1. *Time-Based Release Versioning (JEP 322)*

*With the adoption of the time-based release cycle, Oracle changed the version-string scheme of the Java SE Platform and the JDK, and related versioning information, for present and future time-based release models. The new pattern of the Version number is: $FEATURE.$INTERIM.$UPDATE.$PATCH $FEATURE: counter will be incremented every 6 months and will be based on feature release versions, e.g: JDK 10, JDK 11. $INTERIM: counter will be incremented for non-feature releases that contain compatible bug fixes and enhancements but no incompatible changes. Usually, this will be zero, as there will be no interim release in a six month period. This kept for a future revision to the release model. $UPDATE: counter will be incremented for compatible update releases that fix security issues, regressions, and bugs in newer features. This is updated one month after the feature release and every 3 months thereafter. The April 2018 release is JDK 10.0.1, the July release is JDK 10.0.2, and so forth $PATCH: counter will be incremented for an emergency release to fix a critical issue. New API’s have been added to get these counter values programmatically.*

1. *Local-Variable Type Inference is the biggest new feature in Java 10 for developers. It adds type inference to declarations of local variables with initializers. Local type inference can be used only in the following scenarios:*

* *Limited only to Local Variable with initializer*
* *Indexes of enhanced for loop or indexes.*
* *Local declared in for loop.*
* [*https://www.baeldung.com/java-10-local-variable-type-inference*](https://www.baeldung.com/java-10-local-variable-type-inference) *- reference*

1. *copyOf() -*

*java.util.List, java.util.Map and java.util.Set each got a new static method copyOf(Collection). It returns the unmodifiable copy of the given Collection:*

*@Test(expected = UnsupportedOperationException.class)*

*public void whenModifyCopyOfList\_thenThrowsException() {*

*List<Integer> copyList = List.copyOf(someIntList);*

*copyList.add(4);*

*}*

*Any attempt to modify such a collection would result in java.lang.UnsupportedOperationExceptionruntime exception.*

1. *toUnmodifiable\*()*

*java.util.stream.Collectors get additional methods to collect a Stream into unmodifiable List, Map or Set:*

*@Test(expected = UnsupportedOperationException.class)*

*public void whenModifyToUnmodifiableList\_thenThrowsException() {*

*List<Integer> evenList = someIntList.stream()*

*.filter(i -> i % 2 == 0)*

*.collect(Collectors.toUnmodifiableList());*

*evenList.add(4);*

*}*

*Any attempt to modify such a collection would result in java.lang.UnsupportedOperationExceptionruntime exception.*

1. *Optional\*.orElseThrow()*

*java.util.Optional, java.util.OptionalDouble, java.util.OptionalInt and java.util.OptionalLong each got a new method orElseThrow() which doesn’t take any argument and throws NoSuchElementException if no value is present*

1. *Parallel Full GC for G1*

*The G1 garbage collector is the default one since JDK 9. However, the full GC for G1 used a single threaded mark-sweep-compact algorithm.*

*This has been changed to the parallel mark-sweep-compact algorithm in Java 10 effectively reducing the stop-the-world time during full GC.*

1. *Container Awareness*

*JVMs are now aware of being run in a Docker container and will extract container-specific configuration instead of querying the operating system itself – it applies to data like the number of CPUs and total memory that have been allocated to the container. this change adds a JVM option that provides the ability to specify the number of CPUs that the JVM will use:*

*-XX:ActiveProcessorCount=count*

*Also, three new JVM options have been added to allow Docker container users to gain more fine-grained control over the amount of system memory that will be used for the Java Heap:*

*-XX:InitialRAMPercentage*

*-XX:MaxRAMPercentage*

*-XX:MinRAMPercentage*