# Best Programming Language for Containerization

*Calvin Liu*
*CS 131 Programming Languages*

## 1. Abstract

The purpose of this project is to see if other languages are fit for creating a proxy herd on a large set of virtual machines using Docker. Besides looking at the Twisted framework, we consider our option of using Java, Python, and Julia instead of Go to implement Docker in order to keep costs low and to improve reliability with more than one programming language.

## 2. Introduction

Implementation of a proxy herd on a large set of virtual machines is costly using Twisted so an option to keep costs low is to implement it using Docker. Unfortunately, Go may not be reliable and so we look at other programming languages to implement this Docker in Java, Python, and Julia. We check out the pros and cons of implementing Docker in these languages.

## 3. Java

Java has asynchronous method invocation which is a plus side when creating a server herd because we want to optimize the efficiency of the herd. Java also comes with a garbage collector and thus, memory management is not an issue as it removes memory leaks and will stop the servers from crashing from memory management. Java also comes with a unique library and APIs that help with things such as multithreading and networking functionality. These APIs allow Java to also be faster than python, which is important for a server herd. As we can see, if Twisted is a good framework for an application server herd then Java is a good programming language to use as it contains the functionalities that Python has.

There is one potential problem with using Java to implement Docker unfortunately which is that Java doesn't have dynamic type casting which might be a problem when packaging and reassembling the containers once opened. If each container has different types in it then that can potentially throw an error. With Java, errors must also be caught at each level of a function call unlike a language like Python where the exceptions will propagate upwards automatically. Also each java file must become its own class rather than having everything in one file which leads to the code to be verbose aside from the actual syntax. Java also needs its own JVM in order for it to run correctly and this can use up a lot of memory.

## 4. Python

To manage deployment, python is a suitable language because it has object oriented aspects and scripting aspects. Python also does not explicitly define types which allows the use of duck typing. This can help when trying to package a bunch of different applications together in the same container. Python supports outside APIs as well and a user can override the necessary library functions to fit the deployment aspects. Errors also propagate upward automatically in function calls which allows easier exception throwing between nested function calls because each call will get the same exception error.

Python itself has different container methods such as '__contains__' and other forms such as dictionaries, tuples, etc. This support will be similar to build something that contains different objects that will be deployed together when called. These containers can be modified to containerize different web aspects and when deploying, you can retrieve these different types of applications and put them together. For instance, a tuple can contain a string and an integer and a bool. This itself is a container and when we retrieve the values, it is like deploying them.

## 5. Julia

Julia is a programming language that is very similar to Python. It does not explicitly define types and has containers such as lists and tuples. Julia also supports classes which come with aspects such as inheritance. The purpose of Julia is mostly used in scientific computing, but since this language is fairly new and mostly used for computing, deployment and containerization is difficult to support.

One of the main flaws is that Julia does not support a virtualenv which means there are extra things Julia needs. For instance, a lot of Julia programs need to be built with support of C libraries, but unfortunately Julia's package management system does not handle this at all. There is also no sense of creating an application-specific file, which limits the constraints of packaging certain applications that may not depend on other applications and just simply needs to be deployed. Creating a container and deploying everything to work together is not efficient since Julia already needs its own virtualenv and would consume time and resources to have Julia set everything up and deploy everything without error. The developers receiving the box also would have to learn about how to use Julia and this could take a long time since Julia is fairly new.

## 6. Conclusion

Based on my research I have found that Java and Julia are not the best form of languages to package applications and then deploy them in their own environment. Python on the other hand is a good scripting language to use in order to create a virtualenv and deploy each application in that environment. Its dynamic typing will also subvert a large set of problems that could occur if there was only static typing. Python also has good memory management and garbage collection. Python has similar functionality to java and similar functionality as Julia, but since it is a wide spread language, it is an easy language to use to implement Docker.

## 7. References

"A Summary of Features." The Julia Language. N.p., n.d. Web. 09 Mar. 2015.

"Java Platform SE 7." Java Platform SE 7. N.p., n.d. Web. 06 Mar. 2015.

Stucchio, Chris. "Deploying Julia Servers with Docker." Chris Stucchio ATOM. Chris Stucchio, 11 Nov. 2014. Web. 09 Mar. 2015.

"Welcome to Python.org." Python.org. N.p., n.d. Web. 09 Mar. 2015.

"What Exactly Are "containers" in Python? (And What Are All the Python Container Types?)." - Stack Overflow. N.p., n.d. Web. 09 Mar. 2015.

"What Is Docker?" What Is Docker? An Open Platform for Distributed Apps. N.p., n.d. Web. 09 Mar. 2015.