Automatización en Ruby 101

—

Denis Rodriguez
Santiago Vazquez

Soy Denis

—

QA
Altimetrik
Salesforce

Altimetrik y Salesforce

Altimetrik:

Empresa de software que tiene mas de 8 años en Uruguay

Lider en cloud computing

Especializada en Salesforce

Salesforce:

CRM que ofrece soluciones de Cloud computing

Solución diferente

Proyecto web
Extensión superior al año de trabajo
Cantidad de casos de prueba
Ciclos de ejecución demandantes

La conclusión es que la automatización es necesaria.

¿Qué es Ruby?

Lenguaje de programación dinámico

Código abierto

Simple y de sintaxis elegante

Fácil de leer y de escribir

Flexible y versatil

Portable

ATOM - Increíble editor de texto

Atom es un editor de texto open source altamente customizable

Permite la instalación de plugins para trabajar en entornos específicos

Tiene paquetes útiles para trabajar con Ruby

https://atom.io/

Rspec

drafts_page_spec.rb

```
describe Drafts do
  let(:current_user) do
    instance_double(User, :first_name => 'UnitTestUser')
  end
  let(:params) do
    {name: 'unit_test_page', head_layout: '', header_layout: '', nav_layout: '',
     related_to: '', layout: 'dynamic', use_head_layout: 'false', use_header_lay
     use_footer_layout: 'false', page_layout_id: PageLayout.first.id, use_templa
     proxy_cache_secs: '', version_comment_hdn: 'This is the comment for testing
  end
  let(:locale_params) do
    {en: {'page_title' => 'EN title', 'meta_description' => 'EN MD', 'head' => '
     de: {'page_title' => 'DE title', 'meta_description' => 'DE MD', 'head' => '
     ja: {'page_title' => 'JA title', 'meta_description' => 'JA MD', 'head' => '
     fr: {'page_title' => 'FR title', 'meta_description' => 'FR MD', 'head' => '
    }
  end
```


Cucumber intro

¿Qué es?

Es una herramienta de programación usada para correr casos de prueba automatizados escritos en BDD (Behavior-Driven development)

Está escrito en Ruby pero también existen versiones soportadas para JAVA

Es orientado al cliente

Está dividido en **Features**, subdividido en **Scenarios** los que contienen **Steps Definitions**

Cucumber caso de prueba

salesforce_login.feature

```
Feature: Login to Salesforce and create a simple object

  Scenario: Login to Salesforce and create a simple object
    Given I am in Salesforce Login page
    Then I enter a valid username
    And I enter a valid password
    And I click the Login button
    Then I check that I am logged in Salesforce Org
    And I write Objects in the Quick Find text field on Salesforce Setup page
    And I click Create > Objects link
    And I click New Custom Object button inside Custom Objects page
    Then I check that I am in New Custom Object page
```


Cucumber estructura

salesforce_login.feature

```
Feature: Login to Salesforce and create a simple object

  Scenario: Login to Salesforce and create a simple object
    Given I am in Salesforce Login page
    Then I enter a valid username
    And I enter a valid password
    And I click the Login button
    Then I check that I am logged in Salesforce Org
    And I write Objects in the Quick Find text field on Salesforce Setup page
    And I click Create > Objects link
    And I click New Custom Object button inside Custom Objects page
    Then I check that I am in New Custom Object page
```

salesforce_login_steps.rb

```
Given(/^I am in Salesforce Login page$/) do
  @browser.goto 'login.salesforce.com'
end

Then(/^I enter a valid username$/) do
  @browser.text_field(:css => '#username').set 'your login username here'
end

And(/^I enter a valid password$/) do
  @browser.text_field(:css => '#password').set 'your password'
end

And(/^I click the Login button$/) do
  @browser.button(:css => '#Login').click
end

Then(/^I check that I am logged in Salesforce Org$/) do
  @browser.wait_until(30) {@browser.image(:css => '#phHeaderLogoImage').exist?}
end

And(/^I write Objects in the Quick Find text field on Salesforce Setup page$/) do
  sleep(3)
  @browser.text_field(:css => '#setupSearch').set 'Objects'
end
```

Cucumber Palabras Clave de Test Scenario

Un **Scenario** está definido por una secuencia de **Steps**.
Las palabras clave en los steps son:

Given: Estado inicial antes de iniciar el test, precondición

When: Acción a realizar por el usuario en el test

Then: Resultado esperado de la acción realizada

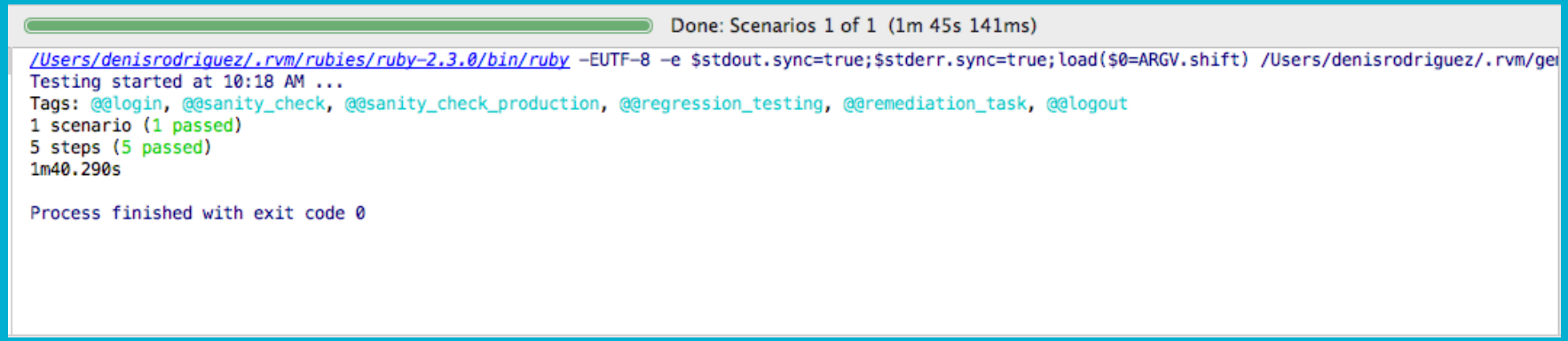
También se usan para combinar las palabras clave uniones lógicas

And: Misma acción que en expresiones lógicas

But: Igual que el And pero usada de manera negativa

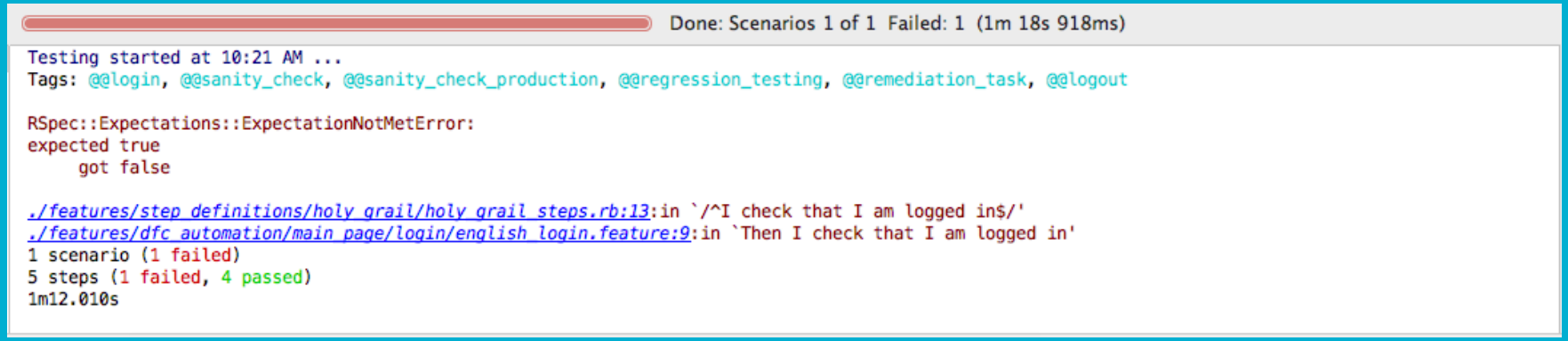
Cucumber 3 estados de ejecución

Pass



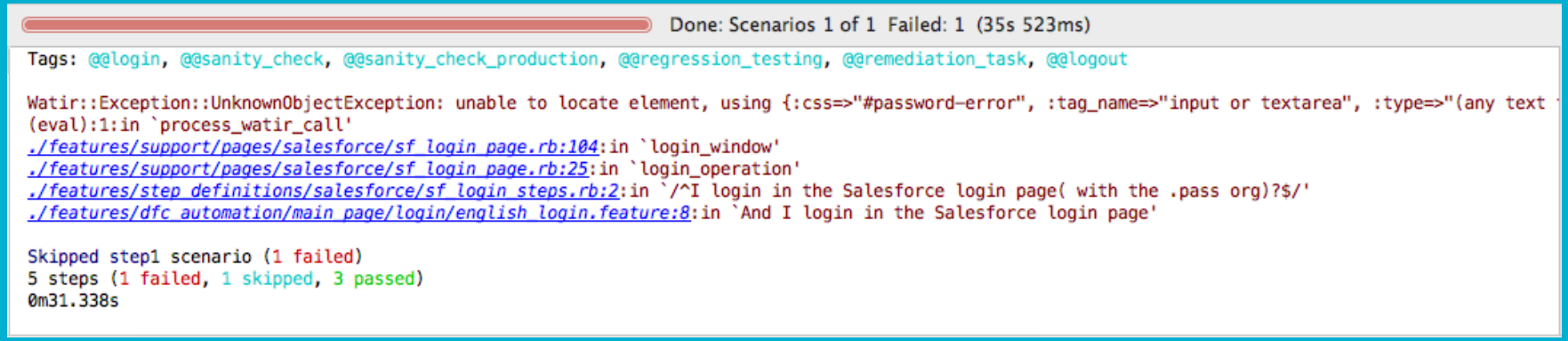
Cucumber 3 estados de ejecución

Fail

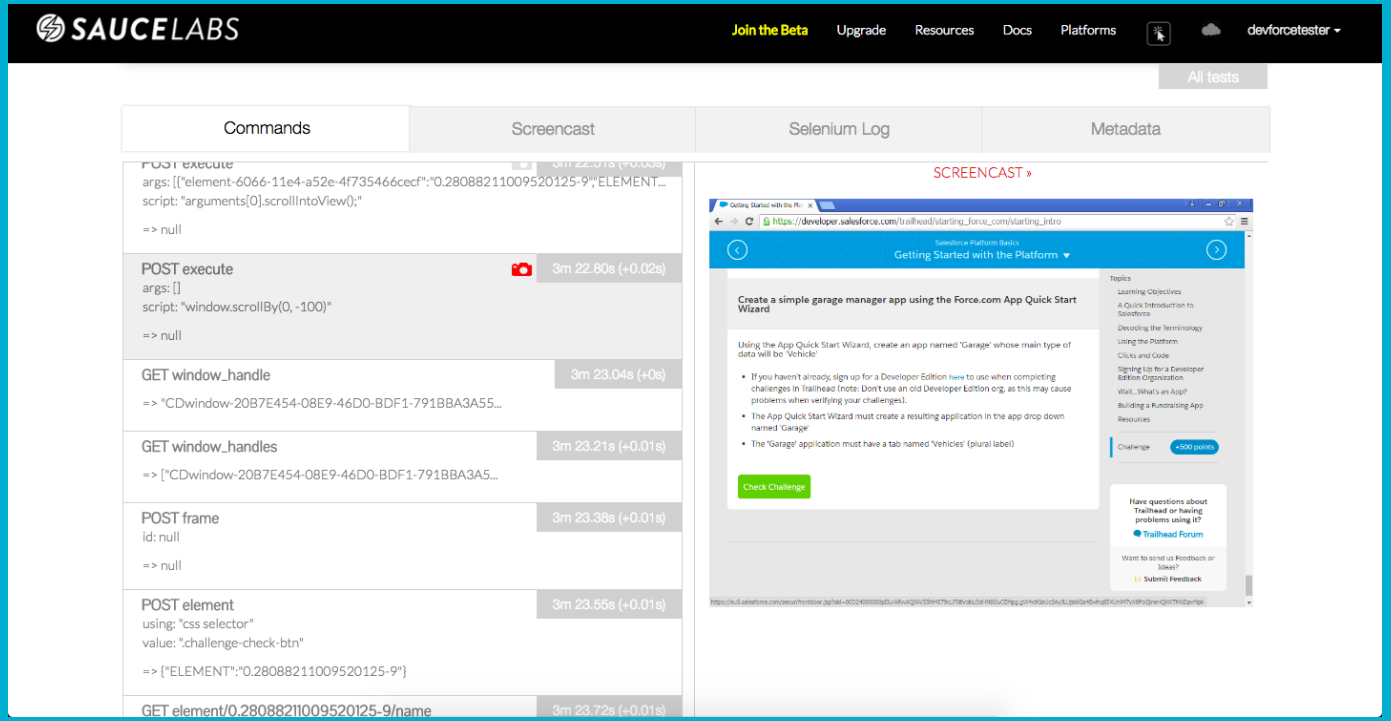


Cucumber 3 estados de ejecución

Error



Cucumber con SauceLabs



Principales ventajas de automatización

Los test pueden ser re usados

Correr tests de manera rápida y eficaz

Todos los integrantes del equipo pueden ver los resultados

Aunque tome tiempo al principio. Una vez que los casos de prueba son automatizados, el proceso se vuelve mucho más rápido

Es más interesante escribir/codear un Test Case en Cucumber que llenar manualmente la misma Form 100 veces en un test manual

Quien dijo que los QA no pueden escribir código?

Conclusión

Todos los proyectos pueden automatizar

Mas eficiencia en projectos largos

Ruby es una herramienta estupenda para empezar

¡Muchas gracias!