Here's an example how to write the test.

Suppose we want to communicate with another system over a socket.
When we're done, the socket should be closed and we should have read the string abc.

1) Begin with
```
testCompleteTransaction() {
...
assertTrue(reader.isClosed());
assertEquals("abc", reply.contents());
}
```

2) Where does the reply come from? The socket, of course:
```
testCompleteTransaction() {
...
Buffer reply= reader.contents();
assertTrue(reader.isClosed());
assertEquals("abc", reply.contents());
}
```

3) And the socket? We create it by connecting to a server:
```
testCompleteTransaction() {
...
Socket reader= Socket("localhost", defaultPort());
Buffer reply= reader.contents();
assertTrue(reader.isClosed());
assertEquals("abc", reply.contents());
}
```

4) But before this, we need to open a server:
```
testCompleteTransaction() {
Server writer= Server(defaultPort(), "abc");
Socket reader= Socket("localhost", defaultPort());
Buffer reply= reader.contents();
assertTrue(reader.isClosed());
assertEquals("abc", reply.contents());
}
```

Now we may have to adjust the names based on actual usage, but we have created the
outlines of the test in teeny-tiny steps, informing each decision with feedback within seconds.