Nathan Gillette

The basis for this project is to microbenchmark two concurrent data structures to measure their throughput. The two data structures I chose were the built in java.concurrent.ConcurrentHashMap and a custom ReadWriteLock based HashTable. The Custom HashTable is read and written to in a 80/20 ratio to use the best performance practice for ReadWriteLock based data structures. The mock use case is a Flight Manifest, where each flight has an ID as well as a passenger number. The data structure can be initialized with a certain number of premade flights, or can be initialized with no flights. For testing purposes, I started the data structures with 20 premade flights. I hard coded these flights because I didn't want to skew my test with io. There were some extra methods needed in the MyBenchmark Class for the jmh benchmark, specifically the setup and teardown methods. These were crucial to the jmh benchmark.

I originally was considering other mock use cases such as a grocery store, a music and book library, and some kind of card game. I eventually settled on the Flight Manifest because I found some ideas online and it seemed like it would be straightforward enough while still working for the project requirements. It should come as no surprise that jmh was the hardest part of this project. Luckily, I was able to find a nice guide that uses maven to set up a nice template to work in. It's become abundantly clear to me that learning a language is really only half the battle, the tools and other infrastructure that is built around a language like Java is just as important.

Links:

Screencap

Graph