AP(IT) locks and conditions

Simon Rogers, February 2017

This lab superceeds the one about jobs on stacks. I think this exercise makes the important points a bit clearer.

Background

- In an odd restaurant, there is only one customer. There is also one chef, and one waiter. The customer will eat a meal of four courses: starter, main, pudding, coffee.
- The chef produces courses that will then be taken by the waiter to the customer.
- The chef should not give a dish to the waiter until the waiter has given the previous one to the customer.
- The customer shouldn't take a dish from the waiter until they have eaten the previous one.

Tasks

- Design and build a set of Java classes that perform these operations.
- You will likely need one class for the chef, one for the customer and one (including locks and conditions) for the waiter (plus a main method somewhere).
- The chef should lock the waiter's lock and then wait for the consumer to take a dish (and signal) before adding the next dish.
- The customer should lock the waiter's lock and then wait for a new dish to appear (i.e. a `signal` from the chef) before taking it.
- At any point, either the chef or the customer should be in `await` awaiting a `signal`.
- The logic will all be wrapped up in two methods within the waiter: `giveDish()` and `takeDish()`