

Iniciació Persistència de fitxers

START --- Preparar NetBeans

1. Obrir NetBeans → **File** → **New Project** → **Java with Ant** → **Java Application** (o Maven si prefereixes).
2. Dona-li un nom (per exemple Projecte1) i crea **un package** (per exemple **Paquet1**)

L'últim pas es crear classes noves amb New → Java Class. (per exemple clase1)

3. Quan executis, el directori de treball (on es creen fitxers amb rutes relatives) sol ser `System.getProperty("user.dir")`
- Crearem una classe simple que imprimirà el directori de treball i llistarà els fitxers (això ens servirà per entendre on es creen els fitxers i com executar un main).

STOP

Recordatori (Conceptes bàsics de Java)

1. Què és un main?

En Java, el **punt d'entrada** d'un programa és un mètode especial anomenat main.
Té aquesta forma exacta:

```
public static void main(String[] args) {  
    // codi a executar  
}
```

Això és com la porta d'entrada: quan li dius a Java "executa aquest programa", començarà sempre per aquest main.

2. Què és una classe?

Una classe és un fitxer .java amb codi que pot contenir:

- **Atributs (variables),**
- **Mètodes (funcions),**
- **I, si volem que es pugui executar, un main.**

Exemple de classe amb main:

```
public class Prova {  
    public static void main(String[] args) {  
        System.out.println("Hola Bemen3!");  
    }  
}
```

RUN --- Crear la classe que llista el directori (VeureDir)

Ara crearem la classe VeureDir amb un main. Pots:

- (NetBeans GUI): clic dret sobre el package Paquet1 → New → Java Class → Name: VeureDir → Finish.

Copia i enganxa aquest codi dins VeureDir.java:

```
package Paquet1;  
  
import java.io.File;  
  
/**  
 * Exemple senzill per veure el directori de treball i llistar fitxers.  
 */  
public class VeureDir {  
    public static void main(String[] args) {  
        // Aquesta línia mostra el directori on Java està executant el  
        programa.  
        // És útil per saber on es crearàn fitxers amb rutes relatives.
```

```
System.out.println("Directori de treball (user.dir): " +
System.getProperty("user.dir"));

// Cream un objecte File que representa el directori actual "."
File directori = new File(".");

// list() retorna un array de noms (String) o null si falla.
String[] llista = directori.list();

if (llista != null) {
    System.out.println("Contingut del directori:");
    for (String nom : llista) {
        System.out.println("  " + nom);
    }
} else {
    System.out.println("No s'ha pogut llistar el directori.");
}
}
```

RUN — Executar la classe a NetBeans

1. Obre VeureDir.java a l'editor.
2. Clic dret a dins del fitxer → Run File (o prem Shift+F6).
3. Mira la finestra Output de NetBeans; allà veuràs el text que imprimeix el System.out.println.

Ara farem una nova classe que escrigui sobre un fitxer que es digui exemple.txt

On es crearà exemple.txt si escrivim un fitxer amb new File("exemple.txt")?

En el mateix directori que mostra System.getProperty("user.dir") (normalment la carpeta del projecte, o la carpeta target depenent de com NetBeans executi el projecte).

RUN — Exemple pràctic següent: escriure i llegir un fitxer de text

Quan ja funcioni el VeureDir, copia aquesta nova classe TextFileExample (a Paquet1):

```
package Paquet1;

import java.io.*;

/**
 * Escriu dues línies a "exemple.txt" (afegint, append) i després les llegeix.
 */
public class TextFileExample {
    public static void main(String[] args) {
        File file = new File("exemple.txt"); // ruta relativa, veure user.dir

        // --- Escriure (append = true) amb try-with-resources per tancar
        automàticament ---
        try (PrintWriter pw = new PrintWriter(new BufferedWriter(new
        FileWriter(file, true)))) {
            pw.println("Alumne1,8.5");
            pw.println("Alumne2,7.0");
            System.out.println("S'han escrit línies a: " +
            file.getAbsolutePath());
        } catch (IOException e) {
            System.err.println("Error escrivint: " + e.getMessage());
        }

        // --- Llegir el fitxer i mostrar per pantalla ---
        try (BufferedReader br = new BufferedReader(new FileReader(file))) {
            System.out.println("Llegint " + file.getAbsolutePath() + ":");
            String linea;
            while ((linea = br.readLine()) != null) {
                System.out.println("  " + linea);
            }
        } catch (IOException e) {
```

```
        System.err.println("Error llegint: " + e.getMessage());  
    }  
}  
}
```

Punts interessants:

- `FileWriter(file, true)` obre en mode append (afegeix al final).
 - `try (...) { }` tanca automàticament els fluxos al final (important per no perdre dades).
 - `PrintWriter` i `BufferedReader` són una combinació fàcil per escriure/llegir text línia a línia.
-
1. Executa `TextFileExample.java` per veure que `exemple.txt` s'ha creat i es llegeix.
 2. Si vols, obre la carpeta del projecte amb l'explorador de fitxers i comprova que `exemple.txt` existeix a la ruta que va mostrar `user.dir`.

Nivells de persistència que aprendrem al mòdul

El .txt que hem fet és el nivell bàsic. Però hi ha més opcions que veurem al llarg del curs:

1. Fitxers de text
 - Exemples: .txt, .csv (com el que hem escrit amb Alumne1,8.5).
 - Molt humans de llegir i editar, però no tan eficients per a grans volums.
2. Fitxers binaris
 - Guardes dades en format compacte (números, booleans, etc.) amb `DataOutputStream`.
 - Millor per a programes, no tant per a humans.
3. Serialització d'objectes
 - Guardar directament objectes Java (Alumne, Persona, etc.) en un fitxer amb `ObjectOutputStream`.
 - Quan el tornes a llegir, recuperes els objectes tal com eren.
4. Accés aleatori (`RandomAccessFile`)
 - Et permet saltar a un registre concret dins d'un fitxer gran (com un catàleg).
 - Útil quan tens molts registres i no vols llegir-ho tot cada vegada.
5. Fitxers XML / JSON
 - Formats estructurats i estàndard, molt usats per intercanvi de dades entre programes.

STOP

Mini-gestor d'alumnes amb CSV (fitxer de text amb camps separats per comes).



1. Crear un fitxer alumnes.csv si no existeix.
2. Afegir alumnes al fitxer (nom,nota).
3. Llegir tots els alumnes del fitxer i mostrar-los.

Exemple de codi: GestorCSV.java

Crea una nova classe dins el mateix i enganxa:

```
package Paquet1;

import java.io.*;
import java.util.Scanner;

public class GestorCSV {

    private static final File FITXER = new File("alumnes.csv");

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int opcio;

        do {

            System.out.println("\n--- GESTOR D'ALUMNES (CSV) ---");

            System.out.println("1. Afegir alumne");

            System.out.println("2. Llistar alumnes");

            System.out.println("0. Sortir");

            System.out.print("Opció: ");

            opcio = sc.nextInt();

            sc.nextLine(); // netejar buffer
```

```
        switch (opcio) {
            case 1 -> afegirAlumne(sc);
            case 2 -> llistarAlumnes();
            case 0 -> System.out.println("Sortint...");
            default -> System.out.println("Opció no vàlida");
        }
    } while (opcio != 0);

    sc.close();
}

private static void afegirAlumne(Scanner sc) {
    System.out.print("Nom: ");
    String nom = sc.nextLine();
    System.out.print("Nota: ");
    double nota = sc.nextDouble();
    sc.nextLine();

    try (PrintWriter pw = new PrintWriter(new BufferedWriter(new
    FileWriter(FITXER, true)))) {
        pw.println(nom + "," + nota);
        System.out.println("Alumne afegit correctament!");
    } catch (IOException e) {
        System.err.println("Error escrivint: " + e.getMessage());
    }
}

private static void llistarAlumnes() {
    if (!FITXER.exists()) {
        System.out.println("Encara no hi ha alumnes.");
        return;
    }

    try (BufferedReader br = new BufferedReader(new FileReader(FITXER))) {
        String linia;
```



```
System.out.println("\n--- LLISTA D'ALUMNES ---");

while ((linia = br.readLine()) != null) {

    String[] parts = linia.split(",");

    String nom = parts[0];

    double nota = Double.parseDouble(parts[1]);

    System.out.println("Nom: " + nom + " | Nota: " + nota);

}

} catch (IOException e) {

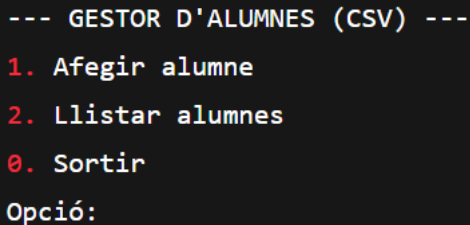
    System.err.println("Error llegint: " + e.getMessage());

}

}
```

Com provar-ho

1. Obre GestorCSV.java.
2. Clic dret → Run File.
3. A la consola (Output) et sortirà el menú:



```
--- GESTOR D'ALUMNES (CSV) ---
1. Afegir alumne
2. Llistar alumnes
0. Sortir
Opció:
```

Escriu 1 → posa un nom i una nota. → Es crea (o s'afegeix a) alumnes.csv.

Escriu 2 → et mostrarà la llista llegida del fitxer.

Pots repetir afegint més alumnes.

Escriu 0 → sortir del programa.

Què hem après:

- Afegir dades persistents (queda a alumnes.csv).
- Llegir i processar dades (parsejar el .csv amb `split(",")`).
- Fer un petit menú interactiu amb Scanner.

Exercicis proposats

Intenta modificar el gestor que hem fet per incorporar noves funcionalitats.

Exercici 1 — Afegir alumnes

Modifica el programa perquè, a més de nom i nota, també guardi:
el cognom i l'edat de l'alumne.

Exercici 2 — Llistar alumnes amb format

Quan es mostrin els alumnes, fes que surti en un format més bonic


Exercici 3 — Afegir una nova opció al menú que calculi la Mitjana de la nota.

Exercici 4 — Afegeix una nova opció al menú per Cercar alumne

Exercici 5 — Mostrar si està Aprovat o suspès

Exercici 6 (requereix un nivell més avançat) — Eliminar un alumne

Bonus — Ordenar per nota



No et preocupis
sino surt!! Estàs
aprenent!!!