

# Développer avec Robot Framework

23-24 Septembre 2025

Nickel NANTES

RÉMI PICARD

APPUYER SUR LA BARRE D'ESPACE OU LES FLÈCHES POUR DÉMARRER

# Tour de table

- Expérience dev
- Expérience testeur
- Expérience avec Robot Framework
- Attente vis à vis de cette formation ?

# Qui suis-je ?

- Rémi PICARD
- Dev Scala Cobalt
- 4 ans chez Nickel
- 4 ans d'expérience avec Robot Framework 🤖
- Passionné par les technos Web, Data et DevOps
- 13 ans d'expérience dans l'IT 💡
- Joueur d'échecs ♟



# Jour 1

## Découverte de Robot Framework

**Robot Framework => "Robot"**  
**RF / RBF / RBT**

# Plan

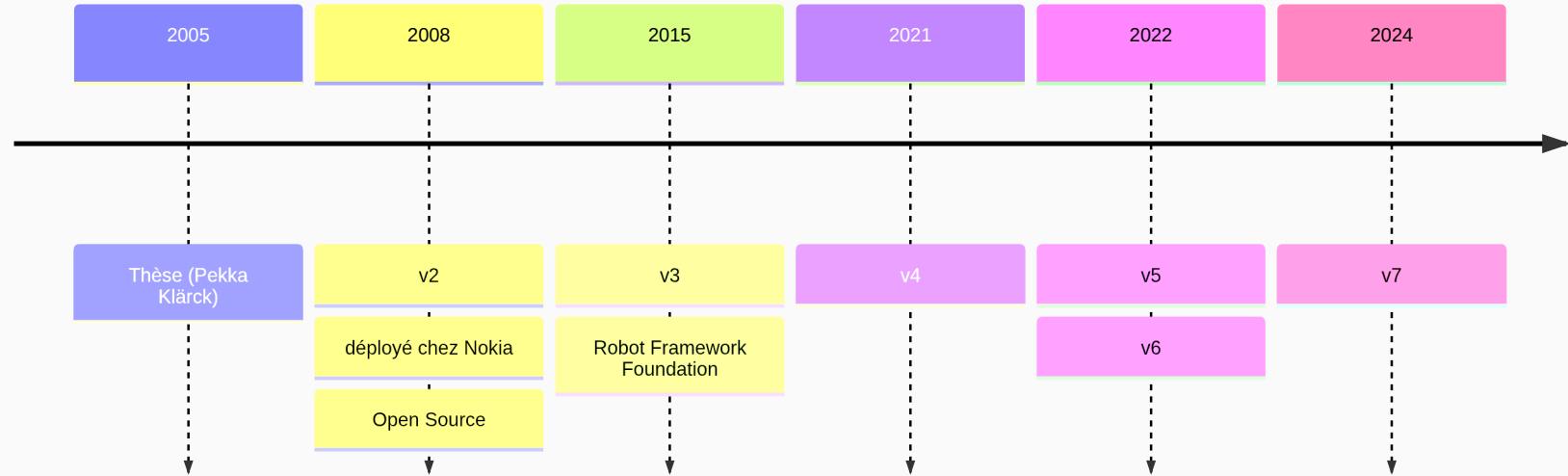
- Généralités
- Rappels Python
- Comprendre le fonctionnement de Robot
- Installer l'env de dev
- Apprendre le langage Robot
- Découvrir la ligne de commande `robot`
- Codelab Tests API
- Codelab Tests UI avec Playwright

# Présentation

- Outil d'automatisation
- Langage
- Open Source codé en Python
- Fonctionnalités clefs en main (assertions, rapport de tests...)
- Extensible via des librairies Robot Framework (HTTP, JSON, SQL, Kafka ...)
- Extensible via des librairies Python

# Histoire

## 20 ans déjà !

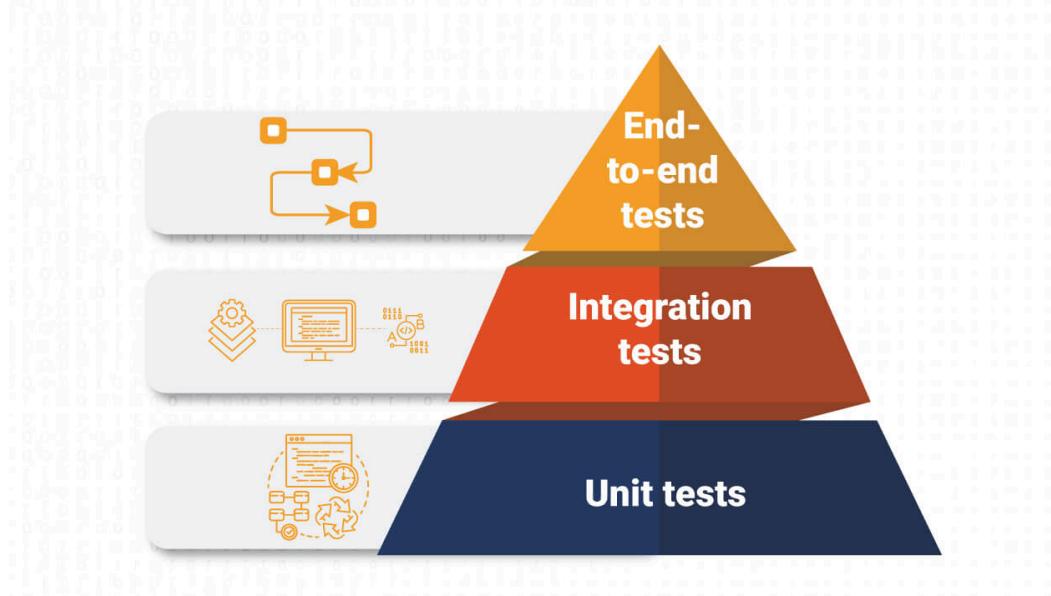


# Communauté

- Slack
- RoboCon / RBCN : conférence annuelle à Helsinki 
- [Documentation](#)

# Pyramide des tests

## Tests End-To-End

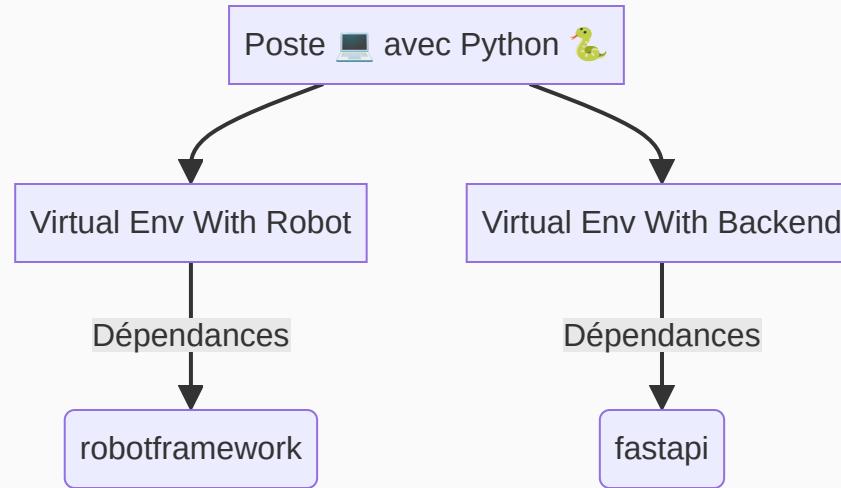


Source: <https://blog.takima.fr/saffranchir-de-la-pyramide-des-tests/>

**Env de dev** 

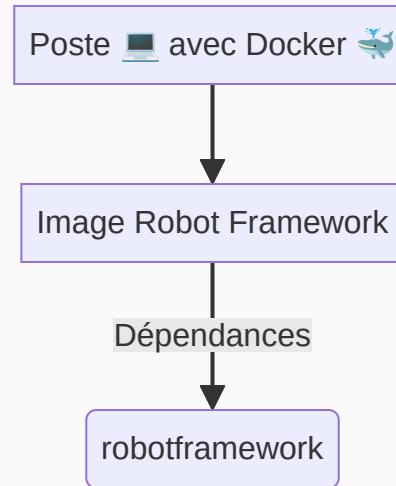
# Installation

## Python Virtual Environment



### Python Virtual Environment

# Installation Docker



# Installation PyCharm / VSCode

Set up your IDE

# Codelab



- [Installation env de dév](#)
- [Hello World](#)
- [Configuration IDE](#)

# Développer avec Python 🐍

## Rappels

# Types natifs

Type	Mot-clé	Exemple
Entier	int	42, -17, 0
Flottant	float	3.14, 2.5e-3
Booléen	bool	True, False
Chaîne	str	"Hello", 'World'
Tableau	list	["Hello", "World"]
Map	dict	{"key": "value", "key2": "value2"}
Aucun	NoneType	None

[Documentation](#)

# Méthodes

```
1 def ma_fonction(arg1: int, arg2: str, default_arg="default", *args, **kwargs) -> int:
2     # ...
3     return 42
4
5 ma_fonction(42, "Quarante-deux")
6 ma_fonction(42, "Quarante-deux", "override default value")
7 ma_fonction(42, "Quarante-deux", "default", 1, 2, 3)
8 ma_fonction(42, "Quarante-deux", "default", robot="Nono", john="Doe")
9 ma_fonction(42, "Quarante-deux", "default", 1, 2, 3, robot="Nono", john="Doe")
```

# Développer avec Robot Framework

## Tests, Variables, Keyword, Python, Structure ...

# Tests

- fichier `*.robot`
- indentation comme en Python 
- ensemble de phrases 
- Section `*** Test Cases ***`
- Import librairies dans `*** Settings ***`

# Tests

## Déclaration

```
1 *** Settings ***
2 Library      String
3
4 *** Test Cases ***
5 Mon Premier Test
6     ${chaine}=  Generate Random String    10
7     Log      Hello ${chaine}
```

# Variables

- Même types qu'en Python 
- Syntaxe  `${...}` (comme en Bash)
- Création dans un Test ou Keyword
- Création en global dans la section  `*** Variables ***`
- Import des variables Python possible

# Variables

## Section Variables

```
1 *** Variables ***
2 ${nombre}      42
3 ${chaine}      Ma chaîne de caractères
4 @{tab}          1  2  3
5 &{map}          clef1=valeur1    clef2=valeur2
6
7 *** Test Cases ***
8 Teste Variables
9     Log    nombre=${nombre}
10    Log    chaine=${chaine}
11    Log    tab=${tab}
12    Log    map=${map}
```

## Variables

# Variables

## Déclaration

```
1 *** Test Cases ***
2 Creation Variable
3     ${ma_variable}    Set Variable    C3PO
4     Log    ma_variable=${ma_variable}
5
6     # Nouvelle syntaxe RF>=7.0
7     VAR    ${ma_variable}    C3PO
8     Log    ma_variable=${ma_variable}
```

# Variables

## Portée

- Local \${hi} = Set Variable Hello
- Test Set Test Variable \${HI} Hello
- Suite Set Suite Variable \${HI} Hello
- Global Set Global Variable \${HI} Hello
- 💡 Limiter au maximum la portée
- Nouvelle syntaxe (uniforme): VAR \${variable} scope=SUITE

# Variables

## Syntaxe VAR

It is recommended to use the VAR syntax introduced in Robot Framework 7.0 for creating variables in different scopes instead of the Set Global/Suite/Test/Local Variable keywords.

### BuiltIn

# Variables

## Import YAML

- option --variablefile / -V
- robot --variablefile conf/local.yaml tests/14-variablefile.robot

# Variables

## Option CLI

- `option --variable / -v`
- `robot --variable env:LOCAL tests/14-variable.robot`

# Variables

## Import Python

```
1 # resources/mes_variables_python.py  
2 variable_python = 42
```

```
1 *** Settings ***  
2 Variables      resources/mes_variables_python.py  
3  
4 *** Test Cases ***  
5 Utiliser Variable Python  
6     Log      variable_python=${variable_python}
```

# Variables

## ENV

- Syntaxe `%{VARIABLE_ENV=default_value}`

# Injection Python

## Evaluate

```
1 *** Test Cases ***
2 Teste Evaluate
3     ${nb}=    Evaluate    41 + 1
4     Log      nb=${nb}
5
6 Teste Evaluate Autre Syntaxe
7     ${nb}=    Set Variable   ${${41 + 1}}
8     Log      nb=${nb}
```

# Keyword Concept

- Ensemble de mots clés (séparés par 1 espace)
- Forme une phrase 
- Représente une **action** 
- Déclaré dans la section `*** Keywords ***`
- Robot Framework traduit les phrases en appels Python 

# Keyword

## Syntaxe Robot Framework

```
1 *** Keywords ***
2 Mon Premier Keyword
3     Log    Hello World
4
5 Mon Premier Keyword Avec Argument
6     [Arguments]    ${name}
7     Log    Hello ${name}
8
9 Mon Premier Keyword Avec Argument Et Return
10    [Arguments]   ${name}
11    Log    Hello ${name}
12    RETURN    42
```

# Keyword

## Arg dans Keyword

```
1  *** Keywords ***
2  Keyword Avec ${arg1} Intégré
3      Log    Hello ${arg1}
4
5  Keyword Avec ${arg1} Intégré Et Arguments
6      [Arguments]    ${name}
7      Log    Hello ${arg1}, ${name}
8
9  *** Test Cases ***
10 Appel Keywords
11     ${arg1}    Set Variable    Ma Variable
12     Keyword Avec ${arg1} Intégré
13     Keyword Avec ${arg1} Intégré Et Arguments    Arg2
```

# Keyword

## List (args) / Dict (kwargs)

```
1  ```text {1-6|1-6,14-18|1,8-12|1,8-12,14-15,20-23|all}
2  *** Keywords ***
3  Keyword Avec Args
4      [Arguments]    @{list}
5      FOR    ${i}    IN    @{list}
6          Log    i=${i}
7      END
8
9  Keyword Avec Kwargs
10     [Arguments]   &{map}
11     FOR    ${k}    ${v}    IN    &{map}
12         Log    key=${k}, value=${v}
13     END
14
15  *** Test Cases ***
16  Appel Keywords
17      ${list}    Create List    1    2    3
18  # ${list}    Create List    1    2    3
19  VAR    @{list}    1    2    3
20  Keyword Avec Args    ${list}
21
22      ${map}    Create Dictionary    cle1=valeur1    cle2=valeur2
23  # ${map}    Create Dictionary    cle1=valeur1    cle2=valeur2
24  VAR    &{map}    cle1=valeur1    cle2=valeur2
```