receives `SET foo bar` as a command from a client, it'll send `*3\r\n$3\r\nSET\r\n$3\r\nfoo\r\n$3\r\nbar\r\n` to all connected replicas over their respective replication connections.

Replicas process commands received over the replication connection just like they would process commands received from a client, but with one difference: Replicas don't send responses back to the master. They just process the command silently and update their state.

Similarly, the master doesn't wait for a response from the replica when propagating commands. It just keeps sending commands as they come in.

There is one exception to this "no response" rule, the `REPLCONF GETACK` command. We'll learn about this in later stages.

Tests

The tester will execute your program like this:

```
./spawn_redis_server.sh --port <PORT>
```

It'll then connect to your TCP server as a replica and execute the following commands:

1. `PING` (expecting `+PONG\r\n` back)
2. `REPLCONF listening-port <PORT>` (expecting `+OK\r\n` back)
3. `REPLCONF capa eof capa psync2` (expecting `+OK\r\n` back)
4. `PSYNC ? -1` (expecting `+FULLRESYNC <REPL_ID> 0\r\n` back)

The tester will then wait for your server to send an RDB file.

Once the RDB file is received, the tester will send series of write commands your program (as a separate Redis client, not the replica).

Ready to run tests... Show logs