

# Instruções para Execução de Testes com Robot Framework + Selenium

Para automatizar e executar testes Roboframework + Selenium basta seguir os próximos passos:

## 1 - Descarregar e instalar Python 2.7.18 (last version of 2.7)

<https://www.python.org/downloads/release/python-2718/>

(no meu caso: *python-2.7.18.amd64.msi*)

(para o correto funcionamento, por favor garantir que o pyhon foi devidamente adicionado às variáveis de ambiente do sistema operativo (automática ou manualmente))

## 2 - Instalar Robotframework and Selenium Libraries for Python 2.7:

Na consola executar os seguintes comandos:

```
pip install robotframework
```

```
pip install robotframework-seleniumlibrary
```

(desta forma o pip irá obter automaticamente a última versão disponível e as suas dependências.

Ligação à internet é necessária, caso contrário as *libraries* deverão ser descarregadas e instaladas uma a uma *offline*.)

## 3 – Desenvolvimento e execução de testes

### 3.1 - Para desenvolvimento e execução pode ser utilizado qualquer editor de texto (ex.: Notepad) e executar via CMD:

```
robot -d Results --test "Validate Anti Robot Mechanism is Active" C:\GOContact_Challenge\Tests
```

**robot ou robot.exe ou robot.bat** > executável Robot Framework para executar testes.

**-d Results** > directoria onde os resultados serão guardados (log.html, report.html, screenshots, etc).  
(Esta directoria será criada automaticamente se não existir.

Se esta *flag* não for passada, o teste irá guardar os resultados na pasta de onde o utilizador está a executar o comando (não recomendado)).

**--test "Test Name"** > flag para especificar qual teste a executar. Caso não seja passado o teste a executar, serão executados todos os testes da Suite existente na directoria passada.

**C:\RFTestes\GitHubTests\Tests** > Directoria onde a Robot Framework pode encontrar o(s) teste(s) (--test) a executar.

Exemplo para executar todos os 10 testes da suite de uma só vez:

```
C:\GOContact_Challenge> robot -d Results C:\GOContact_Challenge\Tests
```

**Exemplo de report de execução:**

## Test Statistics

**REPORT**

Total Statistics	Total	Pass	Fail	Elapsed	Pass / Fail
Critical Tests	10	10	0	00:04:42	<div></div>
All Tests	10	10	0	00:04:42	<div></div>
Statistics by Tag	Total	Pass	Fail	Elapsed	Pass / Fail
No Tags					<div></div>
Statistics by Suite	Total	Pass	Fail	Elapsed	Pass / Fail
Tests	10	10	0	00:04:42	<div></div>
Tests.GitHub Tests	10	10	0	00:04:42	<div></div>

## Test Execution Log

SUITE

Tests

Full Name: Tests

Source: C:\Users\P057997\PycharmProjects\GOContact\_Challenge\Tests

Start / End / Elapsed: 20210719 13:18:31.989 / 20210719 13:23:13.906 / 00:04:41.917

Status: 10 critical test, 10 passed, 0 failed  
10 test total, 10 passed, 0 failed

00:04:41.917

SUITE

GitHub Tests

Full Name: Tests.GitHub Tests

Source: C:\Users\P057997\PycharmProjects\GOContact\_Challenge\Tests\Git\_Hub\_Tests.robot

Start / End / Elapsed: 20210719 13:18:32.021 / 20210719 13:23:13.892 / 00:04:41.871

Status: 10 critical test, 10 passed, 0 failed  
10 test total, 10 passed, 0 failed

00:04:41.871

TEST

Registration - Check Password 3-level Strength Mechanism

00:00:33.943

TEST

Registration - Validate Anti Robot Mechanism is Active

00:00:35.126

TEST

Repo - Create New Empty Repository

00:00:24.284

TEST

Repo - Create New File and Validate it

00:00:27.156

TEST

Repo - Change Repo Access from Private to Public and Check that anyone can access it

00:00:30.828

TEST

Repo - Search and Remove a Repository with Success

00:00:24.232

TEST

Project - Create a New Project

00:00:26.777

TEST

Project - Edit Project Information

00:00:27.024

TEST

Project - Search and Remove an Existing Project

00:00:25.573

TEST

Profile - Edit User Personal Information and Validate it

00:00:26.676

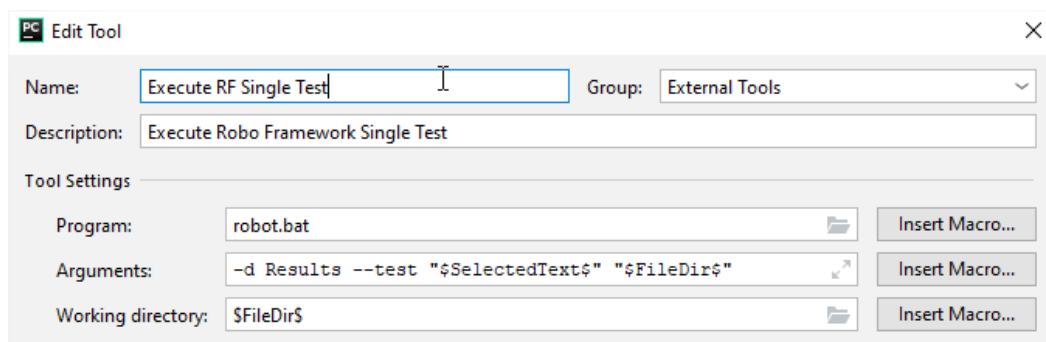
## 3.2 - Em alternativa pode ser utilizado o IDE que eu utilizei para desenvolvimento e execução: PyCharm IDE (*community edition*):

Descarregar e instalar:

<https://www.jetbrains.com/pycharm/download/#section=windows>

Configurar:

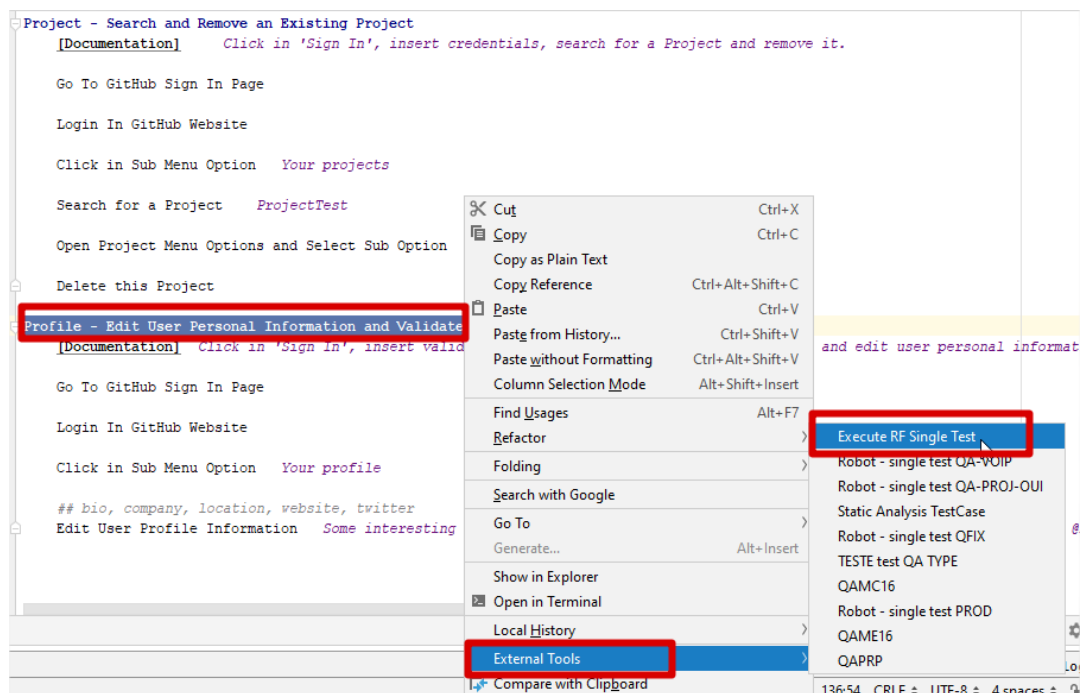
Encontrar e adicionar o interpretador Python nas *Project Settings* > Aplicar ao Projecto > Configurar uma *External tool* (File > Settings) para execução de testes Robot Framework e executar diretamente no IDE:



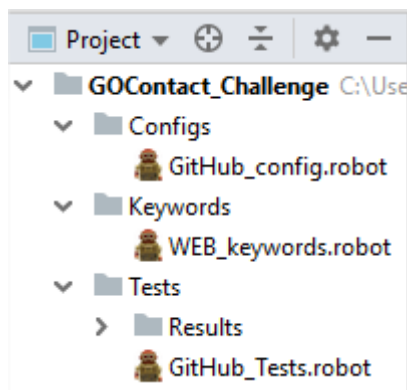
Desta forma já conseguimos executar testes diretamente no IDE (imagem seguinte):

Selecionar o nome do *teste case* com o cursor > clicar botão direito para External Tool > Escolher a external tool criada.

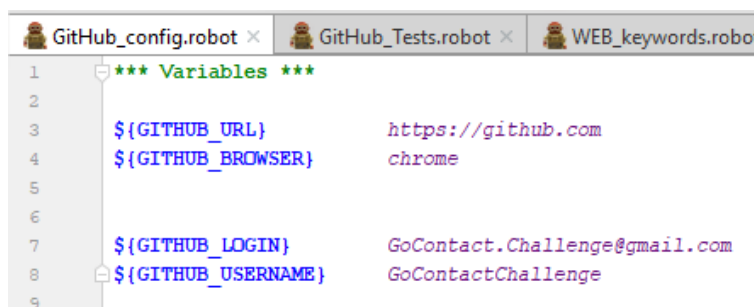
Uma external tool adicional semelhante, em que não tenha a **flag --test** pode ser criada para executar os testes todos da suite de uma só vez sem ter que selecionar testes.



#### 4 – Estrutura de Projeto



**GitHub\_config.robot** – Ficheiro de configuração para guardar informação geral da aplicação (ex.: URL, portos, users, emails, etc).

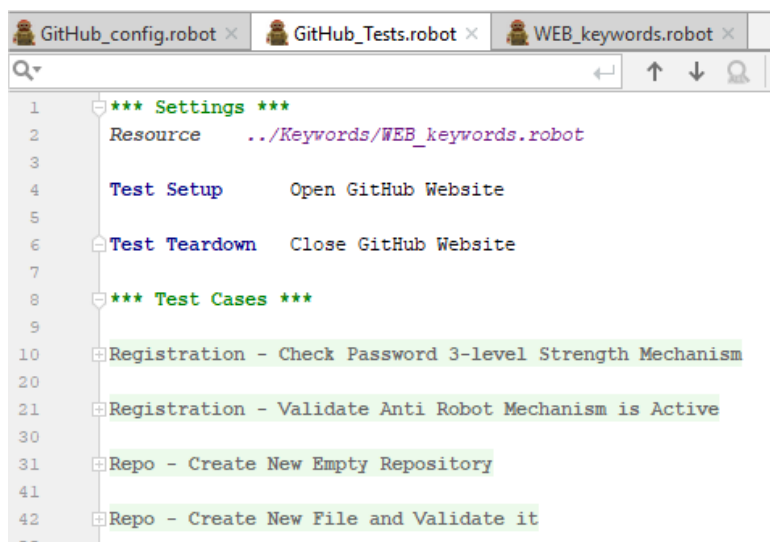


**WEB\_keywords.robot** – Ficheiro com o desenvolvimento das keywords que irão ser depois reutilizadas nos testes ou em outras keywords.  
Pode conter também o *import* das *libraries* necessárias ao teste ou de outros ficheiros (Resources) como por ex. o **GitHub\_config.robot**.



```
1  *** Settings ***
2  Documentation      Resource containing generic keywords to handle git hub web page.
3
4  Library            SeleniumLibrary
5  Library            DateTime
6
7  # Importing common configuration variables (ex. app url, browser default, delays, etc)
8  Resource            ../Configs/GitHub_config.robot
9
10 *** Keywords ***
11 Take App Evidence
12
13
14
15
16
17
18
19 Open GitHub Website
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35 Close GitHub Website
36
37
38
39
40
41
42
43 Go to GitHub Sign In Page
44
45
46
47
48
49
50
51
52 Login in GitHub Website
```

**GitHub\_Tests.robot** – Ficheiro (Suite) que contém os 10 testes implementados e que importa e utiliza as *keywords* implementadas no ficheiro anterior.



```
1  *** Settings ***
2  Resource            ../Keywords/WEB_keywords.robot
3
4  Test Setup          Open GitHub Website
5
6  Test Teardown       Close GitHub Website
7
8  *** Test Cases ***
9
10 Registration - Check Password 3-level Strength Mechanism
11
12
13
14
15
16
17
18
19
20
21 Registration - Validate Anti Robot Mechanism is Active
22
23
24
25
26
27
28
29
30
31 Repo - Create New Empty Repository
32
33
34
35
36
37
38
39
40
41
42 Repo - Create New File and Validate it
```

## 5 – Escolha dos Testes

Depois de interagir primeiramente com a aplicação em modo exploratório, foram identificados os seguintes testes, que na minha opinião me pareceram ser bastante relevantes devido à criticidade de algumas funcionalidades.

Relativamente à segurança do website, foram identificados 2 testes para garantir que o utilizador não consegue registar-se com uma password fraca, testando assim os 3 níveis de força da password inserida.

Foi também identificado um teste para validar que o mecanismo anti-robot está efetivamente em funcionamento tendo inclusive o teste identificado alguns problemas de performance relativamente

a este mecanismo. Nem sempre o mecanismo carrega devidamente obrigando o utilizador a ter que reinserir a password e a ter que resolver novamente o puzzle.

**Registration - Check Password 3-level Strength Mechanism**  
**Registration - Validate Anti Robot Mechanism is Active**

Depois foram automatizados testes a funcionalidades que aparentam ser das mais importantes da aplicação visto o objetivo da aplicação ser manusear repositórios e projetos.

Exemplos: Criar um repositório, criar um ficheiro e adicionar (*commitar*) para o repositório, verificar que quando definimos um repositório como público ou privado, ele possui as características de visibilidade desse tipo, etc.

**Repo - Create New Empty Repository**  
**Repo - Create New File and Validate it**  
**Repo - Change Repo Access from Private to Public and Check that anyone can access it**  
**Repo - Search and Remove a Repository with Success**  
**Project - Create a New Project**  
**Project - Edit Project Information**  
**Project - Search and Remove an Existing Project**

Por último mas não menos importante decidi incluir um teste para validar a funcionalidade de adicionar/alterar os dados do perfil do utilizador.

Pode não ser uma *feature* crítica que afete diretamente a utilização do sistema mas a possibilidade de o utilizador ser capaz de alterar corretamente os seus dados (empresa ou website) é importante, até porque é esta informação que ficará visível para a restante comunidade.

**Profile - Edit User Personal Information and Validate it**