

Programació



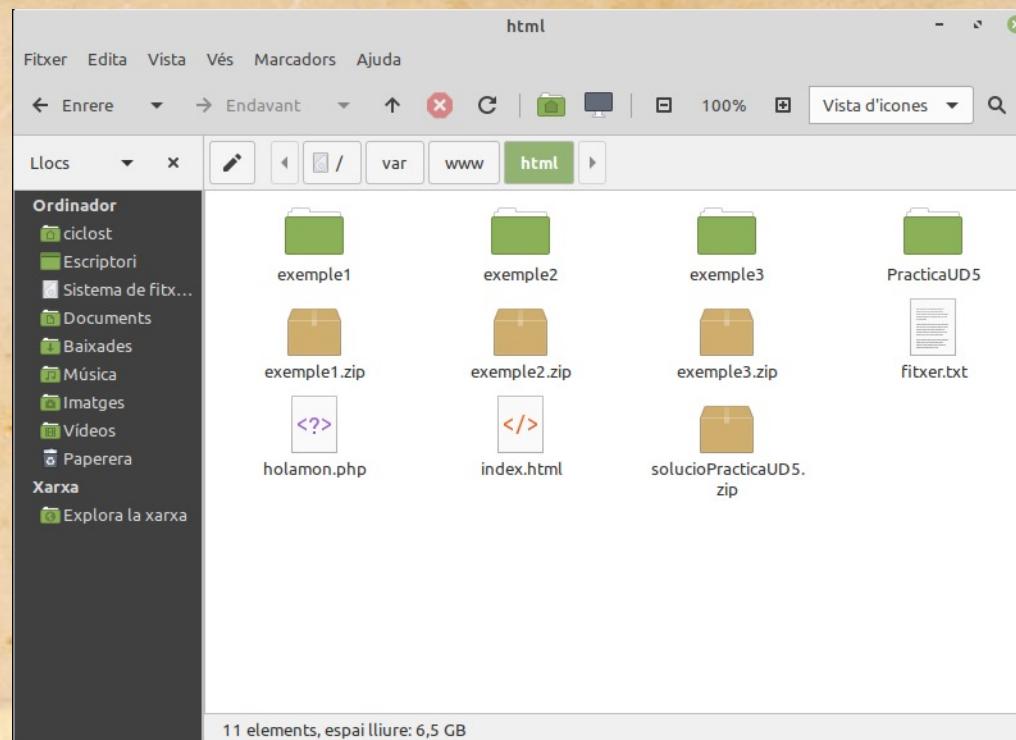
UT13.1 La classe File

Introducció

- Els nostres programes, manipulen i transformen dades, **mentre el programa està en execució.**
- Problemàtica a l'hora d'introduir dades cada vegada que el programa arranca.
- Els fitxers permeten emmagatzemar informació de forma permanent i així evitar la pèrdua de les dades que requereixen persistència.

Gestió de fitxers

- Entre les funcions d'un sistema operatiu, es troba la de poder gestionar sistemes de fitxers (carpetes i fitxers)



Gestió de fitxers

- L'accés a esta interfície de gestió de fitxers **no és únicament pel sistema operatiu.**
- De fet, el explorador, per exemple, no deixa de ser un programa que permet esta gestió, com altres programes també ho fan.
- Java no n'és una excepció, i ofereix una biblioteca (conjunt de classes amb els seus mètodes) per mitjà del paquet `java.io`

La classe File

- Peça fonamental per poder operar amb fitxers (fins JavaSE 6)
- Pertany al paqueta `java.io`
- Encara que s'anomene `File`, no es refereix exactament a un fitxer.
- La classe `File` indica una ruta dins del sistema de fitxers.
- **Permet fer operacions sobre fitxers i/o directoris.**

Classe File. Inicialització

```
File f = new File (String ruta)
```

- On "ruta", és la forma general d'un nom de fitxer o carpeta, de manera que identifica únicament la seu localització en el sistema de fitxers.
- En este nivell, la ruta **pot contenir elements existents o no**. La classe File s'inicialitzarà igualment.

Classe File. Definició de rutes

- Les rutes poden ser diferents segons el sistema operatiu.

- Ruta Unix: /usr/bin
- Ruta Windows: C:\Windows\System32

`File.separator;`

- La constant **File.separator** conté el caràcter separador de rutes, segons el SO), per això no hi ha que canviar el codi en funció del SO al qual executem l'aplicació (seria inviable).

Ruta absoluta

- És aquella que es refereix a un element a partir de la carpeta arrel d'un sistema de fitxers, enumerant **una per una les carpetes** fins aquella que conté l'element destinació.
- Problema: És dependent d'un sistema de fitxers concret.

C:\Fotos\Viatges (ruta a una carpeta)

Windows

Linux

/Fotos/Viatges (ruta a una carpeta)

Rutes relatives

- **Ruta relativa.** Aquella que comença des de la carpeta o directori de treball de l'aplicació.

```
File f = new File("exercicis/apartat1/activitats.txt");
```

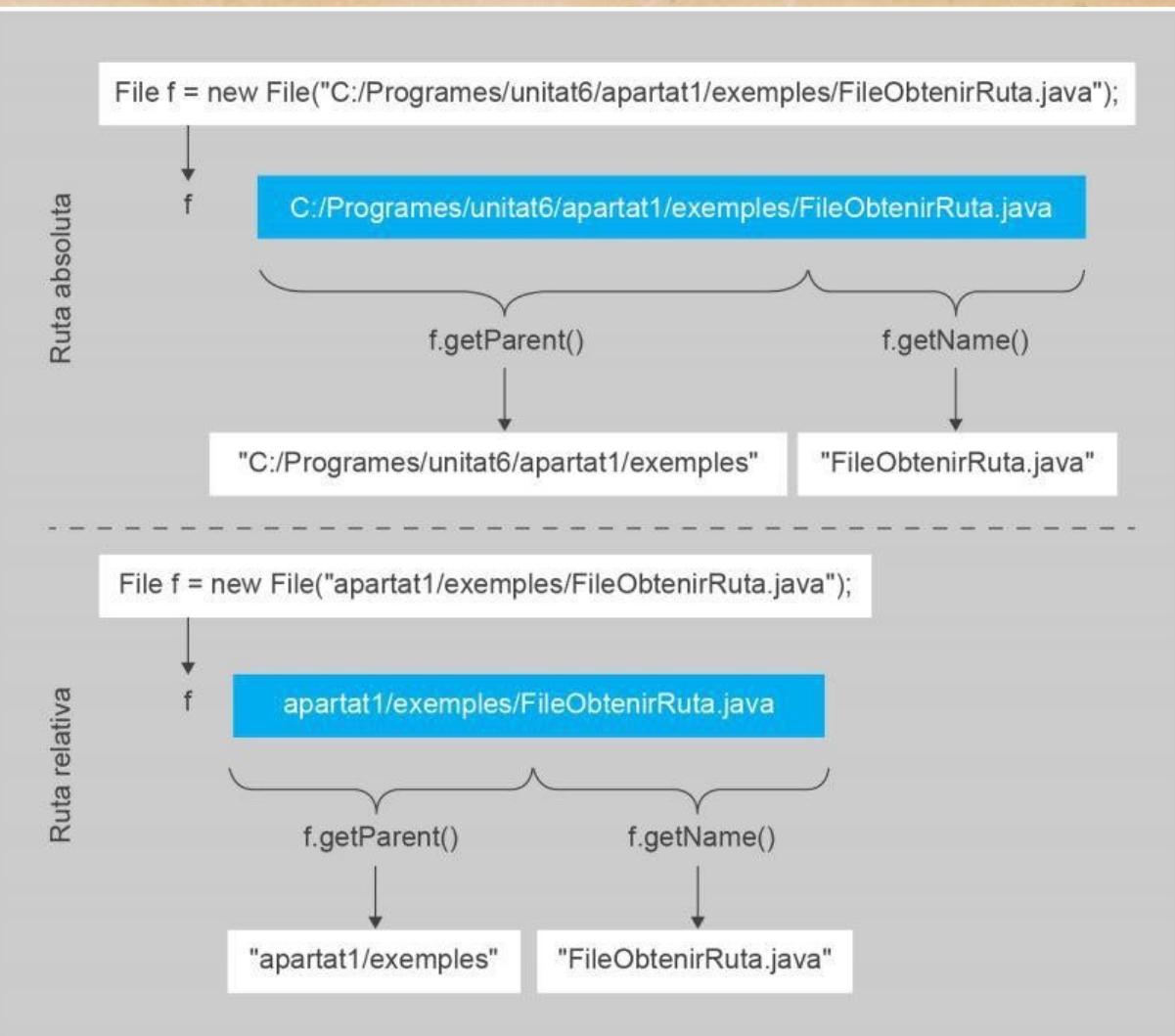
- La **carpeta de treball** sol ser la carpeta on estan ubicats els fitxers del projecte. Es pot obtindre de la següent manera:

```
//Assigna en forma de cadena de text la ruta de la carpeta de treball actual.  
String dirTreball = System.getProperty("user.dir");
```

Operacions (mètodes) típiques de la classe File

- String getParent () → obté la ruta pare de la ruta actual.
- String getName () → nom de la carpeta que conté l'element definit.
- String getAbsolutePath () → obté la ruta absoluta de l'element definit

Operacions bàsiques amb File



Pràctica 1

- Crea un projecte que continga un mètode main que inicialitze quatre variables de la classe `File`, utilitzant en cadascuna les següents rutes:
 - Ruta absoluta a la carpeta del teu usuari.
 - Ruta absoluta a un fitxer que estiga dins de la carpeta del teu usuari `prova.txt`
 - Ruta relativa a una carpeta anomenada `temp` dins de la carpeta arrel del teu usuari
 - Ruta relativa a un fitxer dins de la carpeta `temp`, anomenat `prova.txt`

Mètodes de comprovacions d'estat

- boolean exists() → comprova si existeix la ruta dins del sistema de fitxers
- boolean isFile() → comprova si la ruta es refereix a un fitxer
- boolean isDirectory() → comprova si la ruta es refereix a un directori

Pràctica 2

- Crea un mètode anomenat `mostrarEstat` que reba com a paràmetre una variable de tipus `File`. Este mètode deu mostrar per pantalla la informació que proporciona cadascun dels mètodes descrits anteriorment.
- Invoca el mètode utilitzant cadascuna de les variables de tipus `File` creades en la pràctica anterior.

Mètodes per a la lectura i modificació de propietats

- `long length()` → retorna la longitud del fitxer en bytes (només té sentit si la ruta es refereix a un fitxer)
- `long lastModified()` → torna la última data en que el fitxer va ser modificat (en mil·lisegons des del 01/06/1970)
 - Per a convertir a una data determinada podem utilitzar la classe DATE (consulta la seu API per a conèixer el seu maneig).

```
Date data = new Date(f.lastModified());
```

- Actualment és recomanable utilitzar la classe LocalDateTime de Java:

```
LocalDateTime novaData = LocalDateTime.ofInstant(data.toInstant(), ZoneId.systemDefault());
```

Pràctica 3

- Crea un mètode anomenat `proprietatsFitxer` que reba un paràmetre de tipus `File`.
- Mostra la informació que retornen els mètodes de la diapositiva anterior (mostra la data fent ús de la classe `LocalDateTime`)

Gestió de fitxers

- boolean mkdir() → permet crear un directori (baix certes condicions) d'acord amb la ubicació de la ruta proporcionada (retornarà true o false segons l'èxit o no de la creació).
- boolean delete() → elimina el fitxer o carpeta (retornarà true o false segons l'èxit o no de l'eliminació).

Pràctica 4

- Esbrina baix quines condicions el mètode `mkdir()` retornarà "false" (en quines condicions no pot crear el directori).
- Idem per el mètode `delete()`;
- Amplia el programa creat fins ara per a crear alguna carpeta i posteriorment esborrar-la.
- Intenta eliminar alguna carpeta que no siga possible eliminar, verificant que, efectivament, la carpeta no ha sigut eliminada.

Pràctica 5

- 1.Crea un programa que llija per teclat un text que corresponga a la ruta d'un fitxer existent en tel teu ordinador (mostra un missatge d'error si no existeix).
- 2.Canvia el nom d'eixe fitxer: elimina la seu extensió (per exemple, si el fitxer és document.txt, passarà a anomenar-se document).
El fitxer amb extensió haurà ser eliminat.

Listado

- File[] listFiles() → retorna un array amb tots els elements continguts en la ruta indicada

```
File[] arrayElements = carpeta.listFiles();
System.out.println("El contingut de " + carpeta.getAbsolutePath() + " és:")
;
//Per recórrer un array, s'usa un bucle
for (int i = 0; i < arrayElements.length; i++) {
    File f = arrayElements[i];
    if (f.isDirectory()) {
        System.out.print("[DIR] ");
    } else {
        System.out.print("[FIT] ");
    }
    System.out.println(f.getName());
}
```

Pràctica 6

- Comprova quin valor retorna el mètode presentat a les diapositives anteriors en cas que la ruta estiga referida a un fitxer.
- Comprova també quin valor retorna en cas que la ruta estiga referida a un directori buit (sense elements).

Pràctica 7

1. Fent ús de l'explorador, crea una carpeta que continga al seu interior, carpetes i fitxers. Cada carpeta creada deu contindre fitxers.
2. Realitza un programa que llija per teclat la ruta de la carpeta creada en el punt 1, esborrant tots els elements que hi ha dins d'eixa carpeta.

API de File (I)

java.io.File

+File(pathname: String)	
+File(parent: String, child: String)	
+File(parent: File, child: String)	
+exists(): boolean	
+canRead(): boolean	
+canWrite(): boolean	
+isDirectory(): boolean	
+isFile(): boolean	
+isAbsolute(): boolean	
+isHidden(): boolean	
+getAbsolutePath(): String	
+getCanonicalPath(): String	

Creates a File object for the specified path name. The path name may be a directory or a file.	
Creates a File object for the child under the directory parent. The child may be a file name or a subdirectory.	
Creates a File object for the child under the directory parent. The parent is a File object. In the preceding constructor, the parent is a string.	
Returns true if the file or the directory represented by the File object exists.	
Returns true if the file represented by the File object exists and can be read.	
Returns true if the file represented by the File object exists and can be written.	
Returns true if the File object represents a directory.	
Returns true if the File object represents a file.	
Returns true if the File object is created using an absolute path name.	
Returns true if the file represented in the File object is hidden. The exact definition of <i>hidden</i> is system-dependent. On Windows, you can mark a file hidden in the File Properties dialog box. On Unix systems, a file is hidden if its name begins with a period(.) character.	
Returns the complete absolute file or directory name represented by the File object.	
Returns the same as getAbsolutePath() except that it removes redundant names, such as "." and "..", from the path name, resolves symbolic links (on Unix), and converts drive letters to standard uppercase (on Windows).	

API de File (II)

```
+getName(): String  
  
+getPath(): String  
  
+getParent(): String  
  
+lastModified(): long  
+length(): long  
+listFile(): File[]  
+delete(): boolean  
  
+renameTo(dest: File): boolean  
  
+mkdir(): boolean  
  
+mkdirs(): boolean
```

Returns the last name of the complete directory and file name represented by the `File` object. For example, `new File("c:\\book\\test.dat").getName()` returns `test.dat`.

Returns the complete directory and file name represented by the `File` object.
For example, `new File("c:\\book\\test.dat").getPath()` returns `c:\\book\\test.dat`.

Returns the complete parent directory of the current directory or the file represented by the `File` object. For example, `new File("c:\\book\\test.dat").getParent()` returns `c:\\book`.

Returns the time that the file was last modified.

Returns the size of the file, or 0 if it does not exist or if it is a directory.

Returns the files under the directory for a directory `File` object.

Deletes the file or directory represented by this `File` object. The method returns true if the deletion succeeds.

Renames the file or directory represented by this `File` object to the specified name represented in `dest`. The method returns true if the operation succeeds.

Creates a directory represented in this `File` object. Returns true if the the directory is created successfully.

Same as `mkdir()` except that it creates directory along with its parent directories if the parent directories do not exist.