Java 9 Features

**Java 9 REPL(JShell)** :

Java released Read Evaluate Print Loop(REPL) just like node REPL and python repl to compile and run the programs JIT.

Eg.

G:\>jshell | Welcome to JShell -- Version 9-ea

| For an introduction type: /help intro

jshell> int a = 10 a ==> 10

jshell> System.out.println("a value = " + a ) a value = 10

**Factory Methods for Immutable List, Set, Map and Map.Entry**:

In Java SE 8 and earlier versions, We can use Collections class utility methods like unmodifiableXXX to create Immutable Collection objects. For instance, if we want to create an Immutable List, then we can use Collections.unmodifiableList method.

However, these Collections.unmodifiableXXX methods are a tedious and verbose approach. To overcome those shortcomings, Oracle Corp has added a couple of utility methods to List, Set and Map interfaces.

List.of() – returns ImmutableList

Set.of() – returns ImmutableSet

Map.of(“key1”,1,”key2”,2,”key3”,3) up to 10 key value pairs

**Warning – Iteration order is not guaranteed for maps and sets**

**Private methods in Interfaces**:

In Java .we can have default as well as private methods within an interface:

Advantages :

Code reuse

Reduces redundancy of duplicacy of code across all clients.

**Java 9 Module System:**

**Motivations:**

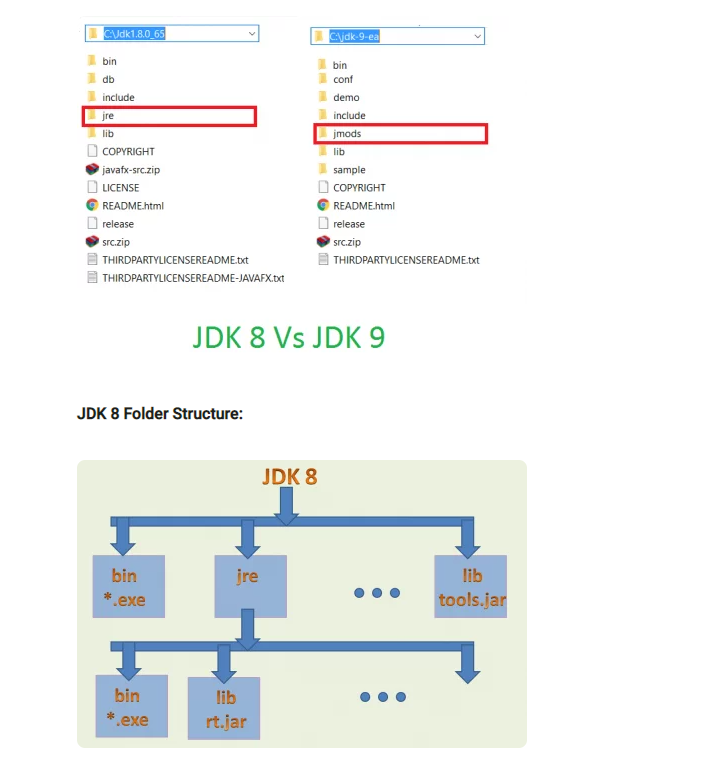
* Imagine rt.jar(runtime.jar) which encapsulates 1000s of classes and is a big monolith.
* The purpose of modules is to break big monolith into more manageable jars.
* java.base is the base module
* A module is a set of packages designed for reuse – Alex Buckley
* Strong encapsulation
* Access control is more powerful
* Reliable dependencies and no missing dependencies
* No Split packages, means if there are 2 packages imported from 2 imports sometimes java loads classes from 1 package and some from another.
* Run with java -p mods -m hello.world
* Monolithic JDK is broken into modules
* java.base is base framework
* When we expose a package via jar we expose all public classes of that package to the outer world which may not always be the case.
* A concern could be that exports and requires and modules could be keywords in existing code, however these are internal keywords only in context of the module and not outside, that makes it extremely backward compatible.
* use java –list-modules – lists all modulesn

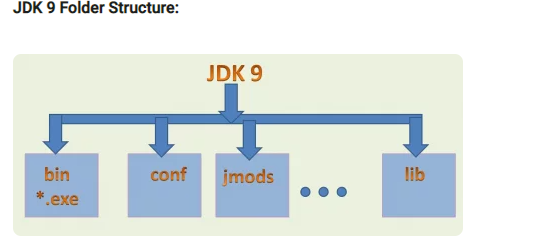
Advantages:

* Increases security y exposing only important classes
* Easy Deprecation, at some point the module may need to be deprecated, in java 9 java corba has been marked as deprecated
* Future Proofing of new modueles
* Evolve , maintainable testable

Issues :

* Issue with encapsulation for classes compiled in pre java9
* If package is not reachable from java.se etc, then it will give complication issue in java 9. Use javac –add-modules javax.xml.bind Main.java to compile and run





Adavantages : Only use exported packages via modules

**Improved Javadocs** :

**Stream Improvements**:

Added methods :

* takeWhile
  + loops while predicate is true
* dropWhile
* ofNullable
* iterate