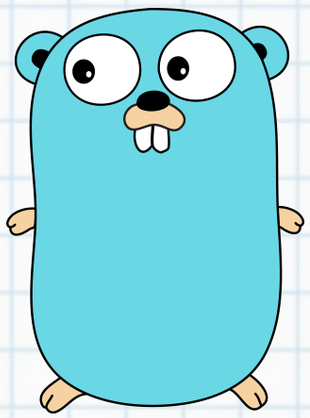
03/07/2017

Go
for non-gophers

Natalie Pistunovich
@nataliepis

Edward Medvedev



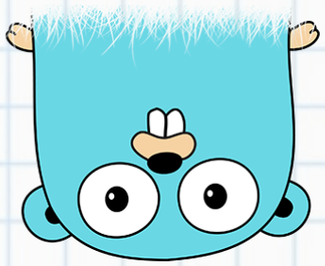
IS Lab, KeMU Hub

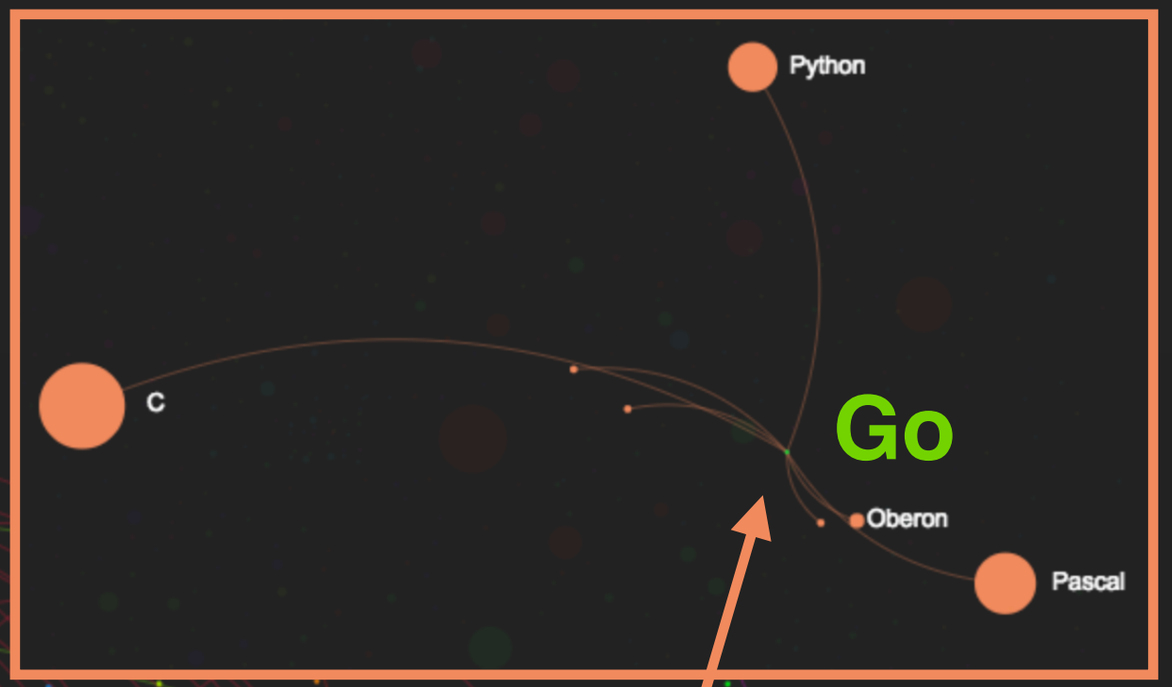
Workshop plan

Part 1: Intro to Go (Theory)

Part 2: A Tour of Go (Practice)

Intro to Go





Logo
Smalltalk
Forth
Self
Ruby
Python
Lisp
Dylan
Objective-C
Lua
AWK
Common Lisp
T
JavaScript
Perl
Simula
Communicating sequential processes
Eiffel
CLU
ALGOL 68
C++
Scheme
SQL
C
Java
Ada
Object Pascal
Oberon
APL
COBOL
C#
Modula-3
BCPL
Modula-2
Pascal
Prolog
Erlang
FP
ALGOL
PL/I
Speedcoding

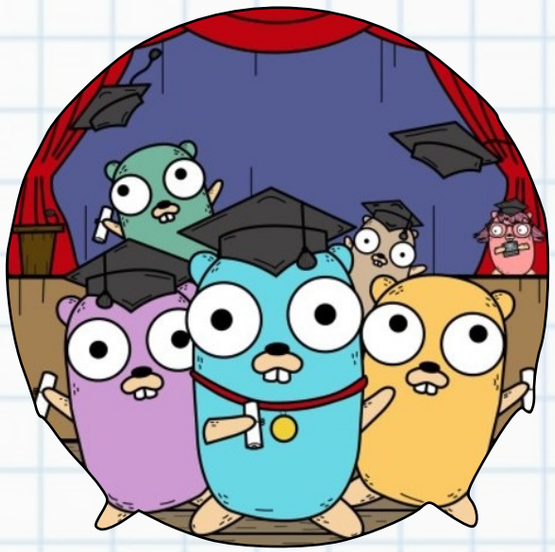
https://exploringdata.github.io/vis/programming-languages-influence-network/

About go

Go is a free and open source programming language created at Google in 2007.

v1.9 released on 15/06.

It's a **system programming** language,
that is **compiled**,
and **statically typed**,
with a **C-like** syntax,
garbage collection,
and explicit support for **concurrency**.



gopheracademy.com

More about go

Programs are constructed from **packages**,
whose properties allow efficient management of
dependencies.

The existing implementations use a traditional
compile/link model to generate executable binaries.

The grammar is compact and regular,
allowing easy analysis by automatic tools
such as **IDEs and code linters**:
https://github.com/golang/go/wiki/
IDEsAndTextEditorPlugins

Interfaces and inheritance

“Understand the power of interfaces, they are one of Go’s great gifts, potentially more important than channels or coroutines”

“By removing inheritance from the language, the opportunity to practice the mantra of composition over inheritance is made manifest, and fighting it will only lead to frustration.

Make things small and compose them together, instead of trying to make inheritance hierarchies.”

Go conventions

“Try and avoid writing code like you would in your current language of choice. Spend some time to learn the “Go way”, learn what idiomatic Go looks like and what are the Go conventions, especially those related to documentation, packages and naming.“

“Use the go tool chain. Write tests. Use interfaces. Take version numbers seriously. Document your code for godoc.”

go fmt

“Learning to use closures effectively is also a huge win as they can be used to write very elegant code.”

“Go adopts a unique programming paradigm, so be prepared to change the way you design and write code. Try to avoid the use of third party libraries and/or frameworks, at least while you are learning: Go’s standard library has so much to offer.”

Concurrency

“Don’t get too excited about channels. They’re awesome but easily over leveraged. Focus instead on learning how to leverage composition and interfaces to create clean and robust code.”

“Beginners make the mistake of assuming that goroutines allow you to do more things in parallel, which is not always the case.

If you’re deeply interested in concurrency study the standard library’s usage. I have learned a ton from the ‘net/http’ package.”

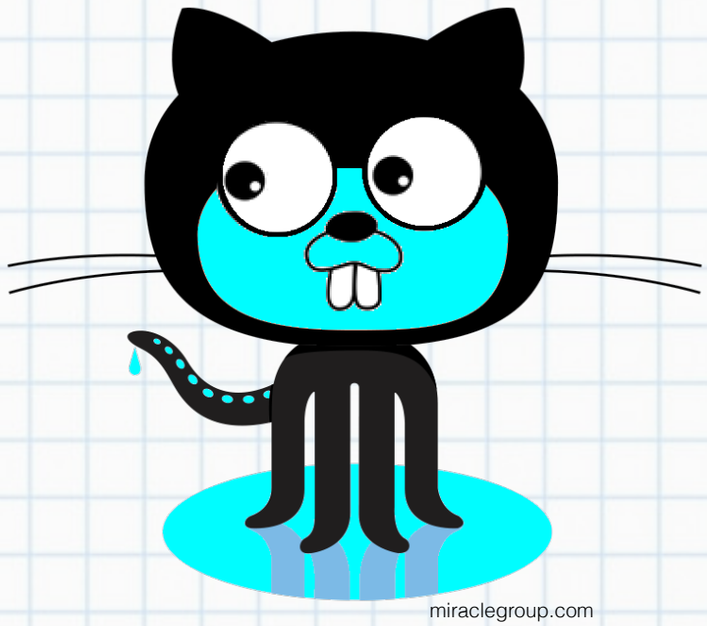
Concurrency vs. Parallelism

Concurrency: programming as the composition of independently executing processes

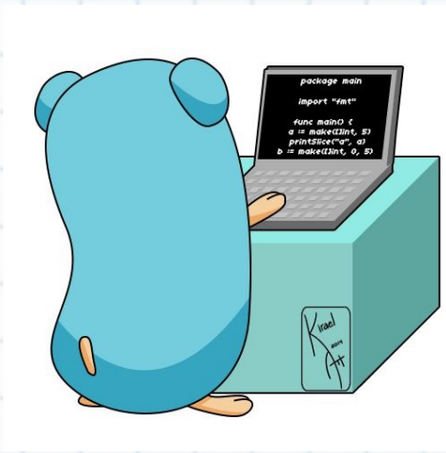
Parallelism: programming as the simultaneous execution of (possibly related) computations

Concurrency is about **dealing** with lots of things at once.
Parallelism is about **doing** lots of things at once.

A Tour of Go

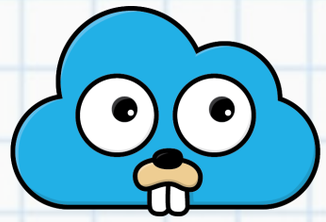


Summary


package main
import "fmt"
func main() {
a := make([]int, 5)
printSlice("a", a)
b := make([]int, 0, 5)


Go!

Wrap up



Kenya is missing a **Go User Group** :(
https://github.com/golang/go/wiki/GoUserGroups

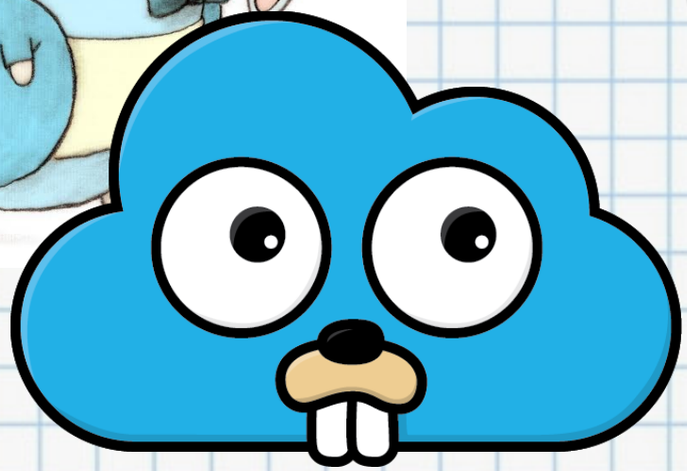
Or other references in the **Go Community**
https://github.com/golang/go/wiki#the-go-community

But there are lots of **projects** looking for love!
https://github.com/golang/go/wiki/Projects

And some cool **conferences** to attend
https://github.com/golang/go/wiki/Conferences

THE END

Reference pics



https://go-traps

go stockholm