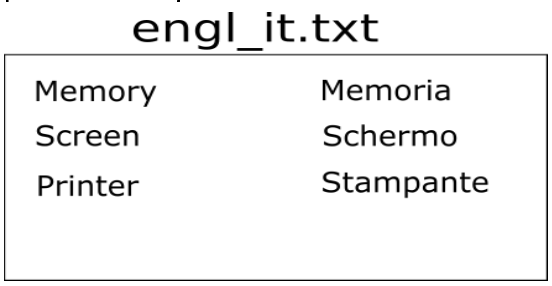
Lesson 17 – Files

⏱ **2h**

What students should know

- What is a file?
- File location in B4J
- File Methods

File we call a collection of data with similar content that is stored in the permanent space usually on the computer's hard disk. It is one of the most important features of programming languages since while all caching is done with central memory, after an application is completed its data should already be stored permanently.



engl_it.txt

Memory	Memoria
Screen	Schermo
Printer	Stampante

Picture 1 File engl_it.txt

Generally, the files are divided into databases and simple files which we will examine.

Storage Folders

Each operating system has many different folders to store applications data or other files. In order for B4J to work with files independently of the operating system, it uses keywords that refer to a specific type of folder.

File.DirAssets

Includes files contained in the application files folder that have been added by B4J file management during the development phase of an application. These files are read-only and no file can be added while the application is running. Files are usually written by the developer so that after installing the application they can be copied to other folders for use.

xui.DefaultFolder

Returns the folder where the application data is stored like File.DirData do. You are first required to call the SetDataFolder command once before using it.

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

File.DirData
Returns the folder where the application data is stored and is suitable for creating files and storingdata.

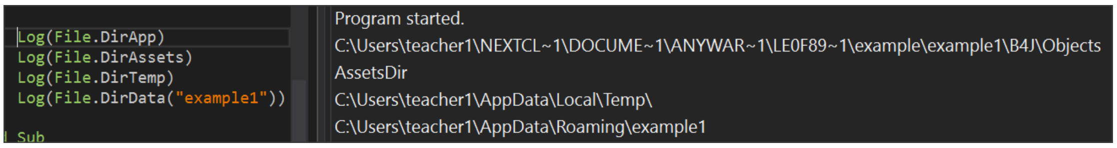
In a Windows environment, returns the "user data folder" that is usually in the path

C:\Users\[user name]\AppData\Roaming\[AppName]

On non-Windows operating systems, returns the folder where the application is installed

File.DirApp
Returns the folder where the application is installed. In Windows, this folder is usually Program Files and is read-only.

File.DirTemp
Returns the Temporary Files folder.



```
Log(File.DirApp)
Log(File.DirAssets)
Log(File.DirTemp)
Log(File.DirData("example1"))

Sub
```

```
Program started.
C:\Users\teacher1\NEXTCL~1\DOCUME~1\ANYWAR~1\LE0F89~1\example\example1\B4J\Objects
AssetsDir
C:\Users\teacher1\AppData\Local\Temp\
C:\Users\teacher1\AppData\Roaming\example1
```

Picture 2 Directories commands

You can use the above commands in combination to declare a new folder within the previous ones. For example,

```
Private strFolder  As String  =  File.DirTemp  & “lesson17\”
Log(strFolder)
```

Displays C:\Users\<*user name*>\AppData\Local\Temp\lesson17\

Create folders
You can create a new folder with the

File.MakeDir (Parent As String, Dir)

```
File.MakeDir(File. DirTemp, “lesson17")
```

Creates a folder named lesson17 within C:\Users\teacher1\AppData\Local\Temp\ of the previous example.

Check file existence
Before you use a file, usually the first task to do is to check for existence. The command is

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

And returns truth or lie that you can control it with a command if.

```
Private fn  as String  = “mydata.txt”
If File.Exists(File.DirTemp, fn)  Then
  Log("File " & fn & " Exists")
```

```
Else
  Log("There  is no File:  " & fn)
End If
```

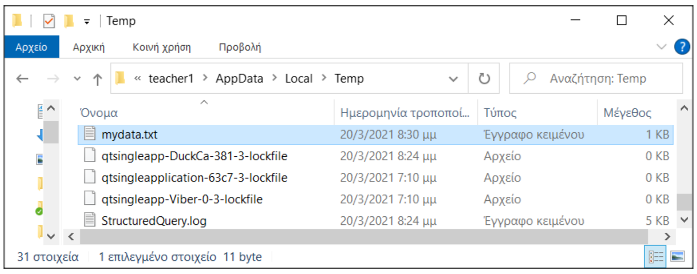
Create and write to a file

The command to create a file is:

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

```
Private fn  as String  = “mydata.txt”
File.OpenOutput(File.DirTemp, fn, True)
```

If the file does not exist then it is created.



Όνομα	Ημερομηνία τροποποί...	Τύπος	Μέγεθος
mydata.txt	20/3/2021 8:30 μμ	Έγγραφο κειμένου	1 KB
qtsingleapp-DuckCa-381-3-lockfile	20/3/2021 8:24 μμ	Αρχείο	0 KB
qtsingleapplication-63c7-3-lockfile	20/3/2021 7:10 μμ	Αρχείο	0 KB
qtsingleapp-Viber-0-3-lockfile	20/3/2021 7:10 μμ	Αρχείο	0 KB
StructuredQuery.log	20/3/2021 8:24 μμ	Έγγραφο κειμένου	5 KB

31 στοιχεία 1 επιλεγμένο στοιχείο 11 byte

Picture 3 mydata.txt inside temp folder

The logical value True or False in the third parameter refers to whether the file is opened for write by erasing the older data or will add to existing records.

Writing and reading data

You can write or read data from one word to more complex structures such as lists and maps.

String-type variables

File.WriteString (Dir As String, FileName As String, Text As String)

```
Private fn  as String  = “mydata.txt”
Private msg As String = "Hello World"
File.WriteString(File.DirTemp, fn, msg)
```

Writes a string variable to the file. Attention the file is created from the beginning even if it already existed. So, its data is deleted.

For a file that already exists you can read its data and transfer it to a string variable with the command

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

```
Private fn as String = “mydata.txt”
Private strFileContent As String
strFileContent = File.ReadString(strFolder, fn)
```

Lists

To write a list to a file, use the command:

File.WriteList(Dir As String, FileName As String, List As List)

```
File.WriteList(strFolder, "mydata.txt", List1)
```

With the above command each item in the list is converted to string and a new row is created in the file.

To read a list from a file, use the command:

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

```
List1 = File.ReadList(File.DirRootExternal, " mydata.txt")
```

The command creates a new item in the list per line of the file.

Maps

An entire map is recorded in a file with the command

File.WriteMap(Dir As String, FileName As String, Map As Map)

```
File.WriteMap(File.DirInternal, "file.txt",  map1)
```

The most common function is to create a configuration file for application.

A map is read with the command

ReadMap(Dir As String, FileName As String)

```
map1 = File.ReadMap(File.DirInternal, "file.txt")
```

The order of the items is not necessarily the same as the order of the file but this does not matter in a map.

Exercises

The Lesson17-ex1 program is given, which creates a list of football teams and a brief history of them.

1. Create a map with a key to the group name and values its history.
2. Create a file that stores the above map named teams.txt
3. Create a string from all map keys and values. Use & CRLF to add a new line at the end of each line
4. Write string in a new file with name teams2.txt
5. Open with an external editor (such as notepad++) teams.txt and teams2.txt and examine their differences.