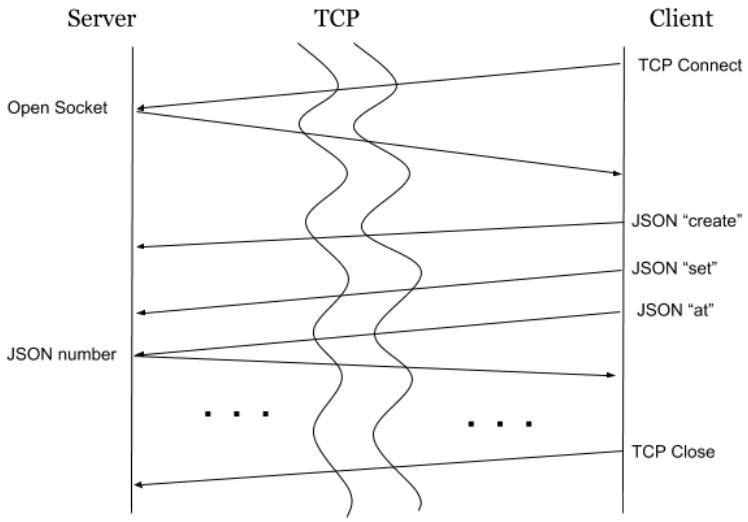

Spreadsheet Microservice

Throughout this writing, `*N*` will refer to a natural number, and *JF* is one of:

-- *N*
-- [">", *N*, *N*]
-- [*JF* , "+", *JF*]
-- [*JF* , "*", *JF*]

The user experience will consist of:

- Connect to the server through TCP **on port 8000** and receive a stream
    - all further messages will be JSON over this stream
- Start by sending a JSON string to create a spreadsheet, of the form:
    - *Sheet*: ["sheet", name:string, [[JF, ...] ...]]
    - ***Sheet* will receive no output from the Server**
- Once a spreadsheet has been created, send `set` or `at` messages to assign a formula, or retrieve a calculated cell value respectively. Data should be of the form:
    - *Set*: ["set" , name:string, x:*N*, y:*N, JF*]
    - *At*: ["at", name:string, x:*N*, y:*N*]
    - ***Set* will get no output back from the Server**
    - ***At* will receive a floating-point number back, representing the calculated value of the Cell chosen**
    - The coordinate system for the spreadsheet abides by standard screen coordinates (“right” is positive on the X-axis, “down” is positive on the Y-axis)
- After sufficiently setting and getting values in the spreadsheet, a user ends their spreadsheet usage by simply closing the TCP socket opened in the first exchange

Throughout this whole exchange, a malformed (but valid) JSON message will receive no output from the server, and have no effect on the underlying sheet, and an invalid JSON message will crash the server.

For further examples of the systems underlying the Server implementation, and what each message should execute, see previous memos alongside the server spec.