

The goal here is there is no deadlock ie everyone is waiting for other person to relinquish a fork while holding the other.

Also no single friend should be waiting forever to get both forks (this would starvation)

To solve this problem it would be best to use the *java.util.concurrent.locks.ReentrantLock*;

This class allows fairness flag. And then use the *trylock* with timeout out so that this fairness flag is honored.

The Domain model is below.

Diner represents the Friend who is dining at that table.

DinnerFork is the class that represents the forks at the table.

The State class is an Enumeration with LEFT,RIGHT as the values since each fork could assume either left fork or right fork state depending on who picks it up.

The constants define some constants. The definition for diner being full I have assumed 20 time taking a bite of the food.

The pros: There will not be any deadlock and starvation.

Cons: there will be cycles where there will failed attempt to eat.

