UF3 FICHEROS.JAVA Cheat Sheet

by kattorres via cheatography.com/148189/cs/32304/

FICHEROS:

Soporte de **memoria externa que almacena y registra datos,** para mantenerlos persistentemente en todas la ejecuciones.

PAQUETE:

package java.io	Contiene las clases para manejar **ficheros**

TIPOS DE RUTAS:

ABSOLUTA: `C:/Pro gra mas /un ida d6/ apa rta do1 /ej emp los /Fi leO b te ner Rut a.java`

RELATIVA: `Aparta do1 /ej emp los /Fi leO bte ner Rut a.java`

RUTA:

Ruta Unix: `/usr/bin`

Ruta Windows: `C:\Win do w s \S ystem32`

CLASE FILE:

Ficheros y carpetas	Operaciones sobre rutas de ficheros existentes o no
Importar e **inicializar:**	`file F = new file (String ruta)`

ARRAY FILE:

E006listado.java

`File[] listFi les( )`	Donde se enumeran todos los elementos de la carpeta

Ejemplo de inicialización:

```
File carpeta = new File("C :/T emp ");
File [] arrayE lem entos = carpet a.l ist Fil es();
```

FILE BYTE:

La diferencia fundamental es que no utiliza la representación en forma sino que usa el formato binario. La ventaja es que ocupan me

E022binario.java

INICIALIZACION:

Clase RandomAccessFile → package java.io

```
Random Acc essFile raf = new Random Acc ess Fil e(Fil
ode);
```

Leer un archivo:

```
Random Acc essFile raf = new Random Acc ess Fil
```

Escribir en un fichero:

```
Random Acc essFile raf = new Random Acc ess Fil
```

METODOS:

E022binario.java

```
raf.wr ite Int();
```

E023sobreescribirBin.java

`setLength(long medida);`	Modifica la medida del fichero S excedente Si es mayor, s
`long getFilePointer();`	Devuelve el desplazamiento

E0.java

``

FILE METODOS:

E001file.java

`getParent()`	Devuelve el padre del `file` o null
`getName()`	Devuelve el nombre del objeto dado
`getAbsolutePath( )`	Muestra ruta absoluta

E002estado.java

`isFile()`	**Boolean**si la ruta del archivo abstracto es un archivo o no.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

FILE METODOS: (cont)

`isDirectory()`	**Boolean** comprueba si el fichero instanciado es un directorio o no
`.exists();`	**Boolean** comprueba si si existe un fichero o directorio instanciado con la clase File

E003lecturaModificacion.java

`long length()`	La función devuelve un valor largo que representa el número de bits; de lo contrario, devuelve 0L
`long lastModified( )`	Evalúa la última fecha de edición del elemento.

```
Date fecha = new Date(documento.lastModified()
```

E004gestionFicheros.java

`mkdir()`	crear un nuevo directorio indicado por la ruta de acceso abstracta. La función devuelve BOOLEAN
`delete()`	Permite eliminar ficheros y carpetas, devuelve un boolean

E005rename.java

`renameTo(File destino)`

METODOS EXT:

E018vocales.java

`.tolowercase() ;`	Convierte una cadena en minúscula.
`.toUppercase ();`	Convierte una cadena en mayúscula.

E019enterosOrdenados.java

`.sort();`	Ordena un arreglo, array.

LECTURA / ESCRITURA FILE

LECTURA:

Clase Scanner → package java.util (→ lectura)

INICIALIZACIÓN:

```
Scanner lectura = new Scanne r(File f);
```

E012lectura.java

`f.close();`	Al terminar una lectura, cerrar siempre.
`lectura.next();`	Lee caracteres no separados por espacios o saltos.
`lectura.nextInt();`	lee una frase completa hasta el siguiente salto de linea

ESCRITURA:

Clase PrintStream → packege java.io (→ escritura)

INICIALIZACIÓN:

```
PrintS tream escribir = new PrintS tre am(File f);
```

E014escribeEnteros.java

`escribir.println() ;`	Hace salto de linea.
`escribir.print();`	El puntador avanza un valor.

SENTENCIA TRY/CATCH

Todas las instrucciones vinculadas a trabajar con ficheros harán uso de esta sentencia.

TRY {}	Mientras no haya excepciones se ejecuta este bloque de código
catch (Exception e) {}	Se ejecuta cuando existen excepciones

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com