


Class	Implements	Duplicates Allowed	Maintains Order	Sorted	Allows Nulls	Thread-Safe	Best Use Case
ArrayList	List	✓ Yes	✓ Yes	✗ No	✓ Yes (1 null)	✗ No	Random access, fast read
LinkedList	List, Deque	✓ Yes	✓ Yes	✗ No	✓ Yes	✗ No	Frequent insert/delete
Vector	List	✓ Yes	✓ Yes	✗ No	✓ Yes	✓ Yes	Legacy synchronized list
Stack	Vector	✓ Yes	✓ Yes (LIFO)	✗ No	✓ Yes	✓ Yes	LIFO operations
HashSet	Set	✗ No	✗ No	✗ No	✓ Yes (1 null)	✗ No	Unique elements, fast lookup
LinkedHashSet	Set	✗ No	✓ Yes	✗ No	✓ Yes	✗ No	Unique with insertion order
TreeSet	NavigableSet	✗ No	✓ Yes (Sorted)	✓ Yes (Natural/Comparable)	✗ No (no nulls)	✗ No	Sorted unique elements
HashMap	Map	<div>✓ Keys: ✗</div> <div>Duplicates</div> <div>✓ Values</div>	✗ No	✗ No	<div>✓ One null key, multiple null values</div>	✗ No	Key-value store, fast lookup
LinkedHashMap	Map	✓ Yes	✓ Yes	✗ No	✓ Yes	✗ No	Maintains insertion order in maps
TreeMap	NavigableMap	✓ Yes	✓ Yes (Sorted)	✓ Yes	✗ No (no null keys)	✗ No	Sorted key-value pairs

Class	Implements	Duplicates Allowed	Maintains Order	Sorted	Allows Nulls	Thread-Safe	Best Use Case
Hashtable	Map	✓ Yes	✗ No	✗ No	✗ No (no nulls)	✓ Yes	Legacy synchronized map
PriorityQueue	Queue	✓ Yes	✗ No (Heap order)	✓ Yes (Priority)	✗ No	✗ No	Priority-based processing
ArrayDeque	Deque	✓ Yes	✓ Yes	✗ No	✗ No	✗ No	Fast stack/queue with no capacity restrictions
EnumSet	Set	✗ No	✓ Yes (Enum order)	✓ Yes	✗ No	✗ No	Efficient set for enum types
WeakHashMap	Map	✓ Yes	✗ No	✗ No	✓ Yes	✗ No	GC-aware map (weak keys)
ConcurrentHashMap	Map	✓ Yes	✗ No	✗ No	✗ No null keys	✓ Yes	Thread-safe map without locking
CopyOnWriteArrayList	List	✓ Yes	✓ Yes	✗ No	✓ Yes	✓ Yes	Thread-safe list, good for read-heavy use

Feature	LinkedHashMap	TreeMap	Hashtable
Allows Duplicate Keys	✗ No (keys must be unique)	✗ No (keys must be unique)	✗ No (keys must be unique)

Feature	LinkedHashMap	TreeMap	Hashtable
Allows Duplicate Values	✔ Yes	✔ Yes	✔ Yes
Key Order Maintained	✔ Yes (insertion order)	✔ Yes (sorted order - natural/comparator)	✘ No (no guaranteed order)
Null Keys Allowed	✔ Yes (only 1 null key)	✘ No (throws NullPointerException)	✘ No (throws NullPointerException)
Null Values Allowed	✔ Yes (multiple null values)	✔ Yes	✘ No (throws NullPointerException)
Thread-Safe	✘ No	✘ No	✔ Yes (fully synchronized)
Use Case	Maintain key insertion order	Maintain sorted order	Legacy synchronized map

Here's a  **complete list of core implementation classes** in the **Java Collection Framework**, organized by their respective interfaces (e.g., List, Set, Map, Queue, etc.).

1. List Interface – Ordered Collection (Allows Duplicates)

Implementation Class Description

ArrayList	Resizable array, fast random access
LinkedList	Doubly linked list, efficient insert/delete
Vector (legacy)	Synchronized dynamic array
Stack (extends Vector)	Legacy LIFO stack
CopyOnWriteArrayList	Thread-safe list (java.util.concurrent)

2. Set Interface – No Duplicates

Implementation Class Description

HashSet	Unordered set, backed by HashMap
LinkedHashSet	Maintains insertion order
TreeSet	Sorted set (Red-Black tree)
EnumSet	High-performance set for enums
CopyOnWriteArraySet	Thread-safe set (java.util.concurrent)

3. Queue & Deque Interfaces – FIFO, LIFO, and Priority

Implementation Class Description

PriorityQueue	Elements ordered by priority
ArrayDeque	Resizable double-ended queue

Implementation	Class	Description
----------------	-------	-------------

LinkedList		Implements both Queue and Deque
ConcurrentLinkedQueue		Thread-safe queue (non-blocking)
LinkedBlockingQueue		Blocking queue (java.util.concurrent)
ArrayBlockingQueue		Bounded blocking queue
PriorityBlockingQueue		Thread-safe priority queue
DelayQueue		Elements become available after delay
SynchronousQueue		For thread handoff, no internal storage
LinkedTransferQueue		High-performance concurrent queue

4. Map Interface – Key-Value Pairs

Implementation	Class	Description
----------------	-------	-------------

HashMap		Fast lookup via hash table
LinkedHashMap		Maintains insertion order
TreeMap		Sorted map (Red-Black tree)
Hashtable (legacy)		Synchronized map
WeakHashMap		Keys are weakly referenced
IdentityHashMap		Compares keys by reference (==)
EnumMap		Efficient map for enum keys
ConcurrentHashMap		Thread-safe, high concurrency

5. Other Specialized Implementations

Class	Implements/Supports
Properties	Subclass of Hashtable, used for configs
Collections (Utility)	Static utility methods for collections
Arrays (Utility)	Static methods for arrays