splash-Screen-for-Android-App>

[6]<http://www.coreservlets.com/android-tutorial/>

[7]<http://www.ashokgelal.com/>

[8]<http://stackoverflow.com/questions/7751525/android-checkboxpreference-how-to-disable-and-enable-other-preferences-on-prefe>