Performance optimization when using Object Oriented Development

Joan I. Krastanov

Fontys University of Applied Sciences

Author Note

Correspondence concerning this article should be addressed to Joan Krastanov, Software Engineering, Fontys University, Eindhoven.

Contact: j.krastanov@student.fontys.nl

Table of Contents

Abstract

Software optimization is becoming more and more important for companies and manufacturers due to the ever-increasing competitiveness of the Information and Communication Technology (ICT) sphere. Software developers are trying to improve the performance of their products so that they can keep up with the competitive standard of the market which is increasing drastically every year. There are many models and methods which can be used to boost the performance of a software product. This paper explores different programming languages and methods used for back-end development when creating software solutions using the Object-Oriented Development (OOD) method. The aim is to find the most optimized back end in order to maximize the performance of the application or website which, in turn will lead to a better user experience. It is worth mentioning that some optimizations are so minor that in most cases they can be neglected, however when trying to produce a product that is as optimized as possible, this research will present the best ways to achieve that goal based on a number of different sources and comparisons.

Keywords: Information and Communication Technologies, Object-Oriented Development

1. Application back-end – What is it?

Every software application consists of two major components: the front end and the back end. The front end or the presentation layer is what the user sees and interacts with, while the back end or the service layer is something that the user does not have direct access to. It is where all the logic and calculations of the application are performed which includes, but is not limited to algorithms, calculations and data processing and modifications.

The back end comprises of three main parts[1]:

- Server - used to process requests and manage network resources.
- Database – used to store application data in the form of tables.
- Application – the application code itself

Since the back end is where all the calculations occur, it is also the layer which can benefit the most from different optimization techniques. This article is aimed to help developers improve their back-end applications by looking at a number of different factors from programing language to application structure and data type usage.

2. Optimization in Software Engineering

There are two different types of optimization types and each one of them focuses on a certain aspect of the code which is aimed to be optimized. [2]

2.1 Space Optimization

2.1.1 Usage of Appropriate Data Types

Using the appropriate data types for the intended feature will not only ensure an expected outcome when the code is run, but it will also improve memory utilization and be less demanding on the CPU in general.

2.1.2 Data Arrangement

It is important to declare similar sized variables in clusters since it will limit the amount of "dummy" area within the structure of the application. Following this approach will improve RAM, but it also might decrease the operational speed of the solution.

2.1.3 Return Value

All return values of functions/methods are stored in registers which if not utilized will take up unnecessary space and time. For that reason it is highly advised to make sure that all methods that do not return any meaningful value are defined as "void".

2.2 Time Optimization

2.2.1 Optimization of Algorithms

Frequently used algorithms should be selected and improved with performance in mind in order to maximize the efficiency of the application and minimize wait time. Needless to say this is a crucial part when trying to optimize a solution and should never be neglected

2.2.2 Avoiding type conversion

When looking to maximize performance, type conversion should be limited to a minimum since it uses more resources and in turn makes the application slower.

2.2.3 Loop Related Optimization

If a loop takes up a lot of cycles in the code, it might be worth trying to redesign the code to ensure lesser loop execution count. There are many optimization techniques which can be used to achieve that, but they are beyond the scope of this article (Loop Optimization Techniques).

3. Which is the fastest programming language?

While there are many other ways to optimize a software application, the language which is used can also make a significant difference when trying to make the most optimized product.

3.1 Types of programming languages [3]

- Procedural Programming languages – executing a sequence of statements, often using multiple variables and heavy loops, which lead to a result.
- Functional Programming languages – typically use stored data and avoid loops in favor of recursive functions.
- Object-oriented Programming languages – viewing the world as a group of objects that have internal data which can be accessed with external accessing parts.
- Scripting Programming languages – they may comprise of object-oriented language elements but fall into their own category as they are normally not full-fledged programming languages but are rather used for automation of execution of certain tasks.
- Logic Programming languages - let programmers make declarative statements and then allow the machine to reason about the consequences of those statements. In a sense, this language doesn't tell the computer how to do something, but employing restrictions on what it must consider doing

This article is focused on OOD so it will compare only this type of languages.

3.2 OOD Languages and their performance [4]

For the purpose of this research the top six OOD languages will be compared in different tests(OOD Language Performance Tests) and the results will be analyzed to find out which is the best performing language.

3.2.1 Regex-redux

Source	seconds	memory	gz	busy	cpu load
Java	5.31	793,572	929	17,50	79% 78% 83% 89%
C++	**1.10**	203,924	1315	3.43	63% 77% 71% 100%
C#	1.42	280,892	1869	2.69	**30% 39% 86% 35%**
Ruby	9.66	277,900	724	21.43	44% 96% 38% 43%
Python	1.36	**111,852**	1403	2.64	32% 40% 33% 88%
PHP	1.73	162,348	816	3.58	37% 88% 44% 39%

3.2.2 Spectral norm

Source	secs	mem	gz	busy	cpu load
Java	1.58	39,408	756	5.97	94% 94% 96% 94%
C++	**0.72**	**1,192**	1044	2.85	99% 99% 100% 100%
C#	0.82	35,048	764	2.99	89% 96% 89% 89%
Ruby	118.16	24,764	326	120.08	**0% 1% 100% 0%**
Python	120.99	13,424	407	7 min	99% 99% 99% 99%
PHP	7.51	35,364	1152	29.55	99% 98% 98% 98

3.2.3 Binary trees

Source	secs	mem	gz	busy	cpu load
Java	2.48	1,725,776	835	7.86	74% 75% 97% 72%
C++	**0.94**	**176,428**	1122	3.39	86% 88% 100% 85%
C#	4.81	1,881,564	676	15.47	81% 74% 83% 83%
Ruby	23.80	566,560	1008	67.07	**58% 96% 63% 64%**
Python	48.03	462,732	472	174.44	89% 97% 88% 89%
PHP	18.64	1,588,704	760	67.29	89% 90% 96% 86%

3.2.4 Pi digits

Source	secs	mem	gz	busy	cpu load
Java	0.93	36,088	764	0.97	4% 0% 1% 99%
C++	**0.66**	**5,152**	986	2.63	100% 100% 100% 100%
C#	0.92	35,404	977	0.96	98% 3% 2% 1%
Ruby	2.11	541,320	485	4.16	53% 6% 100% 39%
Python	1.28	12,024	567	1.29	**0% 1% 100% 0%**
PHP	0.91	13,196	399	0.96	2% 0% 3% 100%

It is visible that C++ is the language that outperforms the rest timewise in all the tests shown and sometimes by considerable margins. However if CPU load is a concern, C# or Ruby might be a better choice. In terms of memory C++ performed the best yet again where in some cases it performed 100 times better than some of the other languages.

(Figure 1)(Figure 2)(Figure 3)

4. Conclusion and Recommendations

In conclusion, clearly the best performing back-end language based on the tests shown above is C++, sometimes outperforming the competition by considerable margins. It is a language that has withstood the test of time and is still a huge part of the software world without showing any signs of slowing down or being replaced by another language.

With all that being said I have still chosen to use Java as my back-end language for the individual track project, because the performance of the language still beats a big part of the competition and for me personally is a language that is a lot easier to learn and use.

When it comes to the performance optimization of different loops, I am going to try to implement some of these techniques in order to maximize the performance of my application, if applicable, but even if I do not get the opportunity to implement them, they have still provided me with some knowledge on optimization which is going to be very useful for my developing career as a software engineer.

5. Recommended further reading

5.1 **Loop Optimization Techniques**

5.2 **OOD Language Performance Tests**

References

[1] Back-End Web-Application Development and the Role of an Admin. (2017, September). VISHESH S. https://ijarcce.com/upload/2017/september-17/IJARCCE%2011.pdf

[2] Upadhyay, R., Nagarbandhara, J., & Bhatt, S. (2017, July). A practical approach to optimize code implementation. https://www.einfochips.com/wp-content/uploads/resources/a-practical-approach-to-optimize-code-implementation.pdf

[3] A. (2021, January 5). Types and Differences between Programming Languages. WatElectronics.Com. https://www.watelectronics.com/types-of-programming-languages-with-differences/

[4] Gouy, I. (2021, September 24). Which programming language is fastest? | Computer Language Benchmarks Game. BenchmarksGame. https://benchmarksgame-team.pages.debian.net/benchmarksgame

Figures

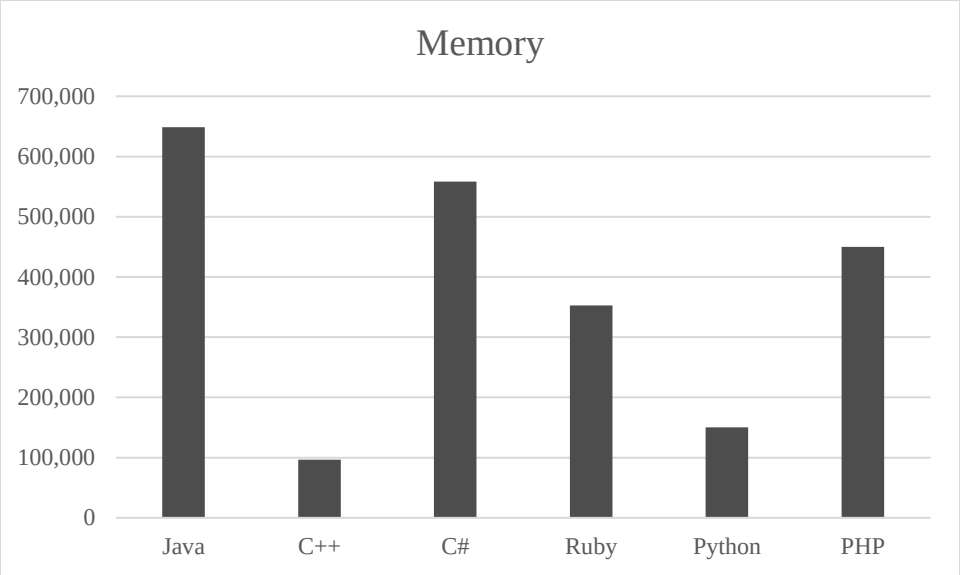

Figure 1: Average memory usage per language for the given tasks

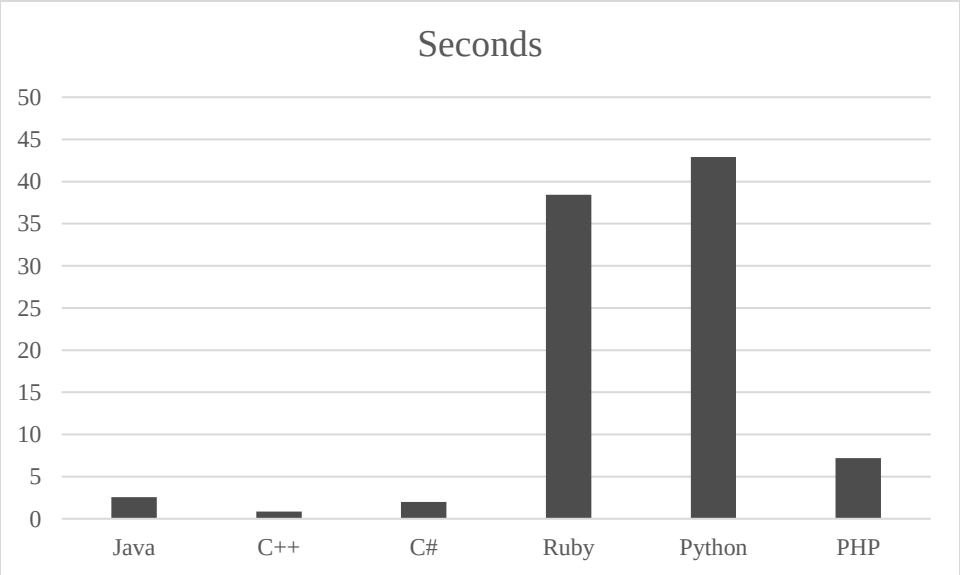

Figure 2 : Average time of completion per language for the given tasks

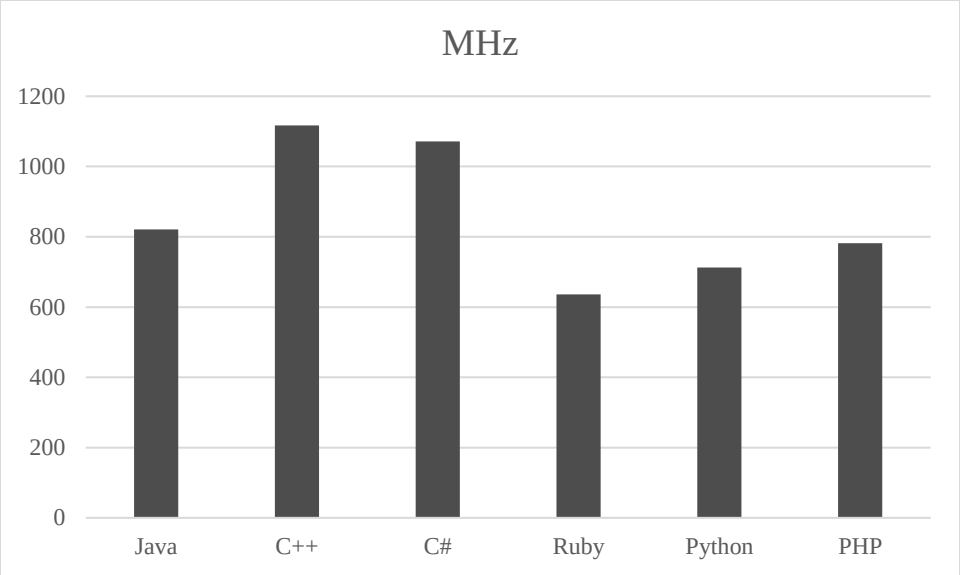

Figure 3 : Average CPU usage per language for the given tasks