Data Storage

**Background:**

Android provides several options for developers to store data locally using different formats such as files, SharedPreferences and SQLite. Data stored using any of these formats will be stored in the same path where the application is stored on the phone which is:

“data/data/package\_name”

If we explore this path we can see that there are three folders which are used to store local data in Android, these folders are:

* Files
* Databases
* SharedPreferences

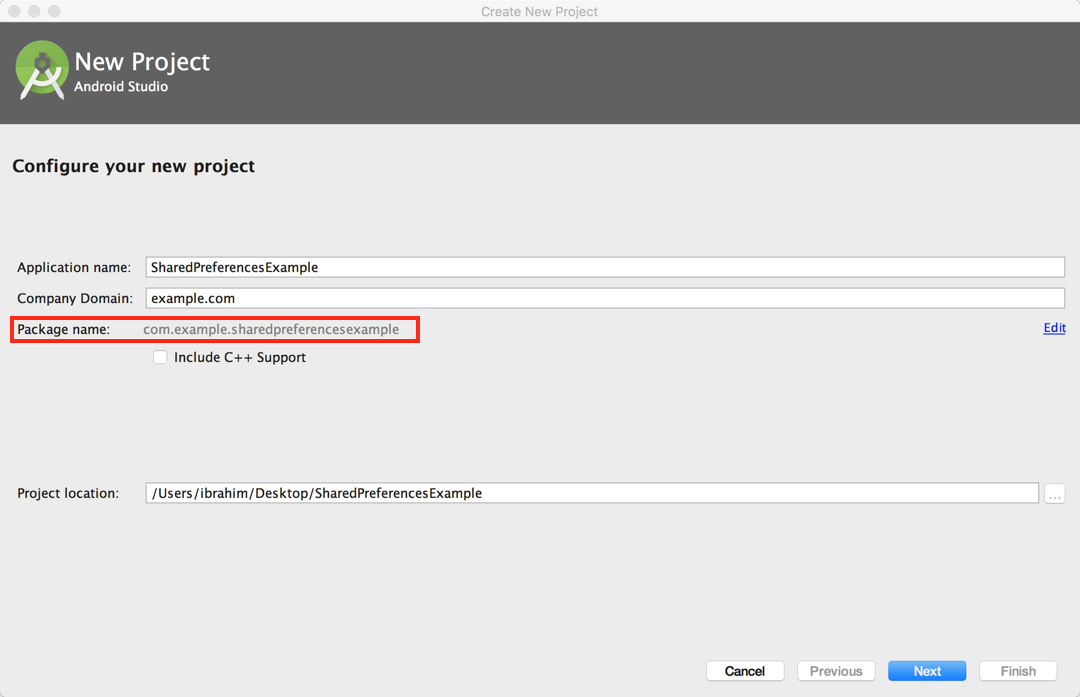
The important thing to know is that Android does not encrypt this data, therefore anyone or any app can go to that path and read the stored data.

Therefore, we advise developers to not use these types of data storage when storing secure information such as username and password since this information can be readable by anyone.

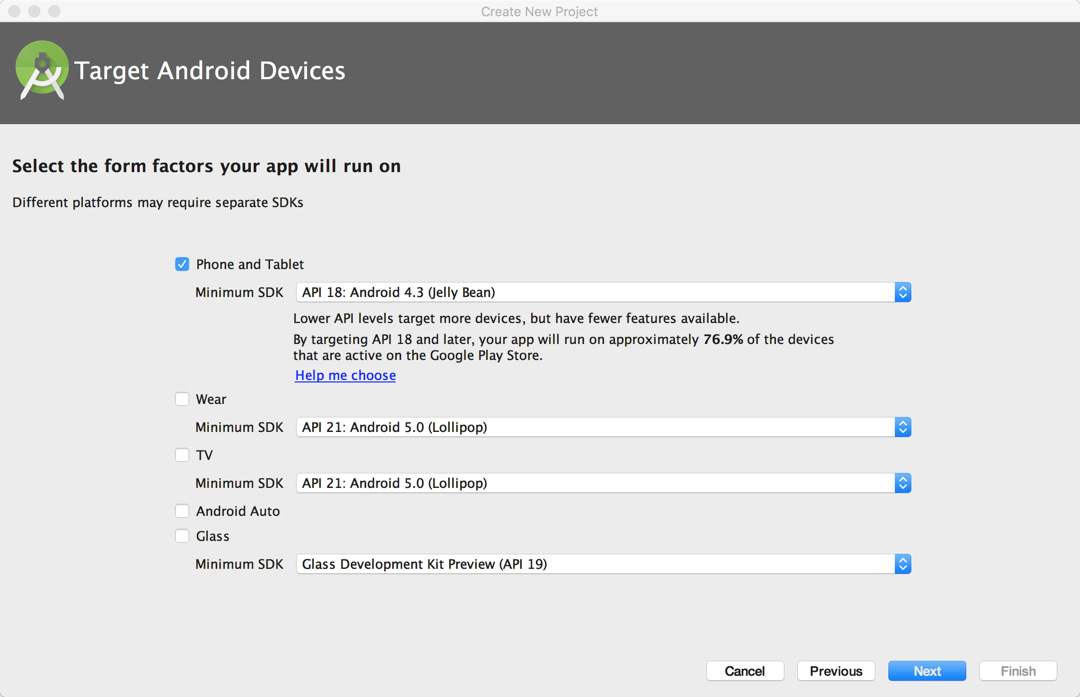
In this tutorial, we will build a simple Android app that allows users to store their usernames and passwords using SharedPreferences data storage, after that we will demonstrate how this data can be accessed and read.

**Steps to Build the App**

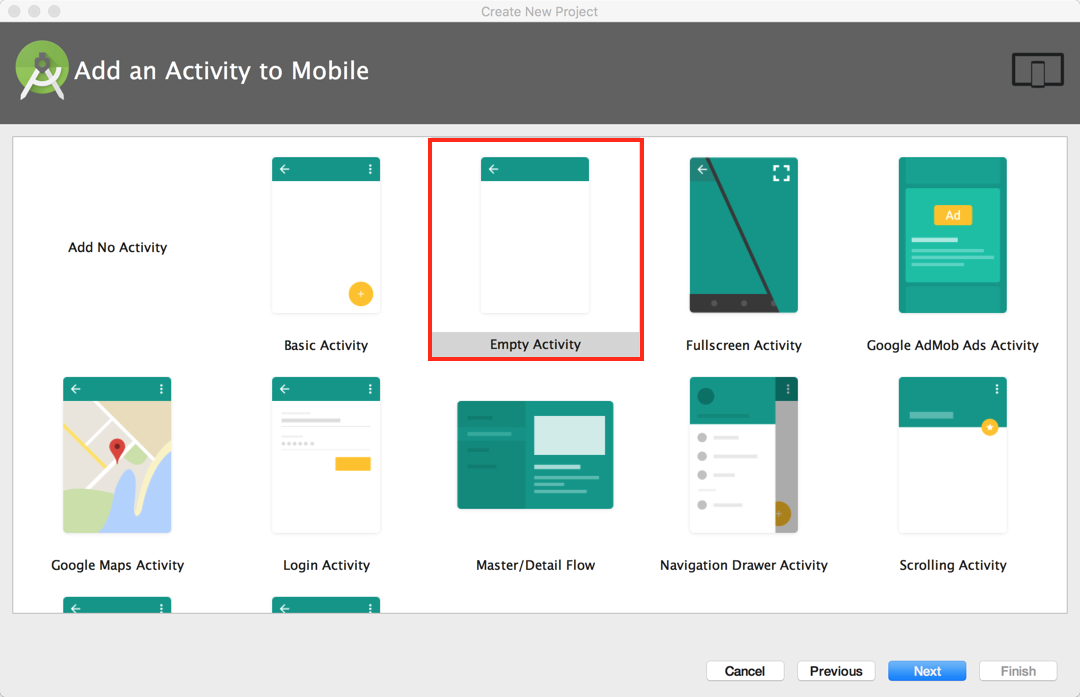
1. Open Android Studio and create new Android project and name it “SharedPreferencesExample”. Make sure to save the package name somewhere on your computer since you will need it in a later stage of this tutorial. Click next.



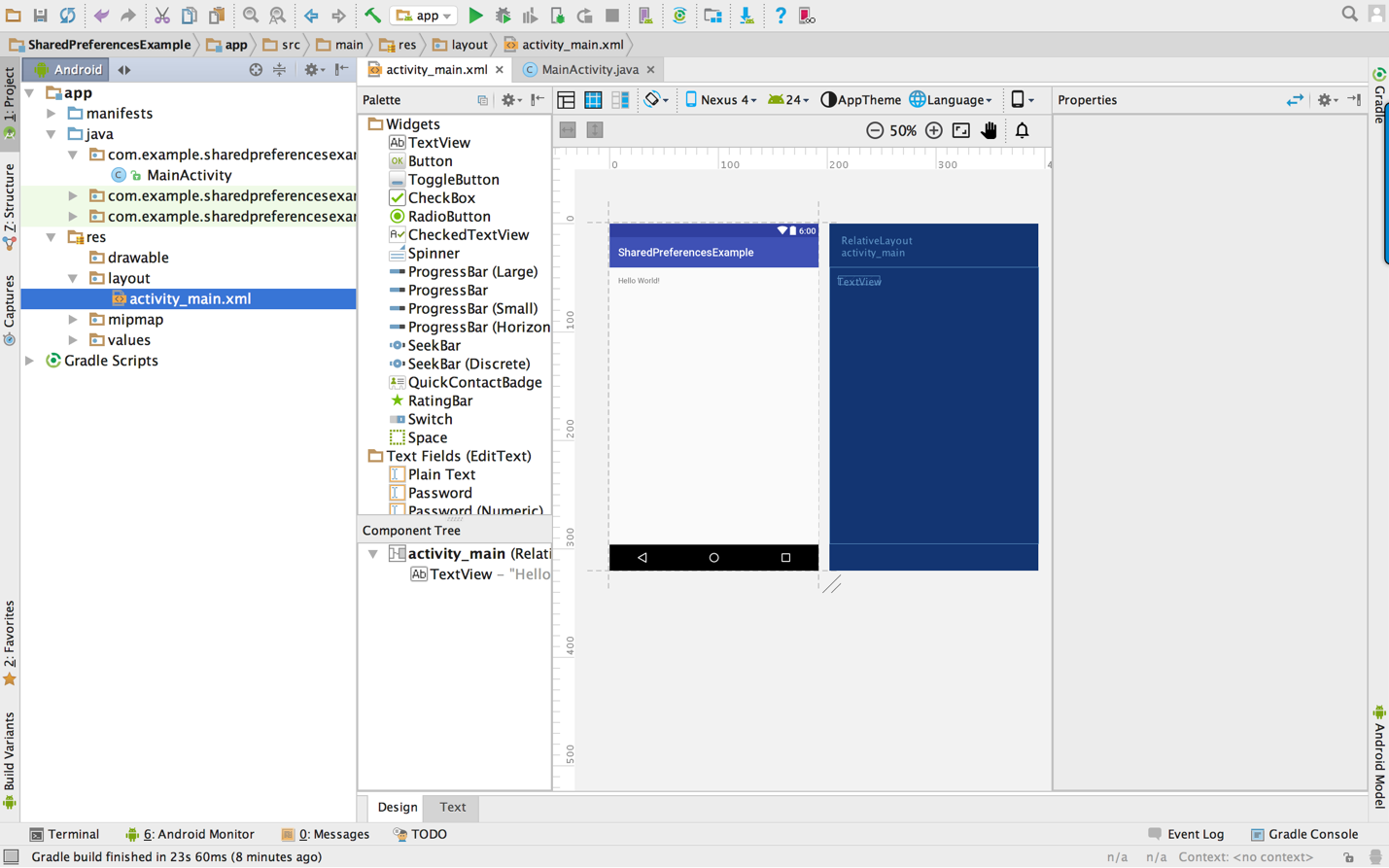
1. Set the minimum SDK to 14, or whatever your default is, and then click on next.



1. Select Empty Activity, click on next and then finish to start the project



1. Open the activity\_main.xml file, which can be found under app/res/layout/. You may need to wait for few seconds for the build and file indexing to complete.



1. Delete the “Hello World” TextView by selecting the TextView and then press the delete button on your keyboard.
2. Add two Text Fields (EditTexts) and one Button to the layout by simply dragging them from the palette block on the left side to the app layout. Note, you can see the names of all elements in the layout on the down left side under the “Component Tree” section.

One EditText object should be a Password one, and the other one should be a Plain Text one.

Once you have the two EditTexts and the Button on the device layout, click on the first EditText and change its properties from right side panel to the following:

* ID: usernameEditText
* Hint: Username
* Delete the value in “text” under TextView
* Keep other properties as they are.

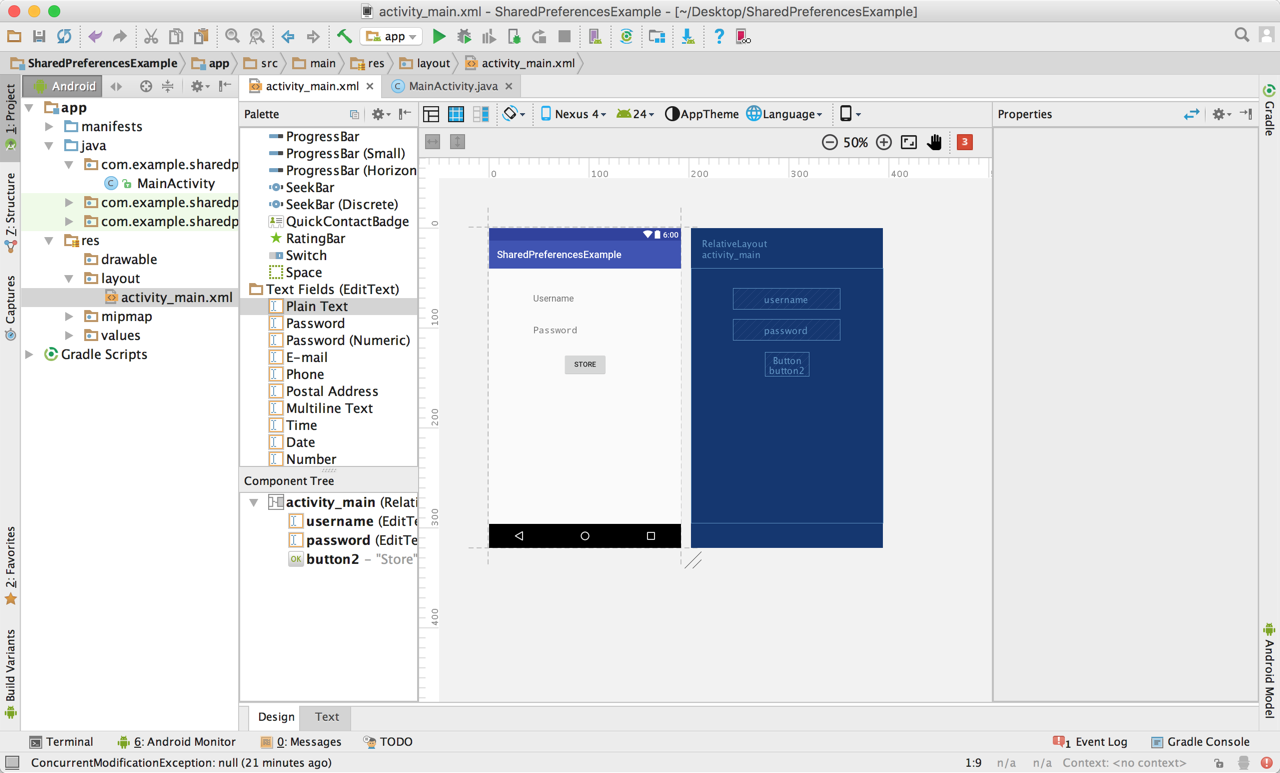
Change the following properties for the second EditText from the properties panel on right side to the following:

* ID: passwordEditText
* Hint: Password

Change the following properties for the Button from the properties panel on right side to the following:

* ID: storeButton
* Text: Store

Once you have completed step 6, your app layout should look like the following image.



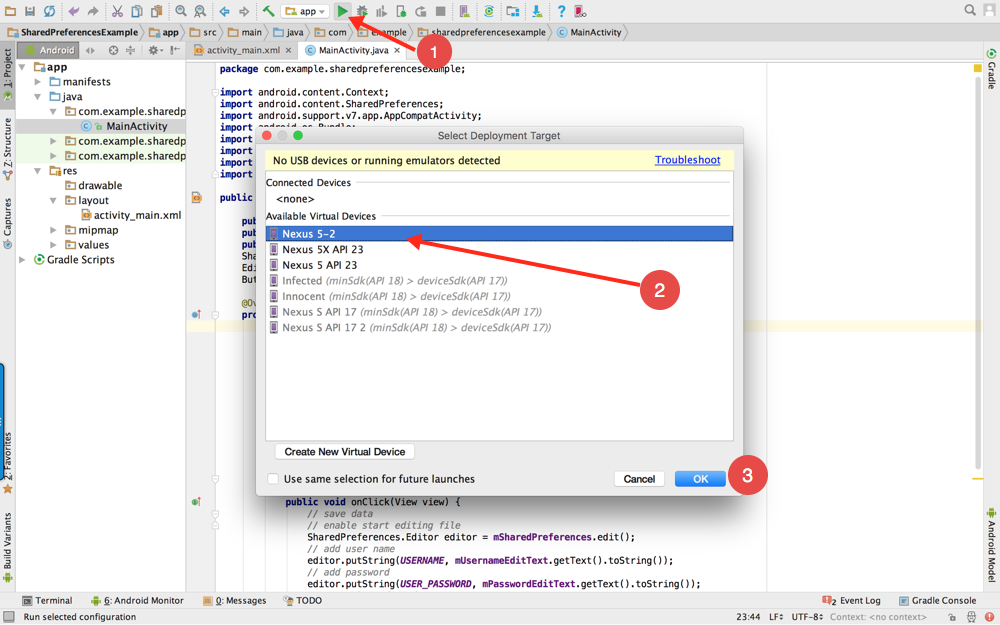
1. Copy and paste the code below to the MainActivity.java file which can be found under “app/java/your\_package\_name /”. Make sure that your package name is written correctly at the top of the file.

**package** com.your\_package\_name;  
  
**import** android.content.Context;  
**import** android.content.SharedPreferences;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 **public static final** String ***MY\_PREFERENCES*** = **"MyPrefLogin"** ;  
 **public static final** String ***USERNAME*** = **"UsernameKey"** ;  
 **public static final** String ***USER\_PASSWORD*** = **"PasswordKey"** ;  
 SharedPreferences **mSharedPreferences**;  
 EditText **mUsernameEditText**, **mPasswordEditText**;  
 Button **mStoreButton**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// initialize username EditText* **mUsernameEditText** = (EditText)findViewById(R.id.***usernameEditText***);  
  
 *// initialize password EditText instance* **mPasswordEditText** = (EditText)findViewById(R.id.***passwordEditText***);

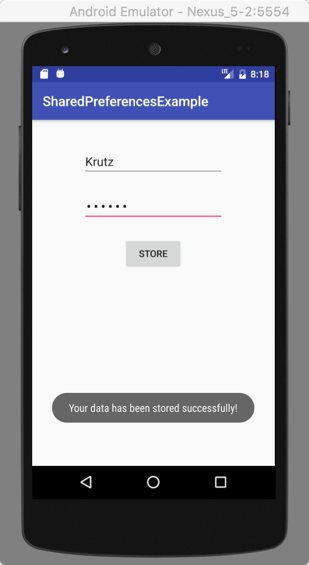
*// initialize button instance*  
 **mStoreButton** = (Button) findViewById(R.id.***storeButton***);  
  
 *// initialize SharedPreferences* **mSharedPreferences** = getSharedPreferences(***MY\_PREFERENCES***, Context.***MODE\_PRIVATE***);

*// Click listener for the button*   
 **mStoreButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 *// store data  
 // enable start editing file* SharedPreferences.Editor editor = **mSharedPreferences**.edit();  
 *// add username* editor.putString(***USERNAME***, **mUsernameEditText**.getText().toString());  
 *// add password* editor.putString(***USER\_PASSWORD***, **mPasswordEditText**.getText().toString());  
 *// store the update data* editor.commit();  
 *//display message stored* Toast.*makeText*(MainActivity.**this**, **"Your data has been stored successfully!"**, Toast.***LENGTH\_LONG***).show();  
 }  
 });  
 }  
}

1. Run the app by clicking on the run button at the top and then select your virtual device and click on OK to run the app. It will take a little while for the virtual device to start and for the app to run. The following image shows how you can start the virtual device and run the app.



1. Once the app has run, enter a username and a password and then click on the store button as shown in the following image.

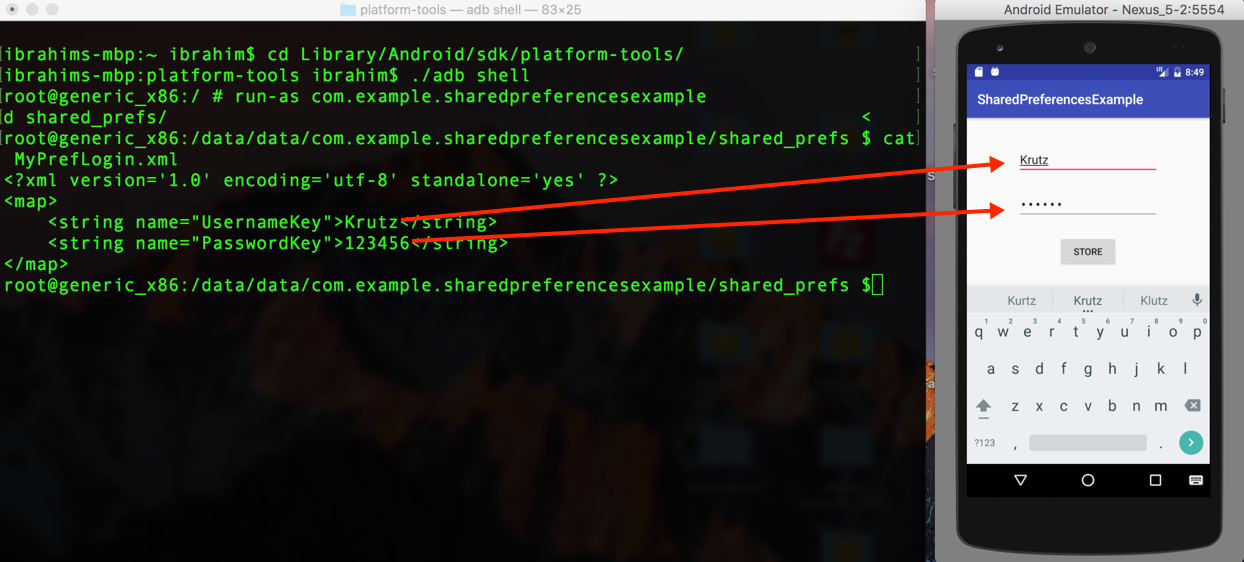


**Steps to Read the App Data**

To read the data which you just stored into the app, you first need to make sure that you still have the virtual device running. After that, open Terminal or Command Prompt and then execute the following commands:

1. cd Library/Android/sdk/platform-tools/
2. ./adb shell
3. run-as your\_package\_name
4. cd shared\_prefs
5. cat MyPrfLogin.xml

Once you have executed the commends above you will be able to read the data which you stored using the app as shown the following image.



**Conclusion:**

As shown in the image above, the data which you entered and stored in the app can be easily accessed and read. Therefore, using such a data storage type to store sensitive users’ information such as passwords or bank information can threaten the users.

**Fixing the Problem:**

There are two solutions which can help in encrypting the stored data and fixing the problem:

1. Encrypting the sensitive data in the phone using encryption and decryption technique.
2. Store the sensitive data on a cloud server and retrieve it when its needed.

**Steps to Fix the Problem**

We are going to use the same app which we just built and then add an encryption and decryption technique to it known as Cipher which will help in encrypting the date before we store using SharedPreferences.

To start encrypting the data using Cipher you first need to add the cipher method to your source code and then you can call the Cipher method to encrypt the entered data before store it using SharedPreferences.

your final source code should look like the following (the additional code and code modified is highlighted in green):

Make sure to replace “your\_package\_name” in the first line with your package name

**package** your\_package\_name;  
  
**import** android.content.Context;  
**import** android.content.SharedPreferences;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 **public static final** String ***MY\_PREFERENCES*** = **"MyPrefLogin"** ;  
 **public static final** String ***USERNAME*** = **"UsernameKey"** ;  
 **public static final** String ***USER\_PASSWORD*** = **"PasswordKey"** ;  
 SharedPreferences **mSharedPreferences**;  
 EditText **mUsernameEditText**, **mPasswordEditText**;  
 Button **mStoreButton**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// initialize the username EditText instance* **mUsernameEditText** = (EditText)findViewById(R.id.***usernameEditText***);  
  
 *// initialize the password EditText instance* **mPasswordEditText** = (EditText)findViewById(R.id.***passwordEditText***);  
  
 *// initialize the Button instance* **mStoreButton** = (Button) findViewById(R.id.***storeButton***);  
  
 *// initialize SharedPreferences* **mSharedPreferences** = getSharedPreferences(***MY\_PREFERENCES***, Context.***MODE\_PRIVATE***);  
  
 *// setting on click listener for the button* **mStoreButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 *// Store data  
 // enable start editing file* SharedPreferences.Editor editor = **mSharedPreferences**.edit();  
 *// add user name* editor.putString(***USERNAME***, cipher(**mUsernameEditText**.getText().toString(),10));  
 *// add password* editor.putString(***USER\_PASSWORD***,cipher( **mPasswordEditText**.getText().toString(),10));  
 *// store the update data* editor.commit();  
 *//display message to inform the user that the stored* Toast.*makeText*(MainActivity.**this**, **"Your data has stored successfully!"**, Toast.***LENGTH\_LONG***).show();  
 }  
 });  
 }  
  
 *// Cipher Method. Cipher encryption add shift for key* String cipher(String msg, **int** shift){  
 String s = **""**;  
 **int** len = msg.length(); *// get string length* **for**(**int** x = 0; x < len; x++){  
 **char** c = (**char**)(msg.charAt(x) + shift); *// shift every character* s += c; *// append the characters* }  
 **return** s;  
 }  
}

Once you have updated the code run the application again by click on the run button.

Basically, this encryption technique considered to be one of the basic encryption techniques, it works by adding shift for every character when we want to encrypt the data, and when we want to decrypt the data we subtract that shift to the origin character.

**Steps to Read the App Data after the encryption**

To read the app data again and make sure that they encrypt it, we first need to open Terminal or Command Prompt and execute the following commands:

1. cd Library/Android/sdk/platform-tools/
2. ./adb shell
3. run-as your\_package\_name
4. cd shared\_prefs
5. cat MyPrfLogin.xml

This time you will notice that the data has encrypted and it’s no longer readable as shown in the image below.

