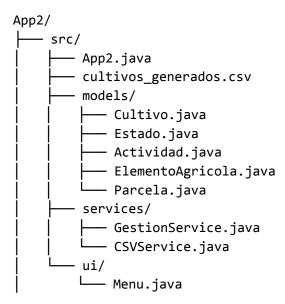
Informe de Diseño - Tarea 2: Gestión Agrícola en Java

Objetivo general

Desarrollar un programa en Java aplicando los principios fundamentales de la Programación Orientada a Objetos (POO), tales como herencia, encapsulamiento, abstracción, composición, uso de interfaces y organización en paquetes, integrando además mecanismos de persistencia de datos mediante archivos CSV.

Estructura del Proyecto



Para mantener el código ordenado y fácil de entender, el proyecto se dividió en varios paquetes. Cada uno agrupa clases que cumplen una función específica dentro del programa.

Paquetes principales

- **models:** Contiene las clases que representan los objetos principales del programa, como Cultivo, Parcela, Actividad, Elemento Agrícola y Estado.
- **services:** Contiene el código que permite hacer funcionar el programa. Por ejemplo, cómo se gestionan las parcelas, como se leen y guardan datos en archivo CSV, etc.
- **ui:** Se encarga de mostrar menús al usuario y recibir lo que escribe por consola. Es la parte del programa que interactúa directamente con quien lo usa.

- App2: Clase principal. Punto de entrada del programa, inicia el menú y carga el CSV.
- **cultivos_generados.csv:** Archivo con los datos iniciales que se procesan y actualizan.

Clases principales

- Cultivo, Parcela y Actividad: modelan los elementos agrícolas.
- **Elemento Agrícola:** Clase abstracta base para los elementos con atributos comunes como nombre, fecha y código.
- **Estado:** Enum que define los posibles estados de un cultivo.
- **Gestion Service:** Clase central que maneja la lógica de operaciones: agregar, eliminar, buscar y listar elementos.
- **CSV Service:** Encargado de la lectura y escritura del archivo CSV que contiene los datos persistentes.
- **Menú:** Clase que maneja la interacción con el usuario, mostrando opciones y llamando a los métodos correspondientes de Gestion Service.

Relaciones entre clases

Composición:

Una **Parcela** contiene múltiples **Cultivo**. Esta es una relación fuerte de composición, ya que los cultivos existen dentro del contexto de una parcela específica.

Agregación:

Un **Cultivo** puede tener múltiples **Actividad**. Las actividades están asociadas a un cultivo específico, pero pueden existir de forma más independiente (relación más débil que la composición).

Herencia:

Todas las clases **Cultivo**, **Parcela y Actividad** heredan de **ElementoAgricola**, lo que permite una estructura reutilizable para futuros elementos agrícolas.

Uso de Colecciones

Se utilizaron estructuras de datos de la biblioteca estándar de Java para optimizar la manipulación de información:

- **ArrayList:** Se emplea para almacenar listas ordenadas de Cultivo y Actividad, ya que permite recorrer, agregar y eliminar elementos de forma dinámica y eficiente.
- **HashMap:** Se utiliza en la clase Gestion Service para mantener un acceso rápido y directo a las Parcela mediante su código único, facilitando operaciones como búsquedas, actualizaciones o eliminaciones.

Estas colecciones permiten implementar funcionalidades complejas como ordenamientos, filtrado y búsquedas de forma sencilla y eficiente.

Modificadores de Acceso

Para proteger los datos y organizar mejor el código, se siguieron las siguientes prácticas:

Los atributos de las clases están marcados como **private**, lo que significa que solo pueden ser usados dentro de su propia clase.

Para acceder o cambiar esos datos desde otras clases, se usan métodos públicos llamados **getters** (para obtener el valor) y **setters** (para modificarlo).

Los métodos importantes que realizan acciones principales (como agregar un cultivo o buscar una parcela) son **public**, así pueden ser usados desde otras partes del programa.

Algunos métodos pequeños o auxiliares, como **stripQuotes** (que limpia las comillas de los textos del CSV), son **private**, ya que solo se usan dentro de la misma clase.

Modelo de Datos

La jerarquía de clases se diseñó para favorecer la reutilización de atributos comunes mediante una clase abstracta Elemento Agrícola.

Esto permite que **Cultivo, Parcela y Actividad** compartan una base común sin duplicar código.

```
public abstract class ElementoAgricola {
  private String nombre;
  private LocalDate fecha;
  // Métodos comunes: getters, setters, toString()
}
```

Reflexiones Finales

Uno de los principales desafíos fue el diseño inicial de las clases, especialmente al decidir qué atributos y métodos incluir en cada una y cómo distribuir correctamente las responsabilidades. Evitar la duplicación de código y mantener una estructura clara y coherente entre cultivos, parcelas y actividades fue clave para lograr un diseño limpio y funcional.

Para el manejo del archivo CSV se desarrolló la clase **CSVService**, la cual encapsula toda la lógica de lectura y escritura. Se tuvo especial cuidado con el tratamiento de cadenas entre comillas y la correcta interpretación de listas de actividades dentro de los campos, asegurando una restauración fiel de los datos.

A lo largo del proyecto se reforzaron conceptos fundamentales como la encapsulación de atributos mediante modificadores de acceso y el uso de **getters y setters**. Además, se aplicó el uso de **Map** para optimizar búsquedas por clave, se profundizó en el uso de **Scanner** para la interacción por consola, y se adquirió experiencia en trabajo colaborativo mediante Git.

Uso de Herramientas de IA

Durante el desarrollo del proyecto se utilizó la herramienta ChatGPT como apoyo en distintas etapas. Su uso permitió proponer una estructura de clases clara y coherente, resolver errores relacionados con la lectura de archivos CSV y se recibieron sugerencias sobre la documentación del código.

Todas las respuestas generadas por la IA fueron revisadas por el equipo y adaptadas según los requerimientos específicos del proyecto. Asimismo, se realizaron pruebas manuales constantes en terminal para garantizar que el programa funcione correctamente.

Integrantes

Nombre Integrante 1 – Cristóbal Meza Palacios crmeza@alumnos.uai.cl

Nombre Integrante 2 – Vasco Vasquez Ramirez vasvasquez@alumnos.uai.cl

Nombre Integrante 3 – Daniela Cuminao Embry dcuminao@alumnos.uai.cl

Nombre Integrante 4 – Benjamin Garcia Muñoz benjamigarcia@alumnos.uai.cl