|  |
| --- |
| http://www.becas.sep.gob.mx/images/logo.png  TECNOLÓGICO NACIONAL DE MÉXICO  Instituto Tecnológico de Chihuahua II |
| Resumen Practica 4 unidad 3 |
| Programa que graba archivos en una sd. |
|  |
|  |
| 10/05/2016  Unidad de la Materia: III  Nombre del Alumno: Jose Luis Castro Lozoya.  No. Control: 12550479  Materia: Programacion de dispositivos móviles plataforma 2 |

# Introducción

Confeccionar un programa que permita ingresar el nombre de un archivo y el

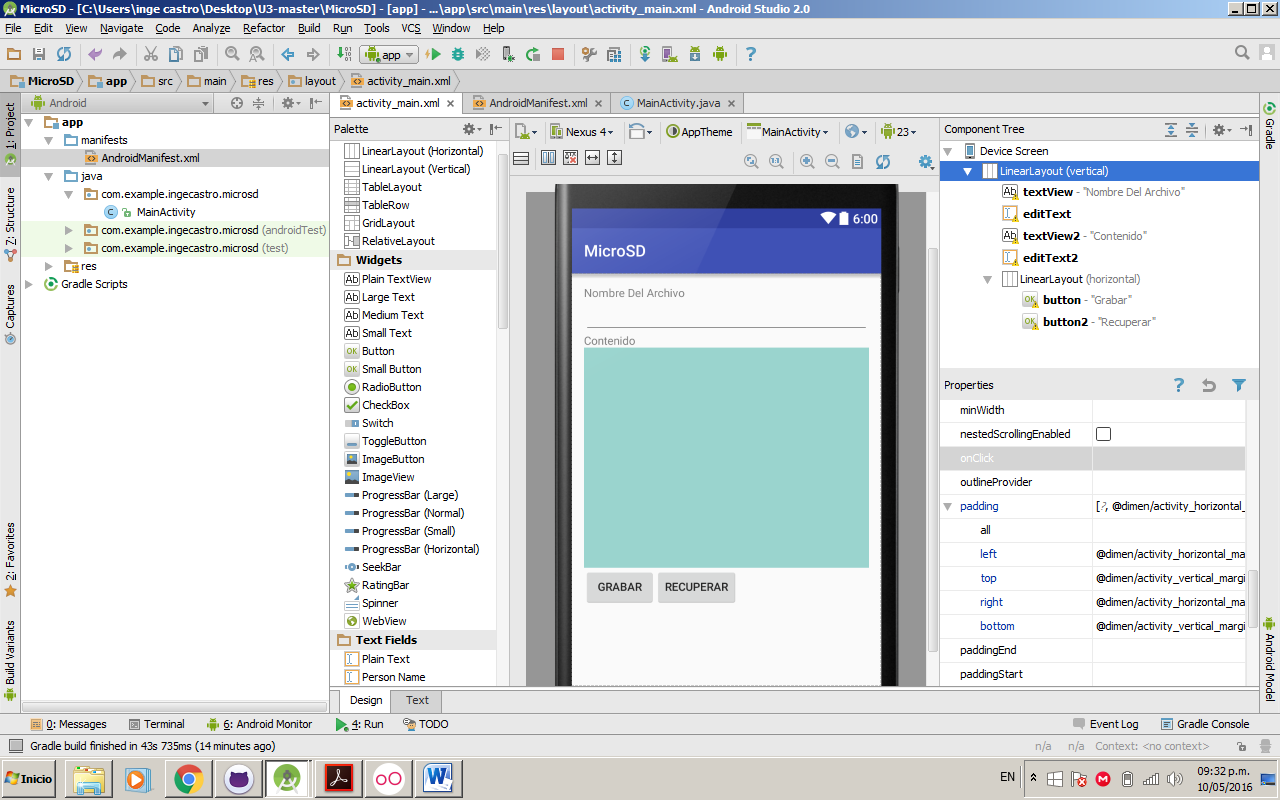
contenido. Permitir grabar los datos ingresados al presionar un botón. Disponer un

segundo botón que permita recuperar los datos del archivo de texto.

1. Hacer que los archivos se graben en una tarjeta SD.

# Desarrollo

La interfaz visual es de la siguiente manera



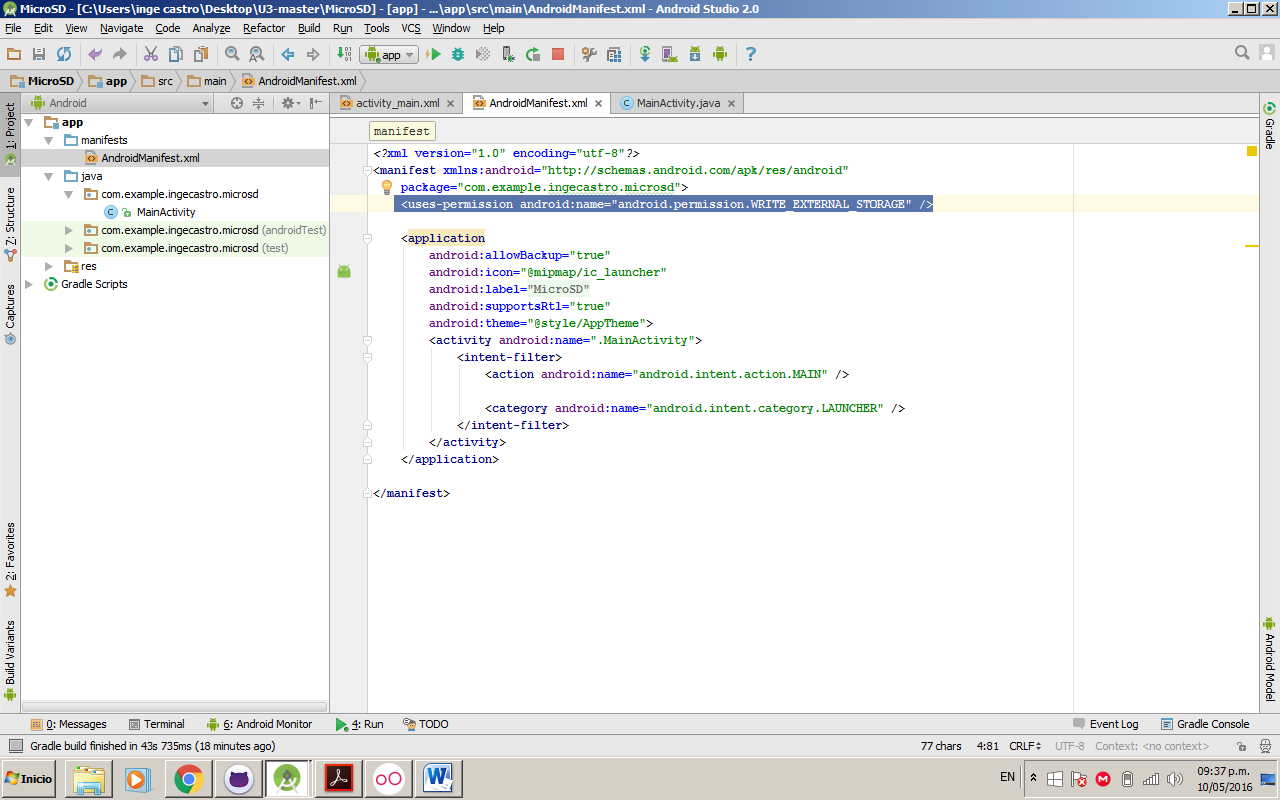
Podemos ver en la ventana "Component Tree" que la interfaz contiene dos TextView, dos EditText y dos Button.

No olvidar inicializar las propiedades onClick de cada botón con los nombres de métodos "grabar" y "recuperar".

El primer paso es modificar el archivo AndroidManifest.xml para permitir el acceso a la tarjeta SD desde nuestra aplicación esto lo hacemos desde el editor de texto del Android Studio

En la carpeta app/manifests podemos abrir el archivo "AndroidManifest.xml" y agregar la línea de permiso de acceso a la memoria externa del dispositivo:

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />



El método para grabar los datos en un archivo de texto localizado en una tarjeta SD comienza obteniendo el directorio raiz de la tarjeta a través del método getExternalStorageDirectory(), el mismo retorna un objeto de la clase File.

public void grabar(View v) {

String nomarchivo = et1.getText().toString();

String contenido=et2.getText().toString();

try

{

File tarjeta = Environment.getExternalStorageDirectory();

Creamos un nuevo objeto de la clase File indicando el camino de la unidad SD y el nombre del archivo a crear:

File file = new File(tarjeta.getAbsolutePath(), nomarchivo);

Por último similar al acceso de un archivo interno creamos un objeto de la clase OutputStreamWriter:

OutputStreamWriter osw =new OutputStreamWriter(new FileOutputStream(file));

Grabamos el contenido del EditText:

osw.write(contenido);

Cerramos el archivo:

osw.flush();

osw.close();

Toast.makeText(this,"Los datos fueron grabados correctamente",Toast.LENGTH\_SHORT).show();

et1.setText("");

et2.setText("");

}

catch (IOException ioe)

{

Toast.makeText(this, "No se pudo grabar",

Toast.LENGTH\_SHORT).show();

}

}

Para la lectura del archivo nuevamente obtenemos la referencia de la tarjeta SD para obtener el path de la unidad de almacenamiento, el resto del algoritmo es similar al visto con un archivo interno:

public void recuperar(View v) {

String nomarchivo = et1.getText().toString();

File tarjeta = Environment.getExternalStorageDirectory();

File file = new File(tarjeta.getAbsolutePath(), nomarchivo);

try {

FileInputStream fIn = new FileInputStream(file);

InputStreamReader archivo=new InputStreamReader(fIn);

BufferedReader br=new BufferedReader(archivo);

String linea=br.readLine();

String todo="";

while (linea!=null)

{

todo=todo+linea+"\n";

linea=br.readLine();

}

br.close();

archivo.close();

et2.setText(todo);

} catch (IOException e)

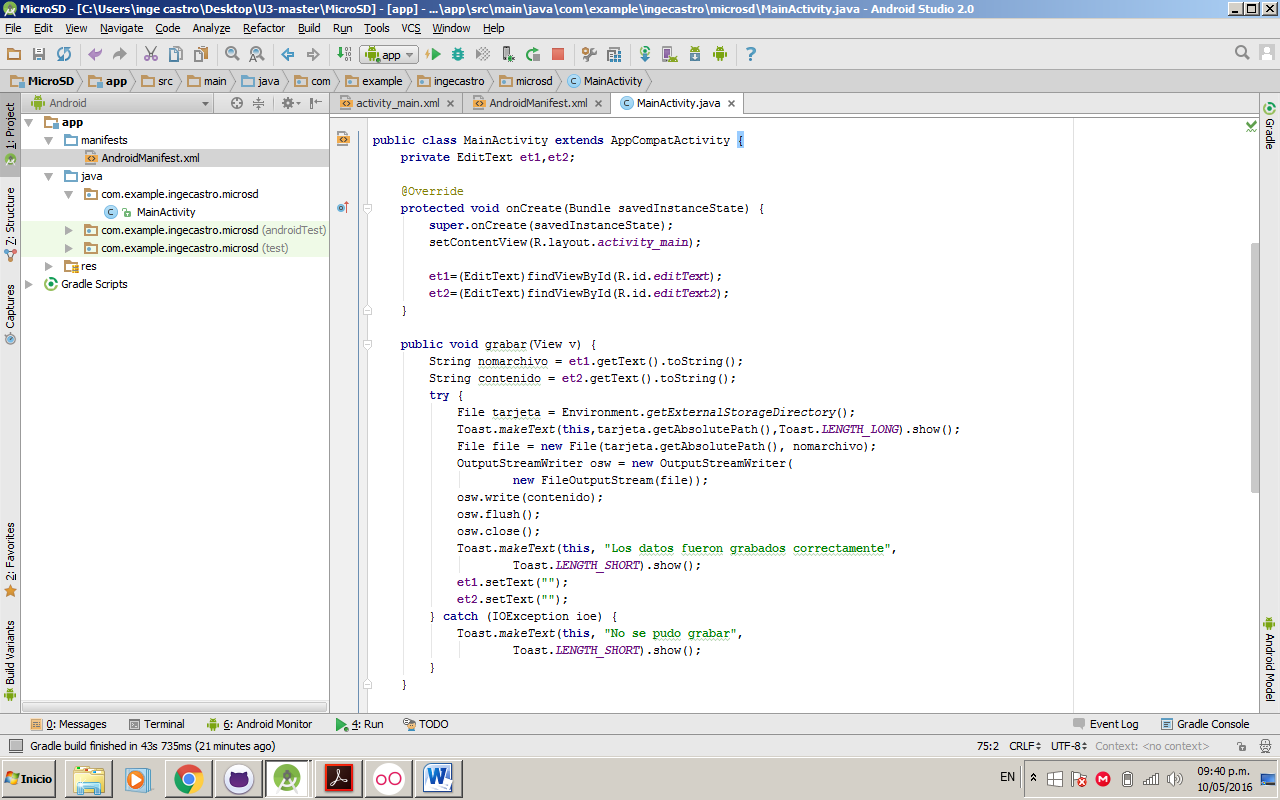
{

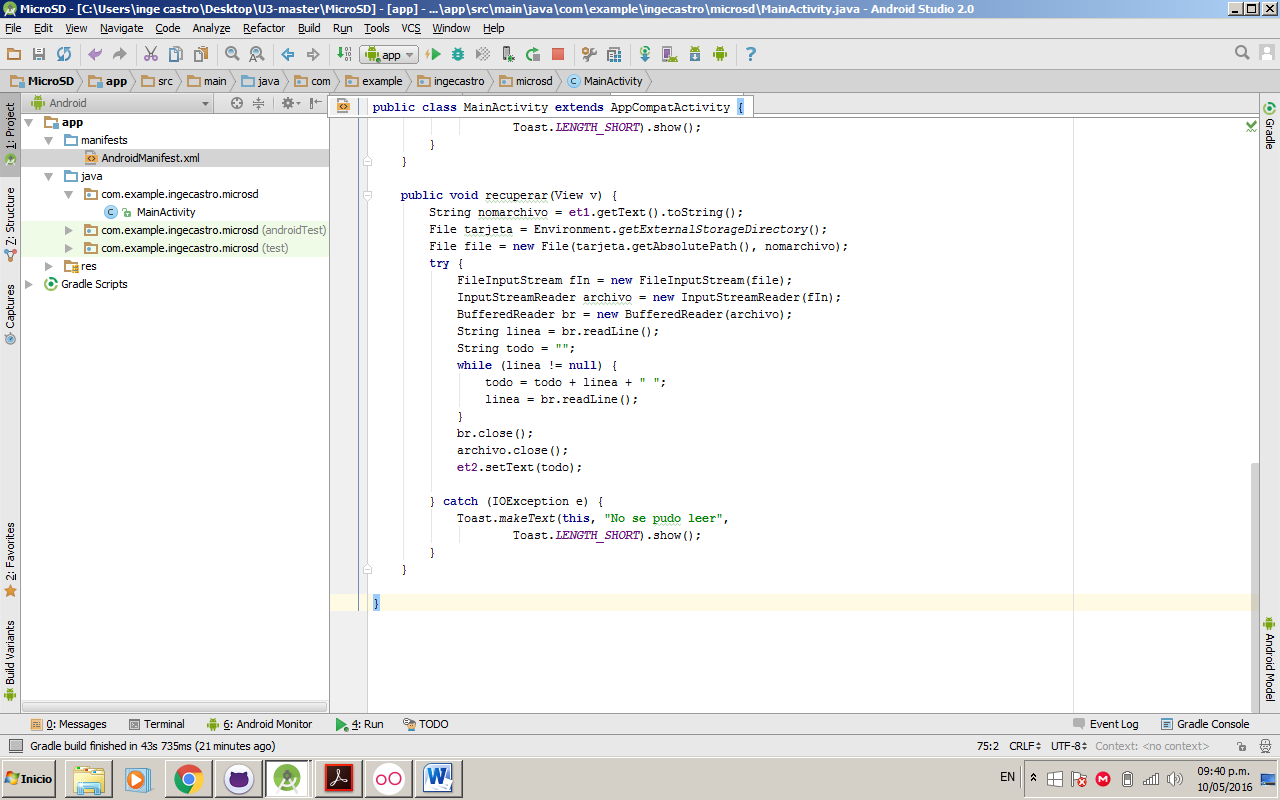
Toast.makeText(this, "No se pudo leer",

Toast.LENGTH\_SHORT).show();

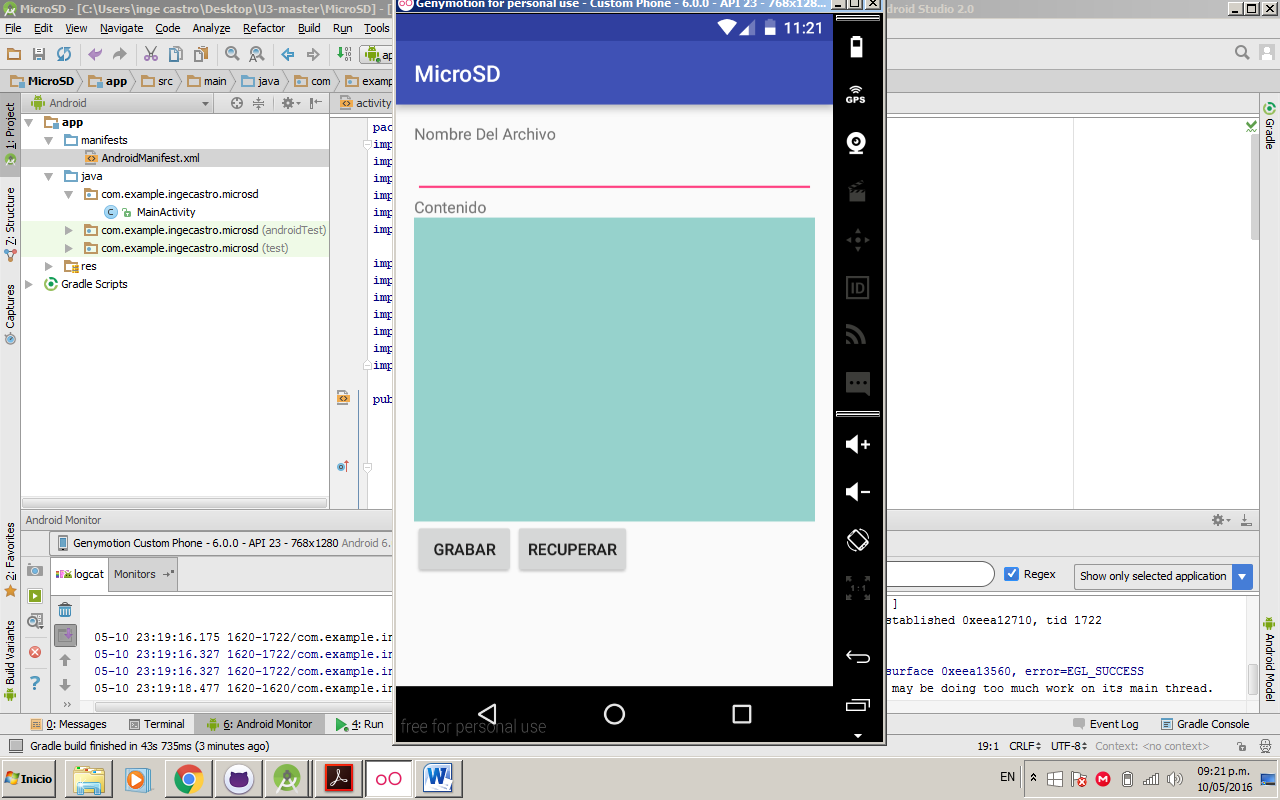
}

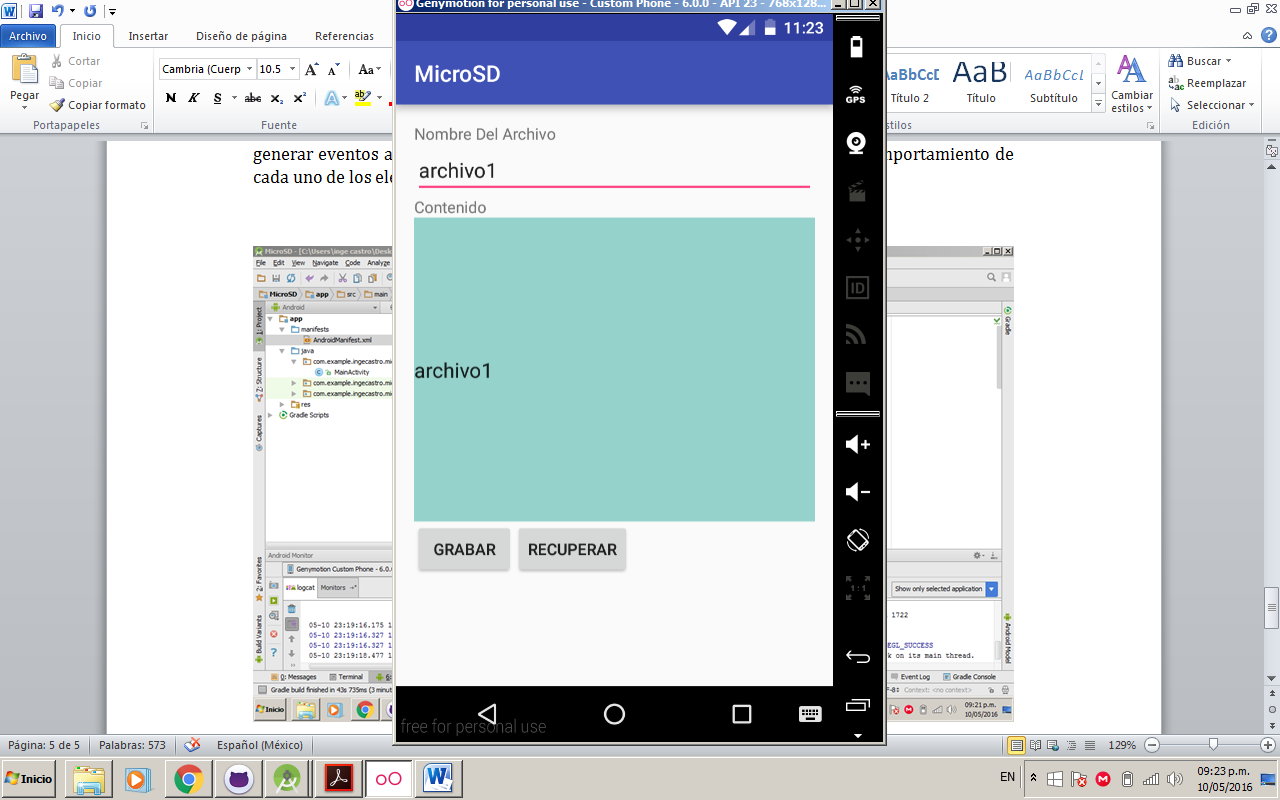
}





A continuación se ejecuta el programa





# Conclusión.

En algunas situaciones podría ser útil almacenar los datos en una tarjeta SD (tener en cuenta que no todos los dispositivos Android cuentan con esta característica), esto debido a su mayor capacidad o la facilidad de compartir los archivos con otras personas entregando la tarjeta SD.