DockAlt in Java, OCaml and Coconut

Atibhav Mittal
804598987

Introduction

The aim of this report is to analyze the pros and cons of writing an alternate version of Docker in Java, OCaml or Coconut. Docker is a tool that uses Linux Containers to make it easier to deploy and run applications (1). The benefit of using Docker is that everything that an application needs can be packaged up together, including libraries. This makes it easier for clients to run an application without having to go through the trouble of installing it.

Why Docker is written in Go

The reasons that docker is written in Go are given in the "Docker: an insider view" given on the project homepage (2). Most of the code written in Go is statically compiled, although C Libraries can be dynamically linked using CGo (2).
The benefit of having static compilation is that it is safer. All the binding is done at compile time and we know what version of libraries would be used. (3).
Go has strong duck-typing, supports asynchronous calls and also has a way to make system calls (2).
The drawbacks of using Go for Docker can help us choose a language to write DockAlt in. One of the main drawbacks is that maps used in Go aren't thread safe and it is the responsibility of the developer to ensure that they are thread safe (2). The disadvantage of using static compilation is that if a library is changed, the entire program has to be recompiled. Another drawback of static compilation is that libraries are duplicated every time that they are included in the code, and due to this duplication, the size of the executable is usually larger (3). Another disadvantage of using Go is that there is no IDE for developing applications. This means that most of the development for Go applications is done in text editors like Vim, Emacs or Sublime Text. Another drawback of using Go is that error handling is verbose. (3)

Java

Java is a dynamically compiled language. Java is a verbose and syntactically complex language and hence isn't an easy to use language for people who have not used either C or Java before. Due to the presence of multi-threading in Java, performance could be enhanced. However it is the responsibility of the developer to ensure that the code is thread-safe. Furthermore, since Java code is first compiled to highly portable byte-code, it is easily portable.

OCaml

OCaml is a statically typed functional programming language. Programs written in OCaml are usually much shorter than their counterparts written in conventional languages like C++ and Java (4). The short and simple syntax makes it a language that is easy to pick up for beginners. OCaml does not support parallelism but does support concurrency (5). However, even without multithreading, programs written in OCaml are relatively as fast as programs written in C. Furthermore, due to the brevity of the language, it is faster to develop in OCaml, and that is a huge plus. (4) Since OCaml is a functional language, it doesn't use memory to store variables. It can be thought of as a function from input to output. Thus, it is generally an extremely safe and reliable programming language.

Coconut

Coconut is a functional programming language inspired by Python. Like Python, it is a concise language. A huge advantage of using Coconut is that due to the simple syntax of Python, anyone can learn to code in it relatively quickly. Due to its conciseness, it saves a lot of developer time. Coconut works with most versions of Python 2 and Python 3 (6). Coconut provides a lot of the similar advantages to OCaml but in addition to that, also supports parallel programming. Therefore, Coconut would have generally good performance.

Conclusion

From the above discussion, I believe that Coconut would be the best language to write DockAlt in. Due to the conciseness and garbage collector that a Pythonic language offers, and having most of the functionality that OCaml offers, Coconut would be the best alternative.

Bibliography

1. "What Is Docker?" Opensource.com. N.p., n.d. Web. (https://opensource.com/resources/what-docker)
2. Jérôme Petazzoni, Working Follow. "Docker and Go: Why Did We Decide to Write Docker in Go?" Share and Discover Knowledge on LinkedIn SlideShare. N.p., 07 Nov. 2013. Web. (http://www.slideshare.net/jpetazzo/docker-and-go-why-did-we-decide-to-write-docker-in-go)
3. "Static Build." Wikipedia. Wikimedia Foundation, n.d. Web. 01 Dec. 2016. (https://en.wikipedia.org/wiki/Static_build)
4. "Benefits of OCaml." Benefits of OCaml: Arithmetic. N.p., n.d. Web. 01 Dec. 2016. (http://www.ffconsultancy.com/ocaml/benefits/comparison.html)
5. "Why OCaml's Threading Is Considered as `not Enough`?" Functional Programming - Why OCaml's Threading Is Considered as `not Enough`? - Stack Overflow. N.p., n.d. Web. 01 Dec. 2016. (http://stackoverflow.com/questions/16342840/why-ocamls-threading-is-considered-as-not-enough)
6. "Coconut Documentation¶." Coconut Documentation — Coconut V1.2.0 [Colonel] Documentation. N.p., n.d. Web. 01 Dec. 2016. (http://coconut.readthedocs.io/en/master/DOCS.html#overview)