Collections en Java

En Java, una Collection es un conjunto de clases e interfaces que sirven para almacenar y manipular grupos de objetos. El paquete que las contiene es java.util. Se pueden entender como contenedores donde guardar objetos.

Principales Interfaces de Collections

1. 1. List

- Permite elementos duplicados.
- Mantiene un orden (índices).
- Ejemplos: ArrayList, LinkedList, Vector

2. 2. Set

- No permite duplicados.
- No garantiza un orden (aunque algunas implementaciones lo tienen).
- Ejemplos: HashSet, LinkedHashSet, TreeSet

3. 3. Map

- Guarda pares clave → valor.
- No puede haber claves repetidas, pero sí valores repetidos.
- Ejemplos: HashMap, LinkedHashMap, TreeMap

Diferencias importantes

ArrayList vs LinkedList:

ArrayList:
- Internamente usa un arreglo dinámico.
- Acceder a un elemento es rápido (índice directo).
- Insertar o eliminar en medio de la lista es más lento (hay que mover elementos).

LinkedList:
- Internamente usa nodos enlazados.
- Acceder a un elemento por índice es más lento (hay que recorrer nodos).
- Insertar o eliminar en cualquier parte es más rápido (solo se cambian enlaces).

¿Cómo funciona un HashMap?

- Es una implementación de la interfaz Map.
- Guarda los datos en forma de pares clave → valor.
- Internamente usa una tabla hash:
  • A cada clave se le aplica una función hash que genera un índice.
  • Ese índice indica en qué “cajón” (bucket) se guarda el valor.
- Ventajas: Búsqueda, inserción y borrado son muy rápidos (casi O(1)).

Ejemplo en Java:

```
Map<String, Integer> edades = new HashMap<>();
edades.put("Ana", 25);
edades.put("Luis", 30);
System.out.println(edades.get("Ana")); // 25
```

Resumen

- **Collection =** marco de trabajo en Java para manejar conjuntos de datos.
- **List**: ordenada, permite duplicados (ArrayList, LinkedList).
- **Set**: no duplicados (HashSet, TreeSet).
- **Map**: pares clave-valor, sin claves duplicadas (HashMap).
- **ArrayList vs LinkedList:** ArrayList rápido para acceder, lento para insertar. LinkedList rápido para insertar/eliminar, lento para acceder.
- **HashMap**: usa función hash para guardar clave-valor de forma eficiente.