Google

Test Behaviors, Not Methods

After writing a method, it's easy to write just one test that verifies everything the method does. But **it can be harmful to think that tests and public methods should have a 1:1 relationship.** What we really want to test are *behaviors*, where a single method can exhibit many behaviors, and a single behavior sometimes spans across multiple methods.

Let's take a look at a bad test that verifies an entire method:

```
@Test public void testProcessTransaction() {
  User user = newUserWithBalance(LOW_BALANCE_THRESHOLD.plus(dollars(2));
  transactionProcessor.processTransaction(
      user,
      new Transaction("Pile of Beanie Babies", dollars(3)));
  assertContains("You bought a Pile of Beanie Babies", ui.getText());
  assertEquals(1, user.getEmails().size());
  assertEquals("Your balance is low", user.getEmails().get(0).getSubject());
}
```

Displaying the name of the purchased item and sending an email about the balance being low are two separate behaviors, but this test looks at both of those behaviors together just because they happen to be triggered by the same method. **Tests like this very often become massive and difficult to maintain over time as additional behaviors keep getting added in**—eventually it will be very hard to tell which parts of the input are responsible for which assertions. The fact that the test's name is a direct mirror of the method's name is a bad sign.

It's a much better idea to use separate tests to verify separate behaviors:

```
@Test public void testProcessTransaction_displaysNotification() {
  transactionProcessor.processTransaction(
      new User(), new Transaction("Pile of Beanie Babies"));
  assertContains("You bought a Pile of Beanie Babies", ui.getText());
}
@Test public void testProcessTransaction_sendsEmailWhenBalanceIsLow() {
  User user = newUserWithBalance(LOW_BALANCE_THRESHOLD.plus(dollars(2));
  transactionProcessor.processTransaction(
      user,
      new Transaction(dollars(3)));
  assertEquals(1, user.getEmails().size());
  assertEquals("Your balance is low", user.getEmails().get(0).getSubject());
}
```

Now, when someone adds a new behavior, they will write a new test for that behavior. Each test will remain focused and easy to understand, no matter how many behaviors are added. **This will make your tests more resilient since adding new behaviors is unlikely to break the existing tests, and clearer since each test contains code to exercise only one behavior.**

More information, discussion, and archives:

http://googletesting.blogspot.com