Documentación Técnica - Simulador de Isla

1. Introducción

Este documento describe la arquitectura, componentes y funcionamiento del simulador de isla desarrollado en Java. El proyecto ha sido reorganizado en paquetes para mejorar la modularidad y el mantenimiento del código. Cada clase y paquete tiene un rol específico dentro del sistema, y se emplean conceptos de Programación Orientada a Objetos (POO), como herencia, polimorfismo y encapsulamiento.

2. Estructura de Paquetes

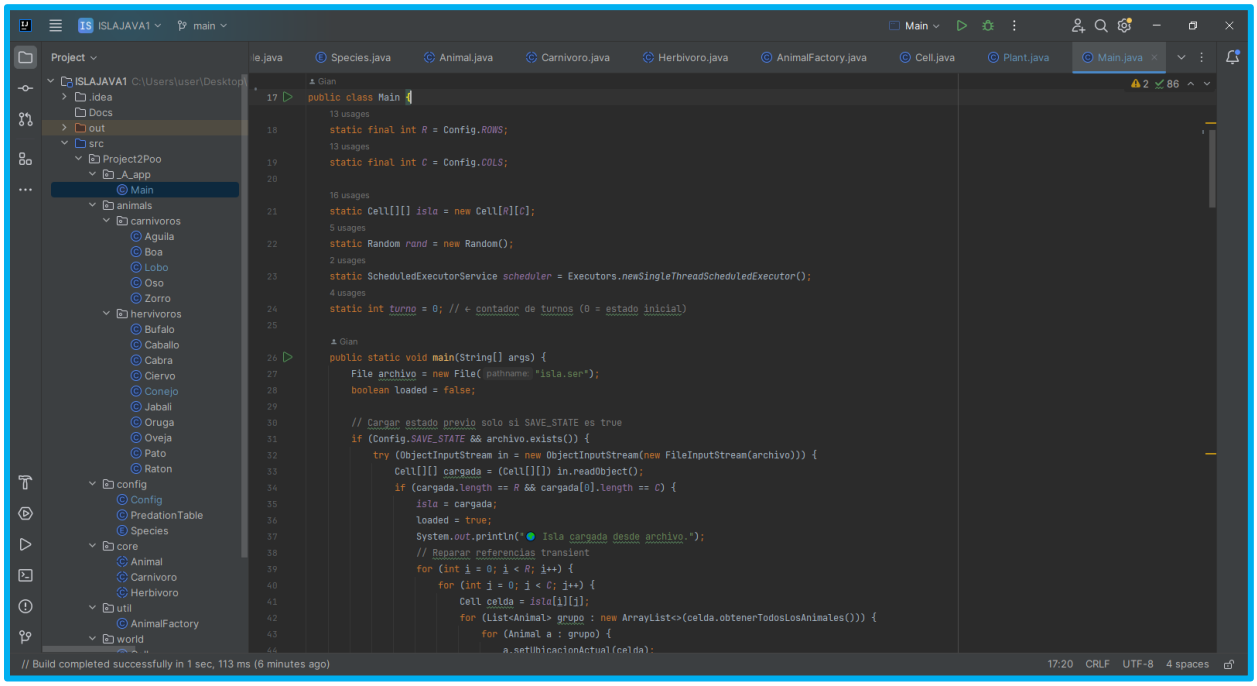
La organización de paquetes es la siguiente:
• Project2Poo._A_app → Contiene la clase Main, punto de entrada del programa.
• Project2Poo.config → Configuración general, tabla de depredación y enumeración de especies.
• Project2Poo.core → Clases base (Animal, Carnivoro, Herbivoro) y lógica común.
• Project2Poo.animals.carnivoros → Clases de animales carnívoros.
• Project2Poo.animals.herbivoros → Clases de animales herbívoros.
• Project2Poo.util → Clase AnimalFactory para creación de instancias.
• Project2Poo.world → Clases que representan celdas y plantas.

3. Clases Principales

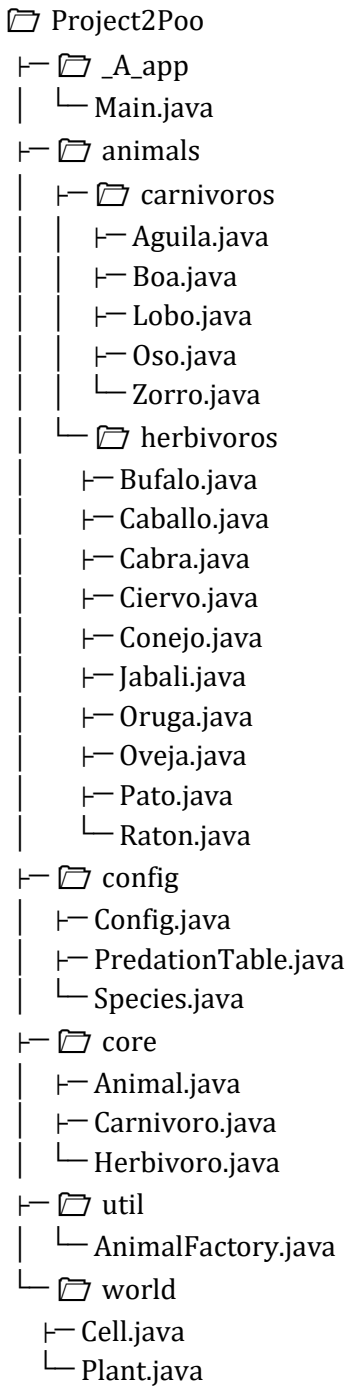
• Main: Gestiona el ciclo de simulación, inicializa la isla y coordina la ejecución.
• Config: Define parámetros globales como tamaño de la isla, probabilidades y población inicial.
• Species: Enumeración con las especies del sistema.
• PredationTable: Define las relaciones depredador-presa y probabilidades de caza.
• Animal: Clase abstracta base para todos los animales.
• Carnivoro / Herbivoro: Clases abstractas derivadas que definen comportamiento alimenticio.
• Cell: Representa una celda de la isla, puede contener animales y plantas.
• Plant: Representa una planta dentro de la isla.
• AnimalFactory: Crea instancias de animales según la especie indicada.

4. Características Técnicas

• Lenguaje: Java 17
• Uso de paquetes para modularidad.
• Uso de Programación Orientada a Objetos (herencia, polimorfismo, encapsulamiento).
• Uso de hilos para simular el ciclo de reloj.
• Guardado y carga opcional del estado de la isla.
• Migración de animales con probabilidades configurables.
• Mecanismo de muerte por hambre con probabilidad configurable.
• Estadísticas de población en cada turno.

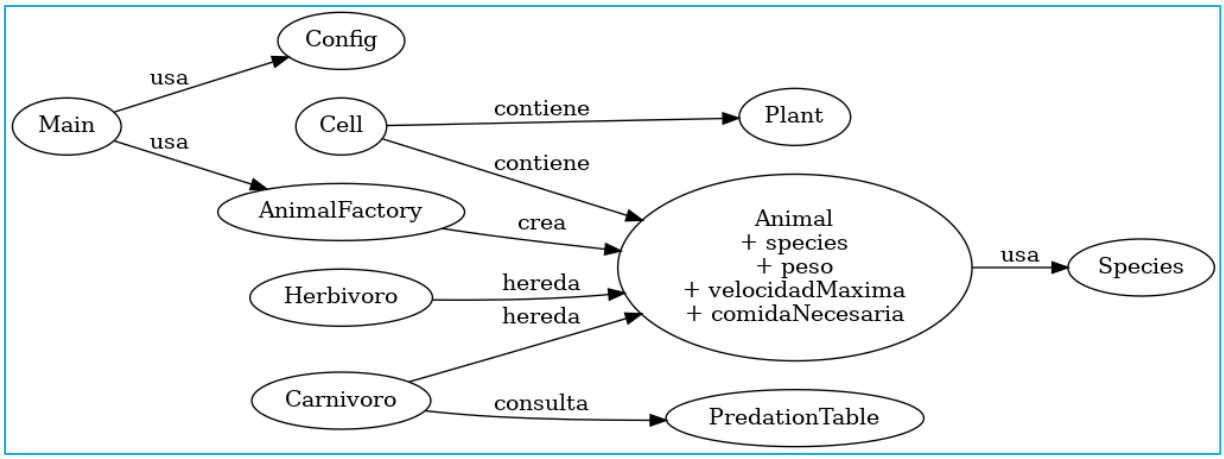


5. Estructura de Archivos del Proyecto



6. Diagrama UML

A continuación, se muestra el diagrama UML simplificado del sistema:



7. Conclusiones

La reorganización en paquetes y la definición clara de visibilidades han permitido que el código sea más mantenible y escalable. El sistema cumple con los requisitos del documento; incluyendo la configuración de parámetros, control de población, migración, y la posibilidad de detener la simulación cuando no quedan animales.