**Data Storage**

Description:

Android allows developer to store local data in the phone using different format (files, Shared references, SQLite). This data will store in same path that application will store in the phone. it is:

**“data/data/package\_name”**

if we explore this path we will see three folders that have all local data storage

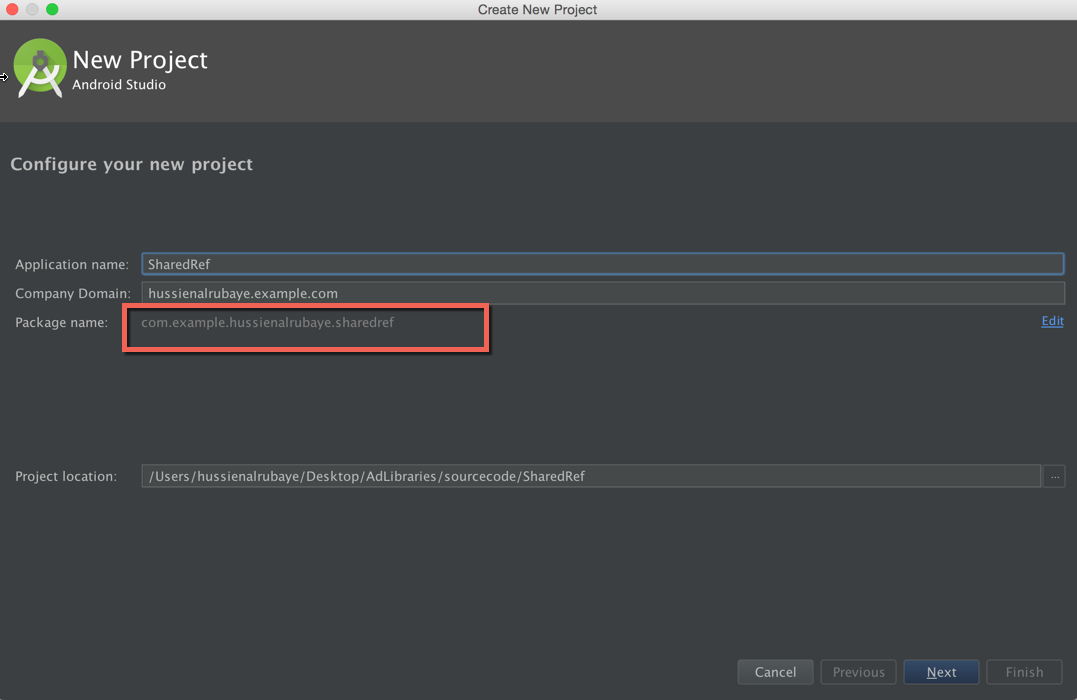
* Files
* Databases
* Shared Ref

The important thing to know that Android did not encrypt this data so anyone or any app could go to that path and read data stored in your (files, Shared references, SQLite.

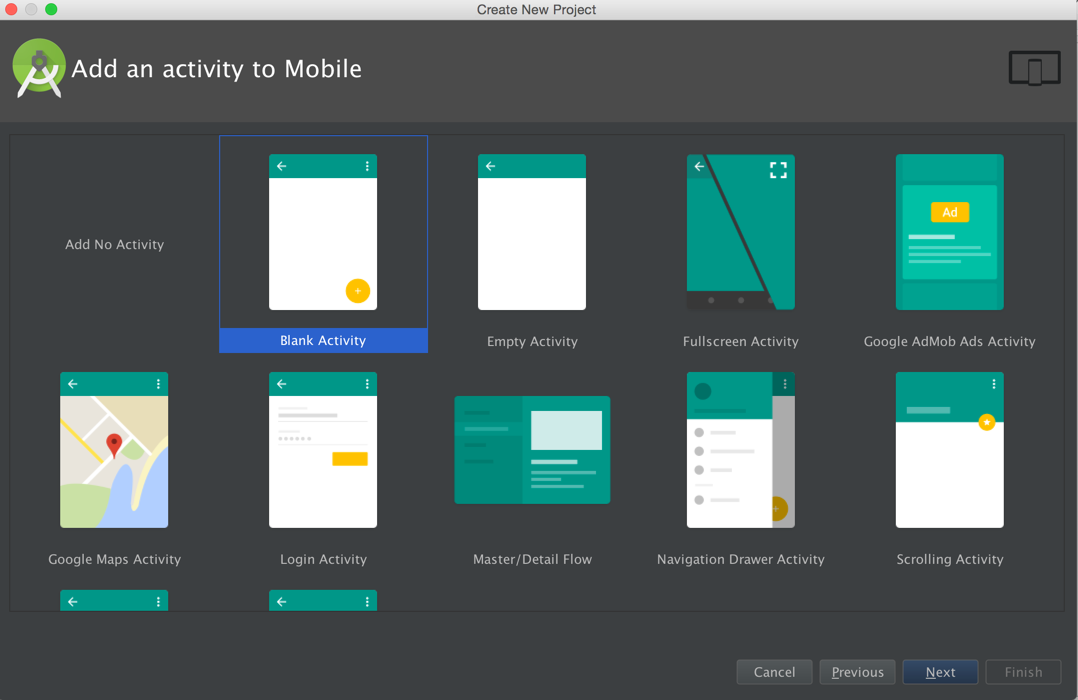
So we advise that don’t use these files to store secure information like (user name and password). We will build simple app that store user name and password in Shared references then we will show how this data could be access and read

**Steps to build the app**

1. Open new project with name “shared Ref”, save the package name will will need next



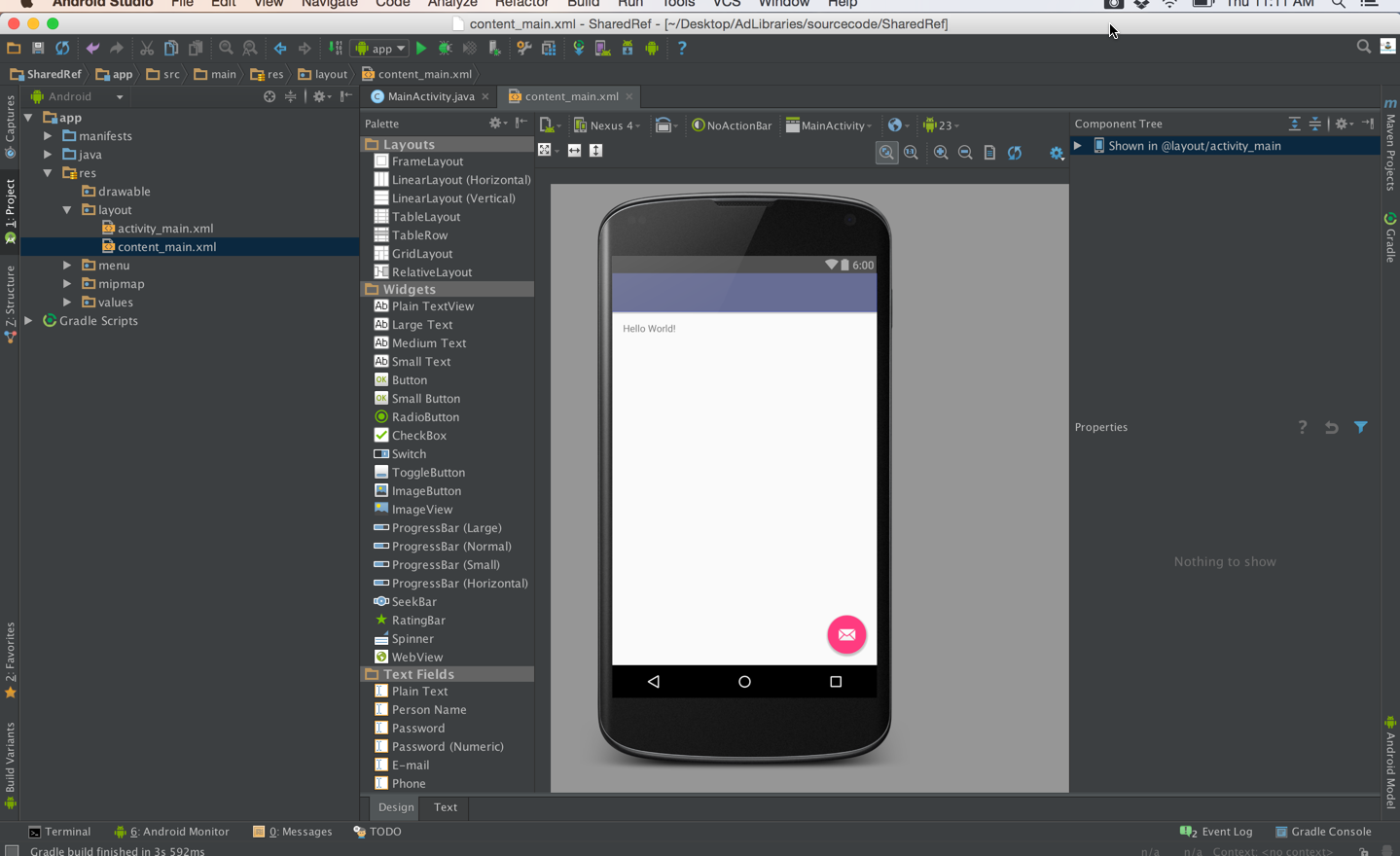
1. Select blank activity (in the most recent version, renamed to basic activity)



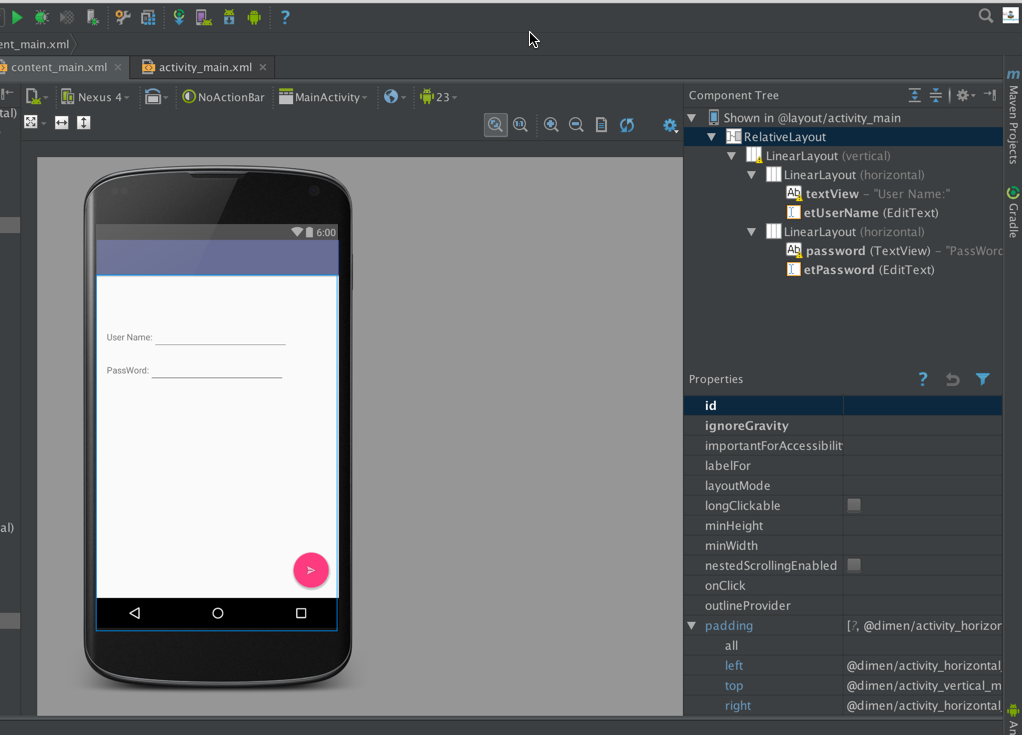
Click yes to all the defaults in the next window. Open the project view (command + 1).

Open the content\_main.xml file, which can be found under app/res/layout/. You may need to wait a bit for the build and file indexing to complete.

We will see the app like this



1. We will add some objects (TextView, EditText) and make the app like this, see the name of every tool on the left.



Drag and drop two TextView objects and two EditText objects over to the phone. Arrange them as you wish. Edit the ID of the first TextView object to be “Username”, with text content being “User Name: “ as you see here. Similarly, edit the ID and text content of the second TextView object to be “password” and “Password: “, in that order.

The EditText object to the right of the “User Name” TextView object should have an ID of “etUserName”. Similarly enough, the EditText object to the right of the “Password” TextView object should have an ID of “etPassword”.

Basically, we have two objects per field (username and password): one TextView to act as a label, identifying what the field is for, and an EditText for the user to put in their information accordingly.

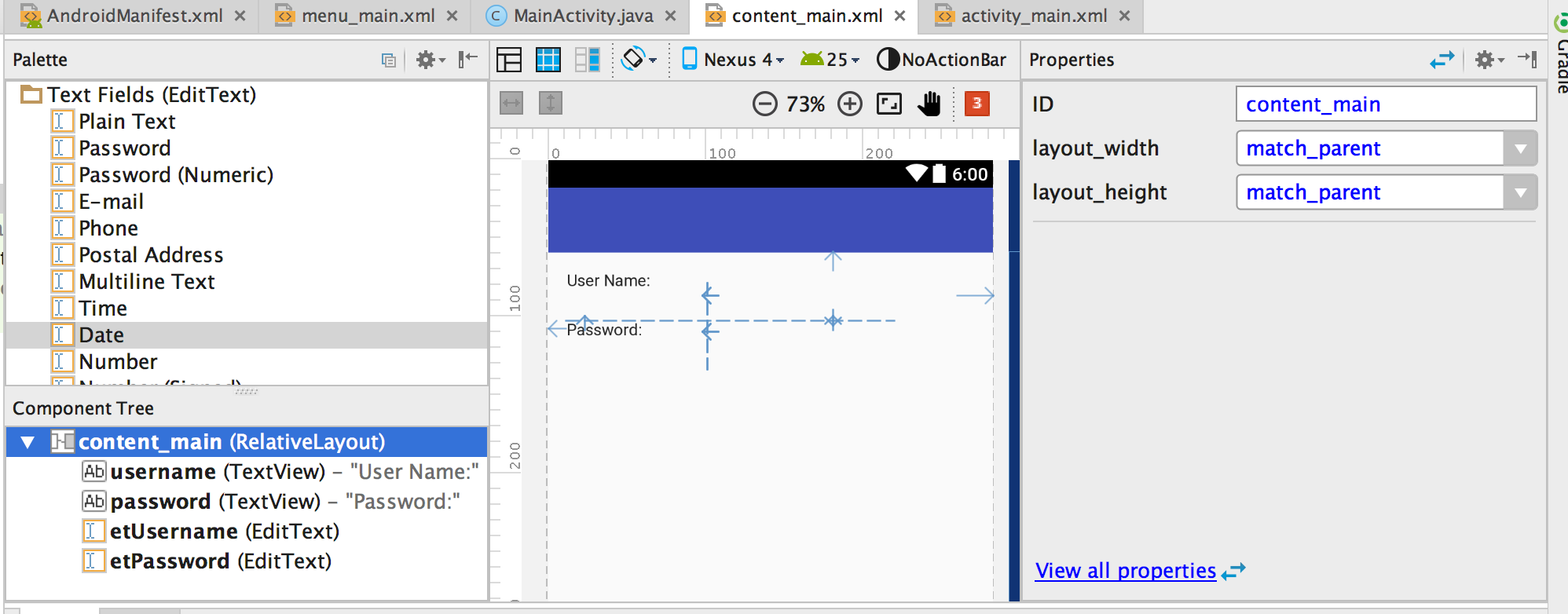
Add the object from the palette block. Scroll up and down until you find the ones you’ll need. You’ll need two TextViews for the labels, one EditText of the type Plain Text, and one more EditText of the type Password. Drag them into the center view of the app.

To edit their properties, simply click on the object under the Component Tree block. Its properties will appear in a block to the right, much like we see in the photo here.

1. Add the code below into the MainActivity.java file, found under the app/java/com.hussein.alrubaye.sharedref/. The code will be like this:

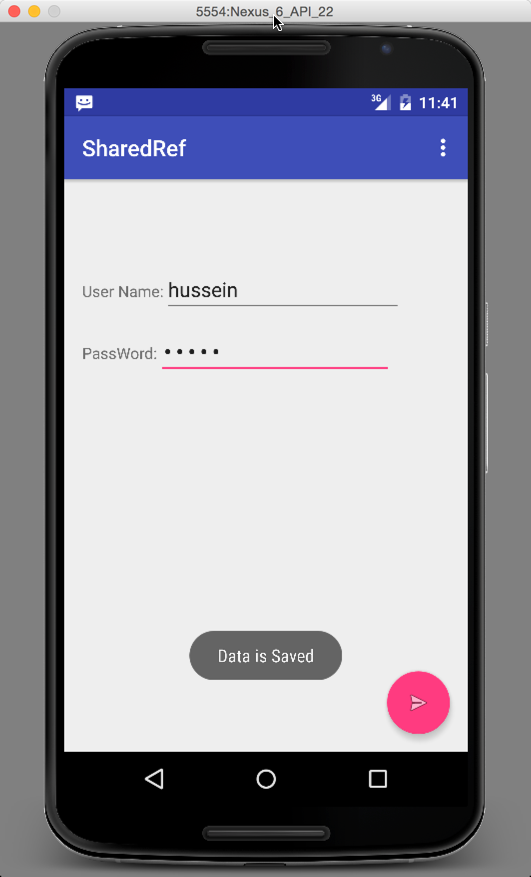
In addition to the code shown below, be sure to add the following line to the other imports at the top.

import android.widget.EditText;



|  |
| --- |
| Java |
| // shared references files name public static final String *MyPREFERENCES* = "MyPrfLogin" ; // key for user name public static final String *UserName* = "UserNameKey"; // key for password public static final String *Password* = "PasswordKey"; // shared references instance to access to virtual file SharedPreferences sharedpreferences; // input text name EditText etUserName; // input text password EditText etPassword; @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);  // initialize user name instance with the real input in xml  etUserName=(EditText)findViewById(R.id.*etUserName*);  // initialize password instance with the real input in xml  etPassword=(EditText)findViewById(R.id.*etPassword*);  // // initialize shared references  sharedpreferences = getSharedPreferences(*MyPREFERENCES*, Context.*MODE\_PRIVATE*);  // access to the floating button  FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.*fab*);  // listen to floating button when click  fab.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View view) {  // save data  // enable start editing file  SharedPreferences.Editor editor = sharedpreferences.edit();  // add user name  editor.putString(*UserName*, etUserName.getText().toString());  // add password  editor.putString(*Password*, etPassword.getText().toString());  // save the update data  editor.commit();  //display message saved  Toast.*makeText*(MainActivity.this, "Data is Saved", Toast.*LENGTH\_LONG*).show();   }}); } |

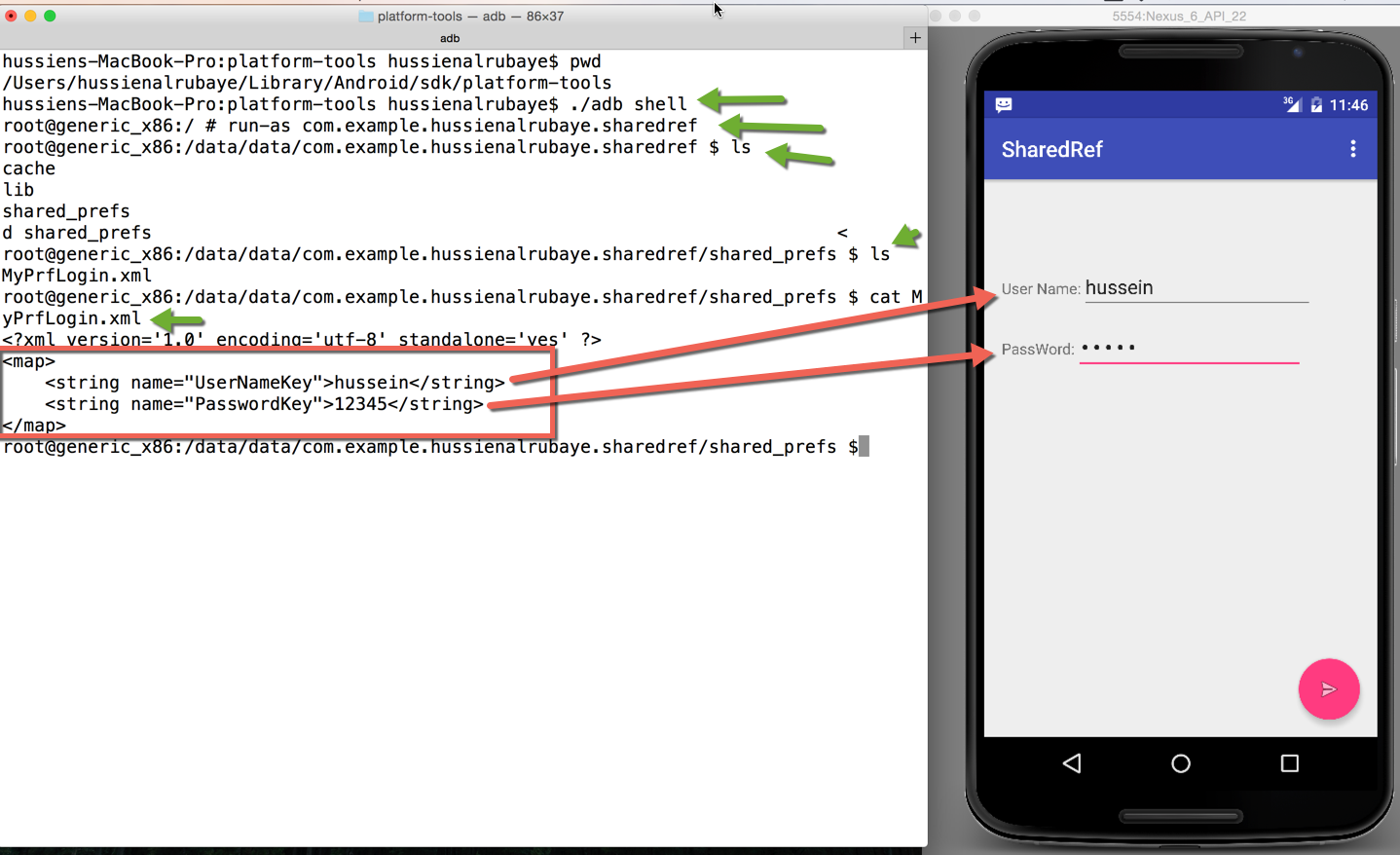
1. Run the app and add user name and password



**Steps to read the app data**

To read the app data, run Terminal and execute the commands as it shows below:

Please note that for the adb shell command to work, you must leave the Android emulator and the app open.



Conclusion:

As you see we could read the same data that stored in the app, if it was sensitive information this will be very bad.

**Fix The Problem:**

To fix the problem you have to have two solutions:

1. Encrypt sensitive data in the phone using encryption and decryption
2. Save sensitive data in the server and retrieve it when we need it.

**What we will do:**

We will demonstrate a small app that encrypts data before saving into the shared references, and when we want to use it, we will decrypt it.

**Steps:**

We will take the same app that we built before and encrypt the data before saving it into the shared file using a basic cipher.

Let us add this additional method into the MainActivity.java file:

|  |
| --- |
| Java |
| //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; } |

This encryption is very basic, just adding shift for every character when we want to encrypt data, and when we want to decrypt we subtract that shift to origin character.

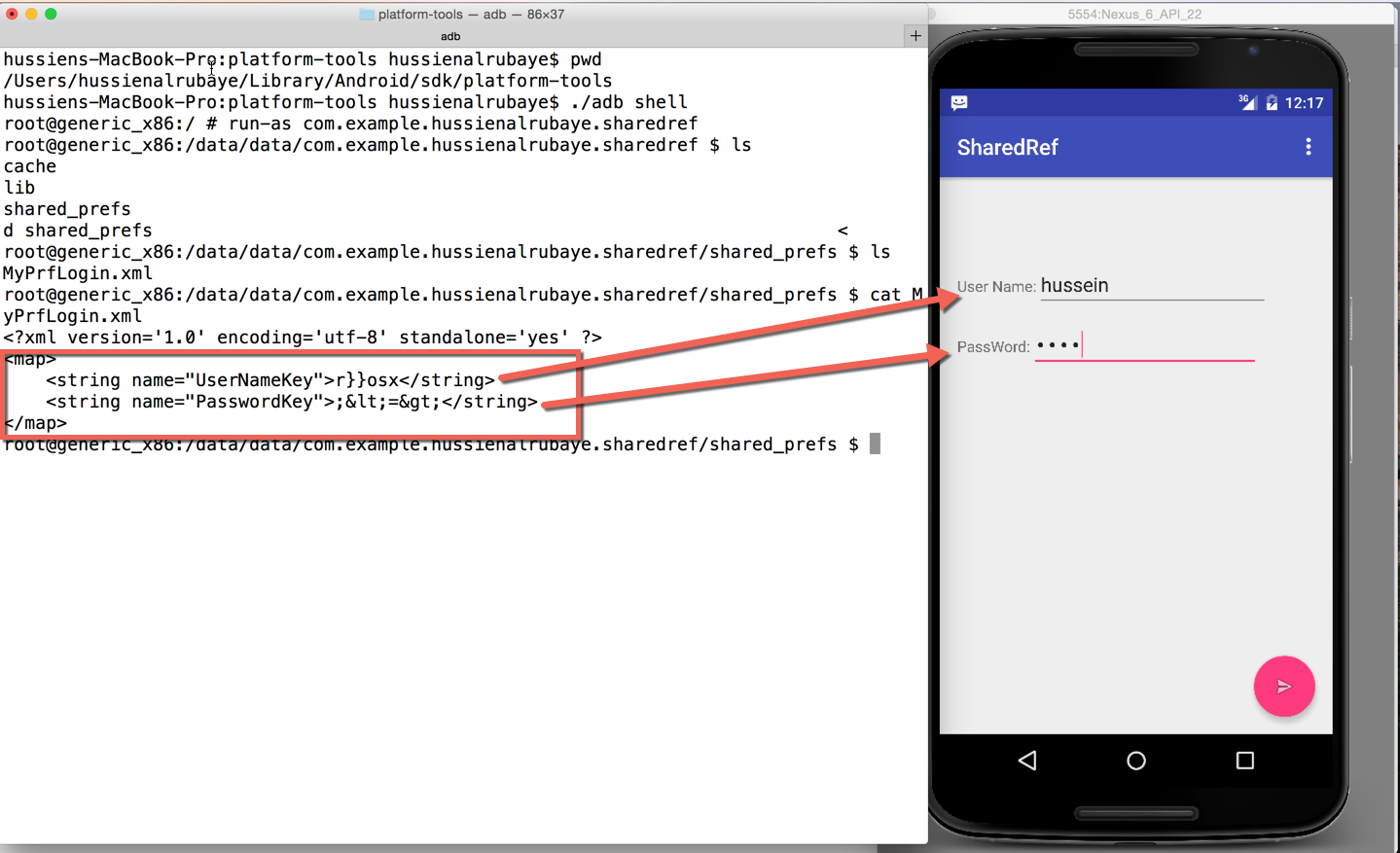
|  |
| --- |
| Java |
| // cipher encryption add shift for key cipher(“hussein”, 10) // result “r }}osx”  // cipher Decryptions  cipher(“r }}osx”, -10) / result “Hussein” |

The code will update to

|  |
| --- |
| Java |
| // shared references files name public static final String *MyPREFERENCES* = "MyPrfLogin" ; // key for user name public static final String *UserName* = "UserNameKey"; // key for password public static final String *Password* = "PasswordKey"; // shared references instance to access to virtual file SharedPreferences sharedpreferences; // input text name EditText etUserName; // input text password EditText etPassword; @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);  // initialize user name instance with the real input in xml  etUserName=(EditText)findViewById(R.id.*etUserName*);  // initialize password instance with the real input in xml  etPassword=(EditText)findViewById(R.id.*etPassword*);  // // initialize shared references  sharedpreferences = getSharedPreferences(*MyPREFERENCES*, Context.*MODE\_PRIVATE*);  // access to the floating button  FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.*fab*);  // listen to floating button when click  fab.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View view) {  // save data  // enable start editing file  SharedPreferences.Editor editor = sharedpreferences.edit();  // add user name  editor.putString(*UserName*,cipher(etUserName.getText().toString(),10));  // add password  editor.putString(*Password*,cipher( etPassword.getText().toString(),10));  // save the update data  editor.commit();  //display message saved  Toast.*makeText*(MainActivity.this, "Data is Saved", Toast.*LENGTH\_LONG*).show();   }  }); }  // 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; } |

**Steps to read the app data**

To read the app data, run Terminal and execute the commands as shown below:



See, you cannot understand the data because it is encrypted.