Java 9 Features

Below are the some of the important features of Java 9.

Jshell -Adds Read-Eval-Print Loop (REPL) functionality to the Java platform.

The jshell tool provides an interactive command-line interface for evaluating declarations, statements, and expressions of the Java programming language. It facilitates prototyping and exploration of coding options with immediate results and feedback.

```
% jshell
|  Welcome to JShell -- Version 9
|  For an introduction type: /help intro

jshell>
```

```
jshell> String grade(int testScore) {
   ...>     if (testScore >= 90) {
   ...>         return "Pass";
   ...>     }
   ...>     return "Fail";
   ...> }
|  created method grade(int)

jshell> grade(88)
$3 ==> "Fail"
```

Compile for Older Platform Versions: Enhance javac so that it can compile Java programs to run on selected older versions of the platform.

A new command-line option, --release, is defined, which automatically configures the compiler to produce class files that will link against an implementation of the given platform version. --release N is roughly equivalent to:

- for N < 9: -source N -target N -bootclasspath <documented-APIs-from-N>,
- for N >= 9: -source N -target N --system <documented-APIs-from-N>.

As of JDK 9, the javac doesn't support -source release settings less than or equal to 5. If settings less than or equal to 5 are used, then the javac command behaves as if -source 6 were specified

Jlink&Modular Java Application Packaging: You can use the *jlink* tool to assemble and optimize a set of modules and their dependencies into a custom runtime image.

The jlink tool defines a plug-in mechanism for transformation and optimization during the assembly process, and for the generation of alternative image formats. It can create a custom runtime optimized for a single program.

Example:jlink --module-path $JAVA_HOME/jmods:mlib --add-modules com.mycustomapp --output MyCustomApp

MyCustomApp/bin/java --list-modules

com.mycustomapp

java.base@9

java.logging@9

org.astro@1.0

Java Doc Improvements:

- Javadoc Search: Provides a search box to the generated API documentation. Use this search box to find program elements, tagged words, and phrases within the documentation.
- Java Doc is fully compliant HTML5 output, ensure that any HTML content provided in documentation comments are compliant with HTML5.
- Supports documentation comments in module declarations

- Allow @SafeVargs on private instance methods.
- Allow effectively final variables to be used as resources in the try-with-resources statement.
- Allow the diamond with anonymous classes if the argument type of the inferred type is denotable.
- Complete the removal, begun in Java SE 8, of the underscore from the set of legal identifier names.
- Add support for private interface methods.
- Removes garbage collector (GC) combinations that were deprecated in JDK 8
    - DefNew + CMS
    - ParNew + SerialOld

- Incremental CMS
- Deprecates the Concurrent Mark Sweep (CMS) garbage collector

- **Makes Garbage-First (G1) the default garbage collector (GC)** on 32- and 64-bit server configurations. Using a low-pause collector such as G1 provides a better overall experience, for most users, than a throughput-oriented collector such as the Parallel GC, which was previously the default.
- **Compact String**:Adopts a more space-efficient internal representation for strings. Previously, the String class stored characters in a char array, using two bytes (16 bits) for each character. The new internal representation of the String class is a byte array plus an encoding-flag field. This is purely an implementation change, with no changes to existing public interfaces.
- **Factory Methods for collections**:New static factory methods on the List, Set, and Map interfaces make it simpler to create immutable instances of those collections.

  To create immutable set:

  Set<String>stringSet = Set.of("a", "b", "c");

  To create immutable List :

  List stringList = List.of("a", "b", "c");

  To create immutable Map :

  Map stringMap = Map.of("a", 1, "b", 2, "c", 3);

- **Enhanced Deprecation:**Revamps the @Deprecated annotation to provide better information about the status and intended disposition of an API in the specification.
  - @Deprecated(forRemoval=true) indicates that the API will be removed in a future release of the Java SE platform.
  - @Deprecated(since="version") contains the Java SE version string that indicates when the API element was deprecated, for those deprecated in Java SE 9 and beyond.

  Jdperscan is a static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements