

# DSLs EM RUBY

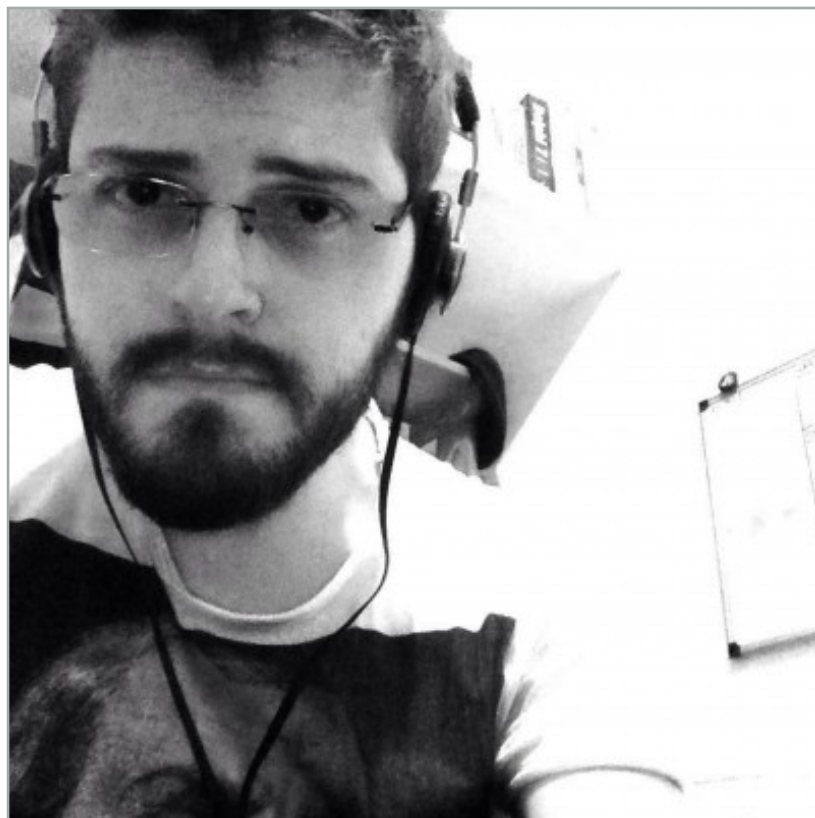
---

GUILHERME CARREIRO – TDC SP – 25.07.2015

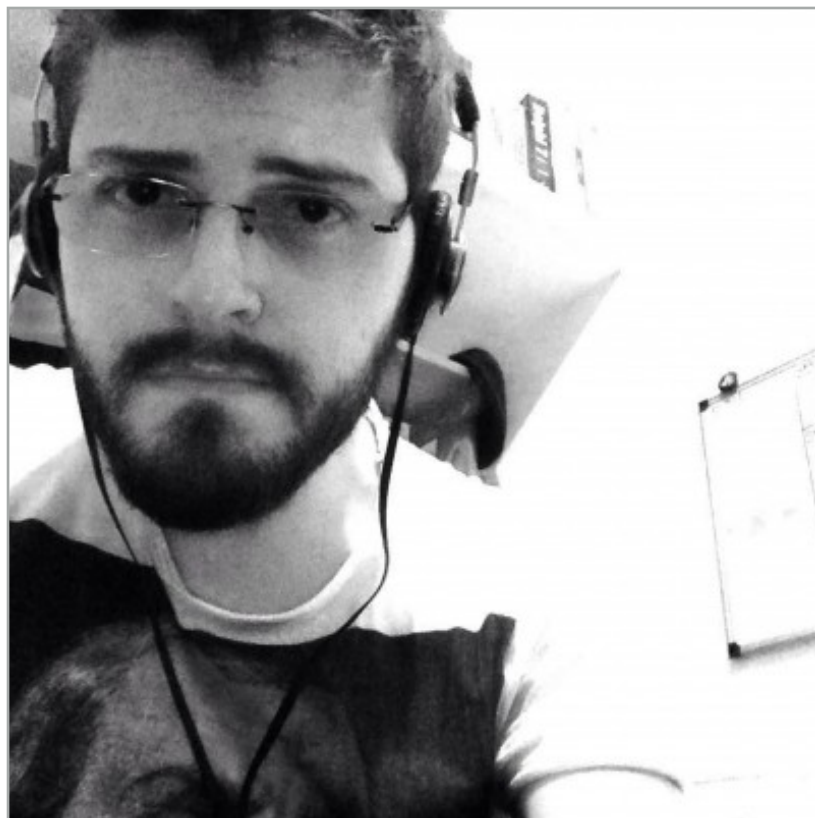
# DSLs EM RUBY

---

GUILHERME CARREIRO – TDC SP – 25.07.2015



**Software Craftsman**  
**Campinas, SP**  
**discourse/OneBox**  
**karreiro/picuture\_from**  
**dextra/projects**




**Software Craftsman**  
**Campinas, SP**  
**discourse/OneBox**  
**karreiro/picuture\_from**  
**dextra/projects**

**dextra**








GitHub, Inc. [github.com/discourse/onebox](#)

discourse/onebox

 This repository Search

Explore Gist Blog Help

 karreiro +   

 discourse / onebox

Watch 19

Unstar 272

Fork 64

A gem for turning URLs into website previews <https://github.com/discourse/onebox>

1,127 commits

4 branches


8 releases

36 contributors

branch: master

onebox / +

Merge pull request #294 from Elberet/patch-1

 ZogStrIP authored 4 days ago latest commit fb8265b1fd

lib	Bump version	5 days ago
spec	Workaround for failing redirects on custom maps	5 days ago
templates	Merge pull request #272 from bgorven/patch-1	28 days ago
web	style change: use go for onebox to action	3 months ago
.gitignore	add '.DS_Store' to .gitignore	3 months ago
.rspec	change rspec command line options to format documentation	2 years ago
.rubocop.yml	Fixed all the rubocop warnings	a year ago
.ruby-gemset	we want to renamespace to Onebox	2 years ago
.travis.yml	travis emails only when build status changes for both success/failure...	2 years ago
Gemfile	Update reference to gemspec in Gemfile to drop discourse-	a year ago
Guardfile	Get Guard working	a year ago
LICENSE.txt	initial commit	2 years ago
README.md	Updated docs for preview interface	3 months ago
Rakefile	FEATURE: debug web server for onebox engine development	3 months ago

<> Code

Issues 27

Pull requests 5

Pulse

Graphs

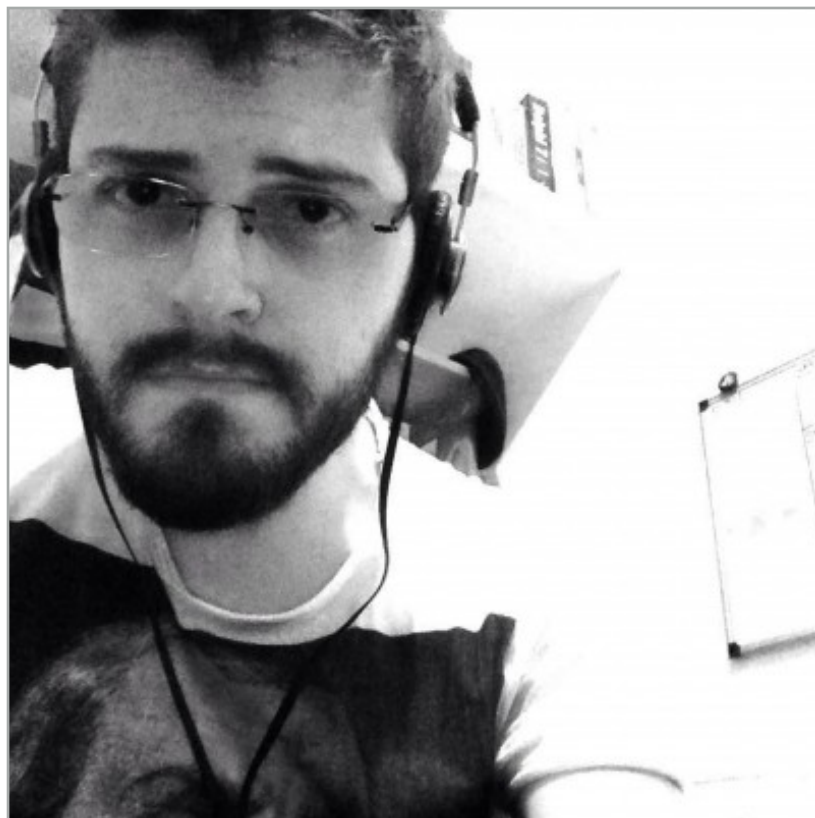
SSH clone URL

git@github.com:dis

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop

Download ZIP



**Software Craftsman**  
**Campinas, SP**  
**discourse/OneBox**  
**karreiro/picuture\_from**  
**dextra/projects**

# PictureFrom

build passing code climate 4.0 coverage 100%

PictureFrom is the most efficient library for getting profile pictures.

## Installation

```
gem install picture_from
```

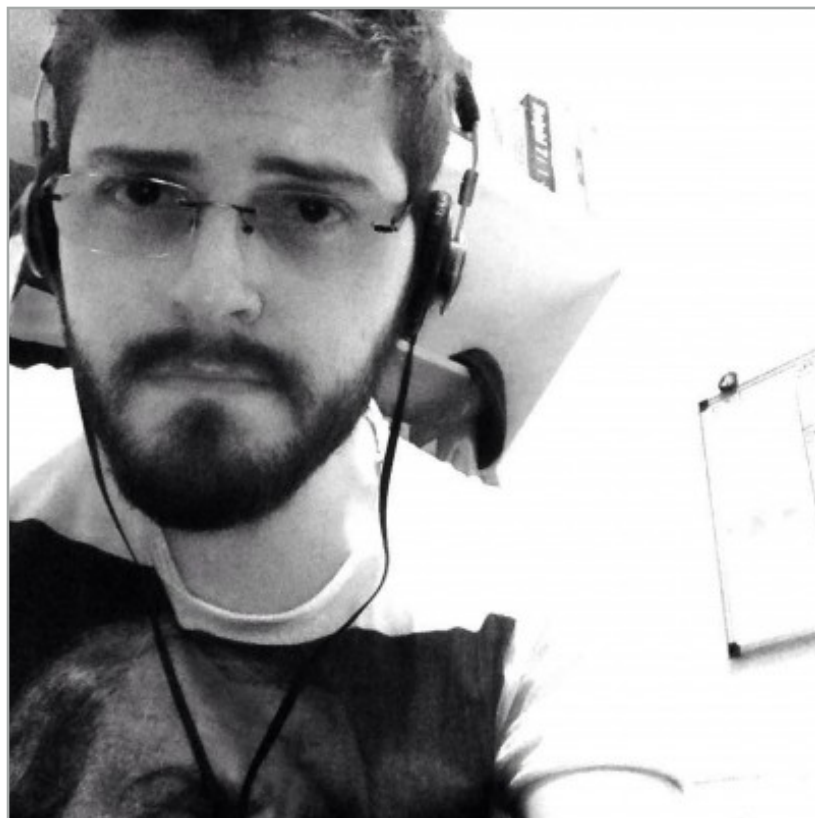
## Usage

Using PictureFrom is pretty simple! First, make sure the library is required:

```
require 'picture_from'
```

Pass an user information to the library's interface:

```
PictureFrom.url('zuck')  
# => http://graph.facebook.com/zuck/picture  
  
PictureFrom.url('Mark Zuckerberg')  
# => http://graph.facebook.com/zuck/picture  
  
PictureFrom.url('@unclebobmartin')  
# => https://pbs.twimg.com/profile_images/1102364992/clean_code_72_color_400x400.png
```



**Software Craftsman**  
**Campinas, SP**  
**discourse/OneBox**  
**karreiro/picuture\_from**  
**dextra/projects**





**:Ruby => ❤️**

(:Ruby => ♥) ?

```
class Person
  def initialize(first_name, last_name)
    @first_name = first_name;
    @last_name = last_name;
  end

  def full_name
    @first_name + ' ' + @last_name;
  end
end

person = Person.new('Guilherme', 'Carreiro');
person.full_name
# => Guilherme Carreiro
```

```
class Person {  
    private String firstName;  
    private String lastName;  
  
    public Person(String firstName, String lastName) {  
        this.firstName = firstName;  
        this.lastName = lastName;  
    }  
  
    public String fullName() {  
        return firstName + " " + lastName;  
    }  
}
```

```
Person person = new Person("Guilherme", "Carreiro");  
person.fullName();  
// => Guilherme Carreiro
```



```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletException;
import java.io.IOException;
import org.eclipse.jetty.server.Server;
import org.eclipse.jetty.server.Request;
import org.eclipse.jetty.server.handler.AbstractHandler;

public class HelloWorld extends AbstractHandler
{
    public void handle(String target,
                        Request baseRequest,
                        HttpServletRequest request,
                        HttpServletResponse response)
        throws IOException, ServletException
    {
        response.setContentType("text/html;charset=utf-8");
        response.setStatus(HttpServletResponse.SC_OK);
        baseRequest.setHandled(true);
        response.getWriter().println("Hello World");
    }

    public static void main(String[] args) throws Exception
    {
        Server server = new Server(8080);
        server.setHandler(new HelloWorld());
        server.start();
        server.join();
    }
}
```

```
require 'sinatra'
```

```
get '/' do  
  'Hello World'  
end
```





# ABSTRAÇÃO

---

CONVENTION OVER CONFIGURATION



# ABSTRAÇÃO

---

CONVENTION OVER CONFIGURATION

*!= Alienação*

---

The essence of abstractions is preserving information that is relevant in a given context, and forgetting information that is irrelevant in that context.

John V. Guttag

---

# DSL

---

DOMAIN SPECIFIC LANGUAGE



```
create_table :posts do |t|
  t.string :content

  t.timestamps null: false
end
```

```
class User
  has_one :job
end
```



```
get '/hello' do
  'Hello, World!'
end
```

```
Notification.new
  .every(30.minutes)
  .until(status: :read)
```

```
create_table :posts do |t|  
  t.string :content  
  
  t.timestamps null: false  
end
```

```
class User  
  has_one :job  
end
```



```
get '/hello' do  
  'Hello, World!'  
end
```

```
Notification.new  
  .every(30.minutes)  
  .until(status: :read)
```

```
create_table :posts do |t|  
  t.string :content
```

```
  t.timestamps null: false  
end
```



# cucumber

```
class User  
  has_one :job  
end
```



```
get '/hello' do  
  'Hello, World!'  
end
```

```
Notification.new  
  .every(30.minutes)  
  .until(status: :read)
```





---

[...] I use a subset of the capabilities of Ruby [...] as my syntax. To an extent, this **is more a matter of attitude than of anything else.**

Martin Fowler

---

```
create_table :posts do |t|  
  t.string :content
```

```
  t.timestamps null: false  
end
```



```
get '/hello' do  
  'Hello, World!'  
end
```

cucumber

```
class User  
  has_one :job  
end
```



```
Notification.new  
  .every(30.minutes)  
  .until(status: :read)
```

**EXTERNAL**

**&&**

**INTERNAL**

**INTERNAL**

## Notification

```
.new  
.every(30.minutes)  
.until(status: :read)
```

```
create_table :users do |t|
  t.string :first_name
  t.string :last_name
  t.string :email
  t.string :encrypted_password

  t.timestamps null: false
end
```

```
class User
  has_one :job
end
```



```
RSpec.describe Post, :type => :model do
  describe 'ActiveModel validations' do
    it { should validate_presence_of(:user) }
    it { should validate_presence_of(:content) }
  end

  describe 'ActiveRecord associations' do
    it { should belong_to(:user) }
  end
end
```

**EXTERNAL**

**Feature:** Trade reconciliation

**Scenario:** Trade logged against the wrong account

**Given** a trade logged on Mike's account

**But** the clearing house recorded it as Kim's trade

**When** the clearing house EOD report is reconciled against fills

**Then** the trade should be flagged as inconsistent

```
SELECT * FROM users;
```

[illegible]



# CONTEXTUAL

```
RSpec.describe PostsController, :type => :controller do
  before :each do
    sign_in create(:user)
  end

  describe 'GET #index' do
    before :each do
      get :index
    end

    it { should route(:get, posts_path).to(:action => :index) }
    it { should respond_with(:success) }
    it { should render_template(:index) }
    it { should_not set_flash }
  end
end
```



```
RSpec.configure do |config|
  config.use_transactional_fixtures = true
  config.infer_spec_type_from_file_location!
  config.render_views
  config.include FactoryGirl::Syntax::Methods
  config.include EmailSpec::Helpers
  config.include EmailSpec::Matchers
  config.include ActiveSupport::Testing::TimeHelpers
  config.include Devise::TestHelpers, :type => :controller
  config.include Warden::Test::Helpers
  config.before(:suite, :type => :feature) { Warden.test_mode! }
  config.after(:each, :type => :feature) { Warden.test_reset! }
end
```

```
class CreateUser < ActiveRecord::Migration
  def change
    create_table :users do |t|
      t.string :first_name
      t.string :last_name
      t.string :email
      t.string :encrypted_password

      t.timestamps null: false
    end

    add_index :users, :email, unique: true
  end
end
```



```
# config.rb
class PictureFrom::Config
  enable_cache!

  search_engine do
    setup :facebook, :twitter

    facebook_api do
      use :crawler, :api
      setup token: '...', secret: '...'
    end

    twitter_api do
      use :crawler
    end
  end
end
```

```
# config.rb
class PictureFrom::Config
  enable_cache!

  search_engine do
    setup :facebook, :twitter

    facebook_api do
      use :crawler, :api
      setup token: '...', secret: '...'
    end

    twitter_api do
      use :crawler
    end
  end
end
```

**DEMO**

```

# config.rb
class PictureFrom::Config
  enable_cache!

  search_engine do
    setup :facebook, :twitter

    facebook_api do
      use :crawler, :api
      setup token: '...', secret: '...'
    end

    twitter_api do
      use :crawler
    end
  end
end

```

```

module PictureFrom
  class Config
    class << self
      def enable_cache!; end

      def search_engine(&block)
        ctx = SearchEngine.new
        ctx.instance_eval &block
      end
    end

    class SearchEngine
      def setup(*apis); end

      def facebook_api(&block)
        ctx = FacebookApi.new
        ctx.instance_eval &block
      end

      def twitter_api(&block)
        ctx = TwitterApi.new
        ctx.instance_eval &block
      end

      class TwitterApi; end
      class FacebookApi; end
    end
  end
end

```

# FLUENT INTERFACE

```
# sem FluentInterface :(  
meetingTime = TimeInterval.new(fiveOClock, sixOClock);
```

```
# com FluentInterface :)  
meetingTime = fiveOClock.until(sixOClock);
```



```
DefaultMailer.new  
  .from('karreiro@gmail.com')  
  .to('jose@gmail.com')  
  .with_subject('Fotos da festa')  
  .with_body('As fotos ficaram ótimas: fotos.exe')  
  .send
```

```
class DefaultMailer
  def initialize
    @message = EmailMessage.new
  end

  def to(email)
    @message.to = email
    self
  end

  def with_subject(subject)
    @message.subject = subject
    self
  end

  def with_body(body)
    @message.body = body
    self
  end

  def send
    EmailDelivery.new(@message).deliver
  end
end
```

**METHOD MISSING**

```
class User
  def initialize(friends, pets = [])
    @friends = friends
    @pets = pets
  end
end
```

```
guilherme = User.new ['Eder', 'Elias', 'Esther', 'Paulo', 'Tiago']
```

```
guilherme.number_of_friends
```

```
guilherme.number_of_pets
```

```
class User
  def initialize(friends, pets = [])
    @friends = friends
    @pets = pets
  end
end
```

```
guilherme = User.new ['Eder', 'Elias', 'Esther', 'Paulo', 'Tiago']
```

```
guilherme.number_of_friends
```

```
# => NoMethodError
```

```
guilherme.number_of_pets
```

```
# => NoMethodError
```

```
module NumberOf
  def method_missing(name)
    super unless name.to_s.start_with?('number_of_')
    instance_variable_get("@#{name}".gsub /number_of_/, '').size
  end
end
```

```
class User
  include NumberOf
  def initialize(friends, pets = [])
    @friends = friends
    @pets = pets
  end
end
```

```
guilherme = User.new ['Eder', 'Elias', 'Esther', 'Paulo', 'Tiago']
guilherme.number_of_friends
# => 5
guilherme.number_of_pets
# => 0
```

```
module NumberOf
  def method_missing(name)
    super unless name.to_s.start_with?('number_of_')
    instance_variable_get("@#{name}".gsub /number_of_/, '').size
  end
end
```

```
class User
  include NumberOf
  def initialize(friends, pets = [])
    @friends = friends
    @pets = pets
  end
end
```

```
guilherme = User.new ['Eder', 'Elias', 'Esther', 'Paulo', 'Tiago']
guilherme.number_of_friends
# => 5
guilherme.number_of_pets
# => 0
guilherme.respond_to? :number_of_pets
# => false
```

```
module NumberOf
  def method_missing(name)
    super unless name.to_s.start_with?('number_of_')
    instance_variable_get("@#{name}".gsub /number_of_/, '').size
  end

  def respond_to_missing?(name, include_all)
    name.to_s.start_with?('number_of_') || super
  end
end
```

```
class User
  include NumberOf
  def initialize(friends, pets = [])
    @friends = friends
    @pets = pets
  end
end
```

```
guilherme = User.new ['Eder', 'Elias', 'Esther', 'Paulo', 'Tiago']
guilherme.number_of_friends
# => 5
guilherme.number_of_pets
# => 0
guilherme.respond_to? :number_of_pets
# => true
```



**DECLARATIVE**

```
class User
  include ImageTrick
  attr_accessor :name, :image
  has_attached_file :image
end
```

```
user = User.new.tap { |u| u.image = '123456.png' }
user.image_url
# => "https://imagetrick.com/123456.png"
```

```
module ImageTrick
  def self.included(klass)
    klass.extend(ClassMethods)
  end
  module ClassMethods
    def has_attached_file(name)
      define_method "#{name}_url" do
        image_name = instance_variable_get("@#{name}")
        "https://imagerick.com/#{image_name}"
      end
    end
  end
end
```

```
class User
  include ImageTrick
  attr_accessor :name, :image
  has_attached_file :image
end
```

```
user = User.new.tap { |u| u.image = '123456.png' }
user.image_url
# => "https://imagerick.com/123456.png"
```





**REGEX RSPEC ACTIVE**

**MODEL XPTO ACTIVE**

**RECORD MIGRATIONS**

**ACTIVE RESOURCE**

**SINATRA JAVASCRIPT**

# METAPROGRAMMING



# METAPROGRAMMING

<http://www.slideshare.net/karreiro/tdc-2015-metaprogramao-na-prtica-48235897>



**DEBUG**

**OVERHEAD**



# DSLs



*The Addison-Wesley Signature Series*



# DOMAIN- SPECIFIC LANGUAGES

MARTIN FOWLER  
WITH REBECCA PARSONS



# OBRIGADO :)

---

Twitter: @karreiro\_ – GitHub: @karreiro