Go Abstract Syntax Tree Visualizer

Christian Bergum Bergersen
chrisbbe@ifi.uio.no

September 8, 2015

Abstract

This article documents a simple tool developed in Go to draw the abstract syntax tree for Go programs. The abstract syntax tree is drawn with Grapvizer for simplizity.

1 Implementation

To keep track of the nodes in the tree, the program uses a stack to `Pop` and `Push` nodes on the stack, `Push` happens each time the program dives a level deeper in the tree, `Pop` happens when the program has to go up one level, this way we can track the relationships between a parent and child node. The `stack` implementation is taken from chapter **1.5 Stack-Custom Types with Methods** in [1].

2 How to use and results

Even small programs has a pretty big abstrac syntax tree, which makes it increansingly harder to visualize all nodes of the abstract syntax tree as the code grows, its best illustrated with an visualization of the abstract syntax three for the standard `HelloWorld.go` program as listed under.

```go
package main

import "fmt"

func main() {
        fmt.Println("Hello World")
}
```

Listing 1: HelloWorld in Go

3 Known problems

The following list contains known problems and potential improvments for the code.

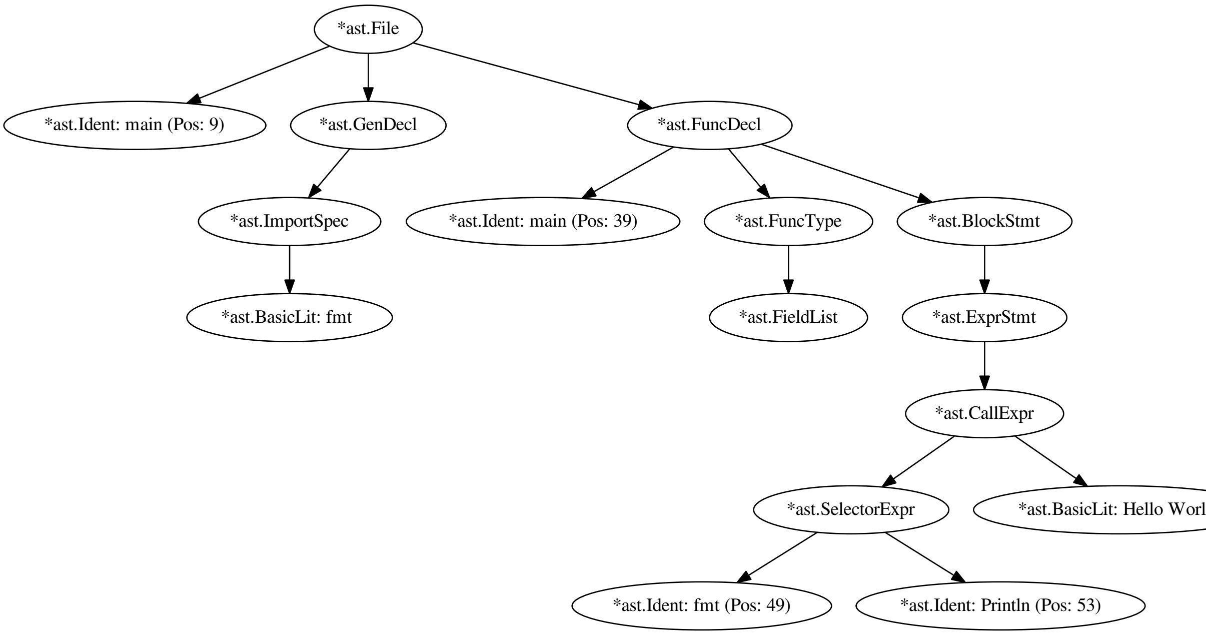

Figure 1: Abstract Syntax Tree visualized for the small "HelloWorld.go" Go program.

1. The generating of Dotty code doe's not know if there already exist a relationship between two nodes, resulting in some situations the graph to have multiple arrows between two or more nodes, which does not have any semantic meaning, but syntactically ugly.

References

[1] Mark Summerfield. *Programming in Go: Creating Application for the 21st Century.* Addison-Wesley Professional, 2012.