**What is fail safe and fail fast Iterator in Java?**

**Answer: -** Java Collections supports two types of Iterator, fail safe and fail fast. The main distinction between a fail-fast and fail-safe Iterator is whether or not the underlying collection can be modified while its begin iterated. If you have used Collection like ArrayList then you know that when you iterate over them, no other thread should modify the collection. If Iterator detects any structural change after iteration has begun e.g adding or removing a new element then it throws ConcurrentModificationException, this is known as fail-fast behavior and these iterators are called fail-fast iterator because they fail as soon as they detect any modification . Though it's not necessary that iterator will throw this exception when multiple threads modified it simultaneously. it can happen even with the single thread when you try to remove elements by using ArrayList's remove() method instead of Iterator's remove method.

Most of the Collection classes from Java 1.4 e.g. Vector, ArrayList, HashMap, HashSet has fail-fast iterators. The other type of iterator was introduced in Java 1.5 when concurrent collection classes e.g. ConcurrentHashMap, CopyOnWriteArrayList and CopyOnWriteArraySet was introduced.

These iterator uses a view of original collection for doing iteration and that's why they doesn't throw ConcurrentModificationException even when original collection was modified after iteration has begun. This means you could iterate and work with stale value, but this is the cost you need to pay for fail-safe iterator and this feature is clearly documented.

**What is difference between Fail Safe and Fail Fast Iterator in Java**

**Answer: -**

* Fail-fast Iterator throws ConcurrentModfiicationException as soon as they detect any structural change in collection during iteration, basically which changes the modCount variable hold by Iterator. While fail-fast iterator doesn't throw CME.
* Fail-fast iterator traverse over original collection class while fail-safe iterator traverse over a copy or view of original collection. That's why they don't detect any change on original collection classes and this also means that you could operate with stale value.
* Iterators from Java 1.4 Collection classes e.g. ArrayList, HashSet and Vector are fail-fast while Iterators returned by concurrent collection classes e.g. CopyOnWriteArrayList or CopyOnWriteArraySet are fail-safe.
* Iterator returned by synchronized Collection are fail-fast while iterator returned by concurrent collections are fail-safe in Java.
* Fail fast iterator works in live data but become invalid when data is modified while fail-safe iterator are weekly consistent.

**When to use fail fast and fail-safe Iterator**

**Answer: -**Use fail-safe iterator when you are not bothered about Collection to be modified during iteration, as fail-fast iterator will not allow that. Unfortunate you can't choose fail safe or fail-fast iterator, it depends on upon which Collection class you are using. Most of the JDK 1.4 Collections e.g. HashSet, Vector, ArrayList has fail-fast Iterator and only Concurrent Collections introduced in JDK 1.5 e.g. CopyOnWriteArrayList and CopyOnWriteArraySet supports fail safe Iteration. Also, if you want to remove elements during iteration please use iterator's remove() method and don't use remove method provided by Collection classes e.g. ArrayList or HashSet because that will result in ConcurrentModificationException.