DockAlt: Containerization Support Languages

Brooke Wenig
University of California, Los Angeles
Computer Science 131, Spring 2015
Professor Eggert

1 Introduction: What is Docker

Docker is an open source project developed by dotCloud in 2013. Its main idea is that with existing methods shipping code to a server is difficult, but it should not be. Thus, Docker set out to solve this problem. Docker for distributing software is analogous to shipping companies for distributing products. In order to separate concerns among the different steps involved in the process of shipping products, the different phases must agree upon some standards so that the same process works regardless of the product one is trying to ship. The shipping container analogy works very well for Docker because similar to how shipping companies standardized a size, dimension, weight and regulations for shipping containers, Docker defines a standard format for software containers that people agree to integrate with it. In turn, this results in a solution that is often cheaper and more reliable.

2 Why Docker

Now one might ask what is the advantage of Docker over a virtual machine (VM). Well, to deploy a new VM, it is essentially shipping a whole new machine. A VM ships more information than needed which causes a lot of overhead for deploying and bad performance because it really does not need all the information that a VM provides. Thus, a VM is too heavyweight. So why choose Docker over an application specific piece? Well, to use an application specific piece, your system must make assumptions about the system in place, and if the application specific piece is meant to run on Python 3.2, but your system uses 2.7, then you do not want to update all of the existing Python libraries and dependencies to 3.2 to just this one piece. What makes Docker so advantageous is that it is more lightweight than a VM, but ships more than just an application specific piece - it is a nice middle ground.

Further, Docker allows for a separation of concerns. By that, I mean that the development team does not have to focus their attention on where it is running or how it is running. Similarly, the ops team does not care what is running inside of it. This makes Docker very portable and an optimal solution for many distributed problems. Its upload speed is very fast because Docker tracks the history (analagous to git push), and the local Docker and registry figure out which diff to upload. This in turn saves bandwidth and disk space locally.

3 Choice of Language Implementation

3.1 What is Go

Docker was implemented in Go, a language developed by Google in 2009. Go is a statically typed language whose syntax is similar to C. However, it varies from C in major ways, notably its garbage collection, type safety, some dynamic-typing capabilities, additional built-in types such as variable-length arrays and key-value maps, and a large standard library. The various reasons for implementing Docker in Go are outlined below.

3.2 Why Go

For starters, Go is a neutral language - it does not have a connotation, such as Java for Android, or C for embedded systems. Secondly, Go has static compilation, which makes it easier to install, test, and adopt code. One does not need to install items in order to run, and is incredibly fast with built-in concurrency primitives. Go essentially contains all of the main items that Docker needs, such as good async primitives (wait for I/O, wait for processes), a low level interface (manage processes, syscalls), extensive std library and data types, and strong duck typing. In addition, it is a full development environment in which Go is able to address multiple issues of the development workflow, from finding documents to fetching dependencies. Lastly, Go has a multi-architectural build without pre-processors.

3.3 Disadvantages of Go

Of course, there are downsides of implementing Docker in Go. For instance, maps in Go are not thread safe (albeit fast), and it is up to the developer to ensure that they are safe. This places a greater role on the developer because if they need their threads to be safe, then they must protect access with sync.Mutex, which is another concern that they would prefer to avoid. In tests, destructors/cleanups do not work. This can be rather cumbersome when trying to run individual tests. In addition, when running go build, even if multiple binaries share the same code, it can be difficult to build. Lastly, error handling can be verbose and there is currently no IDE for Go.

4 Comparison with other languages

4.1 Python

4.1.1 What is Python

Python is a high-level interpreted language which was developed in the 1980s. As of the time of this paper, it is on version 3.4. According to the TIOBE Index, in May 2015, Python is the 6th most popular programming language, with Javascript a close 7th. Python is an objected-oriented language which includes support for dynamic type checking, full garbage collector, multiple inheritance, and unicode, among many other features.

4.1.2 Comparison between Go and Python

Python was the language initially chosen for the Docker prototype. Python has very good container methods built-in, such as dictionaries and has better error handling than Go and is less verbose than Go. Because Python is a dynamically typed language, it makes strong use of duck typing (like Go), which provides for greater flexibility. However, Python was ultimately dropped as the language of choose in favor of Go for the following reasons. Python has problems with redundancy because of it is an interpreted language. Because Python applications load a copy of the Python interpreter each time it runs, if each app runs from its own copy, then there would be many duplicate copies of common files present in the buffer cache. This would have a large performance hit, and performance was the problem that Docker set out to solve. The global interpreter lock is a blocker for Python because it prevents multiple native threads from executing Python byte codes simultaneously. In contrast, Go is incredibly concurrent, and Go programs can be compiled and deployed as a single, stand-alone, binary.

4.2 Java

4.2.1 What is Java

Java is another object-oriented programming languages, and was invented in 1995 by Sun Microsystems (now part of Oracle). As of May 2015, it is the most popular programming language. Its syntax is very similar to that of C and C++, but does not expose as low-level functionalities to the programmer. For example, although the programmer must allocate dynamic memory, Java includes a garbage collector to clean it up when it is no longer needed. Unlike C and C++, Java compiles into byte code which can be run on any Java virtual machine (JVM). Although this bytecode is then interpreted, it has just-in-time compilation in case there is a segment of code that is repeatedly being called, it will convert it to machine code in order to optimize the speed of the program.

4.2.2 Comparison between Go and Java

Java shares a similar idea with Docker in that one program should be able to run on any computer with JVM/Docker installed. Since Java is so portable, with this regard it would make it a very liable substitute for Go in Docker. Similar to Python, Java has better error handling than Go. Go does not have exceptions but instead allows functions to return more than one value, which can then be used for error handling. However, whereas Go can do most things using the standard libraries, Java needs lots of add-ons for some features. Such an example is handling concurrency: Go has built-in capabilities, but these tools in Java are inside of libraries. Java will limit productivity in the cloud, whereas Go excels in this domain. Unlike Go, Java is not open-source, thus is cannot have the same community contributing to the language. Although Java is the most popular language at the time of this writing, Go still has time to mature and grow.

4.3 Groovy

4.3.1 What is Groovy

Groovy is a relatively young language, like Go. Groovy 1.0 was released in 2007, and is currently on version 2.4.3. Groovy is a dynamically typed, object-oriented language. It compiles to JVM byte code, and thus integrates very well with any Java program. In addition, Groovy has static-typing and static-compilation capabilities. Most Java syntax is valid in Groovy, though semantics may vary. Groovy includes features not available in Java, such as support for static and dynamical type checking, as well as native syntax for regular expressions.

4.3.2 Comparison between Go and Groovy

Groovy and Go both support dynamic and static type checking capabilities, and both provide support for concurrency. However, concurrency is easier to implement in Go than in Groovy. Further, because Groovy is an interpreted language, it is not as fast as Go, a compiled language. The garbage collection in Go is much better than the JVM, which Groovy runs on top of. In the JVM there is a lot more system overhead, such as memory issues or resource management. By using Go instead of the JVM, the developer is able to focus more on writing code, testing, and deploying.There is less boiler plate code for Go in comparison to Groovy, which is another bonus for the developer.

5 Overall Comparison

Despite the fact that Docker was released just over 2 years ago, as of the time of this writing it has over 950 contributors on Github. I personally was not a fan of the Go syntax as it seemed to combine syntax conventions from many different languages, but perhaps over time I would grow accustomed to it. The downside of using Go is that most people arent as well versed in the language (perhaps because it is relatively new), so there is a natural learning curve. However, using Go from a functionality standpoint addresses all of the major concerns that Docker hoped to accomplish. It is very fast, flexible, reliable, and open-sourced. Therefore in comparison to the other three languages noted above, Go seems to be the most appropriate decision.

6 References

Brockmeier, Joe. ”Google’s Go Programming Language Grows Up: Now What?” ReadWrite. N.p., 29 Mar. 2012. Web. 04 June 2015.
”Docker and Go: Why Did We Decide to Write Docker in Go?” N.p., Nov. 2013. Web. 03 June 2015.
Hykes, Solomon. ”Introduction to Docker.” YouTube. N.p., 02 Oct. 2013. Web. 03 June 2015.
”A Multi-faceted Language for the Java Platform.” The Groovy Programming Language. N.p., n.d. Web. 03 June 2015.
”Paas under The hood.” DotCloud. N.p., n.d. Web. 03 June 2015.
”TIOBE Software: The Coding Standards Company.” TIOBE. N.p., n.d. Web. 27 May 2015.