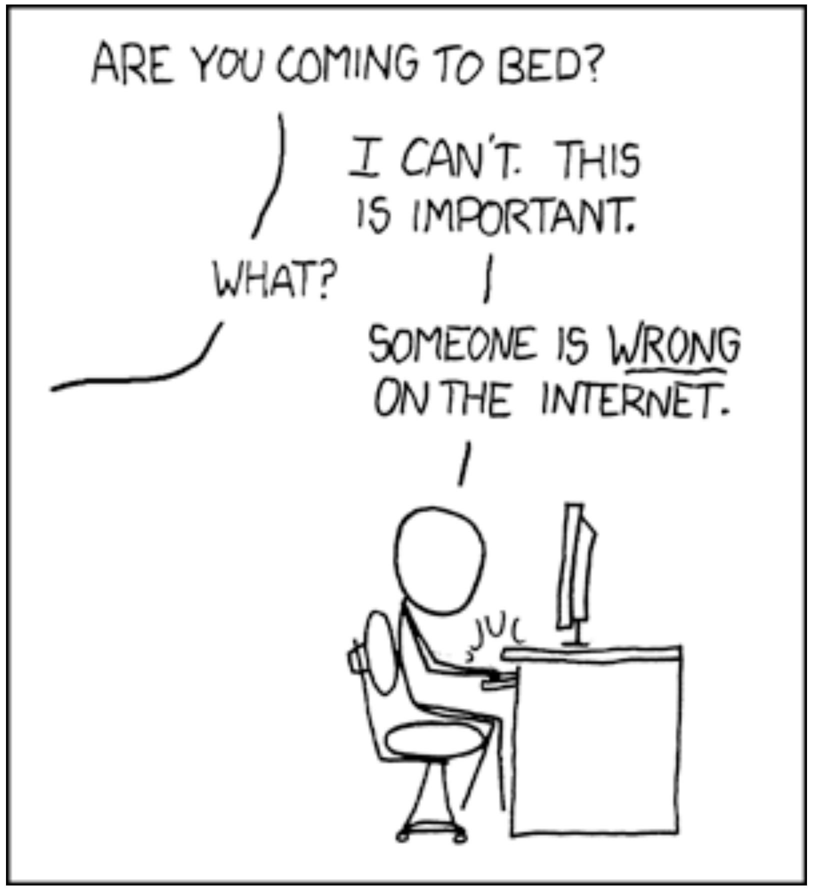

ARE YOU COMING TO BED?
I CAN'T. THIS IS IMPORTANT.
WHAT?
SOMEONE IS WRONG ON THE INTERNET.

http://xkcd.com/386/

Wrong

The right way to assert

Time Is Money (Test::Unit)

```
assert_equal money, time
```

- verb is moved to front
- direct and indirect objects (expected and actual) are reversed

Time Is Money (RSpec)

```
time.should ==(money)
```

- confusing syntax -- space vs. dot vs. underscore vs. parens
- == looks familiar, but "should" is inserted between parameters so it's not apparent what the calculation is

Time Is Money (Minitest)

```
assert time == money
```

- Ah, now I see what you mean
- Default failure message is not helpful

  ```
  “Failed assertion, no message given.”
  ```

- Making the message helpful violates DRY

  ```
  assert time == money, “Time should equal money”
  ```

Time Is Money (Wrong)

```
assert { time == money }
```

- Failure message is helpful:

```
Expected (time == money), but 6 is not equal to 27
    time is 6
    money is 27
```

How do we do it?

Magic

- RubyParser and Ruby2Ruby by Ryan Davis
- We turn the block into source code, parse it into an AST, break it down into parts, then convert back to code for the messages

Also, we cheat

- We open the source file on disk, jump to the right line, and parse it
- If you're constructing your tests with metaprogramming, you've got bigger problems than not being able to use Wrong

Less Is More

```
Test::Unit Asserts
assert_block { x }
assert(x)
assert_equal x, y
assert_raise LoadError { x }
assert_raise { x }
assert_
assert_
assert_
assert_
assert_
assert_
assert_
assert_
assert_
assert_
assert_
assert_
flunk
assert_
assert_
assert_
assert_
assert_
assert_throws
assert_nothing_thrown
assert_in_delta f, g, delta
assert_send [o, m, arg1, arg2]
assert_boolean x
assert_true x
assert_false x
assert_compare x, ">=", y
assert_fail_assertion { x }
assert_raise_message m, { x }
assert_const_defined Test, :Unit
assert_not_const_defined Test, :Unit
assert_predicate o, :empty?
assert_not_predicate
assert_alias_method
assert_path_exist "/foo"
assert_path_not_exist "/foo"
```

```
RSpec Matchers
x.should be_true
x.should be_false
x.should be_nil
x.should == y
x.should have_at_most(number).items
x.should include(y)
x.should match(/regex/)
lambda { do_something_risky }.should raise_exception
lambda { do_something_risky }.should raise_exception
(PoorRiskDecisionError)
lambda { do_something_risky }.should raise_exception
(PoorRiskDecisionError) { |exception|
exception.data.should == 42 }
lambda { do_something_risky }.should raise_exception
(PoorRiskDecisionError, ""that was too risky"")
lambda { do_something_risky }.should raise_exception
(PoorRiskDecisionError, /oo ri/)
x.should respond_to(*names)
```

```
assert { ... }
deny { ... }
```

Helpers

- rescuing { }
- capturing { }
- close_to?
  - assert { x.close_to?(y) }
  - assert { x.close_to?(y, delta) }

Frameworks

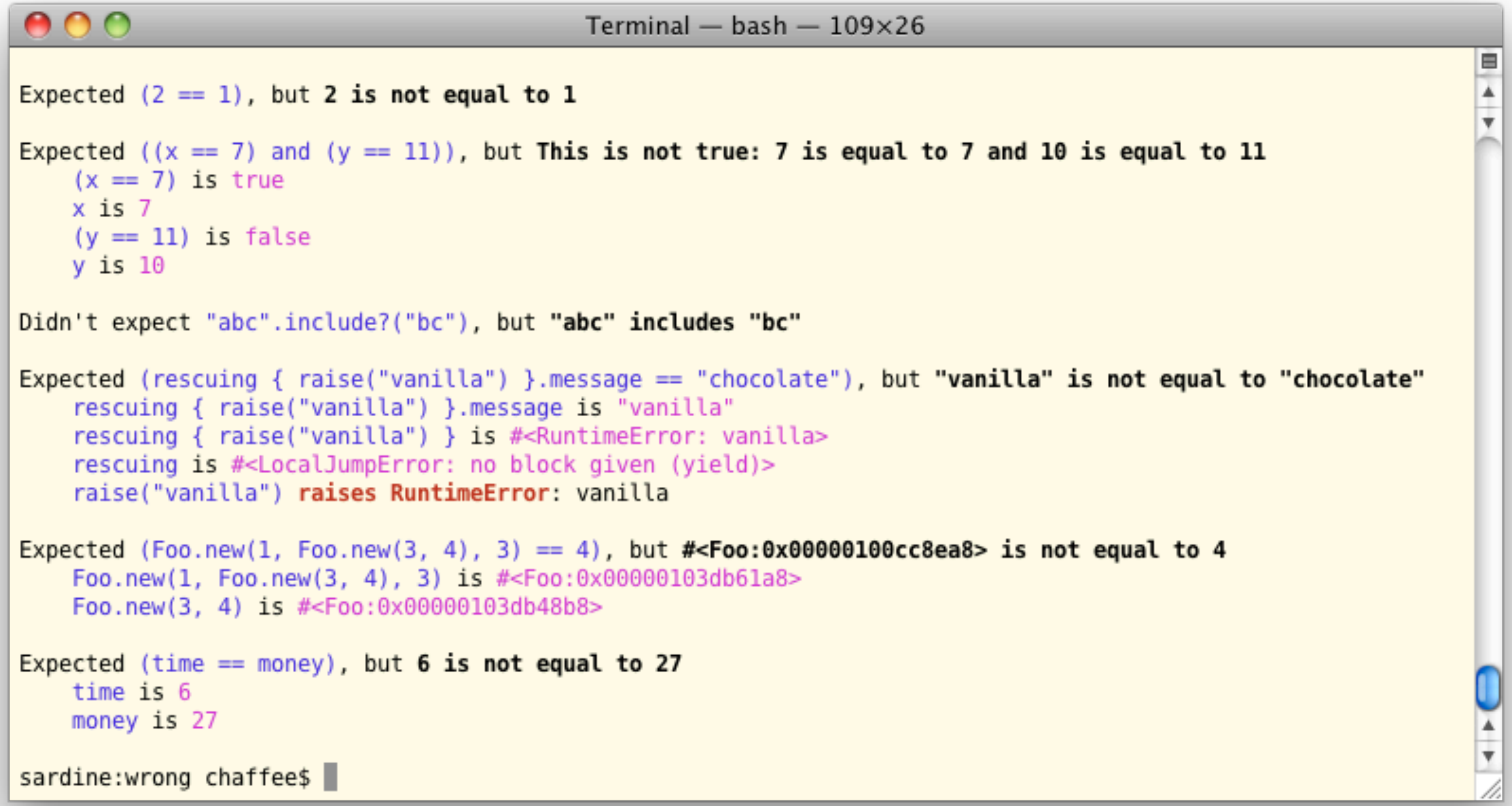
- Minitest
- RSpec
- Test::Unit
- ???

Explanations

```
assert("since we're on Earth")
         { sky.blue? }
```

Color

- Because you can’t succeed without it



```
Expected (2 == 1), but 2 is not equal to 1

Expected ((x == 7) and (y == 11)), but This is not true: 7 is equal to 7 and 10 is equal to 11
    (x == 7) is true
    x is 7
    (y == 11) is false
    y is 10

Didn't expect "abc".include?("bc"), but "abc" includes "bc"

Expected (rescuing { raise("vanilla") }.message == "chocolate"), but "vanilla" is not equal to "chocolate"
    rescuing { raise("vanilla") }.message is "vanilla"
    rescuing { raise("vanilla") } is #<RuntimeError: vanilla>
    rescuing is #<LocalJumpError: no block given (yield)>
    raise("vanilla") raises RuntimeError: vanilla

Expected (Foo.new(1, Foo.new(3, 4), 3) == 4), but #<Foo:0x00000100cc8ea8> is not equal to 4
    Foo.new(1, Foo.new(3, 4), 3) is #<Foo:0x00000103db61a8>
    Foo.new(3, 4) is #<Foo:0x00000103db48b8>

Expected (time == money), but 6 is not equal to 27
    time is 6
    money is 27

sardine:wrong chaffee$
```

Info

- Authors
  - Steve Conover - <sconover@gmail.com>
  - Alex Chaffee - <alex@stinky.com> - <http://alexch.github.com>
- Github: <http://github.com/alexch/wrong>
- Tracker: <http://www.pivotaltracker.com/projects/109993>