I love MiniTest::Unit

Keep it simple
Keep it Ruby

Minimal dependency

MiniTest is part of the standard library

require 'minitest/autorun'

Basic rules

- Test classes should:
  - Have a name ending 'Test'. For example 'MyTest'
  - Inherit from MiniTest::Unit::TestCase
- Each test method:
  - **Must** have a name starting with 'test_'
  - Contain (at least) one assertion

Special methods

- setup : run before each test method
- teardown : run after each test method

Assertions

- Equality:
  - assert_equal 'foo', bar
- Exceptions:
  - assert_raise SomeError { some_method }
- Many others:
  - http://ruby-doc.org/stdlib-1.9.3/libdoc/minitest/unit/rdoc/index.html
- More in Rails
  - Search http://api.rubyonrails.org/ for 'assert_'

Performance

- Minitest is a small library
- Minimal dependency
- Tests can be kept simple to run very quickly – so you run them often.

Write in ruby

- You do not have to use a DSL (domain specific language).
- All the techniques you use in your code to keep your code DRY, succinct, easily maintainable, and readable, can be used in your tests.
- If you love Ruby, why wouldn't you want to write your tests in Ruby?

But if you need to ...

- Use a spec type DSL
  - http://ruby-doc.org/stdlib-1.9.3/libdoc/minitest/spec/rdoc/index.html
- Use 'should “do something” do' syntax
  - https://github.com/thoughtbot/shoulda
- Use with cucumber
  - Cucumber is not dependent on a particular test framework, and can be used with MiniTest::Unit as easily as any other framework.
  - http://cukes.info/

An example

```
class Ball
  attr_accessor :colour
end

require "minitest/autorun"
class BallTest < MiniTest::Unit::TestCase

  def setup
    @ball = Ball.new
  end

  def test_colour
    assert_nil @ball.colour
  end

  def test_set_and_get_colour
    colour = 'blue'
    @ball.colour = colour
    assert_equal colour, @ball.colour
  end

end
```

Two files

```ruby
# lib/ball.rb

class Ball

  attr_accessor :colour

end
```

```ruby
# test/ball_test.rb
require "minitest/autorun"
require_relative '../lib/ball'

class BallTest < MiniTest::Unit::TestCase

  def setup
    @ball = Ball.new
  end

  def test_colour
    assert_nil @ball.colour
  end

  def test_set_and_get_colour
    colour = 'blue'
    @ball.colour = colour
    assert_equal colour, @ball.colour
  end

end
```

And don't forget

Make sure your tests fail before you write the code that makes them pass