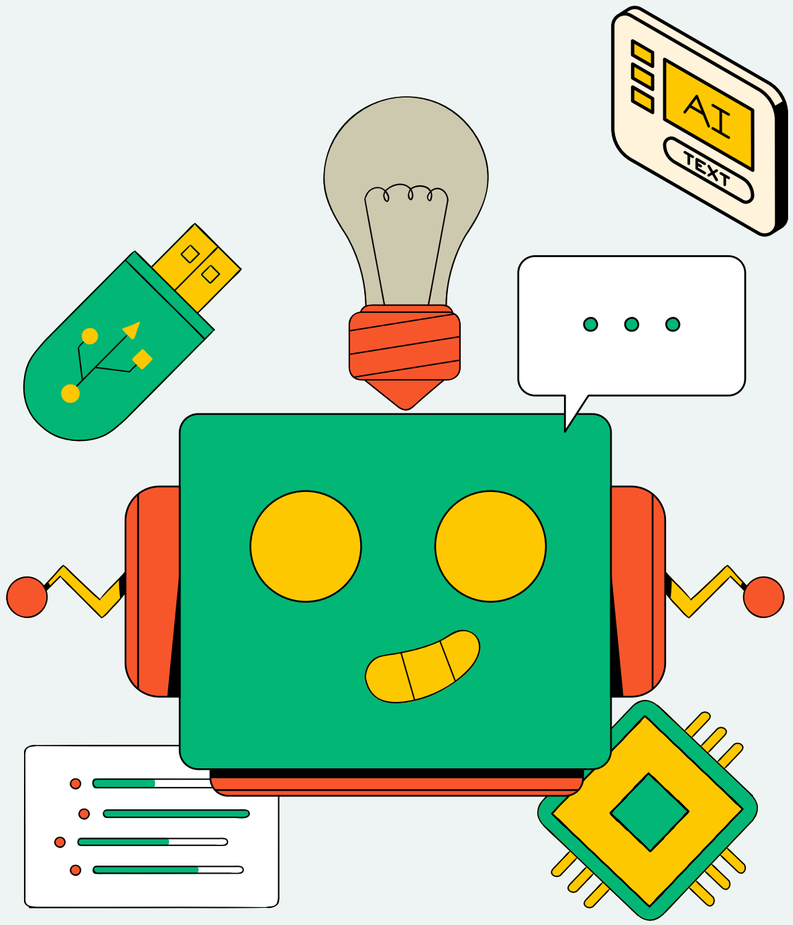
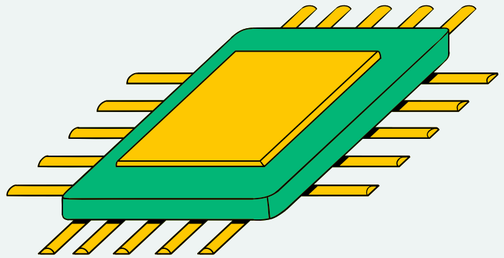


PATRÓN DE DISEÑO: ITERADOR

LABORATORIO 3

JAVIER CRUZ

DISEÑO DE SOFTWARE



DEFINICIÓN

Iterator es un patrón de diseño de comportamiento que permite recorrer los elementos de una colección sin exponer su representación

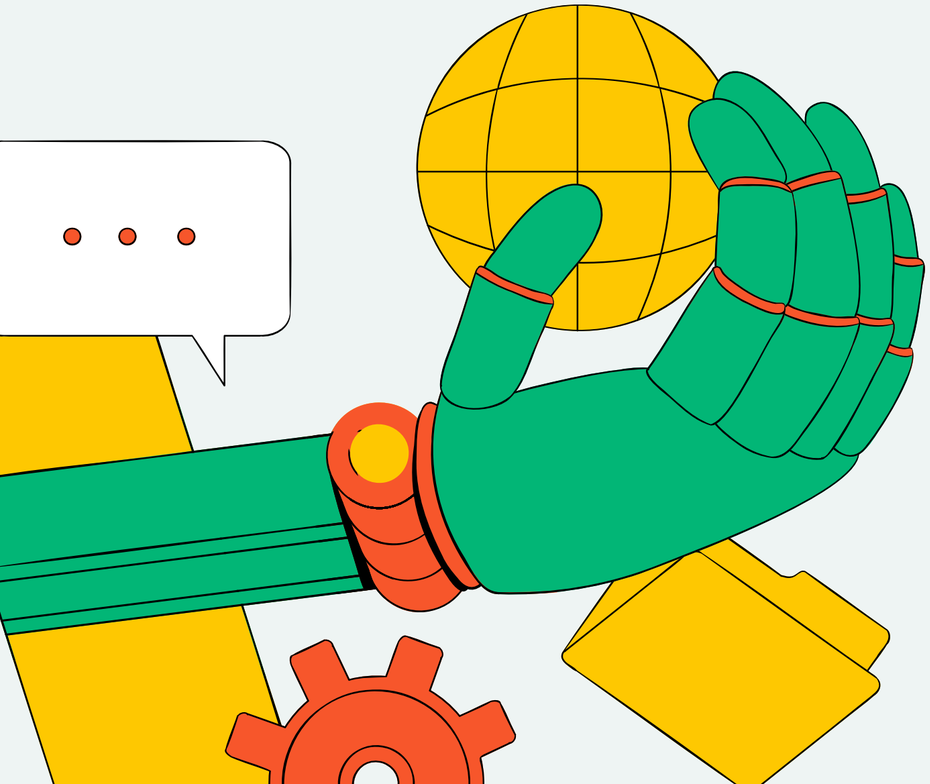
Algunos ejemplos de colección son:

- Listas
- Pilas
- Árboles

PROBLEMA QUE RESUELVE

- Los conjuntos son comunes, pero acceder a sus elementos puede ser complicado en estructuras complejas.
- En listas simples el recorrido es fácil, pero en árboles se requieren algoritmos específicos.
- Varios algoritmos de recorrido pueden afectar la eficiencia del código.
- Las colecciones tienen diferentes métodos de acceso, lo que requiere adaptar el código.

SOLUCIÓN AL PROBLEMA



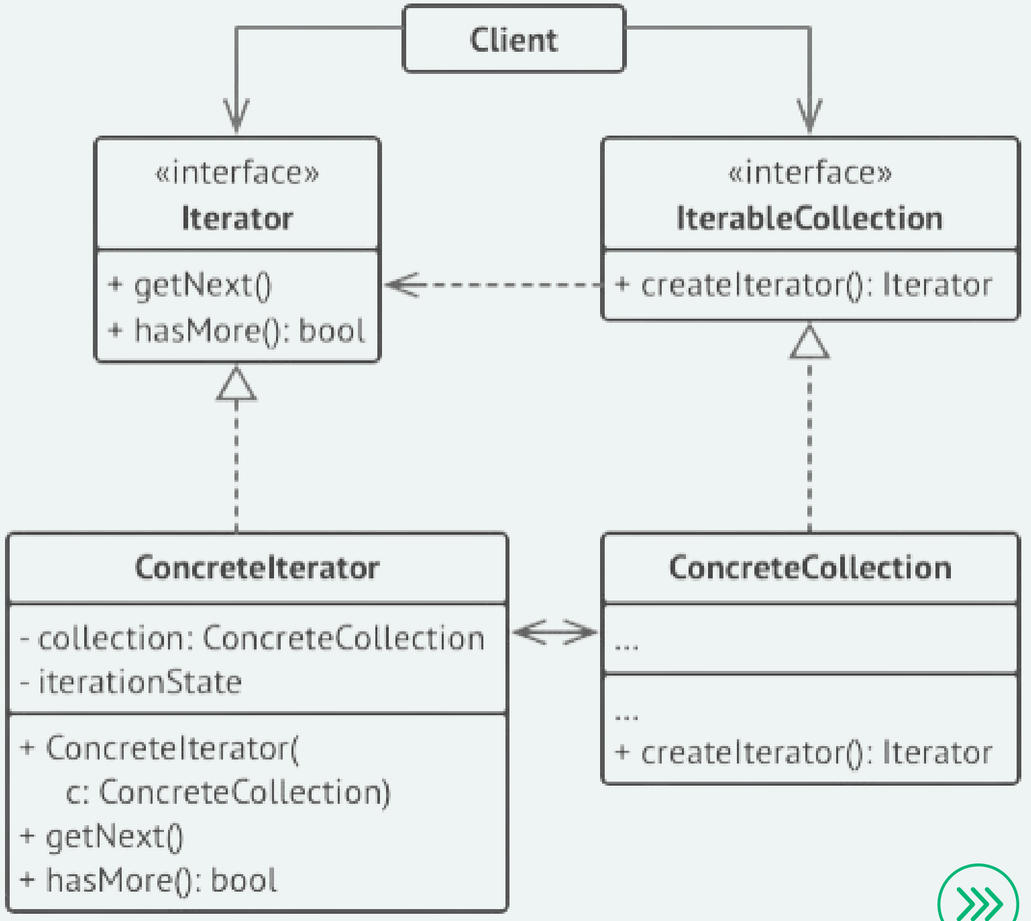
- El patrón Iterator separa el comportamiento de recorrido de una colección en un objeto llamado iterador.
- El iterador maneja detalles como la posición actual y la cantidad de elementos restantes.
- Todos los iteradores implementan la misma interfaz, lo que permite compatibilidad con diferentes colecciones y algoritmos.

EJEMPLOS REALES DE ITERATOR

Conocer una nueva ciudad

Escuchar una playlist

ESTRUCTURA DEL PATRÓN ITERATOR

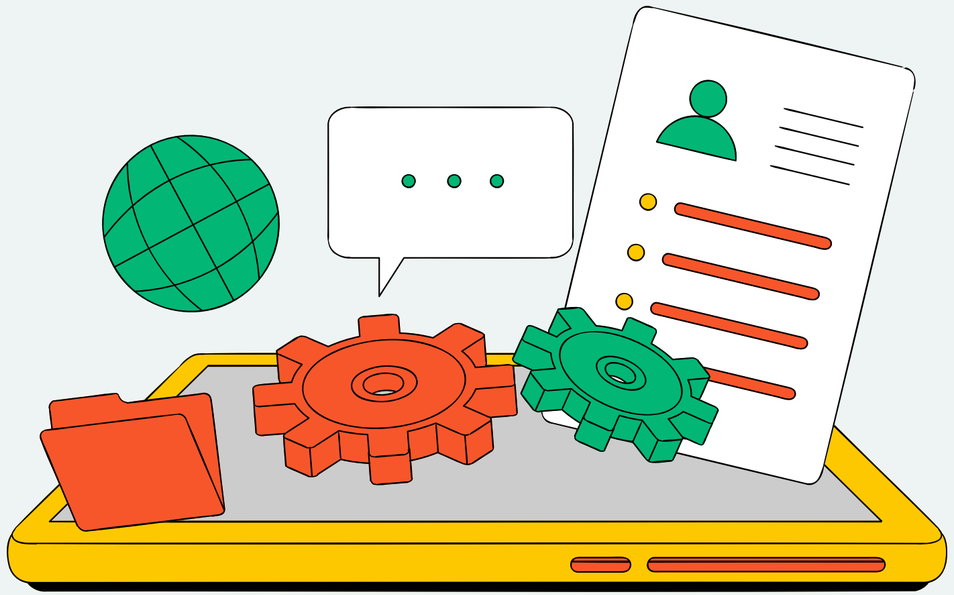


¿CÓMO SE MEJORA EL MANTENIMIENTO Y ESCALABILIDAD

Separación de responsabilidades

Reduce la duplicación de código

Facilita la escalabilidad



VENTAJAS Y DESVENTAJAS

Ventajas:

- **Single Responsibility Principle:** Permite limpiar el código al extraer algoritmos de reocrrido a clases separadas.
- **Open/Closed Principle:** Se puede implementar nuevos tipos de colecciones e iteradores y pasarlos al código existente sin que deje de funcionar el programa.
- **Iterar en paralelo**: Dado que cada objeto iterador contiene su propio estado de iteración, se permite iterar la misma colección en paralelo.
- **Retrasar una iteración:** Por la misma razón (cada iterador contiene su estado de iteración), se puede retrasar una iteración y continuar cuando sea necesario.

Desventajas:

- **Overkill:** Puede darse si la aplicación trabaja con colecciones simples.
- **Eficiencia:** Usar un iterador puede ser menos eficiente para algunas colecciones especializadas