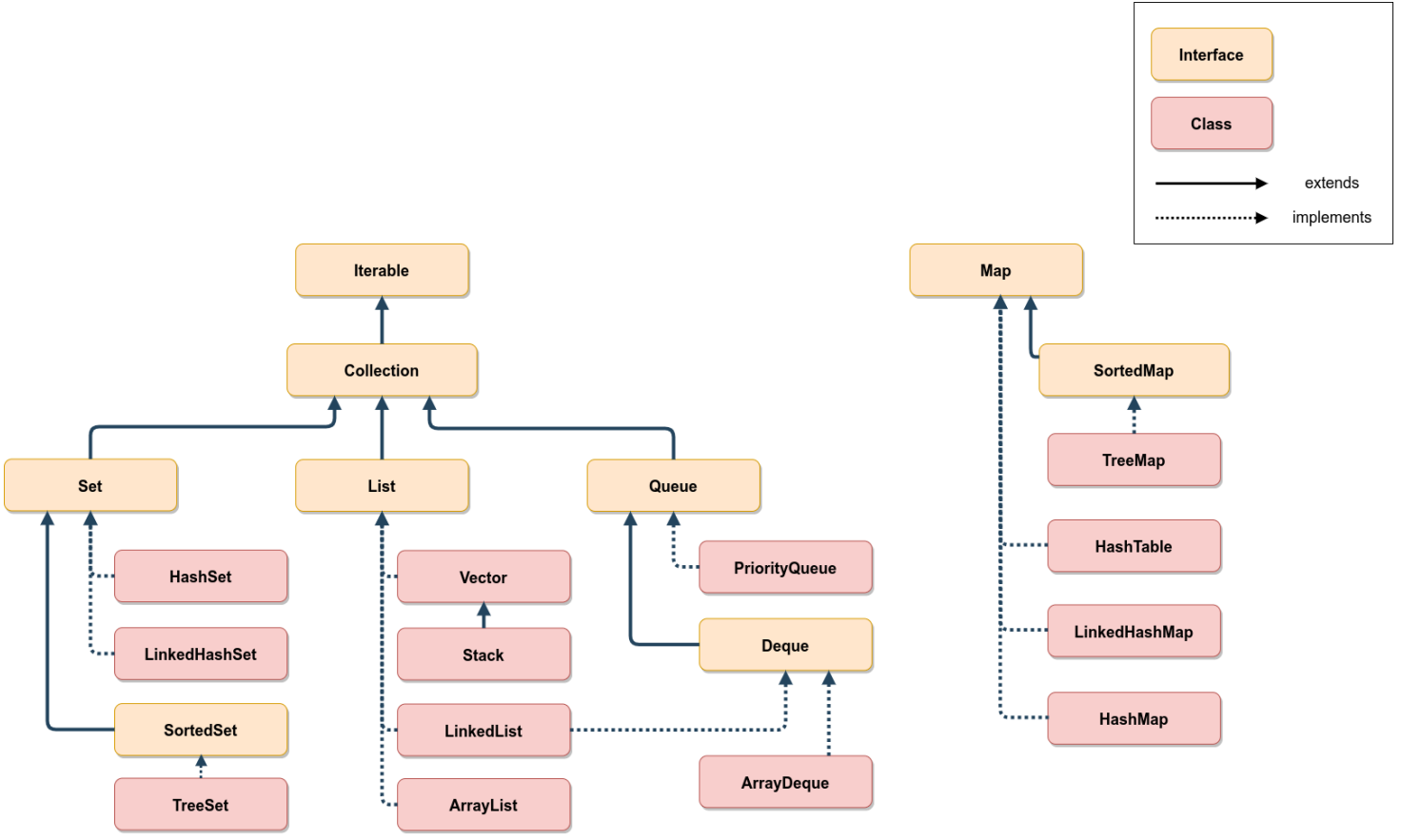
Introducción a Java Collections Framework

Gerson Pérez Ortega
https://twitter.com/iamgersoft
JConf Centroamérica 2020

Java Collections Framework

- Arquitectura unificada que permite una práctica y eficiente manipulación de objetos
- Conjunto de interfaces con sus respectivas implementaciones (estructuras de datos reutilizables)
- Las implementaciones utilizan algoritmos que resuelven operaciones útiles tales como inserción, eliminación, búsqueda u ordenamiento

Jerarquía de Java Collections Framework



Características

	¿Acepta null?	¿Es *thread-safe*?	¿Acepta valores repetidos?	¿Ordenado?
HashSet	Sí	No	No	No
LinkedHashSet	Sí	No	No	Sí
TreeSet	No	No	No	Sí (orden natural)
Vector	Sí	Sí	Sí	Sí
Stack	Sí	Sí	Sí	Sí
LinkedList	Sí	No	Sí	Sí
ArrayList	Sí	No	Sí	Sí
PriorityQueue	Sí	No	Sí	Sí

Características

	¿Acepta null key?	¿Acepta null value?	¿Es *thread-safe*?	¿Acepta keys repetidos?	¿Ordenado?
TreeMap	No	Sí	No	No	Sí (orden natural de keys)
HashTable	No	No	Sí	No	No
LinkedHashMap	Sí	Sí	No	No	Sí
HashMap	Sí	Sí	No	No	No

Consideraciones

- El orden natural puede estar claro en objetos de tipo numérico o en String
- Implementación de la interfaz Comparator con el método compareTo() para objetos cuyo orden natural no está definido
- En qué casos un objeto se considera repetido
- Sobrescritura de los métodos equals() y hashCode()
- Acceso concurrente
- Uso de recursos (memoria, CPU)

Set (java.util.Set)

- Colección de objetos en la que cada objeto es único
- Puede aceptar un único elemento nulo (depende de la implementación)
- Al iterar sobre los objetos de la colección, el orden puede no estar garantizado (HashSet), puede estar sujeto al orden de inserción (LinkedHashSet), o puede depender del orden natural de los objetos (TreeSet)

HashSet (java.util.HashSet)

- Colección de objetos en la que cada objeto es único
- Acepta un único elemento nulo
- Al iterar sobre los objetos de la colección, el orden no está garantizado
- Internamente se aplica una función hash sobre cada objeto con el que se determina su posición en la colección
- Al no seguir un orden determinado, las operaciones sobre HashSet son más rápidas comparadas con TreeSet o LinkedHashSet

LinkedHashSet (java.util.LinkedHashSet)

- Colección de objetos en la que cada objeto es único
- Acepta un único elemento nulo
- Al iterar sobre los objetos de la colección, el orden está garantizado mediante el uso de una lista doblemente enlazada
- Al mantener un orden de inserción, existe una penalización en el rendimiento a diferencia de HashSet

TreeSet (java.util.TreeSet)

- Colección de objetos en la que cada objeto es único
- No acepta elementos nulos
- Sigue un orden natural al iterar sobre los objetos de la colección
- El orden natural está sujeto a la implementación de la interfaz Comparable por medio del método compareTo()
- Mayor penalización en el rendimiento al validar el orden natural en cada operación de inserción, eliminación u obtención

List (java.util.List)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Sigue el orden de inserción al iterar sobre los objetos de la colección

Vector (java.util.Vector)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Sigue el orden de inserción al iterar sobre los objetos de la colección
- Es *thread-safe*, lo que garantiza consistencia en accesos concurrentes desde múltiples hilos de ejecución (internamente todos sus métodos están declarados con el modificador de acceso *synchronized*)

Stack (java.util.Stack)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Sigue el orden de inserción al iterar sobre los objetos de la colección
- Es *thread-safe*, lo que garantiza consistencia en accesos concurrentes desde múltiples hilos de ejecución (internamente todos sus métodos están declarados con el modificador *synchronized*)
- Hereda las características de Vector, y al ser una colección de tipo LIFO (Last In, First Out) provee los métodos push(), pop(), peek(), search() y empty()

LinkedList (java.util.LinkedList)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Al iterar sobre los objetos de la colección, el orden está garantizado mediante el uso de una lista doblemente enlazada
- Por tratarse de una implementación de List como de Queue, puede adoptar las características de una cola
- Buen rendimiento en operaciones de manipulación como add(int index, E element) y remove()

ArrayList (java.util.ArrayList)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Al iterar sobre los objetos de la colección, el orden está sujeto al orden de inserción establecido en los métodos add(E element) como add(int index, E element)
- Buen rendimiento al obtener objetos con el método get(), o al insertar con el método add(E element)

Queue (java.util.Queue)

- Colección de tipo FIFO (First In, First Out)
- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos

PriorityQueue (java.util.PriorityQueue)

- Colección de objetos en la que puede haber objetos repetidos
- Acepta elementos nulos
- Tipo especial de cola (cola de prioridad), la inserción se basa en el orden natural
- El orden natural está sujeto a la implementación de la interfaz Comparable por medio del método compareTo()

ArrayDeque (java.util.ArrayDeque)

- Colección de objetos en la que puede haber objetos repetidos
- No acepta elementos nulos
- Mejor rendimiento que LinkedList (que también implementa la interfaz Queue), internamente no realiza las operaciones de una lista doblemente enlazada

Map (java.util.Map)

- Colección de pares llave/valor
- Un objeto es usado como llave (*key*)
- Un objeto es usado como valor (*value*)
- Un Map no puede contener llaves duplicadas
- Cada llave puede ser mapeada a un solo valor

TreeMap (java.util.TreeMap)

- Ordenamiento basado en el orden natural de sus llaves, o por un *Comparator* definido al crear el TreeMap

HashTable (java.util.HashTable)

- Es *thread-safe*
- No acepta llaves ni valores nulos
- Al iterar, el orden no está garantizado

LinkedHashMap (java.util.LinkedHashMap)

- Solo acepta elementos únicos
- Puede tener una sola llave nula
- Puede haber múltiples valores nulos
- Al iterar, se mantiene el orden de inserción

HashMap (java.util.HashMap)

- Puede tener una sola llave nula
- Puede haber múltiples valores nulos
- Al iterar sobre las llaves, el orden no está garantizado
- Menor huella de memoria al no mantener referencias sobre el orden de inserción a diferencia de LinkedHashMap

¡Gracias!