Java Collections Framework

1. Iterable (Java.util Package)

- **Parent Interface of:** Collection interface
- **Methods:**
  - iterator()
  - forEach()
  - spliterator()

---

2. Collection (Java.util Package)

- **Child Interface of:** Iterable
- **Parent Interface of:** List, Set, Queue
- **Purpose:** Represents a group of objects
- **Methods:**
  - size(), isEmpty(), remove(), removeAll()
  - add(), addAll(), clear(), contains(), containsAll(), toArray()
- **Introduced in:** Java **1.2**

---

3. List (Java.util Package, Java 1.2)

- **Child Interface of:** Collection
- **Implemented By:** ArrayList, LinkedList, Vector, Stack
- **Characteristics:**
  - Allows **duplicates**
  - Allows **null elements**
  - **Maintains order**
  - Supports **heterogeneous** elements

3.1 ArrayList (Java.util Package, Java 1.2)

- **Allows:** null, duplicates, heterogeneous elements
- **Maintains Order:** Yes

3.2 LinkedList (Java.util Package)

- **Allows:** null, duplicates, heterogeneous elements
- **Maintains Order:** Yes

- **Underlying Structure:** Uses **doubly linked list**
- **Default Initial Capacity:** 0

3.3 Vector (Java.util Package, Java 1.0)

- **Allows:** null, duplicates, heterogeneous elements
- **Maintains Order:** Yes
- **Underlying Structure:** Uses **array**
- **Initial Capacity:** 10, increases **2×**
- **Performance:**
    - **Best choice for retrieval**
    - **Worst choice for insertion/deletion (synchronized)**

3.4 Stack (Java.util Package, Extends Vector)

- **Allows:** null, duplicates, heterogeneous elements
- **Maintains Order:** Yes
- **Follows: LIFO (Last In, First Out)**
- **Thread-Safe:** Yes (synchronized)
- **Methods:**
    - push(), pop(), peek(), search(), etc.

---

4. Set (Java.util Package, Java 1.2)

- **Child Interface of:** Collection
- **Parent Interface of:** SortedSet
- **Implemented By:** HashSet, LinkedHashSet
- **Characteristics:**
    - **Does not allow duplicates**
    - **Does not allow null elements**

4.1 HashSet (Java.util Package, Java 1.2)

- **Allows:** null, heterogeneous elements
- **Does Not Allow:** Duplicates
- **Order Maintenance:** No (uses **Hashtable**)
- **Usage:** Removing duplicate elements
- **Initial Capacity:** 16, Load Factor: 0.75
- **Thread Safety: Non-synchronized** (Thread-unsafe)

4.2 LinkedHashSet (Java.util Package, Java 1.2)

- **Allows:** null elements
- **Does Not Allow:** Duplicates
- **Maintains Order:** Yes
- **Initial Capacity:** 16, Load Factor: 0.75
- **Thread Safety: Non-synchronized**

---

5. SortedSet (Java.util Package, Java 1.2)

- **Child Interface of:** Set
- **Parent Interface of:** NavigableSet
- **Characteristics:**
  - **Does not allow duplicates**
  - **Does not allow null elements**
  - **Does not allow heterogeneous elements**
  - **Does not maintain insertion order (Sorts data instead)**
- **Methods:**
  - subset(), headSet(), tailSet(), first(), last()

---

6. NavigableSet (Java.util Package, Java 1.6)

- **Child Interface of:** SortedSet
- **Implemented By:** TreeSet
- **Characteristics:**
  - **Does not allow duplicates, null elements, heterogeneous elements**
  - **Maintains Order**
- **Methods:**
  - higher(), lower(), floor(), ceiling(), pollFirst(), pollLast()

6.1 TreeSet (Java.util Package)

- **Does Not Allow:** null, duplicates, heterogeneous elements
- **Maintains Order:** Yes
- **Sorting:**
  - **Primitive Data Type:** Uses **Default Natural Sorting Order (DNSO)**
  - **Non-Primitive Data Type:** Uses **Customized Natural Sorting Order (CNSO)**

    - **Requires:** Comparable or Comparator implementation
- **Character Comparison:** Uses **ASCII values**

---

7. Generic vs Non-Generic Collection

Feature	Non-Generic Collection (Java 1.2)	Generic Collection (Java 1.5)
Allows Heterogeneous	✅ Yes	❌ No
Compile-time Error	❌ No	✅ Yes (Type-safe)
Processing Support	❌ Limited	✅ Strong

---

7. Queue (Java.util Package, Java 1.5)

- **Child Interface of:** Collection
- **Implemented By:** PriorityQueue, Deque, BlockingQueue
- **Characteristics:**
    - **Follows FIFO (First In, First Out) order**
    - **Allows duplicates**
    - **Allows null elements** (except some implementations like PriorityQueue)
- **Methods:**
    - offer(), poll(), peek(), remove(), element()

7.1 PriorityQueue (Java.util Package, Java 1.5)

- **Child Class of:** Queue
- **Characteristics:**
    - **Does not allow null elements**
    - **Allows duplicates**
    - **Does not maintain insertion order (Uses priority order)**
    - **Elements are ordered using natural ordering (Comparable) or custom comparator (Comparator)**
    - **Implemented using Binary Heap Data Structure**
- **Initial Capacity:** 11 (Default)
- **Thread Safety: Non-synchronized**

---

8. BlockingQueue (Java.util.Concurrent Package, Java 1.5)

- **Extends:** Queue
- **Implemented By:** ArrayBlockingQueue, LinkedBlockingQueue, PriorityBlockingQueue, SynchronousQueue
- **Characteristics:**
    - **Thread-safe** (Used in multi-threading environments)
    - **Blocks the thread** when adding elements to a full queue or retrieving from an empty queue
    - **Does not allow null elements**
- **Methods:**
    - put(), take(), offer(), poll(), drainTo()

8.1 ArrayBlockingQueue (Java.util.Concurrent Package, Java 1.5)

- **Implements:** BlockingQueue
- **Characteristics:**
    - **Fixed-size queue (Capacity set at creation time)**
    - **Thread-safe (Uses locks for synchronization)**
    - **Maintains FIFO order**
    - **Does not allow null elements**
    - **Bounded blocking queue**
- **Best Choice For:** Bounded queues with high performance in producer-consumer problems

---

8.2 LinkedBlockingQueue (Java.util.Concurrent Package, Java 1.5)

- **Implements:** BlockingQueue
- **Characteristics:**
    - **Unbounded queue (default capacity = Integer.MAX_VALUE, but can be set)**
    - **Thread-safe (Uses locks for synchronization)**
    - **Maintains FIFO order**
    - **Does not allow null elements**
- **Best Choice For:** High-throughput producer-consumer scenarios

---

8.3 PriorityBlockingQueue (Java.util.Concurrent Package, Java 1.5)

- **Implements:** BlockingQueue
- **Characteristics:**

    - **Unbounded queue (Dynamically resizes as needed)**
    - **Thread-safe**
    - **Uses priority order instead of FIFO**
    - **Allows duplicates**
    - **Elements must be Comparable or provided with a Comparator**
- **Best Choice For:** Priority-based task scheduling in multi-threading

---

8.4 SynchronousQueue (Java.util.Concurrent Package, Java 1.5)

- **Implements:** BlockingQueue
- **Characteristics:**
    - **Does not store elements (Queue size is always 0)**
    - **Every put() operation must wait for a take() operation and vice versa**
    - **Thread-safe**
    - **Does not allow null elements**
- **Best Choice For:**
    - **Hand-off messaging** between producer and consumer threads
    - **Task delegation where one thread hands over work to another thread immediately**
