Atelier Robot Framework

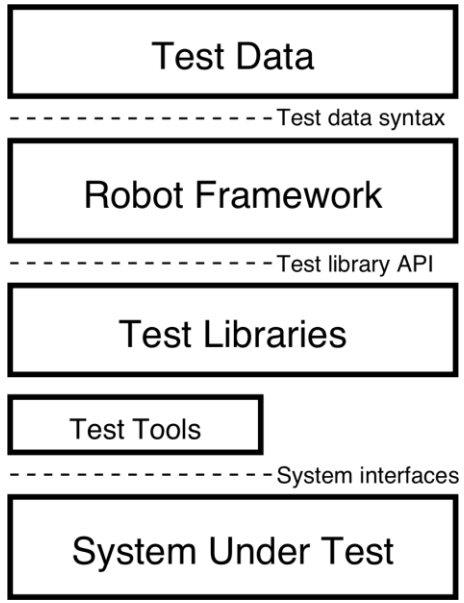
Apprentissage du framework
Installation de poste
Exercice pratique

ACODYME
CONCEPTION WEB & TEST LOGICIEL

Robot Framework - Sommaire

Robot Framework - Introduction

- **Robot Framework** est un framework de test automatique en python pour concevoir et exécuter des tests de bout en bout.
- **Robot Framework** utilise l'approche de test basée sur des mots clés (KDT : Keyword Driven Testing), vulgarisation du code afin de le rendre accessible à tous intervenants d'un projet informatique.
- **Robot Framework** s'appuie sur des bibliothèques de tests (Test Libraries).



Liens utiles

- **User's guide** https://robotframework.org/robotframework
- **Site :** https://robotframework.org/

Robot Framework – Principes KDT (Keyword Driven Test)

- Une ligne par étape de test = une ligne par action.
- Une action se fait sur un élément de la UI (button,link,checkbox).
- Chaque action est identifiée par un mot-clé.

Test manuel standard

#	Action	Résultat	Jeux de données
1	Accéder au site	Le site s'ouvre sur la page d'accueil	www.website.com
2	Ouvrir le menu	Le menu s'affiche	
3	Ouvrir une catégorie	La catégorie s'affiche	Catégorie XXXXX

Keyword Driven Test

#	Action	Element	Jeux de données
1	Go To Url	N/A	www.website.com
2	Click	Bouton menu	
3	Check Menu Open	N/A	
4	Click	Liens catégorie	Catégorie XXXXX
5	Check Catégorie Open	N/A	

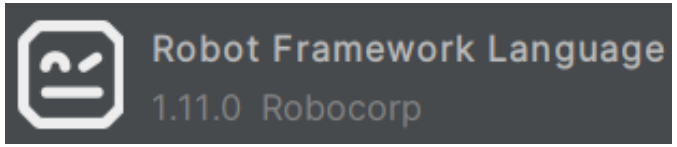
Robot Framework – Outil de développement

Pycharm Community / Aqua Beta / VS Code

- IDE pour le développement en python
- Permet l'éxecution de script python directement dans l'IDE
- Compatible avec GIT
- Support de la syntaxe .robotframework via le plugin RWS
- Gratuit

https://www.jetbrains.com/fr-fr/pycharm/download/
https://www.jetbrains.com/fr-fr/aqua/
https://code.visualstudio.com/Download

+



ACODYME CONCEPTION WEB & TEST LOGICIEL

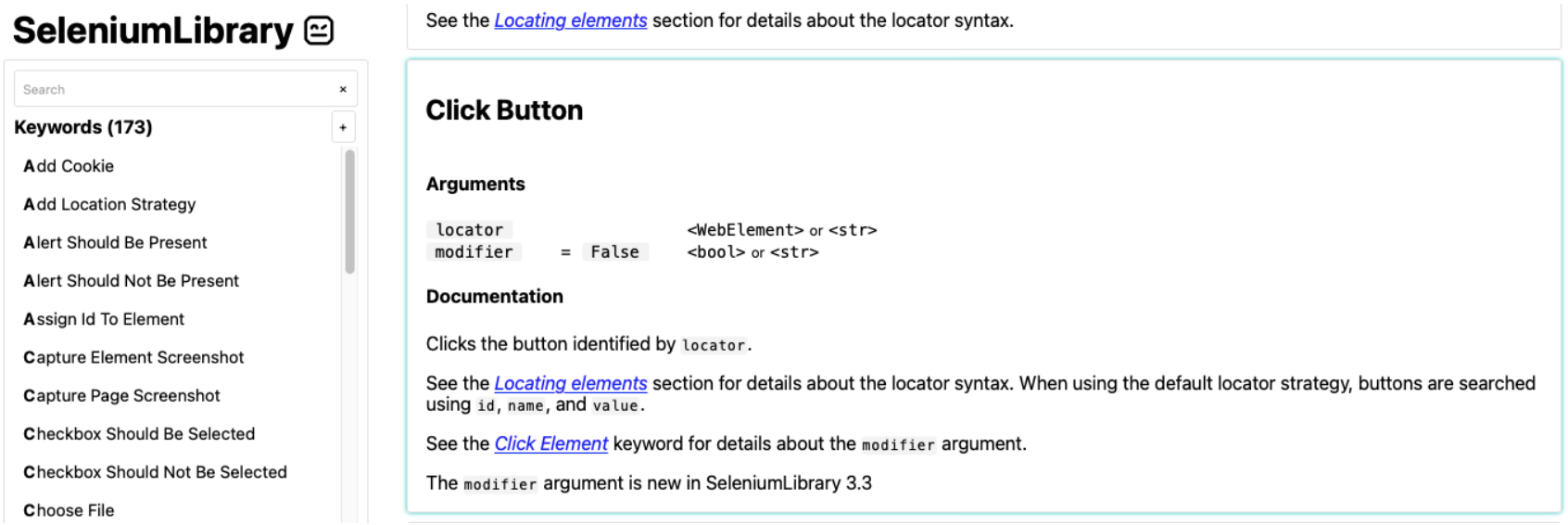
Robot Framework – Librairies les plus utilisés

Liste des <**Libraries**> les plus utilisées :

- **SeleniumLibrary :** Keywords pour test d'applications web, basé sur Selenium.
- **Browser :** Keywords pour test d'applications web, basé sur Playwright.
- **String :** Keywords d'action sur les chaines de caractères.
- **Collections :** Keywords d'action sur les listes/dictionnaires.
- **BuiltIn (native RBF) :** Keywords génériques souvent nécessaires.

Chaque <**Library**> possède une de documentation accessible sur internet

Exemple de documentation :



ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Structure et syntaxe

Structure :

- Les fichiers .robot sont composé de plusieurs sections ***Settings***, ***Variables***, ***Keywords***, ***Test Cases***.
- Les **sections** sont identifiées par trois astérisques (***).

Syntaxe :

- Les noms des **Test-case** ou **Keyword** peuvent être n'importe quelle séquence de mots séparés par un seul espace.
- Chaque ligne sous le nom du **Test-case/Keyword** doit être indentée pour être considérée comme une instruction.
- Il faut impérativement **deux espaces ou plus** entre un **Keyword** et son/ses argument(s).

```
*** Settings ***
Library           BuiltIn
Documentation     Simple test case

*** Variables ***
${HELLO}=   Hello world!
${GOODBY}=  Goodbye!

*** Test Cases ***
Log Hello World
    [Documentation]      Log hello world
    Log Something   ${HELLO}

Log Goodbye
    [Documentation]     Log goodbye
    Log Something   ${GOODBY}

*** Keywords ***
Log Something
    [Arguments]     ${string}
    Log To Console  ${string}
```

Robot Framework - Best practices

1. **Décider quoi tester avant de décider comment le tester** : identifier chaque fonction à tester.
2. **Rester simple** : le cas de test doit vérifier une seule fonction et ne doit échouer que pour une seule raison.
3. **Standard de nommage** : les noms des éléments d'interface utilisateur et des objets de test doivent être explicites.
4. **Ne copiez-collez pas de code de test** : au lieu de répéter les mêmes étapes dans plusieurs tests, créer des modules (Keywords) réutilisables.
5. **Utiliser une structure modulaire** : le scénario de test doit pouvoir fonctionner indépendamment des autres scénarios de test.
6. **Utiliser des sélecteurs fiables** : assurez-vous que vos tests automatisés peuvent trouver vos objets d'interface utilisateur de manière fiable.
7. **Data-Driven Testing (DDT)** : le scénario de test récupère les valeurs d'entrées d'une source de données, le scénario de test est répété automatiquement pour chaque ligne de données.
8. **Limiter le nombre d’étapes** : 10 étapes maximum
9. **(Bonus) Documenter les cas de tests** : indiquer les étapes à exécuter et les résultats attendus

Robot Framework – Norme de nommage

Norme de nommage :

- Privilégier des noms clairs mais pas trop long.
- Déclaration d'une variable dans robotframework -> ${variable}
- L'underscore (_) peut être utilisé comme séparateurs de mots (ex : ${ma_variable}
- Utilisation du snakecase (lower_case_with_underscores)
- Variable en minuscule pour les variables locales (niveau **Keyword**).
- Variable en majuscule (camelcase) pour les variables globales (niveau **Test-Cases**).
- Pour les sélecteurs : attribut_type_nom -> ${class_button_login}

Good:

```
*** Test Cases ***
Withdraw From Account
    Withdraw From Account    $50
    Withdraw Should Have Succeeded

*** Keywords ***
Withdraw From Account
    [Arguments]    ${amount}
    ${STATUS} =    Withdraw From User Account    ${USER}    ${amount}
    Set Test Variable    ${STATUS}

Withdraw Should Have Succeeded
    Should Be Equal    ${STATUS}    SUCCESS
```

Not so good:

```
*** Test Cases ***
Withdraw From Account
    ${status} =    Withdraw From Account    $50
    Withdraw Should Have Succeeded    ${status}

*** Keywords ***
Withdraw From Account
    [Arguments]    ${amount}
    ${status} =    Withdraw From User Account    ${USER}    ${amount}
    [Return]    ${status}

Withdraw Should Have Succeeded
    [Arguments]    ${status}
    Should Be Equal    ${status}    SUCCESS
```

Good:

```
*** Keywords ***
Login With Valid Credentials
```

Bad:

```
*** Keywords ***
Input Valid Username And Valid Password And Click Login Button
```


ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Keywords fréquemment utilisés

Liste des Keywords les plus couramment utilisés :

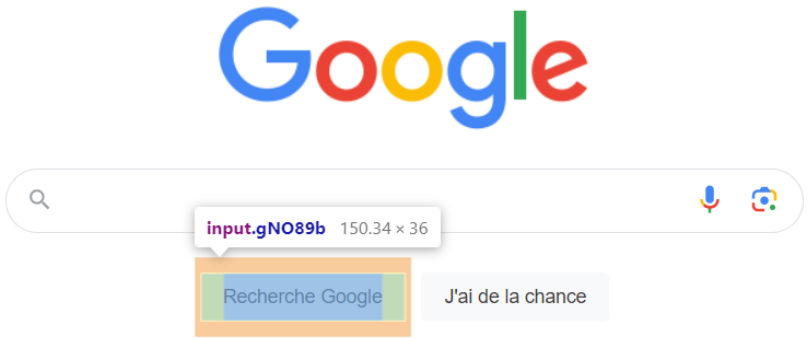
Keyword	Argument	Description
Click Element	locator	Click element identified by locator.
Input Text	locator,text	Types the given text into text field identified by locator.
Input Password	locator,text	Types the given password into text field identified by locator.
Click Button	locator	Click button identified by locator.
Wait Until Element Is Enabled	locator, timeout, error	Waits until element locator is enabled.
Wait Until Element Is Visible	locator, timeout, error	Waits until element locator is visible.
Get Text	locator	Returns the text value of element identified by locator.
Element Should Contain	locator,expected, message	Verifies that element identified with locator contain expected value
Element Should Be Visible	locator,message	Verifies that element identified with locator is visible

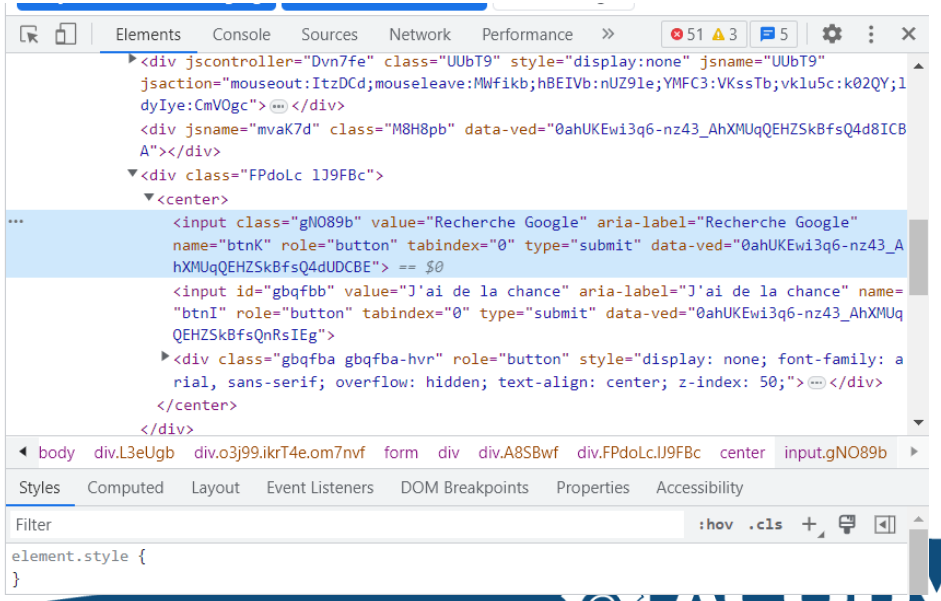
Robot Framework – Les selectors (UI)

Comment choisir un selecteur (locator):

- Un selecteur (*locator*) est la requête qui fait référence à un élément HTML sur la page web.
- Utilisation des selecteurs basés sur **Xpath** ou **CSS Locator.**
- **XPath** permet d'extraire des informations (éléments, attributs, commentaires, etc...) d'un document XML via l'écriture d'expressions.
- **CSS Selector** permet d'extraire des informations du DOM HTML en se basant sur le style CSS.
- Si possible utilisé l'attribut ID, sinon utiliser un attribut d'accessibilité dans le **XPath** pour le selector (ex *aria*)
- Exemple : ici, on se réfère à un élément <**button**> dont l'attribut HTML *value* contient *"Recherche Google".*

  Xpath simple : *//button[@value="Recherche Google"]*
  Xpath avec contraintes :*//button[contains(@value,"Recherche Google") and @tabindex="0"]*
  CSS Selector :*button[value="Recherche Google"]*

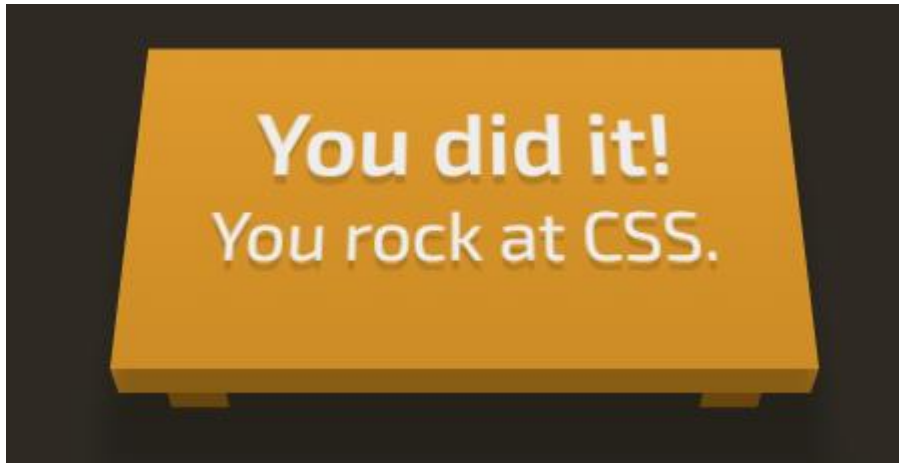




Robot Framework – Les selectors (UI)

Exercez vous sur : https://flukeout.github.io/

Les 32 exercices peuvent être fait en 30 min



Robot Framework – Les expressions Xpath

Exemple d'expression Xpath:

`#id`	`//*[@id="id"]`
`.class`	`//*[@class="class"]` …kinda
`input[type="submit"]`	`//input[@type="submit"]`
`a#abc[for="xyz"]`	`//a[@id="abc"][@for="xyz"]`
`a[rel]`	`//a[@rel]`
`a[href^='/']`	`//a[starts-with(@href, '/')]`
`a[href$='pdf']`	`//a[ends-with(@href, '.pdf')]`
`a[href*='://']`	`//a[contains(@href, '://')]`
`a[rel~='help']`	`//a[contains(@rel, 'help')]` …kinda

`h1:not([id])`	`//h1[not(@id)]`
Text match	`//button[text()="Submit"]`
Text match (substring)	`//button[contains(text(),"Go")]`
Arithmetic	`//product[@price > 2.50]`
Has children	`//ul[*]`
Has children (specific)	`//ul[li]`
Or logic	`//a[@name or @href]`
Union (joins results)	`//a | //div`

Afin de réduire Le coût de maintenance et de mise en place des tests-auto sur la UI :

- Il faut recourir à l'utilisation de « test-id » partagé avec le développeur, lors de développement ou rework de feature

Liens utiles

- **Site :** https://devhints.io/xpath

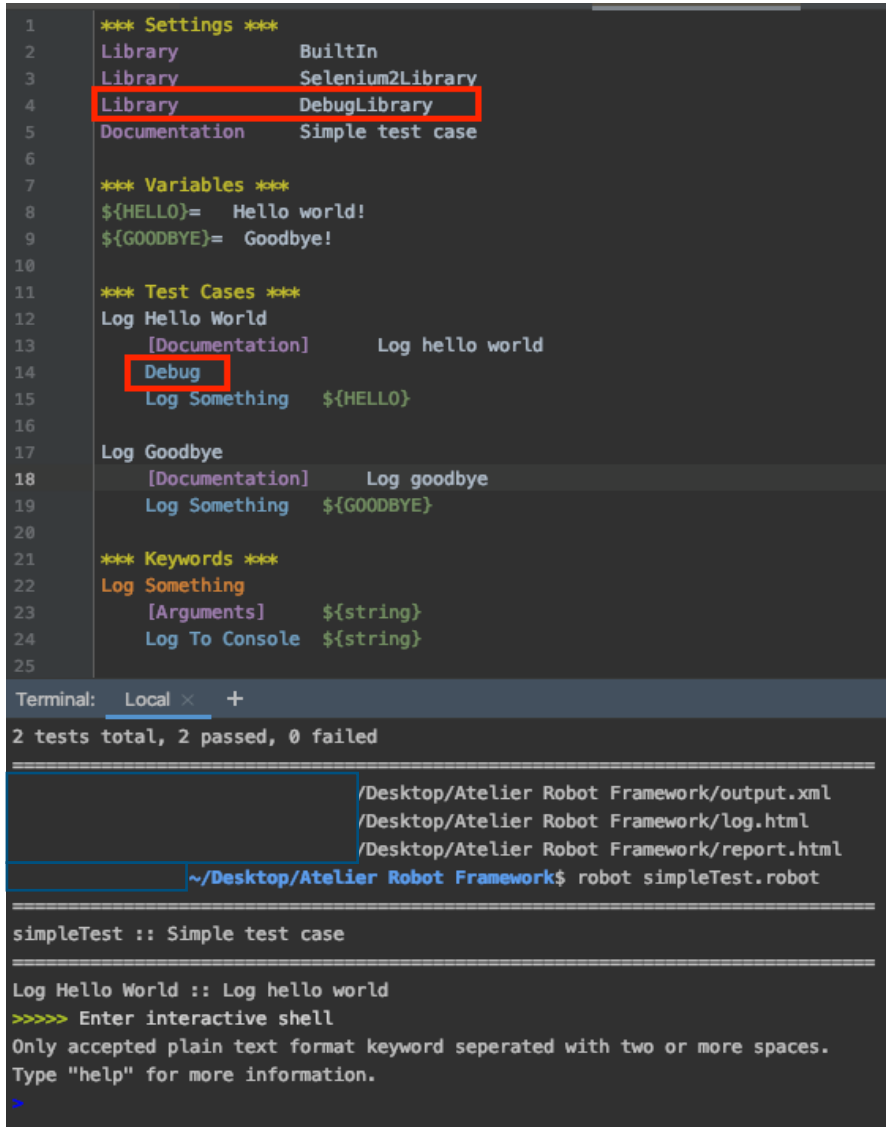
ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Debugger vos scripts

Utilisation de DebugLibrary

Permet de faire un point d'arrêt dans le script afin de le dérouler manuellement et d'identifier la portion de code en erreur ou un sélecteur défaillant.

1. Installer la library
   `pip3 install DebugLibrary`
2. Insérer le mot clé « Debug » à l'étape voulue
3. Exécuter les mots clefs manuellement



ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Gherkin syntaxe

Qu’est ce Gherkin ?

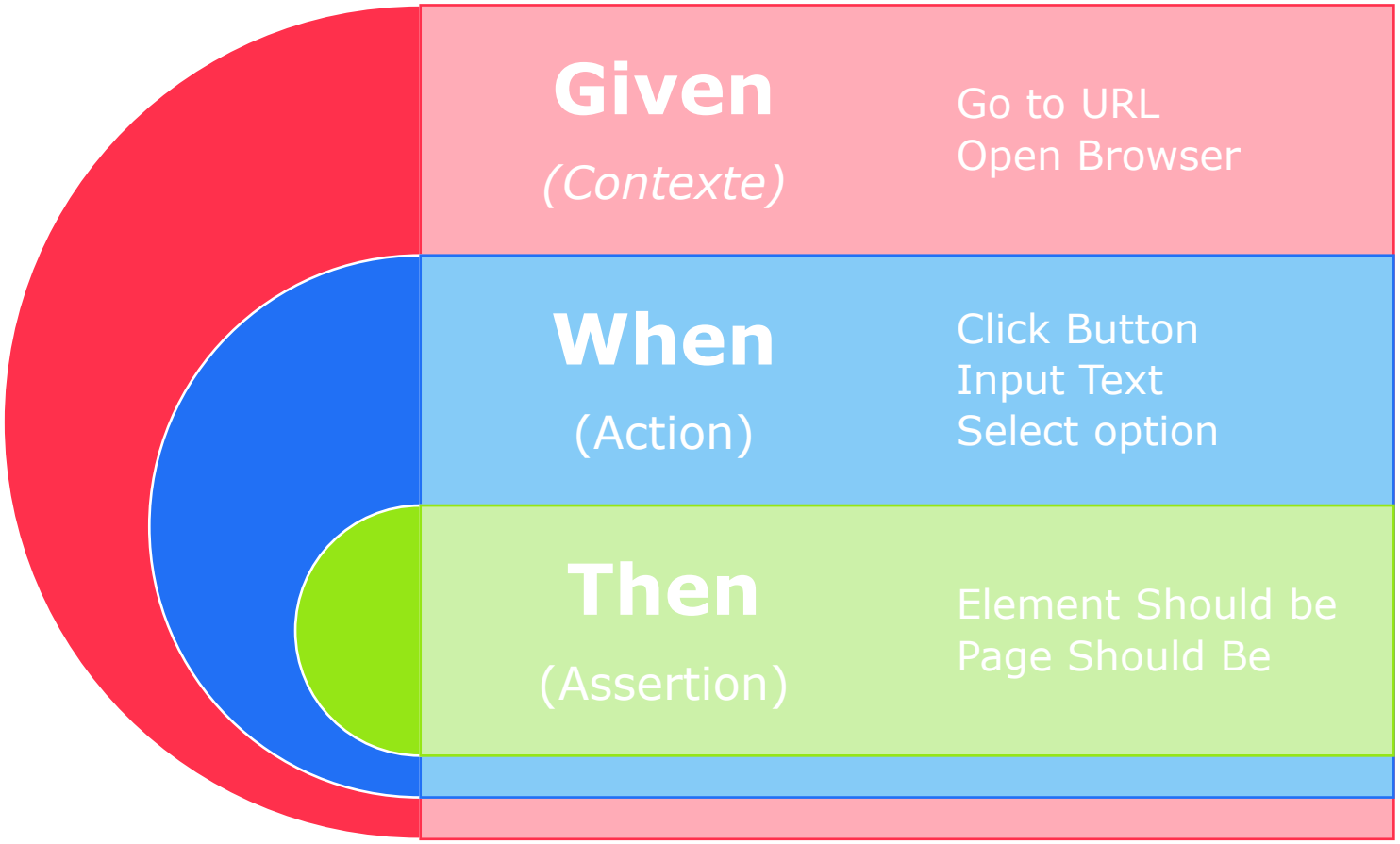
Gherkin est un langage *Spécifique à un Domaine fonctionnel*, lisible par tous, basé sur des mots clés pour décrire un comportement.

Un scénario Gherkin comporte un titre et des instructions :

Given, When, Then

Robot Framework – Type de Keywords par étape

Keyword type par étapes Ghekins:



ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Installation du poste de travail

Cloner le projet Robotframework débutant (repo BeginnersGuide)

```
git clone https://github.com/Acodyme/BeginnersGuide.git
```

Installer Python3, PIP, les librairies

Installation ubuntu / mac OS :

https://github.com/Acodyme/BeginnersGuide#installation

Installation windows :

https://github.com/Acodyme/BeginnersGuide/blob/master/Installation%20de%20Robotframework.pdf

Robot Framework – Installation du poste de travail

Exécuter les tests avec robot (un par un)

```
make test TEST=selenium_00
```

```
robot -v HEADLESS:false -i  selenium_00 -d /Results /Tests
```

Exécuter les tests avec pabot (parallélisation des test)

```
make ptest TEST=selenium_00ORselenium01ORselenium02
```

```
pabot --processes 8 --pabotlib -x result.xml -v HEADLESS:false -i
selenium_00ORselenium01ORselenium02 -d /Results /Tests
```

Mise en pratique

Robot Framework – Exercices

Tests d’exemple déjà codé

- Rappel_RBF.robot
- TS_RBF.robot

Tests que vous devez codé

- TS_TODO.robot
- TS_EXO_SELENIUM-EASY.robot
- TS_PLAYWRIGHT.robot

Robot Framework – Exercice 1

Cas fonctionnel à implémenter

S’appuyer de cette librairie : https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html

Développer en le test-case <**TS_TODO.robot**> en s’appuyant sur le Page Object Model, ayant pour étapes les actions suivantes :

1. aller sur le site **https://todomvc.com/examples/angular/dist/browser/#/all**
2. ajouter 3 todo
3. traiter une tâche
4. supprimer une tâche

Robot Framework – Exercice 2

Cas fonctionnel à implémenter

S’appuyer de cette librairie : https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html

Développer les test-cases <**TS_EXO_SELENIUM_EASY.robot**>, de **selenium_01 à selenium_10.**

Les énoncés sont renseignés dans les test-cases avec les blocs **[Documentation]**

Robot Framework – Exercice 3 (bonus)

Cas fonctionnel à implémenter

S’appuyer de cette librairie : https://marketsquare.github.io/robotframework-browser/Browser.html

Développer en .robot le test-case <**My First Playwright Robot Test**>, prenant en paramètre les variables **${LOGIN} standard_user, ${PWD} secret_sauce** ayant pour étapes les actions suivantes :

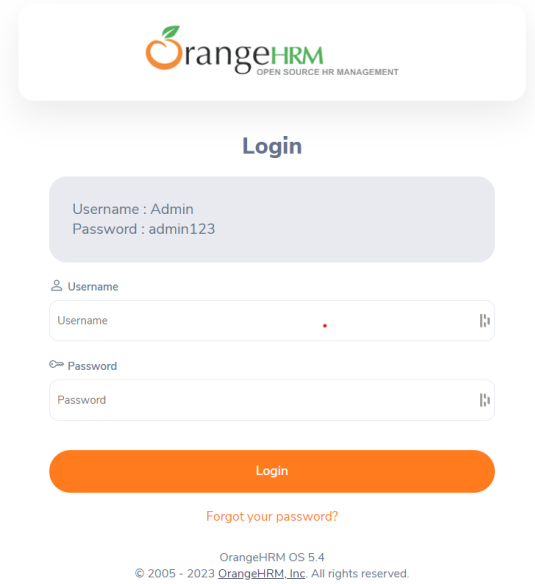
1. aller sur le site **https://www.saucedemo.com/**
2. se connecter au compte **standard_user**
3. vider le panier (avec le bouton « Remove », s’il est visibile sur la grille produit)
4. Ajouter la produit **« Sauce Labs Backpack »** au panier
5. Aller dans le panier
6. Vérifier que le produit **« Sauce Labs Backpack »** est présent dans le panier
7. Continuer le processus de commande
8. Payer la commande
9. Vérifier que la commande est validée grâce au message **‘Thank you for your order’**

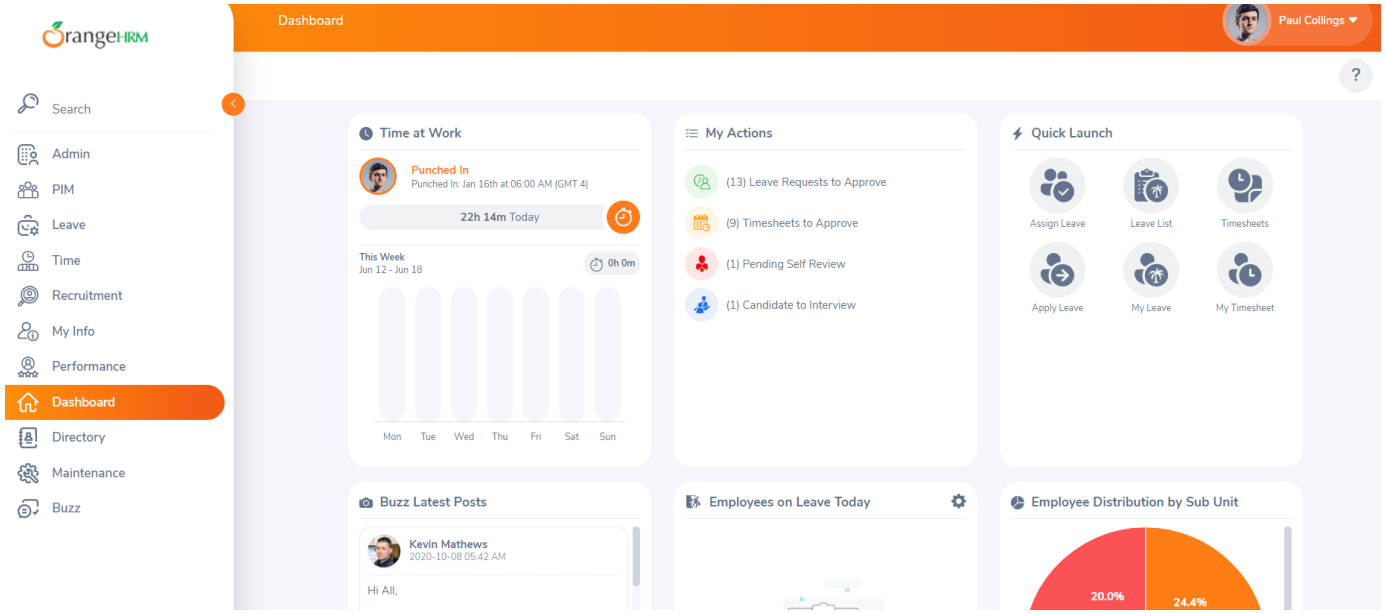
Travaux pratique
(groupe de 2 personnes)

Robot Framework – Travaux pratiques

Présentation du produit à tester

Site web de gestion des Ressources Humaines
https://opensource-demo.orangehrmlive.com/web/index.php/auth/login





- Gestion des utilisateurs
- Gestion des employés
- Gestion des congés
- Gestion du consommé
- Gestion des candidatures
- Gestion du compte connecté
- Gestion de la performance

ACODYME CONCEPTION WEB & TEST LOGICIEL

Robot Framework – Travaux pratiques

Présentation du besoin

Le client X a besoin d'améliorer sa stratégie de test, elle comprend uniquement des cas de test joué manuellement par un QA. Celui-ci souhaite automatiser 2 tests de non régression avec la SeleniumLibrary ou BrowserLibrary de Robot Framework mais il n'a pas les compétences requises. Sauriez-vous l'aider ?

En tant qu'automaticien, vous devez :

1. Couvrir la gestion des utilisateurs systèmes (ajout, recherche, modification, suppression)
2. Couvrir la gestion des employées (ajout, recherche, modification, suppression)

Robot Framework – Travaux pratiques

Critère de notation (sur 20 points)

- **10 points** : si les deux tests e2e ci-dessous fonctionnent correctement et respectent les attendues
- **4 points** : utilisation et respect du language naturel Gherkin (méthodologie BDD)
- **3 points** : écriture des keywords avec gestion de valeur|paramètre dynamique pour privilégier la réutilisation des keywords
- **3 points** : qualité et optimisation du code

Robot Framework – Travaux pratiques

Instruction de restitution des travaux pratiques

- Travaux par groupe de 2 personnes
- A la racine du micro-projet ajouté un fichier **CONTRIBUTORS.md (markdown)** dans lequel vous précisez le nom des personnes qui ont contribués au micro-projet (à noter)
- Compressez votre micro-projet au format .zip|.tar|.tar.gz
- Envoyer votre archive à **lucas.vannoort.ext@eduservices.org** avec comme objet **« TP – Robot Framework MDS B3 »** et dans le corps du message bien préciser les noms et prénoms des deux personnes qui composent le groupe

Robot Framework – Travaux pratiques

Cas fonctionnel à implémenter - Gestion des utilisateurs

Développer en .robot le test-case <**User Managment (Selenium|Playwright) Robot Test**> en langage naturel Gherkin (les keywords alimenteront le test-case) et utiliser les variables **${USER_ROLE} ESS, ${STATUS} Enabled, ${EMPLOYEE_NAME} (type any caracter and select the first result), ${USERNAME} Robotframework, ${PASSWORD} secure password** ayant pour étapes les actions suivantes :

1. aller sur le site https://opensource-demo.orangehrmlive.com/web/index.php/auth/login connecté avec le compte générique (Username : Admin, Password : admin123)
2. cliquer dans le menu sur la page « Admin »
3. Ajouter un utilisateur
4. Recherche l'utilisateur précédemment créé
5. Modifier les informations de l'utilisateur recherché
6. Supprimer l'utilisateur modifié
7. Se déconnecter
8. Vérifier que nous sommes revenus sur la page de connexion

Robot Framework – Travaux pratiques

Cas fonctionnel à implémenter - Gestion des employés

Développer en .robot le test-case <**Employes Managment (Selenium|Playwright) Robot Test**> en langage naturel Gherkin (les keywords alimenteront le test-case) et utiliser les variables **${FIRSTNAME} random firstname, ${LASTNAME} random lastname, ${USERNAME} random username, ${PASSWORD} secure password** ayant pour étapes les actions suivantes :

1. aller sur le site https://opensource-demo.orangehrmlive.com/web/index.php/auth/login connecté avec le compte générique (Username : Admin, Password : admin123)
2. cliquer dans le menu sur la page « PIM »
3. Ajouter un utilisateur avec création de login (checkbox « Create Login Details »)
4. Recherche l'utilisateur précédemment créé
5. Modifier les informations de l'utilisateur recherché
6. Supprimer l'utilisateur modifié
7. Se déconnecter
8. Vérifier que nous sommes revenus sur la page de connexion

Merci de votre Attention !

Avez-vous des questions ?

Get in touch