Table of Contents

Introduction

Get the PDF!

I've been in a Java world for years. Basically after finishing university and switching from beloved, yet platform-bound C#/.NET environment (back in the days when Mono was not really competitive and .NET Core didn't exist) to Java - I became a fan of the Java platform and really liked it. The language was verbose, yet still simple - no pointers, no caring about references or memory management (I got some experience with C++ as well. Let's not talk about this mental breakdown). The ecosystem was huge! Lots of new libraries to choose from, web-oriented JavaEE with the simple and universal Servlet API, JSPs, JSFs and other stuff.

I started experimenting with other JDK languages. Groovy was a great starting point, Clojure opened my mind to functional programming and Kotlin was a well-deserved fresh wind that brought new syntax sugar to the JDK platform.

Later on - I was hardly a programmer - I was just putting blocks of frameworks together and producing universal web applications in a no time. And in parallel - the popularity of cloud-native applications risen. Java community started to do a weird things for Javists - compiling native images, getting rid of classic JVM.

And a wild competitor appeared - Go. Also known as "Golang".

What's so great about this new kind on the block? Why its becoming more and more popular. I decided to figure out - my next project was in Go.

And this is the output of my experience with it. The good, bad and ugly things.

If you are (not limited to) Java developer and want to learn Go or just find out whats the big fuzz about it - read this "book". It might give you some answers.

Chapter One

or setting everything up

Getting the stuff

Where should we Go?

Ok, so you've decided that you you'd like to write and run some Go code. GOOD!

Now we just need to get it.

In Java world, there are currently multiple distributions of a development kit. Do you want a Java from Oracle, Amazon,RedHat or Microsoft? Or perhaps from Zulu or AdoptOpenJDK? Choose wisely, depending on the runtime environment and licence conditions. Then you got multiple versions. There is a latest version, LTS (Long-term-support) version, some obsolete LTS versions, that people still use. Plus you can choose which JVM you want to run - are you a HotSpot or OpenJ9 guy?

I know, I'm kinda exaggerating right now. In most cases you pick your favorite one or you are told by your superiors in your work. And when you really don't know which one to choose, you just go for the latest AdoptOpenJDK build.

Well, good news. You do not have to worry about this decision in the Go world.

Just go to https://golang.org/dl/ and download the latest Go for your platform.

Installation is not complicated, as you can see on the Go install instructions: https://golang.org/doc/install - either run an installer or just extract a package and update the PATH variable.

Go authors are really careful about backward compatibility. Meaning, that code that is written for older version of Go runtime will run on the newer one. So you can find some archaic libs and apps that are still functional.

Running the IDE

Ladies and gentlemen, start your engines!