

---

---

# Ruby

— Rspec Matchers —

---

---

# RSpec Matchers

- RSpec utiliza los métodos **to** y **not\_to** en todas las especificaciones externas.
- **to/not\_to** reciben un parámetro: Un Matcher
- Ejemplos de matchers:
  - be\_true/be\_false
  - eq 3
  - raise\_error

# be\_predicate

- Si el objeto sobre el cual estamos operando el test tiene un método que retorne un valor boolean, automáticamente se puede utilizar el matcher **be\_predicate(boolean)**.
- Entonces, por ejemplo, be\_nil es un matcher válido, ya que todo objeto en ruby tiene el método **:nil?**.

```
it "should sum two odd numbers and become even" do
  expect(@calculator.add(3, 3)).to be_even
  expect(@calculator.add(3, 3)).not_to be_odd
end
```

# format documentation

```
→ leccion-16 git:(master) ✗ rspec --format documentation
```

Calculator

```
  should add 2 numbers correctly (FAILED - 1)  
  should subtract 2 numbers correctly (FAILED - 2)  
  should sum two odd numbers and become even
```

Failures:

```
1) Calculator should add 2 numbers correctly  
   Failure/Error: expect(@calculator.add(2, 2)).to eq 4
```

# Ejemplo de matcher custom

```
def algo?  
  return true  
end  
def has_algo?  
  return false  
end
```



```
it "debería ser algo" do  
  expect(@calculator).to be_algo  
end  
it "debería tener algo" do  
  expect(@calculator).not_to have_algo  
end
```

# Ejecución de ejemplo

```
→ leccion-16 git:(master) x rspec --format documentation
```

Calculator

```
  should add 2 numbers correctly (FAILED - 1)
  should subtract 2 numbers correctly (FAILED - 2)
  should sum two odd numbers and become even
  debería ser algo
  debería tener algo
```

Failures:

```
1) Calculator should add 2 numbers correctly
   Failure/Error: expect(@calculator.add(2, 2)).to eq 4

     expected: 4
     got: 0

     (compared using ==)
# ./spec/calculator_spec.rb:8:in `block (2 levels) in <top (required)>'

2) Calculator should subtract 2 numbers correctly
   Failure/Error: expect(@calculator.subtract(4, 2)).to eq 2

     expected: 2
     got: 6

     (compared using ==)
# ./spec/calculator_spec.rb:12:in `block (2 levels) in <top (required)>'
```

# Mas Matchers (<https://relishapp.com/rspec/rspec-expectations/docs>)



Public projects | Plans & pricing

Sign up | Sign in

Publisher: RSpec

## Project: RSpec Expectations 3.5

Change version



Browse documentation



Built in matchers

Custom matchers

Aggregating Failures

Composing Matchers

Compound Expectations

Define negated matcher

Customized message

Diffing

Implicit docstrings

Syntax Configuration

Test frameworks

Changelog

## RSpec Expectations 3.5



rspec-expectations is used to define expected outcomes.

```
RSpec.describe Account do
  it "has a balance of zero when first created" do
    expect(Account.new.balance).to eq(Money.new(0))
  end
end
```

### Basic structure

The basic structure of an rspec expectation is:

```
expect(actual).to matcher(expected)
expect(actual).not_to matcher(expected)
```

# Entonces...

RSpec tiene un montón de matchers pre-construidos listos para simplificar la escritura de tests!