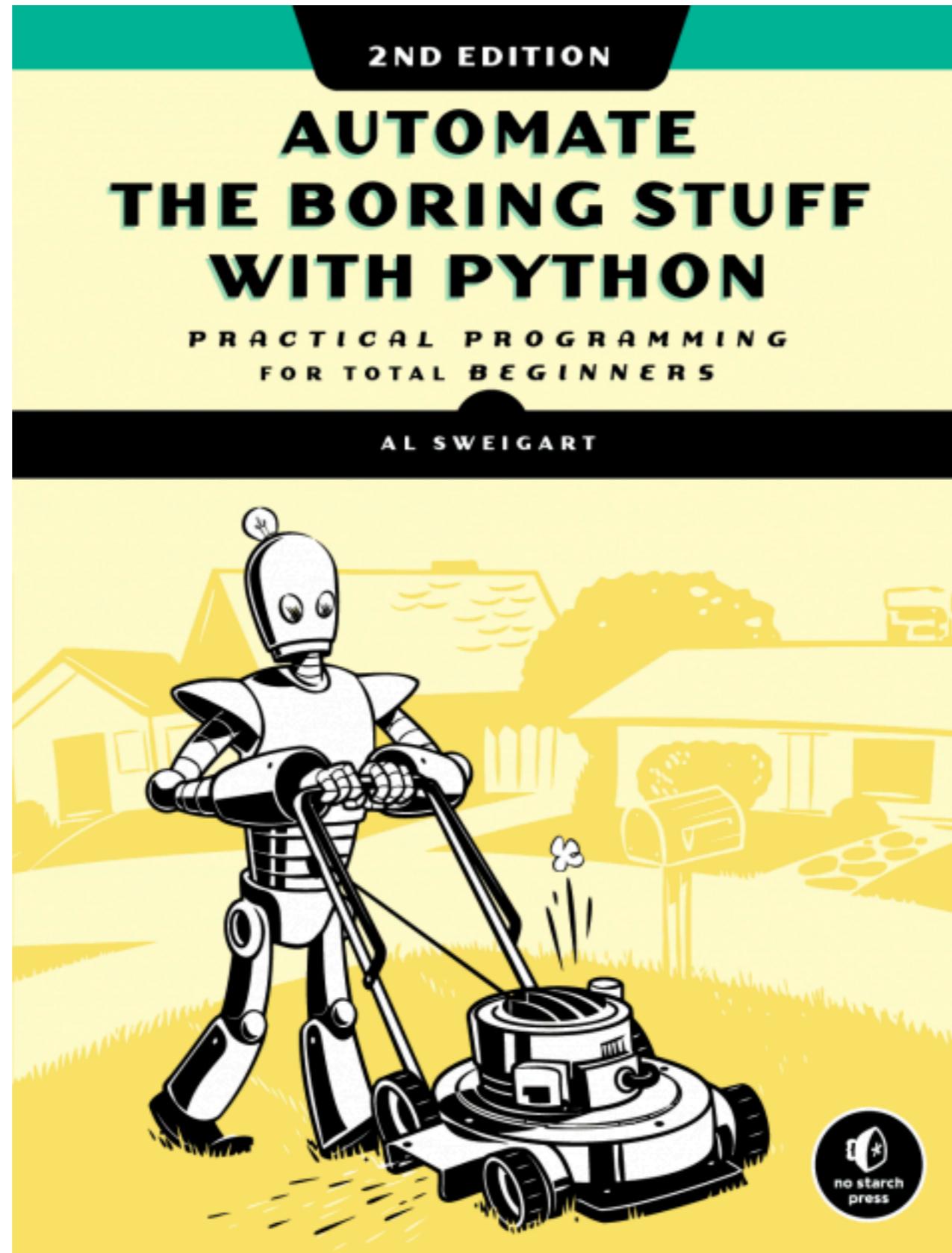


Workshop with tasks







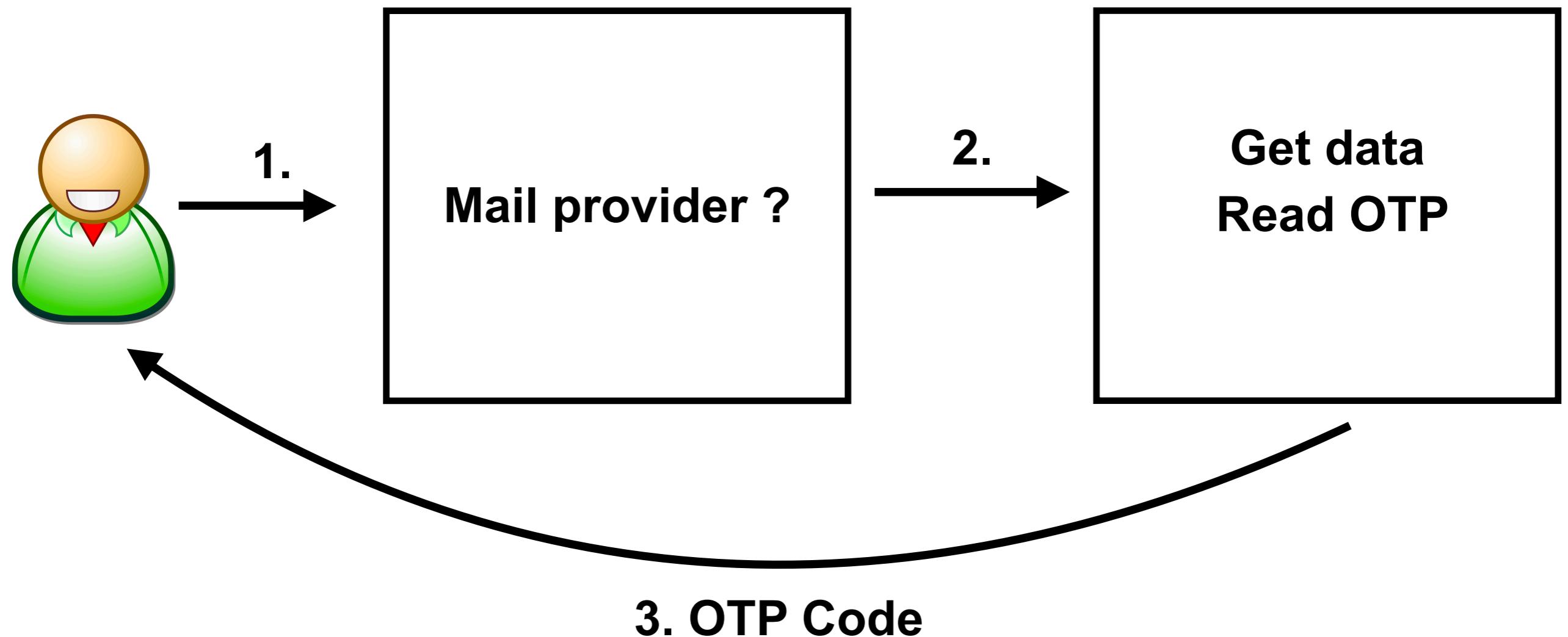
Tasks with Python and Robot Framework



Check OTP from e-mail



Use case



Working with email



Pricing

Docs & API

Blog

Login

Try it free

Email testing for any app, product or website

**The API for automating email tests, works
with frameworks like Cypress and Selenium**

Test user signups, password resets and anything else!

Enter work email

Try it free



<https://mailosaur.com/>



Advance Robot Framework

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Working with email

 **Empty Server**

You do not have anything to test yet...

SEND AN EMAIL TO
soap-disappear.xckfu42e@mailosaur.io 
[Show me another](#)

 This works because any address that ends with `.xckfu42e@mailosaur.io` points to this server. Because `xckfu42e` is your **Server ID**.

CREATE A SAMPLE EMAIL
Just want something to get started with?

[Create a sample email](#)

<https://mailosaur.com/>



Mailosaur library

\$pip install mailosaur

<https://mailosaur.com/docs/email-testing/>



check_email.py

```
from mailosaur import MailosaurClient
from mailosaur.models import SearchCriteria

API_KEY = ''
SERVER_ID = ''

def read_email(email):
    client = MailosaurClient(API_KEY)
    criteria = SearchCriteria()
    criteria.sent_to = email + "." + SERVER_ID + "@mailosaur.io"
    message = client.messages.get(SERVER_ID, criteria)
    return message.text.body
```

<https://mailosaur.com/docs/email-testing/python/>



RPA framework library

\$pip install rpaframework

<https://rpaframework.org/>



check_gmail.py

```
from RPA.Email.ImapSmtp import ImapSmtp

gmail_account = "<your gmail>"
gmail_password = "<your password>"
sender = gmail_account

mail = ImapSmtp(smtp_server="smtp.gmail.com", smtp_port=587)
mail.authorize(account=gmail_account, password=gmail_password)
mail.send_message(
    sender=gmail_account,
    recipients=gmail_account,
    subject="Message from RPA Python",
    body="RPA Python message body",
)

messages = mail.list_messages("SUBJECT RPA")
print(messages[0]['Body'].decode("utf-8"))
```



Config gmail

Enable IMAP
Allow less secure access



Enable IMAP

Settings -> POP/IMAP

Settings

General Labels Inbox Accounts and Import Filters and Blocked Addresses **Forwarding and POP/IMAP** Add-ons

Offline Themes

Forwarding:

[Learn more](#)

Add a forwarding address

Tip: You can also forward only some of your mail by [creating a filter!](#)

POP download:

[Learn more](#)

1. Status: POP is disabled

- Enable POP for **all mail**
- Enable POP for **mail that arrives from now on**

2. When messages are accessed with POP

keep Gmail's copy in the Inbox ▾

3. Configure your email client (e.g. Outlook, Eudora, Netscape Mail)

[Configuration instructions](#)



Allow less secure access

<https://myaccount.google.com/u/2/lesssecureapps>

← Less secure app access

Some apps and devices use less secure sign-in technology, which makes your account vulnerable. You can turn off access for these apps, which we recommend, or turn it on if you want to use them despite the risks. Google will automatically turn this setting OFF if it's not being used. [Learn more](#)

Allow less secure apps: ON



Working with Excel file



RPA framework library

\$pip install rpaframework

Using Excel.Files

https://rpaframework.org/libraries/excel_files/index.html



Read data from Excel file

```
from RPA.Excel.Files import Files

def read_excel_worksheet(path, worksheet):
    lib = Files()
    lib.open_workbook(path)
    try:
        return lib.read_worksheet(worksheet)
    finally:
        lib.close_workbook()

if __name__ == '__main__':
    data1 = read_excel_worksheet('sample.xlsx', 'Sheet1')
    print(data1)
    data2 = read_excel_worksheet('sample.xlsx', 'Sheet2')
    print(data2)
```



Using pandas, xlrd, openpyxl

```
$pip install pandas  
$pip install xlrd openpyxl
```



Read data from Excel file

```
import pandas as pd

# Read File
sample1 = pd.read_excel("sample.xlsx", sheet_name="Sheet1")
print(sample1)

sample2 = pd.read_excel("sample.xlsx", sheet_name="Sheet2")
print(sample2)
```

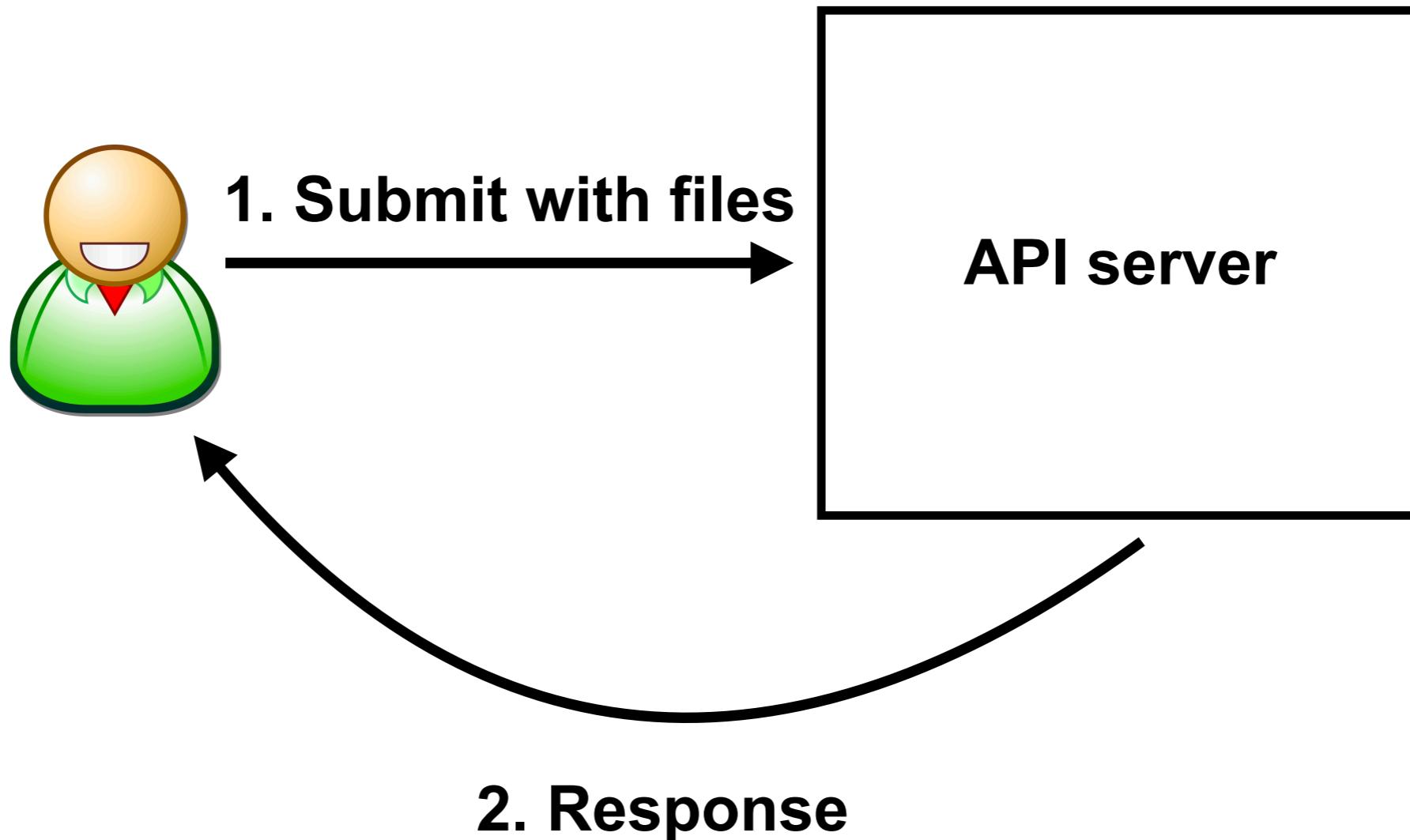


API testing

Upload file with multipart/ request



Use case



Working with RequestsLibrary

\$pip install robotframework-requests

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



Upload file to server

*** Settings ***

Library RequestsLibrary
Library OperatingSystem
Library Collections

*** Test Cases ***

Post Request With File

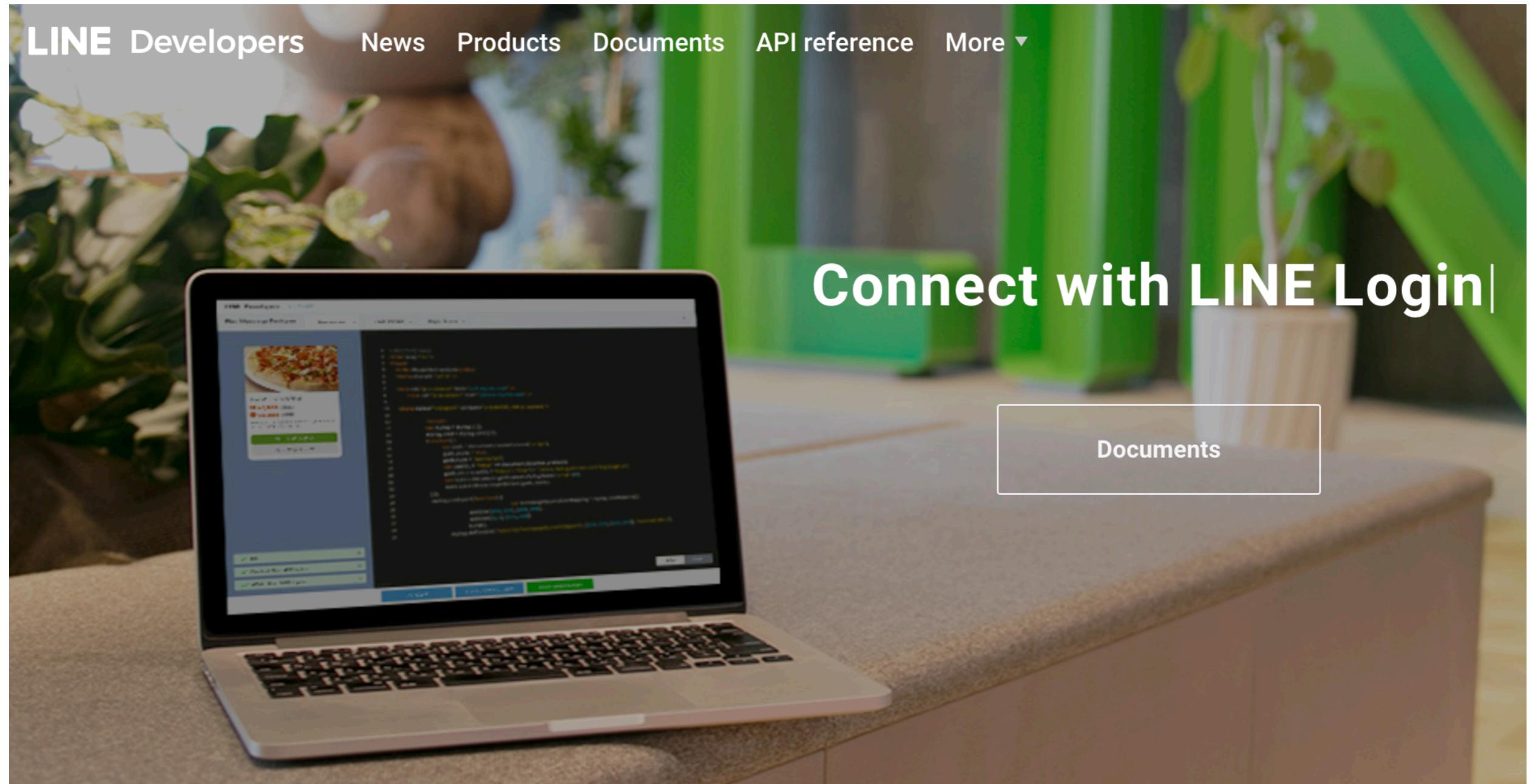
```
Create Session    httpbin    http://httpbin.org    debug=2
max_retries=10
${file_data}=  Get Binary File  ${CURDIR}${/}data.json
${files}=  Create Dictionary  file  ${file_data}
${resp}=  Post Request  httpbin  /post  files=${files}
Log To Console  ${resp.json()}
${file}=  To Json  ${resp.json()['files']['file']}
Dictionary Should Contain Key  ${file}  data1
Dictionary Should Contain Key  ${file}  data2
```



Read data from LINE <chatbot>



Create LINE Developers



Connect with LINE Login|

Documents

<https://developers.line.biz/console/>



Create providers and channel

The screenshot shows the LINE Developers console interface. The top navigation bar includes links for Products, Documents, API reference, News, FAQ, Community, and Blog. On the left, a sidebar titled 'Providers' contains a search bar and a list with 'demo' and 'Admin'. The main content area shows the 'demo' provider details, including a placeholder image, the role 'Admin', and the 'Messaging API' icon. A navigation bar below the provider details includes tabs for Basic settings, Messaging API (which is active), LIFF, Security, Statistics, and Roles. The 'Messaging API settings' section displays the bot's basic ID (@538txndm) and a QR code.

<https://developers.line.biz/console/>



Generate channel secret

Channel secret ⓘ	5fb0f871f95090eff48bc087ff93346b
Assertion Signing Key ⓘ	741caa2f-f5cb-4d95-baf5-5286fd1a6f5e
	Delete
	Issue

<https://developers.line.biz/console/>



Working with LINE Bot SDK

```
$pip install line-bot-sdk
```

<https://github.com/line/line-bot-sdk-python>



Sample server

\$pip install Flask

<https://pypi.org/project/Flask/>



Sample server

Config channel secret and access token

```
# get channel_secret and channel_access_token from your environment
variable
channel_secret = os.getenv('LINE_CHANNEL_SECRET', None)
channel_access_token = os.getenv('LINE_CHANNEL_ACCESS_TOKEN', None)
if channel_secret is None:
    print('Specify LINE_CHANNEL_SECRET as environment variable.')
    sys.exit(1)
if channel_access_token is None:
    print('Specify LINE_CHANNEL_ACCESS_TOKEN as environment variable.')
    sys.exit(1)
```



Sample server

Create callback for webhook

```
@app.route("/callback", methods=['POST'])
def callback():
    signature = request.headers['X-Line-Signature']

    # get request body as text
    body = request.get_data(as_text=True)
    app.logger.info("Request body: " + body)

    # parse webhook body
    events = parser.parse(body, signature)

    # if event is MessageEvent and message is TextMessage, then echo text
    for event in events:
        if not isinstance(event, MessageEvent):
            continue
        if not isinstance(event.message, TextMessage):
            continue

        print('">>>>', event.message.text)

    return 'OK'
```



Config webhook in Message API

TOP > demo > demo > Messaging API

Scan this QR code with LINE to add your LINE Official Account

Available APIs ⓘ

- REPLY_MESSAGE
- PUSH_MESSAGE

Webhook settings

Webhook URL ⓘ

<https://f1c8d90e96ec.ngrok.io/callback>

Verify

Edit

Use webhook ⓘ



<https://developers.line.biz/console/>



Advance Robot Framework

© 2017 - 2018 Siam Chamnkit Company Limited. All rights reserved.

Publish localhost to public server

Using ngrok

The screenshot shows the ngrok homepage. At the top, there is a navigation bar with links for "How it works", "Pricing", "Enterprise solutions", "Docs", "Download", "Login", and "Sign up". The main heading is "Public URLs for SSH acces". Below this, a sub-section highlights "Spend more time programming. One command for an instant, secure URL to your localhost server through any NAT or firewall." A blue button labeled "Get started for free" is visible. To the right, a browser window shows a secure connection to "https://katesapp.ngrok.io" with the message "Welcome to Kate's Site! It's currently under development...". Below the browser is a terminal window showing the command "ngrok http 3000" and its output, which includes session details like "Status online", account information ("Account Kate Libby (Plan: Pro)", "Web Interface http://127.0.0.1:4040"), and forwarding rules ("Forwarding http://katesapp.ngrok.io -> localhost", "Forwarding https://katesapp.ngrok.io -> localhost").

<https://ngrok.com/>



Sample server

\$ngrok http 8080

Region	United States (us)
Web Interface	http://127.0.0.1:4040
Forwarding	http://f1c8d90e96ec.ngrok.io -> http://localhost:8000
Forwarding ngrok by @inconshreveable	https://f1c8d90e96ec.ngrok.io -> http://localhost:8000 (Ctrl+C to quit)

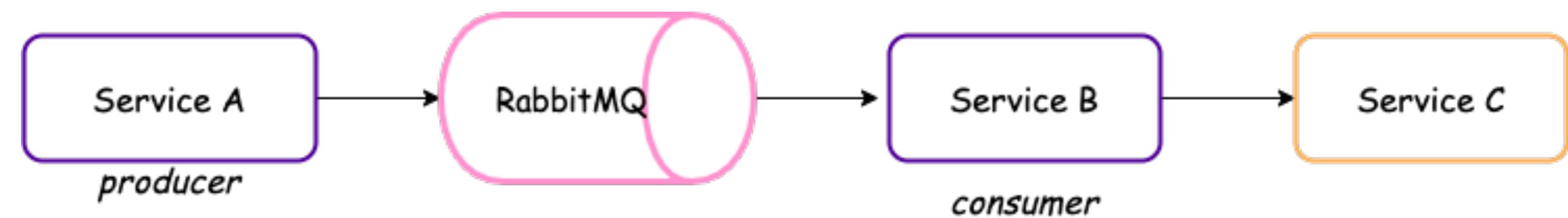


Get data from RabbitMQ



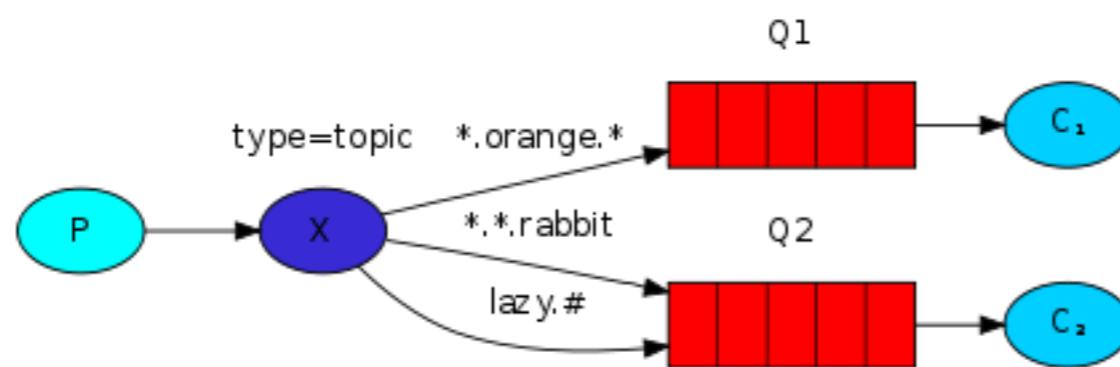
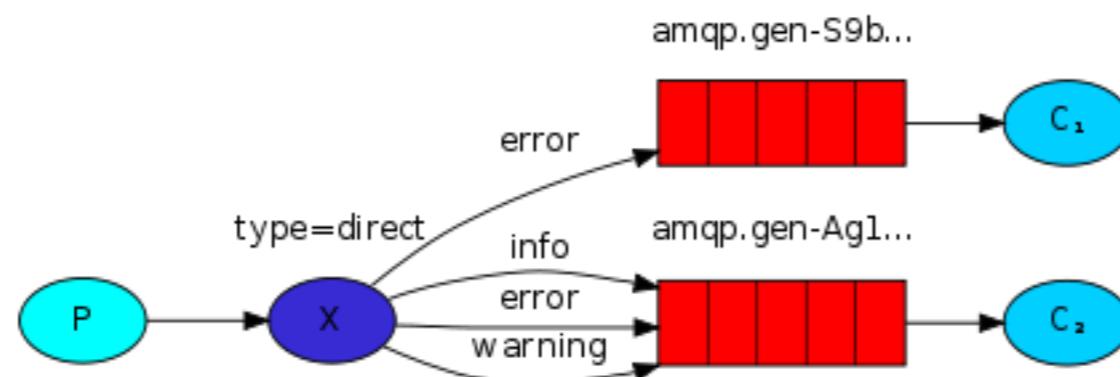
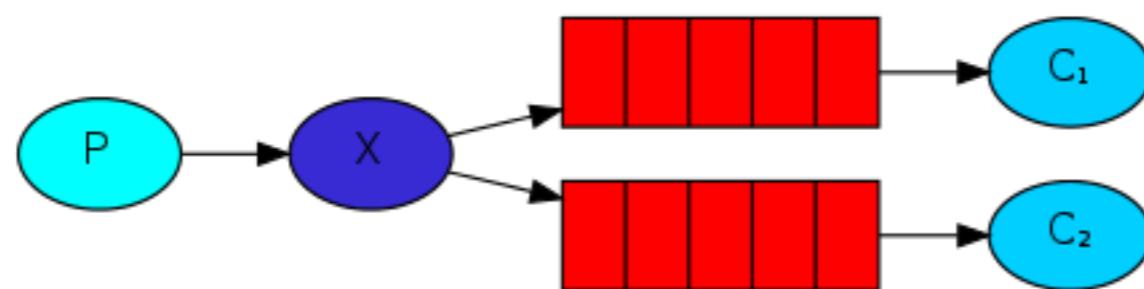
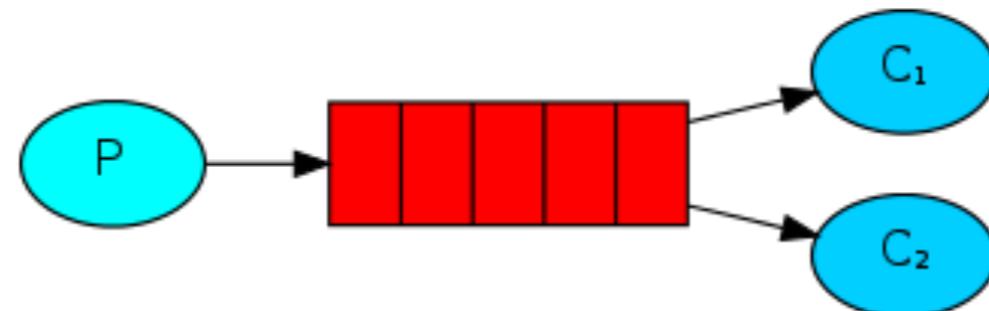
<https://www.rabbitmq.com/getstarted.html>





<https://www.rabbitmq.com/getstarted.html>





Working with RabbitMQ

```
$pip install robotframework-rabbitmq
```

<https://github.com/peterservice-rnd/robotframework-rabbitmq>



Sample robot

*** Settings ***

Library RabbitMq
Library Collections

*** Variables ***

`${SERVER} 139.59.246.17`
 `${USER} admin`
 `${PASSWORD} xitgmLwmp`

*** Test Cases ***

Simple Test

```
Create Rabbitmq Connection  ${SERVER}  15672  5672
...  ${USER}      ${PASSWORD}      alias=rmq    vhost=/
${overview}=    Overview
Log Dictionary    ${overview}
Close All Rabbitmq Connections
```



Compare Image



<https://imagemagick.org/index.php>



Practice about Python

Let's coding

