**Ejercicio: Memoria Interna (ESCRITURA)**

Android proporciona para ello el métodoopenFileOutput(), que recibe como parámetros el nombre del fichero y el modo de acceso con el que queremos abrir el fichero (MODE\_PRIVATE, MODE\_APPEND , MODE\_WORLD\_READABLE o MODE\_WORLD\_WRITABLE)

Este método devuelve una referencia al streamde salida asociado al fichero (en forma de objetoFileOutputStream), a partir del cual ya podremos utilizar los métodos de manipulación de ficheros tradicionales del lenguaje java (api java.io). Como ejemplo, convertiremos este stream a un OutputStreamWriter para escribir una cadena de texto al fichero.



Este archivo se almacena en:

/data/data/paquete\_java/files/nombre\_fichero

**Ejercicio: Memoria Interna (LECTURA)**

Para leer  la única diferencia de que utilizaremos el método openFileInput() para abrir el fichero, y los métodos de lectura de java.io para leer el contenido.



**Ejercicio: Memoria Interna (COMO RECURSO)**

Aunque este método es útil en muchos casos, sólo debemos utilizarlo cuando no necesitemos realizar modificaciones sobre los ficheros, ya que tendremos limitado el acceso a sólo lectura.

Para incluir un fichero como recurso de la aplicación debemos colocarlo en la carpeta “/res/raw” de nuestro proyecto de Eclipse.



Como puede verse en el código anterior, al método openRawResource() le pasamos como parámetro el ID del fichero incluido como recurso, que seguirá el patrón “***R.raw.***nombre\_del\_fichero“, por lo que en nuestro caso particular será R.raw.prueba\_raw.

**Problema:**

Realizar un programa que permita almacenar notas en un control EditText y cuando se presione un botón almacenar los datos del EditText en un archivo de texto llamado "notas.txt".

 Cada vez que se ingrese al programa verificar si existe el archivo de textos "notas.txt" proceder a su lectura y almacenamiento de datos en el EditText.