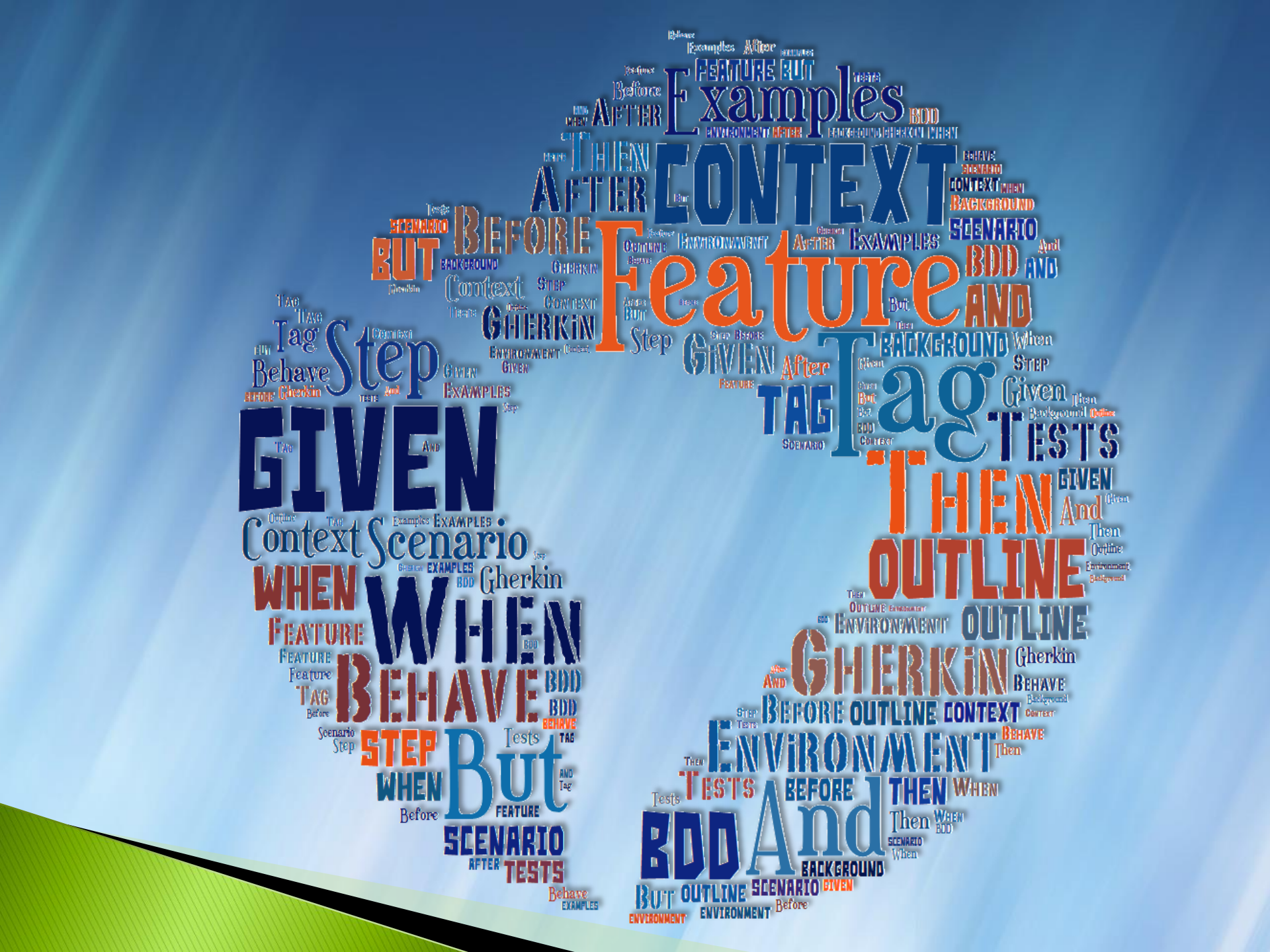
Autor: Reinaldo M. R. Junior


Examples
Feature
Context
Given
When
Then
And
But
Tag
Step
Scenario
Behave
Gherkin
Background
Outline
Environment
Tests
Before
After
BDD
Examples
Feature
Context
Given
When
Then
And
But
Tag
Step
Scenario
Behave
Gherkin
Background
Outline
Environment
Tests
Before
After
BDD

Todo Mundo entende diferente.

Estrutura de pastas:

Behave Estrutura.

features/
features/everything.feature
features/steps/
features/steps/*steps.py*
features/environment.py

Estrutura Extra:

helpers /auxiliar.py
modules/control_api.py
pages/loginPage.py
test_data/config.json

Todos do Time de QA tem que fazer!

Estrutura do nosso Projeto.

1. **Helpers** – Vai conter funções básicas do selenium. Você sempre pode colocar aqui classes genéricas/funções auxiliares.
2. **Modules** – Classes que interage com determinado modulo do teu sistema, como testes de API.
3. **Pages** – Vai Conter todas as classes ou funções relacionadas a Página testada.
4. **Steps** – Vai conter todas os steps definitions do projeto.
5. **Features** – Vai conter todas as .feature do projeto.
6. **Test_Data** – Vai conter todos os arquivos .ini, .json, .txt, .xls, essenciais para os testes.

Gherkin

É uma linguagem comum de fácil entendimento, mas conhecida como "Business Readable, Domain Specific Language", ou seja desenhada para fazer pessoas que não são programadoras ou técnicas, entender o que o pessoal técnico esta fazendo.

Pelo Gherkin descrevemos o comportamento do software, sem detalhar como esse comportamento é implementado.

Gherkin

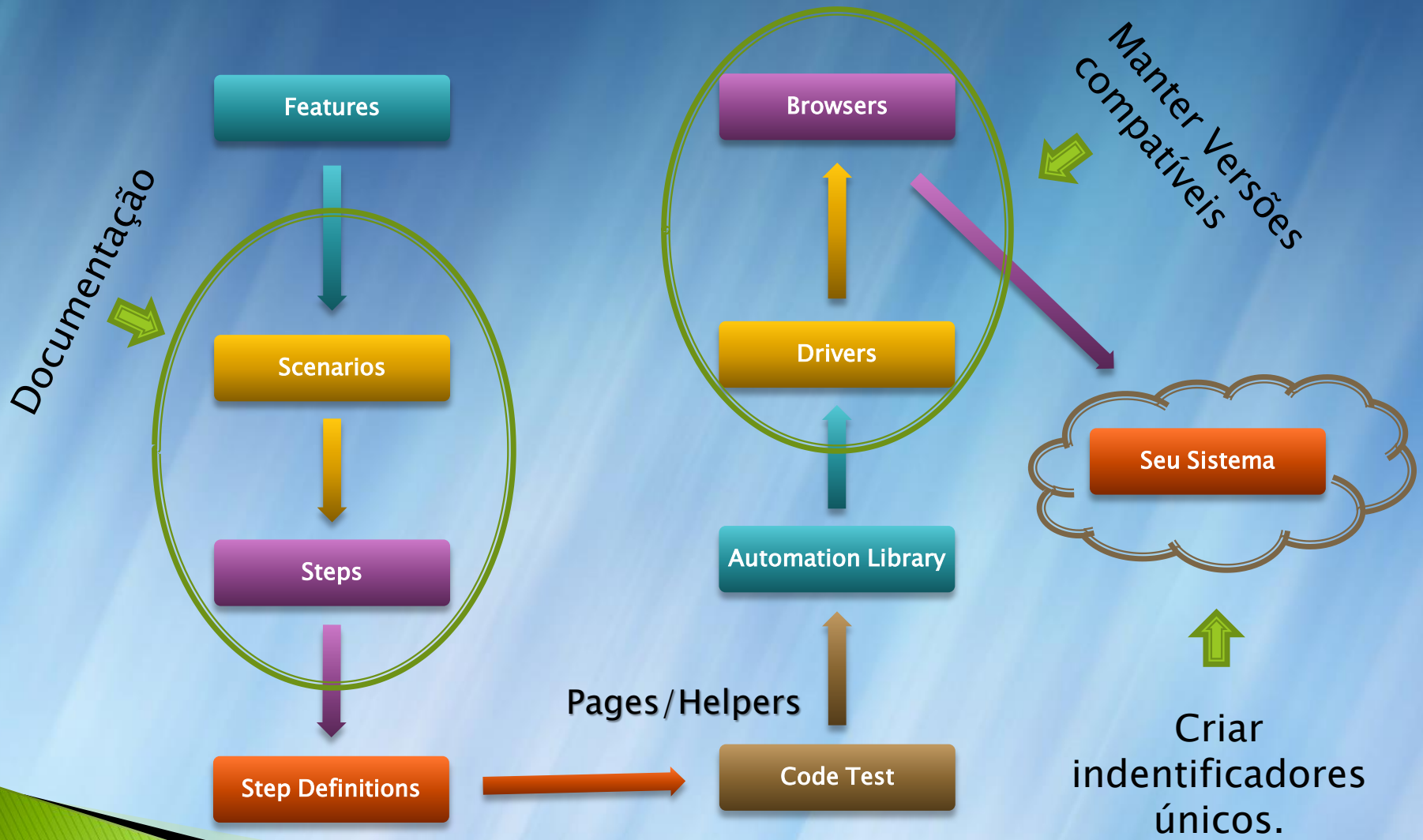
Como Python e YAML, Gherkin é uma linguagem orientada a linha que usa recuo (**Indentação**), para definir a estrutura. Os fins de linha encerrar declarações (por exemplo, passos/steps). Espaços ou tabulações podem ser utilizadas para recuo (mas os espaços são mais portáteis). A maioria das linhas começam com uma palavra-chave (***keyword***).

Keywords / Palavras-Chave

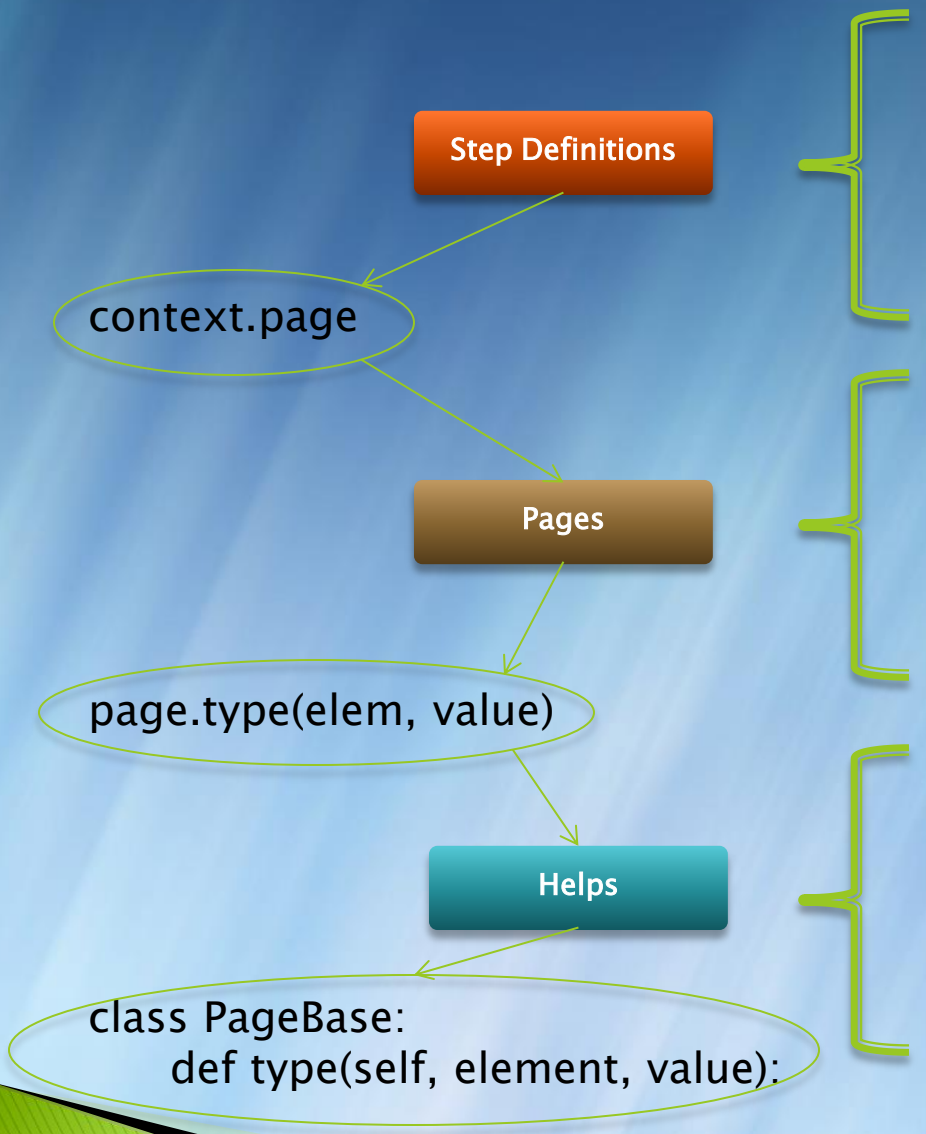
- Feature
- Scenario
- Given, When, Then, And, But (Steps)
- Background
- Scenario Outline
- Examples

- """ (Doc Strings)
- | (Data Tables)
- @ (Tags)
- # (Comments)

Estrutura fluxo
(Responsabilidade do time).



Código bem organizado... A beleza está na simplicidade.

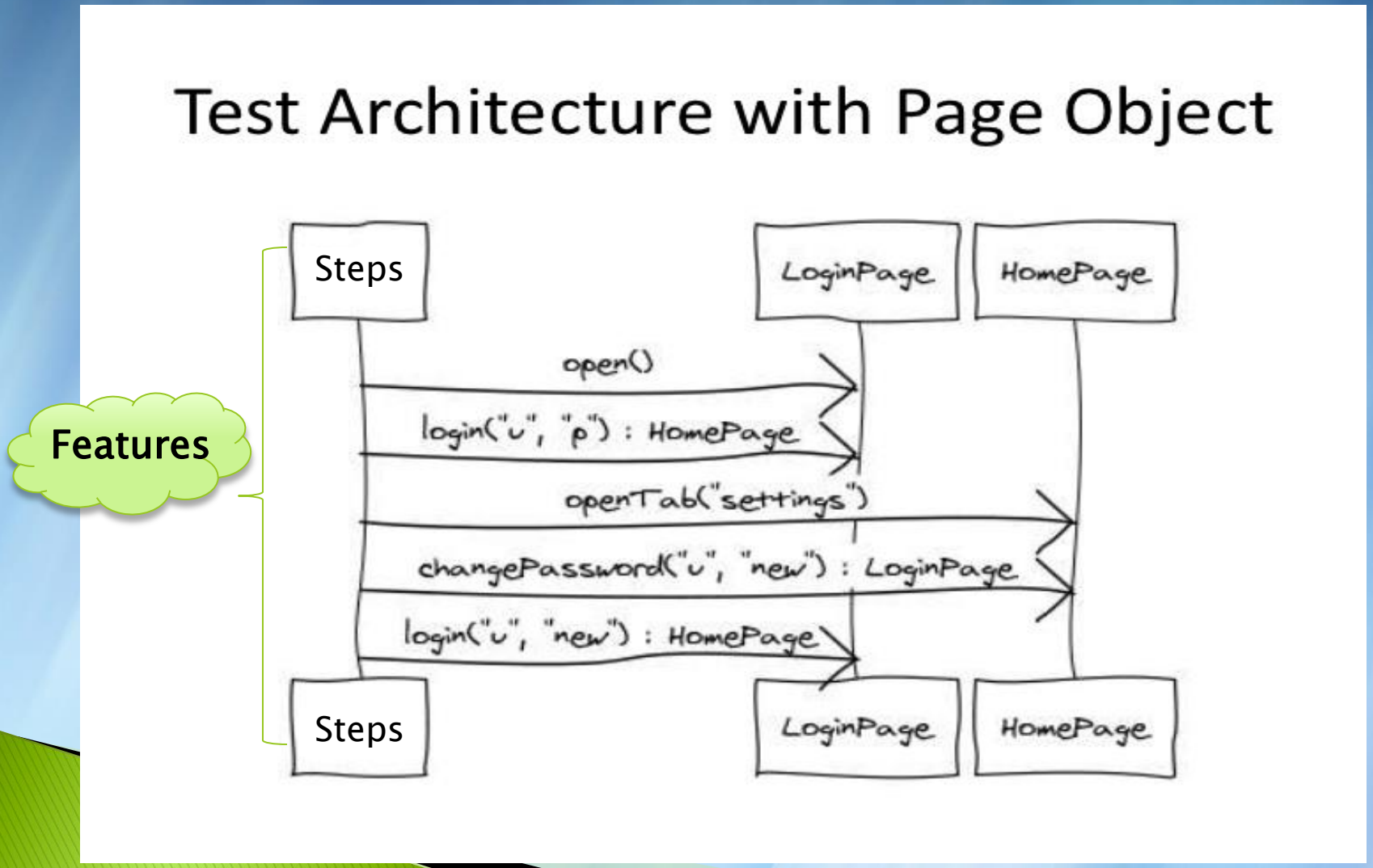


Setar o Driver Remoto;
Instanciar a PageBase passando o driver;
Chamar as Pages(funções).

Chamar as funções da PageBase passando os paramêtros (Responsável pela lógica do teste).

Classes Base responsável por executar as funções.

Arquitetura do Projeto



Features

- Um arquivo .feature deve descrever uma única característica do sistema
- É apenas uma maneira de fornecer uma descrição de alto nível de um recurso de software, e para cenários de grupos relacionados.

Feature: item de Reembolso

"""

Assistentes de vendas devem ser capaz de reembolsar as compras dos clientes.

Isso é exigido por lei, e também é essencial, a fim de manter os clientes satisfeitos.

"""

- Regras:
  - O cliente deve apresentar o comprovante de compra;
  - Aquisição deve ser inferior a 30 dias atrás.

Outro exemplo:

- Feature: Tirar foto e armazenar.
  - Scenario: Tirando foto através da WebCam.
  - Scenario: Pegando uma foto armazenada.
  - Scenario: Comparando antes e depois.

Scenarios

- Um cenário é um conjunto de Passos que descrevi um comportamento esperado.
- Além de ser uma especificação e documentação, o cenário é também um teste.
- Como um todo, os cenários são uma especificação executável do sistema.
- Cenários seguem o mesmo padrão:
  1. É dado um contexto inicial;
  2. É feito uma ou um conjunto de ações;
  3. Descreve um ou vários resultado esperado.

Qual o comportamento esperado?

- **Cenario: Efetuando o Login.**
  - **Dado** que estou na tela de login do sistema
  - **Quando** informo usuário “reinaldo" e senha “X1234"
  - **E** clico no botão "Entrar"
  - **Então** será exibida a página de boas vindas

Steps

- **Given/Dado** – é usado para descrever o contexto inicial do sistema. É tipicamente algo que aconteceu no passado.
- **When/Quando** – é usado descrever um evento ou uma ação. Esta pode ser uma pessoa que interage com o sistema, ou pode ser um evento desencadeada por um outro sistema.
- **Then/Faça** – descrever um resultado ou resultado esperado.
- O **And/E** é usado pra não repetir muito o Given/When/Then, com conotação de continuidade.
- **But/Mas** – é uma exceção, também usado como And.

Steps

Exemplo 01:

Feature: Some terse yet descriptive text of what is desired.

Scenario: Some determinable **business situation**
Given some **precondition**
And some other **precondition**
When some **action** by the actor
And some other **action**
And yet another **action**
Then some testable **outcome** is achieved
And something else we can **check** happens too

Scenario: A different situation.

Login.feature

- **Cenario**: Login Positivo.
  - **Dado** a pagina de login.
  - **Quando** digitar o login e password corretos.
  - **E** efetuar a autenticação.
  - **Faça** o menu com o nome do usuario deve ser exibido.

Exercício

- **Cenario**: Login Negativo.
  - **Dado** a pagina de login.
  - **Quando** digitar o login correto.
  - **E** digitar password incorreto.
  - **E** efetuar a autenticação.
  - **Faça** exibir mensagem de erro de autenticação.

PEP8 + Behave + Python 3

PEP8 – Foco na Legibilidade do Código;
Behave – Foco na Descrição dos Testes de forma que todos possam entender;
Python 3 – Foco na Rapidez da criação dos Testes.

Referências:

- [0] https://github.com/cucumber/cucumber/wiki/Gherkin
- [1] http://martinfowler.com/bliki/BusinessReadableDSL.html
- [2] https://cucumber.io/docs/reference
- [3] http://www.slideshare.net/time2test/gherkin-for-test-automation-in-agile
- [4] https://media.readthedocs.org/pdf/behave/latest/behave.pdf

Autor: Reinaldo M. R. Junior