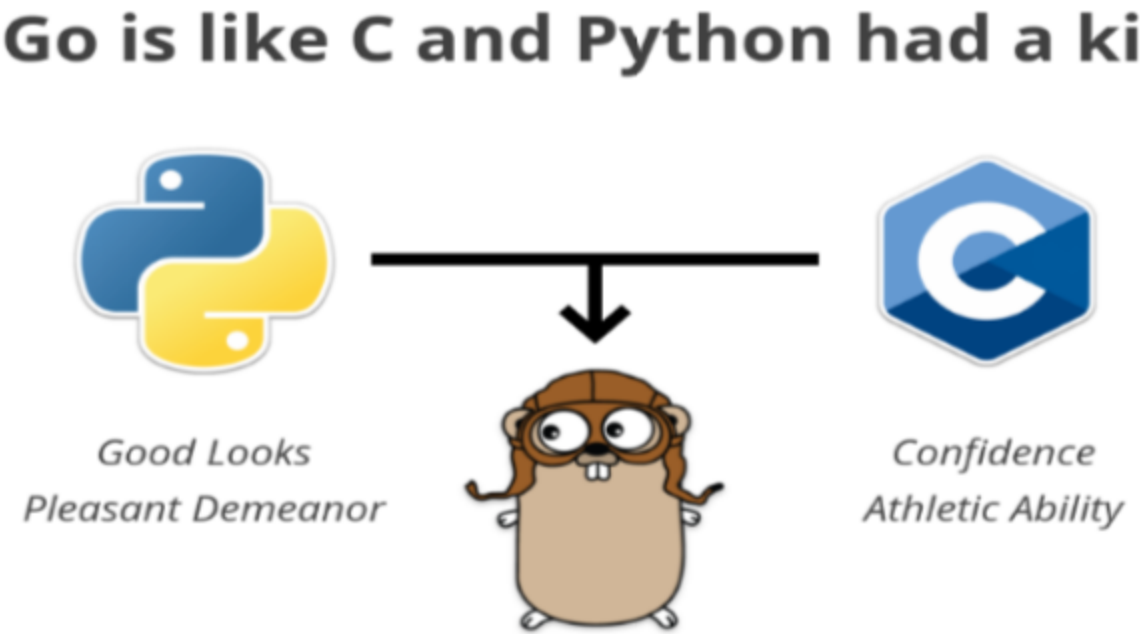
# Go for Javascript developers!

2 February 2019

Rashmi Nagpal
Software Engineer

## Introduction

- Go is like C and Python had a kid, who inherited Python's good looks and pleasant demeanor, along with C's confidence and athletic ability.



# What is GO?

- Go is an open-source, compiled, garbage-collected, concurrent system programming language.
- **Productive**

```
Go posess clean and elegant syntax, unobtructive static typing, lightning-fast compilation.
```

- **Fast**

```
Go is blazing fast.Go has a young compiler with minimal speed optimizations.
```

- **Optinionated**

```
Go has a strict compiler,things like - unused imports or variables-are hard compile errors in Go.
```

- **No OOP**

```
Go is structured in such a way that class,inheritance & polymorphism isn't possible.
```

## Similarities

- Like Javascript, Go uses Garbage Collection.
- Variables and functions have a scope.
- In similar fashion, we define variables, structures, functions and do **for** loops and **if** statements.

## Differences

- Javascript is based on main thread, which handles event loop & several other threads which do external IO.While in Go, concurrency is the **KING**!
- JavaScript is interpreted, and code is compiled just before it runs while Go is compiled.
- Go is better suited for returning **multiple return values** i.e. a function can respond back with more granular errors & avoid inconsistent and hard to maintain systems like Javascript objects.

# Syntax

## Basic rules

- Lines don't end with a semicolon.
- Last element in the array must have , after the value.

```
var arr =  [3]int{
  1,
  2,
  3,
}
```

## Basic types

```
var num int =  5
var pi float = 3.14
var isActive bool = true

//short version using inference
num := 5
```

## Loops

```
//for
for i:= 0;i<100;i++{
    sum += i
}

//while
for sum < 100{
    sum += sum
}

//infinite loop
for {
}
```

## Flow control

```
//no paranthesis
if age < 18{
    return true
}else{
    return false
}
```

# Advanced Features

## Goroutines

- A lightweight thread managed by the Go runtime.

```
go f(a, b, c) //starts a new goroutine.
```

- Goroutines run in the same address space, so access to shared memory must be synchronized.

## Channels

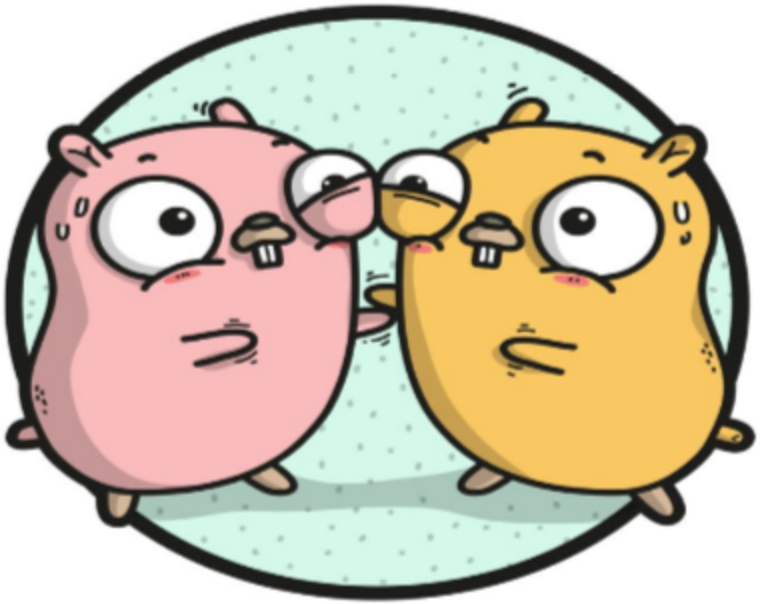
- Channels can be thought as pipes using which Goroutines communicate.
- Each channel has a type associated with it.

```
//send v to channel ch
ch <- v

//receive from ch, and assign value to v
v := <- ch
```

## Personal Takeaway

- It is never too late to start something new, all it takes is: patience . practice . perseverence.



# Thank you

Rashmi Nagpal
Software Engineer
@sparkle_rash
http://github.com/RN0311