SYSTEM DESIGN AND ANALYSIS

Design is the first phase in the development for any engineers project .It is a creative process. A good design is a key to effective system. From a project management point of view, software design is conducted in two steps .Preliminary design is concerned with the transformation of requirements in to data and software architecture. Detailed design focuses on the refinement to the architectural representation that lead to detailed algorithm data structure and representation of software.

3.1 DETAILED DESIGN

The design part mainly includes the designing of layouts and a user-defined gesture set. The gesture recognition service can recognize new performed gestures and provide the recognition result to android applications (activities) that subscribe to this service. The result contains the tag of the best matching gesture and the distance of the classified gesture to the best matching gesture. The smaller the distance the better the similarity. The service supports different training sets that allow to personalize recorded gestures or to define specific gesture training sets for individual applications. For gesture recognition the service uses the genetic algorithm - dynamic time warp which has been extended to classify multidimensional signals .The design includes,

3.1.1 Creating the gesture library

Android 1.6 and higher SDK platforms include a new application pre-installed on the emulator, called GestureBuilder(Figure 3.1.1). We can use this application to create a set of pre-defined gestures for our

own application. It also serves as an example of how to let the user define his own gestures in our applications. We can find the source code of Gestures Builders in the samples directory of each SDK platform. In our project we will use Gestures Builder to generate a set of gestures for us. A gesture is always associated with a name. That name is very important because it identifies each gesture within our application. The names do not have to be unique.

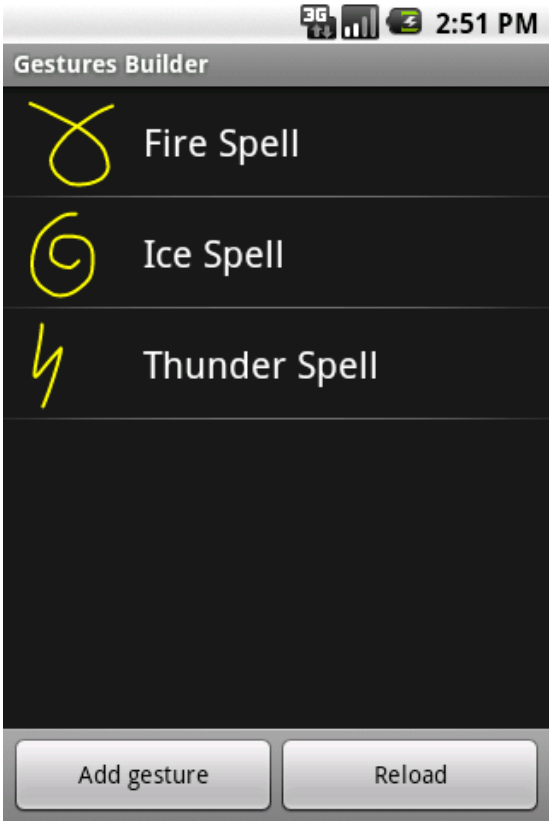


Fig.3.1.1

Actually it can be very useful to have several gestures with the same name to increase the precision of the recognition. Every time we add or edit a gesture in the Gestures Builder, a file is generated on the emulator's SD card, /SD card/gestures. This file contains the description of all the gestures, and you will need to package it inside your application inside the resources directory, in/res/raw.

3.1.2 Loading the gesture library

We have a set of pre-defined gestures, we must load it inside our application. This can be achieved in several ways but the easiest is to use the Gesture Libraries class. Hierarchical representation of library is shown in figure 3.1.2. We can easily load libraries from other sources, like the SD card, which is very important if we want your application to be able to save the library; a library loaded from a raw resource is read-only and cannot be modified. The following diagram shows the structure of a library:

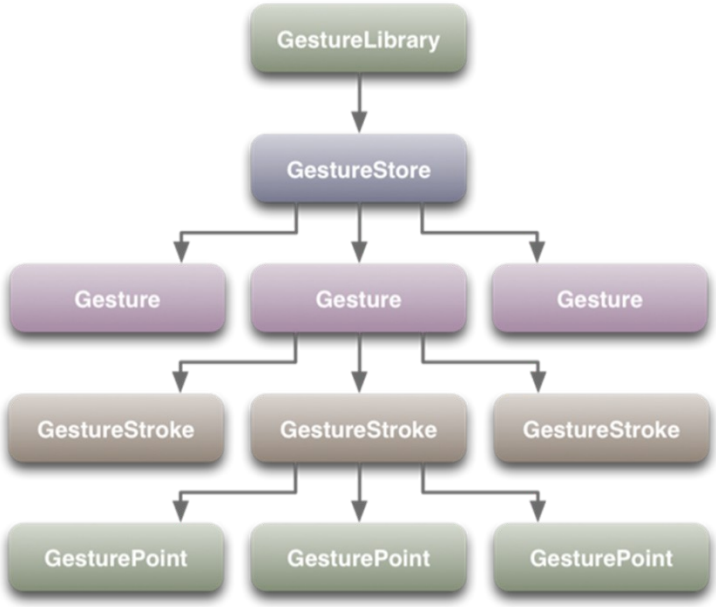


Fig 3.1.2

3.1.3 Recognizing gestures

To start recognizing gestures in our application, we have to add a Gesture Overlay View to our XML layout .We are creating layouts in XML by just make use of drag and drop mechanism .The layouts include layout for gesture builder, gesture library, Renaming window and also input. Gesture overlay view i.e. the input window which acts as a simple drawing board on which the user can draw his gestures. We can tweak several visual properties, like the colour and the width of the stroke used to draw gestures, and register various listeners to follow what the user is doing. The most commonly used listener is,
Gesture Overlay View. On Gesture Performed Listener, which fires whenever a user is done drawing a gesture .The input window contains a button for selecting the languages English ,Malayalam ,or digits, toggle buttons for uppercase ,lowercase, or caps lock and also for a button to go to the gesture builder to add new gestures.

3.1.4 Types of gestures

Include maximum possible gestures in the gesture library so that it will be able to recognize gestures drawn by people having different handwriting. One of the major applications is, it should be useful to the peoples having poor eyesight. Until recently, most touch screens provided few or no accessibility features, leaving them largely unusable by blind people .So during design we also wished to provide gestures in our application that will be appropriate for a blind user. Although blind people may use the same hardware as their sighted peers, it is possible that they will prefer to use different gestures, or that they will perform the same gestures differently than a sighted person. Sighted people perform gestures differently when they lack visual feedback, and it is reasonable to assume that a blind person may also perform gestures differently than a sighted person. This is shown in figure 3.1.3.

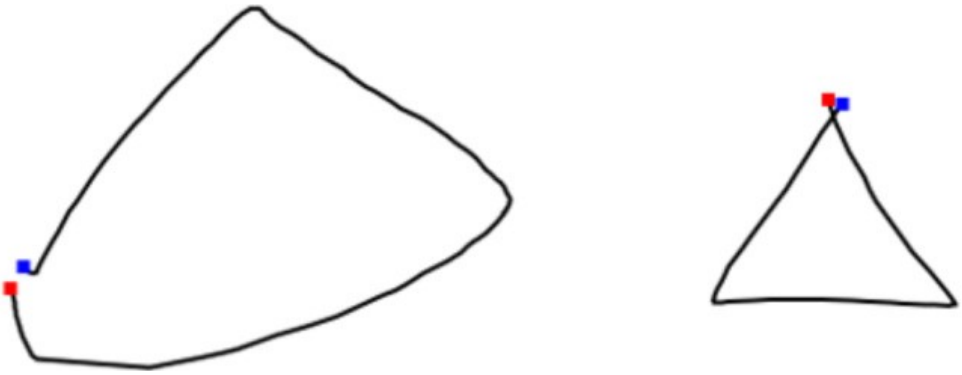


Fig 3.1.3 Two representative versions of a triangle gesture produced by a blind person (left) and a sighted person (right).

One specialty is that in the gesture library can include shorthand method to specify certain words that we are constantly using. This reduces the difficulty in typing messages. As we can include Malayalam gestures we will be able to send messages in our mother tongue. In this way inputting in Malayalam becomes easy.

3.2 FRAMEWORK OVERVIEW

The framework is build using Java language. The choice to use Java was to achieve a higher number of devices with the same framework, allowing its usage by any accelerated/touch device that can handle the acceleration data and/or touch data, and that has a Java virtual machine, like many devices from different manufacturers: Blackberries, Nokia, Sony Ericson, Motorola, Android, LG and Samsung. Also this framework can be used in a PC with a touch device like Microsoft surface or an accelerometer device like the Wiimote. Gesture recognition with touch/accelerated devices are represented by their patterns of the input data. The recognition is made by comparing the input pattern with the database pattern, checking if they match. In order to extract the pattern from the input data stream and comparing with the database pattern, this data must be prepared and analyzed. This work uses Hidden Markov Models in order to fulfill that need. In order to build a database, the framework needs to train and saves a set of gestures, which are also used by the mobile phone for recognition.

The proposed framework has the following steps:

- Segmentation: is used to determine when the gesture begin sand when it ends
- Filtering: is used in order to eliminate some parts of the data stream that do not contribute to the gesture;
- Quantitizer: is used to approximate the stream of input data into a smaller set of values;
- Model: is used to compute likelihood of analyzed gestures.
- Classifier: is used in order to identify the input gesture accordingly to the database.

It is possible to notice that the framework has two distinct modes: one for gesture training, i.e., build the database, and one for gesture recognition, i.e., comparing the input gesture with the database.

3.2.1 Segmentation

Segmentation is used mainly to automatically determinate the begin and end of the gesture. This identification in touch gestures is very easy since the begin of the gesture is when the user first touch the screen and it ends when the user release the finger/pen from the screen. In order to correct segment ate an accelerometer gesture, a definition of this kind of gesture is needed. This definition is made during the observation of the accelerated data and the movement of user during the recognition of different gestures. Normally, gestures begin with a fast acceleration, a continuous direction change during the gesture, and it ends with a stop of the movement.

3.2.2 Filtering

This pass is used in order to eliminate some parts of the data stream that do not contribute to the gesture. This work uses two kinds of filters in order to eliminate noises and data that are very similar. When a gesture is made, the data stream of the gesture may contain errors that if are sent to the HMM, some errors on the recognition may occur. In order to avoid such errors, a low pass filter is applied which is a very common filter used for noise remove. When the gesture is made, there are a lot of data on the data stream that does not contribute to the overall characteristic of the gesture. In order to diminished the data passed to the HMM, this work uses an idle threshold filter.

3.2.3 Quantizer

This step is only used for accelerated gestures. Because the accelerometer continues sends its data to the processor, the amount of data may be to big to for handling into a single HMM. Also, since the amount of RAM memory of mobile phones is limited, the use of a quantizer keeps less information in the gesture database. This work uses a k-mean algorithm which is a method of cluster analyzer. The algorithm aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.

3.2.4 Classifier

The classifier is used in order to identify gesture selecting the gesture with more like hood between the input gesture and the database gesture. This work uses a naive Bayes classifier also called simple Bayesian. Naive Bayes is simple probabilistic classifier based on the so-called Bayesian theorem and it is a well-known algorithm both in statistics and machine learning .There will be some probability value like prediction size, so that any gesture probability below that threshold is considered as a noise or poor match. Here if the prediction value is less than 1 it is taken as a poor match.

3.2.5 Gesture Recorder

The gesture recorder component listens to events that are emitted from the accelerometer sensors, so it steadily monitors the acceleration of the device. A gesture is detected if the absolute value of the acceleration exceeds a special threshold for a sufficient

period of time. Thus, it is assured that noises produced by unintended movements of the hand do not emit gestures.

3.2.6 Gesture Recognition Service

The Gesture Recognition Service is a service that can be subscribed by any android application. Gesture recognition is running in the background and reacts on new gestures recorded by the Gesture Recorder. Depended on the current mode, that can be set via the service's interface; new gestures are added to a training set or classified. The result is delivered to the application that is registered as a listener.

3.2.7 Gesture Training Application

The gesture training application is an application that uses the Gesture Recognition Service to manage and to extend training sets with new gestures. It uses the service interface to create new training sets, delete training sets, record new gestures, and delete existing gestures. Furthermore it supports two modes. The first mode is the training mode. When it is active, new recorded gestures are added to the actual selected training set. Second is the classification mode .To classify the new gestures.

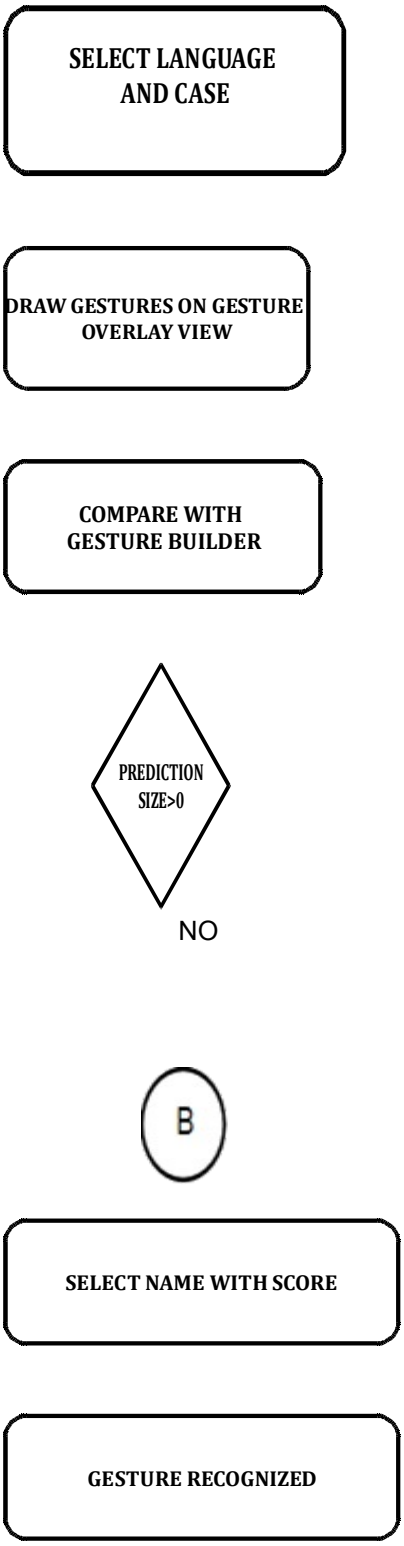
3.3 DATAFLOW DIAGRAM

A Data flow diagram is a graphical technique that depicts information flow and transforms that are applied as data move from input to output. The DFD is used to represent increasing information flow and functional details .A Level 0 DFD is called a fundamental system model or a context model,represents the entire software elements as a single bubble with input and output indicated by incoming and outgoing arrows respectively .Additional process and information flow are represented in the next level i.e. Level 1 DFD .Each of the process represented at level 1 as sub functions of overall system depicted in the context model.

3.3.1 Context Diagram

A System Context Diagram (SCD) in software engineering and systems engineering is a diagram that represents the actors outside a system that could interact with that system. This diagram is the highest level view of a system. It is similar to a Block diagram.

3.3.2 Level 1


SELECT LANGUAGE AND CASE
DRAW GESTURES ON GESTURE OVERLAY VIEW
COMPARE WITH GESTURE BUILDER
PREDICTION SIZE>0
NO
B
SELECT NAME WITH SCORE
GESTURE RECOGNIZED

FINISH

POOR MATCH

3.3.3 Gesture trainer package design model

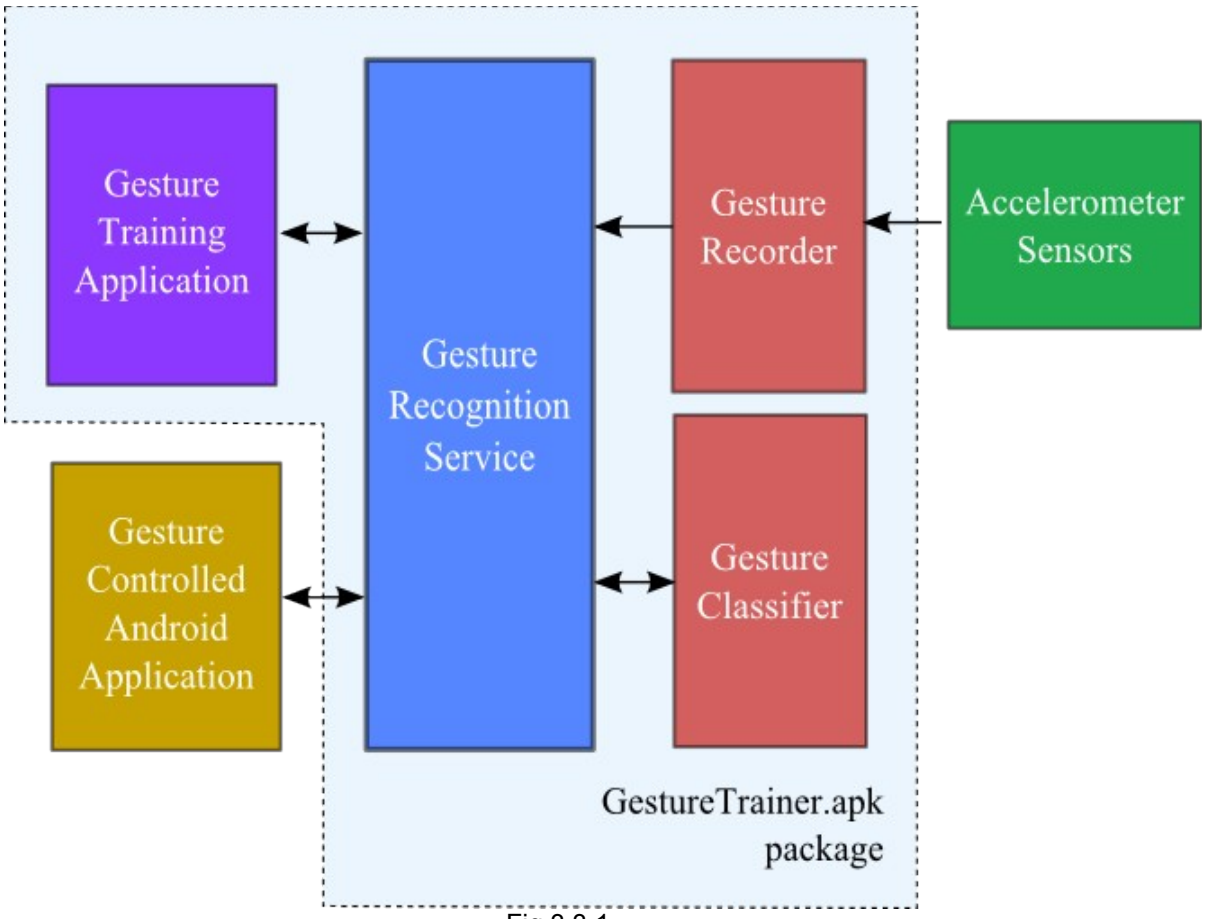


Fig 3.3.1

4. SYSTEM SPECIFICATION

4.1 HARDWARE SPECIFICATION

The selection of hardware is very important task related to the software development. The processor should be powerful to handle the entire operations .Intel Pentium system with the minimum specification as

System	:IBM Compatible Pc
Processor	:Intel Pentium i3 or later(64 bit)
Speed	:3.2 GHZ
Memory	:512 MB RAM
Hard Disk Drive	:80 GB
Monitor	:15'' Colour Monitor

4.2 SOFTWARE SPECIFICATION

- Development Platform :Windows 7,Ubuntu 11.04 or later
- JAVA AND XML
- ANDROID SDK 2.2 OR MORE
- ECLIPSE WITH ADT PLUGIN
- ANDROID VIRTUAL MACHINE EMULATOR

5. IMPLEMENTATION

Implementation of project on Gesture Keyboard consists of the following phases.

5.1 THE INSTALLATION PHASE

This phase consists of installing the required software in the system which is required for developing any application in android. This phase is of critical importance because we must choose the correct set of associated software along with its runtime environments. The set of software are chosen according to our machine environment (32 or 64 bit) and also the current operating system (Ubuntu or other variants of Linux/Windows). Special care is taken in choosing the associated plugins which is a prerequisite for fast and efficient development of android application.

The installation phase can be sub-divided into 4 parts

5.1.1 Installing Eclipse

The Eclipse Platform is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written mostly in Java. By means of various plug-ins, it can be used to develop applications in various programming languages including Ada, C, C++, COBOL, Erlang, Java, Perl, PHP, Python, R, Ruby (including Ruby on Rails framework), Scala, Clojure, Groovy and Scheme. It can also be used to develop packages for the software Mathematica. Development environments include the ADT plugin for

Android Application development ,Eclipse Java development tools (JDT) for Java, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others. The Eclipse SDK (which includes the Java development tools) is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules .The eclipse is downloaded from http://www.eclipse.org/downloads/

5.1.2 Installing Android SDK(Software Development Kit)

The Android SDK can be installed from

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

The SDK starter package is not a full development environment—it includes only the core SDK Tools, which you can use to download the rest of the SDK components (such as the latest Android platform).

Downloaded the Windows installer (.exe file),if we wish it to build in windows platform and run it now and it will check whether the proper Java SE Development Kit (JDK) is installed (installing it, if necessary), then install the SDK Tools into a default location (which you can modify).Make a note of the name and location of the SDK directory on your system—you will need to refer to the SDK directory later, when setting up the ADT plugin and when using the SDK tools from the command line.

5.1.3 Installing the ADT Plugin for Eclipse

Android offers a custom plugin for the Eclipse IDE, called Android Development Tools (ADT), that is designed to give you a powerful, integrated environment in which to build Android applications. It extends the capabilities of Eclipse and helps us to quickly set up new Android projects, create an application UI, debug your applications using the Android SDK tools, and even export signed (or unsigned) APKs in order to distribute your application. In general, developing in Eclipse with ADT is a highly recommended approach and is the fastest way to get started with Android.

5.1.4 Setting AVD(Android Virtual Device)

The last step in setting up your SDK is using the Android SDK and AVD Manager (a tool included in the SDK starter package) to download essential SDK components into your development environment.

The SDK uses a modular structure that separates the major parts of the SDK—Android platform versions, add-ons, tools, samples, and documentation—into a set of separately installable components. The SDK starter package, which you've already downloaded, includes only a single component: the latest version of the SDK Tools. To develop an Android application, you also need to download at least one Android platform and the associated platform tools. You can add other components and platforms as well, which is highly recommended.

We can launch the Android SDK and AVD Manager in one of the following ways:

•From within Eclipse, select Window > Android SDK and AVD Manager.

•On Windows, double-click the SDK Manager.exe file at the root of the Android SDK directory.

•On Mac or Linux, open a terminal and navigate to the tools/ directory in the Android SDK, then execute:

5.2 SETTING UP PHASE

To download components, use the graphical UI of the Android SDK and AVD Manager to browse the SDK repository and select new or updated components. The Android SDK and AVD Manager install the selected components in your SDK environment (Figure 5.2.1).

.

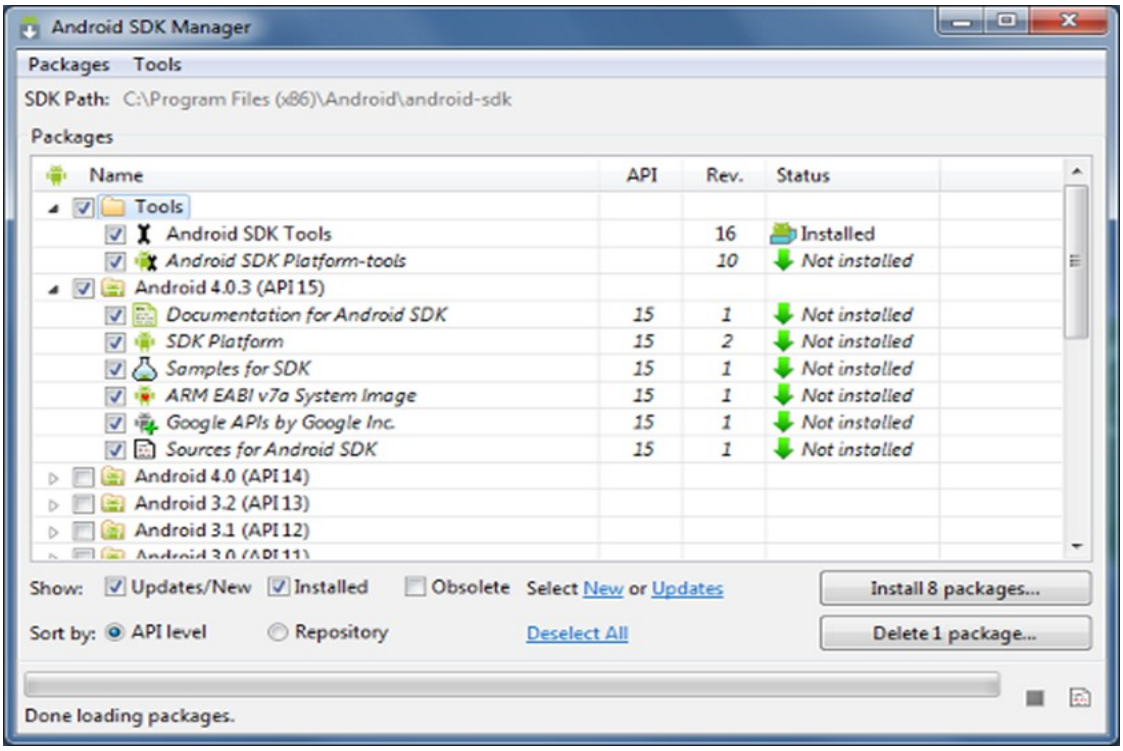


Fig 5.2.1 The Android SDK and AVD Manager's Available Packages panel, which shows the SDK components that are available for you to download into your environment.

5.3 DESIGNING THE FRONTEND

The front end of the application in solely prepared in XML(Xtended Markup Language). XML is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications, all gratis open standards.

The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services. Many application programming interfaces (APIs) have been developed for software developers to use to process XML data, and several schema systems exist to aid in the definition of XML-based languages.

In an Android application, the user interface is built using View and ViewGroup objects. There are many types of views and view groups, each of which is a descendant of the View class .View objects are the basic units of user interface expression on the Android platform. The View class serves as the base for subclasses called "widgets," which offer fully implemented UI objects, like text fields and buttons. The ViewGroup class serves as the base for subclasses called "layouts," which offer different kinds of layout architecture, like linear, tabular and relative.

A View object is a data structure whose properties store the layout parameters and content for a specific rectangular area of the screen. A View object handles its own measurement, layout, drawing, focus change, scrolling, and key/gesture interactions for the rectangular area of the screen in which it resides. As an object in the user interface, a View is also a point of interaction for the user and the receiver of the interaction events.

5.3.1 View hierarchy

On the Android platform, you define an Activity's UI using a hierarchy of View and ViewGroup nodes, as shown in the diagram below. Shown in figure 5.3.1 .This hierarchy tree can be as simple or complex as you need it to be, and you can build it up using Android's set of predefined widgets and layouts, or with custom Views that you create yourself.

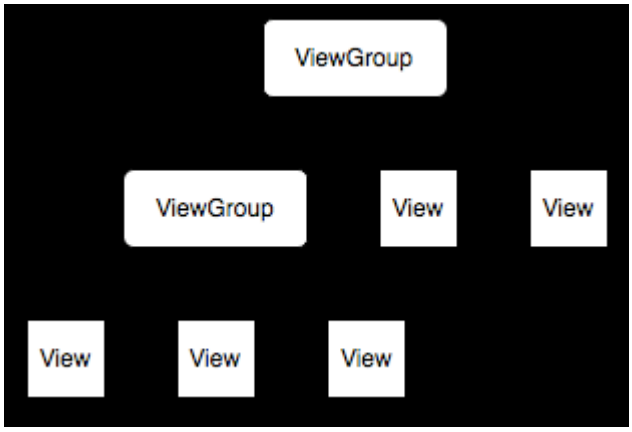


Fig 5.3.1

In order to attach the view hierarchy tree to the screen for rendering, your Activity must call the setContentView() method and pass a reference to the root node object. The Android system receives this reference and uses it to invalidate, measure, and draw the tree. The root node of the hierarchy requests that its child nodes draw themselves — in turn, each view group node is responsible for calling upon each of its own child views to draw themselves. The children may request a size and location within the parent, but the parent object has the final decision on where how big each child can be. Android parses the elements of your layout in-order (from the top of the hierarchy tree), instantiating the Views and adding them to their parent(s). Because these are drawn in-order, if there are elements that overlap positions, the last one to be drawn will lie on top of others previously drawn to that space.

5.3.2 Layout

The most common way to define your layout and express the view hierarchy is with an XML layout file. XML offers a human-readable structure for the layout, much like HTML. Each element in XML is either a View or ViewGroup object (or descendant thereof). View objects are leaves in the tree, ViewGroup objects are branches in the tree (see the View Hierarchy figure above).

The name of an XML element is respective to the Java class that it represents. So a <TextView> element creates a TextView in your UI, and a <LinearLayout> element creates a LinearLayout view group. When you load a layout resource, the Android system initializes these run-time objects, corresponding to the elements in your layout.

5.3.3 Widgets

A widget is a View object that serves as an interface for interaction with the user. Android provides a set of fully implemented widgets, like buttons, checkboxes, and text-entry fields, so you can quickly build your UI. Some widgets provided by Android are more complex, like a date picker, a clock, and zoom controls. But you're not limited to the kinds of widgets provided by the Android platform. If you'd like to do something more customized and create your own actionable elements, you can, by defining your own View object or by extending and combining existing widgets.

5.3.4 Menus

Application menus are another important part of an application's UI. Menus offers a reliable interface that reveals application functions and settings. The most common application menu is revealed by pressing the *Menu* button on the device. However, you can also add Context Menus, which may be revealed when the user presses and holds down on an item.

Menus are also structured using a View hierarchy, but you don't define this structure yourself. Instead, you define the onCreateOptionsMenu() or onCreateContextMenu() callback methods for your Activity and declare the items that you want to include in your menu. At the appropriate time, Android will automatically create the necessary View hierarchy for the menu and draw each of your menu items in it.

Menus also handle their own events, so there's no need to register event listeners on the items in your menu. When an item in your menu is selected, the onOptionsItemSelected() or onContextItemSelected() method will be called by the framework. And just like your application layout, you have the option to declare the items for you menu in an XML file.

5.4 MAPPING THE FRONT END TO THE CODE

We can externalize application resources such as images and strings from our code, so that we can maintain them independently. Developers should also provide alternative resources for specific device configurations, by grouping them in specially-named resource directories. At runtime, Android uses the appropriate resource based on the current configuration. For example, we might want to provide a different UI layout depending on the screen size or different strings depending on the language setting. Once we externalize the application resources, you can access them using resource IDs that are generated in the project's R class.

5.4.1 Grouping Resource Types

You have to place each type of resource in a specific subdirectory of the project's res/ directory. For example, here's the file hierarchy for a simple project:

```
MyProject/
  src/
    MyActivity.java
  res/
    drawable/
      icon.png
    layout/
      main.xml
      info.xml
    values/
```

strings.xml

As you can see in this example, the res/ directory contains all the resources (in subdirectories): an image resource, two layout resources, and a string resource file.

5.4.2 Accessing Resources

Once we provide a resource in our application, we can apply it by referencing its resource ID. All resource IDs are defined in our project's R class, which the apt tool automatically generates. When the application is compiled, apt generates the R class, which contains resource IDs for all the resources in the res/ directory. For each type of resource, there is an R subclass (for example, R.drawable for all drawable resources) and for each resource of that type, there is a static integer (for example, R.drawable.icon). This integer is the resource ID that you can use to retrieve your resource. Although the R class is where resource IDs are specified, you should never need to look there to discover a resource ID. A resource ID is always composed of:

•The resource type: Each resource is grouped into a "type," such as string, drawable, and layout. For more about the different types.

•The resource name, which is either: the filename, excluding the extension; or the value in the XML android: name attribute, if the resource is a simple value (such as a string).

There are two ways to access a resource:

•In code: Using a static integer from a sub-class of your R class

•In XML: Using a special XML syntax that also corresponds to the resource ID defined in our R class

Here is the syntax to reference a resource in an XML resource:

@[<package_name>:]<resource_type>/<resource_name>

•<package_name> is the name of the package in which the resource is located (not required when referencing resources from the same package)

•<resource_type> is the R subclass for the resource type

•<resource_name> is either the resource filename without the extension or the android:name attribute value in the XML element (for simple values).

5.5 CODING

The coding is done in Eclipse IDE .IDE provides a simplified and integrated development environment to code .The code is basically developed in java .Use specialized packages and inbuilt classes to inherit additional capabilities needed to code an android application .We also uses various API(Application Programming Interface) to do device specific functionalities.

5.6 DEBUGGING

5.6.1 Debugging from Eclipse with ADT

If you are developing in Eclipse with the ADT plugin, can use the built-in Java Debugger, along with DDMS, to debug the application. To access the debugger and DDMS, Eclipse displays the debugger and DDMS features as perspectives, which are customized Eclipse views that display certain tabs and windows depending on the perspective that we are in. Eclipse also takes care of starting the ADB host daemon for us, so you do not have to run this manually.

5.6.2 The Debug Perspective in Eclipse

The Debug Perspective in Eclipse gives us to access the following tabs:

•Debug - Displays previously and currently debugged Android applications and its currently running threads

•Variables - When breakpoints are set, displays variable values during code execution

•Breakpoints - Displays a list of the set breakpoints in our application code

•LogCat - Allows us to view system log messages in real time. The LogCat tab is also available in the DDMS perspective.

We can access the Debug Perspective by clicking Window > Open Perspective > Debug.

5.6.3 The DDMS Perspective

The DDMS Perspective in Eclipse lets you access all of the features of DDMS from within the Eclipse IDE. The following sections of DDMS are available:

•Devices - Shows the list of devices and AVDs that are connected to ADB.

•Emulator Control - To carry out device functions.

•LogCat - To view system log messages in real time.

•Threads - Shows currently running threads within a VM.

•Heap - Shows heap usage for a VM.

•Allocation Tracker - Shows the memory allocation of objects.

•File Explorer - To explore the device's file system.

To access the DDMS perspective, go to Window > Open Perspective > DDMS. If DDMS does not appear, go to Window > Open Perspective > Other ... and select DDMS from the Open Perspective window that appears.

5.6.4 Interaction of DDMS with debugger

On Android, every application runs in its own process, each of which runs in its own virtual machine (VM). Each VM exposes a unique port that a debugger can attach to. When DDMS starts, it connects to adb. When a device is connected, a VM monitoring service is created between adb and DDMS, which notifies DDMS when a VM on the device is started or terminated. Once a VM is running, DDMS retrieves the the VM's process ID (pid), via adb, and opens a connection to the VM's debugger, through the adb daemon (adbd) on the device. DDMS can now talk to the VM using a custom wire protocol.

DDMS assigns a debugging port to each VM on the device. Typically, DDMS assigns port 8600 for the first debuggable VM, the next on 8601, and so on. When a

debugger connects to one of these ports, all traffic is forwarded to the debugger from the associated VM. You can only attach a single debugger to a single port, but DDMS can handle multiple, attached debuggers. By default, DDMS also listens on another debugging port, the DDMS "base port" (8700, by default). The base port is a port forwarder, which can accept VM traffic from any debugging port and forward it to the debugger on port 8700. This allows you to attach one debugger to port 8700, and debug all the VMs on a device. The traffic that is forwarded is determined by the currently selected process in the DDMS Devices view.

5.7 INSTALLING MALAYALAM FONT

After installing this application to phone, in order for proper recognition of Malayalam, Malayalam font should be installed to the phone .Malayalam font is available in android market .Malayalam font can be incorporated by installing the Malayalam font in the phone.

6. TESTING

Testing is a dynamic technique of verification and validation, which ensures that the software meets the expectations of the user .It involves executing an implementation of the software with test data. Test data is the set of data that the system will process as a normal input. System testing verifies that whole set of programs hang together .It makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved .Inadequate/non-testing leads to errors. The whole system has been tested and found to be working.

System testing follows the logical conclusion that is all the part of the system are tested and found to be working properly under all kinds of situations, then the system is achieving its goal of processing the data perfectly according to the user rules and requirements.

The following are the attributes of the good test:

- A good test has high probability of finding an error
- A good test is not redundant
- A good test should be “Best of Breed”
- A good test should be neither too simple nor too complex

Types of Testing

- Unit testing
- Integration testing
- Validation testing
- Output testing
- User acceptance testing

Unit testing

The first level of testing is called an unit testing. Here all modules are tested separately. Checked whether the layouts prepared are correct according to the needs and

then avoided all the errors generated in the Gesture Keyboard file. Each module is found to be working satisfactorily as regarded to be expected output from the module

Integration testing

The second level of testing includes integration testing .the integration testing is to confirm that the individual module showing results will also show perfect result when run as a whole. This testing activity can be considered as the design and emphasizes on testing modules interaction. Here the layout is mapped to the source code and compiled as a whole.

Validation testing

The next level of testing is validation testing .here the entire software is tested. The reference document for this process is the requirement and goal is to see if the software meets its requirements.

Output testing

The output of the software should be acceptable to the system user. The output requirement is defined during the system analysis. Testing of the software system is done against the output and the following errors are found.

- Interface errors
- Performance errors
- Incompatibility of Multi language keyboard
- Logical errors

6.1 TESTING FROM ECLISE WITH ADT

ADT provides several features that help us to set up and manage the testing environment effectively:

- When we create a project, it automatically inserts the necessary <instrumentation> element in the test package's manifest file.
- Import the classes of the application under test, so that tests can inspect them.
- Create run configurations for package and include in them flags that are passed to the Android testing framework.
- Run test package without leaving Eclipse. ADT builds both the application under test and the test package automatically, installs them if necessary to our device or emulator, runs the test package, and displays the results in a separate window in Eclipse.

6.2 RUNNING TESTS

6.2.1 Running tests from the command line

To run tests created in Eclipse with ADT with command-line tools, you must first install additional files into the test project using the android tool's "create test-project" option. When run a test package in Eclipse with ADT, the output appears in the Eclipse JUnit view. You Can run the entire test package or one test case class. To do run tests, Eclipse runs the adb command for running a test package, and displays the output, so

there is no difference between running tests inside Eclipse and running them from the command line.

As with any other package, to run a test package in Eclipse with ADT you must either attach a device to our computer or use the Android emulator. You must have an Android Virtual Device (AVD) that uses the same target as the test package.

To run a test in Eclipse, we have two choices:

- Run a test just as we run an application, by selecting Run As... > Android JUnit Test from the project's context menu or from the main menu's Run item.
- Create an Eclipse run configuration for our test project. This is useful if we want multiple test suites, each consisting of selected tests from the project. To run a test suite, we run the test configuration.

7. CONCLUSION

Android offers a fresh take on the way mobile application interact with users, along with the technical underpinnings to make it possible. It combines the ubiquity of cell phones, the excitement of open source software ,and the corporate backing of Google and other Open Handset Alliance members. Touch screens are a great way to interact with applications on mobile devices. Here by using our technology we can recognise Malayalam gestures which are not yet developed. The Android framework makes it's easy to recognize simple action. As touch screens are now one of the most common ways of interacting with computers, they can do so effectively and efficiently in the recognition of human gestures. This work provides new information about how Malayalam gestures can be incorporated in a gesture recognition system even though it contains huge amount of alphabets and also symbols. We believe that this work will bring us closer to the creation of robust and usable touch screen interfaces that work equally well for all classes of people.

8. FUTURE SCOPE

Gestures can be used to perform particular actions. Gesture application can be integrated with other applications- like a flip to move the next window ,a swirl to delete files .Here for getting a song or a picture just a simple touch is needed .Future surgeons may use Robotic nurse ,'gesture recognition' .Here we can control the actions of a robot by using wired glouses. Whatever operations that we are doing with that wired glouses will captured by the robots hands and it will do according to that. For example if we perform the action of cutting something by using the wired glouses then the robot also do the same action .Here not just a simple touching is needed. Another important future application is MSIS-Multi semantic interpretation system .It is the combination of all gestures(3D) and sounds. If this technology comes into the market it will create revolutionary changes in all the fields.