VALA

What makes Robot Framework stand out?

Experiences of using another test automation framework

Timo Stordell, Test Automation Lead at VALA

Myself

30 years in IT

15+ years somehow in test automation

Experience in telecom, healthcare, finance, insurance

Passionate pipeliner, on a crusade to build the future of SW development.

I’ve been using Robot Framework since 2014

Then I had 5 years of experience using another framework.

Here is my story, comparing the experiences.

Not everything should be tested with Robot Framework

Just trying to tell here
what are the important features
of a good acceptance test framework
from my perspective

VALA

Robot Framework

Acceptance test automation framework

Enables writing test cases in business language

Developed since 2005

Open source, developed by foundation backed by 70+ companies

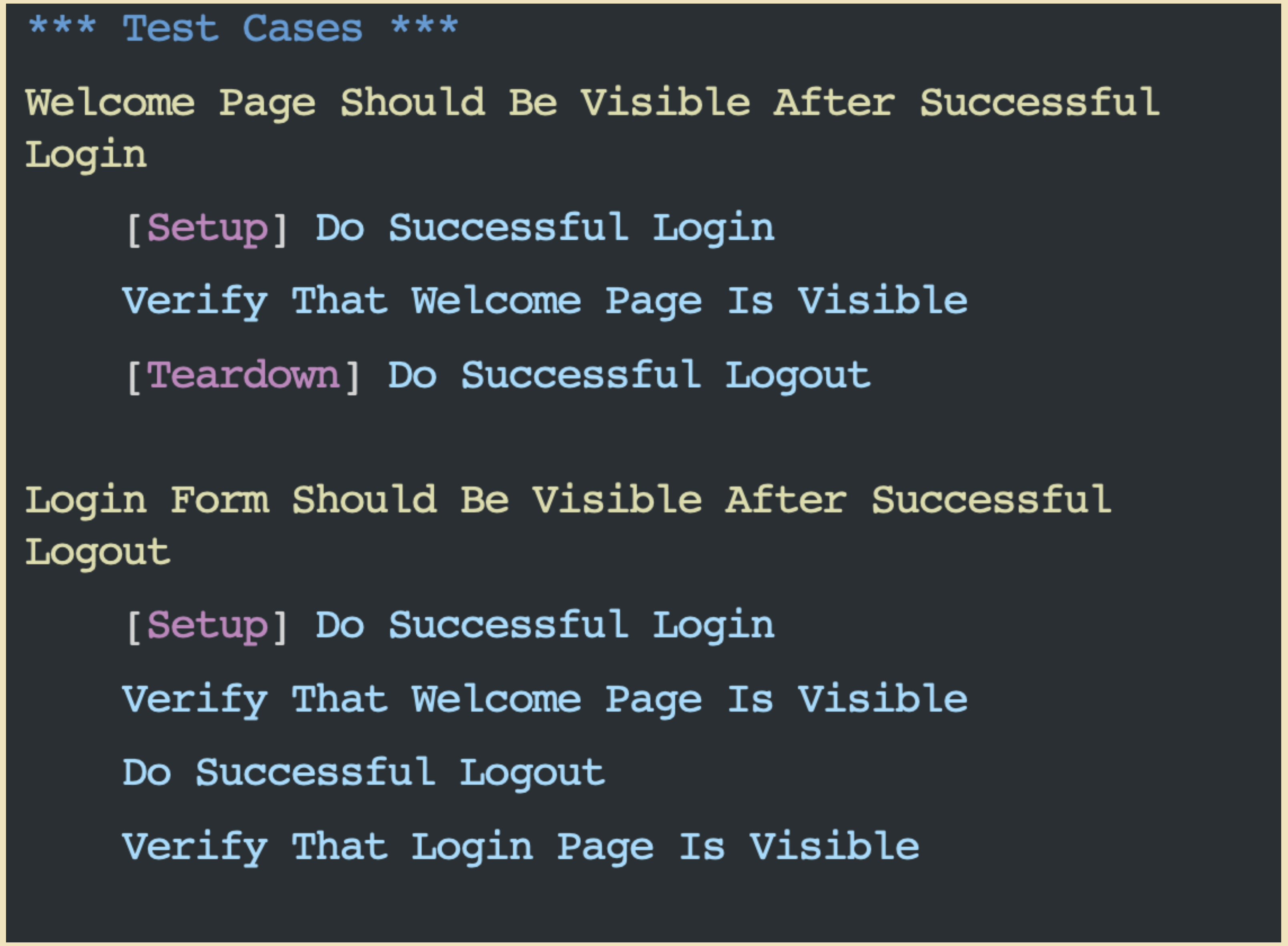
33 000 users*

So what are good qualities for an acceptance test automation framework?

Some things coming up on the following slides

Ability to use business language

- Ability to share the tests with all stakeholders.
- Ability to link requirements to verification.
- Still, design needs to be considered. It’s easy to build incomprehensible solutions.
- Naturally, this is a need only for high level testing. Unit tests are done by developers for developers. There the best language is most likely the coding language.



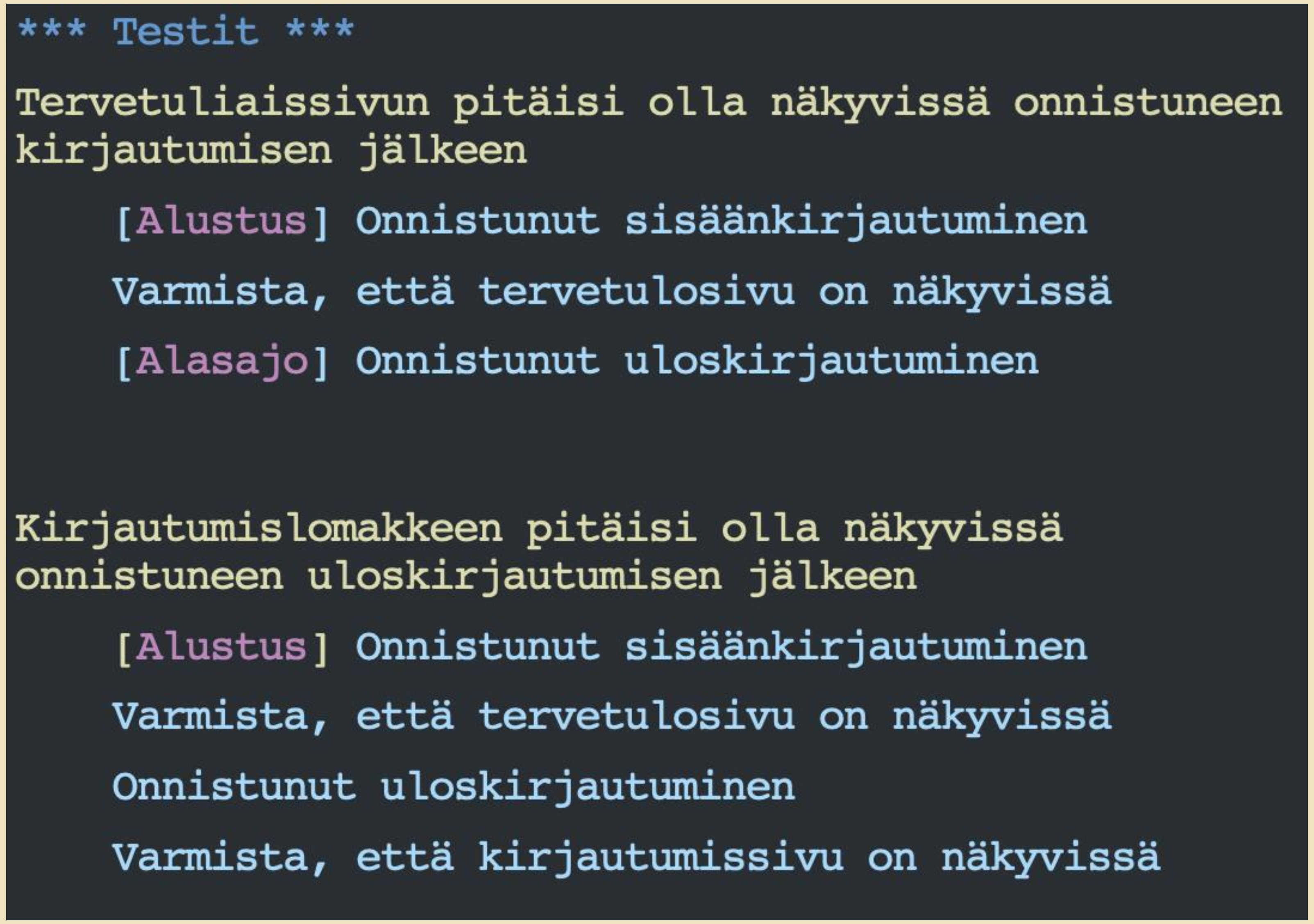
```
*** Test Cases ***
Welcome Page Should Be Visible After Successful
Login
    [Setup] Do Successful Login
    Verify That Welcome Page Is Visible
    [Teardown] Do Successful Logout

Login Form Should Be Visible After Successful
Logout
    [Setup] Do Successful Login
    Verify That Welcome Page Is Visible
    Do Successful Logout
    Verify That Login Page Is Visible
```

VALA

Ymmärränkö myös suomea?

Tottakai!

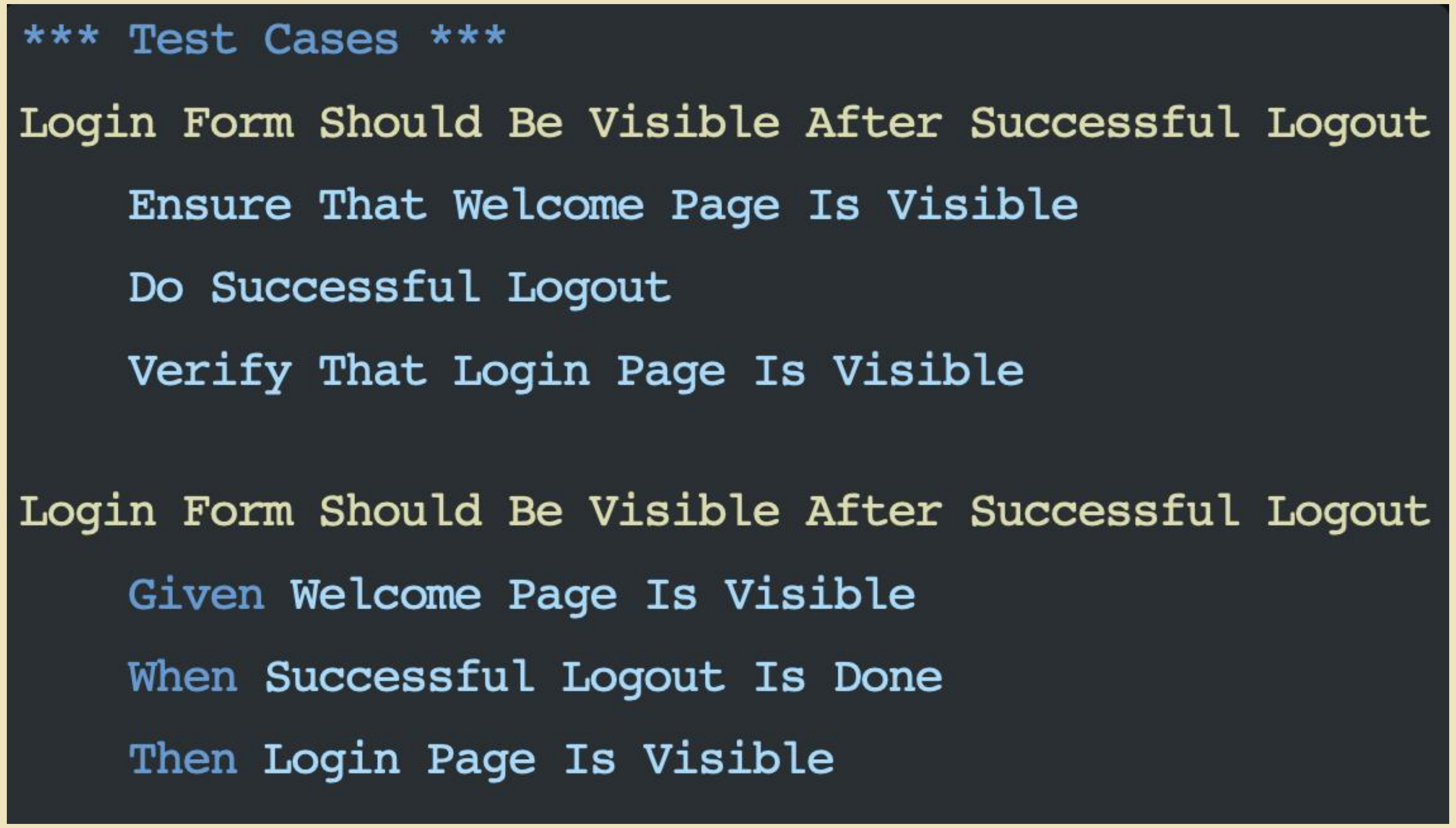
It should be straightforward to use any language you wish.



```
*** Testit ***

Tervetuliaissivun pitäisi olla näkyvissä onnistuneen
kirjautumisen jälkeen
    [Alustus] Onnistunut sisäänkirjautuminen
    Varmista, että tervetulosivu on näkyvissä
    [Alasajo] Onnistunut uloskirjautuminen


Kirjautumislomakkeen pitäisi olla näkyvissä
onnistuneen uloskirjautumisen jälkeen
    [Alustus] Onnistunut sisäänkirjautuminen
    Varmista, että tervetulosivu on näkyvissä
    Onnistunut uloskirjautuminen
    Varmista, että kirjautumissivu on näkyvissä
```

Supporting Gherkin

- Many tools support Gherkin structure (given-when-then).
- Program code should be used only for complex parts.

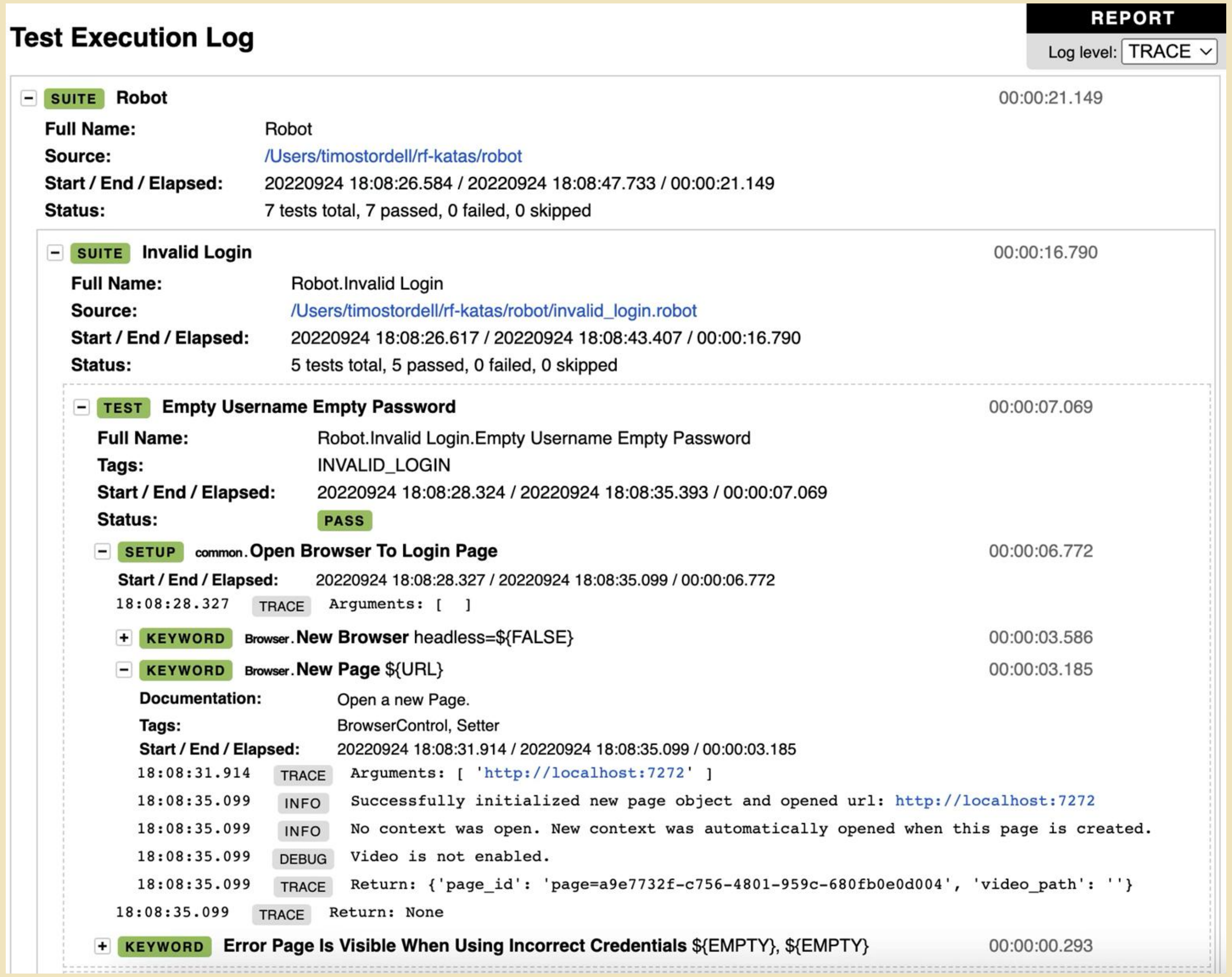


```
*** Test Cases ***
Login Form Should Be Visible After Successful Logout
    Ensure That Welcome Page Is Visible
    Do Successful Logout
    Verify That Login Page Is Visible

Login Form Should Be Visible After Successful Logout
    Given Welcome Page Is Visible
    When Successful Logout Is Done
    Then Login Page Is Visible
```

VALA

Providing detailed logs

Selectable viewing level (info / debug / trace)

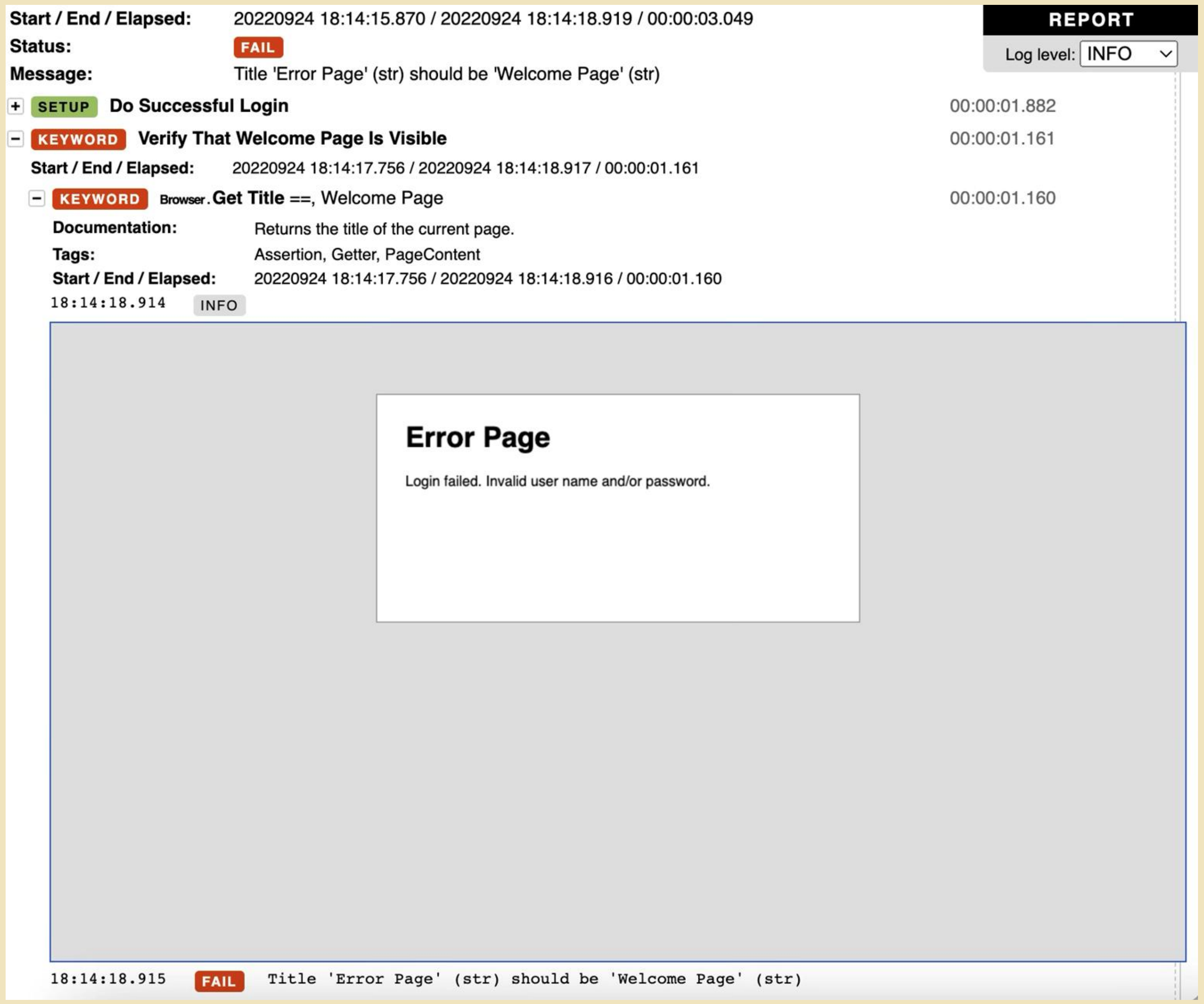


Test Execution Log

```
SUITE Robot                                                         00:00:21.149
  Full Name:            Robot
  Source:               /Users/timostordell/rf-katas/robot
  Start / End / Elapsed: 20220924 18:08:26.584 / 20220924 18:08:47.733 / 00:00:21.149
  Status:               7 tests total, 7 passed, 0 failed, 0 skipped

  SUITE Invalid Login                                               00:00:16.790
    Full Name:            Robot.Invalid Login
    Source:               /Users/timostordell/rf-katas/robot/invalid_login.robot
    Start / End / Elapsed: 20220924 18:08:26.617 / 20220924 18:08:43.407 / 00:00:16.790
    Status:               5 tests total, 5 passed, 0 failed, 0 skipped

    TEST Empty Username Empty Password                              00:00:07.069
      Full Name:            Robot.Invalid Login.Empty Username Empty Password
      Tags:                 INVALID_LOGIN
      Start / End / Elapsed: 20220924 18:08:28.324 / 20220924 18:08:35.393 / 00:00:07.069
      Status:               PASS

      SETUP common.Open Browser To Login Page                       00:00:06.772
        Start / End / Elapsed: 20220924 18:08:28.327 / 20220924 18:08:35.099 / 00:00:06.772
        18:08:28.327  TRACE  Arguments: [  ]

        KEYWORD Browser.New Browser headless=${FALSE}               00:00:03.586
        KEYWORD Browser.New Page ${URL}                             00:00:03.185
          Documentation:         Open a new Page.
          Tags:                  BrowserControl, Setter
          Start / End / Elapsed: 20220924 18:08:31.914 / 20220924 18:08:35.099 / 00:00:03.185
          18:08:31.914  TRACE  Arguments: [ 'http://localhost:7272' ]
          18:08:35.099  INFO   Successfully initialized new page object and opened url: http://localhost:7272
          18:08:35.099  INFO   No context was open. New context was automatically opened when this page is created.
          18:08:35.099  DEBUG  Video is not enabled.
          18:08:35.099  TRACE  Return: {'page_id': 'page=a9e7732f-c756-4801-959c-680fb0e0d004', 'video_path': ''}
        18:08:35.099  TRACE  Return: None

      KEYWORD Error Page Is Visible When Using Incorrect Credentials ${EMPTY}, ${EMPTY}   00:00:00.293
```

VALA

Failures come with screenshots



```
Start / End / Elapsed:   20220924 18:14:15.870 / 20220924 18:14:18.919 / 00:00:03.049
Status:                  FAIL
Message:                 Title 'Error Page' (str) should be 'Welcome Page' (str)
+ SETUP Do Successful Login                                              00:00:01.882
− KEYWORD Verify That Welcome Page Is Visible                            00:00:01.161
  Start / End / Elapsed:   20220924 18:14:17.756 / 20220924 18:14:18.917 / 00:00:01.161
  − KEYWORD Browser.Get Title ==, Welcome Page                           00:00:01.160
    Documentation:         Returns the title of the current page.
    Tags:                  Assertion, Getter, PageContent
    Start / End / Elapsed: 20220924 18:14:17.756 / 20220924 18:14:18.916 / 00:00:01.160
    18:14:18.914   INFO
```

Error Page

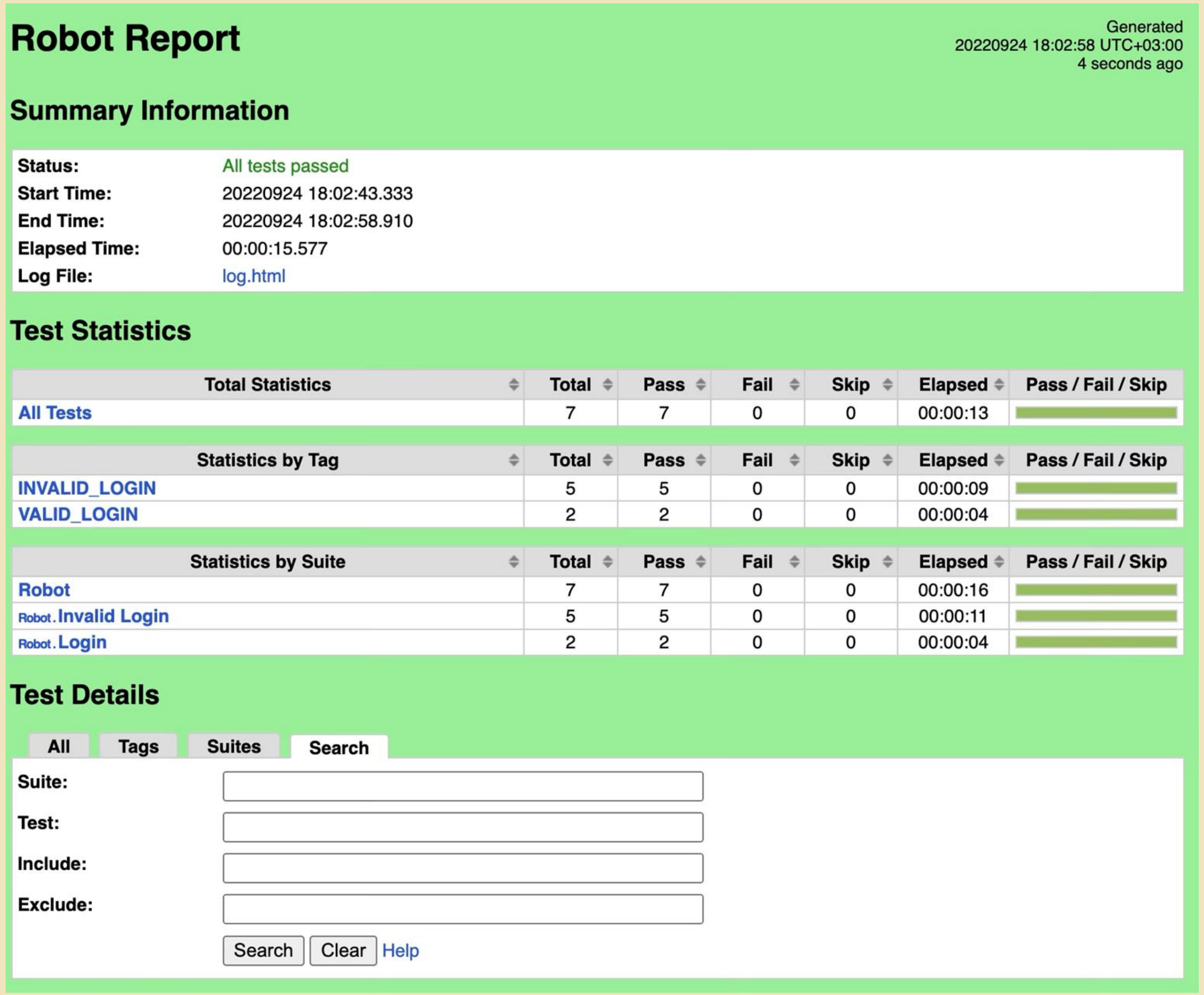
Login failed. Invalid user name and/or password.

```
    18:14:18.915   FAIL   Title 'Error Page' (str) should be 'Welcome Page' (str)
```

VALA

Creating high-level reports

- Something that is easy to share (e.g. HTML)
- Ability to make custom reports



Robot Report

Generated
20220924 18:02:58 UTC+03:00
4 seconds ago

Summary Information

Status:	All tests passed
Start Time:	20220924 18:02:43.333
End Time:	20220924 18:02:58.910
Elapsed Time:	00:00:15.577
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	7	7	0	0	00:00:13	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
INVALID_LOGIN	5	5	0	0	00:00:09	
VALID_LOGIN	2	2	0	0	00:00:04	

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Robot	7	7	0	0	00:00:16	
Robot . Invalid Login	5	5	0	0	00:00:11	
Robot . Login	2	2	0	0	00:00:04	

Test Details

All Tags Suites Search

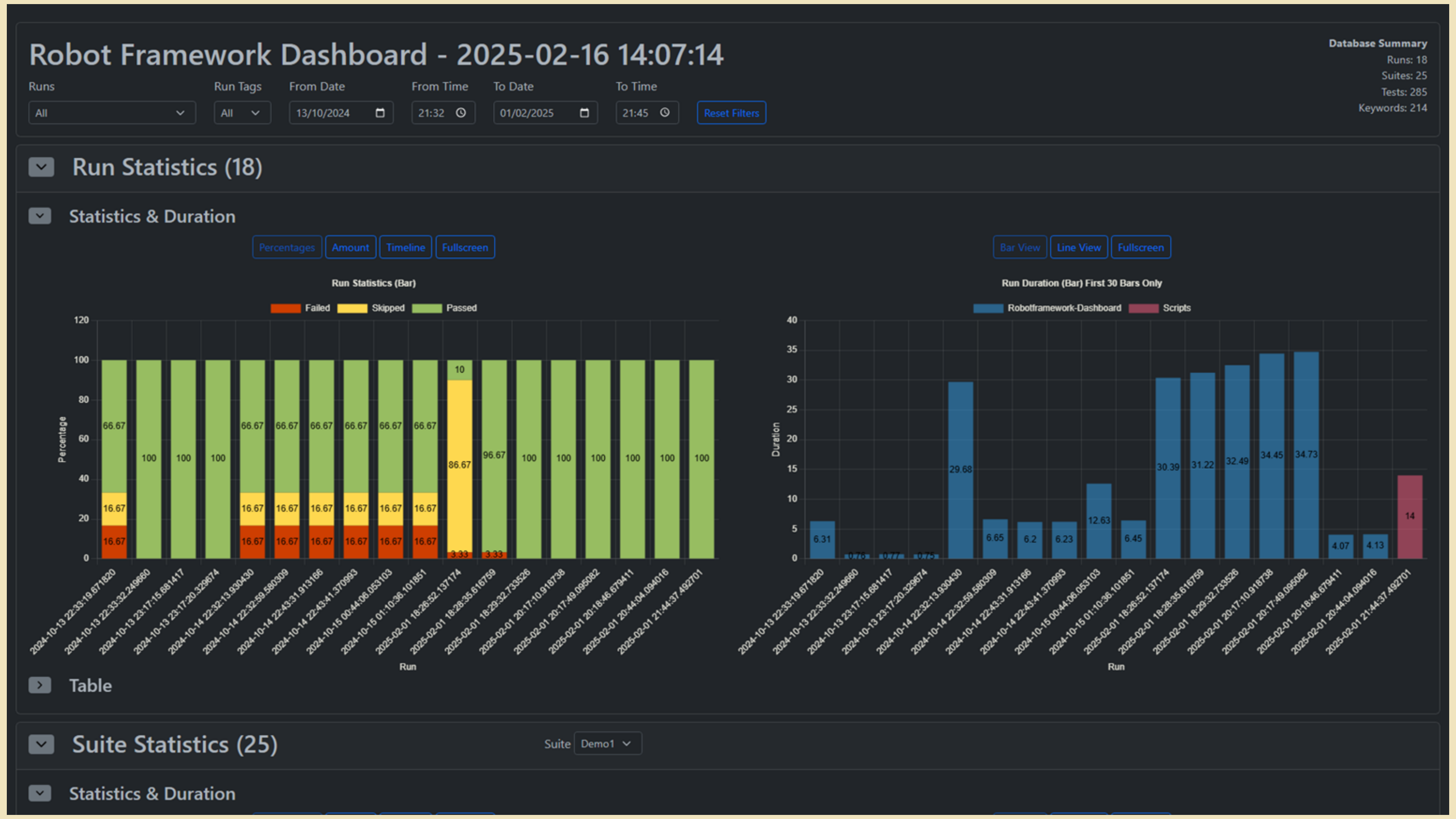
Suite:	
Test:	
Include:	
Exclude:	

Search Clear Help

VALA

Automatic dashboards

- For following trends
- For a summary of repeated tests



Robot Framework Dashboard - 2025-02-16 14:07:14

Runs: All
Run Tags: All
From Date: 13/10/2024
From Time: 21:32
To Date: 01/02/2025
To Time: 21:45

Database Summary
Runs: 18
Suites: 25
Tests: 285
Keywords: 214

Run Statistics (18)

Statistics & Duration

Table

Suite Statistics (25)

Suite: Demo1

Statistics & Duration

Remember all the use cases

For reports, logs and screenshots

Reviewing tests

Collecting statistics

Debugging

VALA

Dry-run - reflecting what you have at hand

- Check the tests that will be included, before executing
- Plan long test runs
- Find tests that are never executed
- Collect statistics

VALA

Tags

- Just use, without need for implementation.
- Include and exclude tests from runs.
- Utilise in creating custom reports.

VALA

Tests need a scripting language

to speed up the development.

No time used for compilation before execution, including downloading all the libraries.

No need to keep the whole code base compilable during development or debugging.

Automation

Test frameworks are for creating tests, automation is done with other tools.

Robot Framework is executed on command-line, thus easy to integrate to CI/CD pipelines.

It has no dependencies by itself.

Extending the framework

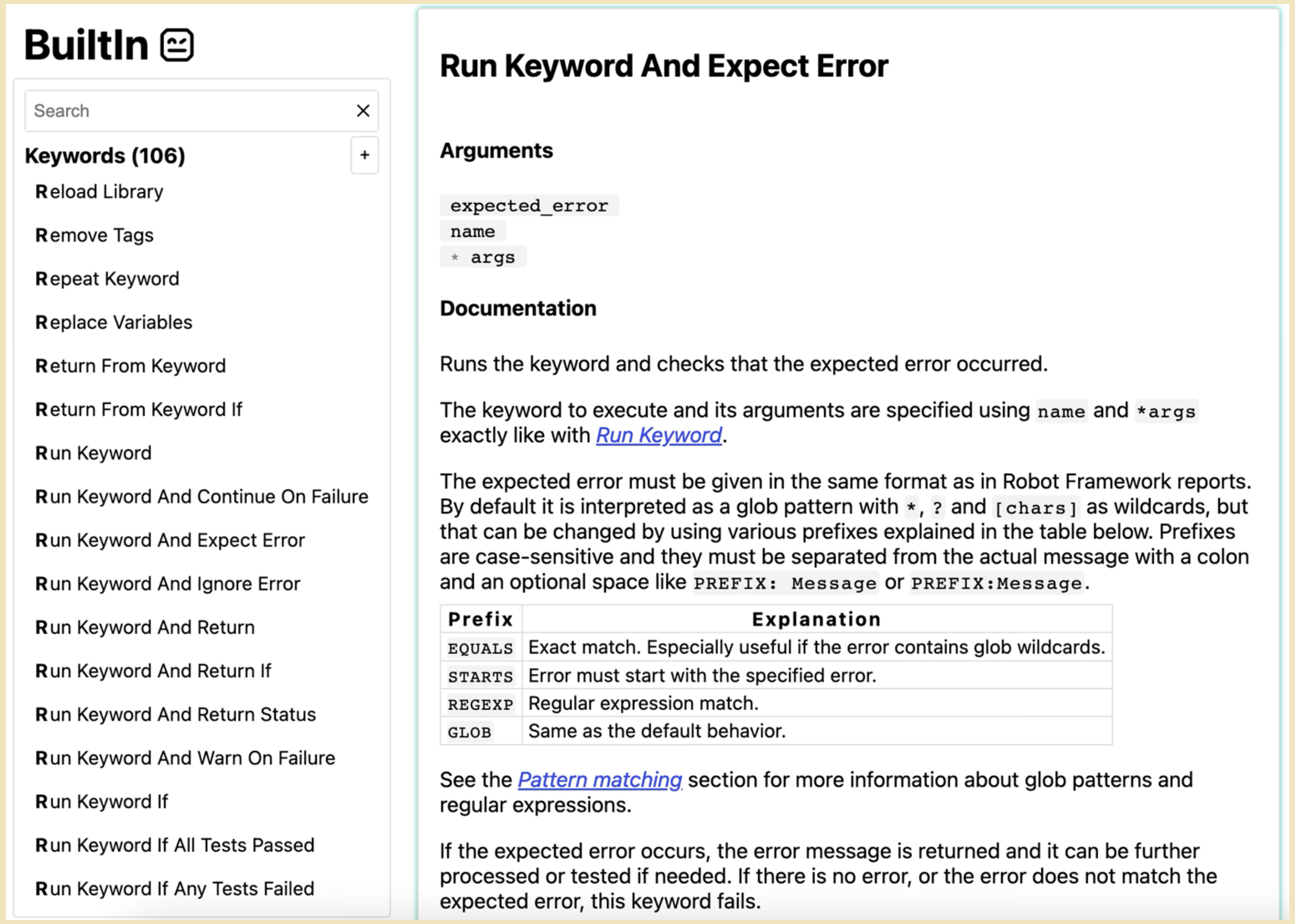
Creating own libraries for custom or proprietary needs.

Doing remote execution e.g. with other languages.

Using listeners to act based on events in test execution.

Documenting it all

- Have a detailed user‘s guide.
- Include very good documentation on public test libraries.
- Provide tools to generate the documentation for your own libraries.



BuiltIn
Search
Keywords (106)
Reload Library
Remove Tags
Repeat Keyword
Replace Variables
Return From Keyword
Return From Keyword If
Run Keyword
Run Keyword And Continue On Failure
Run Keyword And Expect Error
Run Keyword And Ignore Error
Run Keyword And Return
Run Keyword And Return If
Run Keyword And Return Status
Run Keyword And Warn On Failure
Run Keyword If
Run Keyword If All Tests Passed
Run Keyword If Any Tests Failed

Run Keyword And Expect Error

Arguments

`expected_error`
`name`
`* args`

Documentation

Runs the keyword and checks that the expected error occurred.

The keyword to execute and its arguments are specified using `name` and `*args` exactly like with *Run Keyword*.

The expected error must be given in the same format as in Robot Framework reports. By default it is interpreted as a glob pattern with `*`, `?` and `[chars]` as wildcards, but that can be changed by using various prefixes explained in the table below. Prefixes are case-sensitive and they must be separated from the actual message with a colon and an optional space like `PREFIX: Message` or `PREFIX:Message`.

Prefix	Explanation
EQUALS	Exact match. Especially useful if the error contains glob wildcards.
STARTS	Error must start with the specified error.
REGEXP	Regular expression match.
GLOB	Same as the default behavior.

See the *Pattern matching* section for more information about glob patterns and regular expressions.

If the expected error occurs, the error message is returned and it can be further processed or tested if needed. If there is no error, or the error does not match the expected error, this keyword fails.

VALA

Updating

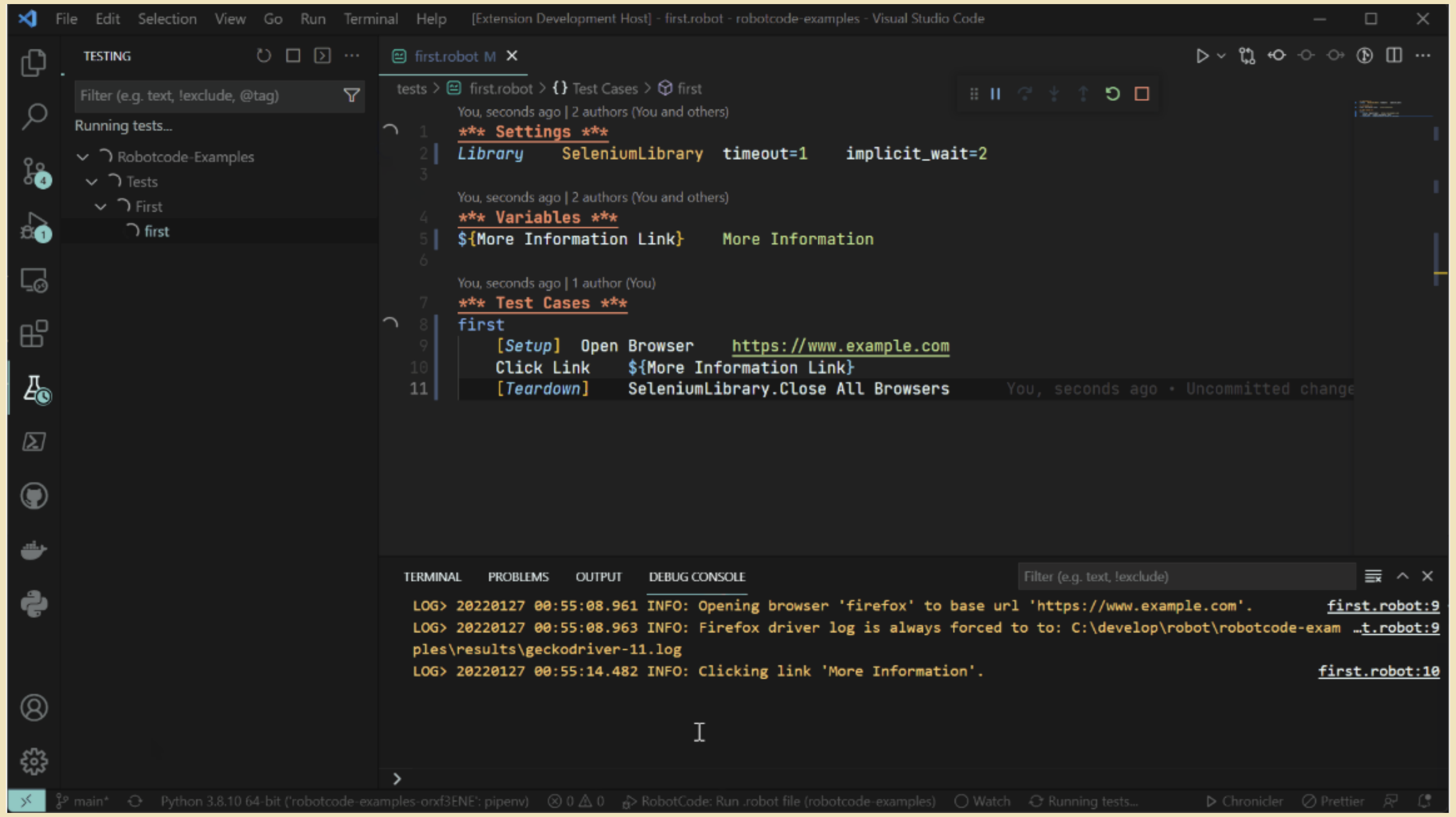
Development of Robot Framework is backed by foundation, foundation backed by dozens of companies (and RoboCon events).

Regular releasing, yearly major releases.

Popular libraries are kept well up-to-date, updating your environment to latest made easy.

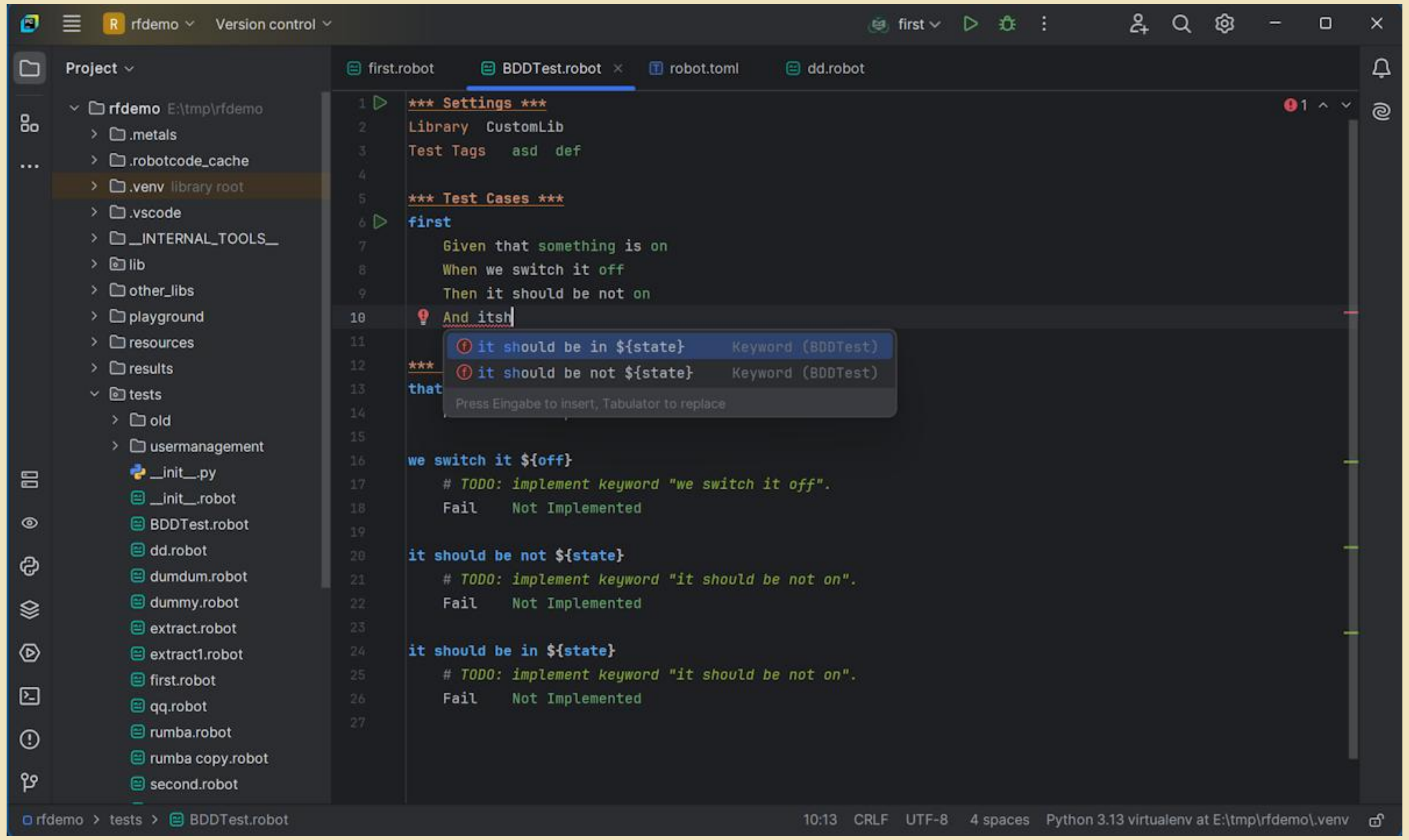
Support for popular development environments

- Robotcode for Visual Studio Code and JetBrains Pycharm/IntelliJ.
- Some others are supported as well but marginally used nowadays.
- RIDE an option for newcomers and non-coders.



```
*** Settings ***
Library    SeleniumLibrary  timeout=1    implicit_wait=2

*** Variables ***
${More Information Link}    More Information

*** Test Cases ***
first
    [Setup]  Open Browser    https://www.example.com
    Click Link    ${More Information Link}
    [Teardown]    SeleniumLibrary.Close All Browsers
```

```
LOG> 20220127 00:55:08.961 INFO: Opening browser 'firefox' to base url 'https://www.example.com'.      first.robot:9
LOG> 20220127 00:55:08.963 INFO: Firefox driver log is always forced to to: C:\develop\robot\robotcode-exam …t.robot:9
ples\results\geckodriver-11.log
LOG> 20220127 00:55:14.482 INFO: Clicking link 'More Information'.                                      first.robot:10
```



```
*** Settings ***
Library  CustomLib
Test Tags   asd  def

*** Test Cases ***
first
    Given that something is on
    When we switch it off
    Then it should be not on
    And itsh
        it should be in ${state}        Keyword (BDDTest)
        it should be not ${state}       Keyword (BDDTest)
        Press Eingabe to insert, Tabulator to replace
***
that

we switch it ${off}
    # TODO: implement keyword "we switch it off".
    Fail    Not Implemented

it should be not ${state}
    # TODO: implement keyword "it should be not on".
    Fail    Not Implemented

it should be in ${state}
    # TODO: implement keyword "it should be not on".
    Fail    Not Implemented
```

VALA

Reasons or excuses for not using Robot Framework

Disliking Python.

“It’s easier for developers to use the same language as for code”.

Robot Framework is for acceptance testing = testing features, not code. It’s important to share the tests with all stakeholders (not just coders).

Playwright is better.

AI won’t need it.

My project

That chose a different framework

What happened?

Selecting the framework

No proper evaluation was done.

Selected one that is not very well known.

Taken just because someone tossed it in (and was arrogant?).

Company is a member of Robot Framework Foundation.

VALA

VALA

Statistics of two projects

	Project 1	**Project 2**
Test framework	Robot Framework	Framework X
System under testing	Embedded	Web app / data node
Governing regulation	Tight	Moderate
Test developers	20	10
Test developed in 1.5 years	10 000	about 500
Tests executed regularly in CI	7 000	100
Execution cycle	weekly	nightly

Summary of the two frameworks

	Robot Framework	**Framework X**
Use of business language	Excellent	Limited
Reports	Excellent	Ok-ish
Logs and screenshots	Excellent	What you make it
Dryrun	Yes	You may create one
Tags	Easy	Cumbersome
Compilation required	No	Yes
Automation is fluent	Yes	Yes
Extending capabilities	Excellent	Good
Documentation	Excellent	Decent
Updating	Fluent	Difficult
IDE support	Good	Good

The community is what makes it happen

Robot Framework has Slack with 33 000 members

Hundreds of public libraries

Thousands of GitHub repositories

Foundation, open-source

RoboCon

So, what makes a good test automation framework?

Key takeaways

Sharing

Test cases in business language
High-level reports
Dashboards

Executing

Command-line execution for easy integration
Tags for including and excluding
Dryrun to check test run content

Development

IDE support
Scripting, no compilation needs
Regular releases
Documentation
Extendability

Reporting

Detailed logs and reports out-of-the-box
Tags for filtering
Screenshots for debugging (web apps)
Dashboards for summaries and trends

Timo Stordell
Test Automation Lead
timo.stordell@valagroup.com
+358 40 545 9865

Thanks! Questions?

VALA