



Programando con B4X

Tema 17 – Ficheros

Version 1.0, mayo 2021

Autor original: [Prokopis Leon](#)

Traducido al español por [José Miguel López](#)



Tema 17 – Ficheros

🕒 2h

Lo que los estudiantes aprenderán

- ¿Qué es un fichero?
- Ubicación de ficheros en B4J
- Métodos con Ficheros

Un fichero es una colección de datos con contenido similar que normalmente se almacena de forma permanente en el disco duro del ordenador. Es una de las características más importantes de un lenguaje de programación ya que, aunque los datos temporales se van guardando en la memoria del ordenador, cuando finaliza nuestro programa, sus datos deben quedar guardados de forma permanente.

engl_it.txt

| | |
|---------|-----------|
| Memory | Memoria |
| Screen | Schermo |
| Printer | Stampante |

Imagen 1. Fichero engl_it.txt

En general, se puede afirmar que los ficheros se pueden dividir en bases de datos y en ficheros simples que serán los que veamos.

Carpetas o Directorios de almacenamiento

Cada sistema operativo posee diferentes carpetas/directorios para guardar los datos de las aplicaciones y otros ficheros. Para que B4J pueda trabajar con ficheros independientemente del sistema operativo usado, utiliza palabras clave para referirse a un tipo concreto de carpeta.

File.DirAssets

Incluye los ficheros contenidos en la carpeta de la aplicación que ha sido añadida al gestor de ficheros de B4J durante la fase de desarrollo de la misma. Estos ficheros son de sólo lectura y no se pueden añadir ficheros mientras se está ejecutando la aplicación. Estos ficheros son creados por el programador para que sean copiados en el proceso de instalación.

xui.DefaultFolder

Devuelve la carpeta donde se guardan los datos de la aplicación, de forma similar a lo que hace **File.DirData**. Es obligatorio invocar una vez al método **SetDataFolder** antes de poder usarlo.

```
xui.SetDataFolder(AppName As String)
```



File.DirData

Devuelve la carpeta donde se guardan los datos de la aplicación y se puede usar para crear ficheros y guardar datos.

En el SO Windows, devuelve la carpeta de datos del usuario ("user data folder") que normalmente está en esta ruta:

C:\Users\[nombre usuario]\AppData\Roaming\[AppName]

En SO no-Windows, devuelve la carpeta donde está instalada la aplicación.

File.DirApp

Devuelve la carpeta donde está instalada la aplicación. En Windows esta carpeta está en "Archivos de Programa" y es de sólo lectura.

File.DirTemp

Devuelve la carpeta para ficheros temporales.

| | |
|-------------------------------|---|
| Log(File.DirApp) | C:\Users\usuario\DOCUMENTOS\GitHub\TEACH\1\TEMA17\1\Ejemplo\Ejemplo1\B4\Objects |
| Log(File.DirAssets) | AssetsDir |
| Log(File.DirTemp) | C:\Users\usuario\AppData\Local\Temp\ |
| Log(File.DirData("Ejemplo1")) | C:\Users\usuario\AppData\Roaming\Ejemplo1 |

Imagen 2. Métodos para directorios

Puedes combinar los métodos anteriores para crear una carpeta dentro de las anteriores. Por ejemplo:

```
Private strCarpeta As String = File.DirTemp & "tema17\"  
Log(strCarpeta)
```

Mostraría **C:\Users\usuario\AppData\Local\Temp\tema17**

Crear carpetas/directorios

Puedes crear una nueva carpeta con el método:

File.MakeDir (Padre As String, Dir)

```
File.MakeDir(File.DirTemp, "tema17")
```

Crea una carpeta llamada "tema17" dentro de la carpeta "C:\Users\usuario\AppData\Local\Temp\" del anterior ejemplo.

Comprobar la existencia de un fichero

Antes de usar un fichero es recomendable comprobar si existe. El método para ello es:

File.Exists (Dir As String, NombreFichero As String)

Que devuelve "True" o "False" según el fichero exista o no. Puedes usar una sentencia if para comprobar el resultado:

```
Private nf as String = "misdatos.txt"  
If File.Exists(File.DirTemp, nf) Then  
    Log("El fichero " & nf & " existe")  
Else  
    Log("No existe el fichero: " & nf)  
End If
```

Crear y escribir en un fichero

El método para crear un fichero es:

File.OpenOutput (Dir As String, NombreFi As String, Append As Boolean)

```
Private nf as String = "misdatos.txt"  
File.OpenOutput(File.DirTemp, nf, True)
```

Si el fichero no existe, se crea.

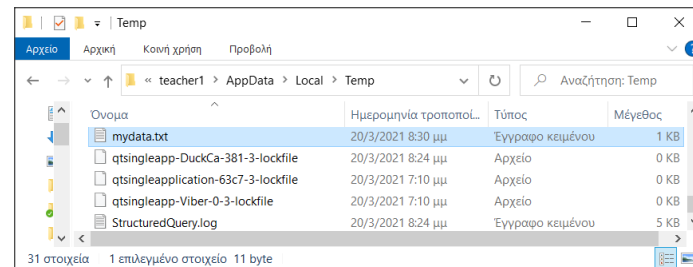


Imagen 3. Crear misdatos.txt dentro de la carpeta temporal

El valor lógico "True" o "False" del tercer parámetro indica si el fichero se abrirá para escritura (borrando todos sus datos anteriores) o si se abrirá para añadir datos al final del mismo.

Lectura y escritura de datos

Puedes leer o escribir desde una palabra a estructuras más complejas como listas o mapas.

Variables tipo cadena de texto

File.WriteString (Dir As String, NombreFichero As String, Texto As String)

```
Private nf as String = "misdatos.txt"  
Private msj As String = "Hola Mundo"  
File.WriteString(File.DirTemp, nf, msj)
```

Escribe una variable de tipo cadena de texto en el fichero. Fíjate en que el fichero se crea al principio, con lo que se borrarían todos sus datos.

Si el fichero ya existe, se puede leer su contenido en una variable de tipo cadena con el método:

File.ReadString (Dir As String, NombreFichero As String) As String

```
Private nf as String = "misdatos.txt"  
Private strContenidoFichero As String  
strContenidoFichero = File.ReadString(strCarpeta, nf)
```


Listas

Para escribir una lista en un fichero se usa el método:

File.WriteList(Dir As String, NombreFich As String, Lista As List)

```
File.WriteList(strCarpeta, " misdatos.txt", Lista)
```

Con el método anterior cada ítem de la lista se convierte en una cadena de texto y se añade una nueva fila al fichero.



Para leer una lista de un fichero se usa el método:

File.ReadList (Dir As String, NombreFichero As String)

```
Lista = File.ReadList(File.DirRootExternal, "misdatos.txt")
```

El método crea un nuevo ítem de la lista para cada línea del fichero

Mapas

Se puede escribir el contenido de un mapa con el método:

File.WriteMap(Dir As String, NombreFich As String, Mapa As Map)

```
File.WriteMap(File.DirInternal, "fichero.txt", mapa)
```

El uso más típico es para crear un fichero de configuración para el programa.

Un mapa se puede leer con el método:

ReadMap(Dir As String, NombreFichero As String)

```
mapa = File.ReadMap(File.DirInternal, " fichero.txt")
```

El orden de los ítems no será necesariamente el mismo que el que aparece en el fichero, pero eso no importa en un mapa.



Ejercicios

Se proporciona el programa “Tema17 – Ej1” que crea una lista de equipos de fútbol y una pequeña historia de ellos. Se pide:

1. Crear un mapa con una clave para el nombre del equipo y como valores su historia.
2. Crear un fichero que guarde el mapa anterior en el fichero “equipos.txt”
3. Crear una cadena de texto que contenga todas las claves y valores del mapa. Emplea **& CRLF** para añadir una nueva línea al final de cada línea.
4. Escribe la cadena generada en un nuevo fichero con el nombre “equipos2.txt”.
5. Abre los ficheros “equipos.txt” y “equipos2.txt” con un editor de textos (por ejemplo, Notepad++) para analizar las diferencias.