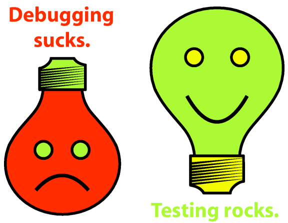

Testing on the Toilet

Avoiding Flaky Tests

Apr. 17, 2008

Flaky tests make your life more difficult. You get failure notifications that aren't helpful. You might become numb to failures and miss an actual failure condition. Your changes might get unfairly blamed for causing a flaky test to fail.

Unfortunately, **a myriad of factors can make a test flaky**. Today, we tackle a simple example: file access from a unit test. Take this function and its test:

```
def CreateGapLease(self):
  data_file = open('/leases/gap', 'w+')
  data_file.write(contract_data)
  data_file.close()

def testCreateGapLease(self):
  contract_writer.CreateGapLease()
  self.assertEqual(ReadFileContents('/leases/gap'), contract_data)
```

What if `/leases/gap` already exists and contains some data? `testCreateGapLease` will fail. This is a general problem where preconditions are assumed to be correct. This could just as easily happen by assuming a database contains the proper information (or no information). **What if another test that uses that file was running concurrently?**

If you really want to test your code using live resources, always check your assumptions. In this case, clearing the file at the start of the test can reduce its brittleness:

```
def testCreateGapLease(self):
  if os.path.exists(lease_file):
    RemoveFile(lease_file)
  ...
```

Unfortunately, this doesn't completely eliminate the flakiness of our test. If `/leases/gap` is an NFS path or can be written to by a different test, our test can still fail unexpectedly. It's better for the test to use a unique resource. This can be accomplished with a small refactoring of `CreateGapLease`:

```
def CreateGapLease(self, lease_path=None):
  if lease_path is None:
    lease_path = '/leases/gap'
  ...
```

The callers of `CreateGapLease` can continue invoking it as usual, but the unit test can pass in a different path:

```
def testCreateGapLease(self):
  lease_file = os.path.join(FLAGS.test_tmpdir, 'gap')
  contract_writer.CreateGapLease(lease_path=lease_file)
  self.assertEqual(ReadFileContents(lease_file), contract_data)
```

Of course, to make your test as fast as possible, it would be better to forgo disk access altogether by using a mock file system.

More information, discussion, and archives:
http://googletesting.blogspot.com