Android App Common Tasks library

Android App Common Tasks library is developed to reduce efforts to achieve common features of the android apps. While developing the apps, we realized that we're coding for many common features in all the apps. For e.g. check the network's availability, using shared preferences, parsing, etc. And like us, many other android developers might be doing the same. So it needs to be reduced for all to save the development time with ease. This is how an idea popped in our mind, and we decided to develop an SDK which can reduce developers' time and efforts.

You can get the code from Github and submit common methods that you think can be useful to other android developers.

Here is the demo apk file that you can install and check in your devices.
https://github.com/multidots/android-app-common-tasks/blob/master/docs/CommonTaskDemo.apk

This SDK includes following common features.

1. **Check empty EditText**

   This method is used to check whether data is available or not in EditText.

   Parameter:

   edt_name (Pass EditText name).

   Returns:

   Returns **true** if data is available**,** else **false**.

   How to use:

   Common.isEmptyEditText (edt_name);

2. **Check network availability**

   This method is used to check network availability in device.

   Parameter:

   mContext (Pass application context)

   Returns:

   Returns **true** if network is available**,** else **false**.

   How to use:

   Boolean isNetworkAvailable = Common.isNetworkAvailable (mContext);

3. **Check Email**

   This method is used to check whether email is valid or not.

Parameter:

strEmail (Pass email address)

Returns:

Returns **true** if email is valid, else **false**.

How to use:

Boolean isEmailValid = Common.isEmailIdValid (strEmail);

4. **Get current Date in String format.**

   This method is used to get current date in string format.

   Parameter:

   strDate (Pass date format as a string).

   Examples:

   yyyy-MM-dd
   yyyy-MM-dd HH:mm
   yyyy-MM-dd HH:mmZ
   yyyy-MM-dd HH:mm:ss.SSSZ
   yyyy-MM-dd 'T' HH:mm:ss.SSSZ

   Returns:

   Current Date in **string** format, otherwise returns **null**.

   How to use:

   String strDate = Common.getCurrentDate (strDate);

5. **Get Device id from device.**

   This method is used to get device **Id**.

   Note: You need to add following permission in manifest file READ_PHONE_STATE.

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

   Parameter:

   mContext (Pass application context).

   Returns:

   Device Id in **string**, otherwise returns **null**.

   How to use:

String strDeviceId = Common.getDeviceId (mContext);

6. **Set Preferences and get Preferences. (String, int, long, Boolean, Float).**

   Setting preferences...

   setStringPrefrences

   This method is used to set string preference.

   Parameter:

   mContext (Pass application context).

   prefName (Name of Preference)

   Value (Value of Preference)

   PREFS_FILE_NAME (file name of preferences).

   Returns:

   This method does not return any value.

   How to use:

   Common.setStringPrefrences(context, prefName, Value, PREFS_FILE_NAME);

   Getting preferences...

   getStringPrefrences

   This method is used to get string from preference.

   Parameter:

   mContext (Pass application context)

   prefName (Name of Preference)

   PREFS_FILE_NAME (file name of preferences)

   Returns:

   This method does not return any value.

   How to use:

   Common.setStringPrefrences(context, prefName, PREFS_FILE_NAME);

7. **Clear Preferences**

   This method is used to clear all the preferences from device.

Parameter:

mContext (Pass application context)

PREFS_FILE_NAME (file name of preferences)

Returns:

This method does not return any value.

How to use:

Common.removeAllPrefrences (context, PREFS_FILE_NAME);

8. **Get Current Location**

   This method used to get current location.

   Parameter:

   mContext (Pass application context)

   Returns:

   Returns location object.

   How to use:

   Location location = Common.getCurrentLocation (mContext);

9. **Apply Pinch Zoom on image.**

   This method is used to apply pinch zoom functionality on image.

   Parameter:

   imgPinchZoom (Pass Imageview object)

   Returns:

   This method does not return any value. It just applies Pinch Zoom on ImageView.

   How to use:

   Common.applyPinchZoomOnImage (imgPinchZoom);

   Note: Use this ImageView in xml file.

```
<ImageView
  android:id="@+id/imageView1"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:scaleType="matrix"  // Do not forget to add this line
  android:src="@drawable/airdroid" />
```

10. **Get application Launcher Icon**

    This method is used to get application launcher icon.

    Parameter:

    mContext(Pass you application context).

    Returns:

    drawable of the application launcher icon.

    How to use:

    Drawable drawable = Common.getAppIcon (mContext);

11. **Local Notification**

    This method is used to send local notification in application.

    Parameter:

    mContext (Pass application context)

    strTitle (Title of notification)

    strMessage (Content of notification)

    Intent (Pass intent to open application or null to clear notification on touch)

    How to use:

    Common.sendLocalNotification (mContext, title, message, mIntent);

12. **Get random Character**

    This method is used to get random character.

    Returns:

    Returns any random character.

    How to use:

    char character = Common.getRandomCharacter();

13. **Disable Sleep Mode while using application**

    This method is used to disable sleep mode while using application.

    Parameter:

    mContext (Pass application context)

Returns:

This method does not return any value.

How to use:

Common.disableSleepMode (mContext);

14. **Enable Sleep Mode while using application**

    This method is used to enable sleep mode while using application.

    Parameter:

    mContext (Pass application context)

    Returns:

    This method does not return any value.

    How to use:

    Common.enabelSleepMode (mContext);

15. **Open image from selected directory path**

    This method is used to open image from given directory path.

    Parameter:

    mContext (Pass application context)

    strPath (Directory path of the image)

    For e.g. “ /mnt/sdcard/Picture/imagename.png”

    Returns:

    This method does not return any value.

    How to use:

    Common.openImage (mContext,strPath);

16. **Open video from selected directory path**

    This method is used to open video from given directory path.

    Parameter:

    mContext (Pass application context)

    strPath (Directory path of the video)

For e.g. " /mnt/sdcard/Picture//mnt/Video/1.mp4"

Returns:

This method does not return any value.

How to use:

Common.openImage (mContext,strPath);

17. **Open URL in browser**

    This method is used to open URL in mobile browser.

    Parameter:

    mContext (Pass application context)

    strUrl – String URL to open in browser, like http://www.google.com

    Returns:

    This method does not return any value. It just redirects to browser.

    How to use:

    Common.openURL (mContext,strUrl);

18. **Shows address location on map**

    This method is used for getting location on Google map using specified address.

    Parameter:

    mContext (Pass application context)

    strAddress (Pass address to locate on map)

    Returns:

    This method does not return any value.

    How to use:

    Common.showAddressOnMap (mContext,strAddress);

19. **Create directory in sdcard specified location**

    This method is used to create directory on specified location.

    Parameter:

    mContext (Pass application context)

strPath (Pass you sdcard path), for e.g. ” /mnt/sdcard/specifiedFolder/”

strDirectoryname (Pass directory name)

Returns:

Returns **true** if folder created else **false.**

How to use:

Common.createFolder (mContext,strPath,strDirectoryname);

20. **Download image from URL**

    This method is used to download image from specified URL.

    Parameter:

    strImgURL (image URL in string)

    imgView (imageView in which image is to be set)

    How to use:

    Common.downloadImageFromURL (imageURL,imageview);

21. **Show Date Picker**

    This method is used to open DatePickerDialog and get selected date in TextView.

    Parameter:

    mContext (Pass application context)

    strFormat (format of date)

    txtTextView (Pass TextView in which date is to be set)

    Returns:

    This method does not return any value.

    How to use:

    Common.showDatePickerDialog (mContext, strFormat, txtTextView) ;

22. **Show time picker**

    This method is used to open TimePickerDialog and get selected date in textview.

    Parameter:

    mContext (Pass application context)

mTextView (textView in which selected date has to be set)

Returns:

This method does not return any value.

How to use:

Common.showTimePickerDialog (mContext, mTextView);

23. **Get number of file counts in sdcard directory**

    This method is used to get number of files available in specified sdcarddirectory.

    Parameter:

    strFormat (Pass file format). For e.g. .jpg, .mp3, .png

    strDirectoryPath (Pass path to get files count in directory)

    For e.g. "/mnt/sdcard/Pictures/Screenshots/"

    Returns:

    It returns number of count in **int** format**.**

    How to use:

    int intCount = Common.getFileCounts (strFormat, strDirectoryPath);

24. **Calculate date difference between two dates**

    This method is used to calculate number of days between two dates.

    Note: Both dates must be in same format.

    Parameter:

    date1 (Pass date1)

    date2 (Pass date2)

    Returns:

    It returns **long** value of date difference.
    Positive value for future days and Negative value for passed days from current.

    How to use:

    long longDayCount = Common.calculateDays (date1, date2);

25. **Convert Stringdate to Date format**

    This method is used to convert String Date to Date format.

Parameter:

strDate (Pass string Date)

strDateFormat (Pass string format)

Returns:

It returns Date in date format.

How to use:

Date date = Common.stringToDate(strDate, strDateFormat);

26. **Get device height**

    This method is used to get device height.

    Parameter:

    mContext (Pass application context)

    Returns:

    Returns device height.

    How to use:

    int intHeight = Common.getDeviceHeight(mContext);

27. **Get device width**

    This method is used to get device width.

    Parameter:

    mContext (Pass application context)

    Returns:

    It returns device width.

    How to use:

    int intWidth = Common.getDeviceWidth(mContext);

28. **Get random number**

    This method is used to get random number.

    Parameter:

    number (Maximum number)
    For e.g. if you want to get random number up to **999.**

Returns:

It returns any random picked number.

How to use:

int number = Common.getRandom(number);

29. **Add postfix to numbers**

    This method is used to set postfix to a number.

    Parameter:

    intNumber (Pass integer number to add postfix)

    For e.g. pass 10, it will return **10th.**

    Returns:

    It returns number in string format with added postfix in it.

    How to use:

    String strNumber = Common.getPostFixForNumber(number);

30. **Convert comma separated String to ArrayList**

    This method is used to convert comma separated string to array list.

    Parameter:

    strCommaSeparatedString (Pass comma separated string)

    For e.g. “test1, test2,test3,test4,test5”.

    Returns:

    It returns ArrayList of comma separated string.

    How to use:

    ArrayList arrList = Common.stringToArrayList(strCommaSeparatedString);

31. **Convert ArrayList to comma separated String**

    This method is used to convert ArrayList to comma separated String.

    Parameter:

    arrList (Pass ArrayList)

    Returns:

It returns comma separated string from ArrayList data.

How to use:

String strlist = Common.arrayListToString(arrList);

32. **Play background music**

This method is used to play background music.

Note: In raw folder please add sound with this name **“background_music”.**

Parameters:

mContext(Pass application context).

How to use:

Common.backgroundMusicStart(mContext);

33. **Stop background music**

This method is used for **Stop** background music.
Note: In raw folder please add sound with this name **“background_music”.**

Parameters:

mContext(Pass application context).

How to use:

Common.backgroundMusicStart(mContext);

34. **Apply blur effect on image(Bitmap)**

This method is used to apply blur effect on bitmap image.

Parameter:

mContext(Pass application context).

bitmap(Pass bitmap).

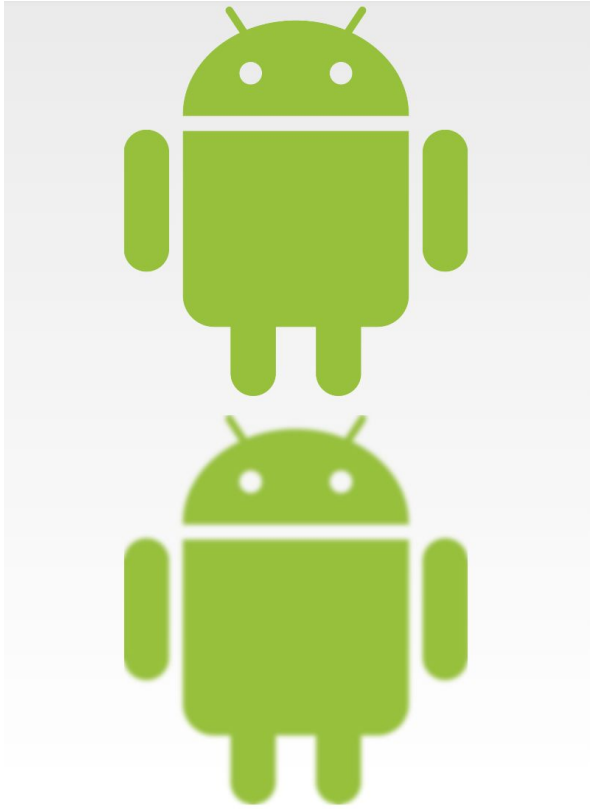
Returns:

It returns blur image in bitmap.

How to use:

itmap bmpImg = Common.blurEffectOnBitmap(mContext, bitmap);

35. **Apply blur effect on image (Drawable)**



This method is used to apply blur effect on drawable image.

Parameter:

mContext (Pass application context)

drawable (Pass drawable object)

radius (for blur effect, integer value it must be below 25)

Returns:

It returns blur image in drawable.

How to use:

Drawable drawableObj = Common.blurEffectOnDrawable(Context, bitmap, radius);

36. **Convert drawable to bitmap**

This method is used to convert drawable to bitmap image.

Parameter:

mContext (Pass application context)

drawableResourceId (drawable Resource id i.e. :R.drawable.ic_launcher)

Returns:

Returns bitmap image.

How to use:

Bitmap bitmap = Common.drawableToBitmap(mContext, drawableId);

37. **Convert bitmap to drawable**

This method is used to convert bitmap image to drawable.

Parameter:

mContext (Pass application context)

bitmap (Pass bitmap)

Returns:

It returns drawable image.

How to use:

Common.bitmapToDrawable(mContext, bitmap);

38. **Set Application Volume**

This method gets device volume and set application sound volume same as device volume.

Parameter:

mContext (Pass application context)

Returns:

This method does not return any value.

How to use:

Common.setCurrentDeviceVolume(mContext);

39. **Set bitmap to preferences**

This method is used to save bitmap image in preferences as string.

Parameter:

bitmap (Pass bitmap)

mContext (Pass application context)

prefName (Pass preferences name)

prefFileName (Pass shared preference file name)

How to use:

Common.setBitmapToPreference(bitmap, mContext, prefName, appName);

40. **Get bitmap image form preference**

    This method is used to get saved Bitmap image from preferences.

    Parameter:

    mContext (Pass application context)

    prefName (Pass preferences name)

    prefFileName (Pass shared preference file name)

    Returns:

    It returns bitmap image.

    How to use:

    Common.getBitmapFromPreference(mContext, prefName, appName);

41. **Get application version code**

    This method is used to get app current version code.

    Parameter:

    mContext (Pass application context)

    Returns:

    It returns version code.

    How to use:

    int intVersionCode = Common.getApplicationVersionCode(mContext);

42. **Set vertical text-view (left & right)**

    The following code is used to set textView on left or right side on the device screen.

    For left side textview:

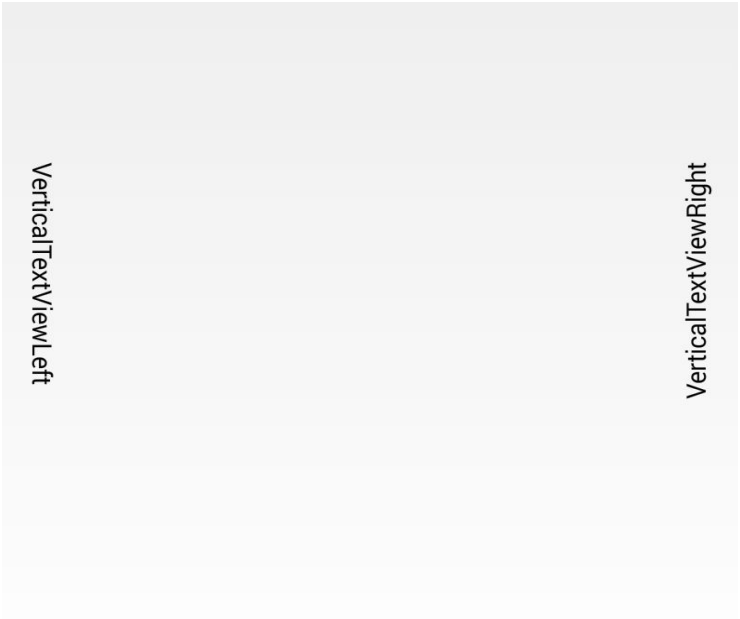
```
<com.common.utils.VerticalTextViewLeft
  android:id="@+id/text1"
```

```
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/hello_world"
  android:textColor="@android:color/black" />
```

For right side textview:

```
<com.common.utils.VerticalTextViewRight
  android:id="@+id/text2"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/hello_world" />
```

Above code will provide below output.



43. Check weather SD card is available on device

Method checks if SDcard is available on device.

Parameter:

mContext (Pass application context)

Returns:

This method returns boolean value.

How to use:

boolean available = Common.isSDCardAvailable(mContext);

44. Show Share Dialog

This method is used to open share dialog.

Note: title, uri, shareText and shareSubject are optional fields.

Parameter:

mContext (Pass application context)

strTitle (Title of dialog)

strUri (Local image uri path)

strShareText (Message to share)

strShareSubject (Subject to share)

Returns:

This method does not return any value.

How to use:

Common.openShareDialog(context , “Share Title”, “/mnt/sdcard/file.jpeg” , “Share Text” , “Share Subject”);

45. Change Device Profile (Silent or Vibrate or Normal)

This method is used to change device profile to Silent,Vibrate or Normal Mode.

Parameter:

mContext (Pass application context)

intMode (Pass mode in integer, i.e. 0 for silent, 1 for vibrate, 2 for normal)

Returns:

This method does not return any value.

How to use:

Silent
Common.chooseProfile(mContext,0);

Vibrate
Common.chooseProfile(mContext,1);

Normal

Common.chooseProfile(mContext,2);

46. **Make bitmap Rounded Corner**

    This method is used to make bitmap to rounded corner or circled bitmap.

    Parameter:

    bitmap (Pass bitmap)

    roundPixel (Pass radius in integer)

    Returns:

    This method returns bitmap object.

    How to use:

    Bitmap bitmap = Common.getRoundedCornerBitmap(bitmap , roundPixel);

47. **Show Alert Dialog or Toast**

    This method is used to show AlertDialog or Toast message to user.

    Parameter:

    mContext (Pass application context)

    strTitle (Pass dialog title)

    strMessage (Pass Message)

    toast (true for displaying toast, false for displaying dialog)

    Returns:

    This method does not return any value.

    How to use:

    For Displaying dialog
    Common.showAlertDialog(context , “App Name”, “Welcome User” , false);

    For Displaying toast
    Common.showAlertDialog(context , “”, “Welcome User” , true);

48. **Preventing double click**

    This method is used to prevent user from clicking multiple times.

    Parameter:

view (Pass view to prevent click).
i.e. button, textView, etc.

Returns:

This method does not return any value.

How to use:

Common.preventDoubleClick(btnLogin);

49. Turn Bluetooth or Wi-Fi ON/OFF

WIFI

By this method you can forcefully switch wifi ON/OFF.

Parameter:

mContext (Pass application context)
action (pass ON or OFF string as action)
Returns:

This method does not return any value.

How to use:

Common.onWifi(mContext, action);

Bluetooth

By this method you can forcefully switch Bluetooth ON/OFF.

Parameters:

action (pass ON or OFF string as action)

Returns:

This method does not return any value.

How to use:

Common.onBluetooth(action);

50. Capture Image from camera

This method is used to capture image from camera.

Note:
Call Common.onSaveInstanceState(Bundle) from onSaveInstanceState and onRestoreInstanceState(Bundle) from onRestoreInstanceState of activity

Parameter:

mContext (Pass application context).

CAMERA_CAPTURE_IMAGE_REQUEST_CODE (request code for capturing image)

camera (“Front” or “Back” to open front or back camera)

How to use:

For Front Camera
Common.captureImage(mContext , 123, “Front”);

For Back Camera
Common.captureImage(mContext , 123, “Back”);

51. Pick Image

This method is used to pick image from gallery.

Parameter:

mContext (Pass application context)

CAMERA_PICK_IMAGE_REQUEST_CODE (request code to pick image)

How to use:

Common.pickImage(mContext , 234);

52. Preview Captured Image

This method is used to preview captured image.

Parameter:

ivImagePreview (imageView to preview image)

How to use:

Common.previewCapturedImage(ivImagePreview);

53. Record Video

This method is used to record video.

Note:
Call Common.onSaveInstanceState(Bundle) from onSaveInstanceState and onRestoreInstanceState(Bundle) from onRestoreInstanceState of activity.

Parameter:

mContext (Pass application context)

CAMERA_CAPTURE_VIDEO_REQUEST_CODE (request code for recording video)

camera (“Front” or “Back” to open front or back camera)

How to use:

For Front Camera
Common.recordVideo(mContext , 123, “Front”);

For Back Camera
Common.recordVideo(mContext , 123, “Back”);

54. **Pick Video**

This method is used to pick video from gallery. **Also refer getPath() method**.

Parameter:

mContext (Pass application context)

CAMERA_PICK_VIDEO_REQUEST_CODE (request code to pick video)

How to use:

Common.pickVideo(mContext , 234);

55. **Preview Captured Video**

By this method you can preview captured image.

Parameter:

videoPreview (VideoView in which you want to play Video)

How to use:

Common.previewVideo(videoPreview);

56. **Get path of picked image or video (for all versions)**

By this method you can get path of images, videos and documents from all versions of android.

Parameter:

mContext (Pass application context)

uri (uri you get in onActivityResult(). i.e. Uri fileUri = data.getData();)

How to use:

Common.getPath(mContext, fileUri);

57. Integrating Social Platforms

Note :
For a complete activity implementation, look at **Common Activity below step 9.**
If you want **Skills** from linkedin, follow from **Step 9.**

Follow the steps below to integrate **LinkedIn, Twitter, Facebook, Google Plus**.

1) Create a Class that implements **DialogListener** in **activity** which is Response listener.
For e.g.

```
private final class ResponseListener implements DialogListener {
    @Override
    public void onComplete(Bundle values) {
        // Do what you want on completion of social login
    }

    @Override
    public void onError(SocialAuthError error) {
       error.printStackTrace();
    }

    @Override
    public void onCancel() {
    }

    @Override
    public void onBack() {
    }
}
```

2) Create new SocialAuthAdapter in **onCreate of Activity.**
```
SocialAuthAdapter adapter = new SocialAuthAdapter(new ResponseListener())
```
3) Call authorization for social site **on Click of Button**
```
adapter.authorize(mContext, SocialAuthAdapter.Provider.LINKEDIN);
                                                                    .FACEBOOK
                                                                    .TWITTER
                                                                    .GOOGLEPLUS
```
4) For Signing Out on **SignOut button Click**
```
adapter.signOut(mContext, SocialAuthAdapter.Provider.LINKEDIN.toString());
```

5) For Posting Message on **Facebook , Twitter , Linkedin.**

a) Create Class that implements SocialAuthListener<Integer> **in activity**
e.g :
```
private final class MessageListener implements SocialAuthListener<Integer> {
    @Override
    public void onExecute(String provider, Integer status) {
       if (status.intValue() == 200 || status.intValue() == 201 || status.intValue() == 204)
            // "Message posted on" + provider
       else
            // "Message not posted" + provider
        }
```

```
        @Override
        public void onError(SocialAuthError e) {
           e.getInnerException().printStackTrace();
        }
      }
```

b) Call update Status **on Click of Button**

```
adapter.updateStatus("Hi Test Status Update @" + Calendar.getInstance().getTimeInMillis(),
new MessageListener(), false);
```

6) For posting Image on **Facebook and Twitter.**

a) Create Class that implements SocialAuthListener<Integer>

e.g :

```
 private final class UploadImageListener implements SocialAuthListener<Integer> {
        @Override
        public void onExecute(String provider, Integer t) {
           if (status.intValue() == 200 || status.intValue() == 201 || status.intValue() == 204)
                    // "Message posted on" + provider
           else
                    // "Message not posted" + provider
            }
        @Override
        public void onError(SocialAuthError e) {
           e.getInnerException().printStackTrace();
        }
       }
```

b) Call upload image **on Click of Button**

```
 Bitmap bitmap = Common.drawableTobitmap(CommonActivity.this, R.drawable.facebook);
          try {
             adapter.uploadImageAsync("Image Message", "icon.png", bitmap, 0,
                  new UploadImageListener());
          } catch (Exception e) {
             e.printStackTrace();
          }
```

7) Get Profile of Social User

a) Create Class that implements SocialAuthListener<Profile> **in activity**

E.g. :

```
private final class ProfileDataListener implements SocialAuthListener<Profile> {
        @Override
        public void onExecute(String provider, Profile profileMap) {
           // Get Details of profile
           Log.d("Validate ID = " , profileMap.getValidatedId());
           Log.d("First Name  = " , profileMap.getFirstName());
           Log.d("Last Name   = " , profileMap.getLastName());
           Log.d("Email       = " , profileMap.getEmail());
           Log.d("Gender            = " , profileMap.getGender());
           Log.d("Country           = " , profileMap.getCountry());
           Log.d("Language          = " , profileMap.getLanguage());
           Log.d("Location          = " , profileMap.getLocation());
           Log.d("Profile Image URL  = " , profileMap.getProfileImageURL());
        }
```

```
        @Override
        public void onError(SocialAuthError e) {
            e.printStackTrace();
        }
    }
```

b) Call get profile **on Click of Button**

```
adapter.getUserProfileAsync(new ProfileDataListener());
```

8) Get Contacts List

a) Create Class that implements SocialAuthListener<List<Contact>> **in activity**

E.g. :

```
 private final class ContactDataListener implements SocialAuthListener<List<Contact>> {
        @Override
        public void onExecute(String provider, List<Contact> contactsList) {
            for( i = 0 ; i < contactsList.size() ; i++) {
            Contact contact =contactsList.get(i);
            Log.d("Contact", "Display Name = " + contact.getDisplayName());
            Log.d("Contact", "Name = " + contact.getFirstName()+" "+contact.getLastName());
            Log.d("Contact", "Contact ID = " + contact.getId());
            Log.d("Contact", "Profile URL = " + contact.getProfileUrl());
            Log.d("Contact", "Profile Image URL = " + contact.getProfileImageURL());
            Log.d("Contact", "Email = " + contact.getEmail());
            }
            }
        @Override
        public void onError(SocialAuthError e) {
            e.printStackTrace();
        }
    }
```

b) Call get Contact List **on Click of Button**

```
adapter.getContactListAsync(new ContactDataListener());
```

9) For getting Skills from **LinkedIn :**

a) start activity for result **on Click of Button**

```
Intent intent = new Intent(this, LinkedInActivity.class);
intent.putExtra("APIKEY", "xxxxxxxxxxxxx");
intent.putExtra("APISECRET", "xxxxxxxxxxxxxxx");
startActivityForResult(intent, REQUEST_CODE);
```

b) onActivityResult

```
@Override
   protected void onActivityResult(int requestCode, int resultCode, Intent data) {
   super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_OK) {
     if (requestCode == 101) {
       try {
        JSONObject personJson = new JSONObject(data.getStringExtra("RESPONSE"));
         text4.setText(" Email id : " + personJson.getString("emailAddress") + " \n Name : " +
personJson.getString("firstName") + "  " + personJson.getString("lastName") + " \n id : " +
personJson.getString("id"));
```

```
                    JSONObject skillsJson = personJson.getJSONObject("skills");
                    JSONArray skillsArray = skillsJson.getJSONArray("values");
                    String personSkills;
                    StringBuilder sbSkills = new StringBuilder();
if (skillsArray != null && skillsArray.length() > 0) {
        for (int i = 0; i < skillsArray.length(); i++) {
          sbSkills.append(skillsArray.getJSONObject(i).getJSONObject("skill").getString("name") + ",");
        }
        personSkills = sbSkills.toString();
        if (personSkills.length() > 0
              && personSkills.charAt(personSkills.length() - 1) == ',') {
           personSkills = personSkills.substring(0, personSkills.length() - 1);
        }
        text5.setText(personSkills);
     }
           } catch (Exception e) {
              e.printStackTrace();
           }
         }
      }
   }
```

CommonActivity.java

```
 public class CommonActivity extends Activity implements OnClickListener {

    TextView text1, text2, text3, text4, text5, text6, text7, text8;
    private SocialAuthAdapter adapter;
    private int provider = 0;
    private ProgressDialog pDialog;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_common);
       adapter = new SocialAuthAdapter(new ResponseListener());

       pDialog = new ProgressDialog(this);
       pDialog.setMessage("Please Wait");

       text1 = (TextView) findViewById(R.id.text1);
       text2 = (TextView) findViewById(R.id.text2);
       text3 = (TextView) findViewById(R.id.text3);
       text4 = (TextView) findViewById(R.id.text4);
       text5 = (TextView) findViewById(R.id.text5);
       text6 = (TextView) findViewById(R.id.text6);
       text7 = (TextView) findViewById(R.id.text7);
       text8 = (TextView) findViewById(R.id.text8);

       text1.setOnClickListener(this);
```

```
    text2.setOnClickListener(this);
    text3.setOnClickListener(this);
    text4.setOnClickListener(this);
    text5.setOnClickListener(this);
    text6.setOnClickListener(this);
    text7.setOnClickListener(this);
    text8.setOnClickListener(this);

}


@Override
public void onClick(View v) {

    if (v == text1) {
        Intent intent = new Intent(this, LinkedInActivity.class);
        intent.putExtra("APIKEY", "bh82t52rdos6");
        intent.putExtra("APISECRET", "zQ1LLrGbhDZ36fH8");
        startActivityForResult(intent, 123);
    } else if (v == text2) {
        text2.setVisibility(View.GONE);
        adapter.signOut(CommonActivity.this, SocialAuthAdapter.Provider.LINKEDIN.toString());
    } else if (v == text3) {
        provider = 2;
        pDialog.show();
        adapter.authorize(CommonActivity.this, SocialAuthAdapter.Provider.FACEBOOK);
    } else if (v == text4) {
        text4.setVisibility(View.GONE);
        adapter.signOut(CommonActivity.this, SocialAuthAdapter.Provider.FACEBOOK.toString());
    } else if (v == text5) {
        provider = 3;
        pDialog.show();
        adapter.authorize(CommonActivity.this, SocialAuthAdapter.Provider.TWITTER);
    } else if (v == text6) {
        text6.setVisibility(View.GONE);
        adapter.signOut(CommonActivity.this, SocialAuthAdapter.Provider.TWITTER.toString());
    } else if (v == text7) {
        provider = 4;
        pDialog.show();
        adapter.authorize(CommonActivity.this, SocialAuthAdapter.Provider.GOOGLEPLUS);
    } else if (v == text8) {
        text8.setVisibility(View.GONE);
        adapter.signOut(CommonActivity.this, SocialAuthAdapter.Provider.GOOGLEPLUS.toString());
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_OK) {
        if (requestCode == 101) {
            try {
                JSONObject personJson = new JSONObject(data.getStringExtra("RESPONSE"));
```

```
                Toast.makeText(CommonActivity.this, "LinkedIn Connected",
Toast.LENGTH_SHORT).show();

//            Get Skills
                JSONObject skillsJson = personJson.getJSONObject("skills");
                JSONArray skillsArray = skillsJson.getJSONArray("values");
                String personSkills;
                StringBuilder sbSkills = new StringBuilder();
                if (skillsArray != null && skillsArray.length() > 0) {
                    for (int i = 0; i < skillsArray.length(); i++) {

sbSkills.append(skillsArray.getJSONObject(i).getJSONObject("skill").getString("name") + ",");
                    }
                    personSkills = sbSkills.toString();
                    if (personSkills.length() > 0
                            && personSkills.charAt(personSkills.length() - 1) == ',') {
                        personSkills = personSkills.substring(0, personSkills.length() - 1);
                    }
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
  }

  // To get status of message after authentication
  private final class MessageListener implements SocialAuthListener<Integer> {
    @Override
    public void onExecute(String provider, Integer t) {
        Integer status = t;
        if (status.intValue() == 200 || status.intValue() == 201 || status.intValue() == 204)
            Toast.makeText(CommonActivity.this, "Message posted on" + provider,
Toast.LENGTH_LONG).show();
        else
            Toast.makeText(CommonActivity.this, "Message not posted" + provider,
Toast.LENGTH_LONG).show();
    }

    @Override
    public void onError(SocialAuthError e) {
        e.getInnerException().printStackTrace();
    }
  }

  private final class ResponseListener implements DialogListener {

    @Override
    public void onComplete(Bundle values) {
        Log.d("Custom-UI", "Successful");
        pDialog.dismiss();

        switch (provider) {
```

```
            case 1:
                Toast.makeText(CommonActivity.this, "LinkedIn Connected",
Toast.LENGTH_SHORT).show();
                text2.setVisibility(View.VISIBLE);
                break;
            case 2:
                Toast.makeText(CommonActivity.this, "Facebook Connected",
Toast.LENGTH_SHORT).show();
                text4.setVisibility(View.VISIBLE);
                break;
            case 3:
                Toast.makeText(CommonActivity.this, "Twitter Connected",
Toast.LENGTH_SHORT).show();
                text6.setVisibility(View.VISIBLE);
                break;
            case 4:
                Toast.makeText(CommonActivity.this, "Google+ Connected",
Toast.LENGTH_SHORT).show();
                text8.setVisibility(View.VISIBLE);
                break;
        }


//          Get Profile
//          adapter.getUserProfileAsync(new ProfileDataListener());

//          Get Contacts
//          adapter.getContactListAsync(new ContactDataListener());

//          Share Status
//          adapter.updateStatus("Hi Test Status Update @" +
Calendar.getInstance().getTimeInMillis(), new MessageListener(), false);

//          Share Image
//          try {
//              Bitmap bitmap = Common.drawableTobitmap(CommonActivity.this,
R.drawable.facebook);
//              adapter.uploadImageAsync("Image Message", "icon.png", bitmap, 0, new
UploadImageListener());
//          } catch (Exception e) {
//              e.printStackTrace();
//          }

      }

      @Override
      public void onError(SocialAuthError error) {
         Log.d("Custom-UI", "Error");
         pDialog.dismiss();
         error.printStackTrace();
      }

      @Override
```

```
        public void onCancel() {
            Log.d("Custom-UI", "Cancelled");
            pDialog.dismiss();
        }

        @Override
        public void onBack() {
            pDialog.dismiss();
            Log.d("Custom-UI", "Dialog Closed by pressing Back Key");

        }
    }

    private final class ContactDataListener implements SocialAuthListener<List<Contact>> {

        @Override
        public void onExecute(String provider, List<Contact> t) {

            Log.d("Custom-UI", "Receiving Data");
//            mDialog.dismiss();
            List<Contact> contactsList = t;

            if (contactsList != null && contactsList.size() > 0) {
                Intent intent = new Intent(CommonActivity.this, ContactActivity.class);
                intent.putExtra("provider", provider);
                intent.putExtra("contact", (Serializable) contactsList);
                startActivity(intent);
            } else {
                Log.d("Custom-UI", "Contact List Empty");
            }
        }

        @Override
        public void onError(SocialAuthError e) {

        }
    }

    // To receive the profile response after authentication
    private final class ProfileDataListener implements SocialAuthListener<Profile> {

        @Override
        public void onExecute(String provider, Profile t) {

            Log.d("Custom-UI", "Receiving Data");
//            mDialog.dismiss();
            Profile profileMap = t;

            Intent intent = new Intent(CommonActivity.this, ProfileActivity.class);
            intent.putExtra("provider", provider);
            intent.putExtra("profile", profileMap);
            startActivity(intent);
        }
```

```
        @Override
        public void onError(SocialAuthError e) {

        }
    }

    // To get status of image upload after authentication
    private final class UploadImageListener implements SocialAuthListener<Integer> {

        @Override
        public void onExecute(String provider, Integer t) {
//            mDialog.dismiss();
            Integer status = t;
            Log.d("Custom-UI", String.valueOf(status));
            if (status.intValue() == 200 || status.intValue() == 201 || status.intValue() == 204)
                Toast.makeText(CommonActivity.this, "Image Uploaded", Toast.LENGTH_SHORT).show();
            else
                Toast.makeText(CommonActivity.this, "Image not Uploaded",
 Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onError(SocialAuthError e) {

        }
    }


}
```

58. Adding ripple effect

You can apply ripple effect to button, textview, etc. Just add following code to your xml file.

```
    <com.andexert.library.RippleView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        ripple:rv_centered="false">

        <Button
            android:id="@+id/text5"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="10dp"
            android:background="#00cc99"
            android:padding="30dp"
            android:text="Ripple Effect 3"
            android:textColor="@android:color/white"
            android:textSize="30sp" />
     </com.andexert.library.RippleView>
```

You can also add below xml attributes.

ripple:rv_alpha [integer def:90 0-255] --> Alpha of the ripple
ripple:rv_framerate [integer def:10] --> Frame rate of the ripple animation
ripple:rv_rippleDuration [integer def:400] --> Duration of the ripple animation
ripple:rv_ripplePadding [dimension def:0] --> Add a padding to the ripple
ripple:rv_color [color def:@android:color/white] --> Color of the ripple
ripple:rv_centered [boolean def:false] --> Center ripple in the child view
ripple:rv_type [enum (simpleRipple, doubleRipple) def:simpleRipple] --> Simple or double ripple
ripple:rv_zoom [boolean def:false] --> Enable zoom animation
ripple:rv_zoomDuration [integer def:150] --> Duration of zoom animation
ripple:rv_zoomScale [float def:1.03] --> Scale of zoom animation

59. Check if website url is valid or not

Method is used to check website url is valid or not.

Parameter:

strUrl (website url as string)

Returns:
True if website address is valid, otherwise **false**.

How to use:

boolean valid = Common.isWebsiteUrlValid (strUrl);

60. Get all contacts that have email address

Method is used to get contacts that have email address from contact book.

Parameter:

mContext(Pass application context).

Returns:

ArrayList of string with name and email as comma separated. i.e. name, name@name.com

How to use:

ArrayList<String> strContacts = Common.getNameEmailDetails(context);

61. Get bitmap of view (ScreenShot)

Used to get bitmap of layout.

Parameter:

View (view you want image of ie. : Layout, Views, etc).

Returns:

Bitmap of view.

How to use:

Bitmap screenShot = Common.captureView(textView);

62. **Pick Color from ImageView**

Used to pick color from image.Pass x, y and view from onTouch of ImageView onTouch Listener.

Parameter:

View (image view from which color is to be picked).

x (X-position of touch)

y (Y-position of touch)

Returns:

Color in integer format.

How to use:

int color = Common.pickColor(view, x, y);

e.g:

```
imageView.setOnTouchListener(onTouchListener);
View.OnTouchListener onTouchListener = new View.OnTouchListener() {
    @Override
    public boolean onTouch(View view, MotionEvent motionEvent) {
      int action = motionEvent.getAction();
      switch (action) {
        case (MotionEvent.ACTION_DOWN):
          int x = (int) motionEvent.getX();
          int y = (int) motionEvent.getY();
          try {
            color = Common.pickColor(view, x, y);
          } catch (NullPointerException e) {
            return false;
          }
      }
```

```
        return false;
    }
};
```

63. Get File Size

Used to get size of file.

Parameter:

mContext (Pass application context)

strUrl (Pass url of file)

Return:

This method does not return anything.

How to use:

getRemoteFileSize(mContext, strUrl);

64. Download File

Used to download any file.

Parameter:

mContext (Pass application context)

strUrl (Url of file as string)

fileName (Filename with extension i.e. image.jpg)

Return:

This method does not return anything.

How to use:

downloadRemoteFile(mContext, strUrl, fileName);

Disclaimer: *We have taken enough care of quality while developing this library. It still might have some issues. So use it at your own risks. You can post the issues you find or suggestions on GitHub or our blog. We will do our best to make it better for you. Happy coding!*