

Bachel Thesis 2020 **DRAFT**

Ramon Rüttimann

# Functional Go

## Situation

With the rise of Javascript, Rust, and Go, the functional programming paradigm has gained popularity too. Though none of these programming languages are purely functional,they all share the common feature of having the possibility to use functional concepts.  
However, a lot of programmers struggle initially with the concept of functional programming. Learning a purely functional programming language is extremely useful to gain familiarity with these concepts. Purely functional programming languages like Haskell though are not known for their beginner-friendliness. That makes learning a functional language difficult is that not only does the programmer have to learn an entirely different paradigm, but also a syntax that is uncommon for people coming from imperative or object-oriented languages.

## Objective

The objective is to ease the entry into functional programming by providing a "harness" for Go that enforces a purely functional style. This harness can either be a separate stand-alone tool (like `gofmt` or `gopls`), some static code analysis tool (like `gosec`), or be built into the Go compiler directly.   
In most cases, functional Go code should still be valid Go code, but the harness could also apply simple transformation rules to make the code more performant.   
Depending of the “level” of that harness (static code analysis vs built into the compiler vs transpiling), there would be different things to consider. Common problems that are to be solved however would be, for example, immutability (without introducing unnecessary complications) or how to handle pointers (or forbid them completely).  
If a transpiler would be chosen, another issue to consider is the possibility to make functional Go code run and / or compile with regular Go code.  
Furthermore, as Go does not support polymorphism (yet, this is a feature that should be coming with Go 2), a “standard library” of functions could be implemented in the compiler. Currently, data structures like maps are implemented within the compiler to make it possible to do static type checks. Support for standard functions, for example “map”, “reduce”, “filter” etc., could be added to the compiler.

In the end, functional Go should be syntactically familiar to people that have worked with Go (or C in that regard). With that, one can learn the concepts related to functional programming, without also needing to learn a new language and syntax.

## Submission Date

Friday, July 5th, 2020

Winterthur, February 10th, 2020

Gerrit Burkert, Karl Rege