Automatische Tests mit Minitest

Automatische Tests

Unit-Tests

Spezifizieren korrektes Verhalten von Programm, Funktionalität, Klasse (Unit = Einheit)

Test-Driven Development (TDD)

Tests als erste Nutzer bzw. Konsumenten von Code

- Perspektive des Nutzers

Regressionstests

Wiederholung von schon behobenen Fehlern vermeiden

Verfügbare Bibliotheken in Ruby:

- **`minitest`** – Unit-Tests, Benchmarks
- **`rspec`** – Unit-Tests, “Specs”, deklarative Syntax
- **`cucumber`** – Behavior-Driven Development

Minitest

- Schnelle, minimale Implementierung eines Test-Frameworks
- Wenig Magie, Tests bauen mit “plain Ruby”
- Integration in prominente Projekte, z.B. Ruby on Rails

- Testklassen erben von `Minitest::Test`
- Jede Methoden `test_*` enthält einen Testfall

- Zusätzliche optionale Komponenten
  - “Specs”: Deklarative Syntax für Tests
  - Benchmarks: Performance-Messung von Algorithmen
  - Mocks: Generierung von Objekten als Ersatz für Kollaborateure in Unit-Tests

Minitest: Installation

Minitest installieren

- Gem `minitest` installieren

```
> gem install minitest
Successfully installed minitest-5.11.3
1 gem installed
```

Tests ausführen

tests/greeter_test.rb

```ruby
require "greeter"
require "minitest/autorun"

class GreeterTest < Minitest::Test
  # ...
end
```

```
> ruby -I lib tests/greeter_test.rb
Run options: --seed 45351

# Running:

.

Finished in 0.000675s, 1482.4764 runs/s, 1482.4764 assertions/s.
1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
```

Tests schreiben

```ruby
require "greeter"
require "minitest/autorun"

class GreeterTest < Minitest::Test
  def setup
    @greeter = Greeter.new
  end

  def test_that_greeter_yells
    assert_equal "HI THERE!", @greeter.greet
  end
end
```

Assertions

```ruby
assert true                          # => true
assert Object.new                    # => true
assert false                         # Minitest::Assertion
assert nil                           # Minitest::Assertion

assert_equal "Jan", my.name          # => true
assert_equal "Horst", my.name        # Minitest::Assertion

assert_same "openhpi", "openhpi"     # Minitest::Assertion
assert_same :openhpi, :openhpi       # => true

assert_includes 1..3, 2              # => true
assert_includes 1..3, 4              # Minitest::Assertion
```

Assertions

Eigene Fehlertexte mit zusätzlichem Argument:

```
assert nil
# Minitest::Assertion: Expected nil to be truthy.

assert nil, "My condition failed"
# Minitest::Assertion: My condition failed.

assert_includes 1..3, 4
# Minitest::Assertion: Expected 1..3 to include 4.

assert_includes 1..3, 4, "Element not in list"
# Minitest::Assertion: Element not in list.
```

Tests mit Minitest