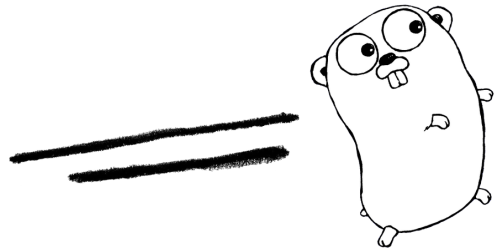
Die Programmiersprache Go - Eine Einführung

Seminarvortrag

Student: Adrian Helberg
Prüfer: Prof. Dr. Axel Schmolitzky

21. März 2018



“Go is an open source programming language that makes it easy to build simple, reliable and efficient software.”

(Go Website: golang.org)

Go ist eine Open-Source-Programmiersprache, die es einfach macht, einfache, zuverlässige und effiziente Software zu erstellen.

(Eigene Übersetzung)

Entwickler

- Konzipiert September 2007
- Robert Griesemer, Rob Pike und Ken Thompson
- Mitarbeiter von Google LLC. ®
- Aus Frust heraus entstanden
- *"Complexity is multiplicative"* - Rob Pike

Entwurfsphase

- Ausdrucksstarke und effiziente Kombination aus Kompilierung uns Ausführung
- Ähnlichkeiten mit C
- Adaptiert gute Ideen aus einigen Programmiersprachen: Pascal, Modula-2, Oberon, Oberon-2, Alef, ...

Entwurfsphase

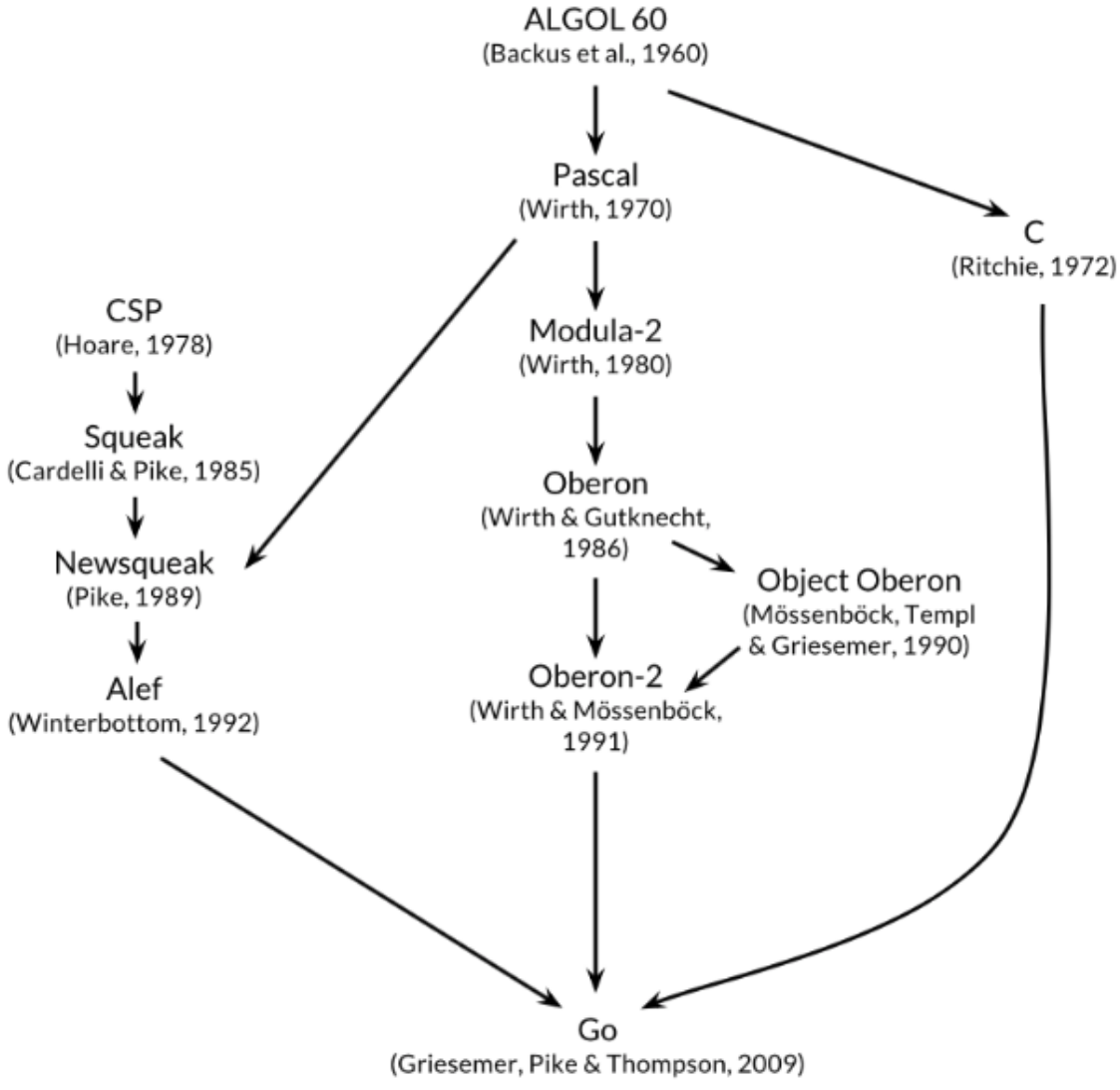


Abbildung: The Go Programming Language, Preface xii

Entwurfsphase

- Vermeiden von features, die zu komplexen, unzuverlässigen code führen würden
- Möglichkeiten zur Nebenläufigkeit sind neu und effizient
- Datenabstraktion und Objektorientierung sind ungewohnt flexibel
- Automatische Speicherverwaltung (garbage collection)

Veröffentlichung

- Vorgestellt November 2009

- Berühmt als Nachfolger für nicht typisierte Sriptsprachen
  $\rightarrow$ Verbindung aus Ausdruckskraft und Sicherheit

Go-Community

- Open-source projekt

  $\rightarrow$ Quellcode des Compilers, Bibliotheken (libraries) und Tools sind frei verfügbar

- Aktive, weltweite Community

- Läuft auf Unix, Mac und Windows

  $\rightarrow$ Üblicherweise ohne Modifikation transpotrierbar

Überblick

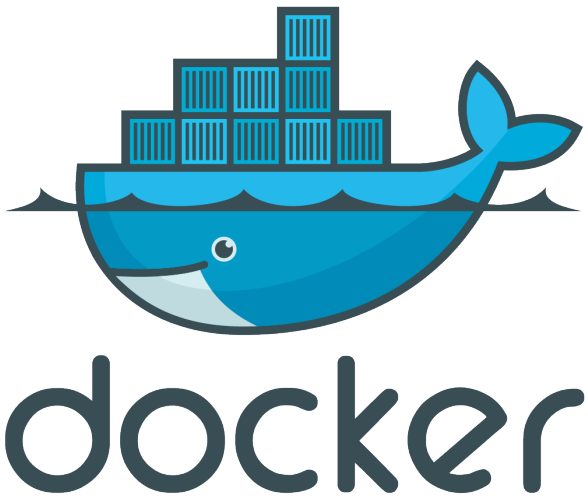
- Motivation
- Zielsetzung
- Skalierbarkeit
- Einsatzgebiete
- Erster Vergleich mit Java™

Merkmale

- Closures
- Reflexion
- Typsicherheit
- Autmoatische Speicherbereinigung
- Interfaces, Mixins $\rightarrow$ Objektorientierung
- Keine Klassen $\rightarrow$ Java Vergleich
- Pakete

Projekte

Go



Java

elasticsearch

Syntax

Go

```go
package main

import "fmt"

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

Java

```java
public class HelloWorld
{
    public static void main (String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

Paradigmen

- Nebenläufig
- Imperativ
- Strukturiert
- Modular
- Objektorientiert

Zentrale Fragestellung

„flexibel wie dynamisch getyped,
aber mit statischer Typsicherheit?“

Pro & Contra

Pro	Contra
▶ Minimalismus	▶ Keine generische Programmierung
▶ Statisches Duck-Typing	▶ nil statt Option
▶ Parallelisierung	▶ Wenig grundlegende Datenstrukturen
▶ Aufgeräumte Syntax	▶ Keine Methodenüberladung
▶ Schneller Compiler	

Pro & Contra

Pro	Contra
▶ Statisch gelinkte Binärdateien	▶ Unbefriedigende API-Dokumentation
▶ Laufzeiteigenschaften	▶ Teilweise umständliche APIs
▶ Integriertes Unit-Test-Framework	▶ Umständliches Mocking
▶ Paketmanager	▶ Kleines Ökosystem

Compiler

- Gc
- Gccgo

Einzelnachweise

- ...