**MÓVILES.**
**Archivos en Android.**

**INTRODUCCIÓN.**
En Android se pueden utilizar los archivos de texto para guardar en el almacenamiento interno del equipo o la posibilidad es almacenarlo en una tarjeta SD Card.

Se puede diseñar un programa que permita abrir o almacenar datos utilizando la etiqueta del componente `EditText`, de tal forma que cuando se digite un botón se abran o almacenen los datos del `EditText` de un archivo de texto denominado, por ejemplo, datos.txt.

Cada vez que se pretenda abrir el archivo, se verifica si existe el archivo de textos datos.txt, proceder a su lectura o almacenamiento de datos con el componente `EditText`.

Utilizar un editor de texto para obtener el archivo misdatos.txt; el contenido de este archivo debe incluir la primera estrofa del himno nacional mexicano.

**DESARROLLO**

**EJERCICIO 1**.
**Paso 1**. Crear un nuevo proyecto y abrir el archivo MainActivity.java para modificar su contenido con el código que se indica enseguida.

```
// NOTA: Agregar las bibliotecas de clases necesarias.
public class MainActivity extends Activity{ // Abre un archivo almacenado
      TextView    tv;
      String            s;
      InputStream is;
      InputStreamReader isr;
      BufferedReader  br;
      public void onCreate(Bundle b){
            super.onCreate(b);
            setContentView(R.layout.activity_main);
            tv    = (TextView) findViewById(R.id.xtv);
            tv.append("\nAbriendo: res/raw/misdatos.txt"); // NOTA: Este es su archivo.
            is    = getResources().openRawResource(R.raw.misdatos);
            isr   = new InputStreamReader(is);
            br    = new BufferedReader(isr, 8192);
            try{
                  while( null != (s=br.readLine()) )
                        tv.append("\n" + s);
                  is.close();
                  isr.close();
                  br.close();
            } catch(Exception e){
                  tv.append("\n " + e);
            }
            tv.append("\nEnd of file.");
      }
}
```

**Paso 2**. Abrir el archivo activitiy_main.xml para modificar el contenido con el código siguiente.

```
<!—Archivos xml para manejo de archivos -->
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:showIn="@layout/activity_main"
    tools:context=".MainActivity"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/xtv"
        android:text="***** ALMACEN *****"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

**Paso 3**. Abrir el archivo strings.xml para modificar su contenido con el código siguiente.

```
<resources>
    <string name="app_name">MisArchivos</string>
    <string name="action_settings">Settings</string>
</resources>
```

**Paso 4**. Abrir el archivo manifest.xml para modificar su contenido con el siguiente código mostrado en negritas.

```
<!-- El archivo manifest.xml -->
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.escom.misarchivos" >
    <application
        :
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            :
        </activity>
    </application>
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE">
    </uses-permission>
</manifest>
```

**Paso 5**. Obtener las imágenes de la ejecución de la aplicación.

**EJERCICIO 2**.

**Paso 1**. Crear un nuevo proyecto y abrir el archivo MainActivity.java para modificar su contenido con el código que se indica enseguida.

```
import java.io.*;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.*;
public class MainActivity extends Activity {
    EditText jet1;
    static final int READ_BLOCK_SIZE = 100;
    @Override
    protected void onCreate(Bundle b) {
        super.onCreate(b);
```

```
        setContentView(R.layout.activity_main);
        jet1 = (EditText) findViewById(R.id.xet1);
    }
    public void guardar(View v){
        String str = jet1.getText().toString();
        try{
            FileOutputStream fos = openFileOutput("misdatos.txt", MODE_PRIVATE);
            OutputStreamWriter osw = new OutputStreamWriter(fos);
            osw.write(str);
            osw.flush();
            osw.close();
            Toast.makeText(getBaseContext(),              "Archivo              guardado",
Toast.LENGTH_SHORT).show();
            jet1.setText("");
        }catch (IOException ex){
            ex.printStackTrace();
        }
    }
    public void abrir(View v){
        try{
            FileInputStream fis = openFileInput("misdatos.txt");
            InputStreamReader isr = new InputStreamReader(fis);
            char[] buffer = new char[READ_BLOCK_SIZE];
            String s = "";
            int n;
            while((n = isr.read(buffer)) > 0){
                String readString = String.copyValueOf(buffer, 0, n);
                s += readString;
                buffer = new char[READ_BLOCK_SIZE];
            }
            jet1.setText(s);
            Toast.makeText(getBaseContext(),              "Archivo              abierto",
Toast.LENGTH_SHORT).show();
            isr.close();
        }catch (IOException ex){
            ex.printStackTrace();
        }
    }
}

<?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" >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Archivos" />
    <EditText
        android:id="@+id/xet1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Ingresar texto..."/>
    <Button
        android:id="@+id/xbn1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
        android:onClick="guardar"
        android:text="Guardar" />
    <Button
        android:id="@+id/xbn2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="abrir"
        android:text="Abrir" />
</LinearLayout>
```

**Paso 2**. Abrir el archivo activitiy_main.xml para modificar el contenido con el código siguiente.

```
<?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" >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Archivos" />
    <EditText
        android:id="@+id/xet1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Ingresar texto..."/>
    <Button
        android:id="@+id/xbn1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="guardar"
        android:text="Guardar" />
    <Button
        android:id="@+id/xbn2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="abrir"
        android:text="Abrir" />
</LinearLayout>
```

**Paso 3**. Obtener las imágenes de la ejecución de la aplicación.

**EJERCICIO 3**.

**Seguridad de archivos.**

Para calcular el valor hash criptográfico en Java, se utiliza la clase `MessageDigest`, en el paquete `java.security`. La clase `MessageDigest` proporciona la siguiente función hash criptográfica para encontrar el valor hash de un texto de los siguientes algoritmos:

MD2, MD5, SHA-1, SHA-224, SHA-256, SHA-384 y SHA-512

De la siguiente manera. Estos algoritmos se inicializan en un método estático llamado `getInstance`(). Después de seleccionar el algoritmo, se calcula el valor de resumen del mensaje y los resultados se devuelven como una matriz de bytes. Se utiliza la clase `BigInteger` para convertir la matriz de bytes resultante en su representación `signum`. Esta representación luego se convierte a un formato hexadecimal para obtener el MessageDigest esperado. Por ejemplo:

**Input:** hola mundo **Output:** 2aae6c35c94fcfb415dbe95f408b9ce91ee846ed

**Las claves SHA1, MD5 y SHA-256 en Android Studio**
SHA1, MD5 y SHA-256 son funciones criptográficas que convertirán la entrada a un valor de 160 bits (20 bytes). Es una clave segura que se utiliza para almacenar datos muy importantes. Si se desea agregar API externas de Google, como Maps y otras funciones externas dentro de la aplicación, se debe agregar la clave a la consola de Google. Las claves SHA son la única identidad de una aplicación. Estas claves son importantes y necesarias cuando se agrega la aplicación a la tienda Google Play.

**1. Claves SHA-1**
SHA-1 también se conoce como algoritmo de hash seguro. Es una función hash criptográfica que toma una entrada y produce un valor hash de 160 bits. Este valor hash generado se conoce como resumen de mensaje y se representa luego en un número de formato hexadecimal que tiene 40 dígitos. Esta clave fue diseñada por la NSA, Agencia de Seguridad Nacional de los Estados Unidos, y se utilizó como estándar de procesamiento de información. La clave SHA se introdujo en 1995.

La clave SHA-1 se utiliza para criptografía. Los datos de entrada se convierten en un valor hash de 160 bits que es difícil de decodificar y se utilizan para la integridad de datos.

**2. Claves MD-5**
Las claves MD-5 se conocen como Message Digest. Son más rápidas que SHA-1 y su uso es más simple que las claves SHA-1. MD-5 tiene una longitud del resumen del mensaje de 128 bits. MD-5 tiene poca seguridad en comparación con las claves SHA-1. La llave MD-5 se introdujo en 1992.

Las claves MD-5 se utilizan para criptografía para almacenar datos en valores de 128 bits.

**3. Teclas SHA-256**
SHA-256 se conoce como algoritmo de hash seguro 256. Es una función hash criptográfica que toma una entrada de 20 bytes y representa este valor en formato hexadecimal. El valor generado se conoce como el resumen del mensaje y se representa con 40 dígitos.

Las claves SHA-256 se utilizan en criptografía para la integridad de datos.

**Generación de claves SHA1, MD5 y SHA-256 en Android Studio**
1. Crear un nuevo proyecto en Android Studio. También se pueden generar estas claves en proyectos ya existentes.
2. Dentro de Android Studio, hacer clic en la pestaña Gradle.

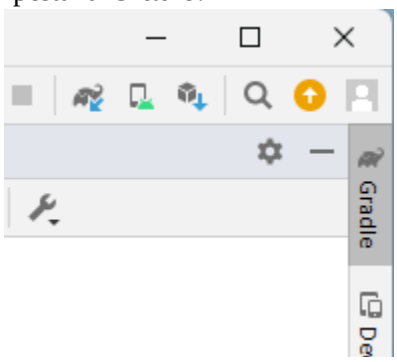



**Figura 1**. Pestaña Gradle.

3. En la siguiente pantalla hacer clic en su AppName. En las tres opciones, hacer clic en las últimas opciones app. Enseguida, ir a Tasks y android, y hacer doble clic en la opción signingReport.

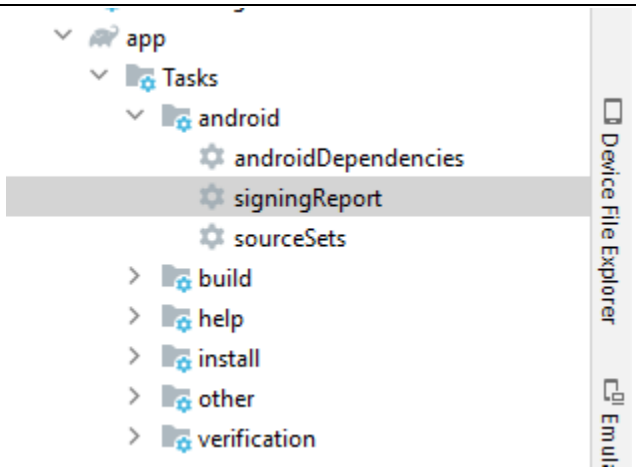



**Figura 2**. El menú app>Tasks>android>signingReport.

Después de hacer clic en el signingReport, en la consola se generarán las claves como se muestra enseguida en la figura 3.

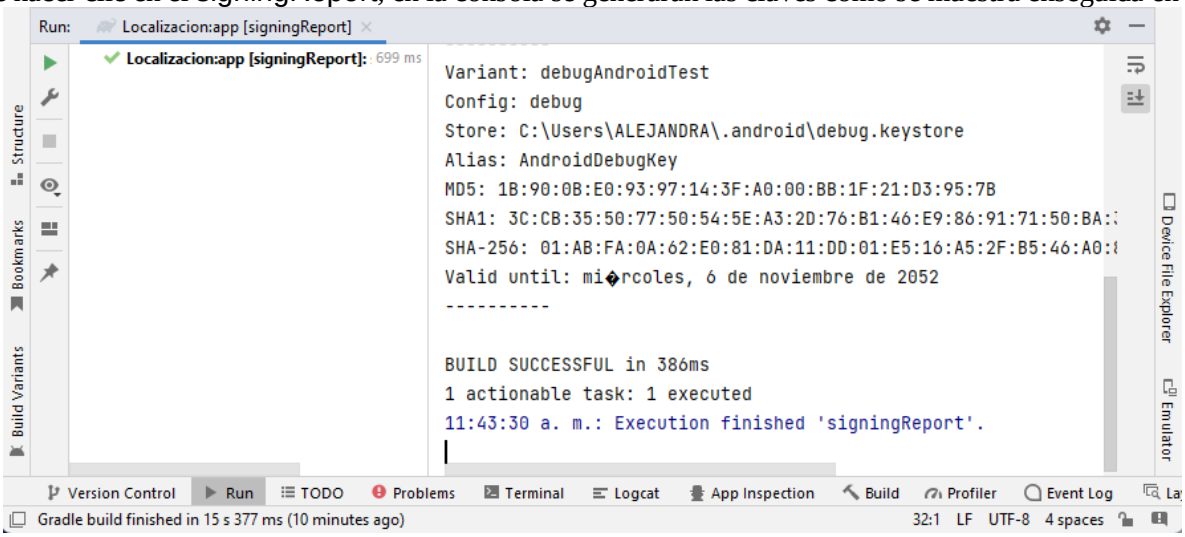



**Figura 3**. La consola de signingReport y las claves MD5, SHA1 y SHA-256 obtenidas.

Se puede usar estas claves mientras se integra una aplicación; por ejemplo, con Firebase. Estas claves son muy importantes cuando se necesita agregar las API de Google dentro de una aplicación.

**NOTA.**
Opciones para el acceso a las claves de signingReport.
**Opción 1**. Acceder a las claves con signingReport utilizando la Terminal:
En la ventana del IDE Android Studio, en la sección inferior de la Consola, seleccionar la pestaña Terminal.

Ingresar la siguiente instrucción: `./gradlew signingReport` , y digitar Enter.

**Opción 2**. En el menú principal, seleccionar File>Settings>Experimental.
- Deshabilitar la opción Do not build Gradle task list during Gradle sync

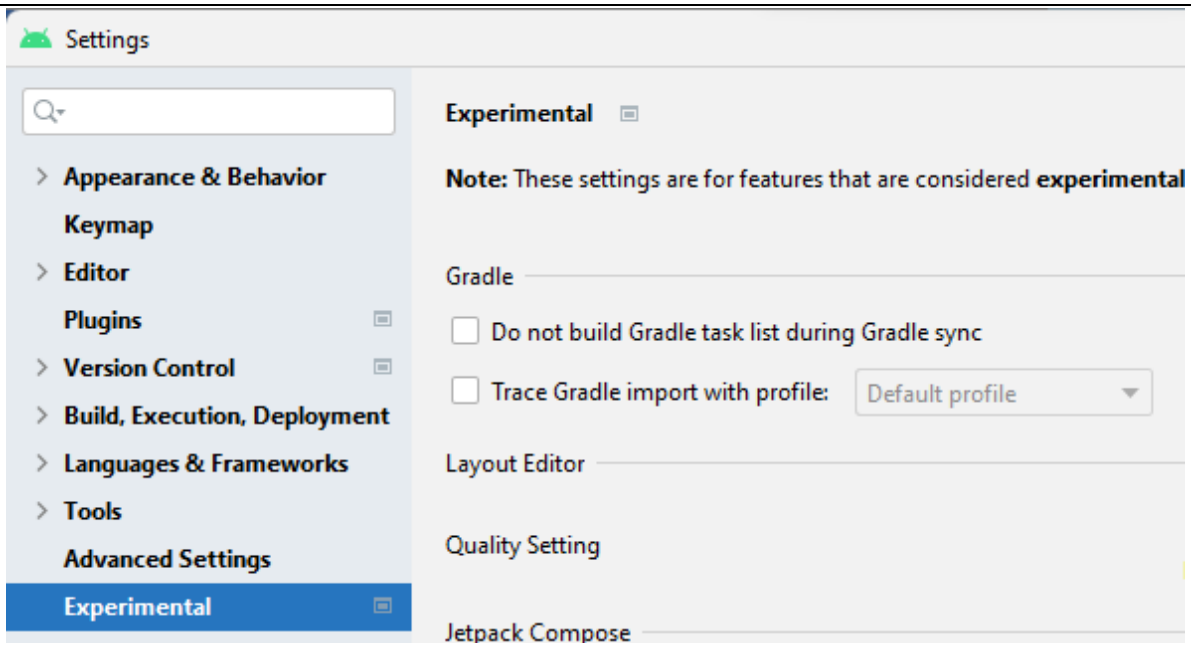
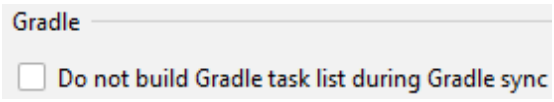
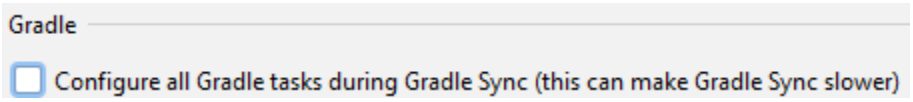


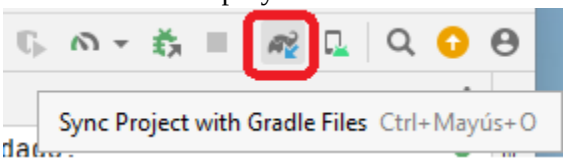

**Figura 4**. Deshabilitar la opción Do no build Gradle task during Gradle sync.




O también:




- Sincronizar el proyecto.




- Observar el resultado del signingReport en la pestaña de Gradle.

Programa en Java para utilizar el algoritmo SHA-1:

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class GFG {
    public static void main(String args[]) throws NoSuchAlgorithmException{
        System.out.println("HashCode Generado por SHA-1 para: ");
        String s = "Hola ESCOM";
        System.out.println("\n" + s + " : " + cifrar(s));
    }
    public static String cifrar(String input){
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-1");
            byte[] messageDigest = md.digest(input.getBytes());
            BigInteger no = new BigInteger(1, messageDigest);
            String hashtext = no.toString(16);
```

```
            while (hashtext.length() < 32) {
                hashtext = "0" + hashtext;
            }
            return hashtext;
        }
        catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Ejecutar el código, verificar y reportar los resultados.

**NOTA**. Generar un reporte con las imágenes de la ejecución de las aplicaciones. Guardar el archivo con la sintaxis ArchivosArchivosGrupo.pdf y enviarlo al sitio indicado por el profesor.