**TreeMap**

TreeMap es una implementación de la interfaz Map. Map es heredada a la interfaz SortedMap, que a su vez es heredada a la interfaz NavigableMap que finalmente es heredada a TreeMap.

La estructura Map nos da la capacidad de guardar pares de datos, uno fungiendo como clave y otro como el valor en sí que se guarda. El TreeMap sirve para guardar los pares de datos de forma ordenada en función de la clave, por ejemplo si esta fuera un int se podrían ordenar los datos de manera ascendente.

El TreeMap utiliza un árbol de ébano rojo como estructura de almacenamiento.

EJ:

Map<String, Integer> treeMap = new TreeMap<>();

**TreeSet**

TreeSet es una implementación de la interfaz Set. Set es heredada a la interfaz SortedSet que finalmente hereda a la clase TreeSet. La característica de Set es que no permite la duplicidad de datos.

TreeSet tiene la estructura de un árbol rojo-negro auto balanceado, lo que lo hace idea para almacenar datos de forma ordenada. En conjunto de su herencia de Set, es un árbol rojo-negro auto balanceado que no permite datos repetidos.

EJ:

TreeSet ts = new TreeSet();

TreeSet ts = new TreeSet(Comparator comp);

TreeSet t = new TreeSet(Collection col);

**Vector**

Vector implementa la clase List. Vector funciona como arreglo de dimensión dinámica, es decir que puede aumentar y disminuir su dimensión. Es una estructura que permite cualquier tipo de dato.

Ej:

Vector<E> v = new Vector<E>();

Vector<E> v = new Vector<E>(int dim);

Vector<E> v = new Vector<E>(int dim, int incr);

Vector<E> v = new Vector<E>(Collection c);

**Typesafe view**

Las typesafe view o vistas seguras son vistas de una colección pero que garantiza la seguridad de los tipos de datos. Por ejemplo, con una lista normal se pueden agregar datos que no son los indicados y esto ocasionara un error, mientras una vista segura no nos permite hacer eso.

|  |  |  |
| --- | --- | --- |
| Modificador y Tipo | Métodos y parametros | Explicación |
| static <T> boolean | addAll(Collection <?super T>,T..elementos) | Sirve para insertar elementos en una colección cualquiera. |
| static <T> Queue<T> | asLifoQueue(Deque<T> deque) | Retorna una Deque como Queue |
| static <T> int | binarySearch(List<? extends Comparable<? super T>> list, T buscado)  //Sobrecarga  binarySearch(List<? extends Comparable<? super T>> list, T buscado),Comparator<?super T> comparador>) | Usa la búsqueda binaria para encontrar un elemento en una colección. |
| static <E> Collection<E> | checkedCollection(Collection<E> coleccion, Class<E> tipo) | Retorna una vista segura dinámica de una collecion. |
| static <E> List<E> | checkedList(ist<E> lista, Class<E> tipo) | Retorna una vista segura dinámica de una lista. |
| static <K,V> Map<K,V> | checkedMap (Map<K,V> mapa , Class<K> TipoLlave, Class<V> TipoVlalor) | Retorna una vista segura dinámica de un Map. |
| static <K,V> NavigableMap<K,V> | checkedNavigableMap(NMap<K,V> mapa , Class<K> TipoLlave, Class<V> TipoVlalor) | Retorna una vista dinámica segura de un Navigable Set |
| static <E> NavigableSet<E> | checkedNavigableSet (NavigableSet<E> NS, Class<E> tipo) | Retorna una vista segura dinámica de un navigable set |
| static <E> Queue<E> | checkedQueue(Queue<E> queue, Class<E> tipo) | Retorna una vista segura dinamica de una queue |
| static <E>Set<E> | checkedSetcheckedSet(Set<E> s, Class<E> tipo) | Retorna una vista segura dinámica de un set especifico |
| static <K,V> SortedMap<K,V> | checkedSortedMap(SortedMap<K,V> m, Class<K> TipoLlave, Class<V> TipoValor) | Retorna una vista segura dinámica de un sorted map |
| static <E> SortedSet<E> | checkedSortedSet(SortedSet<E> s, Class<E> tipo) | Retorna una vista segura de un sorted set |

Bibliografía

Бранниган З. Características de TreeMap en Java. JavaRush. https://javarush.com/es/groups/posts/es.2584.caractersticas-de-treemap-en-java. Published 29 de marzo de 2024.

Jarroba R [Admin. Map en Java, con ejemplos - Jarroba. Jarroba. https://jarroba.com/map-en-java-con-ejemplos/. Published 2 de junio de 2017.

Squirrels J. Conjunto de árboles en Java. CodeGym. https://codegym.cc/es/groups/posts/es.1111.conjunto-de-rboles-en-java. Published 14 de febrero de 2023.

GfG. TreeSet in Java. GeeksforGeeks. https://www.geeksforgeeks.org/treeset-in-java-with-examples/. Published 10 de enero de 2023.

Divertitto A. Java Vector. CodeGym. https://codegym.cc/es/groups/posts/es.749.java-vector. Published 31 de enero de 2024.

GfG. Vector Class in Java. GeeksforGeeks. https://www.geeksforgeeks.org/java-util-vector-class-java/. Published 14 de marzo de 2024.

Collections (Java Platform SE 8 ).

https://docs.oracle.com/javase/8/docs/api/java/util/Collections.html. Published 8 de enero de 2024.

Java Collections class - javatpoint. www.javatpoint.com. https://www.javatpoint.com/java-collections-class.