ANDROID

Class notes

By

Mr . **Sunil** sir

ST/SU

ANDROID

RS = 110/-

→ There are some types of operations like Manual operations and operations by using Machine.

Example: for manual operation → 10+20 = 30

sending message to someone by post

Disadvantages of Manual operation:

i) Time consuming process (↑)

ii) accuracy is less (↓)

iii) efficiency is less (↓)

→ TO overcome this disadvantages some machines are introduced such a special machine as computer.

→ To perform the operations on computer some applications are needed.

↓

computer based applications

nothing but software.

→ There are some disadvantages of computer operations.

The Disadvantages are:

i) (30% - 40%) people having the computer. Out of this (18% - 20%) of people have the awareness regarding computer.

ii) It is not movable device.

iii) costly device.

iv) these applications are not reaching to every end user.

→ To overcome these disadvantages mobile device is introduced

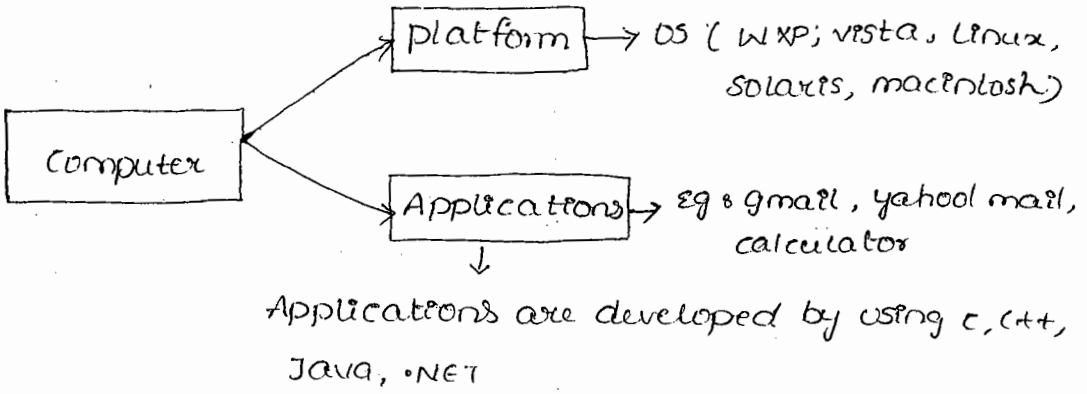
Advantages:

i) movable device

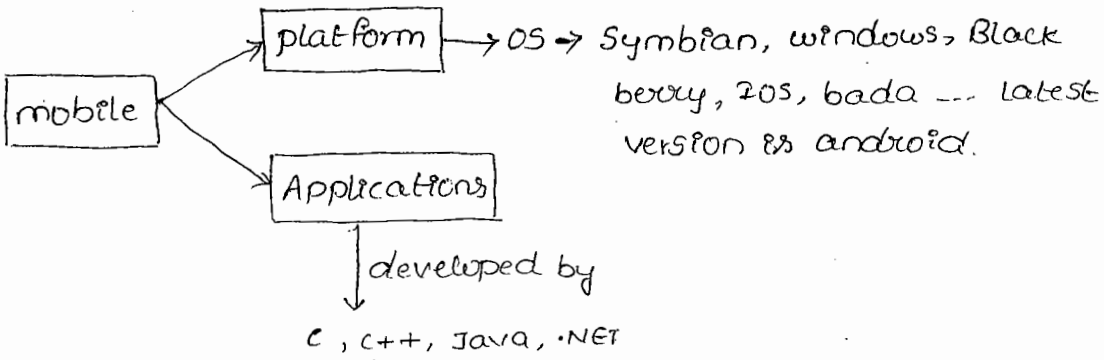
ii) more flexibility

iii) these are not costly devices.

→ To develope a computer device we need to have application and one platform (OS).

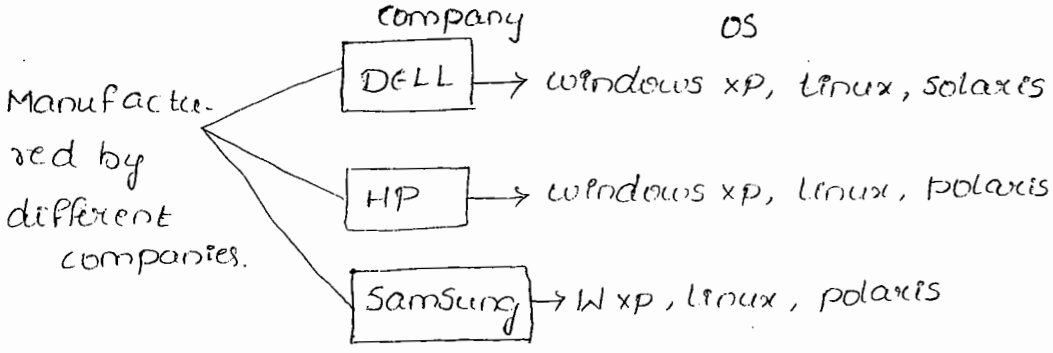


→ To develop a mobile device.

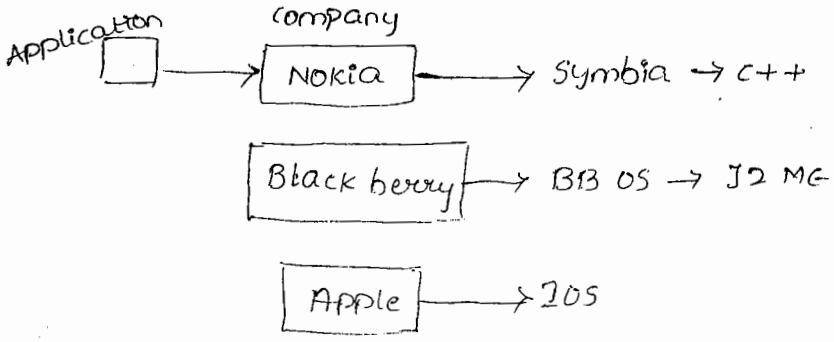


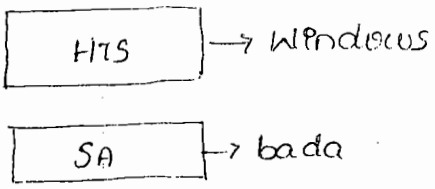
→ Android is one of the mobile operating system.

→ Regarding computers we are concentrate on application not on OS because these are device independent



→ While coming to Mobile devices, it is device dependent.





→ any application developed by symbian os are symbian based applications.

→ Other OS

1) liscenced operating systems

2) device dependent operating system

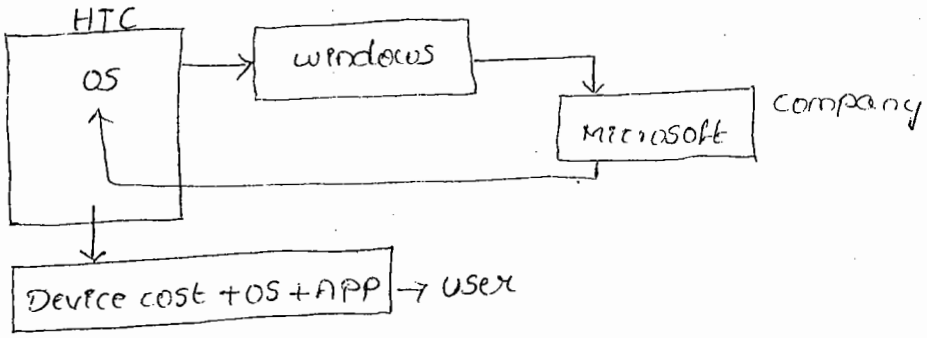
→ Android is special mobile os with 2 features.

introduced by Google

i) opensource → any mobile company directly installing without any single piece.

ii) Device endependent

→



→ Android based applications are developed by using core Java.

→ Java concepts used for developing android based applications

1) OOPs concepts

ii) basics in exception handling

iii) basics in Multithreading

iv) collection framework

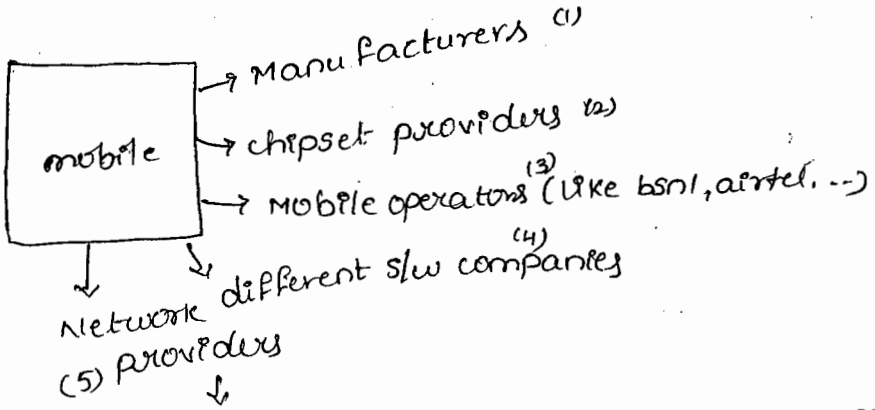
v) IO streams

History:

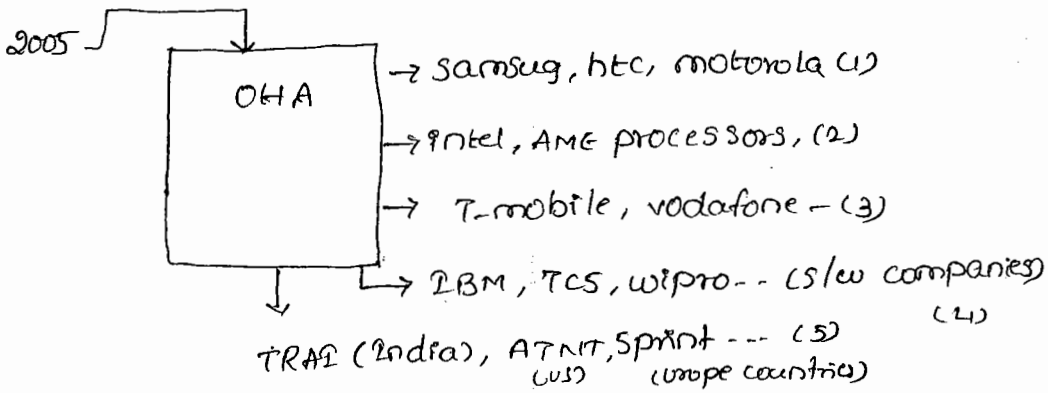
→ Android os developed by Andy Rubin by taking the support of Google.

→ Started developing in 2003 and finished in 2005

→ Google Main Intension is to introduce a new mobile with new features and less cost.



→ Google establishes a community OHA (Open handset alliances) and invites all the 5 types of companies to develope the new mobile.



→ Initially 34 companies are joined in OHA

2012 feb → 84 companies

2012 NOV → 700 companies registered officially.

→ In 2007 Google made android as openSource.

→ OHA affcially announced that android is a openSource in Aug 2007.

→ first android mobile was introduced in Aug 2008 by Google with the help of all the companies

→ and this first mobile was introduced by HTC which is named as 'HTC dream' later it is renamed as 'G1 mobile' (Google 1st mobile)

→ In 2007 : android 1.0 is introduced

android 4.0 (Icecream) ↳ nickname

android 4.0.3 (sandwitch)

2012 : android 4.1 (Jelly Bean)

→ Till now 16 versions are released in 4 categories (1.x, 2.x, 3.x, 4.x)

1.x, 2.x → developed only for mobile phones

3.x → developed for tablet pcs

4.x → both for mobiles and tablet pcs.

→ 1.x does not supports Google maps, sensors

from 2.x onwards all these are supported

4.x → 4.0 → samsung S3 is released with 4.0

→ 4.0.3

→ 4.1

→ Android OS is comprehensive os because it will give two properties after installing

i) Android OS → acting as platform to execute android applns

ii) Android API → helps in developing android applns in our own

→ Java 1.x → language

Java 2.x onwards → Technology

→ Programming language : which does not provide any planning and any implementation (pl→x , imp→x)

* We have to plan (algorithm → flowchart → code)

Eg : C, C++, Core Java

→ Technology : planning is there but no implementation
↳ 1) load driver (pl → ✓, imp → ×)
2) connection b/n prg & DB
3) statement ---
Eg : JDBC, Servlets

→ framework : planning is there & partial implementation is there (pl → ✓, imp → ✓ partial)
Eg : .Net

NOTES :

→ Android is a openSource mobile operating system, mobile can be a movable device in which mobile applications can be executed

Mobile operating system :

→ Mobile OS will acts as platform to execute mobile applications in the mobile device.

→ Mobile applications are one type of software used to perform operations in the mobile phones

Example for mobile devices :

mobile phone, tablet pc etc...

Example for mobile OS :

windows, symbian, iOS, Blackberry, android .. etc.,

Nokia → symbian
HTC → windows
BB → Blackberryos
Apple → iOS

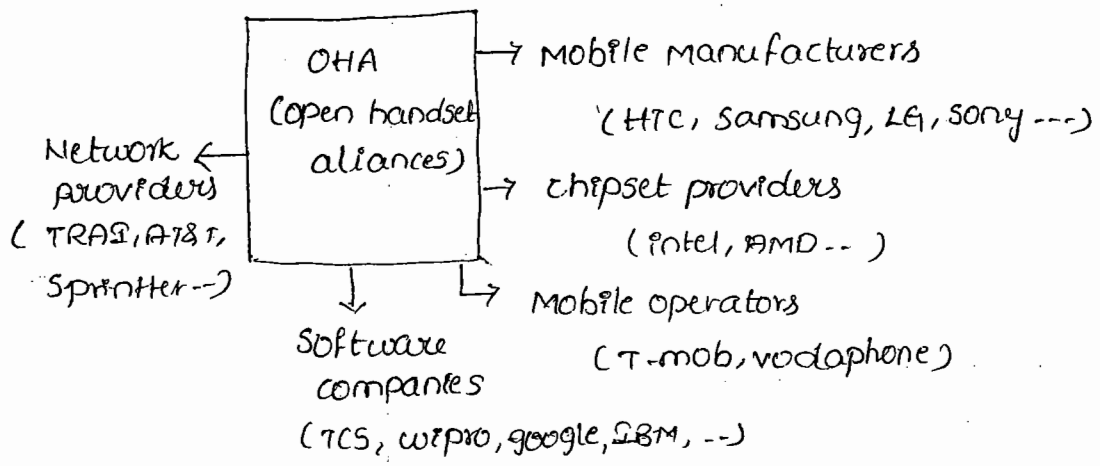
History

→ Android OS was developed by Andy rubin in the year 2005, it was take over by Google in the same year.

→ The main aim of Google is to introduce a new mobile phone in the market with a new OS and with new features for very less cost.

→ So that they have established a community called 'OHA' (Open handset alliances)

→ In OHA following five categories of companies are joined to take new revelutions in the mobile technology.



→ In August 2007 OHA announced that Android is a open source Operating System.

→ In the Aug 2008 the first android mobile was introduced by OHA, that was manufactured by HTC and that was named as 'G1 mobile'

→ From Aug 2007 to Aug 2012, 17 versions of android was released in 4 categories (1.x, 2.x, 3.x, 4.x)

1.x & 2.x → These OS is for mobile phones

3.x → for tablet pc

4.x → for both mobile phone & tablet pc.

→ Android is a comprehensive operating system that means which will give both OS & API

* OS is for executing android applns in the mobile device

* API can be helpful to the users to develope their own android applications

NOTE:

Because of provided huge API android application development is very easy for the developers and it is simply treated as 'framework'

12/11/2012

ANDROID FEATURES:

→ Android operating system supports following features.

1. It is device independent OS [This operating system can be instaled in any mobile device irrespective of manufacturer]

2. It is opensource operating system

3. Adroid provides huge API, so that as an end user we can able to develope our own applications very easily.

4. Which supports Internet tethering

* The process of using Internet service of mobile device in computer and computer device in mobile is known as Internet tethering

5. android supports android bean with NFC (Near Feild com) Technology

* Which can be used to transfer/share the data b/n two android devices just by touching them each other

6. Android Supports text to speech conversion that means all text data can be converted into voice format / command.

7. Any android device can be controlled through our own voice that means which Supports voice recognization feature and any operation can be performed in the android device just by speaking a fixed predefined command.

8. Which supports pushMail Technology that means all the latest mails will be pushed into android device automatically by taking from registered Mail account.

9. Android device Supports efficient Google Maps for better Navigation.

10. * With the help of GoogleMaps we can able to develope our own applications related to finding locations, retrieving driving directions / routes or traffic information, getting nearest restaurents, theatres etc.,

10. Which Supports Integrated webbrowser with webkit Engine (WKE), WKE always updates the current webbrowser automatically.

11. Which Supports easy webbrowsing that means Multiple web-pages can view at a time in a single window.

12. android Supports IPC (Inter process communication), which will increase the execution speed of application.

13. Android OS Supports all types of N/w's like GPS, GPRS, BT, Wi-fi, 3G, EDGE, 4G, -- etc for better comm'n purpose

3G → video chatting is possible

EDGE → advancement of GPRS

14. Android OS Supports all types of Images and audio/video files (.PNG, .3gp)

↓ these two are supported but any image can be converted into '.png' format & any format of audio/video can be converted into '.3gp' format

15. Android supports high level 2D & 3D Graphics along with multimedia very efficiently.

16. Supports Blue tooth health device technology used to ctrl any wireless device like TV, AC, car, -- etc.

17. It supports a light weight RDBMS s/w (SQLite DB) to store the data permanently and securely.

18. Android supports Multimedia operations very efficiently & also it provides a predefined API to develop our own multimedia applns (Multimedia applns are playing A|V, recording A|V)

14/11/2012

Pre-Requisites to Learn android

→ Every android application Mainly depends on presentation Layer (or) GUI Screen and Business Logic layer.

→ TO designs GUI screens 'XML' can be used

NOTE: In Android, xml supports predefined tags to design a GUI screen.

→ In the background of every GUI screen and one program is executed to perform Dynamic operations & that prgm should be written using 'CORE JAVA'

To develope an android appln the following concepts of coreJava should be known

* OOPs
* Collection framework
* Exception handling
* I/o Streams
* Multithreading

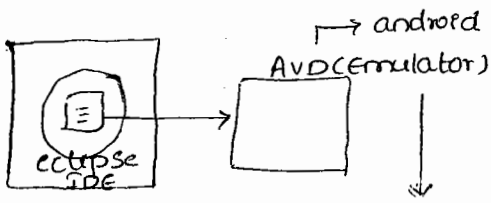
Required Softwares:

→ To Design Android applications we must setup an Environment in the computer by installing following softwares.

1. Java software (Minimum JDK 1.6)
2. Android sdk (Whatever the latest version available) →
   ↳ latest version → 20.0.3 for android API
3. Eclipse IDE (Minimum 3.5 version)
   ↳ latest version 4.2.1

→ To execute we have to install the ADT plugins in Eclipse IDE

→



To test the android applns Emulator is used

STEPS TO INSTALL SOFTWARES:

Step 1:

Install JDK6 software by downloading from following website.

www.oracle.com/technetwork/java/javase/downloads/index.html

Step 2:

Install android sdk by downloading from following location

developer.android.com/sdk/index.html

Step 3:

Extract the eclipse in any location of computer by downloading from following website.

www.eclipse.org/downloads (Eclipse classic 4.2.1)

Installing ADT (Android development tool) plugins in Eclipse IDE:

The location where the android applns resides that we develop using Eclipse IDE

↑

Step 1:

Open Eclipse IDE and set Workspace

Step 2:

Select Help Tool in Eclipse → select install new software option.

Step 3:

Click add button in the new window → provide name & Location as shown below.

Name: adtplugins (can be any userdefined name)

Location: https://dl-ssl.google.com/android/eclipse

→ click OK.

Step 4:

Select ☑ developer tools and ☑ NDK plugins → click next button

Step 5:

Check all the downloaded ADT plugins & click Next Button → ☑ accept liscense agreement → click finish button.

Step 6:

Installation of ADT plugins will be started and restart the Eclipse once the process is completed

15/11/2012

Version	API level
1.0	1
1.1	2
!	!
!	!
!	!
4.1	16
4.2	17

1.x

2.x → 2.0

→ 2.3.x → 2.3.3

2.3.5

2.3.6

3.x 2.3.7

4.x → 4.0

→ 4.0.3

→ 4.1 → released in Aug 2012

↘ 4.2

2.3.3 → 10 onwards Googlemaps are introduced

Steps to Install android API

→ API (Application programming Interface) is collection of predefined classes and interfaces used to perform some predefined operations in android Application.

Step1 :

Select window option in eclipse IDE → select preferences → Select android option in the left panel of new window → browse and select Android Sdk home directory in the right panel of same window → click Apply → click OK.

Step2 :

Select android-sdk Manager in window option.

→ Select Tools, extras and any API level (recommended levels are 10-17)

Working with AVD Manager (Emulator)

AVD (Android virtual Device) (or) Emulator is a logical android device can be used to test the android applications before launching in the real devices.

→ Select window option → select AVD Manager → click new button to create a new Emulator →

AVD Name : [android 2.3.3 GM] → any valid user defined name (user choice)
→ condition is spaces are not allowed.

Device : [any one device ▽] → Select type of Emulator

target : [▽] → Select API level

→ [OK] → click

→ Select any API supported AVD and click start → click Launch [to start the emulator]

NOTE:

We can use either edit (or) delete button either to update the details or to delete the emulator respectively.

Setting paths:

→ It is very mandatory to set path for both Java and android.

Select (Right click) on my computer → select option properties → select advanced → select env variables → select system variables → select path → click edit → variable name: path (by default it is available)

path: ---- ;.; C:/prog files/Java/jdk1.6/bin;.;
↗ Java path

G:/android 6/android-sdk/tools;.;

G:/android 6/android-sdk/platform-tools;.;
↓ android path

→ click OK → OK → OK

NOTE:

Restart the computer once if all the s/w's are installed and setting the paths (set these paths properly)

16/11/2012

Creation of first android application

Step1:

Open Eclipse IDE → select file option → select New option → select project → select Android → select android application project → click Next button

Step 2:

provide valid application name and package name [package name specification is "com.organisationname.application name"] →
↓ company

click Next button

Step 3:

Select ☑ create custom launcher icon
☑ create activity
☑ create project in workspace
} options → click Next button

Step 4:

browse / select any user-defined image as an application icon [default image will be taken if no icon is selected] → click Next button.

Step 5:

Select ☑ create activity → select blank activity → click next

Step 6:

provide a valid activity name, layout file name → click finish button.

Directory Structure of android application

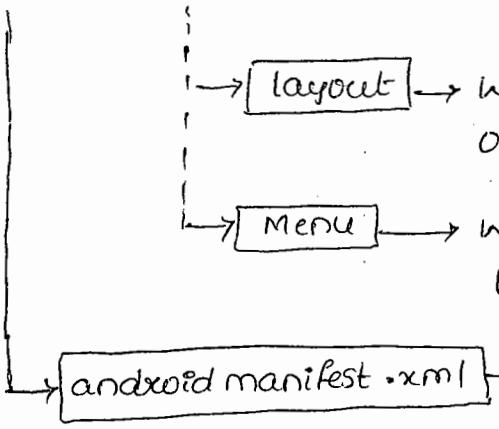
→ If any android application is created automatically some folders and files will be created. these are used to hold different resources which are used to develope and execute android application.

17/11/2012

Application

- src → Which contains all activity files (business logic code in the form of java files)
- Gen → This is autogenerated folder which contains 'R.java' file, all the resources like images, text files, audio/video files etc will be registered automatically whenever those are added to the application

  NOTE: "R.java" file should not be modified.
- Android 4.1 → Which contains all library files in the form of '.jar' file.
- android dependencies → Which contains advanced api to develope advanced android applns.
- assets → Which contains any type of text formatted files which are going to be used in current android applns like '.txt', '.doc', '.pdf' files .. etc.,
- bin → Which contains android executable file (.apk)
  - ↳ android package
- libs → Which contains inbuilt Native Libraries (which are developed in C (or) C++)
- res
  - drawable.hdpi → Which contains high density pixel imgs which are going to be used in the current android appln
  - drawable.ldpi → Which contains low density pixel images



→ all audio (or) video files should be kept in "raw" folder within res, raw folder should be created with the following steps.

rightclick on res → new (select) → select folder → provide name as 'raw'

→ If developer don't know the type of image then that should be kept in res/drawable folder (drawable should be created explicitly)

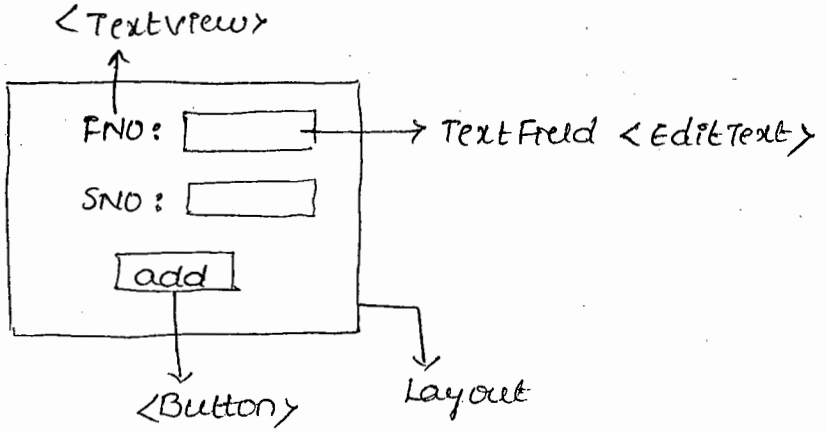
→ All resources are registered automatically in the R.java file in the form of hexadecimal code.

imagefiles abc.png
xyz.png
audio file a.mp3

R.java

```
class R
{
  P S F class drawable
    {
      P S F int xyz = 0x---;   (hdc)
      P S F int abc = 0x--;    (hdc)
    }
  P S F class raw
    {
      P S F int a = 0x--;      (hdc)
    }
}
```

Android Application Development components:



→ Every android application mainly depends on following components.

1. Uncompiled resources
2. Layout files
3. Activity class
4. Android Manifest file.

1. Uncompiled Resources:

→ Before starting development of any android application it is very mandatory to add the resources (like images, audio (or) video files, Text files etc) to the application.

→ Resource is a common property which is used in the application development

resource		folder
Images	should be added to →	drawable
audio/video	→	raw
text files	→	assets

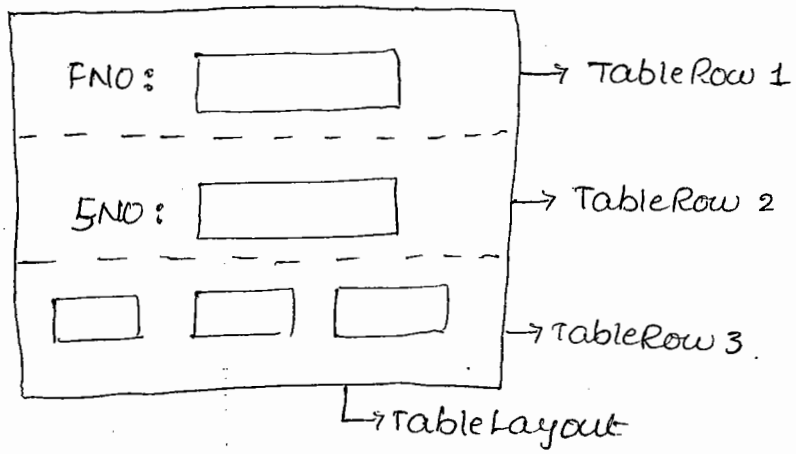
2. Layout Files

→ Layout Files acts as 'GUI' screens from which end user either can provide the inputs or can view the outputs

→ Layout files can be designed using XML

→ In android api predefined tags are available to design various view components like Textbox, button, --etc., in Layout File.

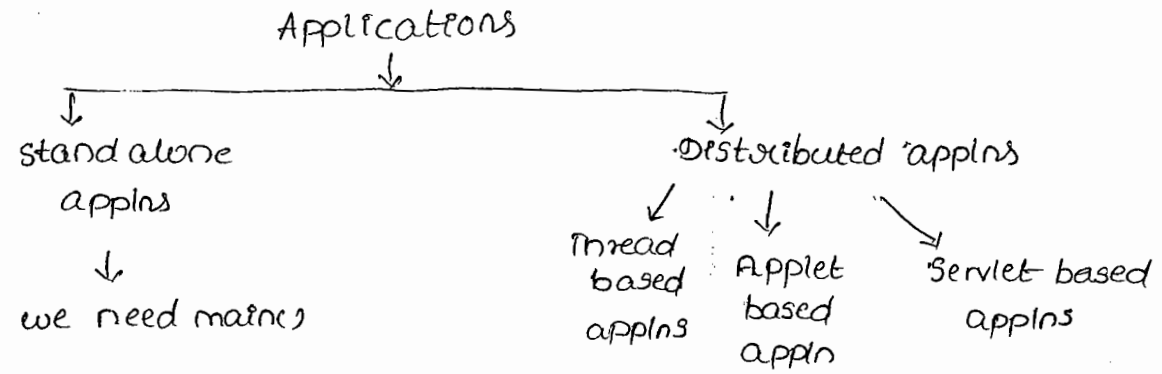
Example :




```
<TableLayout>
    <TableRow>
        <Textview android:text = "FNO"
                  android: layout_width = "20px"
                  android: layout-hight = "10px"
                  android: text color = "#ff0000"
                  android: textSize = "25px"/>
        <EditText android: layout-width = "10px"
                  android: layout_height = "20px"
                  android: ----
                  ------ />
    </TableRow>
```

```
<TableRow>
    <TextView
        —
        —
        — />
    <EditText
        —
        —
        —/>
</TableRow>
<TableRow>
    <Button
        —
        —
        — />
    <Button
        —
        —/>
    <Button
        —
        —/>
</TableRow>
</TableLayout>
```

NOTE:



→ In android we will not use main() any where. we will only use the "Life cycle methods (LCM)"

22/11/2012

③ Activity classes:

→ Activity is a predefined class in Android API, can be used to load a "layout file" in Android devices and also which contains logic to perform an operation in the "GUI screen"

→ Developer can create his own "userdefined Activity" not only to load the "userdefined GUI screen", but also to write the "userdefined Business logic".

Rules to create userdefined Activity class:

Step1:
Declare any userdefined class and extends predefined Activity class.

Step2:
override onCreate() in the derived class, it is the starting executable method of any Activity class.

Step3:
We must call superclass onCreate() within the derived class onCreate() to initialize Activity class.

Step4:
Load any userdefined layout file using "setContentView()"

Step5:
Save the program with. Userdefined activity class name.java

Syntax:

```
class <UCN> extends Activity
{
    @override
    P V onCreate(Bundle objref)
    {
      super.onCreate(objref);
      setContentView(R.layout.layoutfile name);
    }
}
```

(UCN ← Userdefined class name; objref ← any userdefined name; layoutfile name ← a)

Business logic to perform operations.

a.xml

NOTE: No activityclass should contain main().

→ Every Layout file should contain an Activity class.

23/11/2012

DVM → Darvic virtual Machine

It is the main property to execute the Android applns.

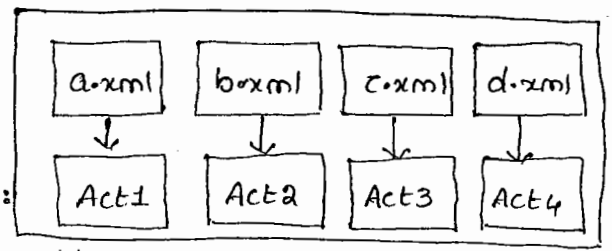
→ Executes only Java class files

Act1.java

Android Manifest (Manifest.xml):

Android Manifest is the starting Executable file by the "DVM", in which all Activities which are going to be used in the Android Application should be registered.

Syntax:




```
<xml ---->
  <manifest>
    <application android:
        icon = "@drawable/imagename".
        android:label = "appname" >
      <activity android:name = ".Act1" >
                  ↳ Activity class name
        <intent-filter>
          <action android:name = "android.intent.action.A"/>
          <category android:name = "android.intent.category.
                                    LAUNCHER"/>
```

```
        </intent-filter>
    </activity>
    <activity android:name=".Act2">
        <intent-filter>
            <action android:name="android.intent.action.B"/>
            <category android:name="android.intent.category.DEFAULT"/>
        </intent-filter>
    </activity>
    <activity android:name=".Act3"/>
    <activity android:name=".Act4"/>
    </application>
</manifest>
```

→ Activities can be categorised into 2 types.

1. LAUNCHER :

Which will be loaded as a first activity, among multiple activities of Android Application.

2. DEFAULT:

If any activity is loaded as a second or more than second optional activity. Then that should be declared as default category.

NOTE:

For "LAUNCHER" activity it is very mandatory to write Both the Action and category. whereas for "DEFAULT" activity, these are optional.

→ "Intent-filter" can be used to achieve the switching between Activity and layout.

Naming conventions:

→ While developing an Android application it is very mandatory to follow naming conventions.

① Every package name should exists in Lowercase

Ex: package com.satya.demoapp → creation of package
import android.app.* → importing package properties

(2) First letter of every word of class name (or) interface name should exists in uppercase

Example:

```
class StudentDetails
{
  ---
  ---
}
```

```
interface FacultyDetails
{
  ---
  ---
}
```

(3) For DataMember (DM), method and object first letter of first word should exist in lowercase, but from the next word the first letter should exist in uppercase

Example:

```
Class StudentDetails
{
  int sno;
  String studentName;        } → Datamembers (or) variables
  void studentCourseDetails() → method
  {
    ---
    ---
    ---
  }
}
```

StudentDetails firstStudent = new StudentDetails();

(4) Every constant DataMember should exist in uppercase, if it is having more than one word, that should be seperated with '_' (underscore)

Eg: final String COLLEGE_NAME = "abcd";

→ src → activity class

→ res → layout

Q. How to run android application?

Right click on Displaymsg → then run as → Android application.

24/11/2012 Android Application:

first.xml
Welcome to Android session

Step1:
Create android application.

Step2:
add an image to the drawable (res folder)

Step3:
Design layout file (first.xml)

```
<!--xml -->
<LinearLayout xmlns:android = "_____"
              android:Layout-width = "fill-parent"
              android:Layout-height = "fill-parent"
              android:background = "@drawable/bg">
<TextView android:text = "welcome to Android session"
          android:textcolor = "#ff0000"
          android:textSize = "10px"
          android:Layout-height = "wrap-content"
          android:Layout-width = "wrap-content"/>
```

Step4:
Activity class

```
class Act extends Activity
{
  @override
  Public void onCreate(Bundle SavedInstanceState)
  {
   super.onCreate(SavedInstanceState);
   setContentView(R.layout.first);
  }
}
```

Step5:
"Manifest.xml"

```
<xml -->
<manifest xmlns:android = "    ">
```

```
<application android:title = "     "
             android:icon = "@drawable/Imagename" />
  <activity android:name = ".Act1" />
    <intent-filter>
      <action android:name = "android.intent.action.
                              FIRST" />
      <category android:name = "android.intent.category.
                              LAUNCHER" />
    </intent-filter>
  </activity>
</application>
```

Red : 256
Green : 256
Blue : 256
} 0 - 255

Red	00-ff	Red → ff0000
Green	00-ff	Green → 00ff00
Blue	00-ff	Blue → 0000ff

fill-parent

complete layout will be filled with width and height

Wrap-content

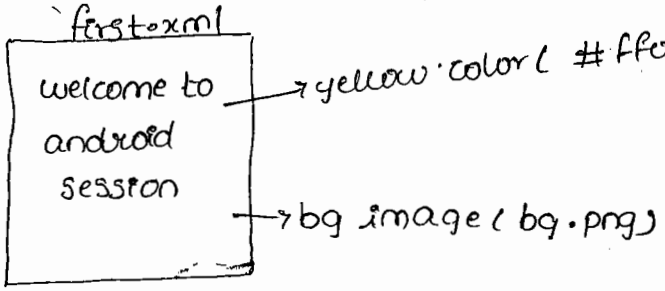
only height and width of the given data will be filled.

** → After completion of 3 programs, go to filename and Rightclick on RUN AS → android application.

The android application will be launched.

26/11/2012

Design an Android application to display an userdefined message as shown below.



Step1:

Create any android application with any userdefined name.

Step2: Add bg.png (Image resource) to the RES/drawable folder

Step3: Design layout file (Any userdefined name.xml)

↓

first.xml

```
<xml →
<LinearLayout xmlns:android="http://schemas.android.
                com/apk/res/android"
  android:layout-width="f-p"
  android:layout-height="fill-parent"
  android:background="@drawable/bg">
  <Textview android:layout-width="wrap-content"
            android:layout-height="wrap-content"
            android:text="welcome to android
                          Session"
            android:textSize="25px"
            android:textColor="#ffff00"/>
</LinearLayout>
```

Step4:

Design Activity class (Act1.java)

```
Package com.sathya.displaymsg;
import android.app.activity;
import android.os.Bundle;
Public class Act1 extends Activity
{
  Super.onCreate(SavedInstanceState);
    @override
  Protected void onCreate(Bundle savedInstanceState)
  {
    ---
    setContentView(R.layout.first();
  }
}
```

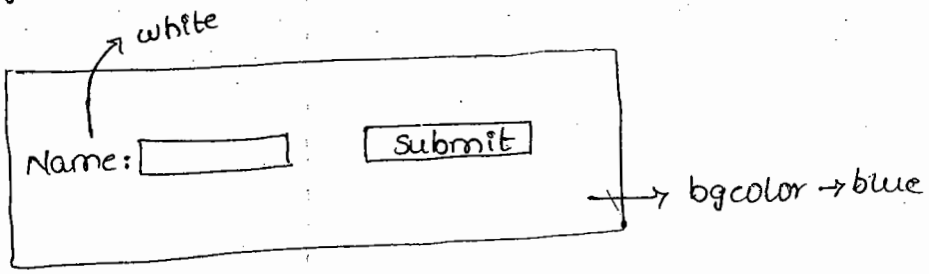
Step5:

Design Manifest file (Android manifest.xml)

```
<?xml version="1.0" encoding="utf-8"?> [] <manifest
xmlns:android="http://schemas.android.com/apk/res/
                                android"
    Package="com.sathya.displaymsg"
    android:versionCode="1"
    android:versionName="1.0">
  <user-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16"/>
  <application
        android:icon="@drawable/ic_launcher"
        android:label="MyDisplayApp">
    <activity
        android:name=".Act1">
      <intent-filter>
        <action android:name="android.intent.action.
                                FIRST"/>
        <category android:name="android.intent.cate-
                        gory.LAUNCHER"/>
      </intent-filter>
    </activity>
  </application>
</manifest>
```

→ The launcher file name can be given as MAIN (or) filename (FIRST)

Design an Android application as shown below.



Step1: Create android application.

Step2: Design layout file

first.xml

```
<xml —>
<linearlayout xmlns:android= ——
              xmlns:tools= ——
              a:l-w="f-p"
              a:l-h="f-p"
              a:background="#0000ff"
              a:orientation="horizontal">
  <Textview a:text="Name"
            a:l-w="w-c"
            a:l-h="w-c"
            a:textcolor="#ffffff"
            a:textsize="20px" />
  <EditText a:layout-width="w-c"
            a:l-h="w-c"
            a:text="Submit" />
```

Step3:

Design Activity class

Act1.java

```
class Act2 extends Activity
{
@override
  P v onCreate(Bundle b)
  {
    super.onCreate(b);
    SetContentView(R.layout.first);
} }
```

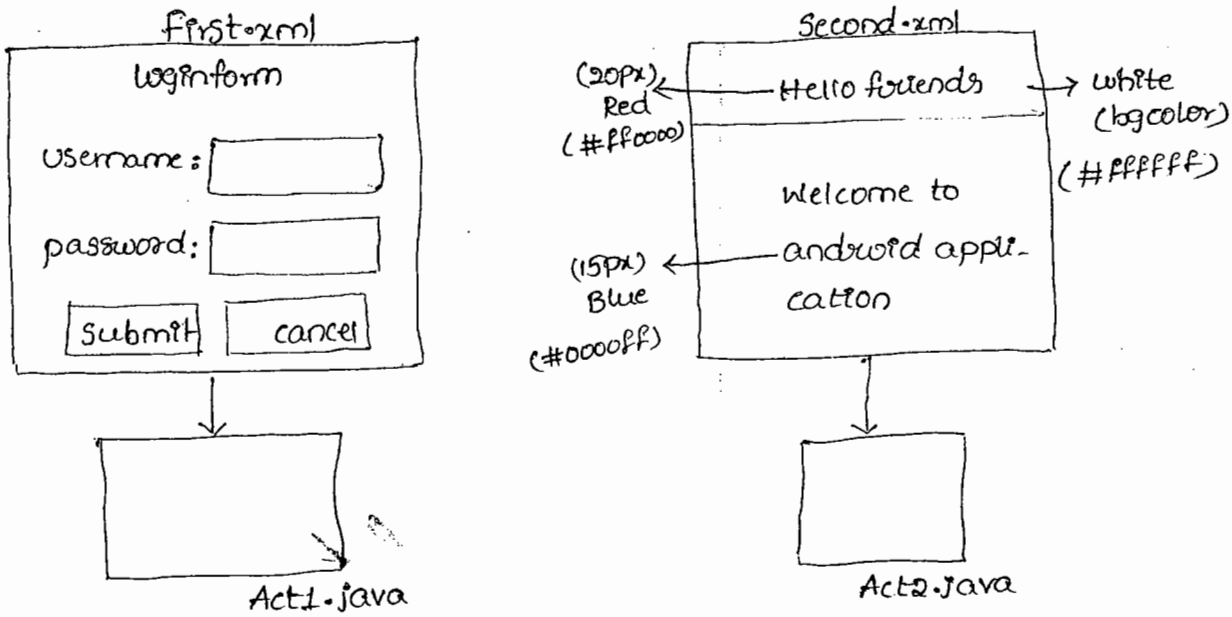
Step4:

Design Manifest.xml

```
<xml->
<manifest>
  <application
        android : icon = " ____ "
        android : label = " ____ "
  <activity a:name = " .Act2 " >
   <intent-filter>
     <action a:name = " android.intent.action.SECOND "/>
     <category a: name = "android.intent.category.LAUNCHER"/>
   </intent-filter>
  </activity>
</application>
</manifest>
```

27/11/2012

Design an android application to display the following.



Step1:

Create an android application.

Step2:

Design layout files.

first.xml

```
<xml ---->
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="vertical">
<Textview a:l-h="w-c"
          a:l-w="w-c"
          a:text="Loginform" />
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="horizontal">
<Textview a:text="username"
          a:l-w="w-c"
          a:l-h="w-c" />
<Edittext a:l-h="f-p"
          a:l-w="f-p" />
</Linear layout>
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="horizontal">
<Textview a:text="password"
          a:l-h="w-c"
          a:l-w="w-c" />
<EditText a:l-h="w-c"
          a:l-w="w-c" />
</LinearLayout>
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="horizontal">
```

```
<Button a:text="submit"
        a:l-w="w-c"
        a:l-h="w-c" />
<Button a:text="cancel"
        a:l-w="w-c"
        a:l-h="w-c" />
</LinearLayout>
```

NOTE: </LinearLayout>

→ To create second.xml file right click on layout → new → other → Android xml file.

→ for activity classes, right click on src.

Second.xml:

```
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="vertical"
              a:background="#ffffff">
  <Textview a:text="Hello friends"
            a:l-h="w-c"
            a:l-w="w-c"
            a:textcolor="#ff0000"
            a:textsize="20px" />
  <Textview a:text="welcome to android application"
            a:l-h="w-c"
            a:l-w="w-c"
            a:textcolor="#0000ff"
            a:textsize="15px" />
</Linear Layout>
```

Step 3:

design activity classes.

Act1.java

```
Public class Act1 extends Activity
  {
    @override
      Public void onCreate(Bundle b)
        {
          super.onCreate(b);
          setContentView(R.layout.first);
        }
  }
```

Act2.java

```
  Public class Act2 extends Activity
    {
      @override
        public void onCreate(Bundle b)
        {
          super.onCreate(b);
          setContentView(R.layout.second);
        }
    }
```

Step 4:

Design Android manifest.xml

manifest.xml

```
<manifest>
  <application a:icon="__"
               a:label="__">
  <activity a:name=".Act1">
   <i-f>
    <action a:name="android.intent.action.FIRST"/>
   <category a:name="android.intent.category.LAUNCHER"/>
   </i-f>
  </activity>
```

```
<activity a:name=".Act2">
    <i-f>
     <action a:name="android.intent.action.MAIN SECOND"/>
      category
     <action a:name="android.intent.category.LAUNCHER"/>

    </i-f>
  </activity>
 </application>
</manifest>
```

NOTE:

We can create Multiple activity classes and layout files with the following syntax.

a) for layout file

right click on res/Layout folder → select new → select others → select android xml file → provide any userdefined name.

b) for activity class

right click on src/package folder → select new → select class → provide any userdefined name.

NOTE:

The Method which is returning the object of same class that is called factory method.

28/11/2012

Toast Message:

NOTE: Typecasting

→ (datatype) variablename.

→ (double
  String
  float
  button) a
  ↳ variable

→ It is one type of dialogue message used to give some notification to the enduser.

→ Toast message will be displayed automatically and disappeared automatically after some specific duration.

→ To display the Toast message a predefined class of android API called 'Toast' can be used.

Syntax:

```
Toast t = Toast.makeText(context, "msg", duration);
    t.show();
        (OR)
Toast.makeText(context, "msg", duration).show();
```

In the above syntax,

→ MakeText() method can be used to design the Toast message in a container box and

→ show() method can be used to display Toast message.

→ Context : always represents current activity, can be identified by 'this'.

→ msg : can be any userdefined message.

→ duration : can be in ms (Eg : 3000, ...)

(OR)

Toast.LENGTH_LONG → used to display toast msg for long time

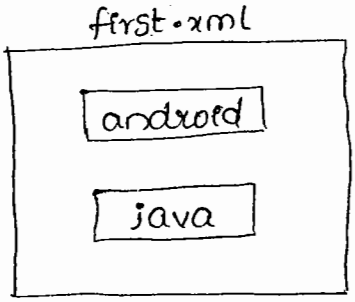
Toast.LENGTH_SHORT → used to display toast msg for shorter period of time.

NOTE:

LENGTH_LONG and LENGTH_SHORT are static. So these are called by classname(Toast)

29/11/2012

Design an android application to display toast msg as shown below.



a) If android button is clicked the toast msg should be displayed as "android was developed by andy rubin"

b) If Java button is clicked the toast msg should be displayed as "Java was developed by jamesgosling"

Step1:

create android application

Step2:

design Layout file

first.xml

```
<LL  a:l-w="f-p"
     a:l-h="f-p"
     a:orientation="vertical>
  <Button  a:text="android"
           a:l-w="w-c"
           a:l-h="w-c"
           a:id="@+id/bt1"
           a:onClick="androidbtn"/>
  <Button  a:text="Java"
           a:l-w="w-c"
           a:l-h="w-c"
           a:id="@+id/bt2"
           a:onClick="javabtn"/>
</LL>
```

Step3:

Design activity classes.

Act1.java

```
Public class Act1 extends Activity
  {
    Button b1, b2;
    @override
    P c onCreate (Bundle b)
    {
      Super.onCreate(b);
      SetContentView (R.Layout.first)
       b1 =(Button) findViewById (R.id.bt1);
       b2 = (Button) findview By Id(R.id.bt2);
     }
     P v android btn(view v)
      {
       Toast.makeText (this, "android was developed by
              andy rubin", 3000).show();
       }
     P v Javabtn(view v)
      {
      Toast.makeText(this, "Java was developed by
         James gosling", 5000).show();
       }
  }
```

Step4:

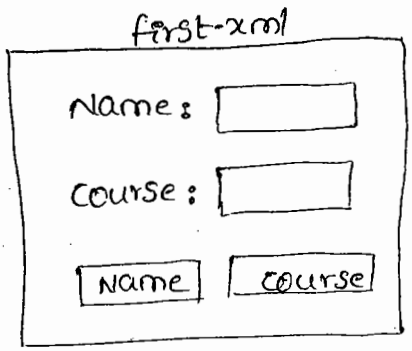
Design Manifest file.

manifest.xml

```
<manifest>
 <application>
```

```
<activity a:name = ".Act1" >
    <intent-filter>
        <action a:name = "android.intent.action.MAIN"/>
        <category a:name = "android.intent.category.LAUNCHER"/>
    </i-f>
</activity>
</application>
</manifest>
```

Design an android application as shown below.



a) if Name button is clicked, Name of the student should be displayed in the form of toast msg.

b) if course button is clicked, the course which is pursued by student should be displayed.

Step1:

Create android application

Step2:

design layout file.

first.xml

```
<LL  a:l-h = "f-p"
     a:l-w = "f-p"
     a:orientation = "vertical" >
<ll  a:l-h = "f-p"
     a:l-w = "f-p"
     a:orientation = "horizontal" >
```

```
<TextView a:text="name"
          a:l-w="w-c"
          a:l-h="w-c"/>
<EditText a:l-w="w-c"
          a:l-h="w-c"
          a:id="@+id/et1"/>
</LL>
<LL a:l-w="f-p"
   a:l-h="f-p"
   a:orientation="horizontal">
 <TextView a:text="course"
           a:l-w="w-c"
           a:l-h="w-c"/>
 <EditText a:l-w="w-c"
           a:l-h="w-c"
           a:id="@+id/et2"/>
</LL>
<LL a:l-w="f-p"
   a:l-h="f-p"
   a:orientation="horizontal">
  <Button a:l-w="w-c"
          a:l-h="w-c"
          a:onClick="namebtn"/>
  <Button a:l-w="w-c"
          a:l-h="w-c"
          a:onClick="coursebtn"/>
 </LL>
</LL>
```

Step3:

Design Activity class

Act1.java

```
P C Act1 extends Activity
  {
```

```
@Override
P v onCreate(Bundle b)
{
  super.onCreate(b);
  setContentView(R.layout.first);
}
P v Namebtn()
{
EditText et1 = (EditText) findViewById(R.id.et1);
  String s1 = et1.getText().toString();
Toast t1 = Toast.makeText(this, "your name is " + s1, 5000);
  t1.show();
}
P v coursebtn()
{
EditText et2 = (EditText) findViewById(R.id.et2);
  String s2 = et2.getText().toString();
Toast t2 = Toast.makeText(this, "your course is " + s2,
                                                  5000);
  t2.show();
}
}
```

Step 5:

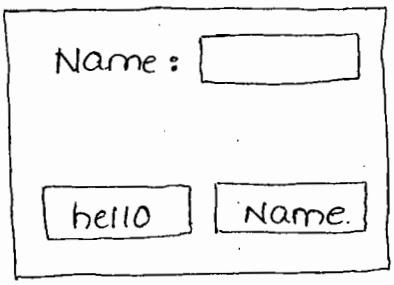
develope android manifest file.

manifest.xml

```
<manifest>
  <application>
    <activity a:name = ".Act1"><i-f>
      <action a:name = "android.intent.action.MAIN"/>
      <category a:name = "android.intent.category.LAUN
                                                  CHER"/>
  </i-f></activity>
  </application>
</manifest>
```

30/11/2012

Design an android application as shown below.



a) If name button is clicked display the toast msg as "Hello (name that is you entered)".

b) If hello button is clicked then display the toast msg "Hello friend".

Step1 :

Create android application.

Step2 :

design Layout file.

first.xml

```
<LinearLayout  a:l-w = "f-p"
               a:l-h = "f-p"
               a:orientation = "vertical" >

   <LL  a:l-w = "f-p"
        a:l-h = "f-p"
        a:orientation = "horizontal" >

      <TextView  a:l-w = "w-c"
                 a:l-h = "w-c"
                 a:text = "Name" />

      <EditText  a:l-w = "w-c"
                 a:l-h = "w-c"
                 a:id = "@+id/et1" />

   </LL>
```

```
<LinearLayout a:l-h="fill_p"
              a:l-w="fill_p"
              a:orientation="horizontal">
  <Button a:l-h="wrap_c"
          a:l-w="wrap_c"
          a:orientation="
          a:onClick="Hellobtn"/>
  <Button a:l-h="wrap_c"
          a:l-w="wrap_c"
          a:onClick="namebtn"/>
</LinearLayout>
</LinearLayout>
```

Step3:

Create Activity class

Act1.java

```
public class Act1 extends Activity
{
  @override
  Public void onCreate(Bundle b)
  {
    super.onCreate(b);
    setContentView(R.layout.first);
  }
  public void Hellobtn(view v)
  {
  Toast t = toast.makeText(this, "Hello friend",
                                        5000);
    t.show();
  }
```

```
p v namebtn(view v)
 {
   EditText et=(EditText) findviewById(R.id.et1)
       string s=et.getText().toString();
    Toast t=Toast.makeText(this," your name is "+s,
                                        5000);
       t.show();
  }
}
```

Step 4:

Design android manifest.xml

```
<manifest> <application>
        < activity a:name=".Act1">
          <intent-filter>
          <action a:name="android.intent.action.MAIN"/>
          <category android:name="android.intent.category.
                                        LAUNCHER"/>
          </intent-filter>
        </activity>
      </application>
</manifest>
```

Intent:

→ Intent is a predefined class in "android.content" package used to achieve the switching from one activity to another activity.

→ In android switching can be done in two different ways

1) Explicit Switching
2) Implicit Switching

1) Explicit Switching:

Whenever switching is done by the end user explicitly is known as Explicit Switching, it can be done in three different ways.

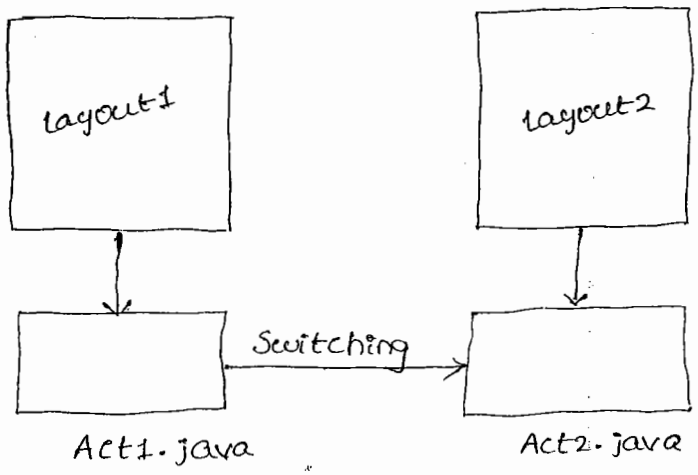
i) Normal Switching
ii) Switching with Data
iii) Switching with result data

i) Normal Switching:

Whenever switching is done between two activities without sending any data is called as normal switching.

Note:

In general switching always done between layout files in the form of activity classes.



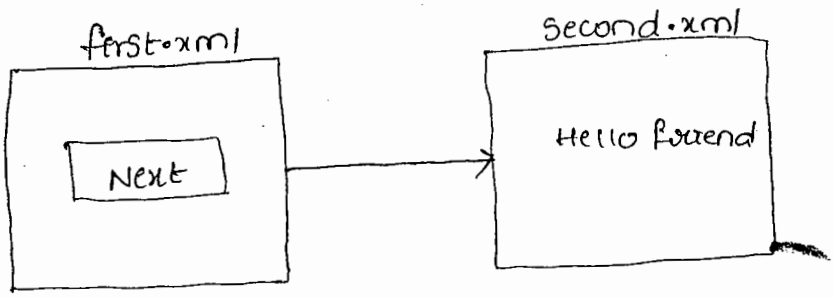
Syntax to achieve Normal Switching :

```
Intent objref = new Intent(SourceActivity, DestinationActivity);
StartActivity(objref);
```

→ In the above syntax source Activity can be always current Activity and it is represented with "this"

→ Destination Activity can be any other secondary Activity and that must be represented "Activity classname.class" format

Design an android application to achieve the following.



Step 1 :

Create android application.

Step 2 :

Design Layout files.

first.xml

```
<linearlayout
    xmlns : android = "http://schemas.android.com/apk/
                       res/android"
    xmlns : tools = "http://schemas.android.com/tools"
      android : Layout_width = "fill_parent"
      android : Layout_height = "fill_parent"
      android: orientation = optional>
  <Button a: text = "Next"
          a : l-w = "w-c"
          a : l-h = "w-c"
          a : onClick = "Nextbtn" />
</linearLayout>
```

Second.xml

```
<LL  xmlns:--
     xmlns:--
     a:l-h="f-p"
     a:l-w="f-p">
<Textview  a:text="Hello friend"
           a:l-w="w-c"
           a:l-h="w-c"
           a:textSize="20px"
           a:textColor="#ff0000"/>
</LL>
```

Step3:

design Activity classes.

Act1.java

```
Public class Act1 extends Activity
 {
  @override
   Public void onCreate(Bundle b)
    {
     Super.onCreate(b);
     SetContentView(R.layout.first);
    }
   Public void Nextbtn(View V)
    {
     Intent i=new Intent(this, Act2.class);
      StartActivity(i);
    }
 }
```

Act2.java

```
Public class Act2 extends Activity
 {
```

```
@Override
Public void onCreate(Bundle b)
{
Super.onCreate(b)
SetContentView(R.layout.second);
}
}
```

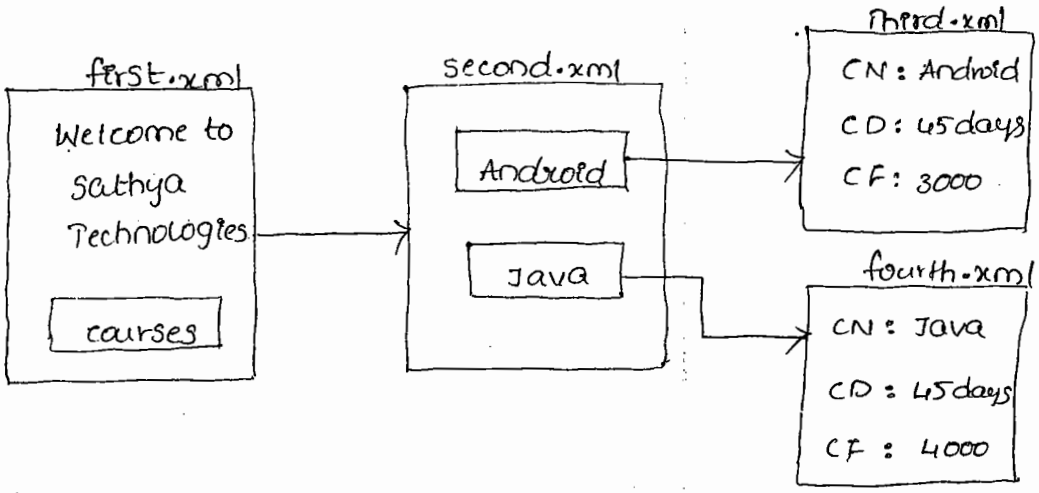
Step 4:

design android manifest.xml

manifest.xml

```
<manifest>
<application>
<activity a:name=".Act1">
<intent-filter>
<action a:name="android.intent.action.MAIN"/>
<category a:name="android.intent.category.
LAUNCHER"/>
</intent-filter>
</activity>
<activity a:name=".Act2">
<intent-filter>
<action a:name="android.intent.action.
SECOND"/>
<category a:name="android.intent.category.
DEFAULT"/>
</intent-filter>
</activity>
</application>
</manifest>
```

Design an android application to fullfil following requirements.



Step1:

Create android application.

Step2:

Design Layout files.

first.xml

```
<LL ---
    --
    a: l-h = "f-p"
    a: l-w = "f-p"
    a: orientation = "vertical" >
<Textview a: text = "welcome to sathya Technologies"
          a: l-h = "w-c"
          a: l-w = "w-c"
          a: textsize = "20px"
          a: textcolor = "#ff0000" >
< Button a: text = "courses"
          a: l-h = "w-c"
          a: l-w = "w-c"
          a: onclick = "coursebtn" >
</LL>
```

Second.xml

```
<LL ---
    --
    a: orientation = "vertical" >
```

```
<Button a:text="android"
        a:l-h="w-c"
        a:l-w="w-c"
        a:onClick="Androidbtn"/>
<Button a:text="Java"
        a:l-h="w-c"
        a:l-w="w-c"
        a:onClick="Javabtn"/>
</linearlayout>
```

Third.xml

```
<linearlayout ----
              ----
              a:orientation="vertical">
<Textview a:text="CName:android"
          a:l-h="w-c"
          a:l-w="w-c"
          a:textSize="20px"
          a:textcolor="#ff0000"/>
<Textview a:text="CDuration:45days"
          a:l-h="w-c"
          a:l-w="w-c"
          a:textsize="20px"
          a:textcolor="#ff0000"/>
<Textview a:text="CFee:3000"
          a:l-h="w-c"
          a:l-w="w-c"
          a:textsize="20px"
          a:textcolor="#ff0000"/>
</ll>
```

fourth.xml

```
<LL  a:l-w="f-p"
     a:l-h="f-p"
     a:orientation="vertical">
  <Textview a:text="CName:Java"
            a:l-w="w-c"
            a:l-h="w-c"
            a:textSize="20px"
            a:textcolor="#ff0000"/>
  <Textview a:text="Cduration:45days"
            a:l-w="w-c"
            a:l-h="w-c"
            a:textsize="20px"
            a:textcolor="#ff0000"/>
  <Textview a:text="Cfee:4000"
            a:l-w="w-c"
            a:l-h="w-c"
            a:textsize="20px"
            a:textcolor="#ff0000"/>
</LL>
```

Step3:

design Activity classes.

Act1.java

```
Public class Act1 extends Activity
{
  @override
  Public void onCreate(Bundle b)
  {
    Super.onCreate(b);
    SetContentView(R.layout.first);
  }
```

```
Public void coursebtn(View v)
  {
Intent i= new Intent(this, Act2.class)
   StartActivity(i);
  }
}
```

Act2.java

```
Public class Act2 extends Activity
 {
  @override
  Public void onCreate(Bundle b)
    {
     super.onCreate(b);
     setContentView(R.layout.second)
    }
  public void androidbtn(view v)
   {
    Intent i= new Intent(this, Act3.class);
    StartActivity(i);
   }
  Public void Javabtn(view v)
   {
    Intent i= new Intent(this, Actu.class);
     StartActivity(i);
   }
 }
```

Act3.java

```
  Public class Act3 extends Activity
    {
      @override
       public void onCreate(Bundle b)
        {
         super.onCreate(b);
         setContentView(R.layout.third)
        }
```

Actu.java

```
Public class Act4 extends Activity
{
  @override
    Public void onCreate(Bundle b)
    {
      Super.onCreate(b);
      SetContentView(R.layout.fourth);
    }
}
```

Stepu:

design android manifest.xml

manifest.xml

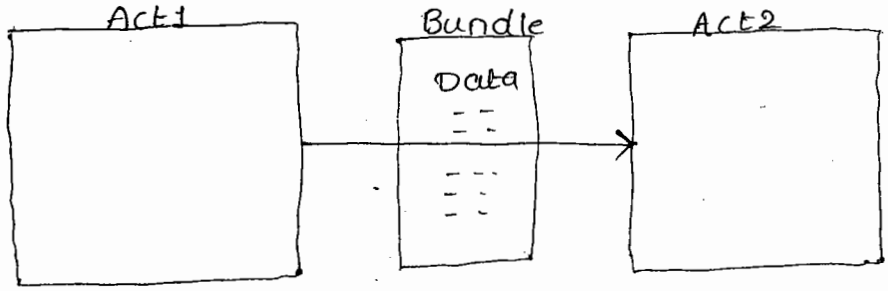
```
<manifest> <application>
  <activity a:name = ".Act1">
    <intent-filter>
      <action a:name = "android.intent.action.MAIN"/>
      <category a:name = "android.intent.action.LAUNC
                                          -HER"/>
    </i-f> </activity>
  <activity a:name = ".Act2">
    <i-f>
      <action a:name = "android.intent.action.SECOND"/>
      <category a:name = "android.intent.action.LAUNCHER"/>
    </i-f> </activity>
  <activity a:name = ".Act3">
    <i-f>
      <action a:name = "android.intent.action.THIRD"/>
      <category a:name = "android.intent.action.DEFAULT"/>
    </i-f> </activity>
  <activity a:name = ".Actu">
    <i-f>
      <action a:name = "android.intent.action.fourth"/>
      <category a:name = "  "  "  ".category.DEFAULT"/>
  </i-f> </activity> </application> </manifest>
```

1/12/2012

Switching with Data:

→ Whenever any data is sending while switching from one activity to another activity is known as switching with Data.

→ Switching with Data between two activities always will be done using Bundle class object.



→ Bundle is a special container which holds Data (either one or more homogeneous or heterogeneous Data)

Syntax:

Act1.java

```
P v click(View v)
{
Intent objref= new
  Intent(this, Act2.class);
Objref.putExtra("Key1",
                "value1");
Objref.putExtra("Key2",
                "value2");
  !     !     !
startActivity(objref);
}
```

Bundle

Key1 = value1
Key2 = value2

Act2.java

```
Bundle b= getIntent().getExtr-
                      as();
String s1= b.getString("Key1");
String s2= b.getString("Key2");
  !
  !
  !   } do some
  !   } operations
```

PutExtra():

It is a predefined method in Intent class used to add the input values in the form of key value pairs (Key should be any user-defined unique value where as value can be duplicate)

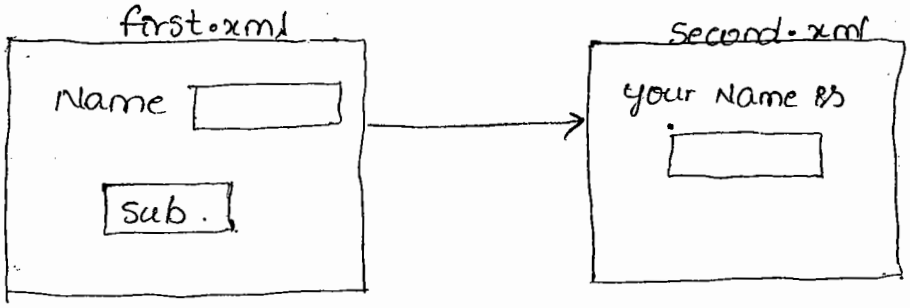
getIntent():

It is a predefined method of Activity class used to get the Intent class object from the source Activity.

getExtras():

It is a predefined method of Intent class used to get the complete data of Bundle from Source Activity.

Design an android application to achieve Switching with Data as shown below.



a) if submit button is clicked in the first.xml the name value should be transfered to the second.xml and must be displayed in the given textbox in blue color.

Step1:

Create android application.

Step2:

Design Layout files.

first.xml

```
<ll ----
    ----
    a:l-w="f-p"
    a:l-h="f-p"
    a:orientation="vertical"!>

<ll ---
    ---
    a:orientation="horizontal">

<Textview a:text="name"
          a:l-w="w-c"
          a:l-h="w-c"/>
```

```
<EditText a:onClick id="@+id/et1"
          a:l-h="w-c"
          a:l-w="w-c"/>
</LL>
<Button a:onClick="subbtn"
        a:text="submit"
        a:l-h="w-c"
        a:l-w="w-c"/>
</LinearLayout>
```

Second.xml

```
<LL>
<TextView a:text="your name is"
          a:l-w="w-c"
          a:l-h="w-c"/>
<EditText a:id="@+id/et2"
          a:l-h="w-c"
          a:l-w="w-c"
          a:textColor="#0000ff"/>
</LL>
```

Step3:

design Activity classes.

Act1.java

```
Public class Act1 extends Activity.
{
    @override
    P v onCreate(Bundle b)
    {
        super.onCreate(b);
        SetContentView(R.layout.first);
    }
    P v subbtn(View v)
    {
```

```
EditText e1=(EditText)findViewById(R.id.et1);
String s1=e1.getText().toString();
Intent i=new Intent(this,Act2.class);
i.putExtra("ab",s1);
StartActivity(i);
}
}
```

Act2.java

```
Public class Act2 extends Activity
{
@override
P v onCreate(Bundle b)
{
Super.onCreate(b);
SetContentView(R.layout.second);
Bundle b=getIntent().getExtras();
String s2=b.getString("ab");
EditText e2=(EditText)findViewById(R.id.et2);
e2.setText(s2);
}
}
```

Steps:

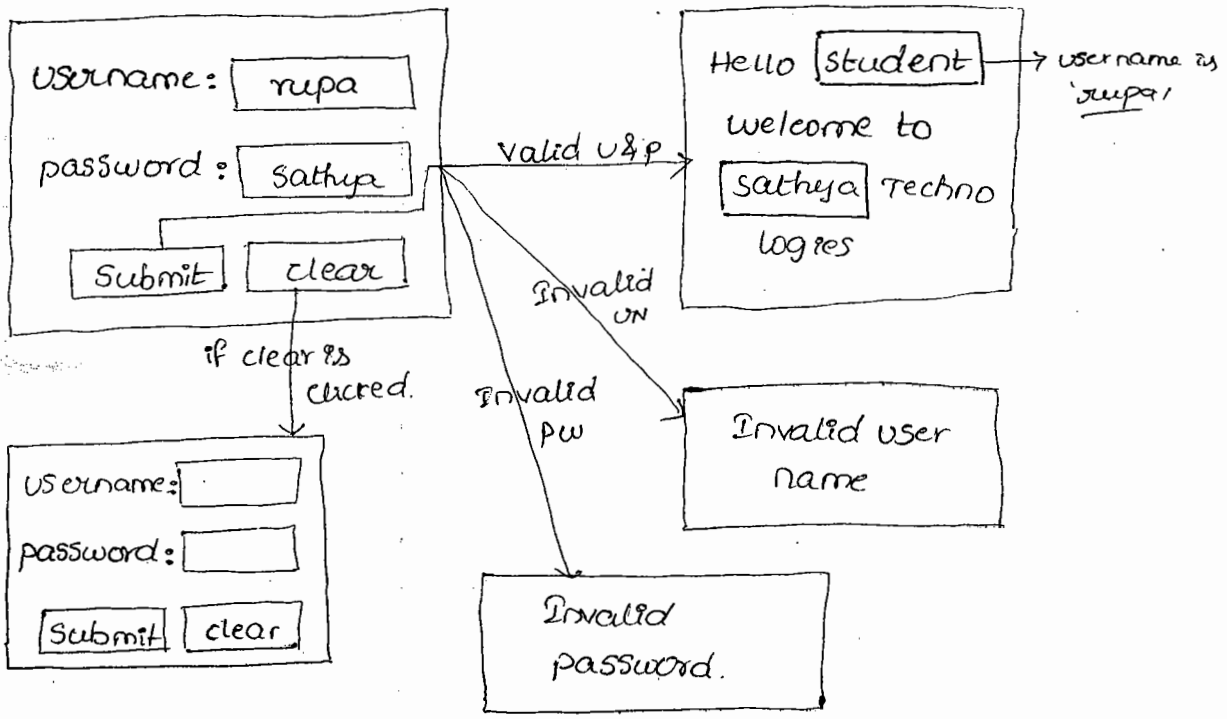
design android manifest.xml

manifest.xml

```
<manifest>
<application>
<activity a:name=".Act1">
<i-f>
<action a:name="android.intent.action.MAIN"/>
<category a:name="  "  "  .category.
LAUNCHER"/>
</i-f></activity>
```

```
<activity a:name=".Act2">
 <i-f>
  <action a:name="android.intent.action.SECOND"/>
  <category a:name="  "  "  ".category.DEFAULT"/>
  </i-f> </activity>
  </application>
 </manifest>
```

Design an android application to display a homepage if username and password is correct otherwise two error msgs seperately displayed for invalid username and for invalid password.



Step1: Create android application.

Step2: design Layout files.

first.xml

```
<ll  ==
     a:l-w="f-p"
     a:l-h="f-p"
     a:Orientation="vertical"/>
 <ll  ==
     a:Orientatio="Horizontal"/>
```

```
<TextView a:text="username"
          a:l-h="w-c"
          a:l-w="w-c"/>
<EditText a:id="@+id/et1"
          a:l-w="w-c"
          a:l-h="w-c"/>
</LL>

<LL ---
    a:orientation="Horizontal">
  <TextView
      a:l-w="w-c"
      a:l-h="w-c"
      a:text="password"/>
  <EditText
      a:l-w="w-c"
      a:l-h="w-c"
      a:id="@+id/et2"/>
</LL>

<LL ----
    a:orientation="Horizontal">
  <Button a:text="Submit"
          a:l-w="w-c"
          a:l-h="w-c"
          a:onClick="subbtn"/>
  <Button a:text="Cancel"
          a:l-w="w-c"
          a:l-h="w-c"
          a:onClick="canbtn"/>
</LL>
</LL>
```

Second.xml

```
<LL ---
    a:l-w="f-p"
    a:l-h="f-p"
    a:orientation="Optional"/>
```

```
<Textview a:l-w="w-c"
          a:l-h="w-c"
          a:axxe:id="@+id/ti"/>

</LL>
```

Third.xml

```
<LL>
  <Textview a:l-w="w-c"
            a:l-h="w-c"  a:text="Invalid username"/>
            a:id="@+id/tv2"/>
</LL>
```

fourth.xml

```
<LL ---
    a:l-w="@-p"
    a:l-h="@-p"/>

  <Textview a:l-w="w-c"
            a:l-h="w-c"
            a:text="Invalid password"/>
            a:id="@
```

Step3:

Activity classes

Act1.java

```
Public class Act1 extends Activity
{
  Public void onCreate(Bundle b)
  {
    Super.onCreate(b);
    setContentView(R.layout.first)
  }
  Public void Subbtn(View v)
  {
    EditText e1=(EditText) findViewById(R.id.et1)
     String s1=e1.getText().tostring();
    EditText e2=(EditText) findViewById(R.id.et2)
     String s2=e2.getText().tostring();
```

```
if(s1.equals("rupa"))
  {
    if(s2.equals("Sathya"))
      {
        Intent i = new Intent(this, Act2.class);
          i.putExtra("ab", s1);
          i.putExtra("cd", s2);
        StartActivity(i);
      }
    else
     {
      Intent i1 = new Intent(this, Actu.class)
        startActivity(i1);
      }
  }
 else
   {
    Intent i2 = new Intent(this, Act3.class)
      startActivity(i2);
   }
Public void canbtn(view v)
  {
    EditText e3 = (EditText) findViewById(R.id.eb1);
     EditText eu = (EditText) findViewById(R.id.eb2);
      e3.setValue(Text "  ");
      eu.setValue(Text "  ");
  }
}
```

Act2.java

```
public class Act2 extends Activity
  {
    @override
      P v onCreate(Bundle b)
        {
```

```
        Super.onCreate(b);
        SetContentView(R.layout.Second);
        Bundle b = getIntent().getExtras();
        String name = b.getString("ab");
        String pwd = b.getString("cd");
        Textview tv = (Textview) findViewById(R.id.t1);
          tv.setText("Hello " + S1 + "welcome to " + S2 + " Technolog-
                                                        ies");
      }
    }
```

Act3.java

```
Public class Act3 extends Activity
  {
    P v onCreate(Bundle b)
     {
       Super.onCreate(b);
       SetContentView(R.layout.third);
     }
  }
```

Act4.java

```
Public class Act4 extends Activity
  {
    P v onCreate(Bundle b)
     {
       Super.onCreate(b);
       SetContentView(R.layout.Fourth);
     }
  }
```

Steps:

design android manifest.xml

manifest.xml

```
<manifest><application>
<activity a:name=".Act1">
  <i-f>
   <action a:name="android.intent.action.MAIN"/>
   <category a:name="  "  "  .category.LAUNCHER"/>
  </i-f></activity>
<activity a:name=".Act2">
  <i-f>
  <action a:name="android.intent.action.SECOND"/>
  <category a:name="  "  "  .category.DEFAULT"/>
  </i-f></activity>
<activity a:name=".Act3">
  <i-f>
   <action a:name="android.intent.action.THIRD"/>
   <category a:name="  "  "  .category.DEFAULT"/>
  </i-f></activity>
<activity a:name=".Act4">
  <i-f>
    <action a:name="android.intent.action.FOURTH"/>
    <category a:name="  "  "  .category.DEFAULT"/>
  </i-f></activity>
</application>
</manifest>
```

3. Switching with Result Data:

→ Whenever the source activity is expecting response from the destination activity after sending some request data at the time of switching is known as "Switching with Result Data"

Syntax:

startActivityForResult()

Which can be used to not only to send the request to the Destination activity but also expecting the response back from the Destenation activity.

onActivityResult()

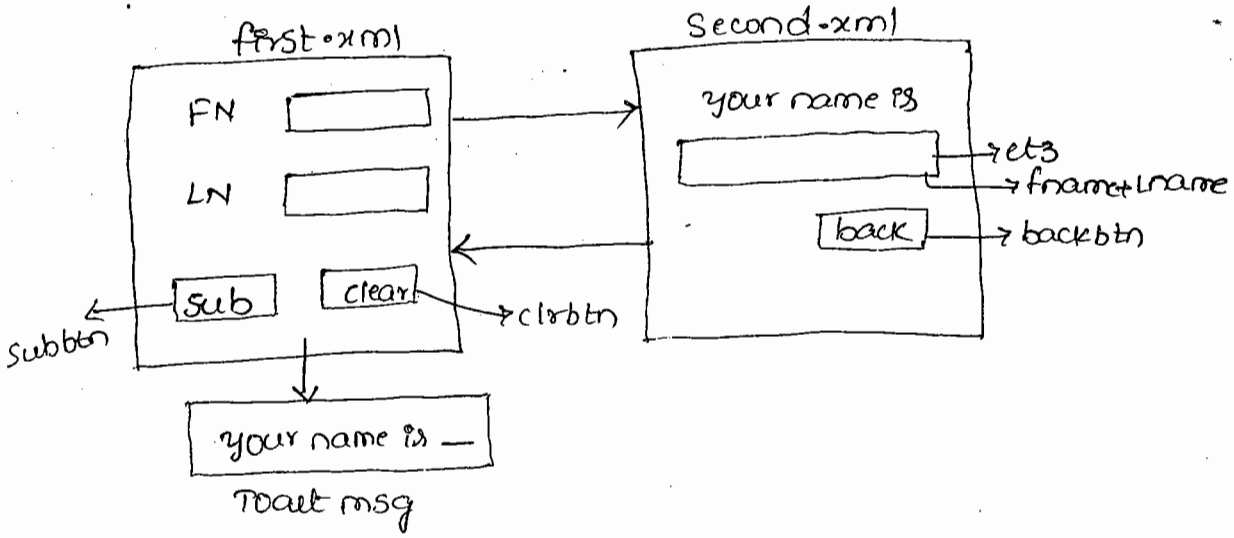
It is a predefined method of Activity class should be overridden in the source Activity to accept the response from the destination activity.

finish()

It is a predefined method of Activity class should be overridden in the Dest Activity to send the response back to Source Activity.

4/12/2012

Design an android appln to achieve switching with Result Data as shown below.



Step1: Create android application

Step2: design layout files.

first.xml

```
<LL ---
    a:orientation = "vertical" >
```

```
<ll ---
  a:Orientation = "horizontal" >
   <TextView  a:text = "firstName"
              a:l-h = "w-c"
              a:l-w = "w-c"
              a:textSize = "20px" />
   <EditText  a:l-h = "w-c"
              a:l-w = "w-c"
              a:id = "@+id/et1" />
</ll>
<ll> ---
   a:orientation = "horizontal" >
   <TextView  a:text = "LastName"
              a:l-h = "w-c"
              a:l-w = "w-c"
              a:textSize = "20px" />
   <EditText  a:id = "@+id/et2"
              a:l-h = "w-c"
              a:l-w = "w-c" />
</ll>
<ll ---
   a:orientation = "horizontal" >
   <Button  a:text = "submit"
            a:l-h = "w-c"
            a:l-w = "w-c"
            a:onClick = "subbtn" />
   <Button  a:text = "clear"
            a:l-h = "w-c"
            a:l-w = "w-c"
            a:onClick = "clrbtn" />
</ll>
</ll>
```

Second.xml

```
<LL ---
    ---
    a: Orientation = "vertical" >
  < TextView  a: text = " your Name is "
              a: l-w = " w-c "
              a: l-h = " w-c " />
  < EditText  a: l-w = " w-c "
              a: l-h = " w-c "
              a: id = " @+id/et3 " />
  < Button a: text = " back "
           a: l-w = " w-c "
           a: l-h = " w-c "
           a: onClick = " backbtn " />
</LL>
```

Step3:

design Activity classes.

Act1.java

```
Public class Act1 extends Activity
{
    EditText e1, e2;
    Public static final int REQUEST-CODE = 10;
    @Override
    P v onCreate (Bundle b)
    {
      Super.OnCreate (b);
      SetContentView (R.layout.first);
      e1 = (EditText) findViewById (R.id.et1);
      e2 = (EditText) findViewById (R.id.et2);
    }
```

```
P v Subbtn (view v)
 {
   String fn = e1.getText().toString();
   String ln = e2.getText().toString();
   Intent i = new Intent (this, Act2.class);
      i.putExtra ("K1", fn);
      i.putExtra ("K2", ln);
   StartActivityforResult (i, REQUEST-CODE);
 }
P v clrbtn(view v)
 {
   e1.setText (" ");
   e2.setText (" ");
 }
P v onActivityResult (int reqcode, int rescode, Intent iobj)
 {
   if ( REQUEST-CODE == reqcode)
    {
      Bundle b1 = iobj.getExtras();
      String n = b1.getString ("nm");   directly we can write this also
      Toast.MakeText ( this, " your name is "+n; 3000).
                                          Show();
    }
   else
    {
      Toast.makeText ( this, " operation failed", 5000).
                                          show();
    }
 }
}
```

Act2.java

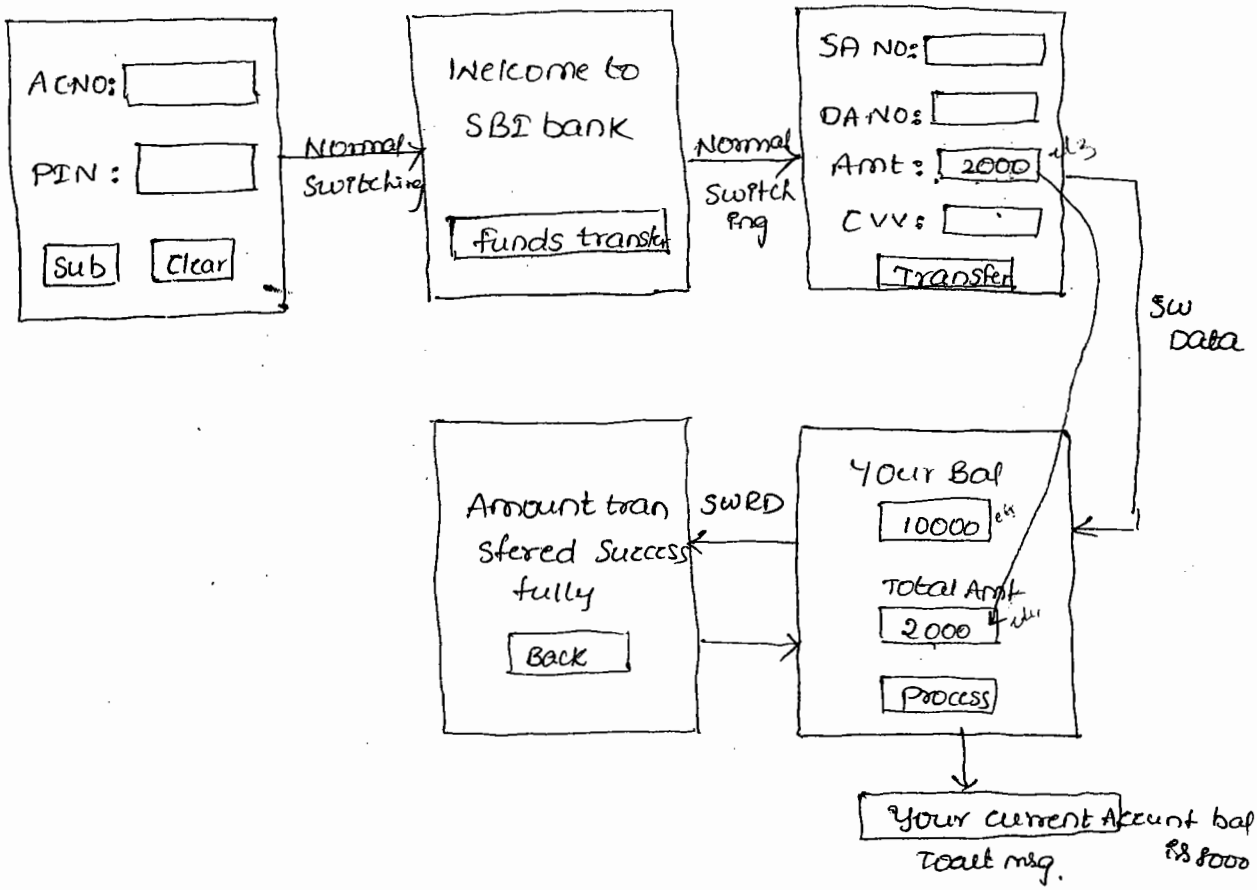
```
public class Act2 extends Activity
{
    String name; Public s f int RESULT-OK = 20;
    @override
  public void onCreate(Bundle b)
   {
      Super.onCreate(b);
       setContentView(R.layout.second);
    Bundle b = getIntent().getExtras();
     String f = b.getString("K1");
     String l = b.getString("K2");
     String name = f.concat(l);
                    (or)
                  f+l;
     EditText e3 = (EditText) findViewById(R.id.et3);
        e3.setText(name);
   }
  P v backbtn(View v);
   {
    finish();
    }
 P v finish()
   {
    Intent i1 = new Intent();
     i1.putExtra("nm", name);
     SetResult(i1, RESULT-OK);
      Super.finish();
      }
  }
```

Step 4: develope android manifest.xml

manifest.xml

```
<manifest>
    <application>
        <activity a:name = ".Act1"> <i-f>
        <action a:name = "android.intent.action.MAIN"/>
        <category a:name = "android.intent.category.
                                        LAUNCHER"/>
        </i-f> </activity>
        <activity a:name = ".Act2"> <i-f>
        <action a:name = "android.intent.action.SECOND"/>
        <category a:name = "android.intent.category.
                                        DEFAULT"/>
        </i-f> </activity>
    </application> </manifest>
```

Design an android appln for transfering funds in a bank as shown follows.



Step1: Create android application.

Step2: Design layout files.

first.xml:

```
<ll - - -
    - - -
    a:orientation="vertical">
  <ll - - -
      - - -
      a:orientation="horizontal">
    <Textview a:text="ACNO"
              - - - -
              - - -/>
    <Edittext a:id="@+id/et1"
              - - -
              - - -/>
  </ll>
  <ll - - -
      - - -
      a:orientation="horizontal">
    <Textview a:text="PIN"
              - - -
              - - -/>
    <Edittext a:tex id="@+id/et2"
              - - -
              - - -/>
  </ll>
  <ll - - -
      - - -
      a:orientation="horizontal">
    <Button a:onclick="subbtn"
            - - -
            - - -/>
    <Button a:onclick="clrbtn"
            - - -
            - - -/>
  </ll>
</ll>
```

Second.xml

```
<LL ---
    a:ori="vertical">
  <TV --
      a:text="Welcome --
              --- "/>
  <BT --
      a:text="funds
              transfer"/>
</LL>
```

Third.xml

```
<LL ---
    a:ori="vertical">
  <LL ---
      a:ori="horizontal">
    <TV --
        a:text="SA NO"/>
    <ET --
        a:id="@+id/-- "/>
  </LL>
  <LL ---
    a:orien="horizontal">
    <TV --
        a:text="DA NO"/>
    <ET --
        a:id="@+id/-- "/>
  </LL>
  <LL>
    <TV --
        a:text="Amt"/>
    <ET --
        a:id="@+id/- "/>
  </LL>
  <LL --
      a:ori="horizontal">
    <TV --
        a:text="cvv"/>
    <ET --
        a:id="@+id/- "/>
  </LL>
  <Button ---
      a:text="Transfer"/>
</LL>
```

fourth.xml

```
<LL ---
    a:ori="vertical">
  <TV --
      a:text="your balance"/>
  <ET --
      a:id="@+id/-"/>
  <TV --
      a:text="Total amt"/>
  <ET --
      a:id="@+id/--"/>
  <Bt ---
      a:text="process"/>
</LL>
```

fifth.xml

```
<LL ---
    a:ori="vertical">
  <TV ---
      a:text="amt transfered
              Sucessfully"/>
  <Button --
      a:text="back"/>
</LL>
```

Step3 :

design Activity classes.

Act1.java

```
public class Act1 extends Activity
 {
    EditText e1, e2;
  @Override
   P v onCreate(Bundle b)
   {
     Super.onCreate(b);
     SetContentView(R.layout.first);
        e1 = (EditText) findViewById(R.id.et1);
        e2 = (EditText) findViewById(R.id.et2);
   }.
   P v subbtn(View v)
    {
     String m = e1.getText().toString();
     String n = e2.getText().toString();
     Intent i = new Intent(this, Act2.class);
          i.putExtra("K1", m);
          i.putExtra("K2", n);
      StartActivity(i);
   }
   P v clrbtn(View v)
    {
      e1.setText("");
      e2.setText("");
    }
 }
```

Act2.java

```
Public class Act2 extends Activity
{
    @override
     P v onCreate (Bundle b)
     {
       Super.onCreate(b);
       SetContentView(R.layout.second);
       Bundle y
     }
                    P v FundsBtn(View v)
                    {
Act3.java           Intent i = new Intent(this, Act3.class);
                      StartActivity(i);
                    }

Public class Act3 extends Activity
{
    @override
    P v onCreate (Bundle b)
    {
      Super.oncreate(b);
      setcontent view(R.layout.third)
    }
P v TransferBtn(  EditText e3 = (EditText) findViewById(R.id.et3);
      View v)     EditText e4 = (EditText) findViewById(R.id.et4);
    {             EditText e5 = (EditText) findViewById(R.id.et5);
                  EditText e6 = (EditText) findViewById(R.id.et6);
                  String s = e3.getText().toString();
                  Intent i1 = new Intent(this, Act4.class);
                    i1.putExtra("k3", s);
                    startActivity(i1);
                }
}
```

Act4.java

```
public Act4 extends Activity
{
    EditText e4, e5; PS f int REQUEST_CODE = 10;
    String su;
    @Override
    P v onCreate(Bundle b)
    {
        super.onCreate(b);
        setContentView(R.layout.fourth);
        Bundle b = getIntent().getExtras();
        String su = b.getString("k3");
        EditText e4 = (EditText) findViewById(R.id.et4);
        e4.setText(su);
    }
    P v processEvent
    {
        e5 = (EditText) findViewById(R.id.et5);
        String s5 = e5.getText().toString();
        double d1 = Double.parseDouble(s4);
        double d2 = Double.parseDouble(s5);
        int in = INTEGER.parseInt(s4);
        int in1 = INTEGER.ParseInt(s5);
        Intent i = new Intent(this, Act5.class);
        i.putExtra("k4", in)
        i.putExtra("k5", in1)
        if (d1 >= d2)
        {
        StartActivityforRESULT(i, REQUEST_CODE);
    }
    else
    P v onActivityResult(int reqcode, int rescode, Intent robj)
    {
        if (REQUEST_CODE == reqcode)
        {
            Bundle bd = robj.getExtras();
            int in2 = bd.getInt("k6");
            in2 = bd.getString("k6");
            Toast.makeText(this, "your current Account balance
                is" + in2, 3000).show();
        }
        else
        {
            Toast.makeText(this, "transaction failed", 3000).
                show();
        }
    }
}
```

Act5.java

```
Public class Act5 extends Activity
{
   double (int) balance ;     P v t int RESULT_OK = 25;
    @Override
    P v onCreate( Bundle b)
    {
     super.onCreate(b)
     setContentView(R.layout.activity_main);
     Bundle bn = getIntent().getExtras();
   int String n1 = bn.getInt("K4");
     int  n2 = bn.getInt("K5");
      double balance = n1-n2;
    }
    P  v backbtn(View v)
      {
       finish();
       }
   P v finish()
     {
     Intent i2 = new Intent();
       i2.putExtra("K6", balance);
       setResult(RESULT_OK, i2);
          super.finish();
       }
}
```

-) Switch lang... by ... (...)

```
while( )
{
  ...
  printf(" ...
  }
else
  {
  }
}
```

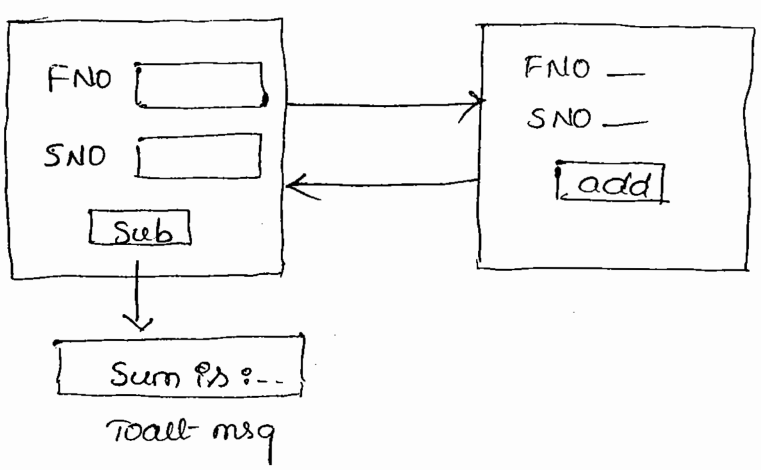
```
int a[5]={10,20,30}, i=1;
for( ;i<5; )
{
  printf("
}
```

3,010,0

5/12/2012

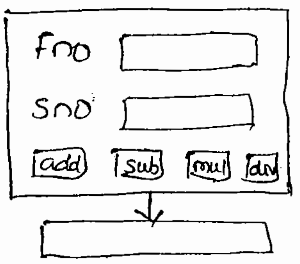
Develope the following applications

1.




```
Click btn(View v)
{
switch(v.getId())
{
case R.id.add: ...
case R.id.sub: ...
}
```

2.



Toast msg

2) Implicit Switching : (OR)

-20,3,0,0

Implicit Intent :

→ Whenever switching is done automatically between user defined layout to predefined layout (or) between predefined layout to predefined layout is known as "Implicit Intent"

→ android mainly supports following predefined Intents

1) ContactsIntent
2) BrowserIntent
3) SearchIntent
4) MapIntent
5) CallIntent

1) contactsIntent :

→ It is a predefined Intent used to get all the contact details from the current Android device.

→ following predefined code can be used to retrieve the contact Details.

```
Intent iobj = new Intent();
iobj.setAction(android.content.Intent.ACTION_VIEW);
iobj.setData(ContactsContract.Contacts.CONTENT_URI);
startActivity(iobj);
```

(OR)

```
Intent iobj = new Intent(Intent.ACTION_VIEW, ContactsCo.
                nTract.Contacts.CONTENT_URI);
StartActivity(iobj);
```

→ In the above syntax setAction(-) method can be used to perform an action to get all the predefined contact details whereas setData(-) method can be used to set all the contact details from the android device to the predefined layout file

2) Browser Intent:

→ Which can be used to perform browsing operation automatically (or) Which can be used to switch from userdefined Activity to predefined Browser Intent Activity. By writing following syntax.

```
Intent iobj = new Intent();
iobj.setAction(android.content.Intent.ACTION_VIEW);
iobj.setData(uri.parse("http://www.google.com"));
StartActivity(iobj);
```

(can be any url)

(OR)

```
Intent iobj = new Intent(Intent.ACTION_VIEW, uri.p-
    arse("http://www.google.com"));
StartActivity(iobj);
```

3) Search Intent:

It is a predefined Intent can be used to switch from any userdefined activity to predefined browser Intent with the following code.

```
Intent iobj = new Intent(Intent.ACTION_WEB_SEARCH);
iobj.putExtra(SearchManager.QUERY, "androidma
    terial");
StartActivity(iobj);
```

(can be any userdefined value.)

1. step1: create android application.

step2: design layout files.

first.xml

```
<ll ---
    a:orientation = "vertical" >
```

```
<LL ---
    a:orientation="horizontal">
  <TextView a:text="FNO"
            a:l-w="w-c"
            a:l-h="w-c" />
  <EditText a:l-w="w-c"
            a:l-h="w-c"
            a:id="@+id/et1" />
</LLL>

<LL ---
    a:orientation="horizontal">
  <TextView a:text="SNO"
            a:l-w="w-c"
            a:l-h="w-c" />
  <EditText a:l-w="w-c"
            a:l-h="w-c"
            a:id="@+id/et2" />
</LL>

<Button a:l-w="w-c"
        a:l-h="w-c"
        a:onclick="subbtn" />
</LL>
```

second.xml

```
<LL ---
    a:orientation="vertical">
  <LL ---
      a:orientation="horizontal">
    <TextView ---
              a:text="FNO" />
    <EditText ---
              a:id="@+id/et3" />
  </LL>
```

```
<ll - - -
   - - -
  a: orientation = "horizontal" >
   < Textview  - - -
               - - -
               a: text = "SNO" />
   < EditText  - - -
               - - -
               a: id = "@+ id/et4" />
  </ll>
  <Button  - - -
           - - -
           a: onClick = "backbtn" />
  </ll>
```

Step 3:

design activity classes

Act1.java

```
Public class Act1 extends Activity
  {
     EditText e1,e2;
     Public static final int REQUEST-CODE = 30;
     @override
      P v onCreate (Bundle b)
      {
        Super. onCreate (b);
        setContentView (R. layout. first); }
    P v Subbtn (view v){
EditText  e1 = (EditText) findViewById (R. id. et1);
 EditText  e2 = (EditText) findViewById (R. id. et2);
     String fn = e1. getText(). tostring();
     String ln = e2. get Text(). tostring();
      int n1 = INTEGER. ParseInt (fn);
      int n2 = INTEGER. ParseInt (ln);
   Intent i: new Intent (this, Act2. class);
          i. putExtra ("K1", n1);
          i. putExtra ("K2", n2);
```

```
StartActivityForResult(i, REQUEST_CODE);

p v onActivityResult ( int reqcode, int rescode, Intent iobj)
  {
    Bundle b1 = new
     If ( REQUEST_CODE == reqcode)
      {
       Bundle b1 = iobj.getExtras();
       int n = b1.getInt ("k3");
       Toast.makeText ( this, "sum is" + n , 3000).show();
       Else
       {
       Toast.makeText ( this, "operation failed", 5000).show();
       }
    }
}
```

Act2.java

```
Public class Act2 extends Activity
  {
    p s f int RESULT_OK = 40;
    @override
    p v onCreate (Bundle b)
    {
     super.onCreate(b);
     setContentView ( R.layout.second);
     Bundle b = getIntent().getExtras();
      int n3 = b.getInt ("k1");
      int n4 = b.getInt ("k2");
         sum = n3+n4;
     EditText e3 = (EditText) findViewById(R.id.et3);
       e3.setText (sum);
    }
     p v backbtn(View v) {
            finish(); }
```

```
p v finish()
 {
  Intent i1= new Intent();
  i1.putExtra("k3", sum);
  setResult(RESULT_OK, i1);
   super.finish();
  }
}
```

5 steps: design android manifest.xml

Manifest.xml

```
<manifest>
  <application>
   <activity a:name=".Act1">
    <i-f>
   <action a:name="android.intent.action.
                                MAIN" />
   <category a:name="android.intent.category.
                                LAUNCHER"/>
    </i-f>
   </activity>
  <activity a:name=".Act2">
   <i-f>
   <action a:name="android.intent.action.
                                SECOND"/>
   <category a:name="android.intent.category.
                                LAUNCHER"/>
    </i-f>
    </activity>
  </application>
</manifest>
```

2. step1: Create android application

step2: design Layout file

first.xml

```
<LL - - -
    a:orientation = "vertical" >
<LL - - -
    a:orientation = "horizontal" >
 < TextView - - - -
              - - -
              a:text = "Fno" />
 < EditText - - -
              - - -
              a:id = "@+id/et1" />
 </LL>
 <LL - - -
     - - -
     a:orientation = "horizontal" >
  <TextView - - -
            - - -
            a:text = "Sno" />
  <EditText - - -
            - - -
            a:id = "@+id/et2" />
  </LL>
  <LL - - -
      a:orientation = "horizontal" >
  <Button a:text = "add"  a:id = "@+id/bt1"
          - - -
          a:onClick = "btn" />
  <Button a:text = "sub"  a:id = "@+id/bt2"
          - - -
          a:onClick = "btn" />
  < Button a:text = "mul"
           - - -  a:id = "@+id/bt3"
           a:onClick = "btn" />
  < Button a:text = "div"  a:id = "@+id/bt4"
           - - -
           a:onClick = "btn" />
  </LL>
</LL>
```

Step3: design Activity class

Act.java

```
P c Act extends Activity
{
    int n1, n2;
  @override
   P v onCreate(Bundle b)
    {
     super.onCreate(b);
     setContentView(R.layout.first);
    }
   P v clickbtn(View v)
    {
     switch(v.getId())
     {
      case R.id.bt1 : int n = n1+n2;
                      Toast.makeText(this, "Sum is"+n, 3000).
                                                  show();
      case R.id.bt2 : int m = n1-n2;
                      Toast.makeText(this, "difference is"+m,
                      5000).show();
      case R.id.bt3 : int p = n1*n2;
                      Toast.makeText(this, "multiplication is"+p,
                      5000).show();
      case R.id.bt4 : int q = n1/n2;
                      Toast.makeText(this, "division is"+q,
                      5000).show();
     }
    }
}
```

registration

```
EditText e1 = (EditText)
findViewById(R.id.et1);
String s1 = e.getText().toString();
EditText e2 = (EditText)
findViewById(R.id.et2);
String s2 = e.getText().toString();
n1 = Integer.parseInt(s1);
n2 = Integer.parseInt(s2);
```

6/12/2012

4. Map Intent:

→ It is a predefined Intent which can be used to switch from user defined activity to predefined activit Map Intent. That means any location can view in the map directly.

Syntax:

```
Intent iobj = new Intent( Intent. ACTIO ACTION_VIEW, uri.pa-
        rse("geo:0,0?q=hyderabad"));
StartActivity(iobj);
```

→ In the above syntax,

* Geo (geographical point) is a protocol identifies a google map.

* 0,0 are the geographical points. (lattitude & longitude)

* q is a query parameter, that will start searching from (0,0) location.

5. call Intent:

→ It is a predefined Intent can be used to switch from user defined activity to predefined call Intent Activity, (i.e.) which provides predefined logic to make calls from an android application to other mobile device.

Syntax:

```
        iobj
Intent i = new Intent();
iobj.setAction(android.content.Intent.ACTION_CALL);
iobj.setData(uri.parse("tel:any telephone number"));
StartActivity(iobj);
```

(OR)

```
Intent pobj = new Intent( Intent.ACTION_CALL, uri.par
                se( " tel:9550344013 "));
        StartActivity (pobj); //
```

→ In the above syntax,

* tel is a protocol used to make calls through call Intent

NOTE:1

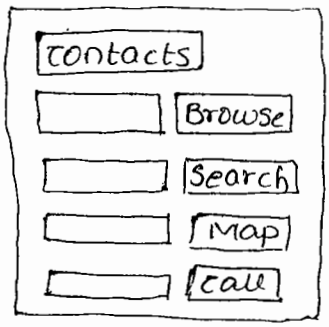
While using Browse Intent (or) Search Intent (or) Map Intent it is very mandatory to take the Internet permission in the manifest file as shown below.

```
<manifest>
    <uses-permission android:name = " android.permi-
                    ssion.INTERNET " />
    <application>.
        - - - - -
        - - - -
    </application>
</manifest>
```

NOTE: 2

any type of permissions should not be written within <application> tag.

Design an android appln to use the Implicit Intents as shown below.



Step 1 :

Step 2 :

first.xml

```
<ll - - -
    - - -
    a:orientation = "vertical" >
                                  → < Button a:text = "contacts"
                                      a:onclick = "con
                                      - - - -        btn"
                                                      />
<ll - - -
    a : orientation = "horizontal" >

  < Textview  a/: text =/*

    < EditText  - - - -
                - - - -
                a:id = "@+id/et1" />

    < Button  a:text = "Browse"
              a:onclick = "Browsebtn"
              - - -
              - - -  . />

</ll>

<ll - - -
    - - -
    a:orientation = "horizontal" >

  < EditText - - -
              - - -
              a:id = "@+id/et2" />

  < Button  a:text = "search"
            a:onclick = "searchbtn"
            - - -
            - - />

</ll>

<ll - - -
    - - -
    a:orientation = "horizontal" >

  < EditText - - -
              a:id = "@+id/et3" />

  < Button a:text = "Map"
           a:onclick = "Mapbtn" />

</ll>

<ll - - -
    - - -
    a:ori = "horizontal" >
```

```
<EditText a:text="call ---
        a:id="@+id/et4"
          -- />
<Button a:text="call"
        a:onclick="callbtn"
          -- />
</LL>
</LL>
```

Step3: Design Activity class.

Act1.java

```
public class Act1 extends Activity
{
  @override
   P v onCreate(Bundle b)
    {
     super.onCreate(b);
     SetContentView(R.layout.first);
    }
   P v Conbtn(View v)
   {
    Intent i=new Intent(Intent.ACTION_VIEW,
          Contacts Contract.Contacts.CONTENT_URI);
       startActivity(i);
    }
   P v Browsebtn(View v)
    {
  EditText e1=(EditText)findViewById(R.id.et1);
     String s1=e1.getText().toString();
  Intent i1=new Intent(Intent.ACTION_VIEW,
                  Uri.parse(s1));
    }  StartActivity(i1);   s.o.pln(uri);
```

```
P v SearchBtn(View v)
 {
EditText e2 = (EditText) findViewById(R.id.et2);
 String s2 = e2.getText().toString();
 Intent i2 = new Intent(Intent.ACTION_WEB_SEARCH);
   i2.putExtra(SearchManager.QUERY, s2);
    startActivity(i2);
 }
P v MapBtn(View v)
 {
EditText e3 = (EditText) findViewById(R.id.et3);
 String s3 = e3.getText().toString();
 Intent i3 = new Intent(Intent.ACTION_VIEW,
          uri.parse("geo:0,0?q="+s3));
   startActivity(i3);
 }
P v callBtn(View v)
 {
 EditText e4 = (EditText) findViewById(R.id.et4);
   String s4 = e3.getText().toString();
  Intent i4 = new Intent(Intent.ACTION_CALL,
            uri.parse("tel:"+s4));
     startActivity(i4);
 }
}
```

our android mobile, if internet connection is not worked in Then setting → wireless network setting → phone network → Register name & In place of internet write www.

NOTE:

→ To view the contact details we must take the following permissions in the manifest file

<uses-permission android:name="android.permission.READ-CONTACTS"/>

→ We must take following permission to make call fowm one device to another device.

<uses-permission android:name="android.permission.CALL-PHONE"/>

7/12/2012

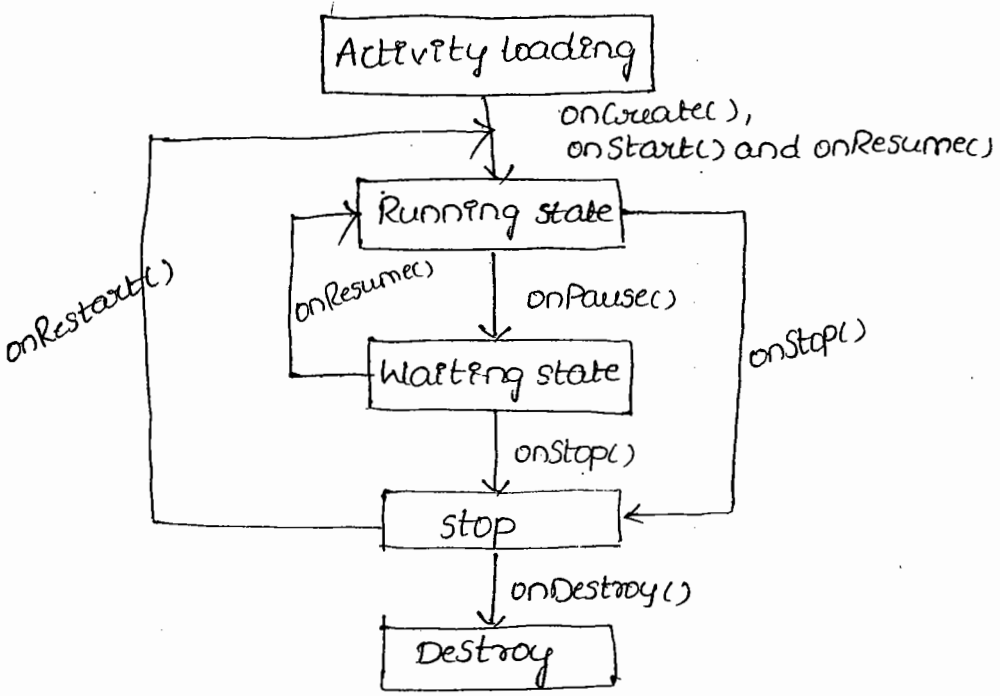
Activity Lifecycle Methods:

→ Activity is a predefined class which contains different states like loading state, running state, waiting state etc.,

→ to convert from one state to another state predefined lifecycles methods can be used.

→ In general Activity lifecycle methods can be used to design either Gaming applications (or) Music player applications.

→ Activity state Diagram:



Lifecycle Methods of Activity class:

1. onCreate():

Which will be executed while loading an activity.

Syntax:

```
Public void onCreate(-)
  {
    Super.onCreate(-)
      ---
      ---   } Userdefined logic
      --
  }
```

2. onStart():

Which will be executed whenever activity changes converting from loading state to running state.

Syntax:

```
Public void onStart(-)
  {
    Super.onstart(-)
      ---
      --   } Userdefined logic
      --
  }
```

3. onPause():

Which will be executed whenever activity changes from running state to waiting state

Syntax:

```
Public void onPause(-)
  {
    Super.onPause(-)
      ---
      ---
      --
  }
```

4. onResume():

Which will be executed whenever activity changes from waiting state to running state

Syntax:

```
Public void onResume(-)
  {
   Super.onResume(-)
     - - -
     - - } userdefined logic
     - -
  }
```

5. onStop():

Which will be executed whenever activity changes from either running state to stopstate (or) waiting state to Stop state

Syntax:

```
Public void onStop(-)
  {
   Super.onStop(-)
     - - -
     - - } U.D.L
     - -
  }
```

6. onDestroy()

Which can be used to destroy the operation permanently. (or) Which will be called whenever activity is stopped permanently (or) destroyed.

Syntax:

```
public void onDestroy(-)
   {
    super.onDestroy(-)
      - -
      - - } U.D.L
      - -
   }
```

```
@override
Protected void onDestroy()
 {
    Super.onDestroy();
  Toast t= Toast.makeText(this, "onDestroy() called...",
          Toast.LENGTH-LONG);t.show();
  }
 @override
  protected void onPause()
   {
     Super.onPause();
  Toast t= Toast.makeText( this, "onPause() called---",
          Toast.LENGTH-LONG);t.show();
    }
  @override
  Protected void onRestart()
   {
     super.onRestart();
  Toast t =Toast.makeText(this, "onRestart() called-",
          Toast.LENGTH-LONG);t.show(); }
    @override
     Protected void onResume(){
     Super.onResume();
  Toast t= Toast.makeText(this, "onResume() called-",
          Toast.LENGTH-LONG);t.show();
  Toast t1= Toast.makeText(this, "Resources loaded
       Successfully..", Toast.LENGTH-LONG);t1.show();}
    @override
     protected void onstart(){
        Super.onstart();
  Toast t= Toast.makeText( this, "onstart() called--",
          Toast.LENGTH-LONG);t.show();}
    @override
     protected void onstop{
        Super.onstop();
  Toast t= Toast.makeText( this, "onstop() called --",
          Toast.LENGTH-LONG);t.show();}
}
```

7. onRestart():

Which will be executed whenever activity changes from Stop state to running state

Syntax:

```
Public void onRestart(-)
{
  super.onRestart(-)
    - - - } U.D.T
}
```

→ By Default onCreate(), onStart() and onResume() methods will be executed in a Sequential order.

NOTE:

onPause() will be executed while switching from one activity to another activity.

⇒ Design an android application to display the lifecycle methods

```
Package com.activitylife;
import android.app.Activity;
import android.os.Bundle;
import android.widget.Toast;
P c ActivityLifeCycleActivity extends Activity
  {
    @override
     P v onCreate(Bundle b)
      {
        super.onCreate(b);
        SetContentView(R.layout.main);
      Toast t = Toast.makeText(this, "oncreate() called--
                                    ...", TOAST.LENGTH_LONG);
      }     t.show();
```

8/12/2012

GUI DESIGNING

→ GUI Designing mainly depends on following categories

1. View components
2. Layout Design
3. resources
4. Styles and themes
5. Menu designing (advanced UI Designing)

1. View components:

→ Which can be used to providing provisioning to the end user either to provide input values or to view the output values or messages

→ In android API every view component is existing as predefined class and those can be used as predefined tags in xml.

1. TextView

→ Which can be used to display a text message.

Syntax:

```
<TextView
        a:l-w="fill-parent"
        a:l-h="wrap-content"
        a:textsize="25px"
        a:text="Hello"
        a:textcolor=" "
        a:---/>
```

2. Button:

Which will give a push button, using this end user can request to perform some operation.

Syntax:

```
<Button a:text=" "
        a:l-w=" "
        a:l-h=" "
        a:id=" @+id/idname"
        a:onClick=" " />
```

3. EditText :

→ used to display a text box.

Syntax:

```
<EditText a:id=
          a:textColor=
          a:textSize=
          |
          a:input=" text" → which will accept any type of characters in text format (alphabets, numerics & special symbols)
```

a:input="textMultiline" → which can be used make the EditText as TextArea

a:input="textPassword" → which will accept the text in password format (masking format)

a:input="number" → which will accept only numeric values

a:input="numberSigned" → accept -ve numeric values

a:input="numberDecimal" → accept Decimal numbers

a:input="date" → accept the value in Date format

a:input="time" → accept the value in time format

```
        ...
        ... />
```

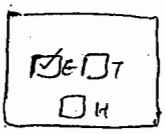
4. CheckBox:

→ Which can be used to display a Multiselection checkBox.

Syntax:

```
<CheckBox a:text="English"
        E
    a:checked="true"/"false"/>
```

CheckBox with English



↳ selected automatically

5. RadioButton:

→ Which can be used to display a single selection Radio Button

Syntax:

```
<RadioButton a:text="Android"
        ---
        --
        --
    a:checked="true"/>
```

(By default selected)

RadioButton with Android (selected)

6. RadioGroup:

→ Which can be used to make set of RadioButtons under one Group. so that only one RadioButton can be selected at a time

Syntax:

```
<RadioGroup a:id="  "
        :
        :      >
    <RadioButton a:....
            :
            :
              />
    <RadioButton a:--
            :
            :/>
        =
        -
</RadioGroup>
```

7. Spinner:

→ Which can be used to design dropdown menu.

Syntax:

```
<Spinner  a:id = "@+id/spinner1"
          a:l-w = " "
          a:l-h = " " />
```

NOTE: Through string-array resource items can be added to Spinner

8. ToggleButton:

→ used to display a button with ON/OFF Message

Syntax:

```
<ToggleButton  a:id = " "
               a:text = "ToggleButton"
               =
               =
               - />
```

9. SeekBar:

→ Which can be used to display a scroll bar (or) seek bar

Syntax:

```
<seekBar  a:id = "@+id/seekBar1"
          a:l-w = " "
          a:l-h = " " />
```

10. ProgressBar:

→ used to display progress bars.

Syntax:

```
<ProgressBar  a:id =
              a:l-w =
              a:l-h =
              Style = " ?android:attr/progressBarStyle
                        ↓ Small " />
```

used to display a small progress Bar

(OR)

```
Style = " ?android:attr/progressBarStyle
                        Large "
```

↓

used to display large circular progressBar

(OR) style = "?android:attr/progressBarStyleHorizontal" → used to display the progressBar in Horizontal format.

NOTE:

→ By default it will display medium circular progressBar.

11. RatingBar:

→ used to display the Rating.

Syntax:

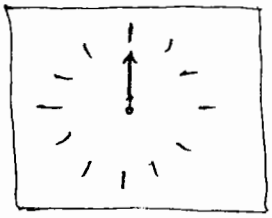
```
<RatingBar a:id=
        a:l-w=" "
        a:l-h=" " />
```

12. AnalogClock:

→ used to display an Analog clock.

Syntax:

```
<AnalogClock a:id=
        a:l-w=" "
        a:l-h=" "/>
```

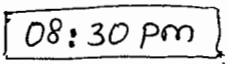


13. DigitalClock

→ used to display a digital clock.

Syntax:

```
<DigitalClock a:id="@+id/digitalclock1"
        a:l-w=" "
        a:l-h=" "/>
```

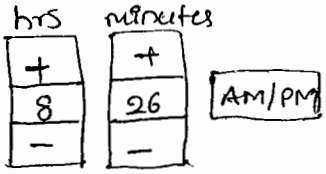


14. TimePicker:

→ which can be used to display a time picker using which we can set the time.

syntax:

```
<TimePicker a:id=" "
        a:l-w=" "
        a:l-h=" "/>
```

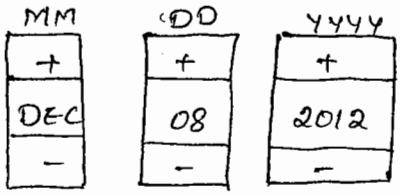


15. DatePicker:

→ Which can be used to display a Date picker dialogue from which date can set.

Syntax:

```
<DatePicker a:id=" "
            a:l-w=" "
            a:l-h=" "/>
```



10/12/2012

16. ImageView:

→ Which can be used to load a foreground image on the android screen.

Syntax:

```
<ImageView a:id=" ____"
           :
           a:src="@drawable/imgname"/>
```

17. ImageButton:

→ Which can be used to set an image as a button.

Syntax:

```
<ImageButton a:id=" ____"
             a:src="@drawable/imgname"
             :
             a:onClick=" ____ "/>
```

18. Gallery:

→ used to set collection of images as a single container

Syntax:

```
<Gallery a:id=" ___"
         :>
    <Imageview a:id="___"
               :/>
    <Imageview a:id="___"
               :/>
</Gallery>
```

19. ZoomButton:

→ used to provide the zoom in option.

Syntax:

```
<ZoomButton  a:id="___"
        :
        />
```

20. Zoomcontrols:

→ Used to provide zoom in and zoom out buttons.

Syntax:

```
<Zoomcontrols  a:id="___"
        :
        />
```

21. Videoview:

→ Which provides a surface to watch the videos.

Syntax:

```
<videoview  a:id="___"
        :
    a:src="___" />
```

22. Webview:

→ Which can be used to provide a surface to watch the webpages.

Syntax:

```
<Webview  a:id="___"
        :
        />
```

23. Mapview:

→ Which provides a surface to watch the Google maps

Syntax:

```
<Mapview  a:id="___"
        :
        />
```

24. SurfaceView :

Which provides a surface to watch the camera preview

Syntax :

```
< SurfaceView a:id = "____"
        |
        |  />
```

25. ListView :

Which can be used to display more than one TextView in a list format

Syntax :

```
< ListView a:id = "____"
        |
        |  />
```

Layout Management :

→ Layout is a container which can be used to arrange the elements in a specific order.

→ In Android the following are the mostly used layouts.

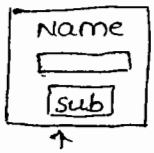
1. LinearLayout
2. TableLayout
3. AbsoluteLayout
4. FrameLayout
5. RelativeLayout

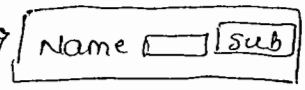
1. LinearLayout :

Which can be used to arrange the view components either in vertical or horizontal direction

Example :

```
< LinearLayout  a:l-w =
                a:l-h =
                a:orientation = "vertical"
                a:ori = "horizontal"
    |>
        < TextView
        |>
```





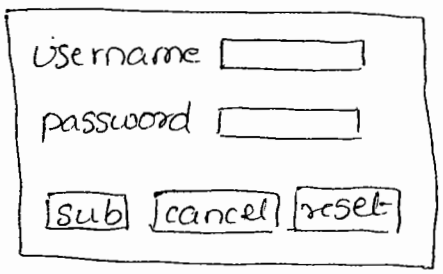
```
<EditText  =
           =/>

<Button    =
           =/>

</LinearLayout>
```

2. TableLayout:

→ Which can be used to display view components in a table format

Example:

```
<TableLayout a:l-w="_"
             a:l-h="_"
             :
             >

  <TableRow  a:l-w="_"
             a:l-h="_"
             :
             >

    <TextView  a:text="_"
               a:l-w="_"
               :
               />

    <EditText  a:id="_"
               :
               />

  </TableRow>

  <TableRow  =
             =>

    <TextView  =
               =/>

    <EditText  =
               =/>

  </TableRow>

  <TableRow  =
             =>

    <Button  a:text="sub"
             =
             =/>

    <Button  a:text="cancel"
             =
             =/>

    <Button  a:text="reset"
             =
             =/>

  </TableRow>

</TableLayout>
```

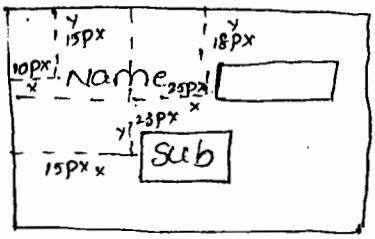


3. AbsoluteLayout:

Which can be used to display view components at any position based on intersection point of x and y coordinate values.

Example:

```
<AbsoluteLayout  a:l-w="__"
                 a:l-h="__"
                 =
                 =>
<TextView  a:text="name"
           =
           =
           a:layout_x="10px"
           a:layout_y="15px"/>
<EditText  a:id="__"
           =
           =
           a:l-x="25px"
           a:l-y="18px"/>
<Button  a:onClick="__"
         =
         =
         a:l-x="15px"
         a:l-y="23px"/>
</AbsoluteLayout>
```

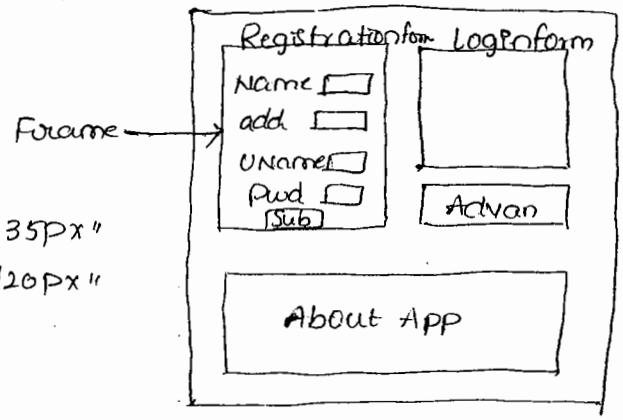


4. FrameLayout:

Which can be used to display set of view components in different frames of any of Layout

Example:

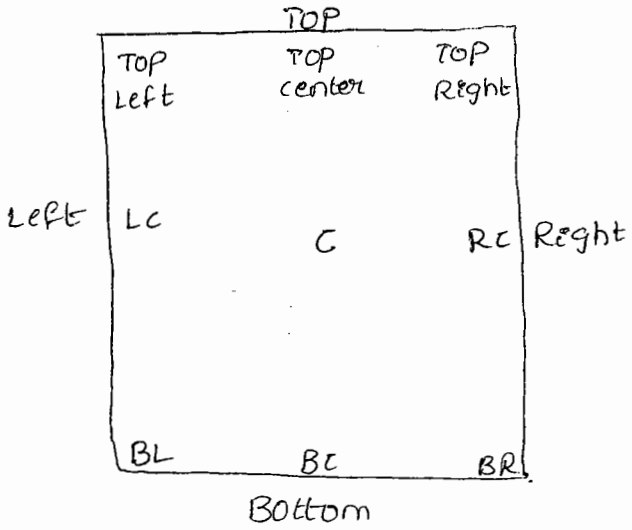
```
<LinearLayout>
   <FrameLayout  a:l-h="35px"
                 a:l-w="20px"
                 =
                 =>
      <TextView  =
                 =/>
   </FrameLayout>
```



```
<FrameLayout ==>
    ==
</FrameLayout>
    ==
</LinearLayout>
```

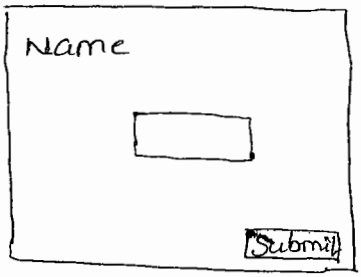
5. RelativeLayout:

→ Which can be used to display any view components in the relative positions.

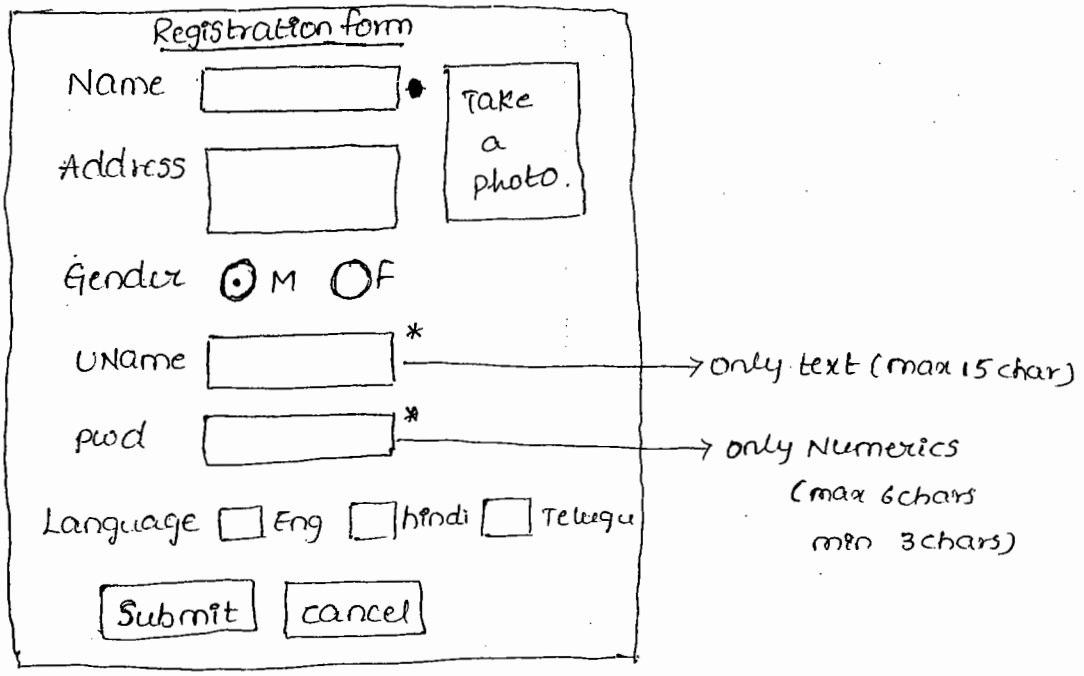


Example :

```
<RelativeLayout a:l-w="__"
                ==>
    <Textview a:text="Name"
              ==
              a:l-alignParentTop="true"
              a:l-alignParentLeft="true"/>
    <EditText a:id="___"
              ==
              a:l-centerHorizontal="true"
              a:l-centerVertical="true"/>
    <Button a:text="Submit"
            ==
            a:l-alignParentBottom="true"
            a:l-alignParent-Right="true/>
</RelativeLayout>
```



Design following Registration form using any Layout except LinearLayout



first.xml

```
<AbsoluteLayout  a:l-w="f-p"
                 a:l-h="f-p"
                 a:background="#ffffff">
  <TextView  a:text="name"
             a:l-w="5px"
             a:l-h="10px" />
  <TextView  a:text="Registration form"
             a:l-w="w.c"
             a:l-h="w.c"
             a:l-x="101dp"
             a:l-y="3dp" />
  <EditText  a:id="@+id/et1"
             a:l-w="  "
             a:l-h="  " />
</AbsoluteLayout>
```

```
<Button a:text = "J">

<TextView a:l_w = "w_c"
          a:l_h = "w_c"
          a:l_x = "102dp"
          a:l_y = "4dp"
          a:text = "___" />

<ImageButton a:id = "@+id/ImageButton1"
             a:tag = "Take photo"
             a:l_w = "96dp"
             a:l_h = "97dp"
             a:l_x = "203dp"
             a:l_y = "43dp"
             a:src = "@drawable/ic_launcher" />

<TextView a:l_w = "w_c"
          a:l_h = "w_c"
          a:l_x = "15dp"
          a:l_y = "165dp"
          a:text = "Gender" />

<RadioGroup a:l_w = "w_c"
            a:l_h = "w_c" >
  <RadioButton a:id = "@+id/rb1"
               a:l_w = "90dp"
               a:l_h = "w_c"
               a:l_x = "76dp"
               a:l_y = "162dp"
               a:text = "male" />
  <RadioButton a:id = "@+id/rb2"
               a:l_w = "w_c"
               a:l_h = "w_c"
               a:l_x = "167dp"
               a:l_y = "161dp"
               a:text = "Female" />
</RadioGroup>
```

```
<TextView a:id="@+id/textview4"
        a:l-w="w-c"
        a:l-h="w-c"
        a:l-x="2dp"
        a:l-y="256dp"
        a:text="password: *"
        a:textAppearance="?android:attr/textAppearanceMe-
                                                    dium"/>
<TextView a:id="@+id/textviews"
        a:l-w="w-c"
        a:l-h="w-c"
        a:l-x="0dp"
        a:l-y="300dp"
        a:text="Languages"
        a:textAppearance="?android:attr/textAppearanceMe
                                                    dium"/>
<checkBox a:id="@+id/checkBox1"
        a:l-w="w-c"  a:l-h="w-c"
        a:l-x="95dp"  a:l-y="302dp"
        a:text="English"/>
<checkBox a:id="@+id/checkBox2"
        a:l-w="w-c"  a:l-h="w-c"
        a:l-x="180dp"  a:l-y="305dp"
        a:text="Telugu"/>
<checkBox a:id="@+id/checkBox3"
        a:l-w="w-c"  a:l-h="w-c"
        a:l-x="100dp"  a:l-y="330dp"
        a:text="Hindi"/>
<EditText a:id="@+id/editText3"
        a:l-w="w-c"  a:l-h="w-c"
        a:l-x="109dp"  a:l-y="209dp"
        a:ems="10"  a:inputType="textPersonName"/>
```

```
<EditText a:id="@+id/editText2"
        a:layout_width="110dp" a:layout_height="50dp"
        a:layout_x="79dp" a:layout_y="100dp"
        a:ems="10"
        a:inputType="textPostalAddress" />
</AbsoluteLayout>
```

Activity class

```
public class Act1 extends Activity
{
    @Override
    protected void onCreate(Bundle b){
        super.onCreate(b);
        setContentView(R.layout.first);
    }

public void subbtn(View v)
{
    EditText e1=(EditText)findViewById(R.id.editText3);
    String s1= e1.getText().toString();
    EditText e2= (EditText) findViewById(R.id.editText4);
    String s2= e2.getText().toString();
    int i1= s1.length();
    int i2= s2.length();
if( i1 !=0 && i1<=15)
    {
        if(i2>=3 && i2<=15){
            Toast.makeText(this,"Registration Success",
                5000).show();
        }
        else { Toast.makeText(this,"enter correct pwd",
                3000).show(); }
    }
else{ Toast.makeText(this,"enter correct username",
                5000).show(); }
} }
```

```
<EditText a:id="@+id/editText4"
    a:l_w="w_c" a:l_h="w_c" a:l_x="109dp" a:l_y="252dp"
    a:inputType="numberPassword"
    a:ems="10" a:textColor="#000000" />

<EditText a:id="@+id/editText1"
    a:l_w="98dp" a:l_h="w_c"
    a:l_x="87dp" a:l_y="47dp" a:ems="10" />

<TextView a:id="@+id/textView2"
    a:l_w="w_c"
    a:l_h="w_c"
    a:l_x="4dp" a:l_y="104dp"
    a:text="Address:"
    a:textAppearance="?a:attr/textAppearanceMedium" />

<TextView a:id="@+id/textView"
    a:l_w="w_c" a:l_h="w_c"
    a:l_x="8dp" a:l_y="49dp" a:text="Name: *"
    a:textAppearance="?a:attr/textAppearanceMedium" />

<TextView a:id="@+id/textView3"
    a:l_w="w_c" a:l_h="w_c"
    a:l_x="2dp" a:l_y="217dp"
    a:text="username: *"
    a:textAppearance="?android:attr/textAppearanceMedium" />

<Button a:id="@+id/button1"
    a:l_w="w_c" a:l_h="w_c"
    a:l_x="60dp" a:l_y="360dp"
    a:text="Submit" a:onClick="Subbtn" />

<Button a:id="@+id/button2"
    a:l_w="w_c" a:l_h="w_c"
    a:l_x="148dp" a:l_y="359dp"
    a:text="cancel" />
```

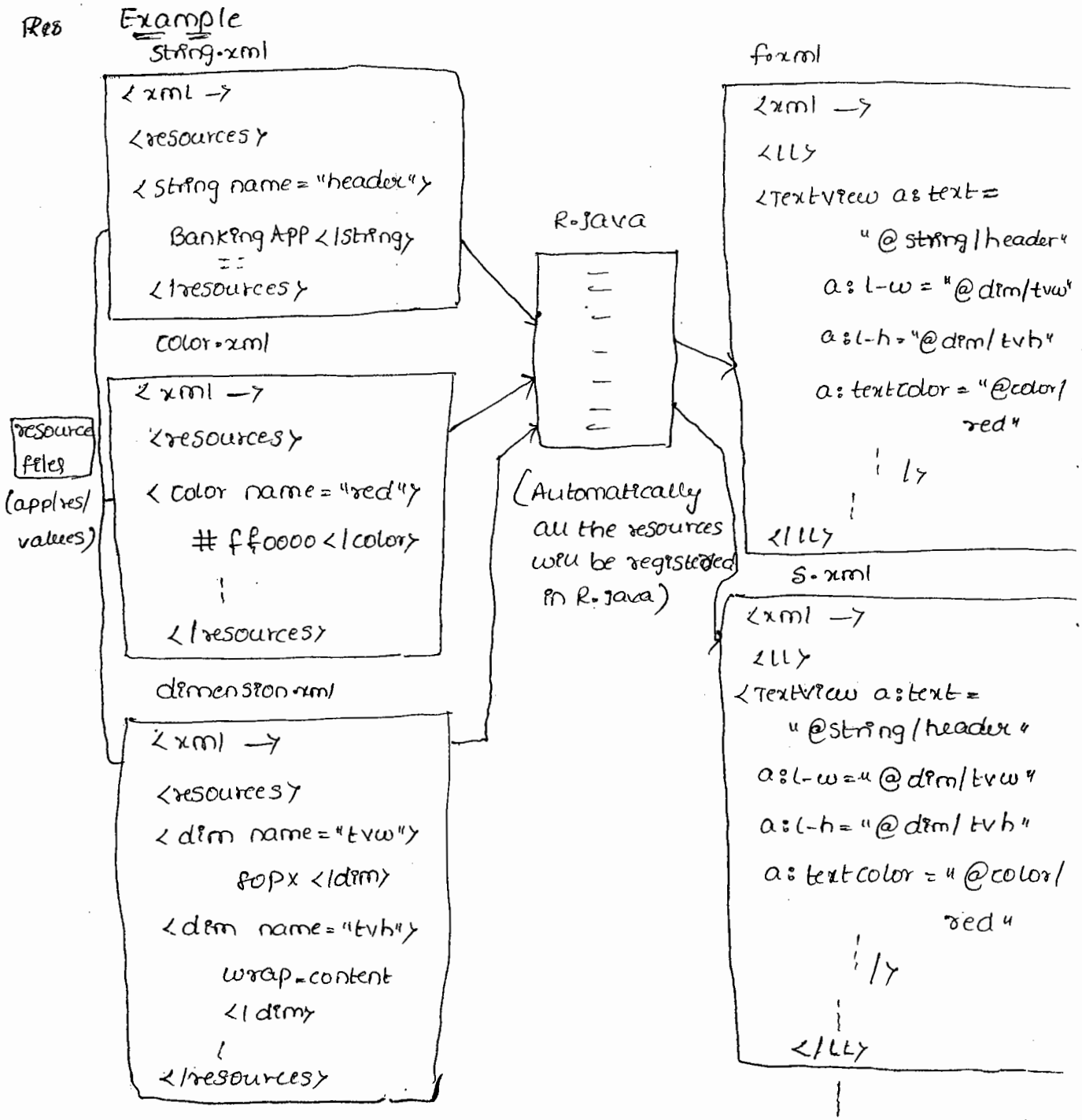
11/12/2012

Resources:

→ Resource is an element which is used commonly in multiple layout files and activity classes.

→ The Main advantage with resources is burden on the developer is reducing while developement of an application and also readability of the program will be increased

→ In android following resources can be used commonly in both layout files and activity classes



Res type	Location	Syntax	How to access	Description
1. String resource	app/res/values/strings.xml	<?xml ...> <resources> <string name= ...> value </string> <string name= ...> value </string> : </resources>	In Layout @string/name of string In Activity class R.string.name of string	Which can be used to make any string value as a common resource for all layouts & Activity classes.
2. String-array resource (underscore)	app/res/values/string-array.xml	<?xml ...> <resources> <string-array name="countries"> <item name="In">India</item> <item name="US">United States of America</item> : </string-array> </resources>	In Layout @string-array/name of string-array In Activity class R.string-array.name of string-array	Which can be used to make set of string values common for all layouts and Activities.

Resource type	Location	Syntax	How to access	Description
3. Integer resource	app/res/values/ Integer.xml	<xml —> <resources> <Integer name = —> numeric value ← value </Integer> : </resources>	In Layout @Integer/ name of Integer In Activity class R.Integer.name of Integer	Which can be used to make an integer value common for all layouts and activities.
4. Integer-array resource	app/res/values/ Integer-array.xml	<xml —> <resources> <Integer-array name = "Age"> <item name="21">21 </item> : </Integer-array> </resources>	In Layout @Integer-array/ name of Integer-array In Activity class R.Integer-array.name of Integer-array	Which can be used to make set of Integer values as common for all layouts & activities.
5. boolean resource	app/res/values/ boolean.xml	<xml —> <resources> <bool name = —> true/false ← value </bool> <bool name = —> value </bool> : </resources>	In layout @bool/ name of boolean resource In Activity class R.bool.name of boolean resource	Which can be used to make a boolean value as common for layouts & activities.

Resource type	Location	Syntax	How to access	Description
6) Image resource	app/res/drawable/ Img.—	—	In Layout @drawable/Image name In Activity class R.drawable.Image name	Which can be used to make an image as common for all layouts and activities
7) Audio (or) video resource	app/res/raw/ Audio (or) video.— fn	—	In Layout @raw/audio (or) video file name In Activity class R.raw.audio (or) video filename	Which can be used to make an audio (or) video file as common for all layouts & activities.
8) Color resource	app/res/values/ color.xml	<xml> <resources> <color name= "red"> #ff0000 [value] </color> : </resources>	In Layout @color/name of the color In Activity class R.color.name of the color	Which can be used to make any color as a common resource for both Layouts & Activity classes
9) Dimension resource	app/res/values/ Dimension.xml	<xml —> <resources> <dim name = "width"> 80px </dim> </resources>	In Layout @dim/name of the dimension In Activity class R.dim.name of the dimension	Which can be used to make dimension values (both (width & height) can be used as common for layouts & activities

12/12/2012

Styles and Themes

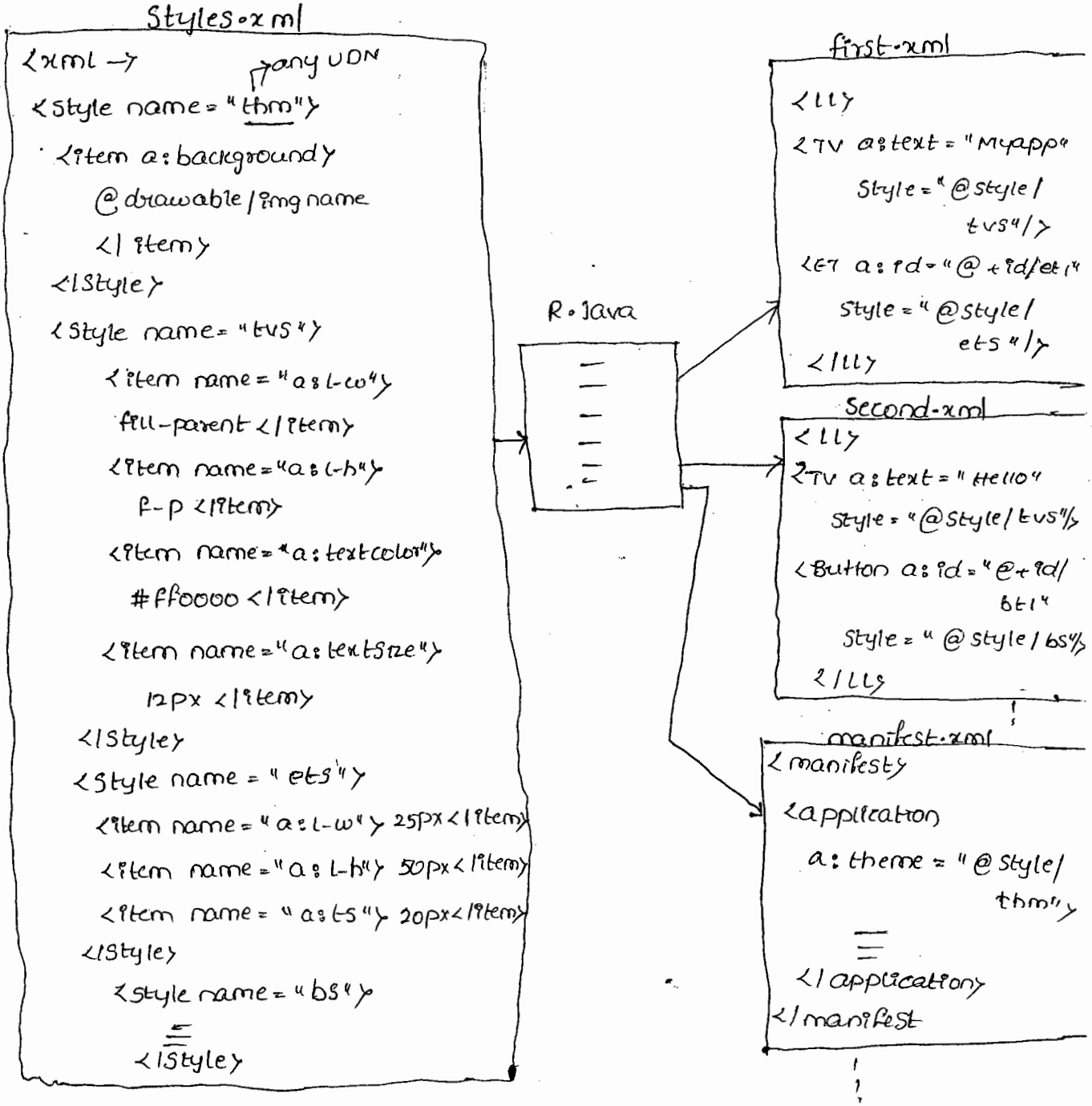
→ Style is set of predefined properties commonly used for Multiple Layout files.

→ Theme is a common background image or color for all layout files.

→ Style or Theme should be declare in a seperate xml file and that is stored in following location.

apps/res/values/styles.xml (or)

apps/res/styles/styles.xml



→ Theme always should be applied in manifest.xml file. so that that will be applied to all layout files

Menu Designing :

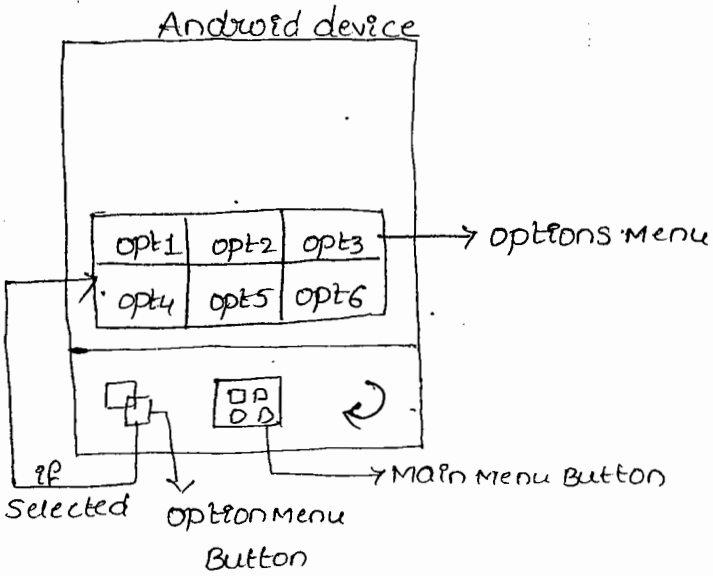
Menu is collection of options, if any option is selected that related operation will be performed.

In android Menu's are classified into two types

1. Options Menu
2. Context Menu

1. OptionsMenu :

→ In every android device a pre defined optionsMenu button is available. if that is clicked a Menu with set of options will be opened technically called as "options Menu."



→ In general options Menu can be created for a specific Layout file (Activity).

→ In Android Application development options Menu can be created in two different ways.

i) With Text options

ii) With Image options.

→ An options Menu with text options can be designed with the help of onCreateOptionsMenu() method and onOptionsItemSelected() method.

→ options Menu with Image options can be designed not only with the help of above methods along with "menu.xml" file.

→ The above two methods are predefined in Activity class and that should be overridden in the userdefined Activity class.

Syntax:

```
class classname extends Activity
  {
    P v onCreate(-)
    {
      =
    }
  Public boolean onCreateOptionsMenu(Menu mobj)
    {                                  ↓ used to create options Menu.
      =
      =
      =
      return true;
    }
  public boolean onOptionsItemSelected(MenuItem mobj)
    {                                  ↓ used to perform an operation
      =                                  whenever option is selected
      =                                  in the menu.
      =
      return true;
    }
  }
```

→ In the above syntax,

Menu is a predefined class will give a logical Menu to which all options can be added with the following syntax.

int values

mobj.add(groupId, itemId, orderNo, "optionname");

itemId → Should be unique

orderNo → Start with zero (0).

groupId → for set of options one groupId is given to represent a seperate group

itemId → An unique id can be given to identify individual item

orderno → represents order of Items (or) options in the Menu.

Design an android appln to display options Menu with Text Items.

first.xml

first.xml

```
<LL  a:l-w = "f-p"
     a:l-h = "f-p" >
   <Textview  a:text = "Welcome to options Menu"
              a:l-w = "w-c"
              a:l-h = "w-c"
              a:textColor = "#ff0000" />
</LinearLayout>
```

Activity class

Act1.java

```
p c  Act1 extends Activity
  {
    @override
      p v  onCreate(Bundle b)
        {
```

```
    super.onCreate(b);
    setContentView(R.layout.first);
  }
public boolean onCreateOptionsMenu(Menu m)
  {
  m.add(1, 1, 3, "chittoor");
  m.add(1, 2, 0, "hyderabad");
  m.add(2, 3, 1, "chennai");
  m.add(2, 4, 2, "Banglore");
  return true;
  }
Public boolean onOptionsItemSelected(MenuItem mi)
  {
  Switch(mi.getItemId())
    {
    case 1: Toast.makeText(this, "you are selected
                chittoor", 5000).show();
            break;
    case 2: Toast.makeText(this, "you are selected
                hyderabad", 3000).show();
            break;
    case 3: Toast.makeText(this, "you are selected
                chennai", 4000).show();
            break;
    case 4: Toast.makeText(this, "you are selected
                Banglore", 5000).show();
            break;
    }
  return true;
}
```

13/12/2012

Option Menu designing With Images:

→ To design an option menu with Images We must create Menu.xml file in application / res/ menu folder. and in which all the images should be registered with the following syntax.

menu.xml

Syntax:

```
<xml →
  <menu xmlns:android>
    <item a:id="@+id/_"
          a:icon="@drawable/imagename"
          a:title="optionname" />
                    ↳ any UON
    :
    :
  </menu>
```

→ This menu will be registered with R.java by default

→ In activity class with the help of MenuInflater class the image options of menu.xml can be added to the logical menu

Syntax:

```
MenuInflater objectreference = this.getMenuInflater();
objectreference.inflate(R.menu.menu, menu class objref);
```

represents menu.xml

→ The above code should be written with onCreateOptionsMenu() method.

Design An android application to create option Menu with Images

Step1:

add all images which are used in option Menu in drawable folder.

Step2:

Design main.xml

main.xml

```
<LL a:l-h="f-p"
    a:l-w="f-p"
    a:orientation="vertical">

  <TextView a:l-w="f-p"
            a:l-h="w-c"
            a:text="Menu design with Images"
            a:textSize="35sp"/>
</LinearLayout>
```

px pixel
sp scale pixel
dp density pixel

Step3:

Design Menu.xml

menu.xml

```
<menu xmlns:android="___">
  <item a:title="Browse"
        a:id="@+id/browse"
        a:icon="@drawable/browse"/>
  <item a:title="Camera"
        a:id="@+id/cam"
        a:icon="@drawable/camera"/>
  <item a:title="Clear"
        a:id="@+id/clear"
        a:icon="@drawable/clear"/>
  <item a:title="Edit"
        a:id="@+id/edit"
        a:icon="@drawable/edit"/>
  <item a:title="Select"
        a:id="@+id/select"
        a:icon="@drawable/select"/>
```

```
<item a:title="Video"
      a:id="@+id/video"
      a:icon="@drawable/video"/>
</menu>
```

Step 4:

If some Local DTSRS data(proj) is want to copy in then go eclipse to file menu & import the program.

Design Activity class.

```
// MenuDemoActivity.java
P C MenuDemoActivity extends Activity
{
  @override
   P v onCreate(Bundle b)
   {
    Super.onCreate(b);
    setContentView(R.layout.main);
   }
  P boolean onCreateOptionsMenu(menu m)
   {
   MenuInflater mi= this.getMenuInflater();
    mi.inflater(R.menu.menu, m); // Items from
                    menu.xml will be assigned to our menu
                     Class obj.
    return true;
   }
 P boolean onOptionsItemSelected(MenuItem item)
   {
    Switch (item.getItemId())
      {
    case R.id.Browse : Intent i = new Intent(Intent.
      ACTION-VIEW, uri.parse("http://www.google.
                                com"));
        StartActivity(i);
        break;
```

```
case R.id.cam : Toast.makeText(this, "cam is selected",
                    Toast.LENGTH_SHORT).show();
                break;
case R.id.clear : Toast.makeText(this, "clear is selected",
                    5000).show();
                break;
case R.id.edit : Toast.makeText(this, "edit is selected",
                    3000).show();
                break;
case R.id.select : Toast.makeText(this, "select is selected",
                    5000).show();
                break;
case R.id.video : Toast.makeText(this, "video is selected",
                    4000).show();
                break;
    }
    return true;
  }
}
```

Steps to load the android applications in Real Device:

Step 1:

Connect android device with the computer using USB cable.

Step 2:

Turn on USB in the mobile device.

Step 3:

Copy all .apk files of android applications and paste into Android folder of Real device.

Step 4:

Install the copied .apk files by going to following Location. (available in bin folder of Eclipse IDE)

Main menu / My files / Android

Step 5:

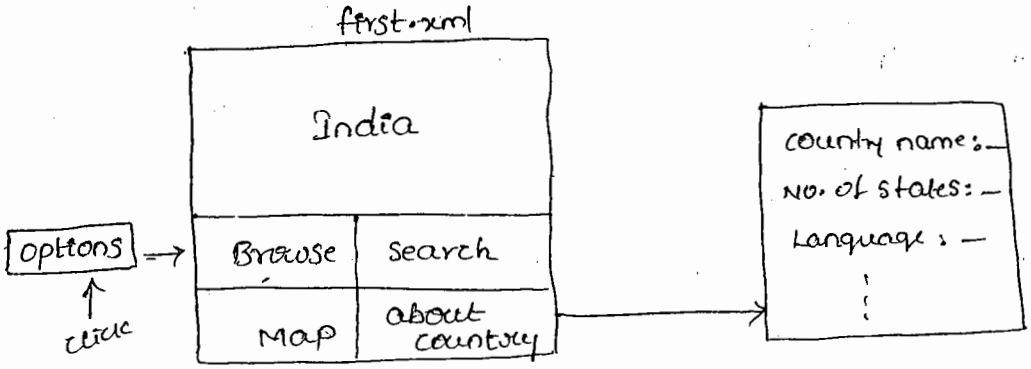
The Instaued android applns are available in the Main Menu

and those can be opened directly.

NOTE:

for first time of connecting real device with a computer it is very mandatory to install driver softwares. (These softwares will be given automatically).

Design an android application for the following



first.xml

```
<LL a:l-w="f-p"
    a:l-h="f-p">
  <TextView a:text="India"
            a:l-w="f-p"
            a:l-w="w-c"
            a:textColor="#ff0000"/>
</LL>
```

Second.xml

```
<LL a:l-w="f-p"
    a:l-h="f-p"
    a:orientation="vertical">
  <TextView a:text="country name : India"
            a:l-w="w-c"
            a:l-h="w-c" />
  <TextView a:text="No. of states in India: 29"
            a:l-w="w-c"
            a:l-h="w-c"/>
```

```
<TextView  a:text = "National flower : Lotus"
           a:l-w = "w-c"
           a:l-h = "w-c" />
<TextView  a:text = "National Game : Hockey"
           a:l-w = "w-c"
           a:l-h = "w-c" />
<TextView  a:text = "National animal : Tiger"
           a:l-w = "w-c"
           a:l-h = "w-c" />
<TextView  a:text = "National Bird : peacock"
           a:l-w = "w-c"
           a:l-h = "w-c" />
<Textview  a:text = "National Language : Hindi"
           a:l-w = "w-c"
           a:l-h = "w-c" />
<TextView  a:text = "president of India : pranab Mukharji"
           a:l-w = "w-c"
           a:l-h = "w-c" />
</LL>
```

Activity class

Act1.java

```
P C Act1 extends Activity
{
  @Override
   P V onCreate(Bundle b)
   {
     Super.onCreate(b);
     SetContentView(R.layout.first);
   }
  Public bodean onCreate OptionsMenu (Menu m)
     {
```

```
m.add(NONE, 1, 0, "Browse");
m.add(NONE, 2, 1, "Search");
m.add(NONE, 3, 2, "Map");
m.add(NONE, 4, 3, "about country");
return true;
}
Public boolean onOptionsItemSelected (MenuItem mi)
{
Switch(mi.getItemId())
{
case 1: Ient i = new Intent(Intent.ACTION_
VIEW, uri.parse("http://www.India.
com"));
startActivity(i);
break;
case 2: Intent i1 = new Intent(Intent.ACTION_
WEB_SEARCH);
i1.putextra(SearchManager.QUERY,
"India");
startActivity(i1);
break;
case 3: Intent i2 = new Intent(Intent.ACTION_
VIEW, uri.parse("geo:0,0?q=India"));
startActivity(i2);
break;
case 4: Intent i3 = new Intent(this, Act2.class);
startActivity(i3);
break;
}
return true;
}
}
```

Act2.java

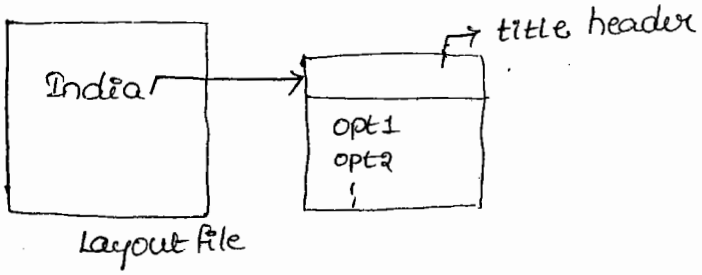
```
p c Act2 extends Activity
{
  @Override
  p v onCreate(Bundle b)
  {
    super.onCreate(b);
    setContentView(R.layout.second);
  }
}
```

15/12/2012
17/12/2012

Context menu:

→ context menu is also collection of options, which will be specific to a view component in a layout file.

→ context menu always will be opened while pressing any view component for a long time in the layout.

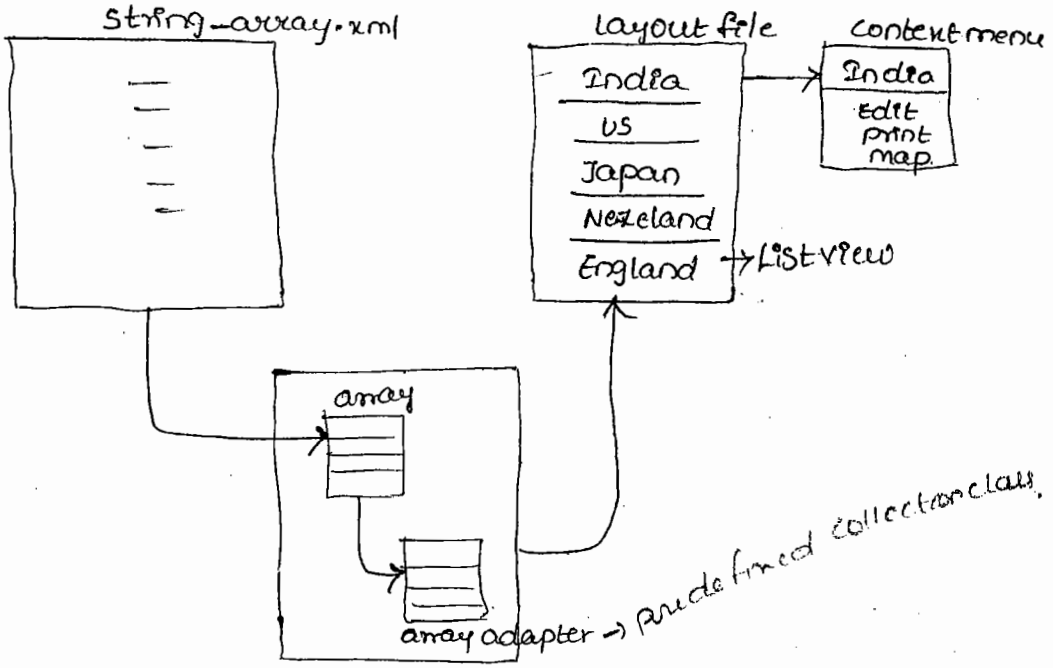


→ options of context menu should be text items (Images are not accepted).

→ context menu can be designed with the help of "onCreateContextMenu()" method (This is predefined method of Activity class)

→ onContextItemselected() method can be used to perform a dynamic operation while selecting an option in the context menu.

Design an android application to display the context menu while selecting an item in the ListView.



main.xml

```
<LL  a:l-w = "f-p"
     a:l-h = "f-p"
     a:orientation = "vertical">
<Listview a:id = "@android:id/list"
          a:l-w = "w-c"
          a:l-h = "w-c" />
</LinearLayout>
```

Strings.xml

```
<resources>
   <String name = "hello"> Hello world, contextMenu
                 Activity! </String>
   <String name = "app_name"> contextMenu </String>
</resources>
```

String (underscore) array.xml

```
<resources>
  <string-array name = "countries">
      <item name = "c1"> India </item>
```

```
<item name="c2">America</item>
<item name="c3">Japan</item>
<item name="c4">Newzealand</item>
<item name="c5">England</item>
</string-array>
</resources>
```

// Activity class (ContextMenuActivity.java)

```
Package com.app.contextmenu;
= } import statements
P C ContextMenuActivity extends Activity
{
  ListView lv;
  String conts[];
  @Override
  P V onCreate(Bundle b)
  {
    Super.onCreate(b);
    setContentView(R.layout.main);
    lv = this.getListView();
    conts = getResources().getStringArray(R.array.coun
                                          tries);
ArrayAdapter<String> adapter = new ArrayAdapter<String>(
                this, android.R.Layout.simple_list_item_1,
                                          conts);
  lv.setAdapter(adapter);
  registerForContextMenu(lv); // used to display contextMenu
                              Whenever we select ListView items
  }
  @Override
  P V onCreateContextMenu(ContextMenu menu, View v,
                          ContextMenuInfo menuInfo)
  {
AdapterContextMenuInfo acm = (AdapterContextMenuInfo)menuInfo;
menu.setHeaderTitle(conts(acm.position));
                    used to display headers for contextMenu
```

```
menu.add(Menu.NONE, 1, Menu.NONE, "Edit");
menu.add(Menu.NONE, 2, Menu.NONE, "Print");
menu.add(Menu.NONE, 3, Menu.NONE, "Map");
super.onCreateContextMenu(menu, v, menuInfo);
}
@Override
public boolean onContextItemSelected(MenuItem item)
{
switch(item.getItemId())
{
case 1: Toast.makeText(this, "Selected Edit",
Toast.LENGTH_LONG).show();
break;
case 2: Toast.makeText(this, "Selected print",
Toast.LENGTH_LONG).show();
break;
case 3: Toast.makeText(this, "Selected Map",
Toast.LENGTH_LONG).show();
break;
}
return super.onContextItemSelected(item);
}
}
```

// manifest.xml

```
<manifest>
<uses-permission a:name="android.permission.INTERNET"/>
<application a:icon="@drawable/ic_launcher"
a:label="@string/app_name">
<activity a:name=".contextMenuActivity"
a:label="@string/app_name">
<intent-filter>
<action a:name="android.intent.action.MAIN"/>
<category a:name="android.intent.category.LAUNCHER"/>
</i-f></activity>
</application>
</manifest>
```

Dialogs:

→ Dialog is an alert message or notification message to the end user, In Android these are classified into following types.

i) ToastDialog
ii) AlertDialog
iii) InputDialog
iv) DateDialog
v) Time/TimePickerDialog
vi) ProgressDialog

i) ToastDialog:

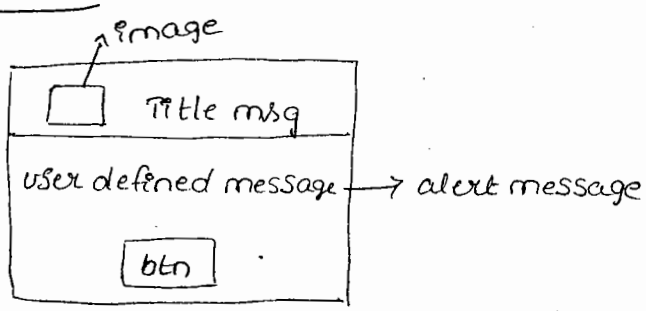
→ Which can be used to display a Toast message, which will be appeared and disappeared automatically.

Syntax:

```
Toast.makeText(context, "message", duration).show();
                  ↓
                 this
```

ii) AlertDialog

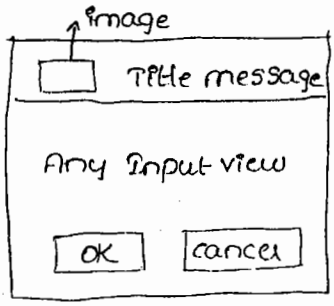
→ Which can be used to display an alert dialog box with single "positive button"



→ This dialog can be designed with the help of "AlertDialog.Builder" class.

iii) InputDialog

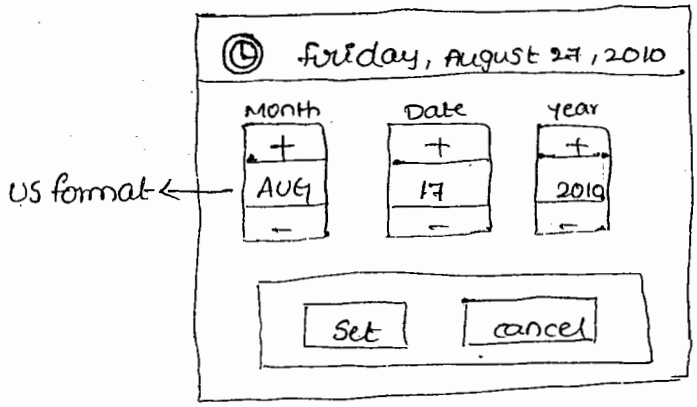
→ Which can be used to display alert message by accepting input from the end user in the form of any view component.



→ Which can be also designed with the help of "AlertDialog.Builder" class

iv) DatePickerDialog / DateDialog

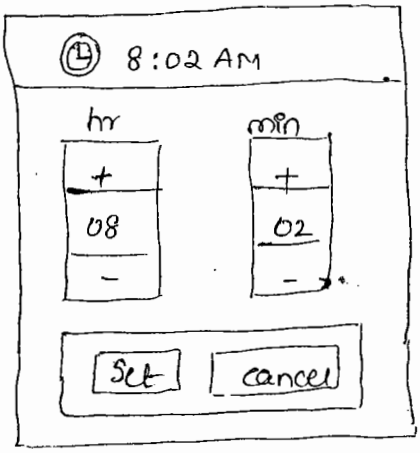
→ Which will give the provisioning to the end user to set a new date.



→ This Dialog can be designed with the help of "DatePickerDialog" class.

v) TimeDialog

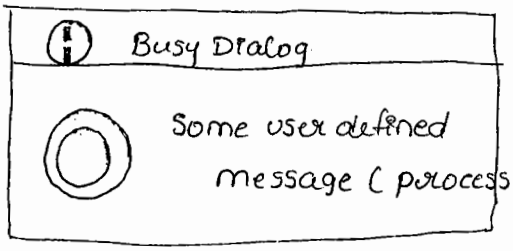
→ Which will give the provisioning to the end user to pick and set the new time.



→ Which can be designed with the help of "TimePickerDialog" class

vii) ProgressDialog

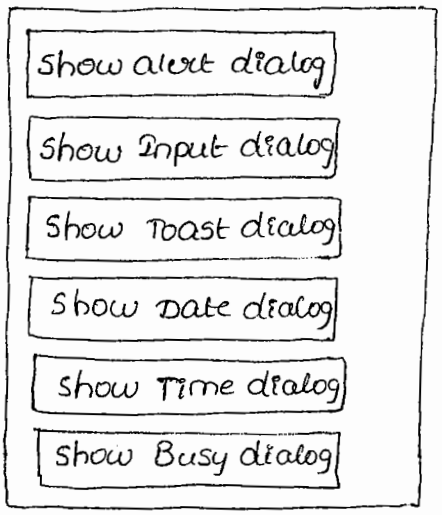
→ Which can be used to display busy message to the enduser.



eg:

→ This Dialog can be designed with the help of "ProgressDialog" class

Design an android application to display dialog messages as shown below.



main.xml.

```
<LL  a:l-w = "f-p"
     a:l-h = "f-p"
     a:Orientation = "vertical"
     a: background = "#ffffff">
<Button  a:id = "@+id/button1"
         a:l-w = "w-c"
         a: l-h = "w-c"
         a: text = "Show alert dialog"/>
<Button  a:id = "@+id/button2"
         a: l-w = "w-c"
         a: l-h = "w-c"
         a: text = "Show Input dialog"/>
<Button  a:id = "@+id/button3"
         a:l-w = "w-c"
         a:l-h = "w-c"
         a: text = "Show Toast dialog"/>
<Button  a:id = "@+id/button4"
         a:l-w = "w-c"
         a: l-h = "w-c"
         a: text = "Show Date dialog"/>
<Button  a:id = "@+id/button5"
         a: l-w = "w-c"
         a: l-h = "w-c"
         a: text = "Show Time dialog"/>
```

```
<Button a:id = "@+id/button6"
        a:l-w = "w-c"
        a:l-h = "w-c"
        a:text = "Show Busy dialog"/>
</LinearLayout>

// Activity class ( DialogsExActivity.java)
    Package com.dialogs;
        = // import stmts.
    Public class DialogsExActivity extends Activity implements
                OnClickListener
    {
        Button b1, b2, b3, b4, b5, b6;
        @override
        P v onCreate(Bundle b)
        {
          super.onCreate(b);
          setContentView(R.layout.main);
        b1 = (Button) findViewById(R.id.button1);
        b2 = (Button) findViewById(R.id.button2);
        b3 = (Button) findViewById(R.id.button3);
        b4 = (Button) findViewById(R.id.button4);
        b5 = (Button) findViewById(R.id.button5);
        b6 = (Button) findViewById(R.id.button6);
          b1.setOnClickListener(this);
          b2.setOnClickListener(this);
          b3.setOnClickListener(this);
          b4.setOnClickListener(this);
          b5.setOnClickListener(this);
          b6.setOnClickListener(this);
        }
        @override
```

```
Public void onClick (View v)
 {
   if (v.getId() == R.id.button1) {                    display alert dialog
AlertDialog.Builder adb = new AlertDialog.Builder(this); ]
   adb.setIcon( R.drawable.ic_launcher);
   adb.setTitle( "This is my Alert Dialog");
   adb.setMessage( "This is your message");
   adb.setPositiveButton( "OK",
          new DialogInterface.OnClickListener()
           {
            @Override
            P v onClick( DialogInterface dialog, int which)
             {
               // defining action for OK button
             }
           });
      adb.show();
 }
else if (v.getId() == R.id.button2) {
 AlertDialog.Builder adb = new AlertDialog.Builder(this);
   adb.setTitle( "This is Input Dialog");
   final EditText et = new EditText (this);
     adb.setView(et);
     adb.setPositiveButton( "OK",
            new DialogInterface.onClickListener(){
            @Override
             P v onClick ( DialogInterface dialog, int which)
               {
                 // action for OK button
               }
            });
       adb.setNegativeButton( "cancel",
            new DialogInterface.onClickListener() {
              @Override
           P v onClick ( DialogInterface dialog, int which)
             {
```

```
        // action for CANCEL button
      }
    }); adb.show();  }
  else if ( v.getId() == R.id.button3) {
Toast.makeText( this, "This is Toast", 3000).show(); // Displaying Toast
                                                          dialog
    }
  else if (v.getId() == R.id.button4) {
Toast.makeText (this, DatePickerDialog dpd = new DatePickerDia
                                    log( this, null, 2010, 7, 27); // Displa
    dpd.show();                                          ying DatePicker
  }                                                        Dialog.
  else if (v.getId() == R.id.button5) {
TimePickerDialog tpd = new TimePickerDialog( this, null, 8, 2,
                                                  true); // Displaying
    tpd.show(); }                                    TimePicker Dialog
  else if ( v.getId() == R.id.button6) {
  // Displaying progress Dialog
    final ProgressDialog pd = new ProgressDialog( this );
      pd.setTitle ( "Busy Dialog");
    pd.setMessage ( "processing is going on - plz be patient");
     pd.show();
    new Thread(){ public v run() {
                        // defining task for Progress dialog for busy
                         try {
                           Thread.sleep(2000); pd.dismiss(); }
                        catch(Exception e){
                                              }
                          }}.start();
                      }
                  }
              }
```

manifest.xml

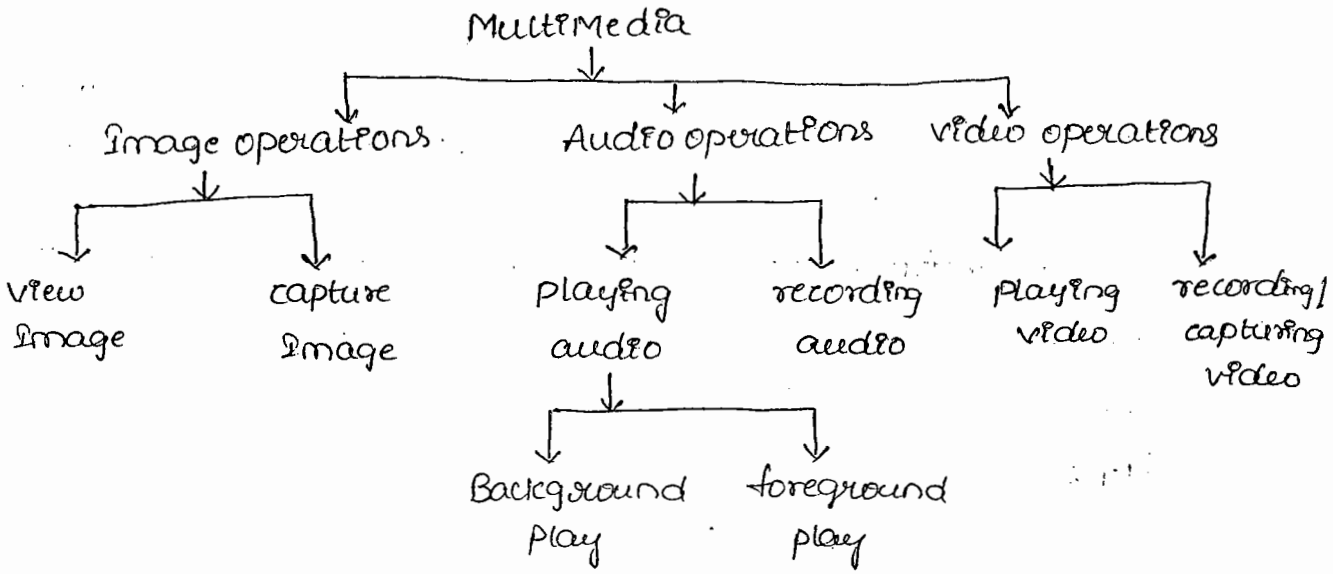
```
<manifest> <application> <activity a:name=".DialogsExActivity">
  <i-f>
   <action a:name = "android.intent.action.MAIN"/>
   <category a:name = "android.intent.category.LAUNCHER"/>
  </i-f> </activity>
 </application> </manifest>
```

18/12/2012

** MULTIMEDIA :

→ In Mobile technology MultiMedia represents audio operations, video operations and Image operations.

→ following hierarchy represents various MultiMedia operations.



Audio Operations :

→ any android device Supports playing audio file and recording audio.

playing Audio files :

→ any audio file which is available in the android device can play either in the background of Android appln or foreground of android device.

Background play

→ To play any Background song android API provides a predefined class called "MediaPlayer"

→ Using MediaPlayer class we can play the songs in the Background in three different ways.

Settings → Unknown resources → enable

a) playing of resource songs:

Medi

→ In this scenario the songs which are available in the "raw" folder can be played in the background.

Syntax:

```
MediaPlayer mp= MediaPlayer.create(context, R.raw.
                                   Songname); // used
                                   to load the song
mp.start(); // used to start the song.
```

Design an android appln to play the audio song in background

Step 1:

add a song to res folder

Step 2:

Design main.xml (Layout file)

```
<LL  a:l-w="f-p" (or) "match-parent"   (underscore)
     a:l-h="f-p" (or) "m-p"
     a:orientation="vertical"/>

<TextView a:l-w="w-c"
          a:l-h="w-c"
          a:textSize="25px"
          a:text="welcome to android session"/>

</LL>
```

Step 3:

Design Activity class

```
Public class MainActivity extends Activity
{
  @override
   P v onCreate(Bundle b)
     {
```

```
    super.onCreate(b);
    setContentView(R.layout.main);
  MediaPlayer mp = MediaPlayer.create(this, R.raw.song);
    mp.start();
  }
  @Override
  public void onDestroy()
  {
    mp.release();
  }
}
```

NOTE:

Whenever problem is occured while executing this application in Real android device then go to settings → unknown resources → and select "enable" option. //

→ In the above syntax release() method can be used to stop the playing of song. whenever activity is destroyed.

by playing of Local songs:

In this scenario we can play local songs which are available in the current android device.

Syntax:

```
Uri myuri = "c:/music/filename.mp3"; // Initialize path
                                        of the audio song
MediaPlayer mp = new MediaPlayer(); // activate player
                                        software
mp.setAudioStreamType(AudioManager.STREAM_MUSIC);
                                        // set to play only audio files
mp.setDataSource(getApplicationContext(), myuri); // used
                                        to load the song
mp.prepare();
mp.start();
```

any objreference

"myfiles/downloads/"+s+".mp3"

underscore

generally here context is 'this' (or) we can write that method to get the context

c) Playing of remote songs:

→ In this scenario we can play any remote song which is available in a server

Syntax:

```
Uri url = " http://---- " ; // path of the song in remote server
MediaPlayer m = new MediaPlayer(); // activate player s/w
m.setAudioStreamType( AudioManager.STREAM_MUSIC );
m.setDataSource( url ); // load the song
m.prepare(); // might be taken long time! (for buffering)
m.start();
```

NOTE:

Internet permission should be given in manifest file to play remote songs

```
<uses-permission
    android:name="android.permission.INTERNET" />
```

19/12/2012

Audio capturing (or) voice Recording:

→ Android API provides a predefined class called "MediaRecorder" to capture audio (or) voice.

→ Following rules should be followed to create an android application for voice recording

1. Create a new instance of android.media.MediaRecorder class
2. Set the audio source using "MediaRecorder.setAudioSource()" (Probably this will be the mic → "MediaRecorder.AudioSource.MIC" is used to enable mic in real android device)

3. Set output file format using

MediaRecorder.setOutputFormat() (Eg: .mp3, .3gp etc.,)

4. Set output file name using

MediaRecorder.setOutputFile() (Eg: myvoice.mp3 etc.,)

(can be of any order)

5. Set the audio encoder using

MediaRecorder.setAudioEncoder()

6. Call MediaRecorder.prepare() on the MediaRecorder instance to prepare mic to capture the voice.

7. To start audio capture, call MediaRecorder.start()

8. To stop audio capture, call MediaRecorder.stop()

9. When you are done with the MediaRecorder instance, call MediaRecorder.release() on it. (Call MediaRecorder.release() to stop voice recording permanently).

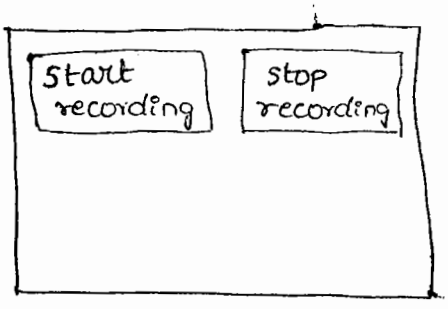
NOTE:

MediaRecorder.release() is always recommended to free the resource immediately rather than stop().

→ following permission should be taken in the manifest file before recording voice through android application.

```
<uses-permission
    android:name="android.permission.
                  RECORD_AUDIO" />
</uses-permission>
```

Design an android application to record the voice



```
//main.xml
<LL a:l_w="f_p"
    a:l_h="f_p"
    a:orientation="horizontal">
<Button a:id="@+id/start"
        a:l_w="w_c"
        a:l_h="w_c"
        a:text="Start Recording"
        a:onClick="StartRecording"/>
<Button a:id="@+id/stop"
        a:l_w="w_c".
        a:l_h="w_c"
        a:text="Stop Recording"
        a:enabled="false"
        a:onClick="StopRecording"/>
</LinearLayout>
```

```
//Activity class (AudioRecordExActivity.java)

    // import stmts.

p c AudioRecordExActivity extends Activity
{
    MediaRecorder recorder;
    private Button startButton;
    private Button stopButton;
    @override
    p v onCreate(Bundle b)
    {
        super.onCreate(b);
        SetContentView(R.layout.main);
        startButton = (Button) findViewById(R.id.start);
        stopButton = (Button) findViewById(R.id.stop);
    }
    p v startRecording(View view) throws IOException
    {
        startButton.setEnabled(false);
        stopButton.setEnabled(true);
```

```
recorder = new MediaRecorder();
recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
recorder.setOutputFormat(MediaRecorder.OutputFormat.
                                        THREE-GPP);
recorder.setAudioEncoder(MediaRecorder.AudioEncoder.
                                        AMR-NB);
recorder.setOutputFile("Sdcard/abc.3gp");
recorder.prepare();
recorder.start();
}
P v stopRecording(View view){
startButton.setEnabled(true);
stopButton.setEnabled(false);
recorder.stop();
recorder.release();
}
}
```

Manifest.xml

```
<manifest><application><activity ==>
<i-f><action —/><category —/></i-f></activity>
</application>
<uses-permission a:name="android.permission.WRITE-EXTERNAL-
                                        STORAGE"/>
<uses-permission a:name="a.permission.RECORD_AUDIO"/>
</manifest>
```

NOTE:

→ Encoding operation is to convert analog signals into digital signals.

NOTE:

→ Generally in all the languages (Java, C, C++ ...) encryption and decryption is used to convert plane text (readable format) into cyber text (unreadable format) and vice versa.

→ Q. When a class is said fully qualified?

Ans: A class is said to be fully qualified whenever it is written along with its packagename.

Image Operations :

→ Images can be captured (creating) and can view on the layout file.

→ Images mainly classified into two types.

1. static Images
2. Dynamic Images.

→ Static Images can view directly on Layout file with the help of <ImageView> tag. and Dynamic Images can view on layout file by capturing using camera device.

20/12/2012

Displaying static Images :

Images are available in drawable folder and these can be displayed in two different ways.

1) through xml

```
<ImageView android:src="@drawable/imagename"
                    :
                    :/>
```

2) using Imageview class

Imageview is a predefined class in android.view package can be used to load the image with the following method.

"SetImageResource()"

Layout file

Eg:
```
<LL>
  <Imageview a:id="@+id/img"
             a:l-w="f-p"
             a:l-h="f-p"/>
</LL>
```

Activity class

```
class <classname> extends Activity
{
    P v onCreate(Bundle b)
    {
        Super.onCreate(b);
        SetContentView(R.layout.xml file name);
    ImageView iv = (ImageView) findViewById(R.id.img);
    iv.setImageResource(R.drawable.imagename);
    }
}
```

Capturing Image

→ Capturing Image can be done using a hardware device called Camera, android API supports some predefined classes to capture an image using camera device.

→ In android, camera based applications mainly depends on following predefined classes.

1) Camera class

Which can be used to enable a hardware camera device.

2) SurfaceView

Which provides a provisioning to the end user to view the outside view within the mobile device.

3) Intent:

Which can be used to achieve the switching from user defined Activity to camera Activity.

⇒ Before developing camera based application, it is very mandatory to enable our application for using hardware camera device. in the manifest file following code should be

written to do this operation.

<uses-feature
android:name="android.hardware.camera"/>

→ In general camera applications follows switching with Result Data and Syntax to enable camera activity with image mode is as shown below.

Syntax:

Intent cameraIntent = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);

Design android appln to capture an image through camera

main.xml

```
1 <ll a:l-w="f_p"
     a:l_h="f_p"
     a:ori="vertical">
2 <Imageview a:id="@+id/photoResultview"
     a:l_w="120dp"
     a:l-h="253dp"
     a:src="@drawable/ic_launcher"/>
3 <Button a:id="@+id/button1"
     a:l-w="w-c"
     a:l-h="w-c"
     a:text="Button"/>
</LinearLayout>
```

// Activity class (TakeAPhotoActivity.java)

```
Package com.takeaphoto;
   // import stmts
P C TakeAPhotoActivity extends Activity implements
      android.view.View.OnClickListener{
Private static final int CAMERA_PIC_REQUEST = 1337;
  @override
  P V onCreate(Bundle b)
   {
    Super.onCreate(b);
    SetContentView(R.layout.main);
```

```
Button b=(Button) findViewById(R.id.button1);
  b.setOnClickListener(this);
}
protected void onActivityResult(int requestCode, int resultC
                                ode, Intent data){
if(requestCode==CAMERA_PIC_REQUEST)
  {
   Bitmap thumbnail=(Bitmap)data.getExtras().get(
                                    "data");
   ImageView image=(ImageView) findViewById(R.id.
                                photoResultview);
   image.setImageBitmap(thumbnail);
  }
}
  @Override
    p v onClick(View arg0){
Intent cameraIntent=new Intent(android.provider.
    MediaStore.ACTION_IMAGE_CAPTURE);
  startActivityForResult(cameraIntent,CAMERA_PIC_RE-
                                QUEST);
    }
  }
```

manifest.xml

```
<manifest><application><activity a:name=".TakeAphoto
                                Activity">
  <i-f><action a:name="a.intent.action.MAIN"/>
       <category a:"  "  .category.LAUNCHER"/>
  </i-f></activity></application>
 <uses-feature a:name="android.hardware.camera"/>
</manifest>
```

21/12/2012

Video Operations :

→ In general mainly we can perform following operations on videos in android device

1. Watching video's.
2. capturing video's

1. Watching (or) viewing video's :

→ Android API provides predefined class to develop the applications to watch the video's on android device.

a) VideoView

→ Which will give the provisioning to the end user to watch the video (or)

→ Which will provide a Surface view to the end user to view the video's.

b) Mediacontroller

→ It is a predefined class used to provide the controls to the videoview.

→ While designing a layout file <videoview> tag can be taken with the following syntax.

Syntax:

```
<LL>
<videoview  a:id="@id/__"
            =
            =/>
</LL>
```

Design an android application to play the video in the foreground

```
<LL  a:l-w="m-p"
     a:l-h="m-p"
     a:ori="vertical">

     <videoview a:id="@+id/
                 Surface_view"
                 (underscore)
                a:l-w="320px"
                a:l-h="240px" />
</LL>
```

Activity class

```
package pl.videoplay;
// import stmts
public class videoPlayActivity extends Activity{
    private VideoView mVideoView;
    @Override
    public void onCreate(Bundle b){
        super.onCreate(b);
        setContentView(R.layout.main);
        mVideoView = (VideoView) findViewById(R.id.Surface_
                                                    View);
mVideoView.setVideoURI(Uri.parse("android.resource://"+
            getPackageName()+"/"+R.raw.samplevideo));
mVideoView.setMediaController(new MediaController(this));
 mVideoView.requestFocus();
  }
}
```

manifest.xml

```
<manifest> <application> <activity a:name=".videoPlayActivi
                                                    ty">
 <i-f> <action —/> <category —/> </i-f>
  </activity> </application>
 </manifest>
```

2. Capturing video's

→ In android device video capturing can be done through camera device, to develope these applications android API provides following predefined classes

1. Camera
2. Surfaceview
3. MediaRecorder

1. **camera**: This class is the primary API for controlling device cameras. This class is used to take pictures or videos when you are building a camera Application.

2. **Surfaceview**: This is used to present a live camera preview to the end user

3. **MediaRecorder**: This class is used to record audio along with video from camera.

4. **Intent**:

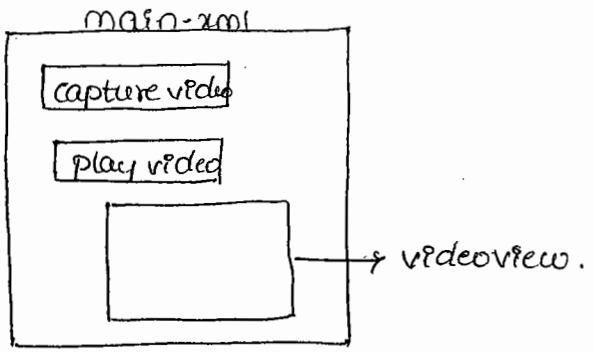
A predefined action type was given in the Intent so that automatically which will open camera with video view.

Syntax:

```
Intent objref = new Intent(android.provider.MediaStore.ACTION_VIDEO_CAPTURE);
```

→ While writing above syntax internally camera, Surfaceview, MediaRecorder classes will be used. so that no need to use these explicitly.

Design an android application to capture video and play that video on android Screen



// main.xml

```
<LL a:l-w="f_p"
    a:l-h="f_p"
    a:ori="vertical">
<Button a:text="capture video"
    a:id="@+id/capturevideo
          Button"
    a:l-w="w-c"
    a:l-h="w-c"/>
<Button a:text="play video"
    a:id="@+id/playvideo
          Button"
    a:l-w="w-c"
    a:l-h="w-c"/>
<videoview a:id="@+id/video-
          view"
    a:l-w="w-c"
    a:l-h="w-c"/>
</LinearLayout>
```

// Activity class (videocaptureActivity.java)

```
package com.videocapt;
   = // import stmts.
P c videocaptureActivity extends Activity implements
      onClickListener {
Button capturevideoButton;
Button playvideoButton;
VideoView videoview;
Uri videoFileUri;
  @override
P v onCreate(Bundle b) {
  super.onCreate(b);
  setContentView(R.layout.main);
captureVideoButton = (Button) this.findViewById(R.id.capture
                          VideoButton);
playVideoButton = (Button) this.findviewById(R.id.playVideo
                          Button);
```

```
videoView = (VideoView) this.findViewById(R.id.videoView);
captureVideoButton.setOnClickListener(this);
playVideoButton.setOnClickListener(this);
playVideoButton.setEnabled(false);
}
public void onClick(View v)
{
if(v==captureVideoButton) {
Intent captureVideoIntent = new Intent(android.provider.
MediaStore.ACTION_VIDEO_CAPTURE);
startActivityForResult(captureVideoIntent, 1);
}
if(v==playVideoButton) {
videoView.setVideoURI(videoFileUri);
videoView.start();
}
}
protected void onActivityResult(int requestCode, int result
code Intent data) {
if(resultCode == RESULT_OK && requestCode==1){
videoFileUri = data.getData();
playVideoButton.setEnabled(true);
}
}
}
```

manifest.xml

```
<manifest> <application> <activity a:name=".VideoCapture
Activity">
<i-f> <action —1> <category —1> </i-f> </activity>
</application> </manifest>
```

22/12/2012

Services:

→ Service is a background operation for any android application, In android service is existing as a class.

→ The main advantage with service is to improve the efficiency of an android application.

→ Services are mainly classified into two types.

1) Predefined Services
2) Userdefined Services

1. predefined Services:

→ If any service class is existing already in the android API and if it is ready to provide the services to any android application is known as predefined service.

Example: AlarmManager, SMSManager, vibrator, LocationManager etc.,

→ following syntax can be used to activate any type of service

Syntax:

```
Service classname objref = (Service classname) getSystemService(Service type);
```

Example:

To activate vibrator following service can be used.

```
Vibrator vb = (vibrator) getSystemService(VIBRATOR_SERVICE);
→ activate vibrator
    Vb.vibrate(3000); // starts vibration
                      ↓
                duration in milliseconds
```

(OR)

```
Vibrator vb = (vibrator) this.getSystemService(context.VIBRATOR_SERVICE);
Vb.vibrate(3000);
```

Design an android application to vibrate current Mobile device with vibrator service.

// main.xml

```
< LinearLayout   a:l-w = " match-parent "
                 a:l-h = " match-parent "
                 a: orientation =" vertical "/>
< Button  a: id = " @+id / bt1 "
          a : text = " vibrate "
          a : onClick = " vibrate btn "
          a: l-w = " w-c "
          a: l-h = "w-c" />
```

// Activity class (VibrationActivity. Java)

```
Public class VibrationActivity extends Activity
 {
   @override
    Public void onCreate ( Bundle b)
     {
      Super. onCreate (b);
      SetContentView (R. layout. main);
     }
    public void vibrate btn (view v)
    {
    Vibrator vb = (vibrator) getSystemService (VIBRA-
                                          TOR_SERVICE);
       vb. vibrate (3000);
     }
 }
```

Another examples of service

Alaram service

Which can be used to enable the alaram in an android device.

AlaramManager alm = (AlaramManager) getSystemService(
ALARAM_SERVICE);

Location service:

Which can be used to find the current Location in google maps.

LocationManager lm = (LocationManager) getSystemService(
LOCATION_SERVICE);

Sending SMS:

Android API provides a predefined class called SmsManager to send Sms from one device to another device.

Syntax to enable SmsManager Service:

SmsManager objref = SmsManager.getDefault();
↓
used to get all the default properties of SmsManager class.

SmsManager objref = (SmsManager) getSystemService(SMS_SERVICE);

→ Sms can send with the following predefined method of SmsManager class.

Syntax:

sm(objref).sendTextMessage(Destphoneno, Msgcentre, textmsg, sentIntent, DeliveryIntent);
*1 *2

*1 → To which number sms should send

*2 → The message which is going to be send.

NOTE:

Message centre, sentIntent, DeliveryIntent values can be null (default values will be taken)

→ Message cent is a service number from which message is going to be delivered.

→ SentIntent is an object reference of PendingIntent class which represents the message which is given after sent to the message cent successfully.

→ DeliveryIntent is also an objectreference of PendingIntent class represents the msg which is to be given to the enduser after delivering sms successfully to the destination Mobile.

PendingIntent:

Which can be used to achieve the switching fuom one activity to other after some pending time.

→ following permission should be taken to send an sms in manifest file.

high fun
<uses-permission

android:name = "android.permission.SEND_SMS"/>

Design an android appln to send an SMS

// main.xml

```
<LL a:id="@+id/LinearLayout1"
    a:l-w="f-p"
    a:l-h="f-p"
    a:ori="vertical">
<TextView a:id="@+id/textViewPhoneNO"
          a:l-w="w-c"
          a:l-h="w-c"
          a:text="Enter phone Number:"/>
<EditText a:id="@+id/editTextPhoreNO"
          a:l-w="f-p"
          a:l-h="W-C"
          a:phoneNumber="true"/>
<TextView a:id="@+id/textViewSMS"
          a:l-w="w-c"
          a:l-h="w-c"
          a:text="Enter SMS Message:"/>
```

```
<EditText a:id="@+id/edit
                TextSMS"
    a:l-w="f-p"
    a:l-h="w-c"
    a:inputType="textMultiLine"
    a:lines="5"
    a:gravity="top"/>
```

```
<Button a:id="@+id/buttonSend"
    a:l-w="f-p"
    a:l-h="w-c"
    a:text="Send"
    a:onClick="onClick"/>
</LinearLayout>
```

// Activity class (SendSMSActivity.java)

```
Package com.sendsms;
    // import stmts
P C SendSMSActivity extends Activity {
    Button buttonSend;
    EditText textPhoneNO;
    EditText textSMS;
    @Override
    P V onCreate(Bundle b) {
        super.onCreate(b);
        SetContentView(R.layout.main);
buttonSend = (Button) findViewById(R.id.buttonSend);
textPhoneNO = (EditText) findViewById(R.id.editTextPhoneNO);
textSMS = (EditText) findViewById(R.id.editTextSMS);
    }
    P V onClick(View v){
      String phoneNO = textPhoneNO.getText().toString();
      String sms = textSMS.getText().toString();
      try {
       SmsManager smsManager = SmsManager.getDefault();
       smsManager.sendTextMessage(phoneNo, null, sms, null, null);
       }
       catch(Exception e) {   }
}}
```

24/12/2012

User Defined Service:

→ If any Service class is designed by the user known as "User Defined Service."

Rules to design Service class:

1. Create any user defined class and extends Service class.
2. Override onBind(), which can be used to establish communication channel between Activity class and Service class.
3. Override Lifecycle Methods (onCreate() ← Mandatory, onStart(), onDestroy()) of service class. [onStart() and onDestroy() are the optional Methods.
4. Save that class with service class name.java

Syntax:

```
class <CN> extends Service
 {
    Public IBinder onBind( Intent obj)
     {
      return null;
     }
    Public void onCreate()
     {
       ==
     }
    Public void onStart(Intent iobj, int startId)
     {
       ==
     }
```

```
Public void onDestroy()
  {
    =
  }
}
```

Life cycle Methods of Service class:

1. onCreate():

→ Which will be executed whenever the service is loaded. and this must be overridden mandatorily in the user defined Service class

2. onStart():

→ Which will be executed whenever a service is running, this will be executed automatically very immediately after onCreate() method.

3. onDestroy():

→ This method will be executed whenever a service is stopped.

NOTE:

onCreate() will be executed only once in the program.

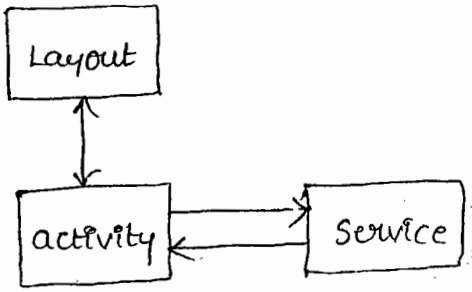
Registering Service class:

Every user defined Service class should be registered in manifest file as shown below

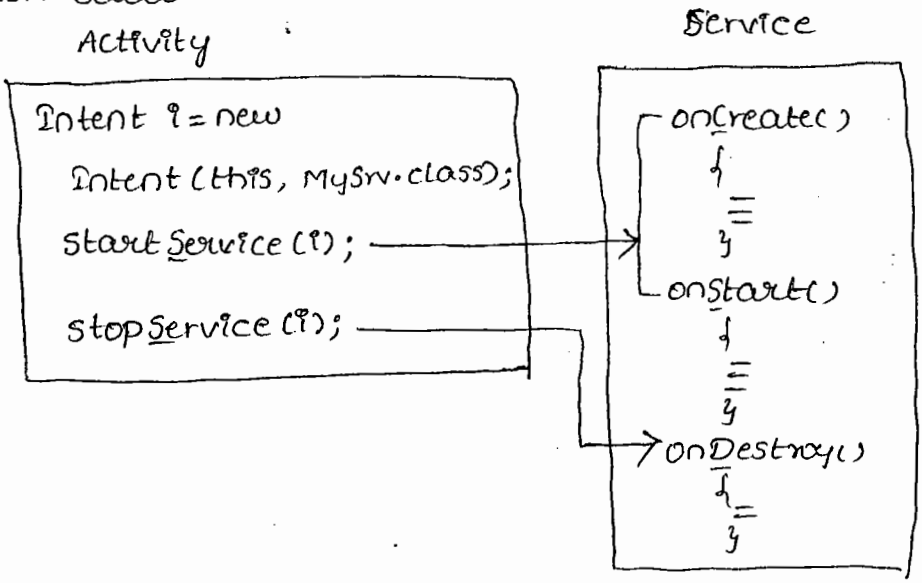
```
<manifest>
   <application>
      <service android:name=".MyService">
      </service>
      :
   </application>
</manifest>
```

Service class name

→ Every Service class will be controlled by Activity class.



→ Switching can be done b/n activity and service class as shown below.



→ In the above syntax startService() can be used to call both onCreate() and onStart() where as stopService() can be used to call onDestroy()

Design an android application to execute Life cycle Methods of Service class

main.xml

```
< LinearLayout  a:l-w = "f-p"
                a:l-h = "f-p"
                a: orientation = "vertical" >
   <Button      a:id = "@+id/button1"
                a:l-w = "w-c"
                a:l-h = "w-c"
                a: text = "startService" />
```

```
<Button  a:id="@+id/button2"
         a:layout_height="wrap_content"
         a:layout_width="wrap_content"
         a:text="stopservice" />
</LL>
```

```
// ServiceLifeCycleActivity.java (Activity class)
public class ServiceLifecycleActivity extends Activity
        implements onClickListener
{
    @override
    public void onCreate(Bundle b)
    {
     super.onCreate(b);
     setContentView(R.layout.main);
     Button b1=(Button) findViewById(R.id.button1);
      b1.setOnClickListener(this);
     Button b2=(Button) findViewById(R.id.button2);
       b2.setOnClickListener(this);
    }
     @override
    public void onClick(View v)
      {
         Intent i;
         if(v.getId()==R.id.button1)
         {
           i=new Intent(this, MyService.class);
            startService(i);
         }
        else if(v.getId()==R.id.button2)
          {
            stopService(i);
    } } }
```

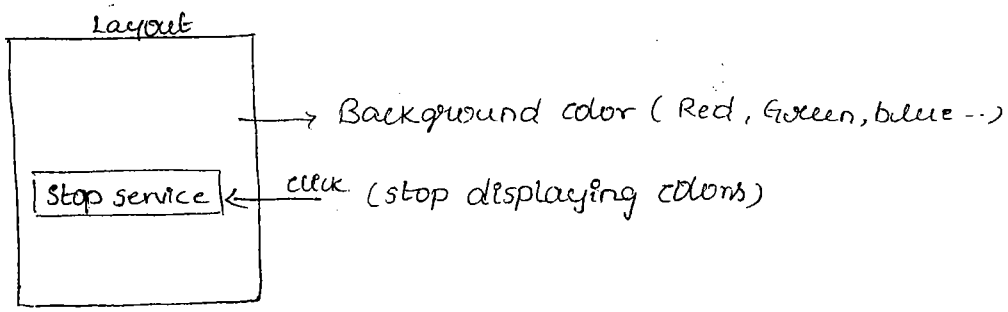
// MyService.java (Service class)

```
Public class MyService extends Service
{
    @override
    public IBinder onBind(Intent obj)
    {
      return null;
    }
    @override
    public void onCreate()
    {
      Super.onCreate();
      Toast.makeText(this, "Service is created",
                  Toast.LENGTH_LONG).show();
    }
    Public void onStart(Intent intent, int startId)
    {
      Super.onStart(intent, startId);
      Toast.makeText(this, "Service is started",
                  3000).show();
    }
    @override
    public void onDestroy()
    {
      super.onDestroy();
      Toast.makeText(this, "service is stopped",
                  5000).show();
    }
}
```

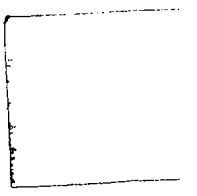
NOTE:

Register above activity and service class in the manifest.xml

Design an android application to change the background color of the layout till we click on Stop button.



Design an android application to change the size of the Imageview.



25/12/2012

Mail Synchronization : (Sending e-mail through Synchronized device).

Android Supports push mail technology, in which the mails will be pushed into the android device from the Synchronized mail account because of above technology we no need to check e-mail every time by providing username and password. Once mail account is synchronized with the current android device then automatically all the mails will be push into the current device / emulator.

Steps to Synchronize a mail account with an android device

1. Open Email application in Emulator/ real device (this appln is available in the Main menu)
2. provide Email-id and password → click [Next]
3. provide name (optional) → click [done]

Select option menu and select various options to perform some dynamic operation like Composing mail, mail settings etc.,

Design an android appln to send the e-mails from the synchronized mail account to other account

main.xml

```
<LinearLayout>
<Button android:layout-height = "w-c"
        a:l-w = "f-p"
        a:onClick = "onclick"
        a:text = "Send Email"
        a:id = "@+id/send"/>
</LLayout>
```

Activity class

```
public class SendingEMAILActivity extends Activity
 {
   @Override
     public void onCreate(Bundle b)
     {
      Super.onCreate(b);
      setContentView(R.layout.main);
      }
     public void onClick(view v)
      {
       String[] to = {"Android@satya.com"};
       String[] cc = {"Java@satya.com"};
      sendEmail (to,cc, "Hello", "welcome to the session");
      }
     private void sendemail( String[] to, String[] cc,
           String subject, String message)
      {
        Intent emailIntent = new Intent(Intent.ACTION_
                    SEND);
         emailIntent.setData(uri.parse("mailto:"));
         emailIntent.putExtra(Intent.EXTRA_CC, cc);
         emailIntent.putExtra(Intent.EXTRA_SUBJECT,
                                  Subject);
         emailIntent.putExtra(Intent.EXTRA_TEXT,
                                      message);
         emailIntent.setType("message/rfc822");
       startActivity(Intent.CreateChooser(emailIntent,
                                  "Email"));
      }
 }
```

27/12/2012

Data Storage:

meaningful data

→ Storing of data (or) information in the memory is known as Data storage, Memory is a special Location either in computer (or) mobile to store the data or information either tempararily or permanently

Temporary memory:

→ If any data is stored in a memory and whose scope is upto a specific session is known as temporary memory.

→ In general these memories can be achieved through datatypes of a programming language.

Permanent memory:

→ If any data is stored in a memory and whose scope is infinite (depends on user requirement) is known as permanent memory.

⇒ In android devices permanent memory can be achieved in two ways.

1) Internal Memory / phone Memory

2) External Memory / SDcard

→ We can store any type of data in the form of files either in Internal (or) External Memory.

→ Android was introduced Local database to store the data permanently and securely, this is part of Internal Memory. (SQLite DataBase)

↳ Light weight DB which will occupy 128KB of Internal Memory.

Working with Internal Memory:

→ In Android device by default Internal Memory will be represented with following location.

Myfiles / data

→ In Internal Memory mainly we can perform two types of operations.

1. Storing the data in the form of files
2. Retrieving data in the form of files.

→ To work with Internal Memory "Activity class" is provided by android API

Methods of Activity class:

1. getFilesDir():

gets the absolute path of the file system directly where your internal files are saved.

2. getDir():

which can be used to get the current working directory of internal storage.

3. DeleteFile():

which can be used to delete the existing file.

Storing files in the Internal Memory:

Any type of file (text or image or audio or video) can be stored in the Internal Memory with the following syntax.

Syntax:

FileOutputStream objref = openFileOutput (path of the file, mode of type);

→ here mode of type can be: MODE_PRIVATE → gives private access only for a specific application

MODE_PUBLIC → gives public accessability for all android applns.

28/12/2012

Example:

```
FileOutputStream fos = OpenFileOutput ( getFilesDir() + "/abc.
                                txt ", MODE_PRIVATE);
```

↓

generally we can write it as

Context.MODE_PRIVATE

↓

this

→ In the above example, the type of the file can be changes by changing the extension

Eg: a.txt → text file a.png → image file

a.mp3 → audio file etc.,

a.mp4 → video file

→ Except on text file, we can't perform any writing (or) reading operations on other files

29/12/2012

Writing and Reading Operations on Text Files:

a) Writing Data:

Data can be written into any file either by using OutputStream class or OutputStreamWriter class

NOTE:

OutputStream is a Super class for DataOutputStream, FileOutputStream, ByteArrayOutputStream, BufferedOutputStream, ObjectOutputStream classes

Syntax:

```
FileOutputStream fos = context.OpenFileOutput ( " path of
                        the file ", mode of type);
OutputStreamWriter OS = new OutputStreamWriter (
                                                fos);
OS.write(--);
```

b) Reading Data from file:

→ Data can be read from any file either by using InputStream class (or) InputStreamReader class.

Syntax:

```
FileInputStream fis = context.openFileInput("path of the file");
InputStreamReader is = new InputStreamReader(fis);
is.read();
```

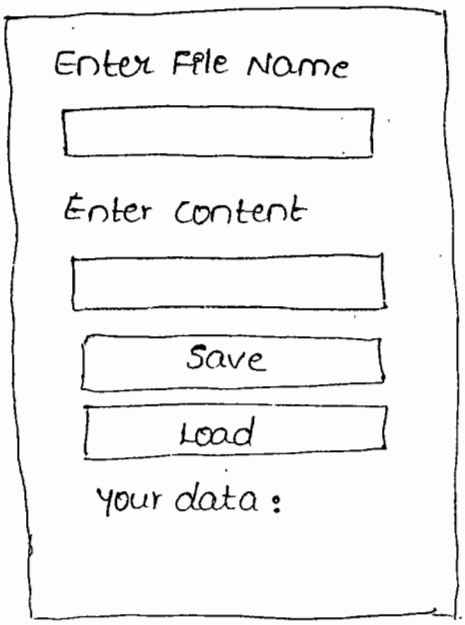
→ In the above syntax,

openFileInput() is a predefined method of Activity class used to open the file in reading mode by retrieving from the given location of Internal Memory.

NOTE:

It is not necessary to take permissions in manifest file while working with Internal Memory

Design an android application to work with Internal Memory.



main.xml

```
<LL  a:l-h="f-p"
     a:t-w="f-p"
     a:orientation="vertical">

<TextView  a:l-h="w-c"
           a:l-w="f-p"
           a:text="Enter File Name"/>

<EditText  a:l-h="w-c"
           a:l-w="f-p"
           a:id="@+id/filename"/>

<TextView  a:l-h="w-c"
           a:l-w="f-p"
           a:text="Enter content"/>

<EditText  a:l-h="w-c"
           a:l-w="f-p"
           a:text="En @+id/content"/>

<Button  a:l-w="wrap-content"
         a:l-h="w-c"
         a:text="Save"
         a:id="@+id/save
               load"/>

<Button  a:l-w="f-p"
         a:l-h="w-c"
         a:text="load"
         a:id="@+id/load"/>

<TV  a:l-h="w-c"
     a:l-w="f-p"
     a:text="Your Data:"
     a:id="@+id/data"/>

</LinearLayout>
```

Activity class

```
- -
- -   // import statements
- -
public class AndroidInternalStorageActivity extends Activity
                                    implements onClickListener
{
  EditText edFilename, edContent;
  Button btnSave, btnLoad;
  TextView tv;
   @Override
   public void onCreate(Bundle b)
   {
     super.onCreate(b);
     SetContentView(R.layout.main);
   edFilename = (EditText) findViewById(R.id.fileNa.
                                          me);
      edContent = (EditText) findViewById(R.id.content);
      btnSave = (Button) findViewById(R.id.save);
      btnLoad = (Button) findViewById(R.id.load);
       tv = (Button)
       tv = (TextView) findViewById(R.id.data);
  btnSave.setOnClickListener(this);
  btnLoad.setOnClickListener(this);
  }
  @Override
    public void onClick(View arg0)
     {
```

```
if(arg0.getId() == R.id.save)
  {
    try
     {
      WriteSettings(this, edcontent.getText().toString(),
            edFileName.getText().toString());
     }
     catch(Exception e)
      {
       System.out.println(e);
      }
    }
    else
    {
     try
       {
        String s= ReadSettings(this, edFileName.getText().
                                        toString());
          tv.setText(s);
       }
      catch(Exception e)
       {
        System.out.println(e);
       }
    }
public void WriteSettings(Context context, String data, String
                          file) throws IOException
    {
     FileOutputStream fos=null;
     OutputStreamWriter osw=null;
```

```
fos = context.openFileOutput(file, context.MODE_PRIVATE);
OSW = new OutputStreamWriter(fos);
OSW.write(data);
Toast t = Toast.makeText(this, "content saved successfully
        into Internal Memory", 1000);
    t.show();
    OSW.close();
    fos.close();
}
public String ReadSettings(Context context, String file) throws
                                                IOException
  {
    FileInputStream fis = null;
    InputStreamReader isr = null;
    String data = null;
    fis = context.openFileInput(file);
    isr = new InputStreamReader(fis);
    char[] inputBuffer = new char[fis.available()];
      isr.read(inputBuffer);
    data = new String(inputBuffer);
        isr.close();
        fis.close();
        return data;
    }
}
```

Working With External Memory: (SDcard)

→ External Memory is optional and expandable Memory of android device, which can be used to store more data in file format Compared to Internal Memory.

→ External Memory can be handled by using Environment class of android API

31/12/2012

→ The Default Memory Location for SD card or External Memory in real device is myfiles/android (or) myfiles/sdcard.

→ Similar to Internal Memory we can work with External Memory with the help of Environment class, It offers following methods to work on sdcard. Location and files within that location.

1. getExternalStorageState()

Which returns current External storage called state; which may returns a) MEDIA-MOUNTED → which represents media is presented with write (or) read access

b) MEDIA-MOUNTED-READ-ONLY → Which represents media is present with Read only access.

c) MEDIA-UNMOUNTED → which represents media is present but no accessability is given for reading (or) writing.

NOTE:

Media represents sdcard

2. getExternalStorageDirectory()

Which returns path of the external storage directory. that means path of the sdcard location.

3) getExternalStoragePublicDirectory(String type)

→ Which can be used to get the path of the public directories in that sdcard like Gallery, Ringtones, Download etc., folder.

→ DIRECTORY_PICTURES, DIRECTORY_MUSIC, DIRECTORY_RINGTONES etc are the default string parameters of above method.

Example:

```
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES);
```

↓

which will return path of the gallery in android device.

4. getRootDirectory():

→ Which returns root directory of android (myfiles/android)

5. isExternalStorageRemovable():

returns "true" if the sdcard is removable otherwise returns false.

Permissions:

→ following permissions should be taken in manifest.xml file while working with SD card.

```
<uses-permission
    android:name="android.permission.READ_EXTERNAL_STORAGE" /> // reading permission
<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE" /> // writing permission.
```

Design an android appln to work with External Memory (or) Sdcard.

main.xml

```
<LL a:orientation="vertical"
    a:l-h="f-p"
    a:l-w="f-p">
  <Textview a:l-h="w-c"
            a:l-w="f-p"
            a:text="@string/hello"
            a:textstyle="bold"
            a:textSize="bold"
            a:id="@+id/tv"/>
  <TV a:l-h="w-c"
      a:l-w="f-p"
      a:textStyle="bold"
      a:textSize="20dp"
      a:id="@+id/dirs"/>
</LLinearLayout>
```

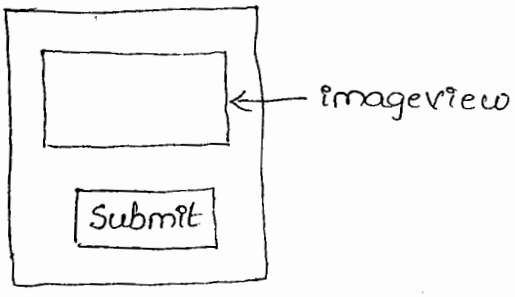
Activity class

```
P C WorkingWithSDcardActtivity
  extends Activity
  {
   Textview tv,dirs;
    @override
   P v onCreate(Bundle b)
    {
     Super.onCreate(b);
     SetContentView(R.layout.main);
   tv=(Textview)findviewById(
                    R.id.tv);
   dirs=(Textview)findviewById(
                    R.id.dirs);
    boolean media_read=false;
    boolean media_write=false;
 if(Environment.MEDIA_MOUNTED.
    equals(Environment.getExtern
     alstorage state()))
   {
   tv.setText("SD card exists");
   media_read=true;
   media_write=true;
   File f=new file(Environment.
   getExternalstorage Directory().
    getAbsolute Path());
   String arr[]=f.list();
   for(int i=0; i<arr.length;i++)
   {
   File ff=new File(f.getAbsolute.
    path()+"/"+arr[i]);
     if(ff.isfile())
    {
     dirs.append(arr[i]+"=>
         is a file\n");
    }
    else {
    dirs.append(arr[i]+"=> is
            directory\n");
      }
    }
  }
```

```
if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(
    Environment.getExternalStorageState()))
    {
      media_read = true;
      media_write = false;
    }
    tv.append("Media Readonly Status :" + media_read +
              "media write only status : " +
                                media_write);
    }
}
```

11/1/2013

logic to store image in the external Memory by retrieving from imageview.




```
public void Saveimgbtn()
  {
   Imageview iv = (Imageview) findViewById(R.id.
                                    img);
     try
     {
       BitmapDrawable bd = (BitmapDrawable) iv.
                          getDrawable();
```

```
Bitmap bi=bd.getBitmap();
String path = Env.getExternalStorageDirectory().toString();
new file(path + "folder/img/").mkDirs();
File f = new File(path + "/folder/img" + iv.getImage());
FileOutputStream out = new FileOutputStream(f);
bi.compress(Bitmap.compressformat.JPEG, 90, out);
out.close();
Mediastore.Images.Media.insertImage(getContentResolv
er(), f.getAbsolutePath(), f.getName(), "any UD desc");
Toast.makeText(getApplicationContext, "file is saved
                    in"+f,1000).show();
}
  catch(Exception e)
   {
    e.printStackTrace();
   }
}
```

Q. Logic to store the image in the Internal Memory

Same above logic should be written except following statement

```
String path= context.getFilesDir().toString();
```

instead of

```
String path= Environment.getExternalStorageDirectory().
                    toString();
```

2/4/2013

Storing Audio/video files in the External Memory:

Syntax:

a) File f = new File(Environment.getExternalStorageDirectory(),
"foldername/subfoldername/---");
f.mkDirs();
FileOutputStream fos = new FileOutputStream(Environment.
getExternalStorageDirectory() + "/foldername/
subfoldername/filename);

filename:
R.raw.song.mp3 (audio)
R.raw.song.mp4 (video)

→ The above syntax can be used to load either audio (or) video files which are already stored in "raw" folder in the External Memory.

b) File f = new File(Environment.getExternalStorageDirectory(),
"foldername/subfolder.name/---");
f.mkDirs();
File of = File.createTempFile("filename", "ext", f);
eg: abc .mp3
MediaRecorder recorder = new MediaRecorder();
——— //set audio source (MIC)
——— // set file format
——— //set encoder type
} → code for voice recording
recorder.setOutputFile(of.getAbsolutePath());

→ This syntax can be used to store the recorded voice in External Memory.

Storing Audio/Video files in the Internal Memory:

a) `File f = new File(context.getFilesDir() + "/foldername/subfoldername/--");`

`f.mkDirs();`

`FileOutputStream fos = context.openFileOutput(context.getFilesDir() + "/foldername/subfoldername/filename");`

audio → R.raw.song.mp3
video → R.raw.video.mp4

→ The above syntax can be used to store audio (or) video files which are already available in the "raw" folder into Internal Memory.

b) `File f = new File(context.getFilesDir() + "/foldername/subfoldername/--");`

`f.mkDirs();`

`File of = File.createTempFile("song", ".mp3", f);`
↳ Creation of empty audio file (song = file name, .mp3 = extension of audio)

```
MediaRecorder recorder = new MediaRecorder();
------ // set audio source (MIC)
------ // set file format
------ // set encoder type
```
} voice recording code

`recorder.setOutputFile(of.getAbsolutePath());`

→ The above logic can be used to store recorded voice in an empty file that is going to be stored in Internal Memory.

3/1/2013 SQLite Database:

→ SQLite is a light weight. local Database introduced along with Android OS, this is the first Database introduced for Mobile devices

→ SQLite Database can be used to store the Local data permanently and securely in the form of tables.

→ In every android device SQLite Database is available in the following Location of Internal Memory.

`Myfiles/data/data`

→ following Method of Environment class can be used to get the above working Directory path. of SQLite Database.

```
Environment.getDataDirectory();
```

→ SQLite Supports following Datatypes.

1. TEXT → chars or strings
2. INTEGER → integer constants (+/-)
3. REAL → Real constants (+/-)

→ SQLite Database Supports almost all SQL commands to perform various operations Like

* Creation of Table
* Inserting record
* Delete record
* Select record ... etc.,

1. Create table:

Which can be used to create a table in the SQLite Database.

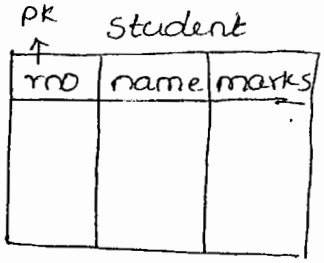
Syntax:

Create table <tablename> (col1 datatype constraint, col2 datatype constraint, - - -);

Example:

not null & unique

Create table Student (rno INTEGER primary key, name TEXT not null, marks Real);



2. Insert record in the table:

→ This command can be used to insert a record in a specific table.

Syntax 1:

insert into <tablename> values (val1, val2, val3, --); → used to insert data in every column

Example: insert into student values (100, 'abc', 73.87);

Syntax 2:

insert into <tablename> (col1, col2, --) values (val1, val2, - -) ;

↳ used to insert data in particular column.

Example: insert into student (rno, name) values (200, 'xyz');

3. Update record:

→ Which can be used to update data of table.

Syntax 1:

update <tablename> set col1 = value, col2 = value, - - -;

↳ all the records will be updated.

Example

update student set marks = 81.3

Syntax 2:

update <tablename> set col1 = value, -- where condition;

↳ used to update a specific record

Example

update student set marks = 80 where rno = 200;

4. select record:

→ Which can be used to retrieve records from the table.

Syntax 1

select * from <tablename>; → used to retrieve all the records with all column data

Example

select * from student;

Syntax 2

select * from <tablename> where condition; → used to retrieve a specific record with all col data

Example: select * from student where rno = 100;

Syntax 3

select col1, col2, -- from <tablename>; → used to retrieve all the records with specific col data

Example: select rno, name from student;

Syntax 4:

select col1, col2, -- from <tablename> where condition;

↳ used to retrieve specific record(s) with specific col(s) data.

Example: select marks from student where rno = 100;

5. delete record:

→ delete command can be used to delete the record from the table

Syntax 1

delete from <tablename>; → all the records of the table will be deleted

Example: delete table student;

Syntax 2

delete table <table name> where condition; → a specific record of the table will be deleted
from

Example: delete table student where rno=100;

NOTE:

In Android Device the above operations can be performed in SQLite DB through android applications with the help of SQLiteOpenHelper and SQLiteDatabase classes of Android API but it is not possible directly.

4/1/2013 **Working with SQLiteDatabase Through Android application:**

→ android API provides following two predefined classes of android.database.sqlite package to work with SQLiteDatabase

1. SQLiteOpenHelper
2. SQLiteDatabase

1. SQLiteOpenHelper:

Which can be used to create a new schema or Database and predefined tables in SQLite Database

Rules to create user defined SQLOp SQLiteOpenHelper class:

1. Create a userdefined class and extends SQLiteOpenHelper class
2. create a userdefined parameterised constructor and call super class constructor within that.
3. this can be used to create new schema in the SQLiteDatabase.
4. Override onCreate() of SQLiteOpenHelper class in the UserDefined class

This method can be used to create the tables in the selected schema of SQLiteDatabase.

4. Override onUpGrade() method, which can contain the logic either to upgrade database version (or) to upgrade tables of SQLite Database.

Syntax:

```
public class <CNY> extends SQLiteOpenHelper
{
   public <CNY>(Context context)
   {
     super(context,"dbname", cursorfactory, version);
   }
   public void onCreate(SQLiteDatabase db)
   {
     db.execSQL("create table cmd");
   }
   P v onUpGrade(SQLiteDatabase db, int oldversion,
                 int newversion)
   {
     // drop older table if existed
      db.execSQL("drop table if exists tablename");
     // create table again
   }
}
```

In the above syntax,

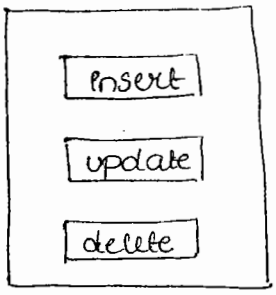
→ Constructor and onCreate() will be executed one after another when an object is created for the given class

→ While calling Super class constructor we can pass any userdefined name as a dbname and cursorfactory value can be null and version also can be any value existing version number

→ onUpGrade() will be executed whenever execution of onCreate() is failed, in which we can write the logic to drop the table if that is existed and also to create new table

⇒ SQLiteDatabase is a predefined class used to execute the SQLcommands through SQLiteOpenHelper class in the above Syntax.

Design an android application to work with SQLite Database (perform insertion, updation and deletion operations)



Activity class

```
public class Act1 extends Activity
{
    MySQLite ms;
  @override
   P v onCreate(Bundle b)
   {
      super.onCreate(b);
       setContentView(R.layout.main);
      ms = new MySQLite(this);
   }
  P v clickInsert(view v)
    {
     ms.insertMethod(); Toast.makeText(this," record
     }                    inserted Successfully ", 5000).show();
  P v clickUpdate(view v)
    {
     ms.updateMethod(); Toast.makeText(this, " record
     }                    updated successfully", 5000).
                          show();
  P v clickDelete(view v)
    {
     ms.deleteMethod(); Toast.makeText(this, " record
   } }                    updeo deleted Successfully ", 3000).
                          show();
```

SQLiteOpenHelper class

```
public class MySQLite extends SQLiteOpenHelper
{
  SQLiteDatabase sd;
  public MySQLite(Context c)
  {
  super(c, "mydb", null, 1); S.o.p("Database created
  }                                                Successfully");
  @override
  P v onCreate(SQLiteDatabase db)
  {
  db.execSQL("create table student(rno integer
     primary key, name text not null, marks real)");
  }                    S.o.p("table created successfully");
@override
  P v onUpdate(SQLiteDatabase db, int oldv, int newv)
  {
  db.execSQL("drop table if exists student");
  this.onCreate(db); S.o.p("table was upgraded");
  }
  P v insertMethod()
  {
 * (Sd = new SQLiteDatabase();)* Sd = this.getWritableDatabase();
   Sd.execSQL("insert into student values(1,'abc',
  }          S.o.p("insert success");      83.7)");
  P v updateMethod()
  {
 * (Sd = new SQLiteDatabase();)* Sd = this.getWritableDatabase();
   Sd.execSQL("update student set marks=70.3
                    where rno=1");
  }          S.o.p("update success");
             // return 1; (if return type is int)
```

```
p v deleteMethod()
{
*(sd = new SQLiteDatabase();)x    sd = this.getWritableDatabase();
                    from
 sd.execSQL("delete table student where rno=14");
 }          s.o.p("delete Success");
            // return 1; (if return-type is int)
}
```

5/1/2013

SQLite Database class

→ It is a predefined class in android.database.sqlite package used to perform various operations on the records of Database table like insert, update, insert, select, etc.,

→ following predefined methods are supported to perform above Database operations.

1. insert():

→ Which can be used to insert a record in the given table

Syntax:

```
                                        null
                                         ↑
long insert(string table, string nullColumnHack, ContentValues objref)
                   ↓                                      ↓
               table name                 inserted values in the
                                          form of content object
```

→ nullColumnHack will explicitly inserts null values in the empty columns.

2. update():

→ Which can be used to update a given record

Syntax:

table name updated values where condition

int update (String table, ContentValues objref, String where clause, String[] whereArgs)

arguments of where condition

→ where clause and arguments of where clause can become null if there is no specific where condition is used.

3. delete():

Which can be used to delete one or more records from the given table

Syntax:

table name where condition

int delete (String table, String whereClause, String[] whereArgs)

arguments for where condition

→ for both delete() and update() return type is int. that means which returns '0' if no rows are affected otherwise returns no. of rows affected

→ for insert() return type is long which returns '-1' if insertion is failed otherwise Rowid of latest inserted row

4. query():

records

Which can be used to select one or more commands from the database table.

Syntax:

tablename list of retrieved columns where clause

cursor query (String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy)

↑ where clause arguments

→ groupBy, having, orderBy can be null values. So that default properties will be applied.

5. execSQL(String sql)

Which can be used to execute any sql command except select command and other commands which are going to Return some data.

NOTE:

query() always returns Cursor class object.

→ Whenever an application is executed (with SQLiteOpenHelper class) then a seperate space will be created with the application name in the SQLite Database as shown below

myfiles/data/data/appname/databases/dbfile

6/1/2013

→ SQLiteOpenHelper class contains following methods to work with sqlite Database

1. onCreate()

called when the database is created for first time, in which we must write the logic to create the tables.

Syntax:

abstract void onCreate(SQLiteDatabase db)

2. onUpgrade()

→ This will be called when the database needs to be upgraded.

→ The implementation should use this method to drop table, add table or do anything else it needs to upgrade to the new schema version.

Syntax:

abstract void onUpgrade(SQLiteDatabase db, int oldversion,
int newversion)

3. onDowngrade():

called when the database needs to be downgraded, this method is similar to onUpgrade() but is called whenever current version is newer than requested one, it is not mandatory to override in the userdefined SQLiteOpenHelper class.

4. getWritableDatabase():

create and/or open database that will be used for reading and writing. (which will set writing or reading mode for the db)

insert, update, delete → write mode select → read mode

5. getReadableDatabase():

which can be used to set read mode for the database.

6. getDatabaseName():

returns the name of SQLiteDatabase being opened as given to the constructor.

7. close():

which can be used to close the Database connection (which was already opened)

NOTE: The dbfile is available to view through DDMS after execution of Database related application in the following location → exists in the form of package

data/data/appname/databases/dbfile

NOTE:

No permissions are required in manifest file while working with SQLite Database

Design an android application to perform insertion, deletion and selection operations on the SQLite Database.

main.xml

```
<LinearLayout a:l-h="f-p"
              a:l-w="f-p"
              a:orientation="vertical">
  <ListView a:l-w="f-p"
            a:l-h="f-p"
            a:id="@+id/contentlist"/>
</LinearLayout>
```

row.xml

```
<Textview a:padding="10dep"
          a:l-h="f-p"
          a:l-w="f-p"
          a:id="@+id/text"/>
```

Activity class

```
public class SQLiteListViewCursorAdapterActivity extends
                                        Activity
{
private SQLiteAdapter mySQLiteAdapter;
    @Override
  P v onCreate(Bundle b)
    {
      Super.onCreate(b);
```

```
setContentView(R.layout.main);
ListView listContent = (ListView) findViewById(R.id.
                                    contentlist);
mySQLiteAdapter.deleteAll();
mySQLiteAdapter.insert("A for Android");
mySQLiteAdapter.insert("B for Boy");
        :
        :
mySQLiteAdapter.insert("J for Java");
        :
mySQLiteAdapter.insert("X for X'mas");
mySQLiteAdapter.insert("Y for yellow");
mySQLiteAdapter.insert("Z for Zoo");
mySQLiteAdapter.close();
// open the same SQLite Database and read all it's content
mySQLiteAdapter = new SQLiteAdapter(this);
Cursor cursor = mySQLiteAdapter.queueAll();
startManagingCursor(cursor);
String[] from = new String[]{SQLiteAdapter.KEY_
                                    CONTENT};
int[] to = new int[]{R.id.text};
SimpleCursorAdapter cursorAdapter = new SimpleCursor
        Adapter(this, R.layout.row, cursor, from, to);
listContent.setAdapter(cursorAdapter);
mySQLiteAdapter.close();
    }
}
```

SQLiteOpenHelper class:

```
public class SQLiteAdapter extends SQLiteOpenHelper
{
  P S F String MYDATABASE-NAME = "MY-DATABASE";
  P S F String MYDATABASE-TABLE= "MY-TABLE";
  P S F int MYDATABASE-VERSION = 1;
  P S F String KEY-ID= "_id";
  P S F String KEY-CONTENT= "content";
// Create table MY-DATABASE( ID integer primary key, content
                                text not null);
  private static final String SCRIPT-CREATE-DATABASE =
   " create table " + MYDATABASE-TABLE + "( " + KEY-ID +
     " integer primary key autoincrement, " + KEY-CONTENT +
       " text not null);";
  private SQLiteDatabase sqLiteDatabase;
    public SQLiteAdapter( Context ctx)
  {
    super( ctx, MYDATABASE-NAME, null, MYDATABASE-VERS-
                                        ION);
  }
    public SQLiteAdapter( Context context, String name,
          CursorFactory factory, int version)
    {
     super( context, name, factory, version);
    }
    @override
     P V onCreate( SQLiteDatabase db)
      {
```

```
db.execSQL( SCRIPT-CREATE - DATABASE);
}
@override
p v onUpgrade( SQLiteDatabase db, Int oldVersion, Int
                                    newVersion)
 {

 }
p v close()
 {
  super.close();
 }
 p long insert(String content)
 {
SqliteDatabase = this.getWritableDatabase();
 ContentValues contentValues = new ContentValues();
  contentValues.put( KEY-CONTENT, content);
  return sqliteDatabase.insert(MYDATABASE, null,
                    contentValues);
  }
  public int deleteAll()
  {
   SqLiteDatabase = this.getWritableDatabase();
   return SqLiteDatabase.delete(MYDATABASE-TABLE,
                                    null,null);
   }
   public Cursor queueAll()
   {
  SqLiteDatabase = super.getReadableDatabase();
   String[] columns = new String[] {KEY-ID, KEY-
                                    CONTENT};
  Cursor cursor = sqliteDatabase.query( MYDATABASE-
      TABLE, columns, null, null, null, null, null);
          return cursor;
} }
```

Q. What is the difference b/n query() and rawQuery()?

Ans: → query() can be used to retrieve / select records from the table by passing different parameters like table name, list of column etc.,

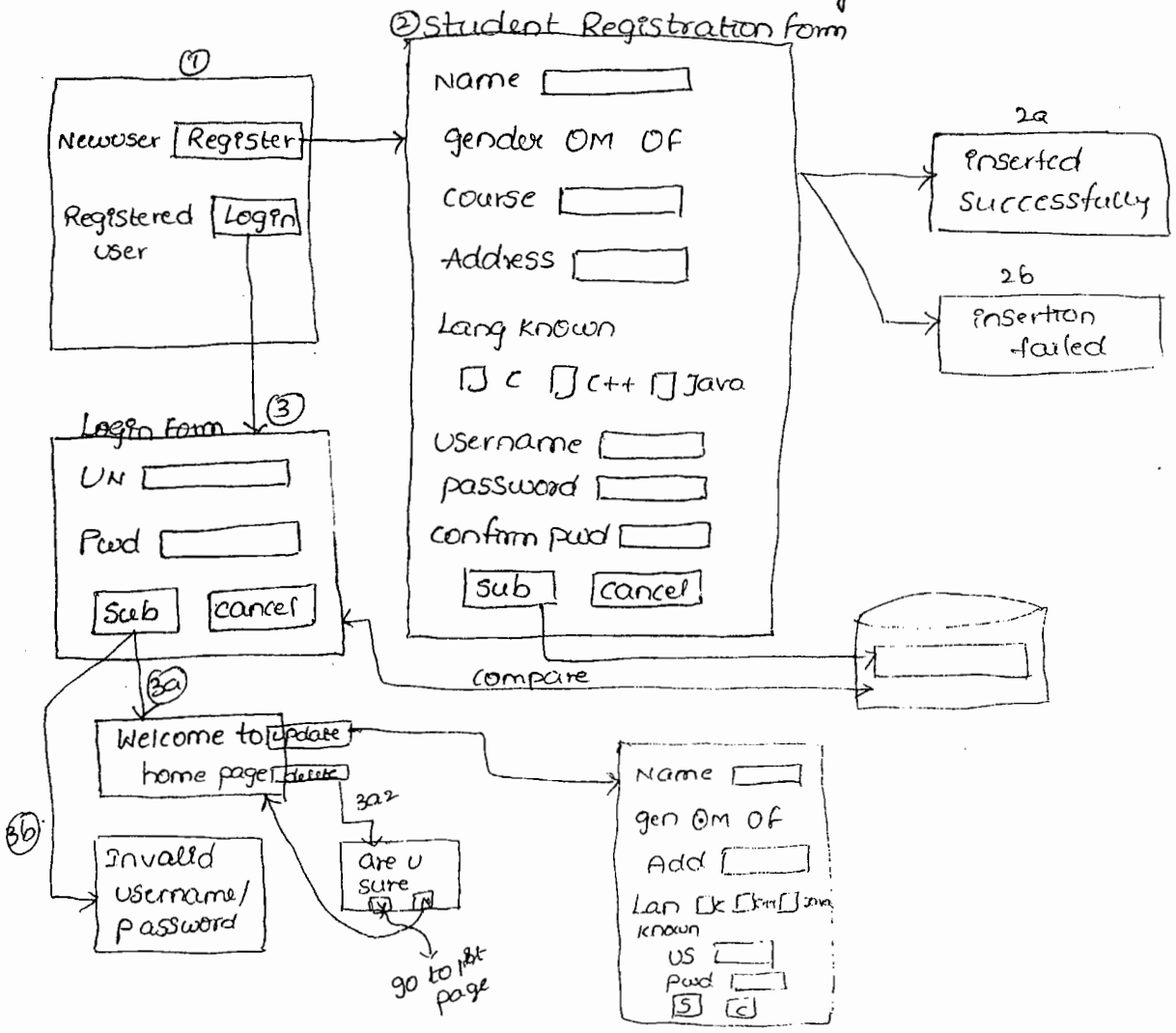
parameters can be passed within query() as shown below

```
Cursor c = db.query(viewEmps, columns, colDeptName +
        "=?", new String[]{value1, ---}, null,
        null, null);
```

→ Where as rawQuery() is also used to retrieve/select the records from the table based on select command and its related arguments as shown below

```
Cursor cur = db.rawQuery("SELECT colname1, colname2, --
    from Tablename where col=?", new String[]{val1, --});
```

Design an application for the following



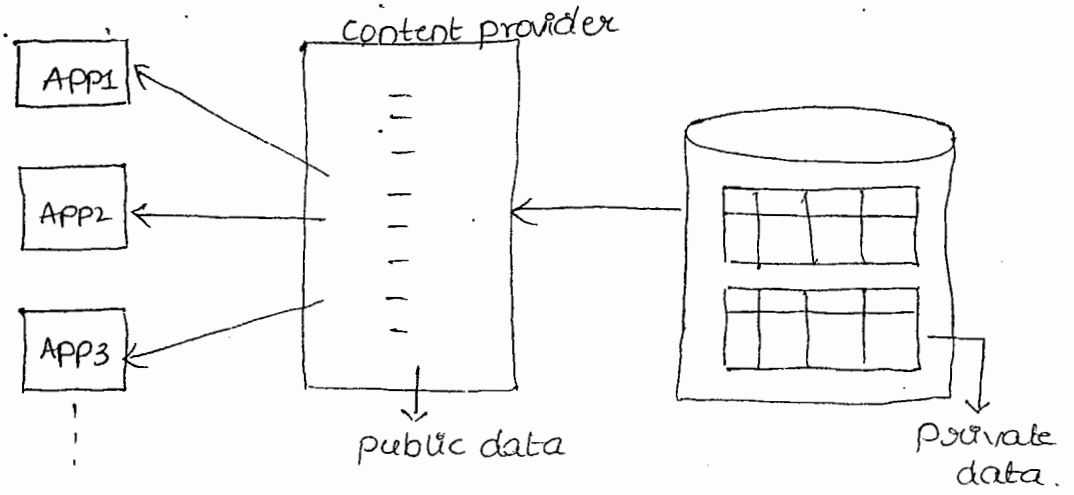
7/1/2013

Content Providers:

→ Content Provider is a predefined android element used to make private data of SQLite Database as public data. so that Multiple applications can able to access that data.

→ In general the data which is available in SQLite Database is private Data. that means only a specified application can access that data but no other applications.

→ Whenever other applications of same device wants to access the same data of SQLite Database then that should be made as public data using content provider.



→ In android every content provider is existing as a class

Classifications of Content Providers:

In android, ContentProvider mainly classified into two types

1) predefined contentProvider
2) User defined / custom ContentProvider

1. predefined contentProvider:

These are the predefined in the android API in the form of classes. used to make private data of pre defined SQLite Database tables as public data so that multiple applns of same

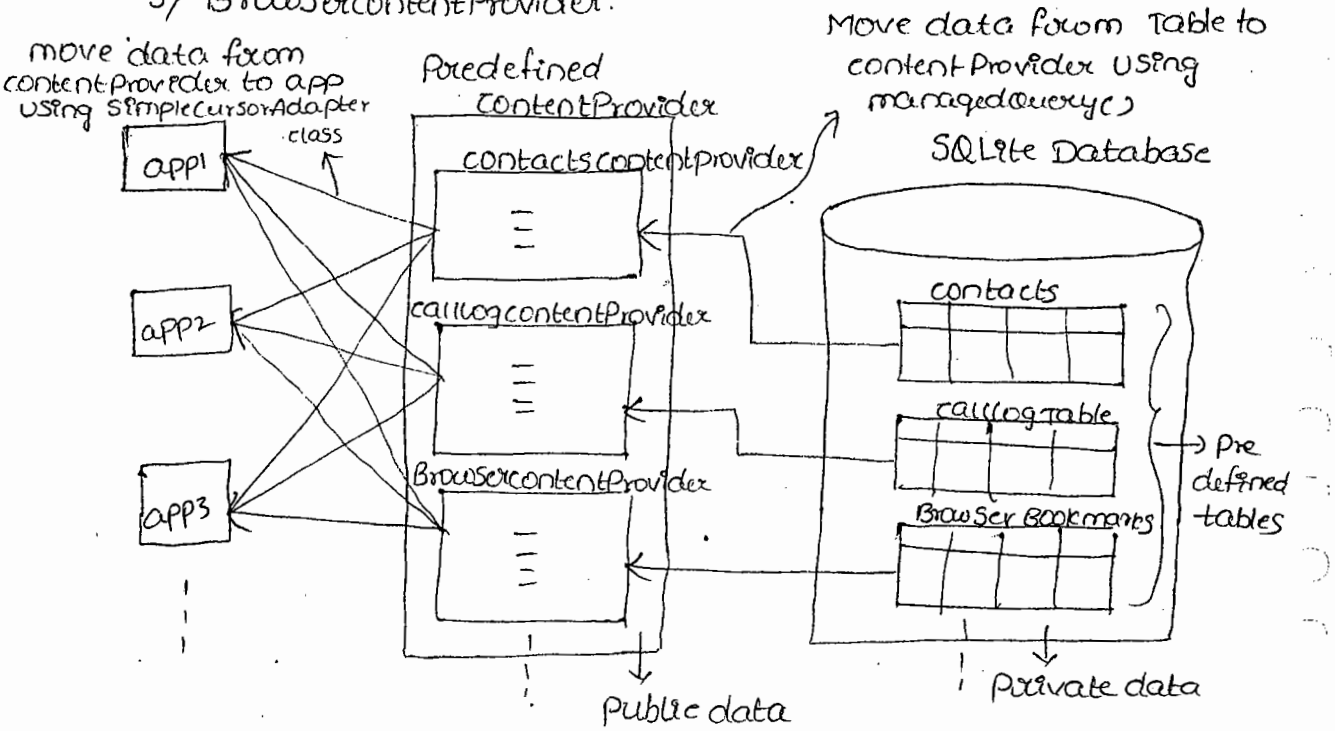
android device can access.

→ These are classified into following types.

1) ContactContentProvider

2) CallLogContentProvider

3) BrowserContentProvider.



→ Without SQLite Database there is no ContentProvider is existing

NOTE:

Multiple applications of same android device can able to use the private data of SQLite Database when it is available in ContentProvider but there is no provision for other applications to modify the data or insert the data or delete the data in the database

1. ContactContentProvider:

→ This is used to share the phone contacts information of current real device to every android application.

→ This contentProvider can be represented by the following predefined class in android API

"android.provider.contactscontract.contacts"
package outerclass innerclass

→ following predefined uri can be used to access phone contacts information. (to activate Contact contentProvider)

"content://contacts/people"

NOTE: above predefined class consists phone contact details as static fields.

→ managedQuery()

It is a predefined method of activity class can be used to move the data from Database to contentProvider. [every content Provider by default acting as Resultset or CursorAdapterView, it is a temporary memory Location in the primary Memory].

Syntax:

managedQuery(contenturi, String[] projection, String (where condition) selection, String[] selectionargs, OrderBy)

projection → required Fields from the table

Design an android application to work with contactContentProvider

main.xml

```
<LinearLayout ....
              ....>
  <ListView a:id="@+id/android:List"
            ...
            .../>
  <TextView a:id="@+id/contactName"
            ...
            .../>
  <TextView a:id="@+id/contactID
            ...
            .../>
</LinearLayout>
```

Activity class:

```
P C ContentProviderReadcontacts-Activity extends
                                    ListActivity
  {
     @override
     P v onCreate (Bundle b)
     {
       Super.onCreate(b);
       setContentView(R.layout.main);
     Uri cts = uri.parse("content://contacts/people");
```

used to covert string format to uri format

```
     Cursor c = managedQuery(cts, null, null, null, null);
     String columns[] = new String[]{
ContactsContract.Contacts.DISPLAY_NAME, ContactsContract.
     Contacts._ID};
Int [] views = new int[] { R.id.contactName, R.id.conta-
                                              ctID};
 SimpleCursorAdapter adapter = new SimpleCursorAdapter(
       this, R.layout.main, c, columns, views);
current activity                    from    to
       this.setListAdapter(adapter);
  }
}
```

NOTE:

We must take following permission in manifest file while reading contacts through any third party application.

```
<uses-permission
   android:name="android.permission.READ_CONTACTS"/>
```

8/1/2013

2. CallLog ContentProvider:

→ This is used to share the call log information (missed calls, dialed numbers, received calls etc.,) of current real device to every android application.

→ This contentProvider can be represented by the following pre-defined class in android API

`android.provider.CallLog.Calls`

→ The uri to access callLog contentProvider is

`CallLog.Calls.CONTENT_URI`

Design an android application to work with CallLog ContentProvider

```
<LL  a:l-w = "f-p"
     a:l-h = "f-p"
     a:orientation = "vertical" >

  <TextView a:l-w = "f-p"
            a:l-h = "w-c"
            a:text = "@string/hello" />
</LinearLayout>
```

```
Public class CallLogContentProviderActivity extends Activity
{
   P v onCreate( Bundle b)
     {
       Super.onCreate(b);
       SetContentView(R.layout.main);
      String[] projection = new String[] { CallLog.Calls.
       NUMBER, CallLog.Calls.DATE, CallLog.Calls.CACHED_NA
         ME);
```

```
uri contacts = CallLog.Calls.CONTENT_URI;

Cursor managedCursor = managedQuery(contacts, Projection,
    null, null, CallLog.Calls.DATE + " ASC");
  getColumnData(managedCursor);
// code for other functionality continues..
}
  private void getColumnDate(Cursor cur)
    {
      try{
          if(cur.moveToFirst()) {
            String name;
            String number;
             long date;
            int nameColumn = cur.getColumnIndex(CallLog.
                                  Calls.CACHED_NAME);
int numberColumn = cur.getColumnIndex(CallLog.Calls.NUMBER);
int dateColumn = cur.getColumnIndex(CallLog.Calls.DATE);

S.O.pln("Reading call Details:");
  do{
    name = cur.getString(nameColumn);
    number = cur.getString(numberColumn);
     date = cur.getLong(dateColumn);
    S.O.pln(number + ":" + new Date(date) + ":" +
                                    name);
     //code for processing logic goes here
  } while(cur.moveToNext());
 }
}
    finally{
            cur.close();
          }
        }
      }
```

3. BrowserContentProvider:

→ It is a predefined contentProvider of Android Framework allows to share Browser History by Multiple android applications.

→ This contentProvider can be represented by the following predefined class in android API

`android.provider.Browser.BookMarkColumns`

→ The uri to access Browser content provider is

`Browser.BOOKMARKS_URI`

NOTE:

following permission should be taken while reading Browser History from the contentProvider in manifest.xml file.

```
<uses-permission
    android:name="com.android.browser.permission.
        READ_HISTORY_BOOKMARKS"/>
```

Design an android application to work with Browser content provider

main.xml

```
<LinearLayout a:Lw="f_p"
              a:Lh="f_p"
              a:orientation="vertical">
    <TextView a:Layout_width="f_p"
              a:Lh="w_c"
              a:text="@string/hello"/>
</LinearLayout>
```

Activity class

```
Public class BrowserContentProviderActivity extends Activity
{
    private static final String DEBUG_TAG = "VIEW 2";
    @Override
    P v onCreate(Bundle b)
    {
        super.onCreate(Bundle b);
        SetContentView(R.layout.main);
        String[] requestedColumns = { Browser.BookmarkCo-
            lumns.TITLE, Browser.BookmarkColumns.VISITS,
            Browser.BookmarkColumns.BOOKMARK };
        Cursor faves = managedQuery(Browser.BOOKMARK
            S_URI, requestedColumns, null, null, null);
        Log.d(DEBUG_TAG, "Bookmarks count: " +
                faves.getCount());
        int titleIdx = faves.getColumnIndex(Browser.
                BookmarkColumns.TITLE);
        int visitsIdx = faves.getColumnIndex(Browser.
                BookmarkColumns.VISITS);
        int bmIdx = faves.getColumnIndex(Browser.
                BookmarkColumns.BOOKMARK);
        faves.moveToFirst();
        int count = 0;
        while (!faves.isAfterLast()) {
// Log.d(DEBUG_TAG, "Inside while ---: " + count);
    Log.d("SimpleBookmarks", faves.getString(titleIdx) +
            "visited " + faves.getInt(visitsIdx) + " times: "
        + faves.getInt(bmIdx) != 0 ? "true" : "false"));
            faves.moveToNext(); count++;
    }
```

09/01/2013

Userdefined ContentProvider

→ if any ContentProvider is defined by the user is known as userdefined ContentProvider (or) Custom ContentProvider

→ Whenever We want to access private data of user defined database by the Multiple Applications of same device then that data should be made as public data, to achieve this userdefined ContentProvider can be used.

Rules to create userdefined ContentProvider:

1. Create a userdefined class and extends ContentProvider class
2. Override all the abstract methods of ContentProvider class

Syntax:

```
class <CN> extends ContentProvider
 {
   delete()
   {
   =
   }
   update()
   {
    =
   }
   onCreate()
   {
   }
   insert()
   {
   =
   }
   getType()
   {
   =
   }
   query()
   {
   =
   }
 }
```

3. Register contentProvider class in manifest file as shown below.

```
<manifest>
    <application>
        :
        :
        <provider android:exported = "true"
                  android:name = ".MyProvider"
                  android:authorities = "com.bened" >
        </provider>
    </application>
</manifest>
```

NOTE:

every userdefined contentProvider class should have userdefined SQLiteOpenHelper class to manage userdefined SQLiteDatabase.

4. ContentResolver class:

→ Which can be used to import userdefined contentProviders

Syntax:

```
ContentResolver c = super.getContentResolver();
```

→ every userdefined contentProvider can invoke with following method of contentResolver class

```
Cursor cs = c.query(url, projection, selection, selection
                    arguments, orderby);
```

NOTE:

To invoke with predefined contentProvider we need Activity.managedQuery() method similarly to invoke with userdefined contentProvider we need contentResolver.query() method, these two methods returns cursor object to navigate values.

Following uri should be taken or used to invoke with userdefined contentProvider

"content://com.bened/contenttable"

NOTE:

→ The above uri and contentResolver class should be used to work with userdefined contentProvider

Design an android application to work with user defined content provider

main.xml

```
<RelativeLayout a:l-w="fill_parent"
                a:l-h="f_p"
                a:orientation="vertical">
  <RL a:l-w="f_p"
      a:l-h="w-c">
    <TableLayout a:id="@+id/xTb"
                 a:l-h="w-c"
                 a:l-w="f_p"
                 a:stretchColumns="*">
      <TableRow a:l-w="f_p"
                a:l-h="f_p">
        <Button a:id="@+id/xBtnsubmit"
                a:layout_span="4"
                a:text="Insert"
                a:onClick="insert"/>
        <EditText a:id="@+id/xEtNum"
                  a:layout_span="3"
                  a:hint="no"
                  a:singleLine="true"/>
        <EditText a:id="@+id/xEtName"
                  a:layout_span="4"
```

```
        a:hint="name"
        a:singleLine="true"/>
    <EditText  a:id="@+id/xEtMail"
               a:layout-span="4"
               a:hint="mail"
               a:singleLine="true"/>
    </TableRow> </TableLayout>
    <Listview  a:id="@+id/xListview"
               a:l_w="f_p"
               a:l_h="w_c"
               a:layout_below="@id/xTb"/>
    </RelativeLayout>
</RelativeLayout>
```

item.xml

```
<RL  a:l_w="f_p"
     a:l_h="f_p".
     a:orientation="vertical">
    <Textview  a:id="@+id/xTv"
               a:l_w="f_p"
               a:l_h="w_c"
               a:textStyle="bold"/>
</RelativeLayout>
```

Activity classes

- ContentProviderOneDemoActivity.java

```
public class ContentProviderOneDemoActivity extends Acti-
                                                    vity
{
    private ArrayList<String> al=new ArrayList<String>();
    private ContentResolver cr;
    private Cursor c;
    private Listview mLv;
    private EditText mETNum, mETName, mETMail;
```

```
// demo columns
    String cols[] = new String[] ("no");
     int ids[] = new int[] { R.id.xTv};
      @override
  Public void onCreate( Bundle savedInstanceState)
     {
        super.onCreate (savedInstanceState);
        SetContentView ( R.layout.main);
         mETNum = (EditText) findViewById ( R.id.xETNum);
         mETName = (EditText) findViewById (R.id.xETName);
         mETMail = (EditText) findViewById (R.id.xETMail);
         mLV = (ListView) findViewById (R.id.xListView);
       // getting content resolver object
          cr = getContentResolver();
       // requesting user defined content Provider
          c = cr.query ( MyProvider.CONTENT_URI, null, null, null,
                                                        null);
     }
// inserting values which is entered in edit text boxes
      Public void insert (View v) {
        String no = mETNum.getText().toString();
        String name = mETName.getText().toString();
        String mail = mETMail.getText().toString();
        ContentValues cv = new ContentValues();
           cv.put ("no", no);
           cv.put("name", name);
           cv.put("email", mail);
           cr.insert( MyProvider.CONTENT_URI, cv);
           mETNum.setText(" ");
           mETName.setText(" ");
           mETMail.setText (" ");
c = cr.query( MyProvider.CONTENT_URI, null, null, null, null);
```

```
ListAdapter mAdapter = new ListAdapter( this, R.layout.main ,c,
                        cols, ids, getLayoutInflater());
        mLv.SetAdapter(mAdapter);
    }
}
```

MyProvider.java

```
P C MyProvider extends
    ContentProvider
{
// CONTENT_URI : BASE URL FOR
    CONTENT PROVIDER
P S final Uri CONTENT_URI =
 Uri.parse("content://com.
   bened/contenttable");
 private MyDataBase mdb;
 private SQLiteDatabase db;
 @Override
 P int delete(Uri uri,
String selection, String[]
  selectionArgs) {
    return 0;
 }
 /*
    * get URL type
 */
 @Override
P String getType(Uri uri){
    return null;
 }
 /*
  * inserting values to
                Database
 */
 @Override
 P Uri insert(Uri uri,
 ContentValues values){
db.insert("contenttable", null, values);
 return null;
}
// allocating memory to db and giving
 permission to database
  @Override
P boolean onCreate() {
mdb = new MyDataBase(getContext(),
 "contentData", null, 2);
 db = mdb.getWritableDatabase();
  return true;
}
 // Fetching values from db
  @Override
P Cursor query(Uri uri, String[] proje.
ction, String selection, String[] selectionA
rgs, String sortOrder) {
 String col[] = {"_id", "no", "name",
                    "email"};
Cursor c = db.query("contenttable",
  col, null, null, null, null, null);
  return c;
}
// for updating values to database
 @Override
 P int update(Uri uri, ContentValues
  values, String selection, String[]
   selectionArgs){
     return 0;
    }
}
```

MyDataBase.java

```
// global database for all
   classes. It's used to create
   custom content providers
P c MyDataBase extends
   SQLiteOpenHelper{
  Public MyDataBase( Context
  context, String name,
  CursorFactory factory, int
        version) {
  super(context, name, factory,
        version);
  }
// create table which contains
  4 Fields like _id, no, name,
  email later we will fetch
  details using these column names
   @override
Public void onCreate( SQLiteDatabase
                      db) {
db.execSQL(" create table content
table(_id int(3) primary key,
" + " no int(3), name varchar(20),
email varchar(20)); ");
  }
  @override
  P v onUpgrade( SQLiteDatabase
  db, int oldversion, int newVer-
   sion) {
     }
}
```

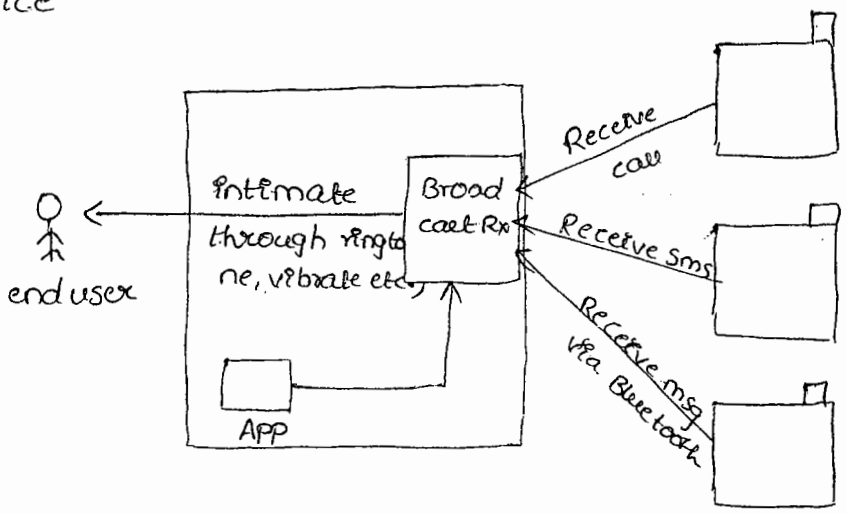
ListAdapter.java

```
P c ListAdapter extends SimpleCur-
    sorAdapter {
  // Adapter class to fetch data
  Private Cursor mCursor;
private LayoutInflater mInflater;
public ListAdapter( Context context,
int layout, Cursor c, String[] from,
int[] to, LayoutInflater inflater){
super( context, layout, c, from, to);
     mCursor = c;
     mInflater = inflater;
  }
 @ Override
public View getView( int position,
 View contentView, ViewGroup pa-
   rent){
  View v = mInflater.inflate(R.
    layout.item, null);
  TextView tv=( TextView) v.findV-
  iewById( R.id.xTv);
  mCursor.moveToPosition(position);
  String str = mCursor.getString(
  mCursor.getColumnIndex("no"))
  + " " + mCursor.getString(mCur-
  sor.getColumnIndex("name")) +
  " " + mCursor.getString(mCursor.
   getColumnIndex("email"));
   tv.setText(str);
   return v;
 }
}
```

10/01/2013

BROADCAST RECEIVER

→ broadcastReceiver is existing as a class in android used to intimate to the end user whenever any information (or) data is received by the current android device either from same Application of same device or from other any device



→ BroadcastReceiver is mainly classified into two types.

1) predefined BroadcastReceiver
2) userdefined BroadcastReceiver

1. predefined BroadcastReceiver:

→ These are the predefined classes existing in android API used to intimate to the end user about predefined data or information which is received by applications of same device or other devices like receiving calls, receiving sms, receiving notifications (Low charging, missed calls -- etc.,) through a Ring (or) Vibrate (or) voice message etc.,

2. Userdefined BroadcastReceiver:

If any BroadcastReceiver class is designed by the developer to give the intimation to the enduser whenever that device is receiving any information is known as "userdefined BroadcastReceiver"

Rules to design userdefined BroadcastReceiver:

1. Create a userdefined class and Make that one as derived class of BroadcastReceiver class
2. Override onReceive( )

Syntax:

```
public class <CN> extends BroadcastReceiver
{
  @override
  public void onReceive( Context context, Intent intent)
   {
     =
   }
}
```

→ onReceive() contains main logic to intimate about any Receiving information

3. Register BroadcastReceiver class in the manifest file as shown below.

```
<manifest>
    <application>
        <receiver a:name=".BCRCN">
        </receiver>
    </application>
</manifest>
```

NOTE:

every userdefined BroadcastReceiver is associated with an activity class

Design an android application to give the intimation about Alarm to the end user through BroadCaltReceiver (By providing Toast message and vibrating the device)

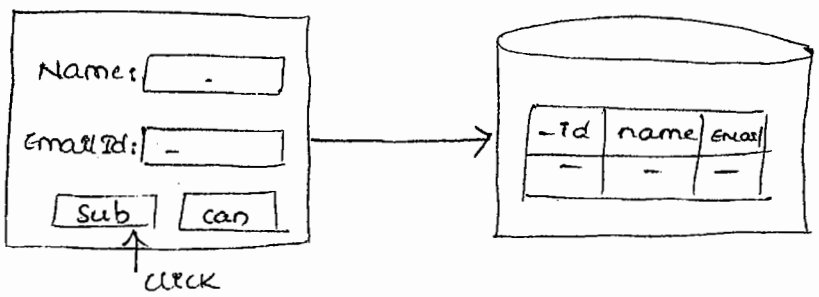
NOTE: in manifest file

We must take the permission to use the vibrator as shown below.

```
<uses-permission
    android:name="android.permission.VIBRATE">
</uses-permission>
```

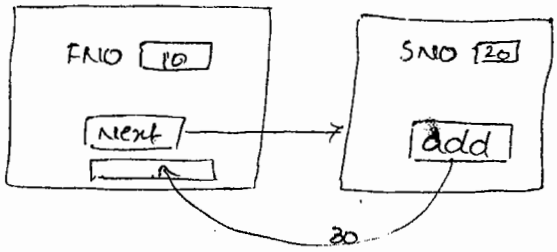
design & write android application for following

1)



Whenever record is inserted then a song should be played through Broadcast Receiver

2)



green color should be blinked as a intimation while displaying result

11/1/2013

NETWORKING:

→ Interconnection of two mobile devices or mobile device with any other remote device like computer, satellite etc., is known as "Network" or "mobileNetwork"

→ The main advantage with Networking is sharing of data

→ android mainly supports following types of wireless Networks

1) Bluetooth
2) Wi-Fi
3) GPRS
4) GPS

1) Bluetooth:

→ Bluetooth is a wireless mobile technology used to share the data between two Bluetooth devices in a short range (10m)

→ Every Bluetooth device contains an inbuilt hardware equipment called "BluetoothAdapter"

→ every Bluetooth device is having its own predefined Bluetooth application and which performs following operations

i) checking whether device is supporting Bluetooth or not
ii) enabling or disabling Bluetooth
iii) displaying List of Registered or bounded devices
iv) Scanned for surrounded Bluetooth devices
v) pairing with a selected device

→ The above same operations can be performed by the user or developer with the help of android API

→ BluetoothAdapter class :

→ It is a predefined class in "android.bluetooth" package used to work with Bluetooth applications

→ Bluetooth operations can be performed in two different ways

a) Using Intent class

Eg: Intent i = new Intent (BluetoothAdapter. ACTION_REQUEST_ENABLE); } =>

which will enable or turn on Bluetooth etc., operations can be performed.

b) Using predefined BluetoothAdapter class

Eg: BluetoothAdapter ba = getDefaultAdapter(); }
ba.enable(); } =>

which can be used to enable or turn on bluetooth etc., operations can be performed

Methods of BluetoothAdapter class :

1. getDefaultAdapter() :

→ Which can be used to handle Local BluetoothAdapter device, which returns BluetoothAdapter class object if the current device is supporting Bluetooth otherwise returns null reference.

2. isEnabled()

→ Which returns true if the Bluetooth is currently enabled and ready to use. otherwise returns false

3. enable()

→ used to turn on Local Bluetooth

4. disable():

→ used to turnoff the Local Bluetooth

5. getBondedDevices():

→ returns the set of BluetoothDevice objects that are Bonded (paired) to the Local adapter.

6. isDiscovering():

→ returns true if the Local BluetoothAdapter is currently in the device discovery process

7. startDiscovery():

→ used to start the remote device discovery process

8. cancelDiscovery():

→ used to cancel the current device discovery process

9. getRemoteDevice (String address):

→ used to get the BluetoothDevice object for the given Bluetooth hardware address, which also can be used to establish pairing between current and other devices

10. getName():

→ used to get the friendly Bluetooth name of the local Bluetooth adapter

11. setName (String name):

→ used to set a friendly Bluetooth name of the Local Bluetooth adapter

12. getAddress():

→ returns the hardware address of local Bluetooth adapter

13. getProfileConnectionState (int profile):

→ Which returns no. of pairing and unpairing devices with current Bluetooth adapter

14. listenUsingRfcommWithServiceRecord (String name, UUID uuid)

→ Which can be used to make a remote device connecting to this socket will be authenticated and communication on this socket will be encrypted (use accept() to retrieve incoming connections from ListeningBluetoothServerSocket)

15. listenUsingInsecureRfcommServiceRecord (String name, UUID uuid)

→ Which will make a Remote device connecting to this socket without authentication (that means insecured connection)

16. getState()

→ get the current state of local Bluetooth adapter

12/1/2013

Bluetooth permissions:

1) <uses-permission

android:name = "android.permission.BLUETOOTH "/>

→ Which can be used to take the permission about using of Bluetooth hardware device.

2) <uses-permission

android:name = "android.permission.BLUETOOTH_ADMIN "/>

Which can be used to enable the default permissions or settings of Bluetooth

Design an android application working with Bluetooth

main.xml

```
<LinearLayout a:l-w="f-p"
              a:l-h="f-p"
              a:orientation="vertical">
  <ToggleButton a:id="@+id/toggleButton1"
                a:l-w="w-c"
                a:l-h="w-c"
                a:onClick="onClick"
                a:text="ToggleButton"/>
</LinearLayout>
```

Activity class

BlueToothActivity.java

```
Public class BlueToothActivity extends Activity {
    ToggleButton tb;
    BluetoothAdapter mBluetoothAdapter;
  // called when the activity is first created
   @override
P  v onCreate(Bundle b) {
   Super.onCreate(b);
   setContentView(R.layout.main);
    tb=(ToggleButton) findViewById(R.id.toggleButton1);
   }
   public void onClick(View v){
     Switch(v.getId()){
case R.id.toggleButton1 : if((tb).isChecked()){
                mBluetoothAdapter = BluetoothAdapter.getDefaultA-
                                    dapter();
                S.O.pln("Got BlueTooth Adapter");
              if(mBluetoothAdapter==null){
        Toast myToast1=Toast.makeText(this,"BlueTooth doesn't
            Support", Toast.LENGTH_SHORT);
```

```
            myToast1.show();
        }
    if (!mBluetoothAdapter.isEnabled()){
        // Enable Bluetooth
    Intent enableIntent = new Intent(BluetoothAdapter.ACTION_
                                    REQUEST_ENABLE);
    startActivityforResult(enableIntent, 5);
    }
    // create intent to discover the other bluetooth devices
Intent discoverableIntent = new Intent(BluetoothAdapter.ACTION_REQUE-
                                    ST_DISCOVERABLE);
discoverableIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_
                                    DURATION, 300);
startActivity(discoverableIntent);
}
else{
Toast myToast = Toast.makeText(getApplicationcontext(), "Bluetooth
        turned off", Toast.LENGTH_SHORT);
    myToast.show();
mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if(mBluetoothAdapter == null){
    Toast myToast1 = Toast.makeText(this, "Bluetooth doesn't Support",
                                    Toast.LENGTH_SHORT);
    myToast1.show();
}
if(mBluetoothAdapter.isEnabled()){
    // disable Bluetooth
if(mBluetoothAdapter != null){
    String address = mBluetoothAdapter.getAddress();
    String name = mBluetoothAdapter.getName();
// S.o.pln("Bluetooth Address "+address+" BlueTooth Name "+
                                    name);
Toast myToast1 = Toast.makeText(getApplicationcontext(), address+" "+
                                    name, Toast.LENGTH_SHORT);
    myToast1.show();
    mBluetoothAdapter.disable(); } } } } }
}
```

2. Wifi (wireless fidelity)

→ Wifi is advanced to Bluetooth which can be used to transfer the data (or) share the data between a Blue wifi enabled mobile device to wifi enabled Network.

→ Using wifi technology data can be sharable in some large distance (upto 500m)

wifi operations

1. checking whether wifi is supporting by the mobile device (or) not
2. enable (or) disable wifi
3. Scan for New wifi Networks
4. Add or remove Networks etc.,

→ Android API Supports following predefined class to work with wifi and to perform above operations.

`android.net.wifi.WifiManager`

METHODS:

1. isWifiEnabled():

returns

→ returns true if wifi is enabled otherwise false

2. setWifiEnabled(boolean value)

→ used to set enable or disable wifi (True → enabled, false → disabled)

3. startScan()

→ which will scan for access points

4. addNetwork(WifiConfiguration config)

→ Add a new network description to the set of configured networks

5. updateNetwork (WifiConfiguration config):

→ update the network description of an existing configured network

6. getConfiguredNetworks()

→ returns a list of all the Networks which are already configured with the current device.

7. removeNetwork (int netId)

→ remove the specified Network from the list of configured Networks

NOTE:

For other methods refer WifiManager class in android API

Wifi Permissions:

1) <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE"/>

→ used to take the permission to access wifi

2) <uses-permission
android:name="android.permission.UPDATE_DEVICE_STATS"/>
↓
statistics

→ which can be used to take the permission to update statistics of device like add network, update n/w, remove n/w etc.)

3) <uses-permission
android:name="android.permission.CHANGE_WIFI_STATE"/>

→ used to take the permission to change the wifi state

Syntax to get Wifi service

```
WifiManager wm = (WifiManager) this.getSystemService(
                                    context.WIFI_SERVICE);
```

Design an android application to develope Wifi based application

main.xml

```
<LL  a:l-w = "f-p"
     a:l-h = "f-p"
     a: orientation = "vertical">
<Textview  a:l-w = "f-p"
           a:l-h = "w-c"
           a:text = "hello"/>
</LinearLayout>
```

Activity class

WifiStatusActivity.java

```
Public class WifiStatusActivity extends Activity
{
   private WifiManager wm;
   @override
   public void onCreate(Bundle b)
   {
     super.onCreate(b);
     setContentview(R.layout.main);
    // get the Wifi Service fxom our device
    wm = (WifiManager) this.getSystemService(Context.
                                    WIFI_SERVICE);
    // check the Wifi is currently turned on or turned off
        if(wm.isWifiEnabled()){
     // wm.setWifiEnabled(false);  // Turn on/off our wifi
```

```
Toast.makeText(this, "Wifi Enabled", Toast.LENGTH_LONG).
                                                    show();
    boolean b = wm.startScan();
     if(b == true)
     {
       List<ScanResult> l=wm.getScanResults();
          S.o.pln(l);
      }
     else
      {
       s.o.p("no networks found");
       }
     }
     else
     {
     // wm.setWifiEnabled(true);
        Toast.makeText(this, "Wifi Disabled", Toast.LENGTH_
                                                    LONG).show();
       }
    }
}
```

3. GPRS (General packet Radio Service)

→ It is acting as internet for mobile device, if any device is supporting GPRS then we can access content from the server system

→ android supports to develope GPRS based applications in two different ways

1) using Webview class

2) using HttpUrlConnection class

→ To design any GPRS based application it is very mandatory to take internet permission in manifest file.

```
<uses-permission
    android:name="android.permission.INTERNET"/>
```

1) Using Webview class

→ with the help of Webview class we can load webpages on android device and Websettings class can be used to enable (or) disable settings to the webview like Javascript enabling, storing cookies settings etc.,

→ Both Webview and WebSettings are predefined classes in android.webkit package

Design an android application to load any url in the Webview

NOTE: LoadUrl() is a predefined method of Webview class used to load the url's

```
wv.loadUrl("http://www.gmail.com");
```

(wv: object reference of Webview class)

main.xml

```
<LinearLayout  a:l_w="f_p"
               a:l_h="f_p"
               a:orientation="vertical">
```

```
<Webview a:id = "@+id/wv"
         a:l-w = "f-p"
         a:l-h = "w-c" />
</Linearlayout>
```

Activity class

```
public class GPRSConnectionActivity extends Activity
{
// called when the activity is first created
    @override
    public void onCreate(Bundle b)
    {
        super.onCreate(b);
        setContentView(R.layout.main);
        Webview w = (Webview) findviewById(R.id.wv);
        w.loadUrl("http://www.icicibank.com");
        WebSettings ws = w.getSettings();
        ws.setJavaScriptEnabled(true); // Java script
                                          enabled
        ws.setLoadWithOverviewMode(true);
    // compress according to the emulator size
    }
}
```

Checking Network Status:

→ android supports following predefined class (Service class) to check the Network status

```
ConnectivityManager connMgr = (ConnectivityManager)
    getSystemService(CONNECTIVITY_SERVICE);
```

getNetworkInfo()

It is predefined method of ConnectivityManager class used to get information about current Network (given network type)

Type of Networks

1) TYPE_BLUETOOTH → Bluetooth Network
2) TYPE_MOBILE → mobile 3G Network
3) TYPE_MOBILE_MMS → an mms specific mobile Data connection.
4) TYPE_WIFI → Wifi Data connection
5) TYPE_WIMAX → represents Wimax Data connection

→ The above Method returns NetworkInfo class object

Eg:
```
NetworkInfo wifi = connMgr.getNetworkInfo(ConnectivityManager.TYPE_WIFI);

if(wifi.isAvailable()){
    Toast.makeText(this, "wifi is supporting",
        Toast.LENGTH_LONG).show();
}
```

Design an android appln to check Network status:

main.xml

```
<LL a:l_w="f_p"
    a:l_h="f_p"
    a:orientation="vertical">
```

```
<Button a:id = "@+id/btn"
        a:l-w = "f-p"
        a:l-h = "w-c"
        a: text = "Show Network Status"
        a: onClick = "onClick"/>
</LinearLayout>
```

Activity class

```
Public class CheckNetworkStatusActivity extends Activity{
  //called when the activity is first created
    @Override
  p v onCreate (Bundle b) {
    super.onCreate(b);
    setContentView(R.layout.main); }
  p v onClick (View v){
    chkConnectionStatus(); }
  void chkConnectionStatus() {
   ConnectivityManager connMgr = (ConnectivityManager) this.getSy-
      stemService(Context.CONNECTIVITY_SERVICE);
   android.net.NetworkInfo wifi = connMgr.getNetworkInfo(
          ConnectivityManager.TYPE_WIFI);
   android.net.NetworkInfo mobile = connMgr.getNetworkIn
          fo(ConnectivityManager.TYPE_MOBILE);
    if( wifi.isAvailable()) {
   Toast.makeText(this, "wifi", Toast.LENGTH_LONG).show();
    }
    else if(mobile.isAvailable()){
   Toast.makeText(this, "Mobile 3G", Toast.LENGTH_LONG).
                                          show();
    }
    else{
  Toast.makeText(this, "No Network", Toast.LENGTH_LONG).
                                          show();
    }
  }
}
```

2. Using HttpUrlConnection class:

→ It is a predefined class in java.net package used to establish connection with HttpServer either for writing data (requesting) or for reading data (getting response)

OpenConnection()

Which can be used to open a new connection for the given URL, it is a predefined Method of URL class

connect()

It is a predefined method of URLConnection class used to establish connection with the given httpServer

→ HttpURLConnection, URLConnection, URL are the predefined classes of java.net package

NOTE:

using above three classes and its related methods we can able to send the data to the server and also able to get the data from the same httpServer

Design an android application to download the content from a specific httpServer using HttpURLConnection class

main.xml

```
<LL  a:l_w = "f_p"
     a: l_h = "f_p"
     a: orientation = "vertical" >
<Textview a: id = "@+id/text"
          a: l_w = "w_c"
          a: l_h = "w_c"
          a: text = "Textview"/>
</LinearLayout>
```

Activity class

```
Public class HttpURLConnectionAndroidActivity extends
                                    Activity
{
  // called when activity is first created
     ImageView img;
   @Override
    public void onCreate(Bundle b)
     {
      super.onCreate(b);
      setContentView(R.layout.main);
      String str = DownloadText("http://www.edumobi-
                                  le.org/android/");
      TextView txt = (TextView) findViewById(R.id.text);
       txt.setText(str);
     }
    private String DownloadText(String URL)
      {
        int BUFFER_SIZE = 2000;
        InputStream in=null;
       try
         {
           in=OpenHttpConnection(URL);
         }
        catch(IOException e1)
          {
            e1.printStackTrace();
            return "";
          }
  InputStreamReader isr = new InputStreamReader(in);
     int charRead;
```

```
String str = " ";
char[] inputBuffer = new char[BUFFER_SIZE];
try
{
 while((charRead = isr.read(inputBuffer)) > 0)
 {
  //convert the chars to a string
 String readString = String.copyValueOf(inputBuffer,
                                        0, charRead);
   str += readString;
 inputBuffer = new char[BUFFER_SIZE];
 }
 in.close();
}
catch(IOException e)
 {
  e.printStackTrace();
  return "";
 }
return str;
}
private InputStream OpenHttpConnection(String urlString)
        throws IOException
 {
  InputStream in=null;
   int response = -1;
  URL url = new URL(urlString);
   URLConnection conn = url.openConnection();

   if(!(conn instanceof HttpURLConnection))
   {
```

```
throw new IOException("Not an HTTP connection");
    }
    else
    {
      try
      {
        HttpURLConnection httpconn = (HttpURLConnection)
                                                conn;
        httpconn.setAllowUserInteraction(false);
        httpconn.setInstanceFollowRedirects(true);
        httpconn.setRequestMethod("GET");
        httpconn.connect();

        response = httpconn.getResponseCode();
        if(response == HttpURLConnection.HTTP_OK)
        {
          in = httpconn.getInputStream();
        }
      }
      catch(Exception ex)
      {
        throw new IOException("Error connecting");
      }
    }
    return in;
  }
}
```

17/1/2013

4) GPS (Global Positioning System)

→ It is one of the mobile technology used to communicate current mobile device with the satellite. With the help of GPS we can able to find current location of that device on the globe. and to show the same location in real device a dummy globe is available in the form of map.

→ By default every android device supports these maps technically known as "googlemaps"

Services offered by Google Maps:

1. showing current and destination Location.
2. showing nearest restaurants, theatres, ... etc.,
3. showing driving directions (Route Map) between given source and destination .. etc.,

→ Google provides a seperate API along with android API so that developer can developed their own Location based application

Example 1: sending current location details in the form of SMS

Example 2: setting alarm based on Location ... etc.,

finding Current Location:

→ If we want to find current Location, device should support following.

1. Both GPS and GPRS
2. Sim Service (SIM card)

NOTE: It is very important to take GPS permission in the manifest file while developing Location based application.

< uses-permission

android:name = " android.permission.ACCESS_FINE_LOCATION " >

</uses-permission>

→ android API contains a predefined class called "LocationManager" (service class), used to find the current location.

Syntax:

```
LocationManager lm = (LocationManager) getSystemService (context.LOCATION_SERVICE);
```

Q. Design an android application to find the current location

```
Public class findCurrentLocation extends Activity
{
    Public void onCreate(Bundle b)
    {
        Super.onCreate(b);
        SetContentView(R.layout.main);
        LocationManager lm = (LocationManager) getSystemService(context.LOCATION_SERVICE);
        LocationListener LL = new mylocationlistener();
        (lm.requestLocationUpdate (LocationManager.
                GPS_PROVIDER, 0, 0, LL);
                                  starting latitude   starting longitude
    }
    Private class mylocationlistener implements LocationListener
    {
        if (l != null)
        {
```

18/1/2013 Loading Updated GoogleMaps in Real device:

→ This is possible by designing an android application with the help of GoogleMaps API Key

Steps to generate API Key:

Step 1:

Verify whether debug.keystore file is available on your system, it is usually in the "C:\Documents and Settings\Welcome (or) administrator\.android folder on windows OS."

Step 2:

Use the keytool utility of JDK to generate certificate finger print code. keytool utility comes with JDK instauation by default, once we navigated to the proper directory, type following command in the cmdprompt

```
Cmd> cd C:\Documents and Settings\Administrator\
     .android

C:\Documents and Settings\Administrator\.android>
Key Keytool -list -alias androiddebugkey -storepass
    android -keypass android -keystore debug.keystore
```

→ once we hit the Enter key, Finger print code will be generated Similarly as shown below

D1:16:4A:BD:73:73:AU:56:9D:CD:9A:44:A2:6C:
11:AC

Step3:

Now open your Browser and go to the following URL

https://developers.google.com/maps/documentation/android/v1/maps-api-signup

→ Accept the agreement and paste the finger print code and click Generate API Key

NOTE:

Use the above generated API key in the android application for updating Google Maps.

→ following website contains steps to generate API Key

http://www.zemwebdevelopment.com/dev/a35/android-How-To-Set-Up-an-API-Key-for-Google-Maps.html

Working with updated Google Maps:

→ Android API Supports following predefined classes to work with our own userdefined updated Google Maps.

1. Mapview
2. MapActivity
3. Mapcontroller
4. GeoPoint

1. Mapview:

It is a view component, makes the Http request to google map server to display the google maps from the server. In a layout xml file we should use the Mapview component with Google Map unique API Key.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout>
    <Mapview
```

```
android:id="@+id/mapview1"
android:l_w="f_p"
android:l_h="f_p"
android:enabled="true"  // map and default controls are enabled
android:clickable="true"  // enable zooming
android:apikey="------"/>
</LinearLayout>
```

2. MapActivity:

MapActivity class is to use the code behind Mapview. for example pointing a location on the google map, setting the controls to the Googlemap, setting the display view modes for the Googlemap, setting the driving directions on the Googlemap, Setting Mapping overlight point etc...

3. MapController:

This is the controller class for the Googlemaps especially used for the GoogleMap setting like zoom controlls, Map overlays (fixing a point for the specific address on google maps with textual data, with a pin mark).

4. GeoPoint:

It is the class used to set a geographical point on the google maps, every geographical point have Latitude and longitude values.

Permissions

Following permissions should be taken in manifest file while working with GoogleMaps.

1. <uses-permission

a:name = "android.permission.INTERNET"/>

→ Which is used to access google maps through GPRS

2. <uses-permission

a:name = "android.permission.ACCESS_COARSE_LOCATION"/>

→ which is used to access google maps through wi-fi

3. <uses-permission

a:name = "android.permission.ACCESS_FINE_LOCATION"/>

→ which is used to access google maps through GPS

19/1/2013

NOTE:

It is very mandatory to import Google API through android Sdk Manager before working with location based applications.

→ While using <Mapview> in xml file we must write the fully qualified name as shown below.

<LinearLayout ... =>

<com.google.android.maps.MapView

a:apikey = "..."/>

</LinearLayout>