INTRODUÇÃO AO ROBOT FRAMEWORK PHA

- Marcela Amorim –

Sobre o Robot

O Robot Framework é uma ferramenta de automação de código aberto para testes de aceitação, desenvolvimento orientado a testes de aceitação (ATDD) e automação de processos robóticos (RPA) de sintaxe simples e facilmente estendida com bibliotecas (libs) genéricas e personalizadas.

É independente do sistema operacional e do aplicativo.

Implementado usando Python (que também é a linguagem primária para estendê-lo) contém estrutura com rico ecossistema em torno dela, consistindo em várias bibliotecas e ferramentas genéricas que são desenvolvidas como projetos separados.

Abordagem Keyword Driven:

Nesta abordagem os cenários são escritos usando palavras chaves são em alto nível e em linguagem natural descrevendo as ações do usuário.

Keywords encapsulam a implementação baixo nível do teste.

Boa para testadores e todo o pessoal não técnico.

Facilite escrita e leitura dos testes bem como o aprendizado.

Alto índice de reutilização.

Estrutura Robot Framework:

ESCOPO	DESCRIÇÃO
Settings	Nesta seção podemos informar documentação (Documentation), as bibliotecas (Library), os scripts de baixo nível (Resources), setup/teardown da suíte e dos testes e o timeout para os testes.
Variables	Com o nome já diz, é nessa seção que iremos declarar variáveis e definir os valores default.
Test Case	Nessa seção escrevemos os casos de teste em linguagem natural em Keywords.
Keywords	Onde implementamos os passos (Keywords) escritas na seção Test Case.

Escopo das Variáveis:

ESCOPO	DESCRIÇÃO	EXEMPLO
Global	Disponíveis para uso em todos os pontos do teste	Set Global Variable ${ABC}
Suite	Disponíveis para uso em todos os pontos da suíte	Set Suite Variable ${DEF}

Test	Disponíveis para uso em todos os pontos do cenário de teste	Set Test Variabel ${GHI}
Local	Disponíveis para uso somente na keyword onde está declarada	${xuz} Evaluate 1+1

Bibliotecas Padrão:

- BuiltIn – Contém as principais Keywords do Robot, importada automaticamente
- Collections – Contém as Keywords para manipular listas (arrays) e dicionários (hashes)
- Date Time – Biblioteca manipular datas/horários/cálculos destes elementos
- Dialogs – Módulo que permite pausar a execução e interação com usuários
- Operating System – Realiza tarefas comuns do sistema operacional
- Process – Executa processos do sistema operacional
- Screenshot – Tira prints de tela e gerência os mesmos
- String – Manipulação e verificação de conteúdo de strings
- Telnet – Conexão e envio de comandos a servidores Telnet
- XML – Biblioteca para verificação e manipulação de XMLs
- Selenium – Biblioteca mais comum de interação com navegadores web
- Archive – Permite comprimir e extrair arquivos compactados
- AutoIt – Biblioteca para interagir com softwares desenvolvidos para Windows
- Sikuli – Interação com elementos da interface através de reconhecimento de imagens
- Requests – Realiza requisições em APIs REST
- MongoDB – Interage com banco de dados MongoDB
- Database – Biblioteca relacionada as intenções com bancos de dados SQL
- Debug – Pausa execução e permite interação em tempo real com o código via terminal
- SapGui – Biblioteca específica para automação de SAP
- Suds – Realiza requisições a web services padrão SOAP

Tags:

São etiquetas que registramos nos cenários, a fim de executar testes individualmente ou em grupos dentro de suítes.

Há possibilidade de também excluir cenários de testes da execução usando tags ou ainda combinar duas ou mais para filtrar somente testes desejados através de operadores lógicos.

COMANDO	DESCRIÇÃO	EXEMPLO
AND	Executar cenários com tag1 E tag2 presentes	robot-i tag1ANDtag2 tst.robot
OR	Executar cenários com tag1 OU tag2 presentes	robot-i tag1ORtag2 tst.robot
NOT	Executar cenários com tag1 NÃO presente	robot-i NOTtag2 tst.robot

SETUP e TERADOWN (HOOKS):

ESCOPO	DESCRIÇÃO
Suite Setup	Uma keyword específica será executada ANTES de começar a execução da suíte.
Test Setup	Uma keyword específica será executada ANTES de começar a execução de cada teste.
Suite Teardown	Uma keyword específica será executada DEPOIS de encerrar a execução da suíte.
Test Teardown	Uma keyword específica será executada DEPOIS de encerrar a execução de cada suíte.

Log e Report:

Log: Usado para saber com detalhes como cada passo fo executado nos testes/suíte.

Report: Usado para saber o status/resultados da execução realizada.

Debug

É uma biblioteca usada para pausar a execução do código em pontos onde forem determinados breakpoints.

Serve para ser utilizada para exibir os valores das variáveis e experimentar comando em tempo real, interagindo com as aplicações como se fossem automações.

Para usar é preciso digitar a keyword “debug” na linha onde deseja inserir um breakpoint e executar a automação

FOR – Simples e in range

COMANDO	DESCRIÇÃO
FOR ${ elemento } IN @{LISTA} Log to console ${elemento} END	Executar o log to console N vezes exibindo no console cada elemento da lista, do primeiro ao último.
FOR ${ indice } IN RANGE 10 Log to console ${ indice } END	Executar o log to console 10 vezes exibindo no console os números de 0 a 9.
FOR ${ indice } IN RANGE 1 11 Log to console ${ indice } END	Executar o log to console 10 vezes exibindo no console os números de 1 a 10.

WHILE

O while no robot avalia uma expressão e executa o conteúdo do bloco caso seja verdadeira, até que a condição seja falsa.

O diferencial é o parâmetro opcional “limit”, que permite limitar a execução em X iterações ou até mesmo em X tempo.

```
${x} = Set Variable 1
WHILE ${x} != 0    limit=10
          ${x} = Keyword que retorna 0 ou 1
END
```

CONTROLES DE LOOP: BREAK e CONTINUE

COMANDO	DESCRIÇÃO
FOR ${ valor } IN 1 2 3 4 5 Log to console ${valor} IF{valor} ==3 BREAK END	Executa o loop exibindo os números de 1 a 3 no console, encerrando o loop com break antes de atingir o último elemento.
FOR ${ valor } IN 1 2 3 4 5 IF{valor} ==2 CONTINUE ELSE Log to Console ${valor} END END	Executar o log to console 10 vezes exibindo no console os números de 0 a 9.

KEYWORDS – Automação WEB

KEYWORD	PARÂMETROS	DESCRIÇÃO
Open Browser	url, browser	Abreu uma nova instância do navegador para url informada. O argumento browser especifica qual navegador irá usar.
Click Element	localizador	Clica no elemento identificado pelo localizador
Input Text	Localizador, texto	Insere o texto no campo apontado pelo localizador.
Close Browser	N/A	Fecha o browser atual
Element Text Should Be	Localizador, texto esperado	Verifica se o localizador de elementos contém exatamente o texto esperado
Wait Until Element Is Visible	Localizador, tempo em segundos	Espera até que o elemento do localizador esteja visível. Falha se o timeout expirar antes que o elemento esteja visível.

Mapeamento de Elementos

Os elementos das páginas web possuem diferentes atributos que podem ser utilizados como endereço para as ações elencadas nos scripts de automação.

Existem diversos tipos de seletores aceitos, porém os mais comuns são:

SELETOR	DESCRIÇÃO
ID	O seletor mais estável para referenciar um elemento da página.
Name	Assim como ID, name não tende a ser alterado entre as versões.
Class	Um seletor estável, mas que pode trazer múltiplos resultados. Em caso de múltipla ocorrência, pode se escolher qual ocorrência será utilizada com índices.
Xpath/CSS Selector	Seletores menos estáveis, pois tendem a quebrar com alterações no layout das páginas.