# ANDROID LEARNING APPLICATION FOR KIDS (e-SLATE)

PROJECT REPORT PRESENTED BY

**Ameya Parab**
**Atikur Rahman Khan**
**Mohit Panjwani**
**Rakesh Pawar**

OF

INFORMATION TECHNOLOGY

UNDER THE GUIDANCE OF

**Prof. Asha Bharambe**

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY
HashuAdvani Memorial Complex, Collector's Colony,
R C Marg, Chembur, Mumbai – 400074.

# VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

Hashu Advani Memorial Complex, Collector's Colony,
R C Marg, Chembur, Mumbai – 400074.

## CERTIFICATE OF APPROVAL

This is to certify that the project entitled as **'Android Learning Application for Kids'** has been approved.

*Submitted by*

**Ameya Parab**
**Atikur Rahman Khan**
**Mohit Panjwani**
**Rakesh Pawar**

**DATE:_____**

_____

Mrs. Asha Bharambe
(Project Guide)

_____ _____

Mrs. M. Vijayalakshmi
(Head of Department)

Internal Examiner

_____ _____

Dr. (Mrs.) J.M. Nair
(Principal)

External Examiner

(In partial fulfilment for the **Degree of Bachelor of Engineering**)

*In*

**Department of Information Technology**

University Of Mumbai

Academic Year: 2012-13

# VIVEKANAND EDUCATION SOCIETY'S
# INSTITUTE OF TECHNOLOGY

Hashu Advani Memorial Complex, Collector's Colony,
R C Marg, Chembur, Mumbai – 400074

## *CERTIFICATE OF APPROVAL OF PROJECT WORK*

This is to Certify that Mr. / Miss. _____

_____ has

satisfactorily carried out the project work entitled _____

_____ in partial

fulfillment of the B.E. Degree in _____

_____ of the University

of Mumbai, Maharashtra state during year 20 - 20

**PRINCIPAL** **HEAD OF DEPT.**

**PROJECT GUIDE** **EXAMINER**

# ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude and deep regards to our project guide Mrs. Asha Bharambe, Assistant Professor of Department of Information Technology at Vivekananda Education Society's Institute of Technology, for her exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by her time to time shall carry us a long way in the journey of life on which we are about to embark.

We would also like to thank the college for extending its support and providing the entire infrastructure our project demanded. We would also like to express our gratitude to the people from Information Technology branch for their help and support from the inception to the final execution of the project.

We also take this opportunity to express a deep sense of gratitude to our classmate Shamalee Thakur for her voice which we have used throughout the application.

A special thanks goes to IDC (Industrial Design Center) department of IIT Bombay, who helped us in getting started with the designing of layout of the application.

Lastly we thank almighty, our parents, brothers, sisters and friends for their constant encouragement without which this assignment would not be possible.

# CONTENTS

# CHAPTER-1

# INTRODUCTION

## 1.1 Introduction

One and a half billion people, all over the world, are walking around with powerful computers in their pockets and purses. The fact is they often do not realize it, because they call them something else viz Cell Phones/Mobiles. Today's high-end cell phones have the computing power of a mid-1990s personal computer (PC)—while consuming only one one-hundredth of the energy. Even the simplest, voice-only phones have more complex and powerful chips than the 1969 on-board computer that landed a spaceship on the moon.

Mobile learning, through the use of mobile technology, will allow citizens of the world to access learning materials and information from anywhere and at any time. Learners will not have to wait for a certain time to learn or go to a certain place to learn. With mobile learning, learners will be empowered since they can learn whenever and wherever they want.

There are thousands of application in the market for popular mobile operating systems mainly iOS, Android, Windows and Blackberry. Chapter 2[2.2] and Chapter 3 in this report shows analysis of various operating system of mobiles based on its existing application. This report also gives the Content [Chapter 4] and Implementation [Chapter 5] of e-Slate application with the layout.

## 1.2 Problem Statement

A small kid is more attracted towards a mobile phone rather than a book. So if we develop a application through which we help the kids of age group 3-5 to learn the Alphabets, Numbers, Comparisons, Colors and objects it would be beneficial for the child for learning through the use of smart technology with fun .Also the children will get use to the new evolving technology and would help for the fundamental development of a child. This software will also allow the child to play games on the sections which kid has learnt.

### 1.3 Project Objective

The objective of this project to develop and implement a new android application which will help teachers as well as parents for teaching kids. The application will help child learn:

- Alphabets: Introduction to alphabets, Phonics introduction, Drawing, Alphabet song.
- Numbers: Introduction to numbers, Phonics introduction, Drawing, Numbers song, addition, subtraction.
- General Awareness: Colors, Animals, Vegetables, Fruits.
- Games: Alphabetic games, Numeric games.

# CHAPTER-2

# LITERATURE SURVEY

## 2.1 Learning Theories

Learning theories are conceptual frameworks that describe how information is absorbed, processed, and retained during learning. Learning brings together cognitive, emotional, and environmental influences and experiences for acquiring, enhancing, or making changes in one's knowledge, skills, values, and world views.

There are three main categories of learning theory: behaviorism, cognitivism, and constructivism. Behaviorism focuses only on the objectively observable aspects of learning. Cognitive theories look beyond behaviour to explain brain-based learning. And constructivism views learning as a process in which the learner actively constructs or builds new ideas or concepts.

### Behaviourism

Behaviorism (or behaviourism), also called the learning perspective (where any physical action is a behavior), is a philosophy of psychology based on the proposition that all things that organisms do—including acting, thinking, and feeling—can and should be regarded as behaviors, and that psychological disorders are best treated by altering behavior patterns or modifying the environment. According to behaviorism, individuals' response to different environmental stimuli shapes our behaviors. Behaviorists believe behavior can be studied in a methodical and recognizable manner with no consideration of internal mental states. Thus, all behavior can be clarified without the need to reflect on psychological mental states.

Three basic assumptions are held to be true. First, learning is manifested by a change in behaviour. Second, the environment shapes behaviour. And third, the principles of contiguity (how close in time two events must be for a bond to be formed) and reinforcement (any means of increasing the likelihood that an event will be repeated) are central to explaining the learning process. For behaviorism, learning is the acquisition of new behaviour through conditioning.

There are two types of possible conditioning:

1) Classical conditioning, where the behaviour becomes a reflex response to stimulus as in the case of Pavlov's Dogs. Pavlov was interested in studying reflexes, when he saw that the dogs drooled without the proper stimulus. Although no food was in sight, their saliva still dribbled. It turned out that the dogs were reacting to lab coats. Every time the dogs were served food, the person who served the food was wearing a lab coat. Therefore, the dogs reacted as if food was on its way whenever they saw a lab coat. In a series of experiments, Pavlov then tried to Figure out how these phenomena were linked. For example, he struck a bell when the dogs were fed. If the bell was sounded in close association with their meal, the

dogs learned to associate the sound of the bell with food. After a while, at the mere sound of the bell, they responded by drooling.

2) Operant conditioning where there is reinforcement of the behaviour by a reward or a punishment. The theory of operant conditioning was developed by B.F. Skinner and is known as Radical Behaviorism. The word 'operant' refers to the way in which behaviour 'operates on the environment'. Briefly, a behaviour may result either in reinforcement, which increases the likelihood of the behaviour recurring, or punishment, which decreases the likelihood of the behaviour recurring. It is important to note that, a punishment is not considered to be applicable if it does not result in the reduction of the behaviour, and so the terms punishment and reinforcement are determined as a result of the actions. Within this framework, behaviourists are particularly interested in measurable changes in behaviour. In operant conditioning we learn to associate a response (our behaviour) and its consequence and thus to repeat acts followed by good results and avoid acts followed by bad results.

Since behaviourists view the learning process as a change in behaviour, educators arrange the environment to elicit desired responses through such devices as behavioural objectives, competency -based education, and skill development and training.

**Cognitivism**

In psychology, cognitivism is a theoretical framework for understanding the mind that gained credence in the 1950s. The movement was a response to behaviorism, which cognitivists said neglected to explain cognition. Cognitive psychology derived its name from the Latin cognoscere, referring to knowing and information, thus cognitive psychology is an information processing psychology derived in part from earlier traditions of the investigation of thought and problem solving. Behaviorists acknowledged the existence of thinking, but identified it as a behavior. Cognitivists argued that the way people think impacts their behavior and therefore cannot be a behavior in and of itself. Cognitivists later argued that thinking is so essential to psychology that the study of thinking should become its own field.

Cognitivism Grew in response to Behaviorism. Knowledge is stored cognitively as symbols. Learning is the process of connecting symbols in a meaningful & memorable way. Studies focused on the mental processes that facilitate symbol connection.

There are two concepts

1) Discovery Learning

2) Meaningful Verbal Learning

Discovery Learning

1. Anybody can learn anything at any age, provided it is stated in terms they can understand.

2. Confront the learner with problems and help them find solutions. Do not present sequenced materials.

Meaningful Verbal Learning

1. New material is presented in a systematic way, and is connected to existing cognitive structures in a meaningful way.
2. When learners have difficulty with new material, go back to the concrete anchors (Advance Organizers). Provide a Discovery approach, and they'll learn.

**Constructivism**

Constructivism is a revolution in educational psychology. Built on the work of Piaget and Bruner, constructivism emphasizes the importance of active involvement of learners in constructing knowledge for themselves...Constructivism emphasizes top-down processing: begin with complex problems and teach basic skills while solving these problems. Constructivism explains why students do not learn deeply by listening to a teacher, or reading from a textbook. Learning sciences research is revealing the deeper underlying basis of how knowledge construction works. To design effective environments, one needs a very good understanding of what children know when they come to the classroom. This requires sophisticated research into children's cognitive development, and the learning sciences draws heavily on psychological studies of cognitive development (e.g., Siegler, 1998). The learning theories of John Dewey, Marie Montessori, and David Kolb serve as the foundation of constructivist learning theory. Constructivism views learning as a process in which the learner actively constructs or builds new ideas or concepts based upon current and past knowledge or experience. In other words, "learning involves constructing one's own knowledge from one's own experiences." Constructivist learning, therefore, is a very personal endeavor, whereby internalized concepts, rules, and general principles may consequently be applied in a practical real-world context. Constructivism itself has many variations, such as Active learning, discovery learning, and knowledge building. Regardless of the variety, constructivism promotes a student's free exploration within a given framework or structure. The teacher acts as a facilitator who encourages students to discover principles for themselves and to construct knowledge by working to solve realistic problems. Aspects of constructivism can be found in self-directed learning, transformational learning, and experiential learning.

**Some learning theories**

**Piaget**

Jean Piaget proposed that children's thinking does not develop entirely smoothly: instead, there are certain points at which it "takes off" and moves into completely new areas

and capabilities. He saw these transitions as taking place at about 18 months, 7 years and 11 or 12 years. This has been taken to mean that before these ages children are not capable (no matter how bright) of understanding things in certain ways, and has been used as the basis for scheduling the school curriculum. Whether or not *should* be the case is a different matter.

**Piaget's Key Ideas**

| | |
|---|---|
| Adaptation | What it says: adapting to the world through assimilation and accommodation |
| Assimilation | The process by which a person takes material into their mind from the environment, which may mean changing the evidence of their senses to make it fit. |
| Accommodation | The difference made to one's mind or concepts by the process of assimilation.<br>Note that assimilation and accommodation go together: you can't have one without the other. |
| Classification | The ability to group objects together on the basis of common features. |
| Class Inclusion | The understanding, more advanced than simple classification, that some classes or sets of objects are also sub-sets of a larger class. (E.g. there is a class of objects called dogs. There is also a class called animals. But all dogs are also animals, so the class of animals includes that of dogs) |
| Conservation | The realization that objects or sets of objects stay the same even when they are changed about or made to look different. |
| Decentration | The ability to move away from one system of classification to another one as appropriate. |
| Egocentrism | The belief that you are the center of the universe and everything revolves around you: the corresponding inability to see the world as someone else does and adapt to it. Not moral "selfishness", just an early stage of psychological development. |
| Operation | The process of working something out in your head. Young children (in the sensorimotor and pre-operational stages) have to act, and try things out in the real world, to work things out (like count on fingers): older children and adults can do more in their heads. |
| Schema (or | The representation in the mind of a set of perceptions, ideas, and/or |

| | |
|---|---|
| scheme) | actions, which go together. |
| Stage | A period in a child's development in which he or she is capable of understanding some things but not others |

Stages of Cognitive Development

| Stage | Characterized by |
|---|---|
| Sensori-motor (Birth-2 yrs) | Differentiates self from objects<br><br>Recognises self as agent of action and begins to act intentionally: e.g. pulls a string to set mobile in motion or shakes a rattle to make a noise<br><br>Achieves object permanence: realises that things continue to exist even when no longer present to the sense (pace Bishop Berkeley) |
| Pre-operational (2-7 years) | Learns to use language and to represent objects by images and words<br><br>Thinking is still egocentric: has difficulty taking the viewpoint of others<br><br>Classifies objects by a single feature: e.g. groups together all the red blocks regardless of shape or all the square blocks regardless of colour |
| Concrete operational (7-11 years) | Can think logically about objects and events<br><br>Achieves conservation of number (age 6), mass (age 7), and weight (age 9)<br><br>Classifies objects according to several features and can order them in series along a single dimension such as size. |
| Formal operational (11 years and up) | Can think logically about abstract propositions and test hypotheses systemtically<br><br>Becomes concerned with the hypothetical, the future, and ideological problems |

The accumulating evidence is that this scheme is too rigid: many children manage concrete operations earlier than he thought, and some people never attain formal operations (or at least are not called upon to use them).

Piaget's approach is central to the school of cognitive theory known as "cognitive constructivism".

**Vygotsky**

Vygotsky's Social Development Theory is the work of Russian psychologist Lev Vygotsky (1896-1934), who lived during Russian Revolution. Vygotsky's work was largely unknown to the West until it was published in 1962.
Vygotsky's theory is one of the foundations of constructivism. It asserts three major themes:

Major themes:

1. Social interaction plays a fundamental role in the process of cognitive development. In contrast to Jean Piaget's understanding of child development (in which development necessarily precedes learning), Vygotsky felt social learning precedes development. He states: "Every function in the child's cultural development appears twice: first, on the social level, and later, on the individual level; first, between people (interpsychological) and then inside the child (intrapsychological)." (Vygotsky, 1978).
2. The More Knowledgeable Other (MKO). The MKO refers to anyone who has a better understanding or a higher ability level than the learner, with respect to a particular task, process, or concept. The MKO is normally thought of as being a teacher, coach, or older adult, but the MKO could also be peers, a younger person, or even computers.
3. The Zone of Proximal Development (ZPD). The ZPD is the distance between a student's ability to perform a task under adult guidance and/or with peer collaboration and the student's ability solving the problem independently. According to Vygotsky, learning occurred in this zone.

Vygotsky focused on the connections between people and the sociocultural context in which they act and interact in shared experiences (Crawford, 1996). According to Vygotsky, humans use tools that develop from a culture, such as speech and writing, to mediate their social environments. Initially children develop these tools to serve solely as social functions, ways to communicate needs. Vygotsky believed that the internalization of these tools led to higher thinking skills.

Applications of the Vygotsky's Social Development Theory:

Many schools have traditionally held a transmissionist or instructionist model in which a teacher or lecturer 'transmits' information to students. In contrast, Vygotsky's theory promotes learning contexts in which students play an active role in learning. Roles of the teacher and student are therefore shifted, as a teacher should collaborate with his or her students in order to help facilitate meaning construction in students. Learning therefore becomes a reciprocal experience for the students and teacher.

Learning Theories Adopted by Our Application:

-Cognitivism

Application consist of listening and drawing functionality which follows cognitive learning. In Listening cognitivism is implemented by presenting alphabets and numbers as symbols with their sounds. Alphabets and numbers are presented in meaningful and memorable way. In drawing kids are allowed to practice over alphabets and numbers which will help kids to store knowledge as symbols.

-Constructivism

Application contains alphabet and number songs which will allow kids to construct order of alphabets and numbers. Application also contains addition and subtraction functions which will help kids to construct addition and subtraction of any numbers.

## 2.2 Platform Analysis

Apple's sizable lead in the world of mobile apps is under attack from Google. Vision Mobile, an analysis and advisory firm, surveyed 401 mobile app developers, and found that developers have more Android experience than Apple iOS.

Visible says Android has passed Apple with developers because Google's developers kit costs less, and because Google has done a good job marketing its open source model. Further, Android phones are selling well now, so there's plenty of customers.

Other than Apple and Android, the rest of the mobile platforms are being left behind. Vision Mobile says, "other mobile platform are lagging far behind Android and Iphone.

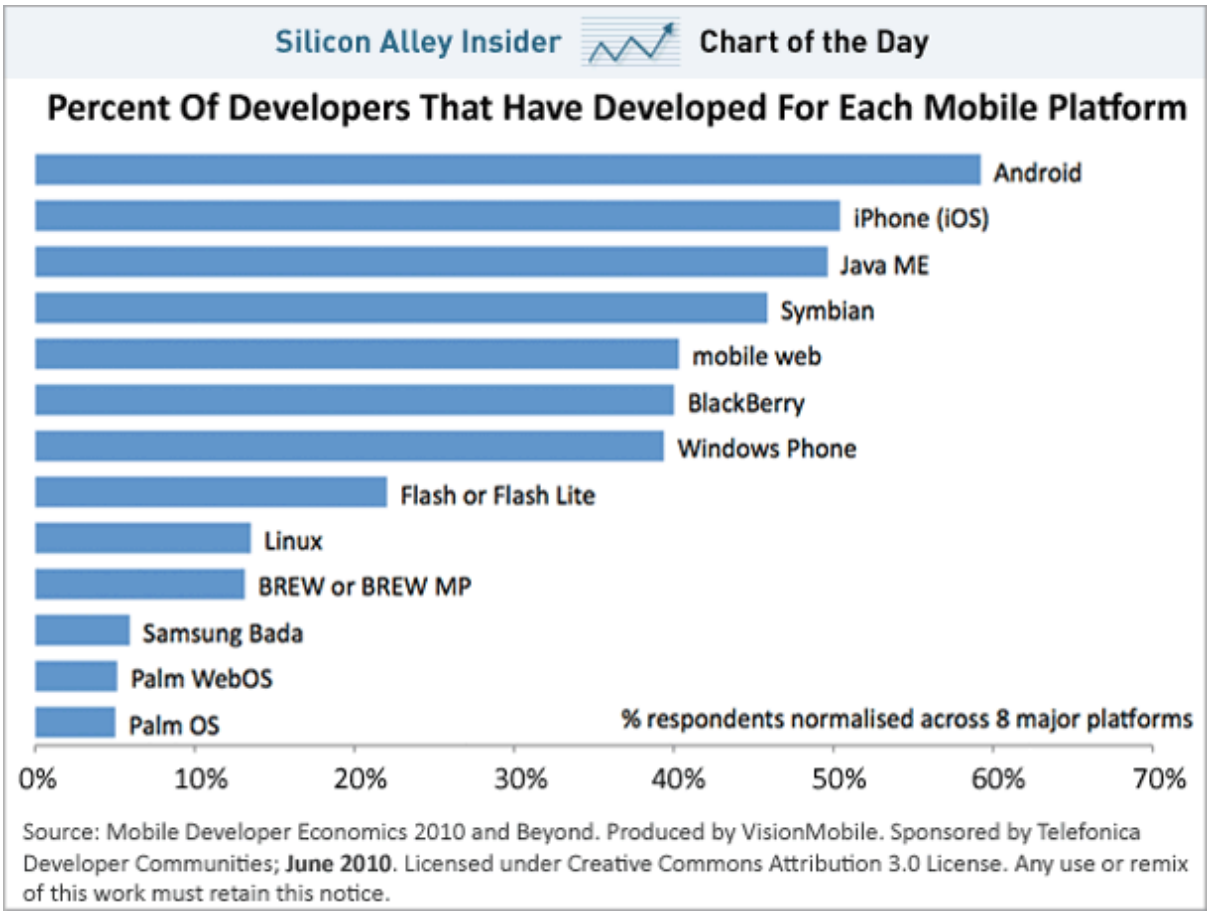



Figure 2.1: Developments per Mobile Platform

**Operating System Highlights**

**Android** finished the quarter as the overall leader among the mobile operating systems, accounting for more than half of all smartphone shipments. In addition, Android boasted the longest list of smartphone vendor partners. Samsung was the largest contributor to Android's success, accounting for 45.4% of all Android-based smartphone shipments. But beyond Samsung was a mix of companies retrenching themselves or slowly growing their volumes.

**iOS** recorded strong year-over-year growth with sustained demand for the iPhone 4S following the holiday quarter and the addition of numerous mobile operators offering the iPhone for the first time. Although end-user demand remains high, the iPhone's popularity brings additional operational pressures for mobile operators through subsidy and data revenue sharing policies.

**Symbian** posted the largest year-over-year decline, a result driven by Nokia's transition to Windows Phone. But even as Symbian volumes have decreased, there continues to be demand for the OS from the most ardent of users. In addition, Nokia continues to support Symbian, as evidenced by the PureView initiative on the Nokia 808. Still, as Nokia

emphasizes Windows Phone, IDC expects further declines for Symbian for the rest of this year.

**BlackBerry** continued on its downward trajectory as demand for older BlackBerry devices decreased and the market awaits the official release of BB 10 smartphones later this year. In addition, many companies now permit users to bring their own smartphones, allowing competitor operating systems to take away from BlackBerry's market share. Although RIM has not officially released BB 10, initial glimpses of the platform have shown improvement.

**Linux** maintained its small presence in the worldwide smartphone market, thanks in large part to Samsung's continued emphasis on bada. By the end of the quarter, Samsung accounted for 81.6% of all Linux-powered smartphones, a 3.6% share gain versus the prior-year period. Other vendors, meanwhile, have been experimenting with Android to drive volume. Still, Linux's fortunes are closely tied to Samsung's strategy, which already encompasses Android, Windows Phone, and later this year, Tizen.

**Windows Mobile/Windows Phone** has yet to make significant inroads in the worldwide smartphone market, but 2012 should be considered a ramp-up year for Nokia and Microsoft to boost volumes. Until Nokia speeds the cadence of its smartphone releases or more vendors launch their own Windows Phone-powered smartphones, IDC anticipates slow growth for the operating system.

Top Six Smartphone Operating Systems, Shipments, and Market Share, 2012 Q1 (Units in Millions)

| Mobile Operating System | 1Q12 Unit Shipments | 1Q12 Market Share | 1Q11 Unit Shipments | 1Q11 Market Share | Year-over-Year Change |
|---|---|---|---|---|---|
| Android | 89.9 | 59.0% | 36.7 | 36.1% | 145.0% |
| iOS | 35.1 | 23.0% | 18.6 | 18.3% | 88.7% |
| Symbian | 10.4 | 6.8% | 26.4 | 26.0% | -60.6% |
| BlackBerry OS | 9.7 | 6.4% | 13.8 | 13.6% | -29.7% |
| Linux | 3.5 | 2.3% | 3.2 | 3.1% | 9.4% |
| Windows Phone 7/Windows Mobile | 3.3 | 2.2% | 2.6 | 2.6% | 26.9% |

| | | | | | |
|---|---|---|---|---|---|
| Other | 0.4 | 0.3% | 0.3 | 0.3% | 33.3% |
| Total | 152.3 | 100.0% | 101.6 | 100.0% | 49.9% |

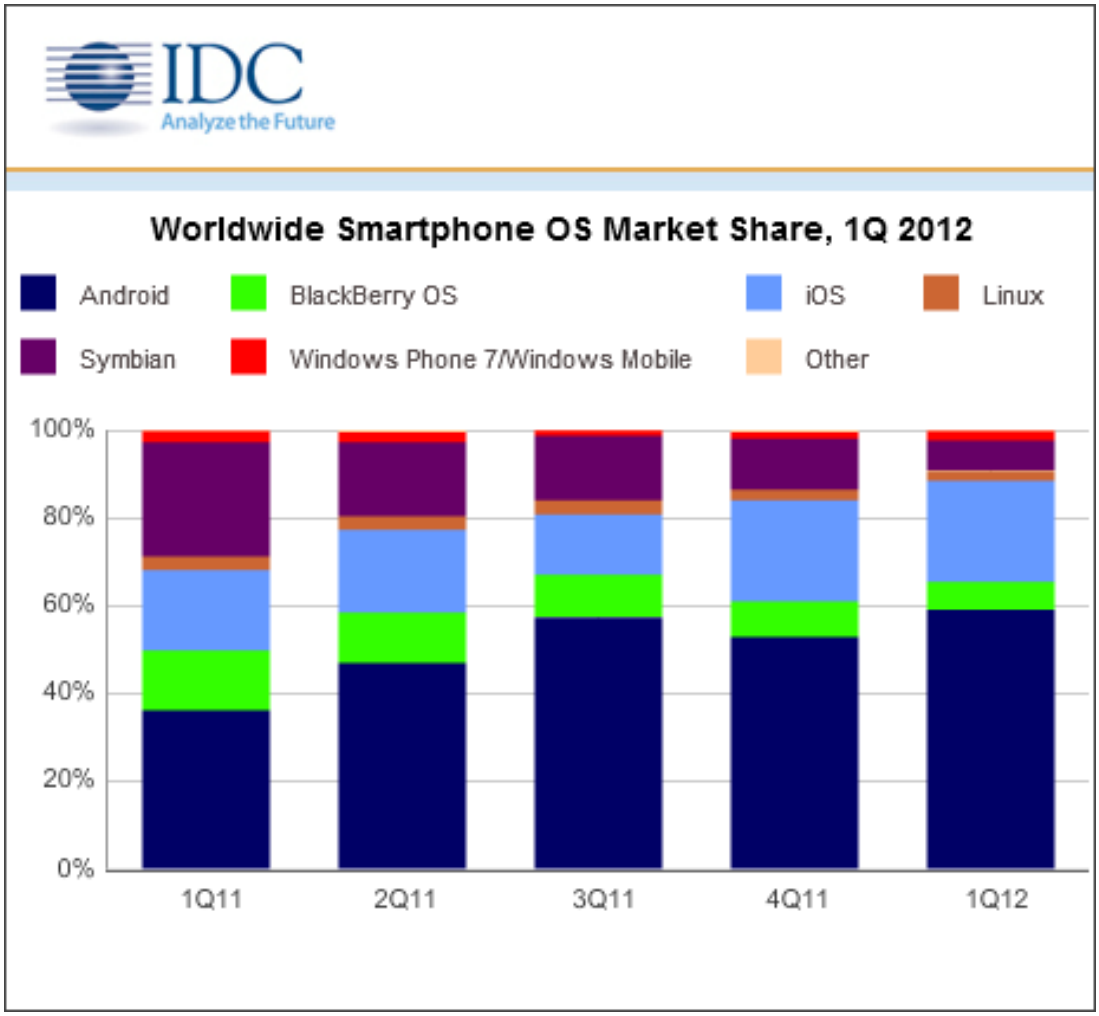



Figure 2.2: Worldwide Smartphone OS Market Share

### 2.3 Why Android

Android, the world's most popular mobile platform

Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast—every day another million users power up their Android devices for the first time and start looking for apps, games, and other digital content.

Android gives you a world-class platform for creating apps and games for Android users everywhere, as well as an open marketplace for distributing to them instantly.


Figure 2.3: Growth of Android OS

Android growth in device activations

Global partnerships and large installed base

Building on the contributions of the open-source Linux community and more than 300 hardware, software, and carrier partners, Android has rapidly become the fastest-growing mobile OS.

Every day more than 1 million new Android devices are activated worldwide.

Android’s openness has made it a favorite for consumers and developers alike, driving strong growth in app consumption. Android users download more than 1.5 billion apps and games from Google Play each month.

With it's partners, Android is continuously pushing the boundaries of hardware and software forward to bring new capabilities to users and developers. For developers, Android innovation lets you build powerful, differentiated applications that use the latest mobile technologies.

**Powerful development framework**

Easily optimize a single binary for phones, tablets, and other devices.

Android gives you everything you need to build best-in-class app experiences. It gives you a single application model that lets you deploy your apps broadly to hundreds of millions of users across a wide range of devices—from phones to tablets and beyond.

Android also gives you tools for creating apps that look great and take advantage of the hardware capabilities available on each device. It automatically adapts your UI to look it's best on each device, while giving you as much control as you want over your UI on different device types.

For example, you can create a single app binary that's optimized for both phone and tablet form factors. You declare your UI in lightweight sets of XML resources, one set for parts of the UI that are common to all form factors and other sets for optimizations specific to phones or tablets. At runtime, Android applies the correct resource sets based on its screen size, density, locale, and so on.

To help you develop efficiently, the Android Developer Tools offer a full Java IDE with advanced features for developing, debugging, and packaging Android apps. Using the IDE, you can develop on any available Android device or create virtual devices that emulate any hardware configuration.

Open marketplace for distributing your apps

Google Play is the premier marketplace for selling and distributing Android apps. When you publish an app on Google Play, you reach the huge installed base of Android. As an open marketplace, Google Play puts you in control of how you sell your products. You can publish whenever you want, as often as you want, and to the customers you want. You can distribute broadly to all markets and devices or focus on specific segments, devices, or ranges of hardware capabilities.

You can monetize in the way that works best for your business—priced or free, with in-app products or subscriptions—for highest engagement and revenues. You also have complete control of the pricing for your apps and in-app products and can set or change prices in any supported currency at any time.

Beyond growing your customer base, Google Play helps you build visibility and engagement across your apps and brand. As your apps rise in popularity, Google Play gives them higher placement in weekly "top" charts and rankings, and for the best apps promotional slots in curated collections.

**Life cycle of an android**

Activities in the system are managed as an *activity stack*. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

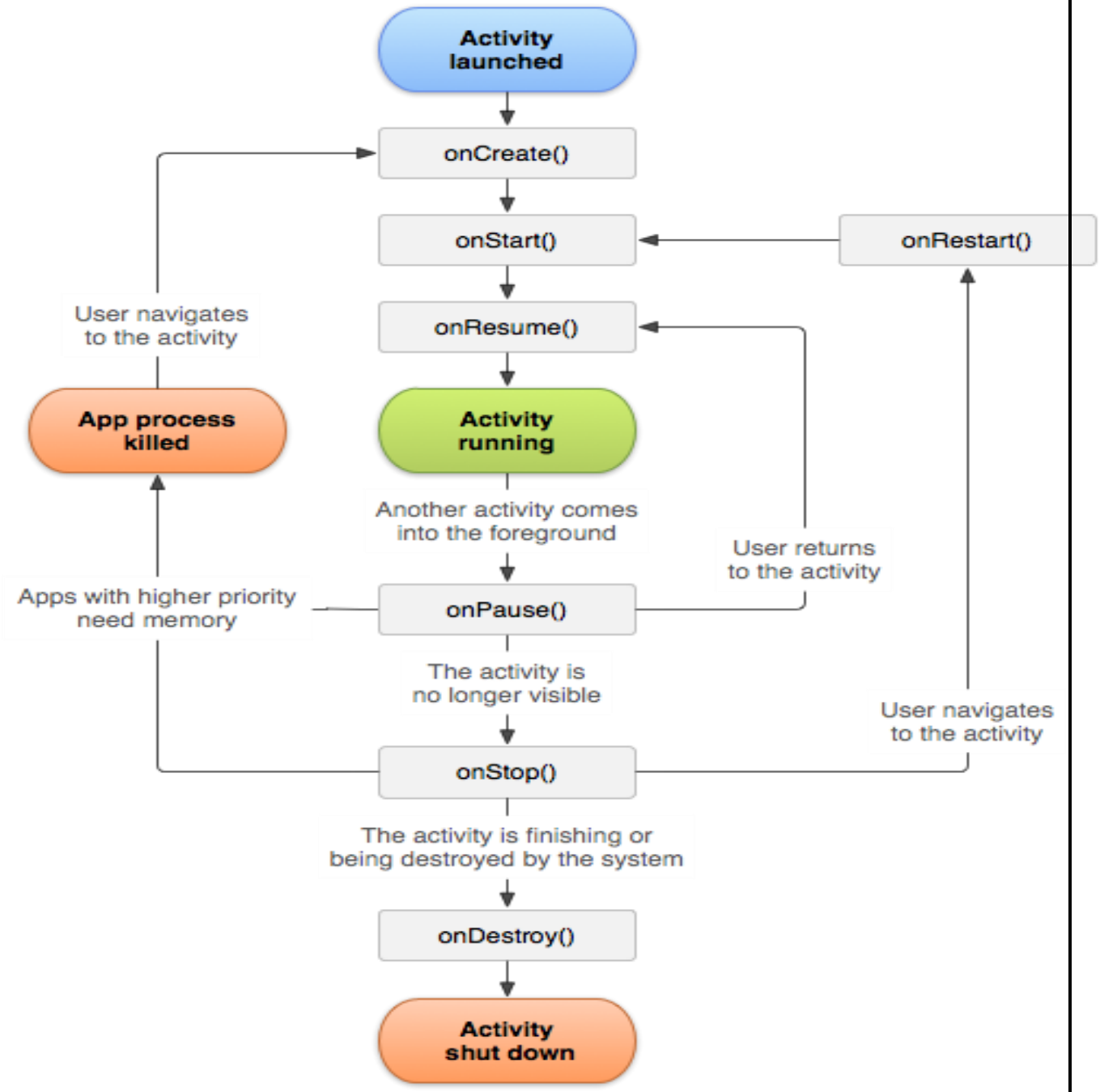
If an activity in the foreground of the screen (at the top of the stack), it is *active* or *running*.

If an activity has lost focus but is still visible (that is, a new non-full-sized or transparent activity has focus on top of your activity), it is *paused*. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.

If an activity is completely obscured by another activity, it is *stopped*. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.

If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

The following diagram shows the important state paths of an Activity. The square rectangles represent callback methods you can implement to perform operations when the Activity moves between states. The colored ovals are major states the Activity can be in.


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

# CHAPTER-3

# EXISTING APPLICATIONS

## 3.1 Existing Application

**In IOS**

First Letters and Phonics

Figure 3.1: First Letters and Phonics

Featuring sleek graphics as well as two original renditions of the alphabet song by popular children's musician Debi Derryberry, First Letters and Phonics will have kids singing about letters and their shapes, the sounds of letters, and words that begin with each letter. Each letter is complemented by a beautiful illustration and a friendly narrator.

Alpha Writer

Figure 3.2: Alpha Writer

The innovative and gorgeous Alpha Writer app from Montessorium teaches children to read, write and spell phonetically while composing words and creating stories. In addition to establishing the basics of language, kids can also work up to identifying consonants and vowels as well as other challenges. Check out the developer's Intro to Letters app for even more practice.

Intro to Math

Figure 3.3: Intro to Math

The Intro to Math app is almost distractingly beautiful, with each graphic looking like a work of art (probably one of the reasons it was featured in a recent Apple commercial). It's intended, however, to be much more than that. Your child will learn how to read and write numbers 0 through 9, to sequence numbers, to tell odds from evens, to solve problems and more.

Park Math

Figure 3.4: Park Math

The very cute Park Math app will help your child learn how to count up to 50 as a rabbit swings, how to add up ducks as they climb to the top of a slide, how to balance a see-saw by adding and subtracting mice, how to subtract as apples fall from a tree, how to order numbered dogs in sequence, and more.

Abby's Train – Learn Colors!


Figure 3.5: Abby's Train

Utilizing a vibrant palette, Abby's Train – Learn Colors! draws children into a world of adorable characters and interesting challenges. Adults can modify the number of colored objects (up to 10) that a child has to choose from, as well as the particular colors the child will be learning. Kids focus on placing toys of a particular color into Abby's train in order to earn stickers as rewards.

**In Android**

ABC Learning Game for Kids


Figure 3.6: ABC Learning Game

This ABC's learning app is designed to enable your child to learn alphabet letters in an enjoyable and rewarding environment.
The program will come in three separate apps:
1. Alphabet letters recognition with animated graphic and sound.
2. Alphabet lowercase and uppercase letters tracing for learning how to write without parent's present.
3. Game to test your child's learning progress.

Kids Learning


Figure 3.7: Kids Learning

An amazing application for your kids.
This application contains fantabulous pictures of Animal, Fruit, Flowers, and Vegetables etc. Just click on the Match button and see the name of things in Hindi and English, so you can learn Hindi and English names of fruits, vegetables and Animals.

0-10 Numbers Baby Flash Cards


Figure 3.8: 0-10 Numbers Baby Flash Cards

0-10 Numbers Baby Flashcards are fun for babies and toddlers. Numbers Flash cards will entertain and teach them as they learn the numbers, count and hear the words. With quiz and practice modes.

0-10 Numbers Baby Flashcards
Language/Sounds: English

Learning Colors for Kids

Figure 3.9: Learning Colors for Kids

This is a small app for easy learning of colors. The app says which color shall be pressed. If a wrong color is pressed the app tells which color was pressed. When the correct color was pressed the app goes on with the next color.

Kids Shapes

Figure 3.10: Kids Shapes

A hands-on, real-life way for kids to learn shapes.

Wouldn't it be just wonderful if there was a simple game for preschoolers that made learning shapes enjoyable? There is! It's called Kids Preschool Shapes.

**Outcome of Existing Application Analysis**

We found that there are many learning application for kids on both Android and iOS platform. All the applications have very rich user interface. In order to compete with these existing application, the user interface of a new application must not only be rich but also user friendly.

One of the drawbacks with these application is that there is no one application which has all sections that are taught to kids in a pre-school. For eg, there are separate application for alphabets, numbers, colors, fruits and animals etc.

Our application contains all sections (alphabets, numbers, objects, games). As a single application it follows learning process. While learning, order should be followed like kids should listen first before writing. Order of learning should be

1. Listen what to learn.

2. Write what has listened

3. Remember what has learned.

Implementation of this process is explained in next chapter.

# CHAPTER-4

# CONTENT OF THE APPLICATION

## 4.1 Content

- Alphabets:

  This section introduces alphabets to the children and allows them to have some practice on the handwriting.

  1. Phonics Introduction to Alphabets.
     Introduces the small and capital alphabets with the sound of the alphabet and an object starting with the alphabet
  2. Drawing
     Allows children to write alphabets (both small and capital).
  3. Alphabet song.
     Shows video of alphabet song.

- Numbers

  This section introduces numbers, allows to write numbers and also shows Addition and Subtraction of numbers.

  1. Phonics Introduction to Numbers
     Introduces numbers from 1 to 9 with the sound of the number and number of objects showing count of that number.
  2. Drawing
     Allows children to write numbers.
  3. Adding Numbers
     Introduces addition of numbers to the children using objects.
  4. Subtracting Numbers
     Introduces subtraction of numbers to the children using objects.
  5. Number Song
     Shows video of Number song.

- Games

  This section introduces games to the children that allows them to have some practice on alphabets and numbers.

  - Alphabetic Games
    Allows children to identify and match (small and capital) alphabets.
    1. Identify Alphabet

Allows children to identify correct alphabet after listening sound of alphabet

2. Match capital with lower case alphabets
   Allows children to match lower case alphabet with capital alphabet.

- Numeric Games
  Allows children to count the objects and find largest and smallest number.
  1. Count the objects
     Allows children to count the objects and choose the correct option.
  2. Find the largest number
     Allows children to choose the largest number among given numbers.
  3. Find the smallest number
     Allows children to choose the smallest number among given numbers.

- General Awareness
  This section introduces general objects with their phonics and shows some information regarding that object
  1. Animals
     Introduces animals to the children with their phonics and shows specific information of that animal.
  2. Fruits
     Introduces fruits to the children with their phonics and shows their importance to our health.
  3. Vegetables
     Introduces vegetables to the children with their phonics and shows their importance.
  4. Colors
     Introduces colors to children with their phonics and shows name of the object representing that color.

# CHAPTER-5

# IMPLEMENTATION

## 5.1 Flow of work

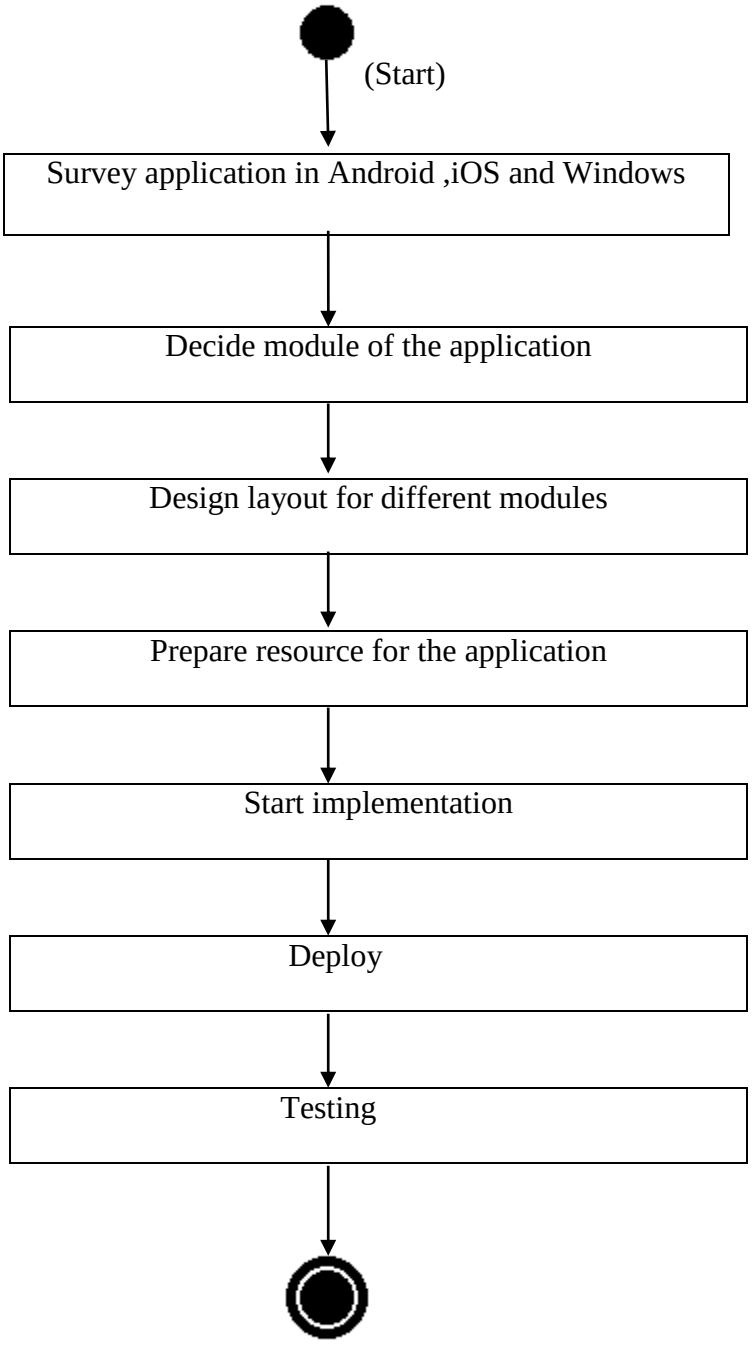
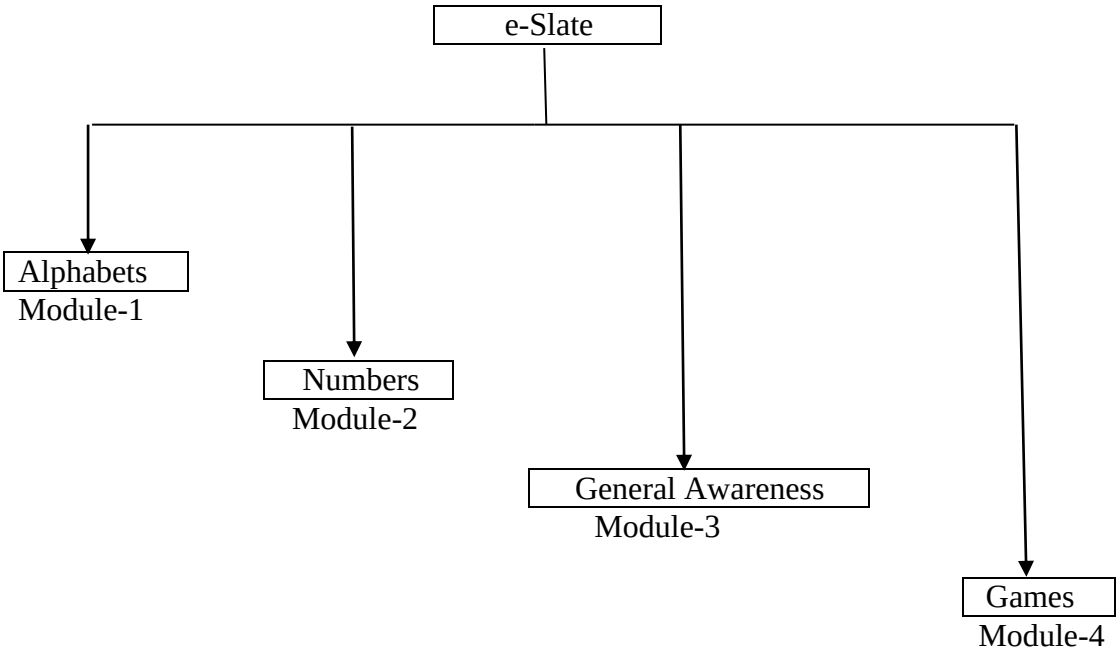



Figure 5.1: Flow of Work

## 5.2 Modules of the application




## 5.3 Screen on launch


Figure 5.3.1: Splash.java

Whenever the application (e-Slate) is launched the Splash Activity is displayed on the screen of the phone. The above animated image is displayed for 4500 milliseconds and then Start.java Activity is called, which displays the following layout.

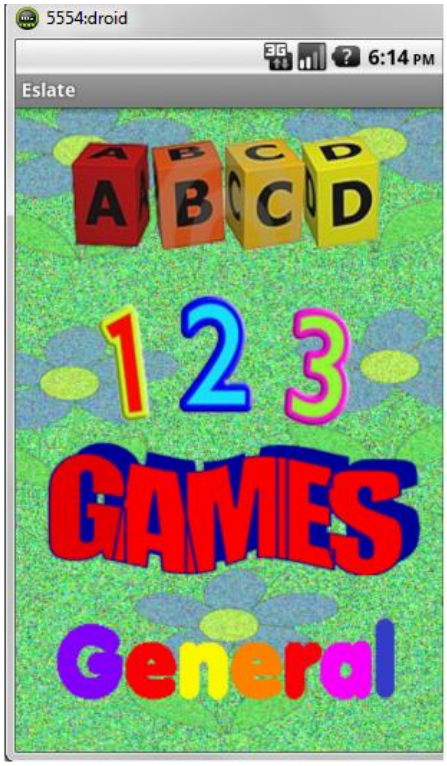



Figure 5.3.2: start.xml

The above layout contains four ImageView and a background image. The ImageViews call to the following respective activities:

- AbcdList.java
- NumberList.java
- GeneralList.java
- GameList.java

Following is the XML code for the above layout.

**start.xml**

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Start"
    android:orientation="vertical"
    android:background="@drawable/back12"
    >
<ImageView
          android:id="@+id/iabcd"
          android:layout_width="fill_parent"
          android:layout_height="125dp"
```

```
            android:focusableInTouchMode="true"
            android:scaleType="fitXY"
            android:src="@drawable/abcd" />

    <ImageView
        android:id="@+id/inum"
        android:layout_width="fill_parent"
        android:layout_height="125dp"
        android:scaleType="fitXY"
        android:src="@drawable/num" />

    <ImageView
        android:id="@+id/igames"
        android:layout_width="fill_parent"
        android:layout_height="100dp"
        android:scaleType="fitXY"
        android:src="@drawable/gamess" />

    <ImageView
        android:id="@+id/igeneral"
        android:layout_width="fill_parent"
        android:layout_height="125dp"
        android:scaleType="fitXY"
        android:src="@drawable/general" />

     </LinearLayout>
```

**Start.java**

```
package com.e_slate;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;
import android.view.animation.AlphaAnimation;
import android.view.animation.Animation;
import android.view.animation.Animation.AnimationListener;
import android.view.animation.AnimationUtils;
import android.view.animation.LinearInterpolator;
import android.widget.ImageView;
public class Start extends Activity implements View.OnClickListener
{
      ImageView abcd, num, games, general;
      Animation rot, blinknum, blinkgame, blinkgeneral;
      MediaPlayer sound;
      Thread t1, t2, t3, t4;
      @Override
      protected void onCreate(Bundle savedInstanceState)
        {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.start);
            abcd = (ImageView) findViewById(R.id.iabcd);
            num = (ImageView) findViewById(R.id.inum);
            games = (ImageView) findViewById(R.id.igames);
            general = (ImageView) findViewById(R.id.igeneral);
            sound = MediaPlayer.create(Start.this, R.raw.alpha);
```

```
        sound.start();
        Animation blinkalpha = cret();
        findViewById(R.id.iabcd).startAnimation(blinkalpha);
        blinkalpha.setAnimationListener(new AnimationListener()
        {         @Override public void onAnimationEnd(Animation
animation) {
              MediaPlayer soundnum = MediaPlayer.create(Start.this,
              R.raw.numbers);
              soundnum.start();
              blinknum = cret();
              findViewById(R.id.inum).startAnimation(blinknum);
              blinknum.setAnimationListener(new AnimationListener()
              {@Override public void onAnimationEnd(Animation
              animation) {
              // TODO Auto-generated method stub
              MediaPlayer soundgame
              =MediaPlayer.create(Start.this,R.raw.games);
              soundgame.start();
              blinkgame = cret();
             findViewById(R.id.igames).startAnimation(blinkgame);
              blinkgame.setAnimationListener(new
              AnimationListener()
              {
              @Override public void onAnimationEnd(Animation animation)
              {
              // TODO Auto-generated method stub
              MediaPlayer soundgeneral=MediaPlayer.create(Start.this,
               R.raw.general);
              soundgeneral.start();
              Animation blinkgeneral = cret();
              findViewById(R.id.igeneral).startAnimation(blinkgeneral);
              }
              @Override
              public void onAnimationRepeat(Animation animation) {
              // TODO Auto-generated method stub
              }
              @Override
              public void onAnimationStart(Animation animation)
              {
              // TODO Auto-generated method stub                    }
              });

        }@Override
        public void onAnimationRepeat(Animation animation) {
        // TODO Auto-generated method stub
        }@Override
        public void onAnimationStart(Animation animation) {
        }});
        }@Override
        public void onAnimationRepeat(Animation animation) {
        // TODO Auto-generated method stub
        }@Override
              public void onAnimationStart(Animation animation) {
```

```
            // TODO Auto-generated method stub
            }});
            abcd.setClickable(true);
            abcd.setOnClickListener(this);
      num.setOnClickListener(this);
      games.setOnClickListener(this);
      general.setOnClickListener(this);
}

      Animation cret() {
      Animation blink = new AlphaAnimation(1, 0);
      blink.setDuration(500); // duration - half a second
      blink.setInterpolator(new LinearInterpolator());
      blink.setRepeatCount(2); // Repeat animation
      // infinitely
      blink.setRepeatMode(Animation.REVERSE); // Reverse animation at
      return blink;
}
      @Override
      public void onClick(View v) {
      switch (v.getId()) {
      case R.id.iabcd:
      Intent i = new Intent(Start.this, AbcdList.class);
            Start.this.startActivity(i);
break;
case R.id.inum:
            Intent i3 = new Intent(Start.this, NumberList.class);
            Start.this.startActivity(i3);
            break;
case R.id.igames:
            Intent i4 = new Intent(Start.this, GameList.class);
            Start.this.startActivity(i4);
            break;
case R.id.igeneral:
            Intent i1 = new Intent(Start.this, GeneralList.class);
            Start.this.startActivity(i1);
            break;
      }
}}
```

### 5.4 Module-1 (Alphabets)

This module starts with the launch of AbcdList.java Activity which displays following layout.

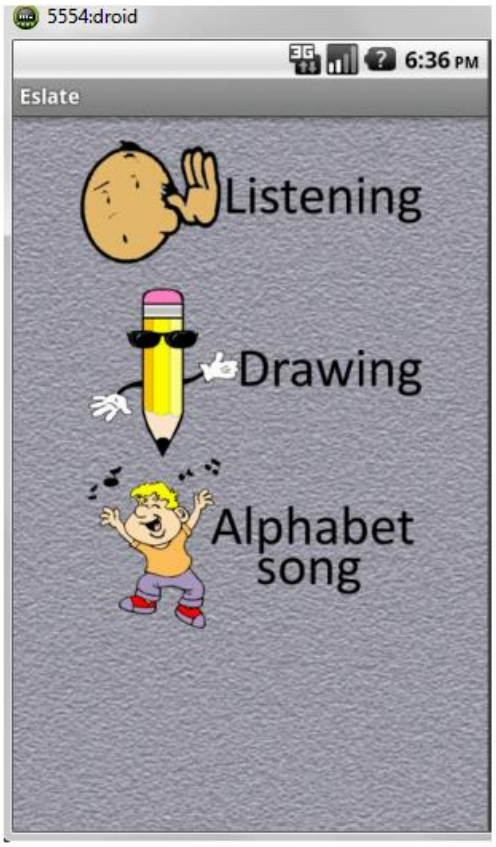



Figure 5.4.1: alphalist.xml

The above layout contains three ImageView. The ImageView calls to following respective Activities

- Listening.java
- Drawing.java
- Alphabet_songs.java

Following is the XML code for above layout.

**AbcdList.java**

```
package com.e_slate;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
```

```
import android.widget.ImageView;

public class AbcdList extends Activity implements View.OnClickListener {


	ImageView listen,draw,song;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.alphalist);
		listen=(ImageView)findViewById(R.id.alphalisteni);
		draw=(ImageView)findViewById(R.id.alphadrawt);
		song=(ImageView)findViewById(R.id.alphasong);

		listen.setOnClickListener(this);
		draw.setOnClickListener(this);
		song.setOnClickListener(this);


	}
	@Override
	public void onClick(View v) {
		// TODO Auto-generated method stub

		switch(v.getId())
		{
		case R.id.alphalisteni:
			Intent i=new Intent(AbcdList.this,Listening.class);
			AbcdList.this.startActivity(i);

		break;
		case R.id.alphadrawt:
			Intent i1=new Intent(AbcdList.this,Drawing.class);
			AbcdList.this.startActivity(i1);

			break;
		case R.id.alphasong:
			Intent i2=new Intent(AbcdList.this,Alphabet_songs.class);
			AbcdList.this.startActivity(i2);

			break;

		}
	}
	public void onBackPressed() {
	   Log.d("CDA", "onBackPressed Called");
	   Intent setIntent = new Intent(AbcdList.this,Start.class);
	   AbcdList.this.startActivity(setIntent);

	}
}
```

This module starts with launch of AbcdList.java Activity. AbcdList.java displays the following layout (alphalist.xml):

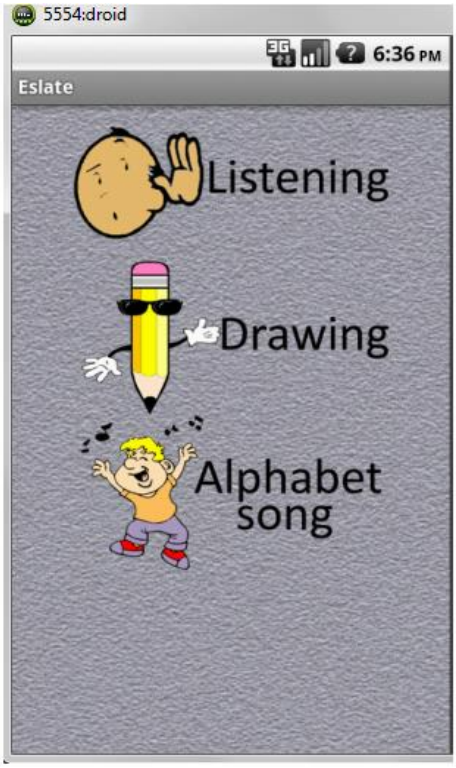



Figure 5.4.2: alphalist.xml

The above layout contains three ImageViews calling the following respective activities:

- Listening.java
- Drawing.java
- Alphabet_songs.java

Following is the code for alphalist.xml and AbcdList.java:

**AbcdList.java**

```
package com.e_slate;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;

public class AbcdList extends Activity implements View.OnClickListener {


	ImageView listen,draw,song;

	@Override
```

```
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.alphalist);
		listen=(ImageView)findViewById(R.id.alphalisteni);
		draw=(ImageView)findViewById(R.id.alphadrawt);
		song=(ImageView)findViewById(R.id.alphasong);

		listen.setOnClickListener(this);
		draw.setOnClickListener(this);
		song.setOnClickListener(this);


	}
	@Override
	public void onClick(View v) {
		// TODO Auto-generated method stub

		switch(v.getId())
		{
		case R.id.alphalisteni:
			Intent i=new Intent(AbcdList.this,Listening.class);
			AbcdList.this.startActivity(i);

		break;
		case R.id.alphadrawt:
			Intent i1=new Intent(AbcdList.this,Drawing.class);
			AbcdList.this.startActivity(i1);

			break;
		case R.id.alphasong:
			Intent i2=new Intent(AbcdList.this,Alphabet_songs.class);
			AbcdList.this.startActivity(i2);

			break;


		}
	}
	public void onBackPressed() {
	   Log.d("CDA", "onBackPressed Called");
	   Intent setIntent = new Intent(AbcdList.this,Start.class);
	   AbcdList.this.startActivity(setIntent);

     }
}
```

When the Listening ImageView in alphalist.xml is clicked, Listening.java is called which displays the following layout:

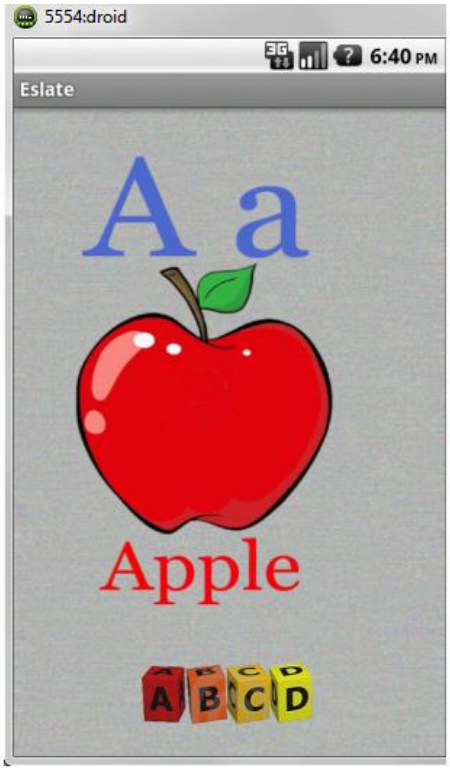



Figure 5.4.3: abcdlisten.xml

The above XML file has HorizontalScrollView, which contains multiple images of alphabets from a to z. The above screen shot displays image of alphabet A. The following screen shot displays image of alphabet B:

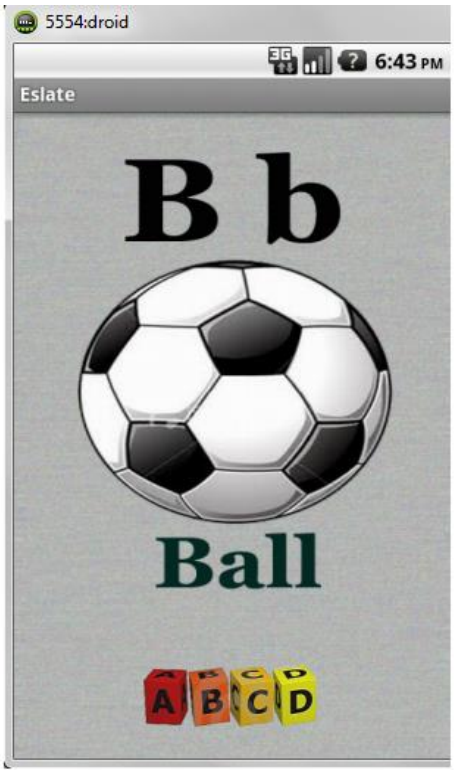



Figure 5.4.4: abcdlisten.xml

Following is the code for abcdlisten.xml and Listening.java:

**Listening.java**

```
package com.e_slate;

import android.app.Activity;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ImageView;

public class Listening extends Activity implements View.OnClickListener {
	ImageView a_alpha,
b_alpha,c_alpha,d_alpha,e_alpha,f_alpha,g_alpha,h_alpha,i_alpha,j_alpha,k_a
lpha,l_alpha
	,m_alpha,n_alpha,o_alpha,p_alpha,q_alpha,r_alpha,s_alpha,t_alpha,u_al
pha,v_alpha,w_alpha,x_alpha,y_alpha,z_alpha;
	MediaPlayer sound;
	Button backtolist;
	Animation scale;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub
		super.onCreate(savedInstanceState);
		setContentView(R.layout.abcdlisten);
		backtolist = (Button) findViewById(R.id.blist);
		backtolist.setOnClickListener(new View.OnClickListener() {
			@Override
			public void onClick(View v) {
				onBackPressed();
			}
		});

		scale = AnimationUtils.loadAnimation(this,
				R.anim.abcdscale);


		a_alpha = (ImageView) findViewById(R.id.aid);
		b_alpha = (ImageView) findViewById(R.id.bid);
		c_alpha = (ImageView) findViewById(R.id.cid);
		d_alpha = (ImageView) findViewById(R.id.did);
		.		.		.
		.		.		.
		.		.		.
		x_alpha = (ImageView) findViewById(R.id.xid);
		y_alpha = (ImageView) findViewById(R.id.yid);
		z_alpha = (ImageView) findViewById(R.id.zid);


		a_alpha.setOnClickListener(this);
		b_alpha.setOnClickListener(this);
		c_alpha.setOnClickListener(this);
		d_alpha.setOnClickListener(this);
		.		.		.
		.		.		.
```

```
            .           .           .
            w_alpha.setOnClickListener(this);
            x_alpha.setOnClickListener(this);
            y_alpha.setOnClickListener(this);
            z_alpha.setOnClickListener(this);



      }
      public void onClick(View v) {
      switch (v.getId()) {
case R.id.aid:findViewById(R.id.aid).startAnimation(scale);
                  sound=MediaPlayer.create(Listening.this,R.raw.aforapple);
                  sound.start();
                  break;

case R.id.bid:
                  findViewById(R.id.bid).startAnimation(scale);
                  sound =MediaPlayer.create(Listening.this,R.raw.bforball);
                  sound.start();
                  break;
case R.id.cid:    findViewById(R.id.cid).startAnimation(scale);
                  sound = MediaPlayer.create(Listening.this,R.raw.cforcat);
                  sound.start();
                  break;
.                 .                 .                 .
.                 .                 .                 .
.                 .                 .                 .

case R.id.yid:
                  findViewById(R.id.yid).startAnimation(scale);
                  sound = MediaPlayer.create(Listening.this,
R.raw.yforyak);
                  sound.start();
                  break;
            case R.id.zid:
                  findViewById(R.id.zid).startAnimation(scale);
                  sound = MediaPlayer.create(Listening.this,
R.raw.zforzoo);
                  sound.start();
                  break;


            }

      }
       public void onBackPressed() {
         Log.d("CDA", "onBackPressed Called");
         Intent setIntent = new Intent(Listening.this,AbcdList.class);
         startActivity(setIntent);
         return;
     }

}
```

When the Drawing ImageView in alphalist.xml is clicked, Drawing.java is called which displays the following layout:

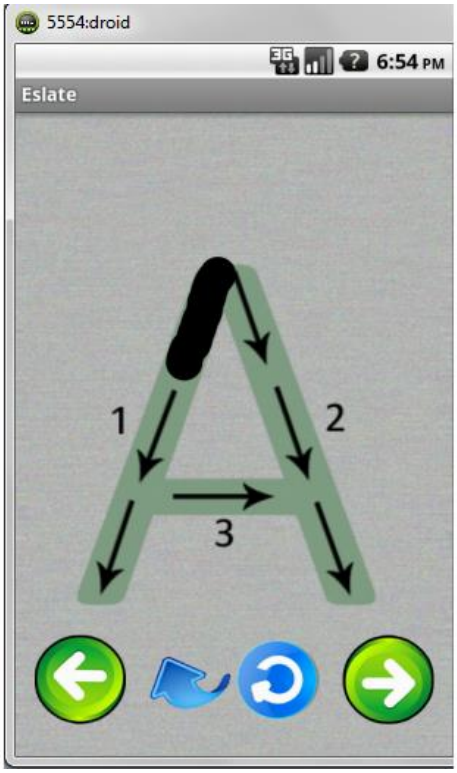



Figure 5.4.5: drawalpha.xml

The above XML file contains a ImageView for image of alphabets and three buttons. Left button is for previous number, right button for next number and center button to refresh the drawing. Whenever the next button is clicked, image of next alphabet is loaded as following:

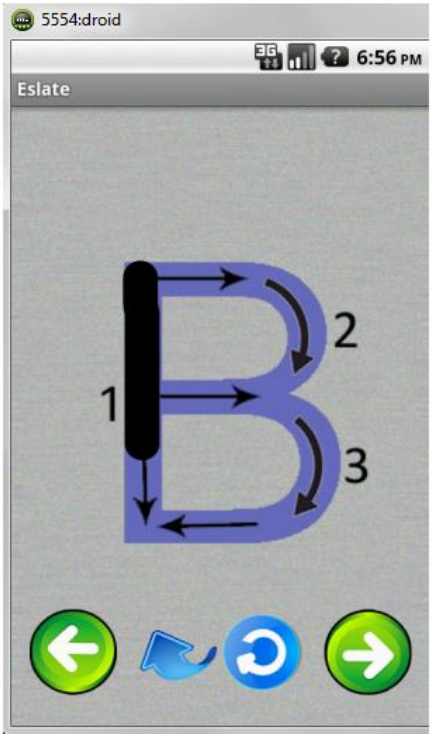



Figure 5.4.6: drawalpha.xml

Following is the code for drawalpha.xml and Drawing.java:

**Drawing.java**

```
package com.e_slate;

import android.app.Activity;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.RelativeLayout;

public class Drawing extends Activity implements View.OnClickListener {
      Button next, back, refresh, flip;
      boolean temp = true;
      int capimages[] ={R.drawable.adraw,R.drawable.bdraw,R.drawable.cdraw,
                  R.drawable.ddraw, R.drawable.edraw, R.drawable.fdraw,
                  R.drawable.gdraw, R.drawable.hdraw, R.drawable.idraw,
                  R.drawable.jdraw, R.drawable.kdraw, R.drawable.ldraw,
                  R.drawable.mdraw, R.drawable.ndraw, R.drawable.odraw,
                  R.drawable.pdraw, R.drawable.qdraw, R.drawable.rdraw,
                  R.drawable.sdraw, R.drawable.tdraw, R.drawable.udraw,
                  R.drawable.vdraw, R.drawable.wdraw, R.drawable.xdraw,
                  R.drawable.ydraw, R.drawable.zdraw };
      int smallimages[] = { R.drawable.sadraw, R.drawable.sbdraw,
                  R.drawable.scdraw, R.drawable.sddraw, R.drawable.sedraw,
                  R.drawable.sfdraw, R.drawable.sgdraw, R.drawable.shdraw,
                  R.drawable.sidraw, R.drawable.sjdraw, R.drawable.skdraw,
                  R.drawable.sldraw, R.drawable.smdraw, R.drawable.sndraw,
                  R.drawable.sodraw, R.drawable.spdraw, R.drawable.sqdraw,
                  R.drawable.srdraw, R.drawable.ssdraw, R.drawable.stdraw,
                  R.drawable.sudraw, R.drawable.svdraw, R.drawable.swdraw,
                  R.drawable.sxdraw, R.drawable.sydraw, R.drawable.szdraw
};
      int sounds[] = { R.raw.a, R.raw.b, R.raw.c, R.raw.d, R.raw.e,
R.raw.f,R.raw.g, R.raw.h, R.raw.i, R.raw.j, R.raw.k, R.raw.l,
R.raw.m,R.raw.n, R.raw.o, R.raw.p, R.raw.q, R.raw.r, R.raw.s,
R.raw.t,R.raw.u, R.raw.v, R.raw.w, R.raw.x, R.raw.y, R.raw.z };
      int x;
      ImageView drawiv;
      DrawView drawview;
      RelativeLayout childLayout;
      RelativeLayout ll;
      MediaPlayer sound;

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);

            setContentView(R.layout.drawalpha);
            drawiv = (ImageView) findViewById(R.id.drawimage);
            drawiv.setImageResource(capimages[0]);
            sound = MediaPlayer.create(Drawing.this, sounds[0]);
            sound.start();
            x = 0;
            addview();
            back = (Button) findViewById(R.id.back);
            next = (Button) findViewById(R.id.next);
            refresh = (Button) findViewById(R.id.refresh);
```

```
            flip = (Button) findViewById(R.id.smallflip);
            back.setOnClickListener(this);
            next.setOnClickListener(this);
            refresh.setOnClickListener(this);
            flip.setOnClickListener(this);

      }

      void addview() {
            ll = (RelativeLayout) findViewById(R.id.ll1);
            drawview = new DrawView(getApplicationContext());
            childLayout = new RelativeLayout(this);
            RelativeLayout.LayoutParams rp = new
RelativeLayout.LayoutParams(
getWallpaperDesiredMinimumWidth()-getWallpaperDesiredMinimumWidth() / 4,
getWallpaperDesiredMinimumHeight()-getWallpaperDesiredMinimumHeight() / 4);
            childLayout.setLayoutParams(rp);
            childLayout.addView(drawview);
            ll.addView(childLayout);
      }

            public void onClick(View v) {

            switch (v.getId()) {
            case R.id.back:
                  if (x != 0) {
                        ll.removeAllViewsInLayout();
                        ll.addView(back);
                        ll.addView(drawiv);
                        ll.addView(next);
                        ll.addView(refresh);
                        ll.addView(flip);
                        x--;
                        drawiv.setImageResource(capimages[x]);
                        sound = MediaPlayer.create(Drawing.this,sounds[x]);
                        sound.start();
                        addview();
                        temp = true;
                  }

                  break;
            case R.id.next:
                  if (x != 25) {

                        ll.removeAllViewsInLayout();
                        ll.addView(back);
                        ll.addView(drawiv);
                        ll.addView(next);
                        ll.addView(refresh);
                        ll.addView(flip);

                        x++;
                        drawiv.setImageResource(capimages[x]);
                        sound = MediaPlayer.create(Drawing.this,sounds[x]);
                        sound.start();
                        addview();
                        temp = true;

                  }
                  break;
```

```
		case R.id.refresh:
			if (x > -1 && x < 26) {
				ll.removeAllViewsInLayout();
				ll.addView(back);
				ll.addView(drawiv);
				ll.addView(next);
				ll.addView(refresh);
				ll.addView(flip);
				if (temp) {
					drawiv.setImageResource(capimages[x]);
			sound = MediaPlayer.create(Drawing.this, sounds[x]);
					sound.start();
				} else{
					drawiv.setImageResource(smallimages[x]);
			sound = MediaPlayer.create(Drawing.this, sounds[x]);
					sound.start();
				}
				addview();
			}
			break;

		case R.id.smallflip:
			if (temp) {
				ll.removeAllViewsInLayout();
				ll.addView(back);
				ll.addView(drawiv);
				ll.addView(next);
				ll.addView(refresh);
				ll.addView(flip);

				drawiv.setImageResource(smallimages[x]);
			sound = MediaPlayer.create(Drawing.this, sounds[x]);
				sound.start();
				addview();
				temp = false;
			} else {
				ll.removeAllViewsInLayout();
				ll.addView(back);
				ll.addView(drawiv);
				ll.addView(next);
				ll.addView(refresh);
				ll.addView(flip);

				drawiv.setImageResource(capimages[x]);
			sound = MediaPlayer.create(Drawing.this, sounds[x]);
				sound.start();
				addview();
				temp = true;
			}

		}
	}

}
```

## DrawView.java

```
package com.e_slate;

import java.util.ArrayList;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;

public class DrawView extends View implements OnTouchListener  {

      Canvas mCanvas;
      Path mPath;
      Paint mPaint;
      ArrayList<Path> paths = new ArrayList<Path>();
      float ax[],ay[];
      int i=0;

      public DrawView(Context context) {
            super(context);
            setFocusable(true);
            setFocusableInTouchMode(true);
            this.setOnTouchListener(this);
            mCanvas = new Canvas();
            mPaint = new Paint();
            mPaint.setAntiAlias(true);
            mPaint.setDither(true);
            mPaint.setColor(Color.BLACK);
            mPaint.setStyle(Paint.Style.STROKE);
            mPaint.setStrokeJoin(Paint.Join.ROUND);
            mPaint.setStrokeCap(Paint.Cap.ROUND);
            mPaint.setStrokeWidth(40);



            mPath = new Path();
            paths.add(mPath);
      }

      public DrawView(Context context, AttributeSet attrs) {
            super(context, attrs);
      }

      public DrawView(Context context,AttributeSet attrs,int
defStyle) {
      super(context, attrs, defStyle);
      }

      @Override
      protected void onSizeChanged(int w, int h, int oldw, int oldh)
{
            super.onSizeChanged(w, h, oldw, oldh);
      }
```

```
         public void colorChanged(int color) {
                mPaint.setColor(color);
            }

        @Override
        protected void onDraw(Canvas canvas) {

              for (Path p : paths) {
                    canvas.drawPath(p, mPaint);


}
        }

        private float mX, mY;
        private static final float TOUCH_TOLERANCE = 4;

        void touch_start(float x, float y) {
              mPath.reset();
              mPath.moveTo(x, y);
              mX = x;
              mY = y;
        }

        void touch_move(float x, float y) {

              float dx = Math.abs(x - mX);
              float dy = Math.abs(y - mY);
              if (dx >= TOUCH_TOLERANCE || dy >= TOUCH_TOLERANCE) {

                    mPath.quadTo(mX, mY, (x + mX) / 2, (y + mY) / 2);

                    mX = x;
                    mY = y;
              }
        }

        void touch_up() {
              mPath.lineTo(mX, mY);
              // commit the path to our offscreen
              mCanvas.drawPath(mPath, mPaint);
              // kill this so we don't double draw
              mPath = new Path();
              paths.add(mPath);
        }

        @Override
        public boolean onTouch(View v, MotionEvent event) {
              float x = event.getX();
              float y = event.getY();

              switch (event.getAction()) {
              case MotionEvent.ACTION_DOWN:
                    touch_start(x, y);
                    invalidate();
                    break;
              case MotionEvent.ACTION_MOVE:
                    touch_move(x, y);
                    invalidate();
                    break;
              case MotionEvent.ACTION_UP:
```

```
                        touch_up();
                        invalidate();
                        break;
                }
                return true;
        }


}
```

When the Alphabet Song ImageView in numlist.xml is clicked, Alphabet_songs.java is called which displays the following layout:

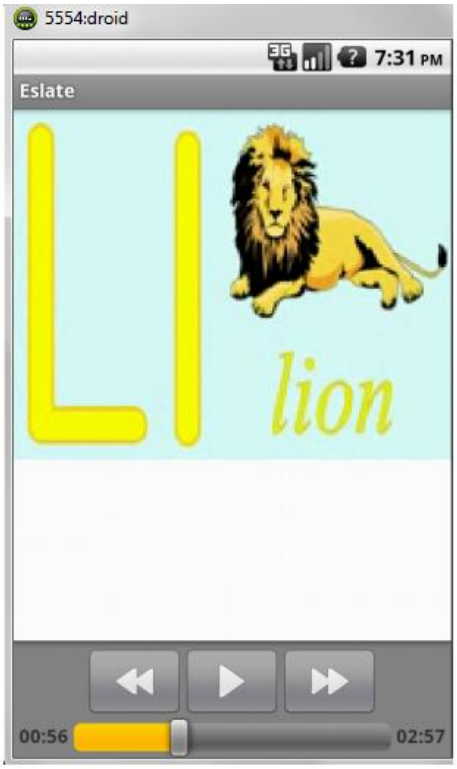



Figure 5.4.7: video.xml

The above XML file contains a video that displays a video of introduction of alphabets. Following is the code for video.xml and Alphabet_songs.java:

Alphabet_songs.java

```
package com.e_slate;

import android.app.Activity;
import android.net.Uri;
import android.os.Bundle;
import android.widget.MediaController;
import android.widget.VideoView;

public class Alphabet_songs extends Activity {

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
```

```
            setContentView(R.layout.video);
       showVideo();
       }
        private void showVideo()
           {
               VideoView vd = (VideoView)findViewById(R.id.videoview1);
               Uri uri =
Uri.parse("android.resource://com.e_slate/"+R.raw.alphasong);
               vd.setVideoURI(uri);
               MediaController mc = new MediaController(this);
               vd.setMediaController(mc);

               vd.requestFocus();
           }

}
```

### 5.5 Module-2 (Numbers)

This module starts with launch of NumberList.java Activity. NumberList.java displays the following layout (numlist.xml):

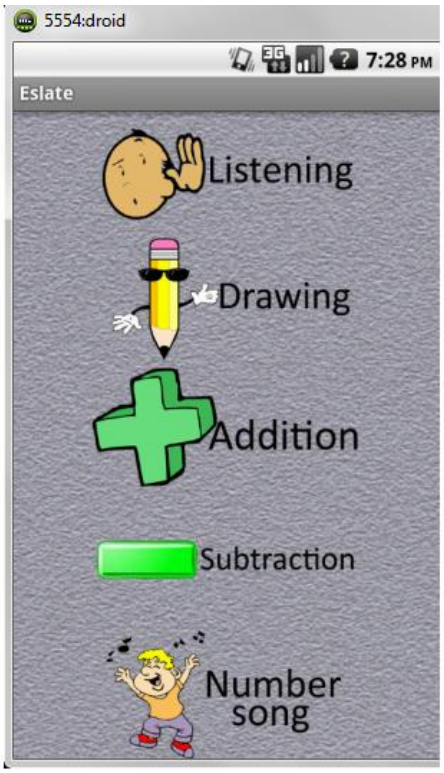



Figure 5.5.1: numlist.xml

The code for numlist.xml is similar to that of alphalist.xml. The above layout contains five ImageViews calling the following respective activities:

- Listen.java
- Draw.java
- Addition.java

- Subtraction.java
- Number _songs.java

When the Listening ImageView in numlist.xml is clicked, Listen.java is called which displays the following layout:


Figure 5.5.2: numberlisten.xml

The above XML file has HorizontalScrollView, which contains multiple images of numbers from 1 to 9. The code for Listen.java and numberlisten.xml is similar to Listening.java and abcdlisten.xml respectively. The above screen shot displays image of number 1. The following screen shot displays image of number 2:


Figure 5.5.3: numberlisten.xml

When the Drawing ImageView in numlist.xml is clicked, Draw.java is called which displays the following layout:

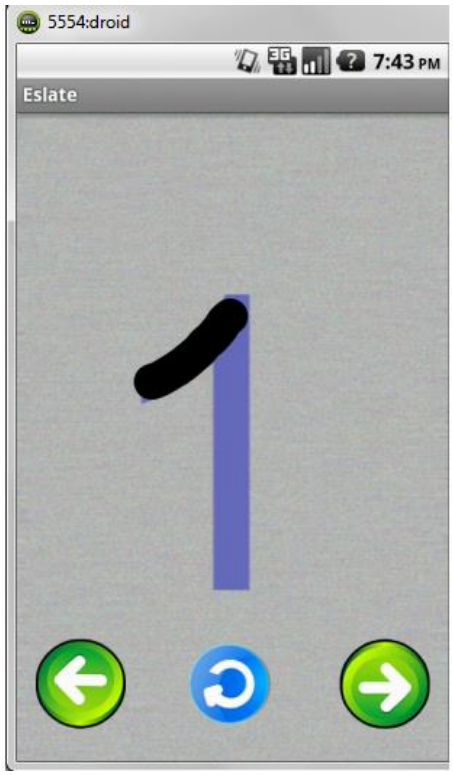



Figure 5.5.4: drawnum.xml

The above XML file contains a ImageView for image of number and three buttons. Left button is for previous number , right button for next number and centre button to refresh the drawing. The code for Draw.java and drawnum.xml is similar to Drawing.java and drawalpha.xml respectively. Whenever the next button is clicked, image of next number is loaded as following:

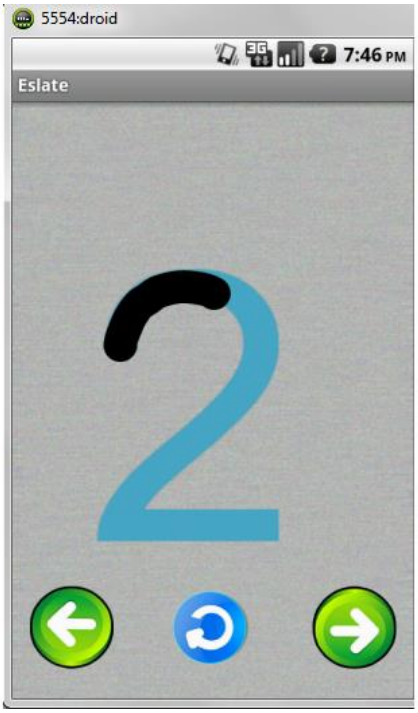



Figure 5.5.6: drawnum.xml

When the Addition ImageView in numlist.xml is clicked, Addition.java is called which displays the following layout:

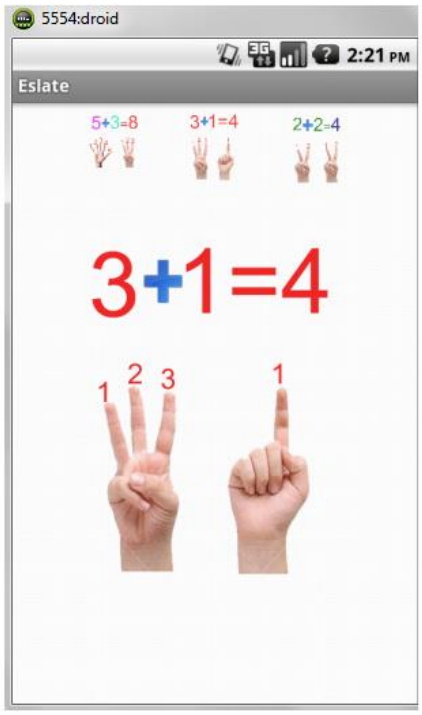



Figure 5.5.7: commonswitcher.xml

The above XML file has ImageSwitcher which contains multiple images displaying concept of addition. Following is the code for commonswitcher.xml and Addition.java:

**Addition.java**

```
package com.e_slate;

import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.AnimationUtils;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.Gallery;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher;
import android.widget.AdapterView.OnItemSelectedListener;


public class Addition  extends Activity implements
ViewSwitcher.ViewFactory, OnItemSelectedListener {

  private Gallery gallery;
  private ImageSwitcher imageSwitcher;

  private ImageAdapter ia;

  @Override
  public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.commonswitcher);

        gallery = (Gallery) findViewById(R.id.gallery1);
        imageSwitcher = (ImageSwitcher) findViewById(R.id.is);

        imageSwitcher.setFactory(this);
        imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                  android.R.anim.fade_in));
        imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                  android.R.anim.fade_out));


        ia = new ImageAdapter(this);
        gallery.setAdapter(ia);

        //Event listener
        gallery.setOnItemSelectedListener(this);
    }

    private class ImageAdapter extends BaseAdapter {
        private Context context;

        private ImageAdapter(Context context) {
            this.context = context;
        }

        private int[] IMAGE_IDS = {

R.drawable.fiveplusthree,R.drawable.threeplusone,R.drawable.twoplustwo
        };

        public int getCount() {
            return IMAGE_IDS.length;
        }

        public Object getItem(int position) {
            return IMAGE_IDS[position];
        }

        public long getItemId(int position) {
            return position;
        }

        public View getView(int position, View convertView, ViewGroup
parent) {
            ImageView iv = new ImageView(context);
            iv.setImageResource(IMAGE_IDS[position]);
            iv.setLayoutParams(new Gallery.LayoutParams(150, 100));
            return iv;
        }
    }

    public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {
        int imageResourceId = (Integer) ia.getItem(position);
        imageSwitcher.setImageResource(imageResourceId);
    }

    public void onNothingSelected(AdapterView<?> parent) {
    }
```

```
  public View makeView() {
        ImageView i = new ImageView(this);
        i.setScaleType(ImageView.ScaleType.FIT_CENTER);
        return i;
  }
}
```

When the Subtraction ImageView in numlist.xml is clicked, Subtraction.java is called which displays the following layout:

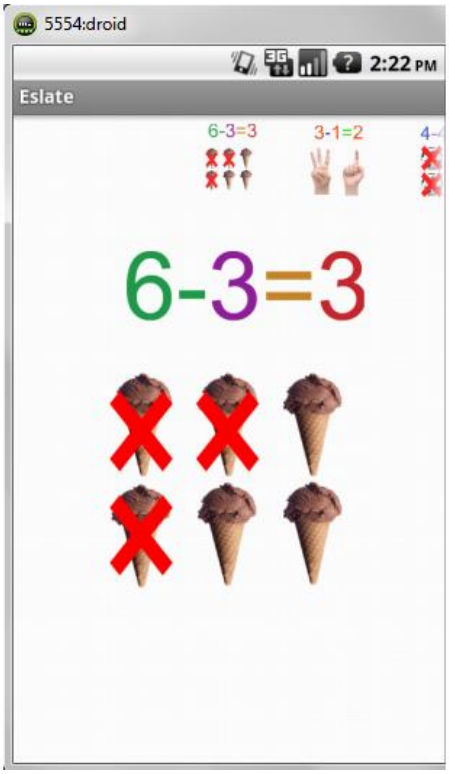



Figure 5.5.8: commonswitcher.xml

The above XML file has ImageSwitcher which contains multiple images displaying concept of subtraction. Following is the code Subtraction.java:

**Subtraction.java**

```
package com.e_slate;

import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.AnimationUtils;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.Gallery;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher;
```

```
import android.widget.AdapterView.OnItemSelectedListener;


public class Subtraction  extends Activity implements
ViewSwitcher.ViewFactory, OnItemSelectedListener {

  private Gallery gallery;
  private ImageSwitcher imageSwitcher;

  private ImageAdapter ia;

  @Override
  public void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.commonswitcher);

       gallery = (Gallery) findViewById(R.id.gallery1);
       imageSwitcher = (ImageSwitcher) findViewById(R.id.is);

       imageSwitcher.setFactory(this);
       imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                 android.R.anim.fade_in));
       imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
                 android.R.anim.fade_out));


       ia = new ImageAdapter(this);
       gallery.setAdapter(ia);

       //Event listener
       gallery.setOnItemSelectedListener(this);
  }

  private class ImageAdapter extends BaseAdapter {
       private Context context;

       private ImageAdapter(Context context) {
            this.context = context;
       }

       private int[] IMAGE_IDS = {

R.drawable.sixminusthree,R.drawable.threeminusone,R.drawable.fourminusfour
       };

       public int getCount() {
            return IMAGE_IDS.length;
       }

       public Object getItem(int position) {
            return IMAGE_IDS[position];
       }

       public long getItemId(int position) {
            return position;
       }

       public View getView(int position, View convertView, ViewGroup
parent) {
            ImageView iv = new ImageView(context);
            iv.setImageResource(IMAGE_IDS[position]);
```

```
            iv.setLayoutParams(new Gallery.LayoutParams(150, 100));
            return iv;
        }
    }

    public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {
        int imageResourceId = (Integer) ia.getItem(position);
        imageSwitcher.setImageResource(imageResourceId);
    }

    public void onNothingSelected(AdapterView<?> parent) {
    }

    public View makeView() {
        ImageView i = new ImageView(this);
        i.setScaleType(ImageView.ScaleType.FIT_CENTER);
        return i;
    }
}
```

When the Number Song ImageView in numlist.xml is clicked, Number_songs.java is called which displays the following layout:


Figure 5.5.9: video.xml

The above XML file contains a video that displays a video of introduction of numbers. The code for Number_songs.java is similar to Alphabet_songs.java.

### 5.6 Module-3 (Games)

This module starts with the launch of GameList.java Activity which displays following layout.

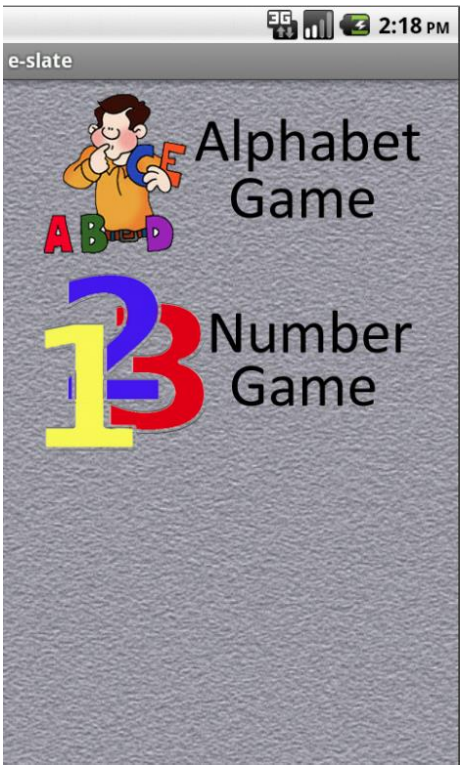



Figure 5.6.1: gamelist.xml

This layout contains two image views which are linked to the two respective game lists. One of them is the alphabet game and other is number game.

**GameList.java**

```
package com.e_slate;
public class GameList extends Activity implements View.OnClickListener {
	ImageView Alphagame,Numgame;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.gamelist);
	Alphagame.setOnClickListener(this);
	Numgame.setOnClickListener(this);	}
	@Override
	public void onClick(View v) {
		switch(v.getId())
		{
		case R.id.alphagame:
			Intent i=new Intent("com.e_slate.AlphaGameList");
			startActivity(i);
			break;
```

```
            case R.id.numgame:
                  Intent i1=new Intent("com.e_slate.NumGameList");
                  startActivity(i1);
                  break;
      }     }
         public void onBackPressed() {
           Log.d("CDA", "onBackPressed Called");
           Intent setIntent = new Intent(GameList.this,Start.class);
           startActivity(setIntent);     return;
}
}
```

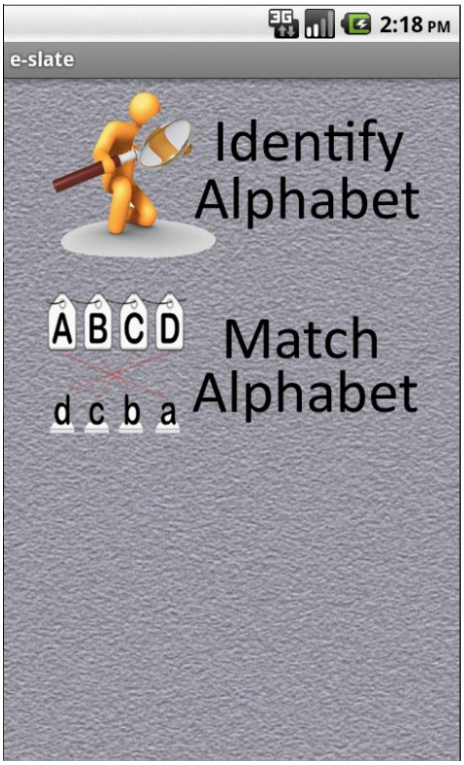



Figure 5.6.2: alphagamelist.xml

This layout contains two image views which are linked to the two respective alphabet game lists. One of them is the identify alphabet game and other is match alphabet game.

**AlphaGameList.java**

```
public class AlphaGameList extends Activity implements View.OnClickListener
{
      ImageView identify,match;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.alphagamelist);
            identify=(ImageView)findViewById(R.id.identify);
            match=(ImageView)findViewById(R.id.match);
            identify.setOnClickListener(this);
            match.setOnClickListener(this);
```

```
	}
	@Override
	public void onClick(View v) {
		switch(v.getId())
		{
		case R.id.identify:
			Intent i=new Intent(AlphaGameList.this,Identify.class);
			startActivity(i);	break;
		case R.id.match:
			Intent i1=new Intent(AlphaGameList.this,Match.class);
			startActivity(i1);		break;
		}
	}
	   public void onBackPressed()
	  {
		Log.d("CDA", "onBackPressed Called");
		Intent setIntent = new
		Intent(AlphaGameList.this,GameList.class);
			startActivity(setIntent);		return;
	   }
	// this function will goto GameList.java when back hardware button is
pressed
	}
```


Figure 5.6.3: identify.xml

In this layout when the play button is clicked any one alphabet voice is played at random. Now the user has to identify the alphabet from the voice played and has to tap one of the four ImageViews given as an option. If the selected answer is wrong then a message is displayed as the user has selected a wrong answer, or if the answer is correct the message is displayed correct.

**Identify.java**

```
public class Identify extends Activity implements View.OnClickListener
{
	int min;
	int nos[]=new int[26];
	ImageView ques;
	ImageView opt1,opt2,opt3,opt4;
	Random r1=new Random();
	boolean st=false;
	MediaPlayer s;
	int temp;
	TextView txt1;
	LinearLayout l1;
	Button n,n1,n2;
protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.identify);
		ques=(ImageView)findViewById(R.id.imageView1);
		opt1=(ImageView)findViewById(R.id.iAOpt1);
		opt2=(ImageView)findViewById(R.id.iAOpt2);
		opt3=(ImageView)findViewById(R.id.iAOpt3);
		opt4=(ImageView)findViewById(R.id.iAOpt4);
		txt1=(TextView)findViewById(R.id.textView1);
		l1=(LinearLayout)findViewById(R.id.lin1);
		n=(Button)findViewById(R.id.next);
		n1=(Button)findViewById(R.id.alphabets);
		n2=(Button)findViewById(R.id.home);
		nos[0]=generateRandom(26);
		changeImage(opt1,nos[0]);
		int temp1;
		//select four random alphabets
		do
		{
		temp1=generateRandom(26);
		}while(temp1==nos[0]);
		changeImage(opt2,temp1);
		nos[1]=temp1;
		do
		{
		temp1=generateRandom(26);
		}while(temp1==nos[0]||temp1==nos[1]);
		changeImage(opt3,temp1);
		nos[2]=temp1;
		do
		{
		temp1=generateRandom(26);
		}while(temp1==nos[0]||temp1==nos[1]||temp1==nos[2]);
		changeImage(opt4,temp1);
		nos[3]=temp1;
		temp1=generateRandom(4);
		temp=nos[temp1];	//any one from the 4
```

```
            //selectVoice(temp);
            //s=MediaPlayer.create(Identify.this,R.raw.a);
            ques.setOnClickListener(this);
            opt1.setOnClickListener(this);
            opt2.setOnClickListener(this);
            opt3.setOnClickListener(this);
            opt4.setOnClickListener(this);
            n.setOnClickListener(this);
            n1.setOnClickListener(this);
            n2.setOnClickListener(this);
      }//end func protected

public void selectVoice (int r)
{
      switch (r)
      {
      case 0:
      {
            s=MediaPlayer.create(Identify.this,R.raw.a);
            break;
      }
      case 1:
      {
            s=MediaPlayer.create(Identify.this,R.raw.b);
            break;
      }
            .
            .
            .
      case 25:
      {
            s=MediaPlayer.create(Identify.this,R.raw.z);
            break;
      }
      }
}

      public int generateRandom(int re)
      {           return (r1.nextInt(re));
      }
      public void changeImage(ImageView rs,int sp)
      {
            switch (sp)
            {
            case 0:
            {
                  rs.setImageResource(R.drawable.gamea);
                  break;
            }
            case 1:
            {
                  rs.setImageResource(R.drawable.gameb);
                  break;
```

```
	}
		.
		.
		.
		.
	case 25:
	{
		rs.setImageResource(R.drawable.gamez);
		break;
	}
	}
}
@Override
public void onClick(View v) {
	switch(v.getId())
{
	case (R.id.iAOpt1):
	{
		if(temp==nos[0])
		{
			txt1.setText("Yes This is the right answer");
		}
		else
		{
		txt1.setText("Sorry This is the wrong answer");
		}
		break;
	}
		case (R.id.imageView1):
		{
			selectVoice(temp);
			s.start();
			break;
		}
		case (R.id.next):
		{
		Intent i=new Intent(Identify.this, Identify.class);
			startActivity(i);
			break;
		}
		case (R.id.alphabets):
		{
		Intent i=new Intent(Identify.this, Listening.class);
			startActivity(i);
			break;
		}
		case (R.id.home):
		{
		Intent i=new Intent(Identify.this, GameList.class);
			startActivity(i);
			break;
		}
}//end switch
```

```
        s=null;
        }
}
```

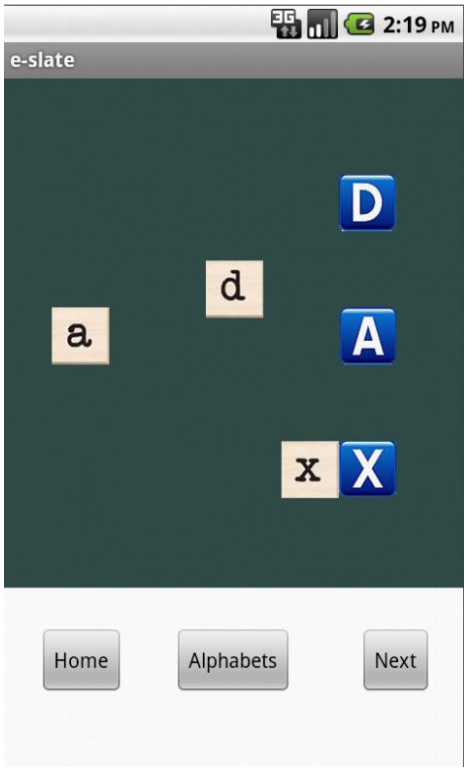



Figure 5.6.4: match.xml

The above layout is for match the following exercise. First column contains randomly generated lowercase alphabets placed in random way. Second column contains the corresponding uppercase alphabets placed randomly. Now the kid has to drag the lowercase alphabet and place it left to the corresponding uppercase alphabet. If all the lowercase alphabets are matched properly then only a new match exercise is displayed else a message is displayed that the matching is not proper. This exercise helps the kid to relate lowercase and uppercase of alphabets.

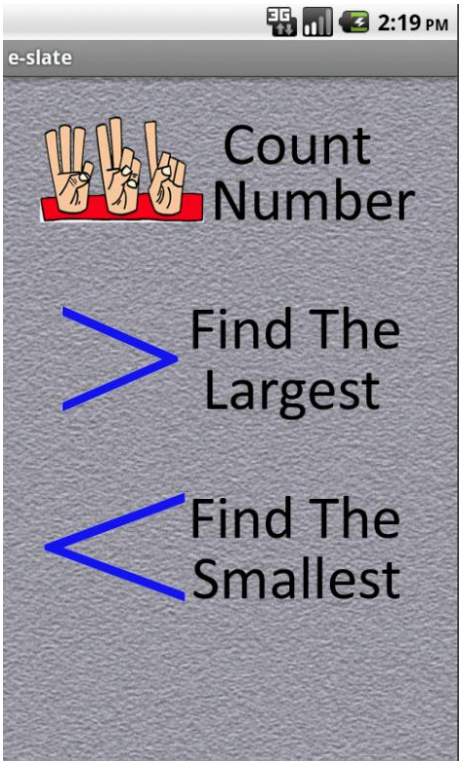


Figure 5.6.5: numgamelist.xml

This layout contains three image views which are linked to the two respective number game lists. One of them is the count number game second is find the largest game and the last one is find the smallest game.

**NumGameList.java**

```
public class NumGameList extends Activity implements View.OnClickListener {
	ImageView count,largest,smallest;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.numgamelist);
		count=(ImageView)findViewById(R.id.count);
		largest=(ImageView)findViewById(R.id.largest);
		smallest=(ImageView)findViewById(R.id.smallest);
		count.setOnClickListener(this);
		largest.setOnClickListener(this);
		smallest.setOnClickListener(this);
	}
	@Override
	public void onClick(View v) {
		switch(v.getId())
		{
		case R.id.largest:
			Intent i=new
	Intent(NumGameList.this,FindTheLargest.class);
			NumGameList.this.startActivity(i);		break;
```

```
            case R.id.smallest:
            Intent i1=new Intent(NumGameList.this,FindTheSmallest.class);
            NumGameList.this.startActivity(i1);       break;

            case R.id.count:
            Intent i2=new Intent(NumGameList.this,Count.class);
            NumGameList.this.startActivity(i2);
            break;
            }
      }
public void onBackPressed() {
           Log.d("CDA", "onBackPressed Called");
           Intent setIntent = new Intent(NumGameList.this,GameList.class);
           startActivity(setIntent);                                  return;
       }
}
```

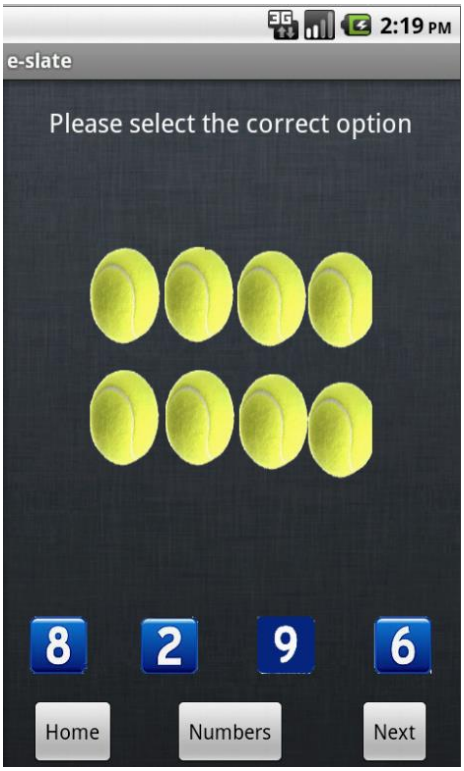



Figure 5.6.6: count.xml

In this layout an image is displayed at random. Now the user has to count the number of object from the image and has to tap one of the four ImageViews given as an option. If the selected answer is wrong then a message is displayed as the user has selected a wrong answer, or if the answer is correct the message is displayed correct.

**Count.java**

```
public class Count extends Activity implements View.OnClickListener
{
	int min;    int nos[]=new int[9];
	ImageView ques;   ImageView opt1,opt2,opt3,opt4;
	Random r1=new Random();       boolean st=false;
	int temp;         TextView txt1;    Button n,n1,n2;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.count);
		ques=(ImageView)findViewById(R.id.mainImage);
		opt1=(ImageView)findViewById(R.id.iAOOpt1);
		opt2=(ImageView)findViewById(R.id.iAOOpt2);
		opt3=(ImageView)findViewById(R.id.iAOOpt3);
		opt4=(ImageView)findViewById(R.id.iAOOpt4);
		txt1=(TextView)findViewById(R.id.textView1);
		n=(Button)findViewById(R.id.next1);
		n1=(Button)findViewById(R.id.numbers);
		n2=(Button)findViewById(R.id.home1);
		nos[0]=generateRandom(9);
		changeImage(opt1,nos[0]);
		int temp1;
		do
		{
		temp1=generateRandom(9);
		}while(temp1==nos[0]);
		changeImage(opt2,temp1);
		nos[1]=temp1;
		do
		{     temp1=generateRandom(9);
		}while(temp1==nos[0]||temp1==nos[1]);
		changeImage(opt3,temp1);
		nos[2]=temp1;
		do
		{     temp1=generateRandom(9);
		}while(temp1==nos[0]||temp1==nos[1]||temp1==nos[2]);
		changeImage(opt4,temp1);
		nos[3]=temp1;
		temp1=generateRandom(4);
		temp=nos[temp1];      //any one from the 4
		changeMainImage(ques,temp);
		opt1.setOnClickListener(this);
		opt2.setOnClickListener(this);
		opt3.setOnClickListener(this);
		opt4.setOnClickListener(this);
		n.setOnClickListener(this);
		n1.setOnClickListener(this);
		n2.setOnClickListener(this);
	}//end func protected
	public int generateRandom(int re)
	{     return (r1.nextInt(re));            }
```

```
public void changeImage(ImageView rs,int sp)
{
	switch (sp)
	{
	case 0:
	{	rs.setImageResource(R.drawable.n1);
		break;
	}
			.
			.
			.
			.
	case 8:
	{	rs.setImageResource(R.drawable.n9);
		break;
	}
	}
}
public void changeMainImage(ImageView rs,int sp)
{
	switch (sp)
	{
	case 0:
	{
		rs.setImageResource(R.drawable.none);
		break;
	}
				.
				.
				.
				.
	case 8:
	{	rs.setImageResource(R.drawable.nnine);
		break;
	}		}		}
@Override
public void onClick(View v) {
	// TODO Auto-generated method stub
	switch(v.getId())
{
	case (R.id.iAOOpt1):
	{
		if(temp==nos[0])
		{
			txt1.setText("Yes This is the right answer");
		}
		else
		{
			txt1.setText("Sorry This is the wrong answer");
		}
		break;
	}
```

```
case (R.id.iAOOpt2):
{
      if(temp==nos[1])
      {
            txt1.setText("Yes This is the right answer");
      }
      else
      {
            txt1.setText("Sorry This is the wrong answer");
      }
      break;
}
case (R.id.iAOOpt3):
{
      if(temp==nos[2])
      {
            txt1.setText("Yes This is the right answer");
      }
else
      {
            txt1.setText("Sorry This is the wrong answer");
      }
      break;
}
case (R.id.iAOOpt4):
{
      if(temp==nos[3])
      {
            txt1.setText("Yes This is the right answer");
      }
      else
      {
            txt1.setText("Sorry This is the wrong answer");
      }
      break;
}
case (R.id.next1):
{
      Intent i=new Intent(Count.this, Count.class);
      startActivity(i);
      break;
}
case (R.id.numbers):
{
      Intent i=new Intent(Count.this, NumGameList.class);
      startActivity(i);
      break;
}
case (R.id.home1):
{
      Intent i=new Intent(Count.this, GameList.class);
      startActivity(i);
      break;
```

```
            }
      }//end switch
      }
}
```


Figure 5.6.7: findthelargest.xml

In this layout four numbers are displayed at random in the ImageView. Now the user has to select the largest number from the four images and has to tap on one of the four ImageViews given as an option. If the selected answer is wrong then a message is displayed as the user has selected a wrong answer, or if the answer is correct the message is displayed correct.

**FindTheLargest.java**

```
public class FindTheLargest extends Activity implements
View.OnClickListener{
      int max;    int nos[]=new int[4];
      TextView ques;    ImageView opt1,opt2,opt3,opt4;
      Random r1=new Random();       boolean st=false;
      Button n,n1,n2;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.findthelargest);
            ques=(TextView)findViewById(R.id.tVQues);
            opt1=(ImageView)findViewById(R.id.iVOpt1);
            opt2=(ImageView)findViewById(R.id.iVOpt2);
            opt3=(ImageView)findViewById(R.id.iVOpt3);
            opt4=(ImageView)findViewById(R.id.iVOpt4);
            n=(Button)findViewById(R.id.next2);
```

```
        n1=(Button)findViewById(R.id.numbers2);
        n2=(Button)findViewById(R.id.home2);
        Animation blink = new AlphaAnimation(1, 0);
        blink.setDuration(500); // duration - half a scond
        blink.setInterpolator(new LinearInterpolator());
        blink.setRepeatCount(2); // Repeat animation
        blink.setRepeatMode(Animation.REVERSE); // Reverse animation at
        opt1.startAnimation(blink);
        opt2.startAnimation(blink);
        opt3.startAnimation(blink);
        opt4.startAnimation(blink);
        nos[0]=generateRandom();
        changeImage(opt1,nos[0]);
        int temp;
        do{
        temp=generateRandom();
        }while(temp==nos[0]);
        changeImage(opt2,temp);
        nos[1]=temp;
        do{
        temp=generateRandom();
        }while(temp==nos[0]||temp==nos[1]);
        changeImage(opt3,temp);
        nos[2]=temp;
        do
        {temp=generateRandom();
        }while(temp==nos[0]||temp==nos[1]||temp==nos[2]);
        changeImage(opt4,temp);
        nos[3]=temp;
        max=nos[0];
        if (nos[1]>max)
              max=nos[1];
        if (nos[2]>max)
              max=nos[2];
        if (nos[3]>max)
              max=nos[3];
        opt1.setOnClickListener(this);
        opt2.setOnClickListener(this);
        opt3.setOnClickListener(this);
        opt4.setOnClickListener(this);
        n.setOnClickListener(this);
        n1.setOnClickListener(this);
        n2.setOnClickListener(this);

}//end func protected
public int generateRandom()
{           return (r1.nextInt(10));
}
public void changeImage(ImageView rs,int sp)
{
      switch (sp)
      {
      case 0:
```

```
        {
              rs.setImageResource(R.drawable.n0);
              break;
        }     .
              .
              .
        case 9:
        {
              rs.setImageResource(R.drawable.n9);
              break;
        }}
}
```


Figure 5.6.8: findthesmallest.xml

In this layout four numbers are displayed at random in the ImageView. Now the user has to select the smallest number from the four images and has to tap on one of the four ImageViews given as an option. If the selected answer is wrong then a message is displayed as the user has selected a wrong answer, or if the answer is correct the message is displayed correct.

## 5.7 Module-4 (General)

This module starts with the launch of General.java Activity which displays following layout.

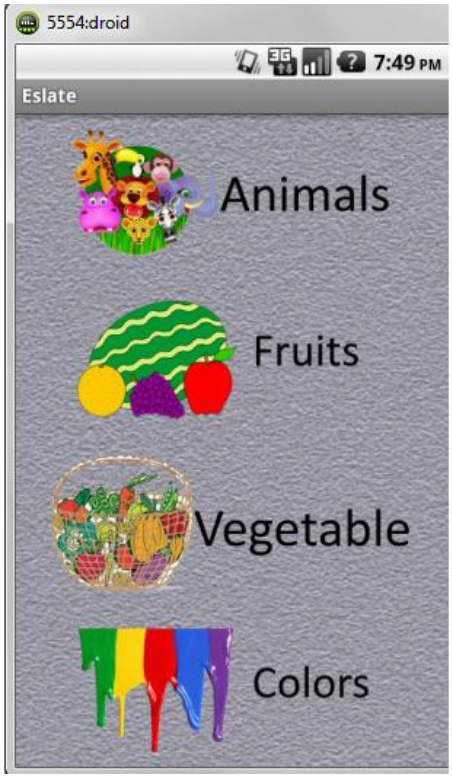



Figure 5.7.1: generalist.xml

This layout contains four image views which are linked to the four respective general lists. One of them is the animals, second is fruits, third is vegetable and the last one is colors.

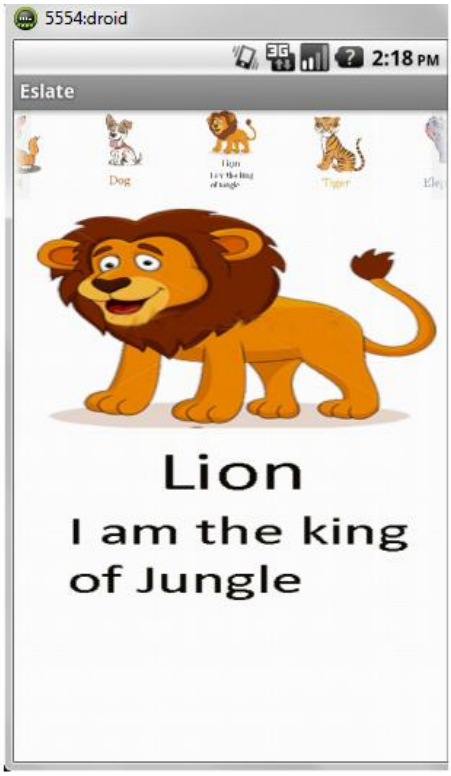



Figure 5.7.2: animals.xml

In this layout, Gallery and Image Switcher is used. Images are added up in the gallery and the user can switch from one image to another in the gallery. Animals images and their description is provided in the image switcher.

**Animals.java**

```
public class Animals extends Activity implements ViewSwitcher.ViewFactory,
            OnItemSelectedListener {
      MediaPlayer sound;
      int sounds[] = { R.raw.cat, R.raw.dog, R.raw.lion, R.raw.tiger,
R.raw.elephant, R.raw.horse, R.raw.cat, R.raw.snake, R.raw.zebra,
R.raw.monkey, R.raw.kangaroo, R.raw.gorilla };



      private Gallery gallery;
      private ImageSwitcher imageSwitcher;
      private ImageAdapter ia;
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.drawabcd);
            gallery = (Gallery) findViewById(R.id.gallery1);
            imageSwitcher = (ImageSwitcher) findViewById(R.id.is);
            imageSwitcher.setFactory(this);
      imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
                        android.R.anim.fade_in));
```

```
      imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
android.R.anim.fade_out));
            ia = new ImageAdapter(this);
            gallery.setAdapter(ia);
            // Event listener
            gallery.setOnItemSelectedListener(this);
      }
      private class ImageAdapter extends BaseAdapter {
            private Context context;
            private ImageAdapter(Context context) {
                        this.context = context;
      }
            private int[] IMAGE_IDS = { R.drawable.cat,
R.drawable.dog,R.drawable.lion, R.drawable.tiger, R.drawable.elephant,
R.drawable.horse, R.drawable.pig, R.drawable.snake,R.drawable.zebra,
R.drawable.monkey, R.drawable.kangaroo, R.drawable.gorilla };
            public int getCount() {
                  return IMAGE_IDS.length;
            }
            public Object getItem(int position) {
                  return IMAGE_IDS[position];
            }
            public long getItemId(int position) {
                  return position;
            }
            public View getView(int position, View convertView, ViewGroup
parent) {
                  ImageView iv = new ImageView(context);
                  iv.setImageResource(IMAGE_IDS[position]);
            iv.setLayoutParams(new Gallery.LayoutParams(150, 100));
                  return iv;
            }
      }
      public void onItemSelected(AdapterView<?> parent, View view, int
position,long id) {
            int imageResourceId = (Integer) ia.getItem(position);
            sound = MediaPlayer.create(Animals.this, sounds[position]);
            sound.start();
            imageSwitcher.setImageResource(imageResourceId);
      }
      public void onNothingSelected(AdapterView<?> parent) {
      }
      public View makeView() {
            ImageView i = new ImageView(this);
            i.setScaleType(ImageView.ScaleType.FIT_XY);
            return i;
      }
}
```

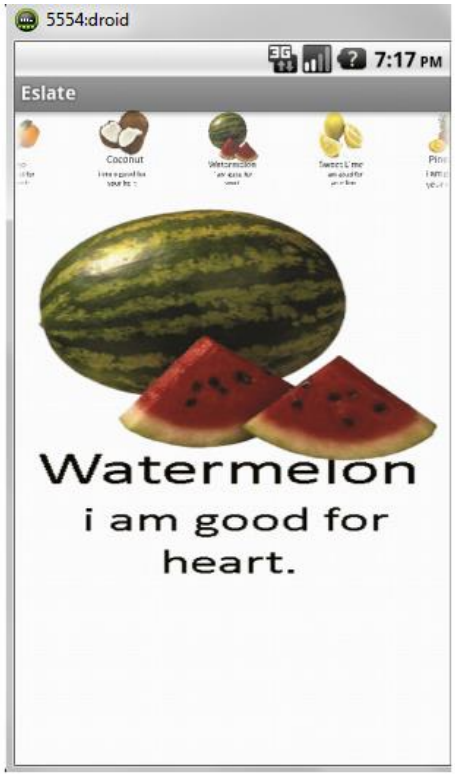



Figure 5.7.3: fruits.xml

In this layout, Gallery and Image Switcher is used. Images are added up in the gallery and the user can switch from one image to another in the gallery. Fruits images and their description is provided in the image switcher.

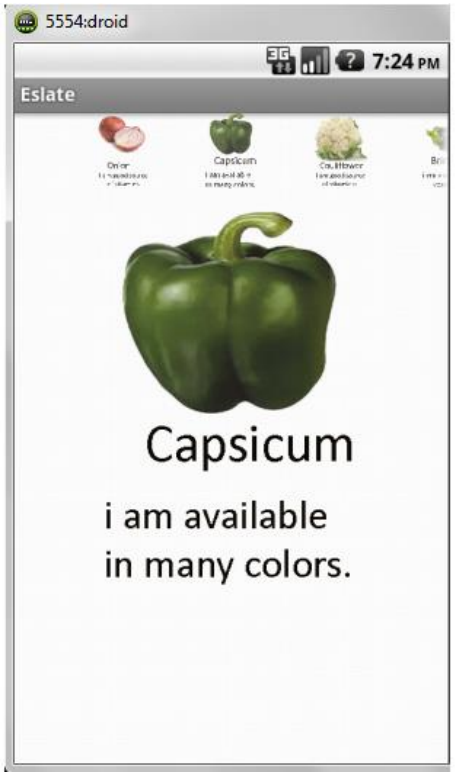



Figure 5.7.4: vegetables.xml

In this layout, Gallery and Image Switcher is used. Images are added up in the gallery and the user can switch from one image to another in the gallery. Vegetables images and their description is provided in the image switcher.


Figure 5.7.4: colors.xml

In this layout, Gallery and Image Switcher is used. Images are added up in the gallery and the user can switch from one image to another in the gallery. Colors images and their description is provided in the image switcher.

# CHAPTER-6

# SOFTWARE AND HARDWARE REQUIREMENTS

## 6.1 Software Requirement

For Development Purpose

1. JDK 1.6 or above.
2. Eclipse Juno.
3. Android SDK.

For Deployment Purpose

1. e-Slate.apk file.
2. Android OS (version 2.3 and above).

## 6.2 Hardware Requirement:

For Development Purpose

1. Processor: 600MHz
2. RAM: 256MB or more

For Deployment Purpose

1. Screen Size at least 2.5 inches.
2. Density must be at least 100dpi.

# CHAPTER-7

# CONCLUSION AND FUTURE WORK

## 7.1 Conclusion

"The kids these days are not digital kids. The digital kids were in the '90s. The kids today are mobile, and that's a difference. Digital is the old way of thinking, mobile is the new way." As parents search for ways to make the most of the travel time during road trips, the answer might be as close as their pocket. A recent study finds that mobile apps can provide an engaging, educational experience for kids. The study findings are particularly relevant as smart phones and mobile devices have become increasingly popular among families and parents are faced with a proliferation of mobile apps designed for kids. According to recent study, smart phone usage is 12% higher in households with children than other households. "Mobile apps can be a great learning tool in the hands of children." "This research is important in helping to better understand and guide the development of new apps that improve the value of children's screen time with significant educational outcomes. "Parents echo this desire for fun, yet educational, screen time, with more than half of those participating in the study saying that it is "very important" for their child to learn from media they use.

## 7.2 Future Work

The next aim in the project is to develop a classroom learning android application. The above application will have two forms, one will be master application and another will be slave application. The master application will be handled by the class teacher and the slave application by the students (kids). The connectivity between the applications can be through Wi-Fi or Bluetooth. All the slave application will be connected to the master application. There will be no connection between slave applications. The master application has the ability to give exercise to slave application and check them too. It is the class teacher using master application that will assist the student (kid) using the slave application throughout. This classroom application will be of great use to Nursery Schools.

# CHAPTER-8

# REFERENCES


[1] Hello Android by Ed Burnette
[2] http://www.idc.com
[3] http://www.learning-theories.com
[4] https://play.google.com
[5] https://itunes.apple.com
[6] http://developer.android.com
[7] http://www.gartner.com