Module: Mobile Application development (Android)

Session 40: Data storage using SQLite assignment (practice)

#### In this practice session you will create Login Registration Screen with SQLite Database Example

Here you will learn complete details about Android Login Registration Screen with SQLite Database Example. Also discuss about how to create or design Android Login (Sign In) and New Registration (Sign Up) screen. you will learn how to save application’s data in it’s own Android SQlite Database.

To make the example very simple. In this example we will create one home screen activity, in which we will use 2 buttons (one is used for ‘Sign In’ option and other is for ‘Sign Up’ option). Then we will show Login and ‘New Registration’ screen on the **onClick**  option of the respective buttons. All required Login and Registration user-data will be save in application’s own Android SQLite database. The database will be stored in the application’s context, so that no other android applications will access the data.

### Steps Required to Create Android Login Registration Application

1. Create a Home Screen Activity , Which will hold ‘Sign In‘ and ‘Sign Up‘ options.

2. Create layouts for home screen and ‘Sign In‘ and ‘Sign Up‘ Screens.

4. Create a SQLite  Database in the application’s context, so that we can save all required user data (‘Sign In‘ and ‘Sign Up‘ details).

5. Code Logic in Application’s Java files.

6. Run Android Login Registration app on Device/Emulator.

### Android Layouts for Login Registration Application

**1. Default layout (main.xml)**

This is used for home screen in this application, which holds 2 buttons for Sign In and Sign Upoptions.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"      xmlns:tools="http://schemas.android.com/tools"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical"      android:gravity="center\_vertical" >        <Button          android:id="@+id/buttonSignIN"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"           android:text="Sign In"           android:onClick="signIn"/>        <Button          android:id="@+id/buttonSignUP"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Sign Up" />    </LinearLayout> |

**2. login.xml**

This xml is used for Log In (Sign In) screen in this application.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical" >        <EditText          android:id="@+id/editTextUserNameToLogin"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:hint="User Name"          android:ems="10" >          <requestFocus />      </EditText>        <EditText          android:id="@+id/editTextPasswordToLogin"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:ems="10"          android:inputType="textPassword"          android:hint="Password" />        <Button          android:id="@+id/buttonSignIn"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Sign In" />    </LinearLayout> |

**3. signup.xml**

This xml is used for first time user, so for new registration we will use this xml file.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39 | <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical"      android:gravity="center\_vertical" >        <EditText          android:id="@+id/editTextUserName"          android:hint="User Name"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          >            <requestFocus />      </EditText>        <EditText          android:id="@+id/editTextPassword"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"           android:hint="Password"          android:inputType="textPassword" />        <EditText          android:id="@+id/editTextConfirmPassword"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:hint="Confirm Password"          android:inputType="textPassword" />        <Button          android:id="@+id/buttonCreateAccount"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Create Account"          android:layout\_marginBottom="60dp" />    </LinearLayout> |

**Now Coming to Java Codes**

We have used 4 Java files in this application. You can also find complete source code description inside the code, once you download complete code from here.

**1. HomeActivity.java**

In this file we have implemented logic for onClickbutton function, which is used for ‘Sign In’ and ‘Sign Up’ options. Also we will just create a reference to the instance of SQLite Database for storing and querying data in the database.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88 | package com.techblogon.loginexample;    import android.app.Activity;  import android.app.Dialog;  import android.content.Intent;  import android.os.Bundle;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;  import android.widget.Toast;    public class HomeActivity extends Activity  {  Button btnSignIn,btnSignUp;  LoginDataBaseAdapter loginDataBaseAdapter;    @Override  protected void onCreate(Bundle savedInstanceState)  {       super.onCreate(savedInstanceState);       setContentView(R.layout.main);         // create a instance of SQLite Database       loginDataBaseAdapter=new LoginDataBaseAdapter(this);       loginDataBaseAdapter=loginDataBaseAdapter.open();         // Get The Refference Of Buttons       btnSignIn=(Button)findViewById(R.id.buttonSignIN);       btnSignUp=(Button)findViewById(R.id.buttonSignUP);        // Set OnClick Listener on SignUp button      btnSignUp.setOnClickListener(new View.OnClickListener() {  public void onClick(View v) {  // TODO Auto-generated method stub    /// Create Intent for SignUpActivity  abd Start The Activity  Intent intentSignUP=new Intent(getApplicationContext(),SignUPActivity.class);  startActivity(intentSignUP);  }  });  }  // Methos to handleClick Event of Sign In Button  public void signIn(View V)     {  final Dialog dialog = new Dialog(HomeActivity.this);  dialog.setContentView(R.layout.login);      dialog.setTitle("Login");        // get the Refferences of views      final  EditText editTextUserName=(EditText)dialog.findViewById(R.id.editTextUserNameToLogin);      final  EditText editTextPassword=(EditText)dialog.findViewById(R.id.editTextPasswordToLogin);    Button btnSignIn=(Button)dialog.findViewById(R.id.buttonSignIn);    // Set On ClickListener  btnSignIn.setOnClickListener(new View.OnClickListener() {    public void onClick(View v) {  // get The User name and Password  String userName=editTextUserName.getText().toString();  String password=editTextPassword.getText().toString();    // fetch the Password form database for respective user name  String storedPassword=loginDataBaseAdapter.getSinlgeEntry(userName);    // check if the Stored password matches with  Password entered by user  if(password.equals(storedPassword))  {  Toast.makeText(HomeActivity.this, "Congrats: Login Successfull", Toast.LENGTH\_LONG).show();  dialog.dismiss();  }  else  {  Toast.makeText(HomeActivity.this, "User Name or Password does not match", Toast.LENGTH\_LONG).show();  }  }  });    dialog.show();  }    @Override  protected void onDestroy() {  super.onDestroy();      // Close The Database  loginDataBaseAdapter.close();  }  } |

2. **SignUPActivity.Java**

Here we will have complete logic, after we will click ‘Sign Up’ option from main screen.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69 | package com.techblogon.loginexample;    import android.app.Activity;  import android.os.Bundle;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;  import android.widget.Toast;    public class SignUPActivity extends Activity  {  EditText editTextUserName,editTextPassword,editTextConfirmPassword;  Button btnCreateAccount;    LoginDataBaseAdapter loginDataBaseAdapter;  @Override  protected void onCreate(Bundle savedInstanceState)  {  super.onCreate(savedInstanceState);  setContentView(R.layout.signup);    // get Instance  of Database Adapter  loginDataBaseAdapter=new LoginDataBaseAdapter(this);  loginDataBaseAdapter=loginDataBaseAdapter.open();    // Get Refferences of Views  editTextUserName=(EditText)findViewById(R.id.editTextUserName);  editTextPassword=(EditText)findViewById(R.id.editTextPassword);  editTextConfirmPassword=(EditText)findViewById(R.id.editTextConfirmPassword);    btnCreateAccount=(Button)findViewById(R.id.buttonCreateAccount);  btnCreateAccount.setOnClickListener(new View.OnClickListener() {    public void onClick(View v) {  // TODO Auto-generated method stub    String userName=editTextUserName.getText().toString();  String password=editTextPassword.getText().toString();  String confirmPassword=editTextConfirmPassword.getText().toString();    // check if any of the fields are vaccant  if(userName.equals("")||password.equals("")||confirmPassword.equals(""))  {  Toast.makeText(getApplicationContext(), "Field Vaccant", Toast.LENGTH\_LONG).show();  return;  }  // check if both password matches  if(!password.equals(confirmPassword))  {  Toast.makeText(getApplicationContext(), "Password does not match", Toast.LENGTH\_LONG).show();  return;  }  else  {      // Save the Data in Database      loginDataBaseAdapter.insertEntry(userName, password);      Toast.makeText(getApplicationContext(), "Account Successfully Created ", Toast.LENGTH\_LONG).show();  }  }  });  }  @Override  protected void onDestroy() {  // TODO Auto-generated method stub  super.onDestroy();    loginDataBaseAdapter.close();  }  } |

**3. DataBaseHelper.Java**

This file will be use to create a new DB, when no database exists in disk and the helper class will do the needful for us. Also this class will be help us to upgrade the version of the DB if required.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | package com.techblogon.loginexample;    import android.content.Context;  import android.database.sqlite.SQLiteDatabase;  import android.database.sqlite.SQLiteDatabase.CursorFactory;  import android.database.sqlite.SQLiteOpenHelper;  import android.util.Log;    public class DataBaseHelper extends SQLiteOpenHelper  {  public DataBaseHelper(Context context, String name,CursorFactory factory, int version)      {             super(context, name, factory, version);  }  // Called when no database exists in disk and the helper class needs  // to create a new one.  @Override  public void onCreate(SQLiteDatabase \_db)  {  \_db.execSQL(LoginDataBaseAdapter.DATABASE\_CREATE);    }  // Called when there is a database version mismatch meaning that the version  // of the database on disk needs to be upgraded to the current version.  @Override  public void onUpgrade(SQLiteDatabase \_db, int \_oldVersion, int \_newVersion)  {  // Log the version upgrade.  Log.w("TaskDBAdapter", "Upgrading from version " +\_oldVersion + " to " +\_newVersion + ", which will destroy all old data");    // Upgrade the existing database to conform to the new version. Multiple  // previous versions can be handled by comparing \_oldVersion and \_newVersion  // values.  // The simplest case is to drop the old table and create a new one.  \_db.execSQL("DROP TABLE IF EXISTS " + "TEMPLATE");  // Create a new one.  onCreate(\_db);  }    } |

**4. LoginDataBaseAdapter.Java**

This file is required to handle all Database operations like (create DB, Insert record, update record, Delete record, Close DB, and Cursor related stuffs.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87 | package com.techblogon.loginexample;    import android.content.ContentValues;  import android.content.Context;  import android.database.Cursor;  import android.database.SQLException;  import android.database.sqlite.SQLiteDatabase;    public class LoginDataBaseAdapter  {  static final String DATABASE\_NAME = "login.db";  static final int DATABASE\_VERSION = 1;  public static final int NAME\_COLUMN = 1;  // TODO: Create public field for each column in your table.  // SQL Statement to create a new database.  static final String DATABASE\_CREATE = "create table "+"LOGIN"+                               "( " +"ID"+" integer primary key autoincrement,"+ "USERNAME  text,PASSWORD text); ";  // Variable to hold the database instance  public  SQLiteDatabase db;  // Context of the application using the database.  private final Context context;  // Database open/upgrade helper  private DataBaseHelper dbHelper;  public  LoginDataBaseAdapter(Context \_context)  {  context = \_context;  dbHelper = new DataBaseHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION);  }  public  LoginDataBaseAdapter open() throws SQLException  {  db = dbHelper.getWritableDatabase();  return this;  }  public void close()  {  db.close();  }    public  SQLiteDatabase getDatabaseInstance()  {  return db;  }    public void insertEntry(String userName,String password)  {         ContentValues newValues = new ContentValues();  // Assign values for each row.  newValues.put("USERNAME", userName);  newValues.put("PASSWORD",password);    // Insert the row into your table  db.insert("LOGIN", null, newValues);  ///Toast.makeText(context, "Reminder Is Successfully Saved", Toast.LENGTH\_LONG).show();  }  public int deleteEntry(String UserName)  {  //String id=String.valueOf(ID);      String where="USERNAME=?";      int numberOFEntriesDeleted= db.delete("LOGIN", where, new String[]{UserName}) ;         // Toast.makeText(context, "Number fo Entry Deleted Successfully : "+numberOFEntriesDeleted, Toast.LENGTH\_LONG).show();          return numberOFEntriesDeleted;  }  public String getSinlgeEntry(String userName)  {  Cursor cursor=db.query("LOGIN", null, " USERNAME=?", new String[]{userName}, null, null, null);          if(cursor.getCount()<1) // UserName Not Exist          {           cursor.close();           return "NOT EXIST";          }      cursor.moveToFirst();  String password= cursor.getString(cursor.getColumnIndex("PASSWORD"));  cursor.close();  return password;  }  public void  updateEntry(String userName,String password)  {  // Define the updated row content.  ContentValues updatedValues = new ContentValues();  // Assign values for each row.  updatedValues.put("USERNAME", userName);  updatedValues.put("PASSWORD",password);            String where="USERNAME = ?";      db.update("LOGIN",updatedValues, where, new String[]{userName});  }  } |

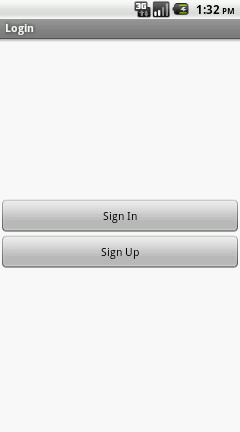
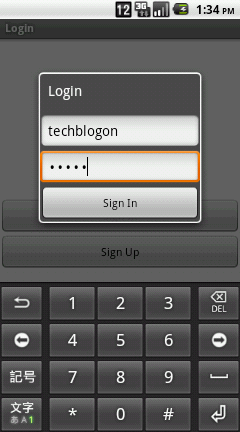
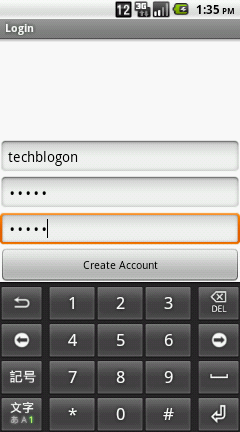
Note: for Log In Dialog Screen, we just used the login.xml with the below code snippet. You can find below code snippet in HomeActivity.java file.

|  |  |
| --- | --- |
| 1  2  3 | final Dialog dialog = new Dialog(HomeActivity.this);  dialog.setContentView(R.layout.login);  dialog.setTitle("Login"); |

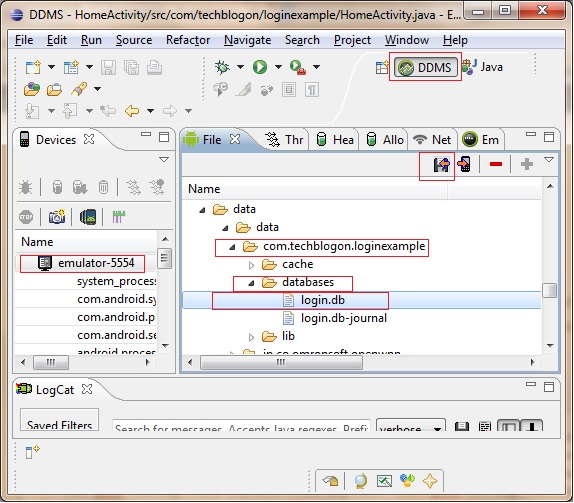
**Finally Applications Manifest File**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"      package="com.techblogon.loginexample"      android:versionCode="1"      android:versionName="1.0" >        <uses-sdk          android:minSdkVersion="8"          android:targetSdkVersion="16" />        <application          android:allowBackup="true"          android:icon="@drawable/ic\_launcher"          android:label="@string/app\_name"          android:theme="@style/AppTheme" >          <activity              android:name="com.techblogon.loginexample.HomeActivity"              android:label="@string/app\_name" >              <intent-filter>                  <action android:name="android.intent.action.MAIN" />                    <category android:name="android.intent.category.LAUNCHER" />              </intent-filter>          </activity>             <activity              android:name=".SignUPActivity"/>      </application>    </manifest> |

#### How to Build and Run the Sample on Android Device

[](http://techblogon.com/wp-content/uploads/2013/03/login-regisrtation-screen-example.png)[](http://techblogon.com/wp-content/uploads/2013/03/login-screen-android.png)[](http://techblogon.com/wp-content/uploads/2013/03/registration-screen-android.png)

After Run the application and save any user data, you can see login.db file that has been generated and saved for your application.

[](http://techblogon.com/wp-content/uploads/2013/03/android-stored-database.jpg)