**1.Create Background(TextView, EditText, Button etc) using XML:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"**

>  
  
 *<!-- Background Color -->* <**solid android:color="#ffffff"** />  
  
 *<!-- Border Color -->* <**stroke android:width="1dp" android:color="#ff9900"** />  
  
 *<!-- Round Corners -->* <**corners android:radius="5dp"** />  
  
  
  
</**shape**>

**2.Create Background(Spinner) using XML:**

*<?***xml version="1.0" encoding="UTF-8"***?>*<**shape xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="rectangle"**>  
  
 <**solid android:color="@android:color/white"**/>  
 <**corners android:radius="4dp"**></**corners**>  
 <**stroke android:color="#cccccc" android:width="1dp"**/>  
  
</**shape**>

**3.Using AlertDialog :**

AlertDialog.Builder alertDialog = **new** AlertDialog.Builder(**context**);

alertDialog.setMessage(getString(R.string.***alertName***));  
alertDialog.setPositiveButton(**"Yes"**,  
 **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 dialog.cancel();  
 }  
 });  
alertDialog.show();

**4. Get Radio Button Value:**

String valueGender = ((RadioButton)findViewById(**rgBtnGender**.getCheckedRadioButtonId() )).getText().toString().trim();

**5. Get Spinner Value:**

String spinnerItem = **spType**.getItemAtPosition(**spType**.getSelectedItemPosition()).toString();

**6.Get CheckBox value and isChecked?:**

**if** (**chkPastBtn**.isChecked()){

String valueCheck = **chkPastBtn**.getText().toString().trim();  
  
}**else**{  
  
 Toast.*makeText*(**context**,**"CheckBox not Checked"**,Toast.***LENGTH\_SHORT***).show();  
}

**7.Using SharedPreferences:**

## a. Write to Shared Preferences

/\*\*\*\*\*\*\* Create SharedPreferences \*\*\*\*\*\*\*/

SharedPreferences pref = **context**.getSharedPreferences(**"MyPref"**, ***MODE\_PRIVATE***);  
SharedPreferences.Editor editor = pref.edit();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Storing data as KEY/VALUE pair \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

    editor.putBoolean("key\_name1", true);           // Saving boolean - true/false

    editor.putInt("key\_name2", "int value");        // Saving integer

    editor.putFloat("key\_name3", "float value");    // Saving float

    editor.putLong("key\_name4", "long value");      // Saving long

    editor.putString("key\_name5", "string value");  // Saving string

    // Save the changes in SharedPreferences

    editor.commit(); // commit changes

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Get SharedPreferences data \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// If value for key not exist then return second param value - In this case null

    pref.getBoolean("key\_name1", null);         // getting boolean

    pref.getInt("key\_name2", null);             // getting Integer

    pref.getFloat("key\_name3", null);           // getting Float

    pref.getLong("key\_name4", null);            // getting Long

    pref.getString("key\_name5", null);          // getting String

/\*\*\*\*\*\*\*\*\*\*\*\* Deleting Key value from SharedPreferences \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

    editor.remove("key\_name3"); // will delete key key\_name3

    editor.remove("key\_name4"); // will delete key key\_name4

    // Save the changes in SharedPreferences

    editor.commit(); // commit changes

/\*\*\*\*\*\*\*\*\*\*\*\* Clear all data from SharedPreferences \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

     editor.clear();

     editor.commit(); // commit changes

**b. Read from Shared Preferences**

boolean--

pref.getBoolean(**"forgetValue"**,**true**);

String--

pref.getString("ketString", "");

**8.Check valid Email:**

**protected boolean** isValidEmail(CharSequence target) {  
 **if** (target == **null**) {  
 **return false**;  
 } **else** {  
 **return** android.util.Patterns.***EMAIL\_ADDRESS***.matcher(target).matches();  
 }  
}

**9.Check Internet:**

**protected boolean** isOnline(){  
 ConnectivityManager cm = (ConnectivityManager) getSystemService(**context**.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netInfo = cm.getActiveNetworkInfo();  
 **if** (netInfo != **null** && netInfo.isConnectedOrConnecting()){  
 **return true**;  
 }**else**{  
 **return false**;  
 }  
}

**10. CustomSpinnerAdapter:**

ArrayList<String> type = **new** ArrayList<String>();  
type.add(**"Choose a Type"**);  
type.add(**"Islam"**);  
type.add(**"Science"**);  
type.add(**"Arts"**);  
type.add(**"Social"**);  
type.add(**"Filem"**);  
type.add(**"Health"**);  
type.add(**"Beauty"**);  
type.add(**"Islam11"**);  
type.add(**"Science11"**);  
type.add(**"Arts11"**);  
type.add(**"Social11"**);  
type.add(**"Filem11"**);  
type.add(**"Health11"**);  
type.add(**"Beauty11"**);  
type.add(**"Islam22"**);  
type.add(**"Science22"**);  
type.add(**"Arts22"**);  
type.add(**"Social22"**);  
type.add(**"Filem22"**);  
type.add(**"Health22"**);  
type.add(**"Beauty22"**);  
  
CustomSpinnerAdapter customSpinnerAdapter=**new** CustomSpinnerAdapter(**context**,type);  
**spType**.setAdapter(customSpinnerAdapter);

**public class** CustomSpinnerAdapter **extends** BaseAdapter **implements** SpinnerAdapter {  
  
 **private final** Context **context**;  
 **private** ArrayList<String> **asr**;  
  
 **public** CustomSpinnerAdapter(Context context,ArrayList<String> asr) {  
 **this**.**asr**=asr;  
 **this**.**context** = context;  
 }  
  
 **public int** getCount()  
 {  
 **return asr**.size();  
 }  
  
 **public** Object getItem(**int** i)  
 {  
 **return asr**.get(i);  
 }  
 **public long** getItemId(**int** i)  
 {  
 **return** (**long**)i;  
 }  
  
 @Override  
 **public** View getDropDownView(**int** position, View convertView, ViewGroup parent) {  
 TextView txt = **new** TextView(**context**);  
 txt.setPadding(16, 16, 16, 16);  
 txt.setTextSize(16);  
 txt.setGravity(Gravity.***CENTER\_VERTICAL***);  
 txt.setText(**asr**.get(position));  
 txt.setTextColor(Color.*parseColor*(**"#000000"**));  
 **return** txt;  
 }  
  
 **public** View getView(**int** i, View view, ViewGroup viewgroup) {  
 TextView txt = **new** TextView(**context**);  
 txt.setGravity(Gravity.***CENTER***);  
 txt.setPadding(16, 16, 16, 16);  
 txt.setTextSize(20);  
 txt.setCompoundDrawablesWithIntrinsicBounds(0, 0, android.R.drawable.***arrow\_down\_float***, 0);  
 txt.setText(**asr**.get(i));  
 txt.setTextColor(Color.*parseColor*(**"#000000"**));  
 **return** txt;  
 }  
  
}

# 11. [How to pass an object from one activity to another on Android](http://stackoverflow.com/questions/2736389/how-to-pass-an-object-from-one-activity-to-another-on-android):

a.From Activity:

Person person = **new** Person();  
Intent intent = **new** Intent(**context**,RegActivity.**class**);  
intent.putExtra(**"personData"**,person);  
startActivity(intent);

b.To Activity:

Intent intent = getIntent();  
Person pdata = (Person)intent.getSerializableExtra(**"personData"**);

# Getting a Result from an Activity

## a. Start the Activity

## private static final int REQUEST\_CODE = 900;

## Intent i = new Intent(context, SecondActivity.class);

## startActivityForResult(i, REQUEST\_CODE);

## b. Receive the Result

## @Override

## protected void onActivityResult(int requestCode, int resultCode, Intent data) {

## if(requestCode==REQUEST\_CODE){

## if (resultCode==RESULT\_OK){

## }

## }

## super.onActivityResult(requestCode, resultCode, data);

## }

## 12. In AppConstant Mehtod:

## a. Online Check:

**public static boolean** isOnline(Context context){  
 ConnectivityManager cm = (ConnectivityManager) context.getSystemService(context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netInfo = cm.getActiveNetworkInfo();  
 **if** (netInfo != **null** && netInfo.isConnectedOrConnecting()){  
 **return true**;  
 }**else**{  
 **return false**;  
 }  
}

## b. valid Email Check:

**public static boolean** isValidEmail(CharSequence target){  
 **if** (target == **null**) {  
 **return false**;  
 } **else** {  
 **return** android.util.Patterns.***EMAIL\_ADDRESS***.matcher(target).matches();  
 }  
}

**c. Show Alert Message**

**public static void** showAlertMessage(Context context,String message){  
 AlertDialog.Builder alertDialog = **new** AlertDialog.Builder(context);  
 alertDialog.setMessage(message);  
 alertDialog.setPositiveButton(**"Yes"**,  
 **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 dialog.cancel();  
 }  
 });  
 alertDialog.show();  
}

## Core Java

## constructor:

## create many number of object ways:

## a. By using new keyword

## b. instance factory method

## c. static factory method

## d. pattern factory method

## e. newInstance() method

## f.clone() method

## g. deSerilization process

## Rules to declare constructor in java:

## 1.const. name and class name must be same.

## 2.const. able to take params.

## 3.const. are not allowed return types.

## Type of constructor:

## 1.Default constructor

## 2.User defined constructor

======================================================================================

**CardView**

<**android.support.v7.widget.CardView  
 android:id="@+id/card\_view"  
 xmlns:card\_view="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 card\_view:cardBackgroundColor="@android:color/white"  
 card\_view:cardCornerRadius="10dp"  
 card\_view:cardElevation="2dp"  
 card\_view:cardUseCompatPadding="true"**>

**RecyclerView**

<**android.support.v7.widget.RecyclerView  
 android:id="@+id/stdRecycler"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:scrollbars="vertical"** />

**Setting Adapter for RecyclerView**

private RecyclerView.LayoutManager layoutManager;

reCycleView.setHasFixedSize(true);

reCycleView.setItemAnimator(new DefaultItemAnimator());

layoutManager = new LinearLayoutManager(context);

reCycleView.setLayoutManager(layoutManager);

**Read Json file form Assets folder**

**public static** String loadJSONFromAsset(Context context,String fileName) {  
  
 String json = **null**;  
 **try** {  
 InputStream is = context.getAssets().open(fileName);  
 **int** size = is.available();  
 **byte**[] buffer = **new byte**[size];  
 is.read(buffer);  
 is.close();  
 json = **new** String(buffer, **"UTF-8"**);  
 } **catch** (IOException ex) {  
 ex.printStackTrace();  
 **return null**;  
 }  
 **return** json;  
  
}

**Parse Using Gson**

GsonBuilder builder = **new** GsonBuilder();  
 Gson g = builder.setPrettyPrinting().create();AllDetailObject allDetailObject = g.fromJson(**new** String(response),

AllDetailObject.**class**);

**Parse without ArryList name**

Type personType = **new** TypeToken<ArrayList<Person>>() {}.getType();  
List<Person> array = **new** Gson().fromJson(response, personType);

**Using Volley**

**private void** loadPersonArray(){  
 showProgressDialog();  
  
 StringRequest strRequest =**new** StringRequest(Request.Method.***GET***, ***DATA\_URL***, **new** Response.Listener<String>() {  
 @Override  
 **public void** onResponse(String response) {  
 hideProgressDialog();  
  
 Type personType = **new** TypeToken<ArrayList<Person>>() {}.getType();  
 List<Person> array = **new** Gson().fromJson(response, personType);  
  
 Log.*e*(**"Size of List"**,**""**+ array.size());  
  
  
 **showStringArray**.setText(response.toString());  
  
  
  
 }  
 }, **new** Response.ErrorListener() {  
 @Override  
 **public void** onErrorResponse(VolleyError error) {  
 Log.*e*(**"Error Message: "**,**""**+error.getMessage());  
 hideProgressDialog();  
  
 }  
 })  
  
 {  
  
 @Override  
 **public** Map<String, String> getHeaders() **throws** AuthFailureError {  
 HashMap<String, String> headers = **new** HashMap<String, String>();  
 headers.put(**"Content-Type"**, **"application/json"**);  
 **return** headers;  
  
 }  
  
 @Override  
 **protected** Map<String, String> getParams() **throws** AuthFailureError {  
 Map<String, String> params = **new** HashMap<String, String>();  
 params.put(**"name"**, **"Androidhive"**);  
 params.put(**"email"**, **"abc@androidhive.info"**);  
 params.put(**"pass"**, **"password123"**);  
 **return** params;  
 }  
 }  
 ;  
  
 RequestQueue rQue = Volley.*newRequestQueue*(**context**);  
 rQue.add(strRequest);  
 }

**API CAll**

Base URL: http://services.hanselandpetal.com/photos/

**1.** **http://services.hanselandpetal.com/feeds/flowers.json**

**2.** **http://services.hanselandpetal.com/secure/flowers.json**

**parameters:**

**userName: feeduser**

**password: feedpassword**