Debugging sucks.
Testing rocks.

Testing on the Toilet

Cleanly Create Test Data

Helper methods make it easier to create test data. But they can become difficult to read over time as you need more variations of the test data to satisfy constantly evolving requirements from new tests:

```
// This helper method starts with just a single parameter:
Company company = newCompany(PUBLIC);

// But soon it acquires more and more parameters.
// Conditionals creep into the newCompany() method body to handle the nulls,
// and the method calls become hard to read due to the long parameter lists:
Company small = newCompany(2, 2, null, PUBLIC);
Company privatelyOwned = newCompany(null, null, null, PRIVATE);
Company bankrupt = newCompany(null, null, PAST_DATE, PUBLIC);

// Or a new method is added each time a test needs a different combination of fields:
Company small = newCompanyWithEmployeesAndBoardMembers(2, 2, PUBLIC);
Company privatelyOwned = newCompanyWithType(PRIVATE);
Company bankrupt = newCompanyWithBankruptcyDate(PAST_DATE, PUBLIC);
```

Instead, **use the *test data builder* pattern: create a helper method that returns a partially-built object** (e.g., a Builder in languages such as Java, or a mutable object) whose state can be overridden in tests. The helper method initializes logically-required fields to reasonable defaults, so **each test can specify only fields relevant to the case being tested**:

```
Company small = newCompany().setEmployees(2).setBoardMembers(2).build();
Company privatelyOwned = newCompany().setType(PRIVATE).build();
Company bankrupt = newCompany().setBankruptcyDate(PAST_DATE).build();
Company arbitraryCompany = newCompany().build();

// Zero parameters makes this method reusable for different variations of Company.
// It also doesn't need conditionals to ignore parameters that aren't set (e.g. null
// values) since a test can simply not set a field if it doesn't care about it.
private static Company.Builder newCompany() {
  return Company.newBuilder().setType(PUBLIC).setEmployees(100); // Set required fields
}
```

Also note that **tests should never rely on default values that are specified by a helper method** since that forces readers to read the helper method's implementation details in order to understand the test.

```
// This test needs a public company, so explicitly set it.
// It also needs a company with no board members, so explicitly clear it.
Company publicNoBoardMembers = newCompany().setType(PUBLIC).clearBoardMembers().build();
```

You can learn more about this topic at http://www.natpryce.com/articles/000714.html

More information, discussion, and archives:

testing.googleblog.com