สยามชำนาญกิจ
SIAM CHAMNAN KIT

Create your own Library

Extending Robot Framework

Why ?

Robot Framework already has many keywords and libraries

STANDARD	EXTERNAL	OTHER
Builtin Provides a set of often needed generic keywords. Always automatically available without imports.	**Dialogs** Provides means for pausing the execution and getting input from users.	**Collections** Provides a set of keywords for handling Python lists and dictionaries.
OperatingSystem Enables various operating system related tasks to be performed in the system where Robot Framework is running.	**Remote** Special library acting as a proxy between Robot Framework and libraries elsewhere. Actual libraries can be running on different machines and be implemented using any programming language supporting XML-RPC protocol.	**Screenshot** Provides keywords to capture screenshots of the desktop.
String Library for generating, modifying and verifying strings.	**Telnet** Makes it possible to connect to Telnet servers and execute commands on the opened connections.	**XML** Library for generating, modifying and verifying XML files.
Process Library for running processes in the system. New in Robot Framework 2.8.	**DateTime** Library for date and time conversions. New in Robot Framework 2.8.5.	

https://robotframework.org/#libraries

STANDARD	EXTERNAL	OTHER

AppiumLibrary

Library for Android and iOS testing. It uses Appium internally.

AutoRecorder

Library which allows to automatically record video for test/suites execution.

ConfluentKafkaLibrary

Library for python confluent kafka.

Database Library (Python)

Python based library for database testing. Works with any Python interpreter, including Jython.

Diff Library

Library to diff two files together.

ArchiveLibrary

Library for handling zip- and tar-archives.

Browser Library

Robot Framework Browser library is a modern web testing library powered by 🎭 **Playwright**.
Aiming for 🚀 speed, ✅ reliability and 🔬 visibility.

CURFLibrary

Library for testing CAN bus with support for ISO-TP and UDS.

DataDriver Library

Library for Data-Driven Testing with external 📩 data tables (csv, xls, xlsx, etc.).
🧬 Pairwise Combinatorial Testing support.

Django Library

Library for **Django**, a Python web framework.

AutoItLibrary

Windows GUI testing library that uses AutoIt freeware tool as a driver.

CncLibrary

Library for driving a CNC milling machine.

Database Library (Java)

Java-based library for database testing. Usable with Jython. Available also at **Maven central**.

Debug Library

A debug library for RobotFramework, which can be used as an interactive shell(REPL) also.

DoesIsLibrary

Library with autogenerated keywords like Is Something, Does Someting created form

https://robotframework.org/#libraries

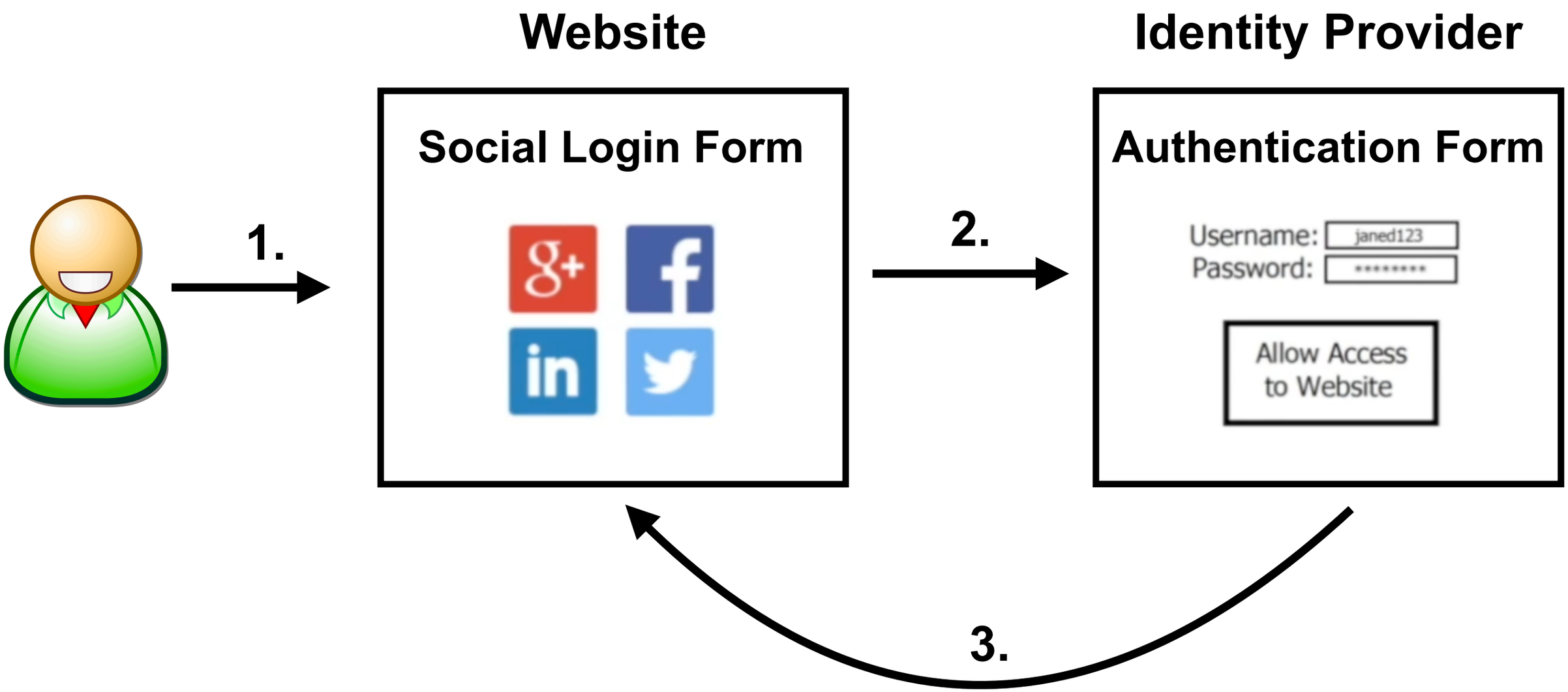
Extending Robot Framework

The need for specific custom keywords
Functionality is missing
Wrapping functionality into custom keywords

Extending Robot Framework

Wrapping functionality into custom keywords
Seed data
Clean up tasks
External APIs

Use case



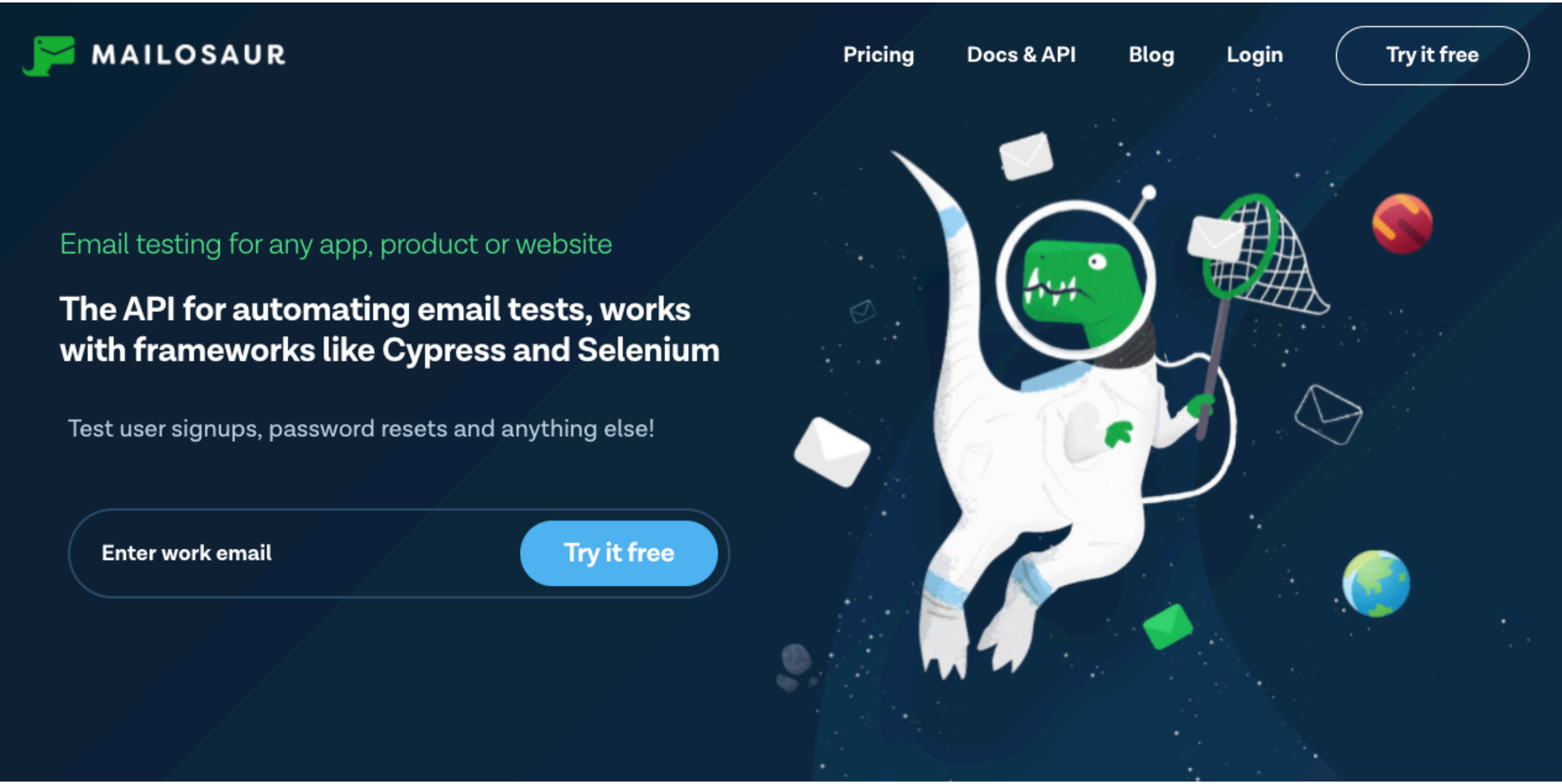
Challenges with GitHub Social Authentication

Github user can only be used once at a time
Github security verifications (via email)
Control the state of a user

Problems ?

Get user for testing
Update status of user
Working with email
External APIs

Working with email



https://mailosaur.com/

Mailosaur library

$pip install mailosaur

https://pypi.org/project/mailosaur/

Problems with email ?

Get verification code from email
Delete mail in inbox
Working with email

Control state of Github user and repos

$pip install PyGithub

https://pypi.org/project/PyGithub/

Write python code !!

Create custom library with Robot Framework

Hello library

Create file hello.robot

```
*** Settings ***
Library    HelloLibrary.py


*** Testcases ***
Testcase 01
    Say Hi  somkiat

Testcase 02
    Say Hi  somkiat
    Result Should Be  Hi, somkiat
```

Run with robot

$pybot hello.robot

```
[ ERROR ] Error in file '/Users/somkiat/data/slide/robot-framework/advance-robot-
urse/workshop/hello/hello.robot': Test library 'HelloLibrary.py' does not exist.
==============================================================================
Hello
==============================================================================
==============================================================================
Testcase 01                                                           | FAIL |
No keyword with name 'Say Hi' found.
------------------------------------------------------------------------------
Testcase 02                                                           | FAIL |
No keyword with name 'Say Hi' found.
------------------------------------------------------------------------------
Hello                                                                 | FAIL |
2 critical tests, 0 passed, 2 failed
2 tests total, 0 passed, 2 failed
==============================================================================
```

Create file HelloLibrary.py

```
class HelloLibrary:

    def __init__(self):
        self._result = ''

    def say_hi(self, name):
        print('Hi, %s' % name)
        self._result = 'Hi, %s' % name

    def result_should_be(self, expected):
        if self._result != expected:
            raise AssertionError('%s != %s' % (self._result,
expected))
```

Create file HelloLibrary.py

```
class HelloLibrary:

    def __init__(self):
        self._result = ''

    def say_hi(self, name):
        print('Hi, %s' % name)
        self._result = 'Hi, %s' % name

    def result_should_be(self, expected):
        if self._result != expected:
            raise AssertionError('%s != %s' % (self._result,
expected))
```

Run with robot

$pybot hello.robot

```
==============================================================================
Hello
==============================================================================
Testcase 01                                                           | PASS |
------------------------------------------------------------------------------
Testcase 02                                                           | PASS |
------------------------------------------------------------------------------
Hello                                                                 | PASS |
2 critical tests, 2 passed, 0 failed
2 tests total, 2 passed, 0 failed
==============================================================================
```

Show log message in console

Show log message in console

```
class HelloLibrary:

    def __init__(self):
        self._result = ''

    def say_hi(self, name):
        print('Hi, %s' % name)
        self._result = 'Hi, %s' % name

    def result_should_be(self, expected):
        if self._result != expected:
            raise AssertionError('%s != %s' % (self._result,
expected))
```

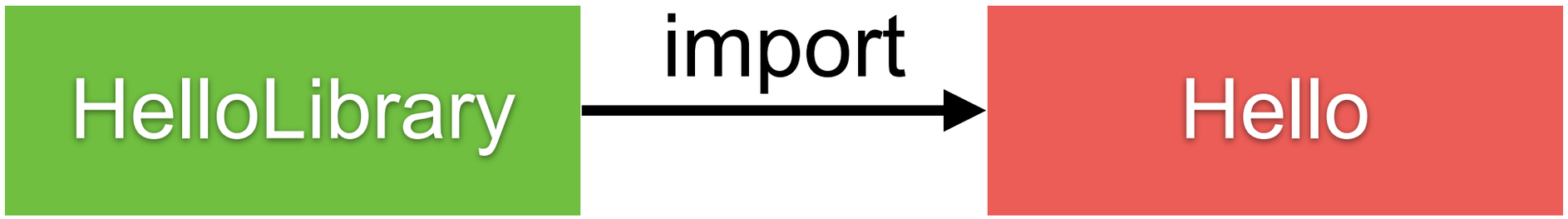
Show log message in console

```
from robot.api import logger

class HelloLibrary:

    def say_hi(self, name):
        self._hello.set_name(name)

        logger.console('Say hi with %s' %(name))
```

https://github.com/robotframework/robotframework

Separate logic from library file

Separate logic from library file



Create file Hello.py

```
class Hello:
    def __init__(self):
        self._result = ''

    def set_name(self, name):
        self._name = name

    def get_result(self):
        return 'Hi, %s' %(self._name)
```

Update file HelloLibrary.py

```
from hello import Hello

class HelloLibrary:

    def __init__(self):
        self._hello = Hello()
        self._result = ''

    def say_hi(self, name):
        self._hello.set_name(name)

    def result_should_be(self, expected):
        if self._hello.get_result() != expected:
            raise AssertionError('%s != %s' %
(self._result, expected))
```

Update file HelloLibrary.py

```
from hello import Hello

class HelloLibrary:

    def __init__(self):
        self._hello = Hello()
        self._result = ''

    def say_hi(self, name):
        self._hello.set_name(name)

    def result_should_be(self, expected):
        if self._hello.get_result() != expected:
            raise AssertionError('%s != %s' %
(self._result, expected))
```

Update file HelloLibrary.py

```
from hello import Hello

class HelloLibrary:

    def __init__(self):
        self._hello = Hello()
        self._result = ''

    def say_hi(self, name):
        self._hello.set_name(name)

    def result_should_be(self, expected):
        if self._hello.get_result() != expected:
            raise AssertionError('%s != %s' %
(self._result, expected))
```

Run with robot

$pybot hello.robot

```
==============================================================================
Hello
==============================================================================
Testcase 01                                                           | PASS |
------------------------------------------------------------------------------
Testcase 02                                                           | PASS |
------------------------------------------------------------------------------
Hello                                                                 | PASS |
2 critical tests, 2 passed, 0 failed
2 tests total, 2 passed, 0 failed
==============================================================================
```

Improve name of library

Create file hello.robot

```
*** Settings ***
Library    HelloLibrary
```

```
*** Testcases ***
Testcase 01
    Say Hi  somkiat

Testcase 02
    Say Hi  somkiat
    Result Should Be  Hi, somkiat
```

Run with robot

$pybot hello.robot

```
[ ERROR ] Error in file '/Users/somkiat/data/slide/robot-framework/adva
nce-robot-course/workshop/hello/hello.robot': Importing test library 'H
elloLibrary' failed: ModuleNotFoundError: No module named 'HelloLibrary
'

Traceback (most recent call last):
  None
PYTHONPATH:
  /usr/local/Cellar/robot-framework/3.0.2_1/libexec/bin
  /usr/local/Cellar/python/3.6.4_3/Frameworks/Python.framework/Versions
/3.6/lib/python36.zip
  /usr/local/Cellar/python/3.6.4_3/Frameworks/Python.framework/Versions
/3.6/lib/python3.6
  /usr/local/Cellar/python/3.6.4_3/Frameworks/Python.framework/Versions
/3.6/lib/python3.6/lib-dynload
```

Run robot with PythonPath

$pybot -P . hello.robot

```
==============================================================================
Hello
==============================================================================
Testcase 01                                                           | PASS |
------------------------------------------------------------------------------
Testcase 02                                                           | PASS |
------------------------------------------------------------------------------
Hello                                                                 | PASS |
2 critical tests, 2 passed, 0 failed
2 tests total, 2 passed, 0 failed
==============================================================================
```

Custom name of keyword

Change name of keyword

```
from robot.api.deco import keyword

class HelloLibrary:

    @keyword('Try to say hi with')
    def say_hi(self, name):
```

Use new keyword

```
*** Settings ***
Library   HelloLibrary.py


*** Testcases ***
Testcase 01
    Try to say hi with  somkiat

Testcase 02
    Try to say hi with  somkiat
    Result Should Be  Hi, somkiat
```

Run robot again

$pybot -P . hello.robot

```
==============================================================================
Hello
==============================================================================
Testcase 01                                                           | PASS |
------------------------------------------------------------------------------
Testcase 02                                                           | PASS |
------------------------------------------------------------------------------
Hello                                                                 | PASS |
2 critical tests, 2 passed, 0 failed
2 tests total, 2 passed, 0 failed
==============================================================================
```

Default value of keyword

Default value of keyword

```
def say_hi2(self, name='no name 1', name2='no name 2’):
```

```
    self._hello.set_name(name)
```

Default value of keyword

```
*** Settings ***
Library   HelloLibrary.py

*** Testcases ***

Testcase 03
    Say Hi2
    Say Hi2  name1
    Say Hi2  name1  name2
    Say Hi2  name2=name2
    Say Hi2  name=name1
    Say Hi2  name2=name2  name=name1
```

Free style keyword

Free style keyword

```
def say_hi_all(self, **names):

    for name, value in names.items():
        print('%s = %s' % (name, value))
```

Free style keyword

```
*** Settings ***
Library   HelloLibrary.py

*** Testcases ***

Testcase 03

    Say Hi All  key=value   name=somkiat  age=30
```

Embedding arguments into keyword names

http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#embedding-arguments-into-keyword-name

Add arguments into keyword

```
*** Settings ***
Library   HelloLibrary2

*** Testcases ***
Testcase 01

    Hello somkiat with age 30 year(s)
```

${name} ${age}

Add arguments into keyword

```
from robot.api import logger

from robot.api.deco import keyword

class HelloLibrary2:

    @keyword('Hello ${name} with age ${age:\d+} year(s)')
    def say_hi(self, name, age):

        logger.console('Hello %s with age %s' %(name,
age))
```

More readable and understanding

```
*** Settings ***
Library   HelloLibrary2

*** Testcases ***
Testcase 01

    Hello "somkiat" with age "30" year(s)
```

${name} ${age}

More readable and understanding

```
from robot.api import logger

from robot.api.deco import keyword

class HelloLibrary2:

    @keyword('Hello "${name}" with age "${age:\d+}" year(s)')
    def say_hi(self, name, age):

        logger.console('Hello %s with age %s' %(name, age))
```

Add document to library

Add document of library

```
from hello import Hello

class HelloLibrary:
    """ Hello Library to *Hello* with name

    Calling from ``set_name`` method
    """
```

Add document of methods

```
def say_hi(self, name):
        """ Say hi with name

        Examples:
        | Say hi | name 1 |
        | Say hi | name 2 |
        """
        self._hello.set_name(name)
```

Add document of methods

```
def result_should_be(self, expected):
        """ Verifies that the current result is
``expected``.

        Examples:
        | Result Should Be | Hi, name 1 |
        | Result Should Be | Hi, name 2 |
        """
        if self._hello.get_result() != expected:
            raise AssertionError('%s != %s' %
(self._result, expected))
```

Generate document of library

http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#libdoc

Generate document of library

$python **-m robot.libdoc** -P . HelloLibrary HelloLibrary.html

Generate document of library

HelloLibrary

Library scope: test case
Named arguments: supported

Introduction

Hello Library to **Hello** with name

Calling from `set_name` method

Shortcuts

Result Should Be · **S**ay Hi

Keywords

Keyword	Arguments	Documentation
Result Should Be	*expected*	Verifies that the current result is `expected`. Examples: Result Should Be | Hi, name 1 Result Should Be | Hi, name 2
Say Hi	*name*	Say hi with name Examples: Say hi | name 1 Say hi | name 2

Altogether 2 keywords.
Generated by Libdoc on 2018-08-28 23:12:19.

Need more Knowledges

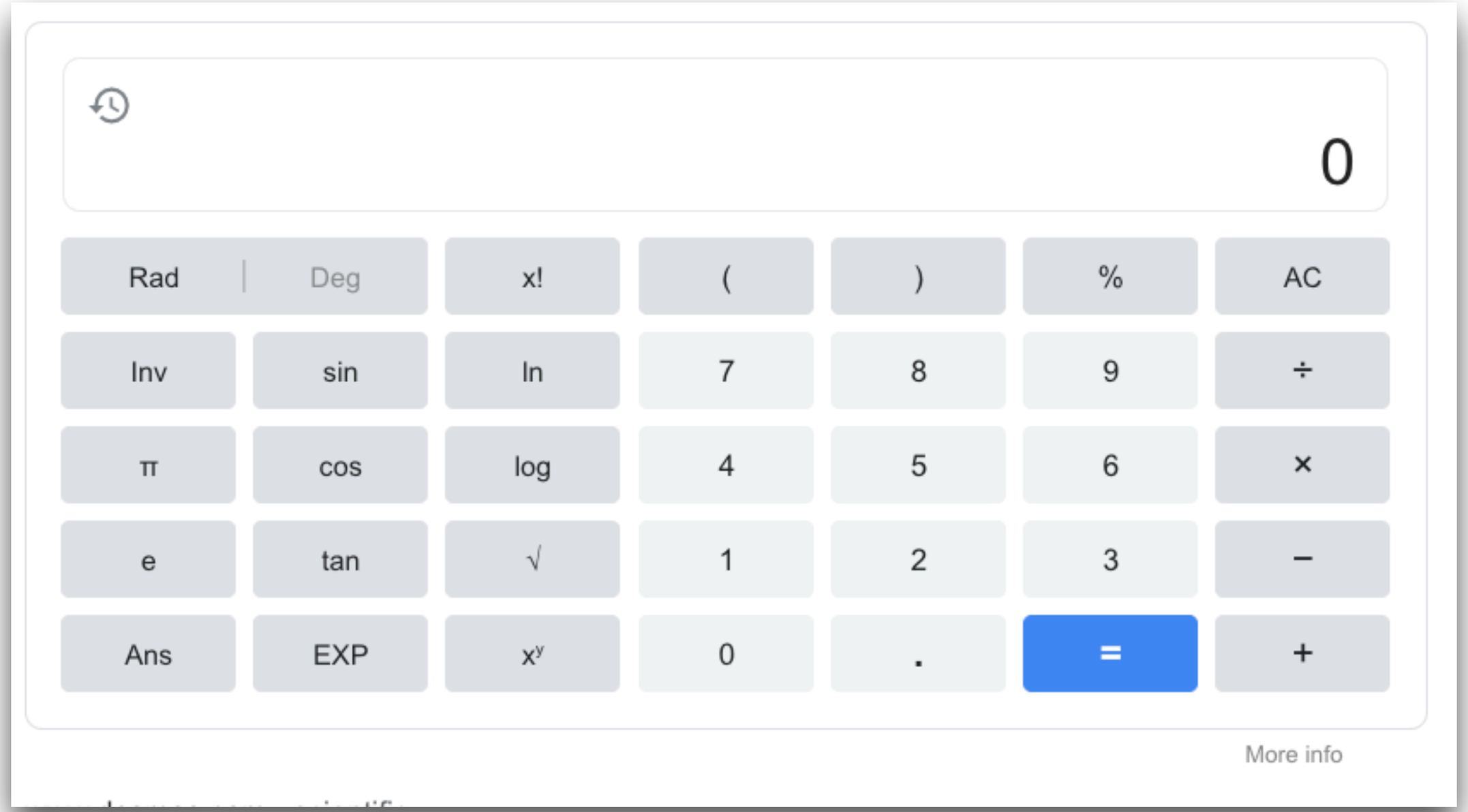
Basic of Python
Object-Oriented Programming

Return value of keyword

Return value of keyword

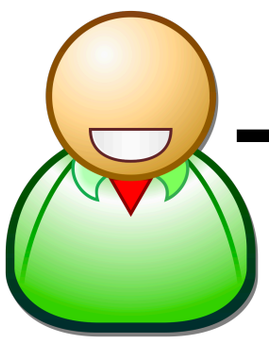
Scalar variables
List variables
Dictionary variables
Environment variables

Calculator library



Try by yourself !!

Calculator library



1.

CalculatorLibrary.py

Push button 1
Push buttons 1 + 2 =
Result 3

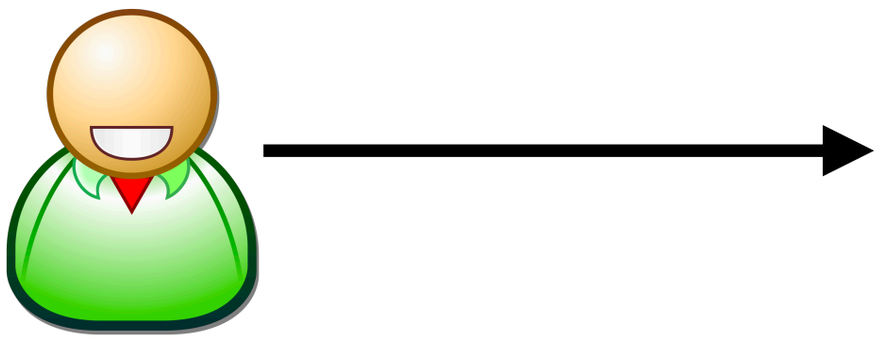
2.

Calculator.py

Push button
(number)
+,-,*,/, =
AC

Working data in CSV files

CRUD data in csv file



employees.csv

List all employees
Add a new employee
Remove all employees
Etc..

Employees.csv

employees.csv

```
firstname,lastname
f01,l01
f02,l02
f03,l03
f04,l04
f05,l05
```

1. List all employees

Working with **csv** and **os** library

```
def list_employees():
    """
    Print the list of stored employees
    :return: None
    """
    employees_list = []
    if os.path.exists(EMPLOYEE_FILE):
        with open(EMPLOYEE_FILE, newline='') as csv_file:
            reader = csv.reader(csv_file, delimiter=',',
quotechar='"')

            for row in reader:
                employees_list.append(' '.join(row))
```

https://docs.python.org/3/library/csv.html

2. Add a new employee

Working with **csv** and **os** library

```
def add_employee(first_name, last_name):
    """
    Adds an employee to the list of employees
    :param first_name: The first name of the employee
    :param last_name: The last name of the employee
    :return: None
    """
    with open(EMPLOYEE_FILE, 'a', newline='') as csv_file:
        writer = csv.writer(csv_file, delimiter=',', quotechar='"',
quoting=csv.QUOTE_NONE)
        writer.writerow([first_name, last_name])
```

3. Remove all employees

Working with **csv** and **os** library

```
def remove_all_employees():
    """
    Remove all employees, the file is removed
    :return: None
    """
    if os.path.exists(EMPLOYEE_FILE):
        os.remove(EMPLOYEE_FILE)
    else:
        print("The file does not exist")
```

Run python code

```
$python employee.py list_employees
$python employee.py add_employee
$python employee.py remove_all_employees
```

Test with Robot Framework

Try01.robot

Test case 1 :: Empty employee

```
*** Settings ***
Documentation    Test the employee with python script
Library    Collections
Library    employee_final.py

*** Test Cases ***
Empty employees list
    [Setup]   Clear employees list
    ${result}=  Get employees list
    Should Be Empty    ${result}
```

Try_01.robot

Test case 2 :: Add a new employee

```
*** Test Cases ***
Add a new employee
    [Setup]   Clear employees list
    employee_final.Add Employee   somkiat  pui
    ${result}=  Get employees list
    ${expected}=  Create List   somkiat pui
    Lists Should Be Equal    ${result}    ${expected}
```

Create custom library in Robot Framework

employee_lib.py

Use robot framework library

```
import csv
import os
import sys
from robot.api.deco import keyword

ROBOT_LIBRARY_VERSION = '0.1'
ROBOT_AUTO_KEYWORDS = False

EMPLOYEE_FILE = 'employees.csv'
```

employee_lib.py

Use decorator @keyword

```
@keyword
def list_employees():
    employees_list = []
    if os.path.exists(EMPLOYEE_FILE):
        with open(EMPLOYEE_FILE, newline='') as csv_file:
            reader = csv.reader(csv_file, delimiter=',',
quotechar='"')

            for row in reader:
                employees_list.append(' '.join(row))

    return employees_list
```

Try_02.robot

```
*** Settings ***
Library    employee_lib.py

*** Test Cases ***
Empty employees list
    [Setup]   Clear employees list
    ${result}=  Get employees list
    Should Be Empty    ${result}

*** Keywords ***
Clear employees list
    employee_lib.Remove all employees

Get employees list
    ${result}=  employee_lib.List employees
    [Return]   ${result}
```

Working with Object-Oriented Programming

EmployeeLibrary.py

Working with OOP

```
from robot.api.deco import keyword
from robot.api.deco import library

@library
class EmployeeLibrary:
    def __init__(self, path='employees.csv'):
        self._path = path

    @keyword
    def list_employees(self):
        pass

    @keyword
    def remove_all_employees(self):
        pass
```

Try_03.robot

```
*** Settings ***
Library    EmployeeLibrary

*** Test Cases ***
Empty employees list
    [Setup]   Clear employees list
    ${result}=  Get employees list
    Should Be Empty    ${result}

*** Keywords ***
Clear employees list
    EmployeeLibrary.Remove all employees

Get employees list
    ${result}=  EmployeeLibrary.List employees
    [Return]   ${result}
```

Run

$robot try_03.robot

```
[ ERROR ] Error in file '/Users/somkiat/data/slide/robotframework/advance-robot-cour
y 'EmployeeLibrary' failed: ModuleNotFoundError: No module named 'EmployeeLibrary'
Traceback (most recent call last):
  None
PYTHONPATH:
  /Library/Frameworks/Python.framework/Versions/3.7/bin
  /Library/Frameworks/Python.framework/Versions/3.7/lib/python37.zip
  /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7
  /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/lib-dynload
  /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages
==============================================================================
```

No module named 'EmployeeLibrary'

Working with PYTHONPATH

$robot --pythonpath <path> try_03.robot

Windows

$set PYTHONPATH=.;%PYTHONPATH%

Linux/Mac

$export PYTHONPATH=.:$PYTHONPATH

http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#module-search-path

Generate document of library

$python -m robot.libdoc EmployeeLibrary EmployeeLibrary.html

Practice about Python

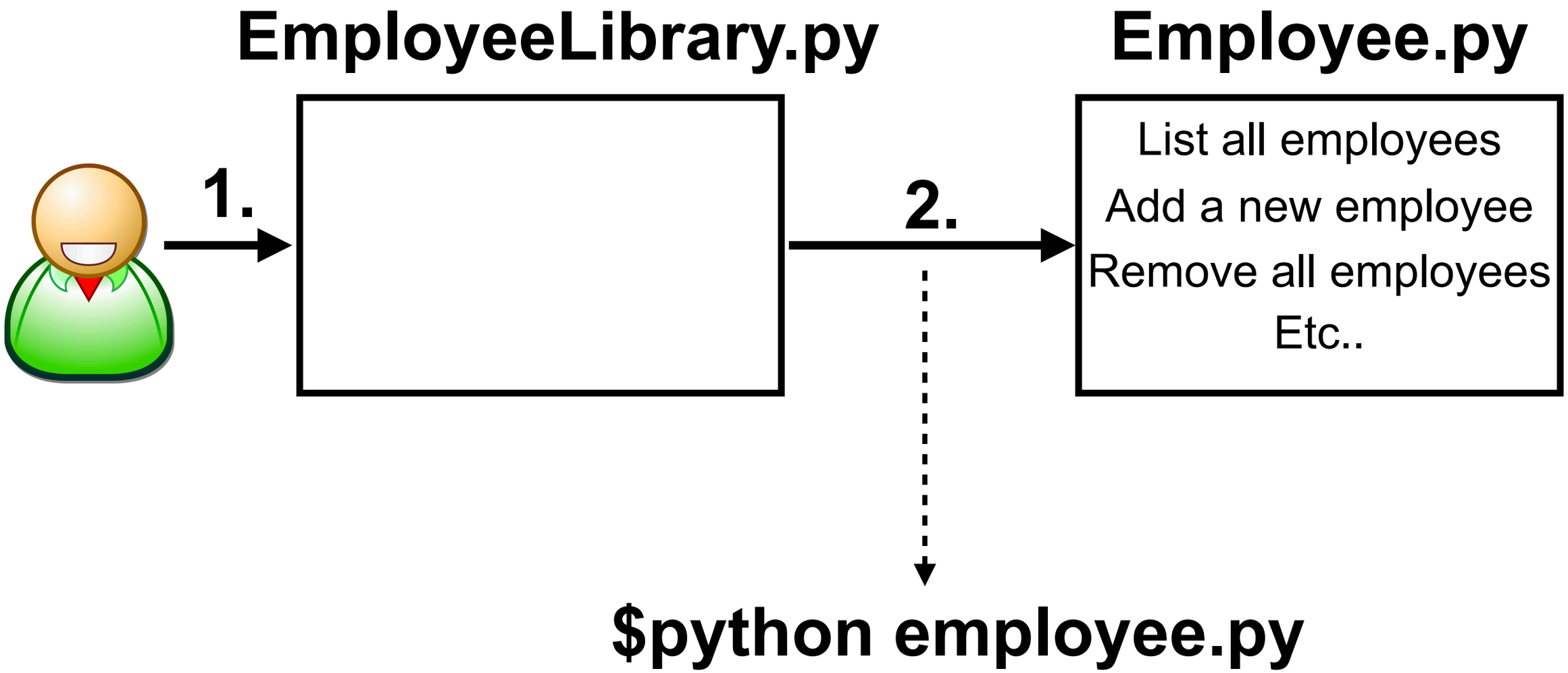
Let’s coding

Quiz 01
Return list of object
Ignore first line in csv

https://docs.python.org/3/library/csv.html

More example !!

Call python in subprocess



Using subprocess library

https://docs.python.org/3/library/subprocess.html

Practice about Python

Let’s coding