There Is two main files for every activity , the xml file for the design , in that we can drag and drop and also type it in text. Second file is the java file… which is the brain of the design , i.e. how the design will work.

There is also a main file in manifest folder, which contains the AdroidManifest.xml file, which is first opened and executed by the android by opening, and then it opens the activity first which contains the word LAUNCHER, that is the first activity to be executed.

Import android.util.Log; //package for log datas

Log.i(“TheTagUsed”,”TheMessageToBePromt”); //used to promt message when this line is excecuted , the second parameter is displayed

It is good to make your every string as a string resource due to many reasons, say for instance the thing

is needed to be converted to spainish or hindi or any other language so that only your string

resource would be changed and not the complete activity xml needed to be searched for, so it’s a lot

easier for the system to go through.

Import androd.widget.RelativeLayout; //since we are using relative layout and similar if any other layout we are using

Import android.widget.button; //for using button

For things to work actually…we need to create event listener so as the button for instance would listen to the event it is expected to listen…

Import following packages:-

Android.view.View;

Android.widget.Button;

Android.widget.TextView;

After that, do the following in the onCreate method

Button <button\_name\_that\_i\_want\_to\_use> = (Button)findViewByID(R.Id.Shreyansbutton);

//here Shreyansbutton is the id given to the button

//suppose the variable’s name used is Shreyan\_button

Shreyan\_button.onClickListener(

New button.onClickListener(){

Public void onClick(view v){

TextView <The\_text\_block> = (TextView)findViewById(R.Id.Shreyanstext);

//say the variable used for the text piece is Shreyan\_text.setText(“the string after the click just to display that the button is clicked”);

}});

**now for gestures…first we have to import following packages… :-**

android.view.TextVIew;

android.view.MotionEvent;

android.view.GestureDetector;

android.support.v4.view.GestureDetectorCompat;

then change the main class’s statement to:-

public class MainActivity extends ActionBarActivity implements GestureDetector.OnGestureListener, GestureDetector.OnDoubleTapListener{

…

..

Private TextView ShreyanMessage;

Private GestureDetectorCompat GestureDetector;

//then just go for ALT + INS , and select user defined method, so in that oress okay it will insert all types

//of gestures..and play in side every one accordingly

//One last thing…we need to override a method named onTouchEvent()

//add a line in it before statement containing super keyword appears…that is:-

This.gestureDetector.onTouchEvent(event); //here event is the variable passed by the //method onTouchEvent

}

/////

//when working with fragments , import:-

import android.app.Activity; //to talk with other activity

////If v r accessing internet in any form, v need to write this one in the manifest so that , the app knows ////we need to do access to the net..

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

//i.e. this is the line due to which after the installation it asks for permission, that this app wants permission to access internet, just like it ask for it needs access to memory etc…

Toast toast = Toast.*makeText*(**this**, **"Could not load dictionary"**, Toast.***LENGTH\_LONG***);  
toast.show();

//To make thing like when we insert headphone jack or something…the black thing that appers for a //while

//In Android Studio ctr + shift + / … yields to commenting the complete section…and same shortcut for //uncommenting it

//good option in place of doing what we used to do earlier…for event listners

View.OnClickListener **generalClickListner** = **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(**final** View v){  
 **switch** (v.getId())  
 {  
 **case** R.id.***BT\_Challenge***:  
 *//if()* **break**;  
 **case** R.id.***BT\_Reset***:  
 **break**;  
 }  
 }  
  
};

…

And in the place where needed:-

Button BT\_Challenge = (Button)findViewById(R.id.***BT\_Challenge***);

BT\_Challenge.setOnClickListener(**generalClickListner**);

/////

Assests folder is very useful in android Studio… in that we can put our txt files , font files for some particular fonts…etc

To connect Android Studio with git which is very important…steps to be followed are:-

-In Settings> Version Control > git > in that in the most upper bar set the path of git.exe in your computer , the one which is in cmd folder in git main folder.

- Second in Settings > Version control > github > in that… put host as github.com …then in login put github username an password as github username’s password and then apply

-click test

-then to share in public repo form here for the first time click on VCS > Import into version control > share project on github.

///Commiting and pushing

* After changing press ctr + k or commit shortcut button from version control toolbar or from VCS drop down menu
* Go ahead and do commit and push in the commit button showed…actually the commit button has a arrow on its right and by hovering on it you can see all three options …i.e. commit , commit and push and another one option. Say You did commit and push
* After that that dialog box disappears and another one comes , between their transitions the changes have been committed locally and the current dialog box is there for pushing the stuff up…

Enabling and using crash analytics in android is very necessary since if say ur app crashes and no one reports the crash then u wont be able to know that a crash has ever happen there fore it is necessary to use it.

Fabric is the most commonly used one

After setting it up for the first time

Click fabric on top and go through the stuff , i.e. installing crash analytics in our current project .

Then all the crashes are shown in our dashboard i.e. <https://fabric.io/dashboard> from there we can see it .