Golang YES!

A Preprint

Jiacheng Yang
Department of Computer Science
The University of Arizona
jiachengyang@email.arizona

Wenkai Zheng
Department of Computer Science
The University of Arizona
wenkaizheng@email.arizona

April 6, 2020

Abstract

abstract Golang

***K*eywords** Go · Golang

1 Introduction

Introduction XXX

2 History

Golang was designed by Rob Pike, Ken Thompson, and Robert Griesemer in order to achieve fast execution speed and high efficiency of development. In comparison to popular compiled languages like C, C++, and Java, programs written in Golang are only slightly slower than the same programs written in C and C++ but faster than the ones written in Java, while it needs much less time to compile source code. Another feature that Golang offers is the ability to easily writing concurrency programs. The keyword "go" enables programmers to create a go-routine, which is similar to but costs less than a thread in Java or C++.

3 Control Structures

Golang has four kinds of control structures: if-else, switch, for-loop, and defer. If-else, switch, and for-loop work very similar in C. The key-word, defer, is designed for error handling and cleaning up opened resources.

3.1 Control Structures: if-else

If-else in Golang works just like in C with two minor differences. One is the removal of parentheses around the condition statement as the code below.

```
if variable%2 == 0 {
    fmt.Println("Variable is even")
} else {
    fmt.Println("Variable is odd")
}
```

The other difference is that variables can be defined in the same line of condition statement as the code below. This feature is useful when the condition statement and the body of if-else both need the same value returned by another function. Programmers can just declare a variable to receive it and use it in both places.

```
if x:= somefunction(); x%2 == 0 {
    fmt.Print(x)
    fmt.Println(" is even")
} else {
    fmt.Print(x)
    fmt.Println(" is odd")
}
```

This feature also benefits error handling. For example, when handling errors, since errors in Golang are returned by functions, callers can declare variables to receive the return values and then check errors in the condition statement. After that, errors can be handled in the body of if-else. This usage of if-else avoids declaring variables in the name-space outside of the if-else.

```
reader := bufio.NewReader(os.Stdin)
if x, err := reader.ReadString('\n'); err==nil {
        fmt.Println("read: "+ x)
} else {
        fmt.Println("fail")
}
```

3.2 Control Structures: switch

Switch in Golang is also slightly different from switch in C. One difference is that programmers no longer need to explicitly use break to break from case as the code below.

```
switch i {
case 0:
        fmt.Println("case 0")
case 1:
        fmt.Println(" case 1")
default:
        fmt.Println("default")
}
```

In case that programmers need to combine cases, Golang provides two ways; One is the key-word, fallthrough. Fallthrough allows programs to execute the case below the current one. For example, if x in the code below is 2 or 3, the program will still print: "x is a prime less than 6."

```
switch x {
case 2:
        fallthrough
case 3:
        fallthrough
case 5:
        fmt.Println("x is a prime less than 6.")
}
```

The other way is to combine expected values into a single case as the code below.

```
switch x {
case 2,3,5,7:
        fmt.Println("x is a prime less than 10")
default:
        fmt.Println("x is not a prime less than 10")
}
```

If a variable for comparison is not provided, switch in Golang then works like multiple if-else. Each case is evaluated from top to bottom, and the one evaluated to true will be executed.

```
switch {
case variable%2 == 0:
        fmt.Println("variable is divisible by 2")
case variable%3 == 0:
        fmt.Println("variable is divisible by 3")
default:
        fmt.Println("variable is not divisible by 2 or 3")
}
```

3.3 Control Structures: for-loop

3.4 Control Structures: defer

3.5 Control Structures: label and goto

4 Data Types

Data Types xxx

5 Subprograms

Subprograms xxx

6 Summary

Summary xxx

References

References xxx