

Workshop Robot Framework

Agenda

- ▶ Introduction
- ▶ Installation
- ▶ General Information
- ▶ Robot Framework Syntax
- ▶ Browser Testing
- ▶ Test Architecture
- ▶ Validations
- ▶ Recap

Tim de Groot - Test Automation Engineer

► Experience



► Tools



► Hobbies



Yuri Verweij - Test Automation Engineer

- ▶ Hobbies: Wandelen, 3D Printers, Fotografie
- ▶ TA Engineer sinds 2016, Tester sinds 2009
 - ▶ o.a. Belastingdienst (Erf&Schenk), Alliander (verschillende teams)
- ▶ Robot Framework
 - ▶ RobotFramework-Ambassador
 - ▶ Co-maintainer SeleniumLibrary
 - ▶ Mede-organisator RoboCon
 - ▶ Mede-organisator #RF-NL



Belastingdienst



TESTCODERS

Installation

- ▶ <https://github.com/TestCoders/workshop-robotframework>
- ▶ NodeJS
- ▶ Python
 - ▶ Robot Framework
 - ▶ Robot Framework Browser Library
- ▶ Visual Studio Code
- ▶ Extension: Plugin



Installing additional libraries

- ▶ Find libraries on github, etc.
- ▶ Follow their installation guidelines
- ▶ Example:

```
C:\workshop-robotframework> pip install robotframework-requests
```

General Information - What is Robot Framework

- ▶ Open source
- ▶ Python based
- ▶ Keyword driven
- ▶ Out of the box
 - ▶ Standard Libraries
 - ▶ Running and reporting
 - ▶ Documentation
- ▶ Extendable (rich ecosystem)
 - ▶ Third party libraries: Browsers, APIs, Databases, User Interfaces, etc.
 - ▶ Tools
- ▶ Community



General Information - When to use Robot Framework

- ▶ What to test
 - ▶ System Testing
 - ▶ System Integration Testing
 - ▶ Acceptance Testing
 - ▶ End to End Testing
 - ▶ ~~Unit Testing~~
- ▶ Human readable
- ▶ Learning curve

Robot Framework Syntax - Settings

- ▶ Importing libraries
- ▶ Importing resources
- ▶ Setups and Teardowns

```
*** Settings ***
```

```
Library      Collections
```

```
# Resource    my-keywords.resource
```

```
Documentation    Collections Library Documentation:
```

```
...              https://robotframework.org/robotframework/latest/libraries/Collections.html
```



Robot Framework Syntax - Variables

- ▶ Predefined values
- ▶ Global scope
- ▶ \$, @, & combined with {}

```
*** Variables ***
```

```
${SCALAR}      Martijn
```

```
@{LIST}        Tim      Mark      Yuri
```

```
&{DICTIONARY}  name=Tim   email=tim@testcoders.nl   age=512
```



Robot Framework Syntax - Test Cases

- ▶ Test name
- ▶ Indentation and whitespaces
- ▶ Keyword Arguments

```
*** Test Cases ***
```

```
Log Some Data
```

```
[Documentation]    This is test documentation
```

```
# This is a comment
```

```
Log    message=Data
```

```
Log With A Keyword
```

```
Log Data With A Keyword
```



Robot Framework Syntax - Keywords

- ▶ Same structure as tests
- ▶ Same keyword usage
- ▶ Keywords contain other keywords

```
*** Test Cases ***
```

```
Log With A Keyword
```

```
Log Data With A Keyword
```

```
*** Keywords ***
```

```
Log Data With A Keyword
```

```
[Documentation] This is keyword documentation
```

```
Log Some Data
```

```
Log message=${SCALAR}
```

```
Log List list_=${LIST}
```

```
Log Dictionary dictionary=${DICTIONARY} level=WARN
```



Robot Framework Syntax - Keywords

- ▶ Same structure as tests
- ▶ Same keyword usage
- ▶ Keywords contain other keywords

```
*** Test Cases ***
```

```
Log With A Keyword
```

```
Log Data With A Keyword
```

```
*** Keywords ***
```

```
Log Data With A Keyword
```

```
[Documentation] This is keyword documentation
```

```
Log Some Data
```

```
Log message=${SCALAR}
```

```
Log List list_=${LIST}
```

```
Log Dictionary dictionary=${DICTIONARY} level=WARN
```



Robot Framework Syntax - Log

01-Syntax Log

Generated
20250903 13:54:42 UTC+02:00
7 seconds ago

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	2	2	0	0	00:00:00	<div></div>

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						<div></div>

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
01-Syntax	2	2	0	0	00:00:00	<div></div>
01-Syntax.Syntax	2	2	0	0	00:00:00	<div></div>

Test Execution Errors

20250903 13:54:41.964 **WARN** Dictionary size is 3 and it contains following items:
age: 512
email: tim@testcoders.nl
name: Tim

Test Execution Log

SUITE 01-Syntax

Full Name: 01-Syntax

Source: C:\Users\tim-d\Documents\workshop-robotframework\01-syntax

Start / End / Elapsed: 20250903 13:54:41.921 / 20250903 13:54:41.968 / 00:00:00.047

Status: 2 tests total, 2 passed, 0 failed, 0 skipped

SUITE Syntax

Full Name: 01-Syntax.Syntax

Documentation: Collections Library Documentation: <https://robotframework.org/robotframework/latest/libraries/Collections.html>

Source: C:\Users\tim-d\Documents\workshop-robotframework\01-syntax\syntax.robot

Start / End / Elapsed: 20250903 13:54:41.952 / 20250903 13:54:41.966 / 00:00:00.014

Status: 2 tests total, 2 passed, 0 failed, 0 skipped

TEST Log Some Data

TEST Log With A Keyword



Robot Framework Syntax - Log

-

TEST

Verify Start Of The Game Invalid Name

Full Name:

Workshop-Robotframework.04-Validations.Solution.Solution.Verify Start Of The Game Invalid Name

Start / End / Elapsed:

20250903 11:48:25.837 / 20250903 11:48:36.709 / 00:00:10.872

Status:

FAIL

Message:

TimeoutError: locator.waitFor: Timeout 10000ms exceeded.
Call log:
- waiting for locator('//*[@data-testid="adventure-container"]') to be visible

+ KEYWORD

Start TestRPG

+ KEYWORD

Start Playing

+ KEYWORD

Prepare And Start Game name=p build=mage

- KEYWORD

Validate The Game Started expected_name=p expected_stats=A level 1 mage

Start / End / Elapsed:

20250903 11:48:26.575 / 20250903 11:48:36.709 / 00:00:10.134

- KEYWORD

Browser.Wait For Elements State selector=\${ADVENTURE_CONTAINER}

Documentation:

Waits for the element found by `selector` to satisfy state option.

Tags:

PageContent, Wait

Start / End / Elapsed:

20250903 11:48:26.576 / 20250903 11:48:36.709 / 00:00:10.133

11:48:36.707

INFO

Robot Framework Syntax - Documentation

- ▶ Builtin Library
<https://robotframework.org/robotframework/latest/libraries/BuiltIn.html>
- ▶ General library information and usage
- ▶ Keywords
- ▶ Mandatory and optional arguments
- ▶ Examples

Browser Testing

*** Settings ***

Library Browser

Documentation Browser Library Documentation:
... <https://marketsquare.github.io/robotframework-browser/Browser.html>

*** Variables ***

General

\${URL} <https://test-rpg.vercel.app/>

Locators

\${HOMEPAGE_BANNER} `//*[@data-testid="hero"]`

*** Test Cases ***

Verify TestRPG Homepage

New Browser browser=chromium headless=\${False}

New Context

New Page url=\${URL}

Wait For Elements State selector=\${HOMEPAGE_BANNER}

Take Screenshot

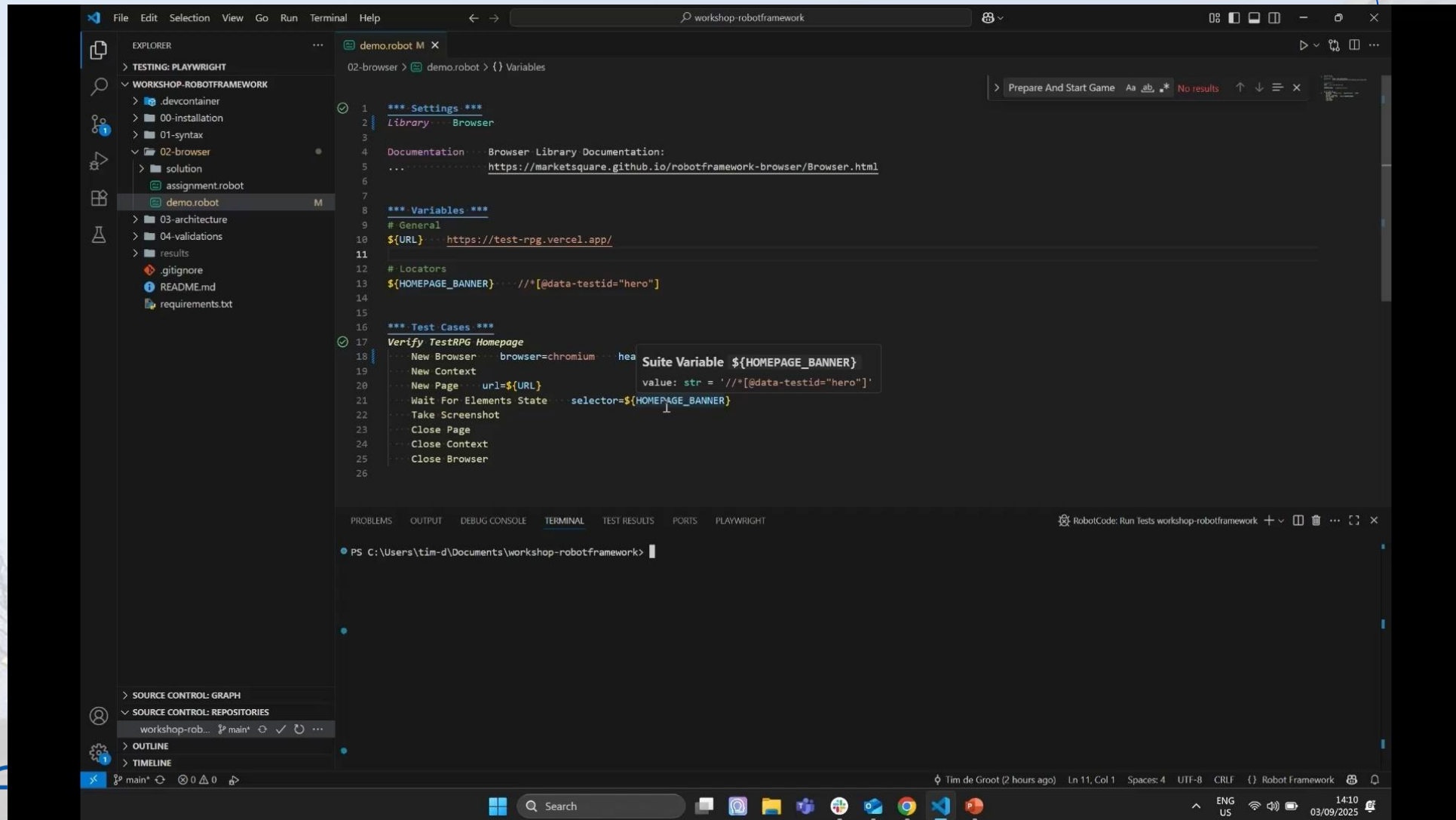
Close Page

Close Context

Close Browser



Browser Testing



The screenshot displays the Visual Studio Code editor with a Robot Framework test file named `demo.robot` open. The Explorer sidebar on the left shows the project structure, including folders for `00-installation`, `01-syntax`, `02-browser`, `03-architecture`, `04-validations`, and `results`. The `demo.robot` file is selected under the `02-browser` folder.

The main editor area shows the following Robot Framework code:

```
*** Settings ***
Library    Browser

Documentation    Browser Library Documentation:
...             https://marketsquare.github.io/robotframework-browser/Browser.html

*** Variables ***
# General
${URL}    https://test-rpg.vercel.app/

# Locators
${HOMEPAGE_BANNER}    //*[@data-testid="hero"]

*** Test Cases ***
Verify TestRPG Homepage
    New Browser    browser=chromium    headless=True
    New Context
    New Page    url=${URL}
    Wait For Elements State    selector=${HOMEPAGE_BANNER}
    Take Screenshot
    Close Page
    Close Context
    Close Browser
```

A tooltip is visible over the `selector=${HOMEPAGE_BANNER}` line, showing the suite variable definition: `Suite Variable ${HOMEPAGE_BANNER} value: str = '//*[@data-testid="hero"]'`.

The bottom of the window shows the Windows taskbar with the search bar and several application icons. The system tray indicates the time is 14:10 on 03/09/2025.

Browser Testing - Assignment

- ▶ <https://github.com/TestCoders/workshop-robotframework>
- ▶ See 02-browser/assignment.robot
- ▶ Add the required keywords to complete the testcase
 - ▶ 1. Start playing
 - ▶ 2. Enter a username
 - ▶ 3. (Optionally) Select a different build
 - ▶ 4. Start the game

Browser Testing - Assignment Solution

*** Test Cases ***

Verify Start Of The Game

```
New Browser      browser=chromium      headless=${False}
New Context
New Page      url=${URL}
# Click "Click here to play"
Click      selector=${CLICK_HERE_TO_PLAY_BUTTON}
# Enter your character name
Type Text      selector=${CHARACTER_NAME_INPUT}      txt=Tim
# (Optional) Select your build
Select Options By      ${BUILD_SELECT}      value      mage
# Click "Start!"
Click      selector=${START_GAME_BUTTON}
Close Page
Close Context
Close Browser
```



Test Architecture - Low Level Keywords

- ▶ Atomic keywords
- ▶ Not composed of other keywords
- ▶ Lowest level of keywords
- ▶ Also known as library keywords

*** Test Cases ***

Verify Login Some Email

[Documentation] Only using low level keywords

Type Text selector=\${USERNAME_INPUT} txt=some-email@test.com

Type Secret selector=\${PASSWORD_INPUT} secret=CoolPassword123!

Click \${LOGIN_BUTTON}

Verify Login Other Email

Type Text selector=\${USERNAME_INPUT} txt=other-email@test.com

Type Secret selector=\${PASSWORD_INPUT} secret=OtherCoolPassword123!

Click \${LOGIN_BUTTON}



Test Architecture - High Level Keywords

- ▶ Composite keywords
- ▶ Composed of other keywords
- ▶ Used for structuring tests
- ▶ Also known as user keywords

*** Test Cases ***

Verify Login With A Keyword For Some Email

[Documentation] Using a high level keyword, also using positional arguments
Login some-email@test.com CoolPassword123!

Verify Login With A Keyword For Other Email

[Documentation] Using a high level keyword, also using named arguments
Login username=other-email@test.com password=OtherPassword123!

*** Keywords ***

Login

[Arguments] \${username} \${password}
Type Text selector=\${USERNAME_INPUT} txt=\${username}
Type Secret selector=\${PASSWORD_INPUT} secret=\${password}
Click \${LOGIN_BUTTON}



Test Architecture - Assignment

- ▶ <https://github.com/TestCoders/workshop-robotframework>
- ▶ See 03-test-architecture/assignment.robot
- ▶ Create the required high level keywords to structure the testcase

Test Architecture - Assignment Solution

*** Test Cases ***

Verify Start Of The Game

```
Start TestRPG
Start Playing
Prepare And Start Game    name=Tim    build=mage
Close TestRPG
```

*** Keywords ***

Start TestRPG

```
New Browser    browser=chromium    headless=${False}
New Context
New Page    url=${URL}
```

Start Playing

```
Click    selector=${CLICK_HERE_TO_PLAY_BUTTON}
```

Prepare And Start Game

```
[Arguments]    ${name}    ${build}
Type Text    selector=${CHARACTER_NAME_INPUT}    txt=${name}
Select Options By    ${BUILD_SELECT}    value    ${build}
Click    selector=${START_GAME_BUTTON}
```

Close TestRPG

```
Close Page
Close Context
Close Browser
```

Validations

*** Test Cases ***

Verify Start Of The Game

Start TestRPG

Start Playing

Prepare And Start Game name=z build=mage

Close TestRPG



Validations

► Validate positives!

*** Test Cases ***

Verify Start Of The Game Valid Name

```
Start TestRPG
Start Playing
Prepare And Start Game    name=Pietje    build=mage
Validate The Game Started    expected_name=Pietje    expected_stats=A level 1 mage
Close TestRPG
```

Verify Start Of The Game Invalid Name

```
Start TestRPG
Start Playing
Prepare And Start Game    name=p    build=mage
Validate The Game Started    expected_name=p    expected_stats=A level 1 mage
Close TestRPG
```

*** Keywords ***

Validate The Game Started

```
[Arguments]    ${expected_name}    ${expected_stats}
Wait For Elements State    selector=${ADVENTURE_CONTAINER}
${actual_name}    Get Text    selector=${CHARACTER_NAME_LABEL}
Should Be Equal As Strings    first=${expected_name}    second=${actual_name}
${actual_stats}    Get Text    selector=${CHARACTER_STATS_LABEL}
Should Be Equal As Strings    first=${expected_stats}    second=${actual_stats}
```

Further Resources

- ▶ Slack
- ▶ User Guide
<https://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html>
- ▶ Robot Framework Docs
<https://docs.robotframework.org/docs>
- ▶ Awesome Robot Framework
<https://github.com/MarketSquare/awesome-robotframework>
- ▶ RoboCon
<https://www.robocon.io/>



Recap

- ▶ Installation
- ▶ Syntax
- ▶ Browser testing
- ▶ Architecture
- ▶ Validations

- ▶ Questions?


TESTCODERS

ROBOT FRAMEWORK FUNDAMENTALS

Leer in 2 dagen hoe je snel en slim jouw tests automatiseert

De training
De 2-daagse training Robot Framework Fundamentals is dé praktische en toegankelijke kickstart voor iedereen die testautomatisering onder de knie wil krijgen. Je leert van de allerbesten: onze trainers zijn Robot Framework Ambassadors en actief in de internationale community. Ze brengen niet alleen diepgaande kennis mee, maar ook jarenlange praktijkervaring in uiteenlopende projecten.

Wat leer je?

- ✓ Zelfstandig een Robot Framework project opzetten
- ✓ Testen automatiseren voor Web én API's
- ✓ Debuggen, structureren en herbruikbare componenten maken
- ✓ Best practices voor leesbare en onderhoudbare tests
- ✓ Slim gebruik van keywords, libraries en variabelen

Praktische info

-  woensdag 5 & 12 november 2025
-  09.00 - 17.00
-  Werkspoorcathedraal Utrecht
-  Inclusief lunch!

Nu tijdelijk €995 excl. BTW
Normaal €1.295

Onze trainers


TIM DE GROOT


MARK MOERTS


YURI VERWEIJ

JOIN US


API Testing

*** Settings ***

Library Collections
Library RequestsLibrary

Documentation Documentation: <https://github.com/MarketSquare/robotframework-requests>

*** Variables ***

\${URL} <https://test-rpg.vercel.app/api>

*** Test Cases ***

Validate Builds

<code>\${response}</code>	GET	<code>url=\${URL}/builds</code>	
Dictionary Should Contain Key		<code>dictionary=\${response.json()}</code>	<code>key=thief</code>
Dictionary Should Contain Key		<code>dictionary=\${response.json()}</code>	<code>key=mage</code>
Dictionary Should Contain Key		<code>dictionary=\${response.json()}</code>	<code>key=knight</code>
Dictionary Should Contain Key		<code>dictionary=\${response.json()}</code>	<code>key=brigadier</code>



Python Keywords

keywords.py

```
import random

def get_random_build_from_list(list):
    result = random.choice(list)
    print(f'Input: {list}, result: {result}')
    return result
```

python.robot

```
*** Settings ***
Library          keywords.py

*** Variables ***
@{BUILDS}        Thief      Mage      Knight      Brigadier

*** Test Cases ***
Get Random Build
    ${build}      Get Random Build From List    list=${BUILDS}
    Should Contain    container=${BUILDS}    item=${build}
```

