Analysis of the Implementation of DockAlt

Kacey Ryan

UCLA

CS 131

What is Docker?

- Utilizes Linux Containers to create a “Separation of Concerns”
- Works similar to chroot in creating virtual working environment
- Creates a runtime tool to be used inside the containers providing tools (push, pull, run, commit)
- Designed with Go

Our Task

- Alternative Docker designed with a different platform (in case problems crop up)
- Decide on the best alternative design platform:
  - Java?
  - Python?
  - Haxe?
- Compare the alternatives to Go and each other for best analysis

Why did Docker Choose Go?

- Static Compilation (Simplicity in Production)
- Neutral (Allowed them to choose necessary features)
  - Asychronous Primitives
  - Low-level Interfaces
  - Extensive Standard Libraries
  - Duck Typing
- Full Development Environment
- Multi-arch building

Should Java replace Go?

Pros

- **Speed!** (One of the faster languages)
- **Stability and Reliability**
  - Low-level interfaces
- **Large Libraries and Community Support**
- Good Support for **Multithreading**
  - Asynchronous primitives

Should Java replace Go?

Cons

- **Complicated** (Adds to production time)
  - Confusing Syntax
- **Inflexible** (Opposite of Go and Python)
  - Static Typing
  - Very low level coding style
- **Different Style** than Go
- Built for safety and stability **NOT simplicity and portability**

What About Python?

Pros

- **Simple Syntax** (Similar to Go)
  - Duck Typing
- **Flexibility**
  - Strong Cross Platform Support
- **Large Libraries and Community Support**
- **Interpreted** (Easy Development)
- Supports **multithreading** (Far from Java)

What About Python?

Cons

- **High Level** Language Style
  - Lacks Low Level Primitives
  - Limits Control
- **Not Fast** (One of Go’s Focuses)
  - Can be helped but still not near Java’s speed
- **Runtime Errors**
- Built for Simplicity **NOT Speed and Stability**

Maybe Haxe?

Pros

- **Neutral** Language
- **Duck Typing** (Similar to Go and Python)
- Moderately **simple** coding style
- **Unique** Features
  - Cross Compiler
- **Multithreading** Support

Maybe Haxe?

Cons

- It is **Average** (Weakness of Go)
  - “Not Solving Any Problems”
  - Average Speed (Not as fast as Java)
  - Average Simplicity (Not Python)
  - Average Flexibility (Not Python)
- **Weak Libraries and Support** (Not Robust)
- Cross Compiler has **Exceptions**

Conclusion

- **Java** – Stable, Safe and Fast
- **Python** – Flexible, Robust and Simple
- **Haxe** – Neutral, Similar to Go, and Average

Choose Python!

- Primary Goal of Go is flexibility!
- Can Be Sped Up to mimic Java! (Secondary Goal)
- Large Libraries!
- Similar to Go in style!

Questions?