Apaixonados por Eficiência

in|metrics

Automação com BDD em Ruby + Capybara

DOJO NÍVEL II

in|metrics

Dinâmica Dojo

Agenda

- Esquema do Cenário
- Pry
- Tags
- SitePrism

1

Esquema do Cenário

Scenario Outline

Esquema do Cenario
Scenario Outline

```
Cenário: Comer 5 de 12 laranjas
    Dado que eu tenha 12 laranjas
    Quando eu como 5 laranjas
    Entao me resta 7 laranjas

Cenário: Comer 7 de 20 laranjas
    Dado que eu tenha 20 laranjas
    Quando eu como 7 laranjas
    Entao me resta 13 laranjas
```

Esquema do Cenario
Scenario Outline

```
Esquema do Cenário: Comer laranjas
    Dado que eu tenha "<qtd_inicial>" laranjas
    Quando eu como "<qtd_consumida>" laranjas
    Entao me resta "<qtd_restante>" laranjas

    Exemplos:
        |   qtd_inicial |   qtd_consumida   |   qtd_restante    |
        |   12          |   5               |   7               |
        |   20          |   7               |   13              |
```

Exemplo de Esquema do Cenário

```
Funcionalidade: Buscar Agencias
    Eu como cliente do banco
    Quero procurar uma agencia dentro do Brasil
    Para saber suas informacoes de contato

    Esquema do Cenario: Buscar agencias por CEP valido
        Dado que eu esteja na home do site do banco
        Quando buscar uma agencia pelo "<CEP>"
        Entao apresentara as agencias disponiveis

        Exemplos:
            |   CEP         |
            |   18410000    |
            |   18141000    |
```

Vamos Praticar?

Desafio 01

Esquema do Cenário

Scenario Outline

Utilizando o conceito de **Esquema do Cenário**, enviar formulários de contato através do site https://vtrmartinez.wixsite.com/dojo informando os nomes abaixo (e demais informações obrigatórias):

Eduardo, Ewerton, Felipe, Luiz

2

PRY

in

Pry

Debug no Ruby

O **Pry** é uma gem do Ruby que nos permite fazer o ***debug*** do nosso código.

Este ***debug*** é feito no **irb** pelo terminal.

Debug, é o ato de depurar o código. Olhar passo a passo cada ação para identificar possíveis erros no fluxo

***IRB**, significa **Interactive Ruby Shell.** Basicamente ele habilitará todos os comandos ruby em seu terminal, onde você poderá executar e ver os resultados em tempo real.

Pry

Debug no Ruby

Instalando o pry

- Incluir a gem no arquivo *Gemfile* `gem 'pry'`
- Efetuar o require da gem no *ENV.rb* `require 'pry'`
- Dentro da pasta do projeto, executar o comando no terminal: *bundle* `✗ bundle`

Usando o pry

O pry pode ser utilizado nos steps ou métodos da automação. Para tal, basta inserir o seguinte comando:

```
binding.pry
```

Pry - Exemplo

Debug no Ruby

```
13  Quando(/^eu buscar agência no bairro$/) do
14    @app.agency.click_something("Agência")
15    binding.pry
16    @app.agency.click_something("Clique aqui")
17  end
18
```

Pry - Exemplo

Debug no Ruby

```
Cenario: Buscar agência por CEP válido
  Dado que eu esteja na home do site Santander
  Quando eu buscar agência pelo CEP
  Então aparecerá as agências disponíveis do CEP

Cenario: Buscar agência por bairro
  Dado que eu esteja na home do site Santander

From: /Users/vtrmartinez/Documents/SantanderSite/SantanderSitePrism/features/step_definitions/agency_steps.rb @ line 15 :

   10:   end
   11: end
   12:
   13: Quando(/^eu buscar agência no bairro$/) do
   14:   @app.agency.click_something("Agência")
=> 15:   binding.pry
   16:   @app.agency.click_something("Clique aqui")
   17:
   18:   @app.agency.last_window
   19:   within_frame(@app.searchAgency.iframe) do
   20:     @app.searchAgency.neighborhood.neighborhoodTab.click
```

Pry - Comandos

Debug no Ruby

```
Navigating pry
  !pry                Start a pry session on current self.
  disable-pry         Stops all future calls to pry and exits the current session.
  exit                Pop the previous binding.
  exit-all            End the current pry session.
  exit-program        End the current program.
  jump-to             Jump to a binding further up the stack.
  nesting             Show nesting information.
  switch-to           Start a new subsession on a binding in the current stack.
```

3

Cucumber Tags

in

Cucumber Tags

Organizando seus testes!

Tag é uma ótima maneira de organizar a execução de suas funcionalidades e cenários:

```
@regressao
Funcionalidade: Registro de usuários

    @importante
    Cenário: Cadastrar cliente

    Cenário: Alterar cadastro do cliente
```

Uma funcionalidade ou cenário pode ter várias **tags**. Basta separar por ‘espaço’:

```
@regressao @diario @smoke
Funcionalidade: Registro de usuários
```

Cucumber Tags

Organizando seus testes!

```
@consulta
Funcionalidade: Pesquisa de termos
    Eu como usuário do sistema
    Quero consultar um termo
    Para saber mais informações referentes ao termo

    Cenário: Pesquisar termos com filtros ativados
        Dado que eu esteja na area de consulta
        Quando aplico o filtros "em estoque" e "pronta entrega"
        E eu pesquiso o termo "café"
        Então eu vejo detalhes sobre o termo
```

Alguns exemplos de tags:

- Momentos em que os testes devem rodar
  @hourly, *@daily*
- De acordo com suas dependências externas
  @local, *@database*, *@network*
- Nível
  @functional, *@system*, *@smoke*
- Sistemas/Produtos
  @home, *@carrinho*

Cucumber Tags - Exemplos de Execução

Organizando seus testes!

```
cucumber --tags @pendente # Executa os cenários/funcionalidades com a tag @pendente
cucumber --tags @smoke # Executa os cenários/funcionalidades com a tag @smoke
cucumber --tags ~@pendente # Executa somente os que NÃO tenham a tag @pendente
cucumber --tags @pendente --tags @smoke # Executa os que tenham a tag @pendente E @smoke
cucumber --tags @pendente,@smoke # Executa os que tenham a tag @pendente OU @smoke
cucumber -t @pendente # -t é um atalho para --tags
```

No terminal, quando quisermos rodar os cenários que estão nomeados com tags, basta utilizarmos o seguinte comando:

cucumber --tags @nome_da_tag

ou

cucumber -t @nome_da_tag

4

Cucumber Profile

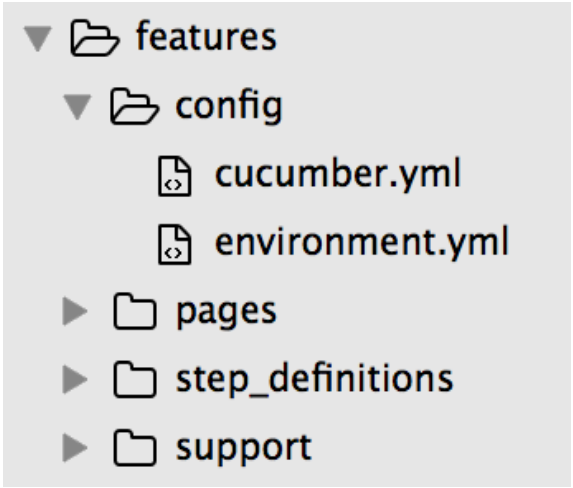
Cucumber Profile

Organizando seus testes!

```
default: --no-source --color --format pretty

html_report: --color -f html -o report_execucao.html

smoke: -t @importante -t @funcional -t @carrinho
```



*para melhorar a organização do projeto, crie o arquivo ***cucumber.yml*** dentro de uma pasta config.

Cucumber Profile - Exemplos de Execução

Organizando seus testes!

```
# Executa os testes parametrizados no profile html_report
cucumber --profile html_report

# -p é um atalho para --profile
cucumber -p smoke

# Executa conforme profile html_report, mas sem incluir os testes com a tag @funcional
cucumber -p html_report -t ~@funcional

# Exibe mais informações das opções que podem ser informadas ao rodar o cucumber
cucumber -h
```

Vamos Praticar?

Desafio 02

Pry e Tags

Aplicar os conceitos vistos anteriormente (***Pry*** e ***Tags***) no **Esquema do Cenário** **que** criamos no Desafio 01

4

SitePrism

in

SitePrism

Aplique Page Objects!

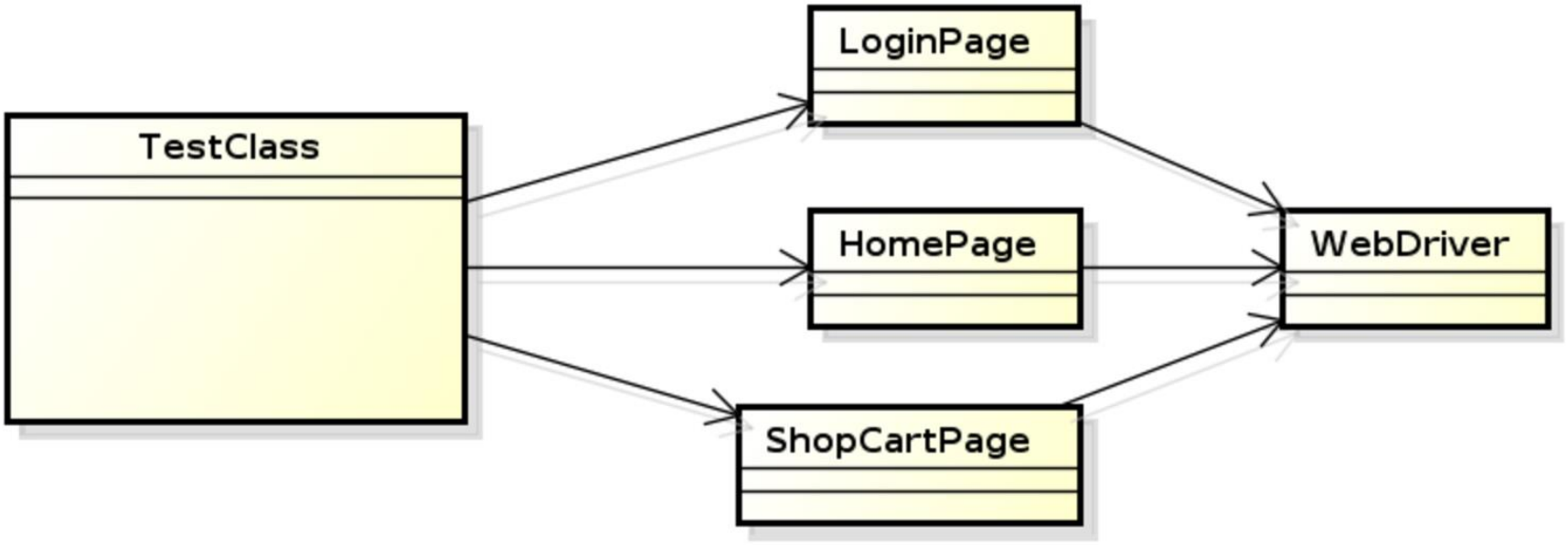
Gem do Ruby que ajuda no mapeamento ou organização de todos os elementos que serão utilizados no projeto de automação.

Facilita o uso do famoso padrão de projeto chamado de **Page Object**.

Sua semântica/sintaxe é tão legível quanto o Capybara (que nós já conhecemos)

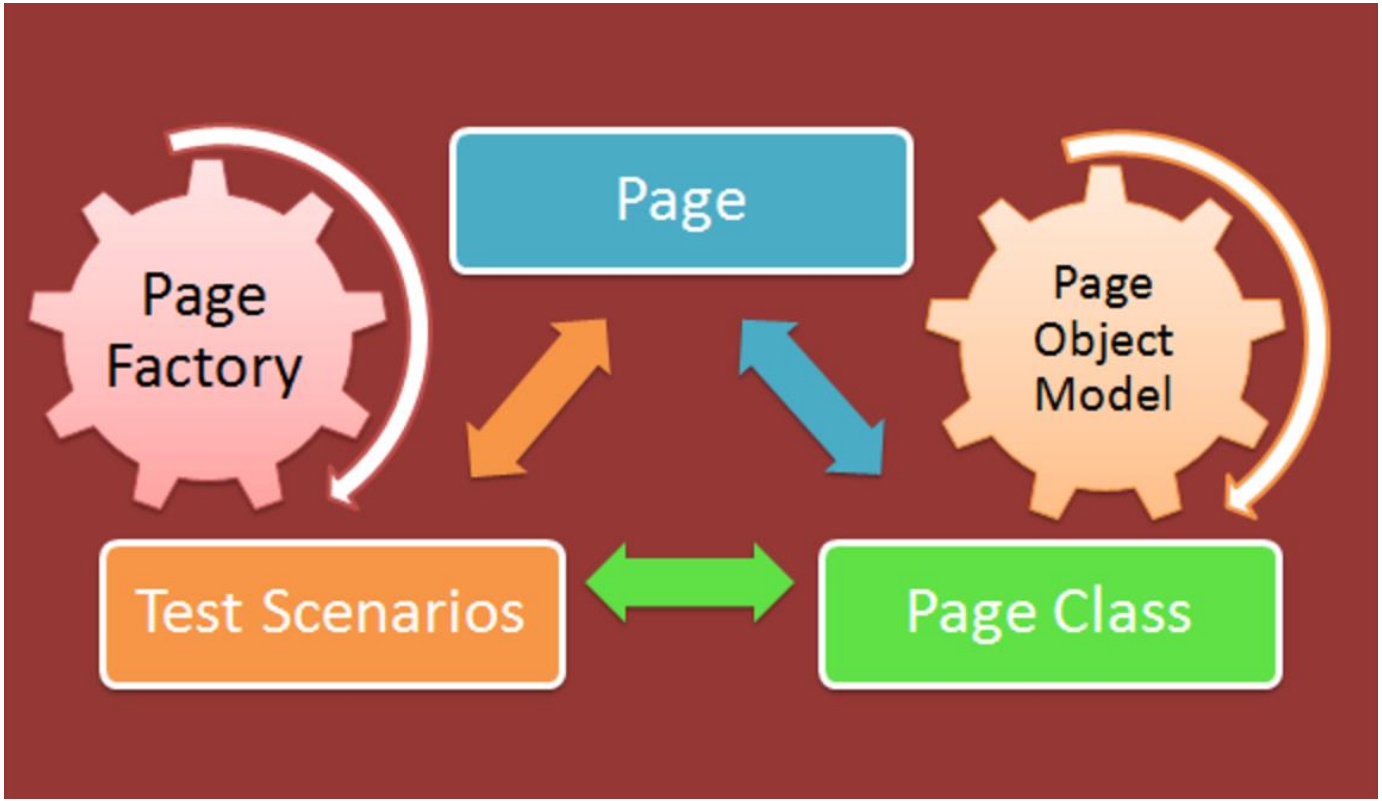
SitePrism

Aplique Page Objects!



SitePrism

Aplique Page Objects!

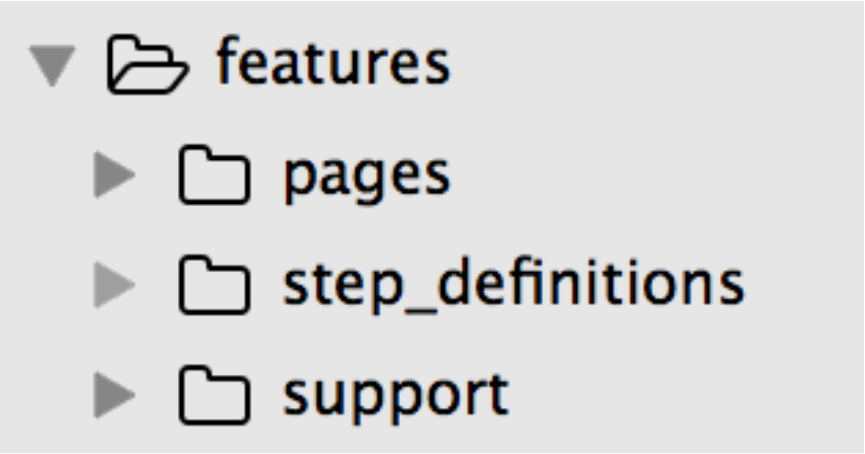


Instalando o SitePrism.

- Incluir a gem no arquivo *Gemfile* `gem 'site_prism'`
- Efetuar o require da gem no *env.rb* `require 'site_prism'`
- Dentro da pasta do projeto, executar o comando no terminal: *bundle* `✗ bundle`

SitePrism

Aplique Page Objects!

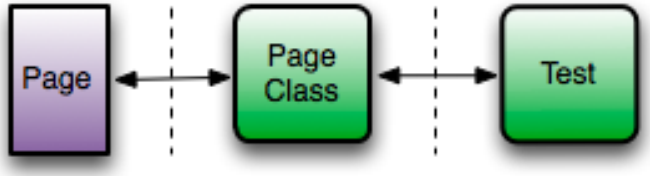


Crie uma pasta para incluir suas classes de ***Page Object*** dentro da pasta features do seu projeto.

Exemplo:
- **pages** para automação de front-end
- **services** ou **objects** para back-end

SitePrism

Aplique Page Objects!



features/step_definitions/user_steps.rb

```
1  Given /^I login with username "Joseph" and password "cuker"$/ do |username, password|
2    login_page = LoginPage.new
3
4    login_page.visit
5    login_page.login(username, password)
6  end
```

features/pages/login_page.rb

```
1  class LoginPage
2    def login(user, password)
3      fill_in :user, user
4      fill_in :password password
5      click 'login'
6    end
7
8    def visit
9      visit "/login"
10   end
11 end
```

SitePrism - Criando suas Classes Page Object

Aplique Page Objects!

A declaração de uma classe no Ruby, é feita da seguinte maneira:

home.rb

```ruby
class Home

end
```

Para a criação de uma Page Model com SitePrism devemos acrescentar:

home.rb

```ruby
class Home < SitePrism::Page

end
```

SitePrism - Mapeando Elementos

Aplique Page Objects!

Para declaração dos elementos, devemos seguir a seguinte estrutura:

home.rb

```ruby
class Home < SitePrism::Page
  element :nearTab, "#OpcaoBuscaAgenProxima"
  element :cep, "#refCep"
  element :search, :css,"#BuscaAgenProximaForm > ul.botoes > li.alignR > a > img"
  element :address, "#refEndereco"
end
```

```ruby
class Home < SitePrism::Page


    # set_url define uma url padrão de acesso.
    set_url "http://www.extra.com.br/"

    # declarando um elemento
    element :nome_do_elemento, 'identificador_do_elemento_id_ou_nome_ou_etc'

    # declarando um elemento único
    element :primeiro_produto_da_lista, '#produto_01'

    # declarando uma lista de elementos
    elements :lista_itens_vitrine, 'div.lista-produto.prateleira'

end
```

SitePrism - Mapeando Seções

Aplique Page Objects!

Podemos também dividir o mapeamento dos elementos por seções dentro da página. Exemplo:

search_agency.rb

```ruby
class SearchAgencyNear < SitePrism::Section
  element :nearTab, "#OpcaoBuscaAgenProxima"
  element :cep, "#refCep"
  element :search, :css,"#BuscaAgenProximaForm > ul.botoes > li.alignR > a > img"
  element :address, "#refEndereco"
end

class SearchAgency < SitePrism::Page
  section :near, SearchAgencyNear, "#ctBuscaAgencia"
  element :iframe, "iframe"
end
```

SitePrism - Exemplo de Utilização

Aplique Page Objects!

```ruby
class Home < SitePrism::Page


    # set_url define uma url padrão de acesso.
    set_url "http://www.extra.com.br/"

    # declarando um elemento
    element :nome_do_elemento, 'identificador_do_elemento_id_ou_nome_ou_etc'

    # declarando um elemento único
    element :primeiro_produto_da_lista, '#produto_01'

    # declarando uma lista de elementos
    elements :lista_itens_vitrine, 'div.lista-produto.prateleira'

end
```

```ruby
Dado(/^que esteja na home do website$/) do
    Home.new.load
end

Dado(/^que escolha o primeiro produto da vitrine$/) do
    Home.new.primeiro_produto_da_lista.click
end
```

SitePrism - Exemplo de Utilização

Aplique Page Objects!

agency_steps.rb

```ruby
Quando(/^eu buscar agência pelo CEP$/) do
  @app.agency.click_something("Agência")
  @app.agency.click_something("Clique aqui")

  @app.agency.last_window
  within_frame(@app.searchAgency.iframe) do
    @app.searchAgency.near.cep.set ADDRESS['NEAR']['CEP']
    @app.searchAgency.near.search.click
    @app.searchAgency.near.address.select(ADDRESS['NEAR']['ADDR
  end
end

Quando(/^eu buscar agência no bairro$/) do
  @app.agency.click_something("Agência")
  @app.agency.click_something("Clique aqui")

  @app.agency.last_window
  within_frame(@app.searchAgency.iframe) do
    @app.searchAgency.neighborhood.neighborhoodTab.click
    @app.searchAgency.neighborhood.selectState.select(ADDRESS['
    @app.searchAgency.neighborhood.selectCity.select(ADDRESS['N
    @app.searchAgency.neighborhood.selectNeighborhood.select(AD
    @app.searchAgency.neighborhood.search.click
  end
end
```

search_agency.rb

```ruby
class SearchAgencyNear < SitePrism::Section
  element :nearTab, "#OpcaoBuscaAgenProxima"
  element :cep, "#refCep"
  element :search, :css,"#BuscaAgenProximaForm > ul.botoes > li.a
  element :address, "#refEndereco"
end

class SearchAgencyNeighborhood < SitePrism::Section
  element :neighborhoodTab, "#OpcaoBuscaAgenBairro"
  element :selectState, "#localizacaoEstado"
  element :selectCity, "#localizacaoCidade"
  element :selectNeighborhood, "#localizacaoBairro"
  element :search, :css,"#BuscaAgenBairro > table > tbody > tr >
end

class SearchAgencyNumber < SitePrism::Section
  element :numberTab, "#OpcaoBuscaAgenNumero"
  element :agencyNumber, "#txNumeroAgencia"
  element :search, :css,"#BuscaAgenNumero > table > tbody > tr >
end

class SearchAgencyRoute < SitePrism::Section
  element :routeTab, "#OpcaoBuscaAgenRota"
  element :cepOrigin, "#refCepOrigem"
  element :cepDestiny, "#refCepDestino"
```

SitePrism - Exemplo de Utilização

Aplique Page Objects!

```
class LoginPage < SitePrism::Page
  element :username_field, "input[name='username']"
  element :password_field, "input[name='password']"
  element :flash, "div.flash"

  def log_in(username, password)
    username_field.set(username)
    password_field.set(password)
    click_on('Log In')
  end
end

class ProfilePage < SitePrism::Page
  element :flash, "div.flash"

  def flash_message
    flash.text
  end
end
```

Vamos Praticar?

Desafio 03

SitePrism

Elabore um cenário que crie 3 registros no site http://demoqa.com/ utilizando todos os conceitos vistos anteriormente

- Todos os cenários estão verdes?

Parabéns!!!

Retrospectiva

- O que foi bom?
- O que podemos melhorar?
- O que não foi bom?

Desafio!!!

Automatizando na prática!

Automatize o seguinte cenário, seguindo as definições que aprendeu nesse dojo:

1. Logar no site **ORANGEHRM** com perfil **ADMIN** e cadastrar um novo empregado <*Add Employee*> (*Devemos utilizar o SitePrism e @tag*)

2. Validar obrigatoriedade dos campos da tela cadastrar um novo empregado <*Add Employee*> (*Devemos utilizar o Scenario Outline e @tag*)

3. Logar no site ORANGEHRM com perfil ADMIN e editar um empregado existente na Lista de Empregado <Employee List> (*Devemos utilizar o Page Object, @tag e Pry*)

Dados de acesso:

https://enterprise-demo.orangehrmlive.com/

username: "Admin"

password: "admin"

Lembre-se: Dúvidas? dojo_automacao@lnmetrics.com.br :D

Links Importantes

Automatizando na prática!

Curso de Ruby: https://www.codecademy.com/learn/ruby

Boas praticas Cucumber: https://github.com/strongqa/howitzer/wiki/Cucumber-Best-Practices

Documentação do Cucumber: https://docs.cucumber.io/

Boas práticas SitePrism: https://github.com/natritmeyer/site_prism

Gherkin: https://github.com/cucumber/cucumber/tree/master/gherkin

in|metrics

Barueri . +55 11 3303-3200

Av. Tamboré, 267 - 21º andar
Torre Norte, Tamboré, Barueri - SP - Brasil
CEP: 06460-000

São Paulo . +55 11 3303-3200

Av. Eng. Luiz Carlos Berrini, 105 - 16º andar
Sala 1607, Brooklin Novo, São Paulo - SP - Brasil
CEP: 04571-010

Rio de Janeiro . +55 21 3232-2611

Rua Visconde de Inhaúma, 134 - 20º andar
Sala 2016/2017, Centro, Rio de Janeiro - RJ - Brasil
CEP: 20091-007

Chile . +56 2 3203-9507

Cerro El Plomo, 5420
Oficina 1503, Las Condes, Santiago - Chile
Código Postal: 7560742

Colômbia . +57 1 646-9642

Carrera 19A #90-13
Oficina 304, Bogotá - Colômbia
Código Postal: 110221