



---

## mongoid-rspec

gem version 4.2.0  

The mongoid-rspec library provides a collection of RSpec-compatible matchers that help to test Mongoid documents.

Tested against: - MRI: 2.6.x, 2.7.x, 3.0.x, 3.1.x, 3.2.x - Mongoid: 4, 5, 6, 7, 8

See [.github/workflows/rspec.yml](#) for the latest test matrix.

### Installation

Drop this line into your Gemfile:

```
1 group :test do
2   gem 'mongoid-rspec'
3 end
```

### Compatibility

This gem is compatible with Mongoid 3, 4, 5, 6 and 7.

### Configuration

#### Rails

Add to your `rails_helper.rb` file

```
1 require 'mongoid-rspec'
2
3 RSpec.configure do |config|
4   config.include Mongoid::Matchers, type: :model
5 end
```

#### Other

Add to your `spec_helper.rb` file.

```
1 require 'mongoid-rspec'
2
3 RSpec.configure do |config|
```

---

```
4   config.include Mongoid::Matchers
5   end
```

## Matchers

### be\_mongoid\_document

```
1   class Post
2     include Mongoid::Document
3   end
4
5   RSpec.describe Post, type: :model do
6     it { is_expected.to be_mongoid_document }
7   end
```

### be\_dynamic\_document

```
1   class User
2     include Mongoid::Document
3     include Mongoid::Attributes::Dynamic
4   end
5
6   RSpec.describe User, type: :model do
7     it { is_expected.to be_dynamic_document }
8   end
```

### have\_timestamps

With full timestamps.

```
1   class Log
2     include Mongoid::Document
3     include Mongoid::Timestamps
4   end
5
6   RSpec.describe Log, type: :model do
7     it { is_expected.to have_timestamps }
8   end
```

With short timestamps.

```
1   class User
2     include Mongoid::Document
3     include Mongoid::Timestamps::Short
```

---

```
4 end
5
6 RSpec.describe User, type: :model do
7   it { is_expected.to have_timestamps.shortened }
8 end
```

With only creating or updating timestamps.

```
1 class Admin
2   include Mongoid::Document
3   include Mongoid::Timestamps::Create
4   include Mongoid::Timestamps::Update
5 end
6
7 RSpec.describe Admin, type: :model do
8   it { is_expected.to have_timestamps.for(:creating) }
9   it { is_expected.to have_timestamps.for(:updating) }
10 end
```

With short creating or updating timestamps.

```
1 class Post
2   include Mongoid::Document
3   include Mongoid::Timestamps::Create::Short
4 end
5
6 RSpec.describe Short, type: :model do
7   it { is_expected.to have_timestamps.for(:creating).shortened }
8 end
```

### be\_stored\_in

```
1 class Post
2   include Mongoid::Document
3
4   store_in database: 'db1', collection: 'messages', client: 'secondary'
5 end
6
7 RSpec.describe Post, type: :model do
8   it { is_expected.to be_stored_in(database: 'db1', collection: '
      messages', client: 'secondary') }
9 end
```

It checks only those options, that you specify. For instance, test in example below will pass, even though expectation contains only `database` option.

```
1 class Comment
2   include Mongoid::Document
```

---

```
3
4   store_in database: 'db2', collection: 'messages'
5 end
6
7 RSpec.describe Comment, type: :model do
8   it { is_expected.to be_stored_in(database: 'db2') }
9 end
```

It works fine with lambdas and procs.

```
1 class User
2   include Mongoid::Document
3
4   store_in database: ->{ Thread.current[:database] }
5 end
6
7 RSpec.describe Post, type: :model do
8   it do
9     Thread.current[:database] = 'db3'
10    is_expected.to be_stored_in(database: 'db3')
11
12    Thread.current[:database] = 'db4'
13    is_expected.to be_stored_in(database: 'db4')
14  end
15 end
```

## have\_index\_for

```
1 class Article
2   index({ title: 1 }, { unique: true, background: true })
3   index({ title: 1, created_at: -1 })
4   index({ category: 1 })
5 end
6
7 RSpec.describe Article, type: :model do
8   it do
9     is_expected
10      .to have_index_for(title: 1)
11      .with_options(unique: true, background: true)
12   end
13   it { is_expected.to have_index_for(title: 1, created_at: -1) }
14   it { is_expected.to have_index_for(category: 1) }
15 end
```

## Field Matchers

```
1 RSpec.describe Article do
```

---

```
2   it { is_expected.to have_field(:published).of_type(Mongoid::Boolean).
      with_default_value_of(false) }
3   it { is_expected.to have_field(:allow_comments).of_type(Mongoid::
      Boolean).with_default_value_of(true) }
4   it { is_expected.not_to have_field(:allow_comments).of_type(Mongoid::
      Boolean).with_default_value_of(false) }
5   it { is_expected.not_to have_field(:number_of_comments).of_type(
      Integer).with_default_value_of(1) }
6 end
7
8 RSpec.describe User do
9   it { is_expected.to have_fields(:email, :login) }
10  it { is_expected.to have_field(:s).with_alias(:status) }
11  it { is_expected.to have_fields(:birthdate, :registered_at).of_type(
      DateTime) }
12 end
```

## Association Matchers

```
1 RSpec.describe User do
2   it { is_expected.to have_many(:articles).with_foreign_key(:author_id)
      .ordered_by(:title) }
3
4   it { is_expected.to have_one(:record) }
5
6   # can verify autobuild is set to true
7   it { is_expected.to have_one(:record).with_autobuild }
8
9   it { is_expected.to have_many :comments }
10
11  # can also specify with_dependent to test if :dependent => :destroy/:
    destroy_all/:delete is set
12  it { is_expected.to have_many(:comments).with_dependent(:destroy) }
13
14  # can verify autosave is set to true
15  it { is_expected.to have_many(:comments).with_autosave }
16
17  it { is_expected.to embed_one :profile }
18
19  it { is_expected.to have_and_belong_to_many(:children) }
20  it { is_expected.to have_and_belong_to_many(:children).of_type(User)
      }
21 end
22
23 RSpec.describe Profile do
24   it { is_expected.to be_embedded_in(:user).as_inverse_of(:profile) }
25 end
26
27 RSpec.describe Article do
```

---

```

28   it { is_expected.to belong_to(:author).of_type(User).as_inverse_of(:
      articles) }
29   it { is_expected.to belong_to(:author).of_type(User).as_inverse_of(:
      articles).with_index }
30   it { is_expected.to embed_many(:comments) }
31 end
32
33 RSpec.describe Comment do
34   it { is_expected.to be_embedded_in(:article).as_inverse_of(:comments)
      }
35   it { is_expected.to belong_to(:user).as_inverse_of(:comments) }
36 end
37
38 RSpec.describe Record do
39   it { is_expected.to belong_to(:user).as_inverse_of(:record) }
40 end
41
42 RSpec.describe Site do
43   it { is_expected.to have_many(:users).as_inverse_of(:site).ordered_by
      (:email.asc).with_counter_cache }
44 end
45
46 RSpec.describe Message do
47   it { is_expected.to belong_to(:user).with_optional }
48 end

```

## Validation Matchers

```

1  RSpec.describe Site do
2    it { is_expected.to validate_presence_of(:name) }
3    it { is_expected.to validate_uniqueness_of(:name) }
4  end
5
6  RSpec.describe User do
7    it { is_expected.to validate_presence_of(:login) }
8    it { is_expected.to validate_uniqueness_of(:login).scoped_to(:site) }
9    it { is_expected.to validate_uniqueness_of(:email).case_insensitive.
      with_message("is already taken") }
10   it { is_expected.to validate_format_of(:login).to_allow("valid_login"
      ).not_to_allow("invalid login") }
11   it { is_expected.to validate_associated(:profile) }
12   it { is_expected.to validate_exclusion_of(:login).to_not_allow("super
      ", "index", "edit") }
13   it { is_expected.to validate_inclusion_of(:role).to_allow("admin", "
      member") }
14   it { is_expected.to validate_confirmation_of(:email) }
15   it { is_expected.to validate_presence_of(:age).on(:create, :update) }
16   it { is_expected.to validate_numericality_of(:age).on(:create, :
      update) }

```

---

```

17   it { is_expected.to validate_inclusion_of(:age).to_allow(23..42).on
    ([:create, :update]) }
18   it { is_expected.to validate_presence_of(:password).on(:create) }
19   it { is_expected.to validate_presence_of(:provider_uid).on(:create) }
20   it { is_expected.to validate_inclusion_of(:locale).to_allow([:en, :ru
    ]) }
21   end
22
23   RSpec.describe Article do
24     it { is_expected.to validate_length_of(:title).within(8..16) }
25     it { is_expected.to validate_absence_of(:deletion_date) }
26   end
27
28   RSpec.describe Visitor do
29     it { is_expected.to validate_length_of(:name).with_maximum(160).
    with_minimum(1) }
30   end
31
32   RSpec.describe Profile do
33     it { is_expected.to validate_numericality_of(:age).greater_than(0) }
34   end
35
36   RSpec.describe MovieArticle do
37     it { is_expected.to validate_numericality_of(:rating).to_allow(:
    greater_than => 0).less_than_or_equal_to(5) }
38     it { is_expected.to validate_numericality_of(:classification).
    to_allow(:even => true, :only_integer => true, :nil => false) }
39   end
40
41   RSpec.describe Person do
42     # in order to be able to use the custom_validate matcher, the custom
    validator class (in this case SsnValidator)
43     # should redefine the kind method to return :custom, i.e. "def self.
    kind() :custom end"
44     it { is_expected.to custom_validate(:ssn).with_validator(SsnValidator
    ) }
45   end
46
47   # If you're using validators with if/unless conditionals, spec subject
    must be object instance
48   # This is supported on Mongoid 4 and newer
49   RSpec.describe User do
50     context 'when user has `admin` role' do
51       subject { User.new(role: 'admin') }
52
53       it { is_expected.to validate_length_of(:password).greater_than(20)
    }
54     end
55
56     context 'when user does not have `admin` role' do
57       subject { User.new(role: 'member') }

```

---

---

```
58
59     it { is_expected.not_to validate_length_of(:password) }
60   end
61 end
```

## Mass Assignment Matcher

```
1 RSpec.describe User do
2   it { is_expected.to allow_mass_assignment_of(:login) }
3   it { is_expected.to allow_mass_assignment_of(:email) }
4   it { is_expected.to allow_mass_assignment_of(:age) }
5   it { is_expected.to allow_mass_assignment_of(:password) }
6   it { is_expected.to allow_mass_assignment_of(:password) }
7   it { is_expected.to allow_mass_assignment_of(:role).as(:admin) }
8
9   it { is_expected.not_to allow_mass_assignment_of(:role) }
10 end
```

## Accepts Nested Attributes Matcher

```
1 RSpec.describe User do
2   it { is_expected.to accept_nested_attributes_for(:articles) }
3   it { is_expected.to accept_nested_attributes_for(:comments) }
4 end
5
6 RSpec.describe Article do
7   it { is_expected.to accept_nested_attributes_for(:permalink) }
8 end
```

## Contributing

You're encouraged to contribute to this library. See CONTRIBUTING for details.

## Copyright and License

Copyright (c) 2009-2018 Evan Sagge and Contributors.

MIT License. See LICENSE for details.