

Comparative Study of Java and Clojure Programming Languages

Ivan Zelenkov
University of New Orleans
New Orleans, USA
itproger.ivan@gmail.com

Abstract

Nowadays many developers create and maintain applications and consider Java as the main programming language to be used. However, there exists a relatively new and simple language that is increasing in popularity, and that is a Clojure functional language. I conducted a detailed comparison of their features and how those programming languages solve a power set generation problem.

Keywords: comparison, Java, Clojure, programming languages, power set generation problem.

1. Introduction

1.1 Java

Java is an object-oriented programming language developed by Sun Microsystems (later acquired by Oracle). The official release date is May 23, 1995.

Java programs are translated into bytecode, which is then executed by the Java Virtual Machine (JVM). JVM is a program that processes byte code and passes instructions to the hardware as an interpreter. The advantage of such an implementation is the independence of the bytecode from the operating system and hardware, which allows you to run Java applications on any device for which a JVM exists.

Java has a very good security system because the programs are controlled by the virtual machine. That is, any attempt of unauthorized access causes an immediate interruption of the system.

Often, the disadvantages of the concept of a virtual machine include the fact that the execution of bytecode by a virtual machine can reduce the performance of programs and algorithms implemented in the Java language. Programs written in Java have a reputation for being slower and taking up more RAM than those written in C. However, when compared to the interpreted languages that are most used in web development, Java's performance is usually noticeably better.

1.2 Clojure

Initially, the concept of closure comes from abstract mathematics, where it denotes sets that are closed to themselves. In functional languages, "closure" refers to the property of functions to refer to the context of the function that spawned it, even if the parent function has already been completed a long time ago. In terms of its meaning, this

concept can be compared with the concept of "encapsulation" in object-oriented programming languages (OOP).

Clojure is a general-purpose Lisp-like language designed for the Java Virtual Machine (JVM). The author of the language is Rich Hickey, who developed the language alone for several years until the release of the first public version in 2007.

Unlike other implementations of Lisp and Scheme for the Java Virtual Machine, such as ABCL, Kawa, etc., Clojure is not 100 percent compatible with either Common Lisp or Scheme but borrows many of the ideas from those languages by adding new things like data immutability, concurrent code execution, etc.

Even though Clojure is a young programming language, quite a lot of people use it in their projects, including commercial ones.

2. Analysis

2.1 Java

Java has two properties that determine which tasks can be performed on it. These properties have complicated names, but it's worth understanding them before moving on. Java is an object-oriented programming language (OOP). All interaction in it occurs through objects. In general, this is like what is happening in the real world: the cat interacts with the owner, the cashier interacts with the buyer, and the bank client interacts with his bank account. All these entities are described in code and taught to interact with each other. As a result, an OOP-style program consists of separate blocks that are well-extensible and scalable. Therefore, the Java language is suitable for developing programs that are planned to be used for a long time and constantly developed.

Java takes the best of compiled and interpreted languages. To understand this property, you need to step back a little. A programming language is a language in which the programmer and the processor agree on how to execute instructions. So, the processor is not a polyglot and is not required to know all the languages in which they want to command. Therefore, the programming language must be translated into the language of the processor. This is done in two ways - interpreting and compiling.

Java is a compiled language, but it is compiled in an unusual way: first, it is compiled into bytecode, a special code that the Java machine understands. And then it interprets the bytecode into machine code.

Java is an "old" programming language, but it has every chance to stay forever. It quickly became popular thanks to the virtual machine. It is an intermediary between code and hardware. Java provides the main advantage of the Java language - cross-platform. Usually, a Windows program cannot be run on macOS, a lot needs to be rewritten. But the Java program is possible. In Java, a programmer writes code, not for macOS, Windows or Linux, but for a Java machine - and it itself adapts the code to the hardware and operating system. Java code is written once and runs on any device for which a Java machine is written. This allows you to spend less resources on software development.

Such popularity has led to the fact that now a lot of code is written in this language for IT companies, insurance companies, banks, and more.

For example, when we pay by phone, information about the payment is processed by a dozen different devices so that it gets into the payment systems, and the money is debited from the buyer and received by the seller. Most of these operations are performed by specific Java programs, and they need to be maintained and improved with upcoming technologies. Therefore, in the coming decades, no Java developer will lose his job because Java will replace some new language - even if it is faster, easier, and safer. The payment system is just one example of using Java. There are many such systems, and all of them need to be maintained. Considering how much code in the world is written in Java, everybody believes that this language has a chance to stay on par with C forever. Everything is written in Java: from calculators to software for industrial installations. [2]

As it was said almost everything is written in Java, therefore the scope of the language is very wide. Here are just some examples of applications that can be developed using Java:

- banking programs;
- desktop applications;
- industrial programs;
- applications for Android;
- web application, web server, application server;
- corporate software.

But Java games are rarely programmed, because the game needs perfect optimization for the processor and video card. If the optimization is bad, then most average computers will not run the game. Because of the Java machine, it is impossible to make perfect optimization, but in C++ it is possible.

Large companies do not use one technology, but in one form or another Java is present in Google, Meta (Facebook), Telegram, and many others. Java is hidden under the hood and the average user cannot see it. What we see in the interface is not Java, but JavaScript. It's a browser-only language, and with Java, they only have a name in common. The Java language works when the user accesses the server.

2.2 Clojure

Clojure is a dialect of Lisp that shares a similar code-as-data philosophy of homoiconicity as well as a powerful macro system. The principles of homoiconicity make it possible to use the extremely small syntax of the Clojure language, in which the structure of the program is like its syntax - so to understand the structure of the program, you just need to read the textual markup of the program. If we add to the homoiconicity and small syntax the implementation of macros - instructions that tell the program exactly what actions to perform - we get a powerful programming language. [3]

The biggest plus of Clojure is its functionality and immutability of functions. You know exactly what the function does, and that there are no changes that it pulls with it. Therefore, when you are familiar with functional languages, and look at the code of a

large project, you can simply take a piece of it and know exactly what it does. It is not necessary to know what the surrounding code is doing. In traditional languages, the created code changes other code, so you always need to understand how parts of the program interact with each other, what certain pieces of code are responsible for, and how everything works inside the algorithms. That is why a debugger is constantly used in Java - after all, it is impossible to keep the entire program in your head, but you need to control everything and predict all possible changes in the code after the developer decides to change some line.

In Clojure, you can simply read the code and find out what it does, and when code executes there will be no unknown changes that it pulls with it. This is especially convenient when you work in a large team - the language itself allows you to create isolation between programmers and one piece of code cannot spoil another in any way.

Another plus is that you can use any libraries from Java and JavaScript in Clojure. This is a huge plus for the language - in itself, it is quite small and there are not very many people working on it. Therefore, it is great that third-party systems can be used.

It also implements two-way interaction with JVM-based libraries - Clojure code can use existing libraries and be called from other libraries, implement classes, and more. [1] Separately, it is worth noting support for working with arrays of Java objects - since they are not collections, Clojure has separate operations for working with arrays: creating, working with individual elements, converting from collections to arrays, etc.

Based on the programmer's experience today who worked both with Java and Clojure, they concluded that the same Clojure projects are created much faster. It takes less code to create a program, and the developer only solves the problem through the programming language. And not like in Java, where you must work a lot with the structure of the language, and not with the solution to the problem.

Programs made with Clojure have fewer bugs. When you come to a team that worked with an enterprise project, then it's much easier to deal with the code. This is where Clojure outperforms many languages, especially when dealing with large data systems.

Clojure is quite different from popular development languages, you need to get used to it, to its approach. Also, Clojure is a bit slower than other languages in some places but can be just as fast as Java. [13] Usually this is not a problem, but if you have few resources, then it is better to use something else.

Another downside is that Clojure only compiles to JVM or JS. If these platforms fit your project, then everything is perfect, if not, you will have to choose another language.

Clojure is a dynamic language. A matter of taste, but if this approach is not very close to you, and you like, for example, to make static variables, then it is also better to look for something else.

This relatively young programming language works best for the financial market, data analytics, and machine learning - areas where the focus is on data that needs to be processed quickly.

However, for the web and front-end, Clojure works great too, even though its own libraries are built on React. I know examples when developers even made games on Clojure - more precisely, the game logic was made on it, and everything else, of course, was written in Unity.

Clojure will grow rapidly and might replace Java, but not soon for sure. This programming language is currently used by companies that want to build large and complex systems that don't have to break and therefore is very well suited for full-stack development.

3. Applications of programming languages

Criteria/PL	Java	Clojure
Mobile development	Most of the popular Android apps, like the mobile clients of Telegram, Twitter, and Chrome, are written in Java. The language has native tooling support for the Android SDK, and Android development plugins are built into popular IDEs like IntelliJ IDEA and Eclipse.	There exists a version of Clojure that compiles to JavaScript and called ClojureScript. It can run in the browser and on the Node.js virtual machine. And using such a beautiful and powerful language for mobile development is a great choice.
Server development	One of the most popular web frameworks for server-side development, the Spring Framework, is written in Java. It is convenient to develop in Spring due to its high modularity.	Clojure became popular in server development. This programming language consists of all the essential features to develop server-side applications as Java.
Multithreaded distributed systems	Distributed systems are the trend of the last decade. Top companies are trying to move their development to the clouds, which run on server installations in remote data centers. There are several Java frameworks for distributed computing, such as Hadoop, Spark, and Kafka curated by Apache. They offer data stores optimized	Clojure can be used to create multithreaded distributed systems. The data structures are shared easily among threads because of their immutability. Many companies like Amazon, Apple, and Netflix use it today for their products.

	for searching and processing and efficient algorithms for working with them.	
Enterprise	<p>Java is used by major companies in the enterprise sector, including Microsoft, IBM, and many banks to develop and maintain internal applications. The language is popular because of its multi-platform, scalability, and variety of software tools.</p>	<p>Clojure provides many libraries with a lot of different features that can be used. Also, Clojure provides a toolbox that can be used to develop different prototypes very quickly while Java would require a lot of boilerplate code. Nowadays, many developers are increasingly using Clojure for enterprise applications, and its popularity is only going to increase.</p>
Game development	<p>Games are rarely programmed in Java because games require perfect optimization for the user's hardware. If the optimization is bad, then on most ordinary computers the game will not start. And that is what happens with Java, therefore you might consider C++ or C# for that. [7]</p>	<p>Clojure fits well for live-reloading development environments. Its performance is slower than any other game development environment. For example, developing some complex games like Call of Duty would not be a good decision, and therefore Clojure is like Java is not a good programming language to develop video games. [8]</p>
Machine Learning	<p>There are many programming languages used in the field of artificial intelligence and machine learning. But Java, despite its quarter-century history, is still one of the most popular in this area. It is also successfully used when programming neural networks, search algorithms, and robotics systems. For the purpose of creating machine learning algorithms, Java has two main advantages. This is the highest scalability, which means the ability to operate</p>	<p>Machine learning contains a lot of different models that require implementing complex algorithms and mathematical operations. Clojure is a functional programming language and programmers can easily describe those math operations for the following reasons: consistency, easy parallelism, and less code. Also, when implementing functions in deep learning algorithms using Clojure, we can use functional chaining.</p>

	<p>on huge data arrays and process multiple queries in parallel, and object-oriented programming, which sets the programming logic suitable for solving machine learning problems. The ability for applications built in Java to run on any platform also benefits. The language allows coding algorithms of various types; the code written on it is easy for subsequent debugging by third-party developers. [9]</p>	<p>Therefore, maintaining and reading Clojure code is much easier and fits better into machine learning than Java. [10]</p>
Cloud Computing	<p>Java runs in different cloud computing environments very well. A lot of programmers know Java and develop different Cloud solutions using this language. They prefer it due to ease of use, multiplatform, security, and robustness. Therefore, Java is one of the top languages for cloud computing. [4]</p>	<p>All cloud computing services support Java and therefore it also allows us to write code for any services in other JVM-based programming languages. Clojure is also supported by cloud computing services, but not popular because many programmers know Java, Python, and other languages that have the same or even better speed than Clojure. [6]</p>

4. Java and Clojure solutions for power set generation problem

Let's look at how Java and Clojure solve a power set generation problem and prove that they are correct by testing programs. But firstly, what is the power set generation problem? [5]

4.1 General idea

Given set $M = \{1, 2, 3\}$, then the subsets of M are:

- $\{\}$ (can be called the empty set or null set)
- $\{1\}$
- $\{2\}$
- $\{3\}$
- $\{1, 2\}$
- $\{1, 3\}$
- $\{2, 3\}$
- $\{1, 2, 3\}$

The algorithm should always be generating 2^n subsets, where n is the number of items in the original set. From the data above, we can conclude that the power set M is $\{\{\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$, and therefore we have $2^3=8$ sets generated.

4.2 Java implementation

Let's analyze a PowerSet.java program. Everything starts with defining a class that consists of two methods, the main method which is an entry point for executing a Java program, and the findPowerSet method which will be called from the main one. For a given set $M = \{1, 2, 3\}$, the power set can be found by generating all binary numbers between 0 and $2^n - 1$, where n is the size of the original set. For example, for the set $M = \{1, 2, 3\}$ we generate binary numbers from 0 to $2^3 - 1$, and for each number generated, the corresponding settings can be found by considering set bits in the number.

- $0 = 000 = \{\}$
- $1 = 001 = \{3\}$
- $2 = 010 = \{2\}$
- $3 = 011 = \{2, 3\}$
- $4 = 100 = \{1\}$
- $5 = 101 = \{1, 3\}$
- $6 = 110 = \{1, 2\}$
- $7 = 111 = \{1, 2, 3\}$

I am creating an integer array consisting of 3 numbers and putting it as an argument when calling a findPowerSet method so that the passed array could be used in a callee method.

In a findPowerSet method, firstly, calculates the total number of subsets that should be generated which is 8 for $M.length = 3$. Next follows the outer for-loop and the inner for-loop. Within every iteration of the outer for-loop new array list is created. The inner for-loop checks every bit of i , and if k 'th bit of i is set, print $M[k]$. So, whenever the inner for-loop populated that array list with items, it outputs the contents of the array list to the console, and the outer for-loop iterates again to generate another subset. It will happen again and again until i equals N which will cause a program to exit from the outer for-loop and end the program.

Let's look at iterations of the outer for-loop, inner for-loop, and what subsets will be generated. The following should provide a clear understanding of why there is such programmatic logic to solve a power set generation problem.

$i = 0; (000)$

$k = 0;$ $0 \& (1 \ll 0) \neq 0 \rightarrow false$

$k = 1;$ $0 \& (1 \ll 1) \neq 0 \rightarrow false$

$k = 2;$ $0 \& (1 \ll 2) \neq 0 \rightarrow false$

Output: `[]`

$i = 1; (001)$

$k = 0;$ $1 \& (1 \ll 0) \neq 0 \rightarrow true \rightarrow add [1] to the subset$

$k = 1; 1 \& (1 \ll 1) \neq 0 \rightarrow \text{false}$

$k = 2; 1 \& (1 \ll 2) \neq 0 \rightarrow \text{false}$

Output: [1]

$i = 2; (010)$

$k = 0; 2 \& (1 \ll 0) \neq 0 \rightarrow \text{false}$

$k = 1; 2 \& (1 \ll 1) \neq 0 \rightarrow \text{true} \rightarrow \text{add [2] to the subset}$

$k = 2; 2 \& (1 \ll 2) \neq 0 \rightarrow \text{false}$

Output: [2]

$i = 3; (011)$

$k = 0; 3 \& (1 \ll 0) \neq 0 \rightarrow \text{true} \rightarrow \text{add [1] to the subset}$

$k = 1; 3 \& (1 \ll 1) \neq 0 \rightarrow \text{true} \rightarrow \text{add [2] to the subset}$

$k = 2; 3 \& (1 \ll 2) \neq 0 \rightarrow \text{false}$

Output: [1, 2]

$i = 4; (100)$

$k = 0; 4 \& (1 \ll 0) \neq 0 \rightarrow \text{false}$

$k = 1; 4 \& (1 \ll 1) \neq 0 \rightarrow \text{false}$

$k = 2; 4 \& (1 \ll 2) \neq 0 \rightarrow \text{true} \rightarrow \text{add [3] to the subset}$

Output: [3]

$i = 5; (101)$

$k = 0; 5 \& (1 \ll 0) \neq 0 \rightarrow \text{true} \rightarrow \text{add [1] to the subset}$

$k = 1; 5 \& (1 \ll 1) \neq 0 \rightarrow \text{false}$

$k = 2; 5 \& (1 \ll 2) \neq 0 \rightarrow \text{true} \rightarrow \text{add [3] to the subset}$

Output: [1, 3]

$i = 6; (110)$

$k = 0; 6 \& (1 \ll 0) \neq 0 \rightarrow \text{false}$

$k = 1; 6 \& (1 \ll 1) \neq 0 \rightarrow \text{true} \rightarrow \text{add [2] to the subset}$

$k = 2; 6 \& (1 \ll 2) \neq 0 \rightarrow \text{true} \rightarrow \text{add [3] to the subset}$

Output: [2, 3]

$i = 7; (111)$

$k = 0; 7 \& (1 \ll 0) \neq 0 \rightarrow \text{true} \rightarrow \text{add [1] to the subset}$

$k = 1; 7 \& (1 \ll 1) \neq 0 \rightarrow \text{true} \rightarrow \text{add [2] to the subset}$

$k = 2; 7 \& (1 \ll 2) \neq 0 \rightarrow \text{true} \rightarrow \text{add [3] to the subset}$

Output: [1, 2, 3]

4.3 Clojure implementation

Define a function powerset with an input s. The valid starting point for every single solution is the set of empty sets, and then basically we want to build up on that set of empty sets. We are going to reduce the input set with an initial value of a set of empty sets and the reducer that is created will be called on every input. Then, we start with an accumulator, which is

going to be initialized with a set of empty sets. Write an *acc* (accumulator) and a *v* (value) for our reducing function. So, what we want to do is concatenate to our accumulator. Then, we are going to add to our accumulator the mapping of inserting our current value into every set of accumulators. So, we will map *conj* (a method conjoining multiple items (conj [1 2] 3 4) => [1 2 3 4]) on the current piece of the accumulator and we are going to count the value that we are reducing into that accumulator. Also, we turn a reducer into a *set* to match the format that Clojure wants. The next step one of the most important is to create a test function that will verify that the powerset function returns an expected output. As a result, I am getting back a successful response that the test-powerset function passed, and we are sure right now that the solution to the power set generation problem is correctly implemented. [11]

5. Conclusion

After analyzing Java and Clojure programming languages I concluded that they are very distinct languages. Java is imperative and is an OOP language, while Clojure is predominantly a functional language. Each of those programming languages can be used in any computer science area, but Java I would say is currently winning this race because of its redundancy, a lot of systems are developed and continue to use Java. This language is good for beginners, and almost everyone loves Java whoever learned it. Mobile, multithreaded systems, and enterprise are most applications that exist today written in Java and it is hard to say that Clojure will overcome Java in upcoming decades. Clojure is a functional language and to learn it you must have a different way of thinking about it. This is extremely complex to relocate your mind to start easily developing applications using a functional programming language even if you had a pretty much experience with imperative. Regarding the power set generation problem, I can conclude that it was very hard to transition from Java implementation to Clojure. It was not too easy to do as it would be to convert a power set generation program from Java to Python, or Java to C. But what was very surprising is that the solution in Clojure is very neat, and short and consists of several lines of code not including the test-powerset function. After reading this paper you might be interested to learn Clojure functional language. Every programmer knows at least one imperative language and starting to understand how functional programming languages work a big advantage is because it can open a new frontier of ideas in your mind. [12]

Abbreviations

JVM	Java Virtual Machine
RAM	Random Access Memory
OOP	Object-Oriented Programming
IT	Information Technology
ABCL	Armed Bear Common Lisp
JS	JavaScript
IDE	Integrated Development Environment

References

- [1] Clojure – Functional Programming for the JVM
<https://objectcomputing.com/resources/publications/sett/march-2009-clojure-functional-programming-for-the-jvm>
- [2] What is Java used for? <https://www.javatpoint.com/what-is-java-used-for>
- [3] Why use Clojure? <https://distantjob.com/blog/why-use-clojure/>
- [4] Java Cloud Development Introduction and Tools <https://howtodoinjava.com/cloud/java-cloud-development-introduction-and-tools/>
- [5] Power Set https://en.wikipedia.org/wiki/Power_set
- [6] Clojure in the Cloud <https://www.youtube.com/watch?v=6GE1XA-sbIE>
- [7] Why Is Java's Use in the Gaming Industry Limited?
<https://www.spiceworks.com/tech/devops/guest-article/why-is-javas-use-in-the-gaming-industry-limited/>
- [8] Game Development with Clojure/ClojureScript <https://lambdaisland.com/blog/2016-12-08-game-development-with-clojure-clojurescript>
- [9] Machine Learning with Java <https://www.youtube.com/watch?v=o8AgG8ivCmE>
- [10] Machine learning in Clojure with XGBoost and clj-boost
<https://towardsdatascience.com/machine-learning-clojure-xgboost-clj-boost-e0d1339df1e1>
- [11] ClojureDocs <https://clojuredocs.org/>
- [12] Clojure vs Java | What are the differences? <https://stackshare.io/stackups/clojure-vs-java>
- [13] A performance comparison of Clojure and Java <https://www.diva-portal.org/smash/get/diva2:1424342/FULLTEXT01.pdf>

A. Source code for the power generation set problem in Java and Clojure

- **Java**

I was running a Java program in an IntelliJ IDEA that already has preinstalled Java SDK, and to compile and run it just use a Run green button on top. The console will pop up with the result of the execution of the program.

PowerSet.java

```
import java.util.List;
import java.util.ArrayList;

public class PowerSet {
    public static void main(String[] args) {
        int[] M = {1, 2, 3};
        findPowerSet(M);
    }

    public static void findPowerSet(int[] M) {
        // calculate the total number of subsets
        int N = (int) Math.pow(2, M.length);

        // generate each subset one by one
        for (int i = 0; i < N; i++) {
            List<Integer> set = new ArrayList<>();

            // check every bit of `i`
            for (int k = 0; k < M.length; k++)
                // if k'th bit of `i` is set, print `M[k]`
                if ((i & (1 << k)) != 0)
                    set.add(M[k]);

            System.out.println(set);
        }
    }
}
```

Output:

```
[ ]
[1]
[2]
[1, 2]
[3]
[1, 3]
[2, 3]
[1, 2, 3]
```

- **Clojure**


I was running a Clojure program in an IntelliJ IDEA. A Cursive plugin should be installed first, and after that go ahead and create a Clojure project in IntelliJ, and you should be able to successfully create and run a powerset.clj file. The console will pop up with the result of the execution of the program.

powerset.clj

```
(ns powerset
  (:use (clojure set test)))

(defn powerset [s]
  (set (reduce (fn [acc v]
                 (concat acc (map #(conj % v) acc))) #{#{}} s)))

(deftest test-powerset
  (is (= #{#{}}
        (powerset #{})))
  (is (= #{#{}} #{1}
        (powerset #{1})))
  (is (= #{#{}} #{1} #{2} #{1 2}
        (powerset #{1 2})))
  (is (= #{#{}} #{1} #{2} #{1 2} #{3} #{1 3} #{2 3} #{1 2 3}
        (powerset #{1 2 3}))))
```

 Tests passed: 1 of 1 test – 4ms