Sikuli in Robot Framework

Contents

[1. Introduction 2](#_Toc196325028)

[1.1 What is Sikuli? 2](#_Toc196325029)

[1.2 Difference Robot Framework vs Sikuli Library 2](#_Toc196325030)

[1.3 Why Use Sikuli? 4](#_Toc196325031)

[1.4 Robot Framework vs Sikuli 5](#_Toc196325032)

[2. Install Sikuli 6](#_Toc196325033)

# 1. Introduction

# 1.1 What is Sikuli?

Sikuli is an image-based GUI automation tool or visual automation tool. Selenium Work with DOM element or Html, Sikuli automates anything you see on the screen like desktop apps, browser, UI (user interface), games, PDFs --- Using Screenshots.

Instead of accessing elements via code like Selenium does with HTML

Sikuli finds element based on screenshot (images)

java -Duser.language=en -Duser.country=US -jar C:\Users\WELCOME\Downloads\sikulixide-2.0.5.jar

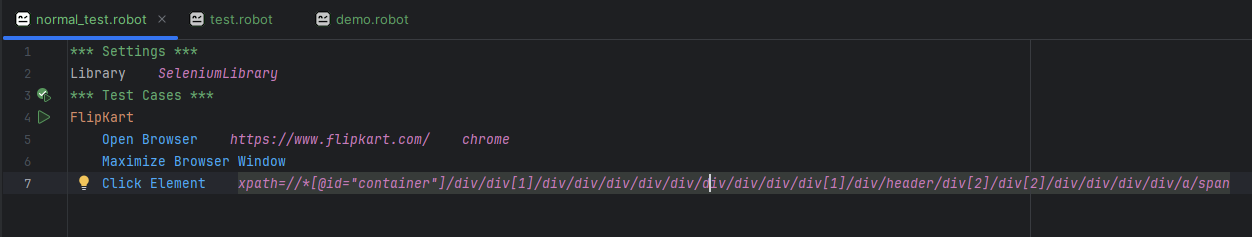
sikuliX idl open

Example

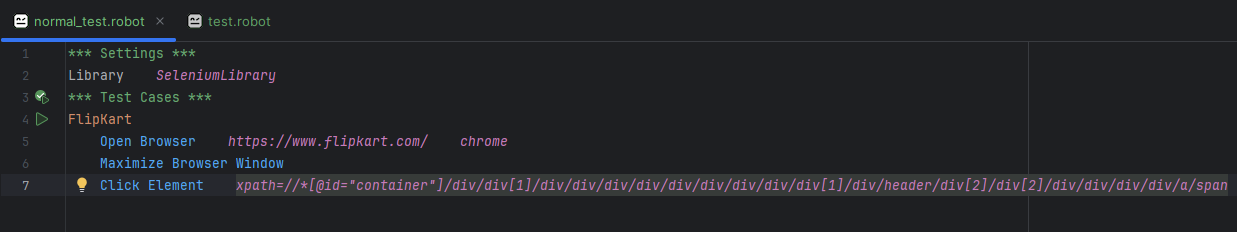
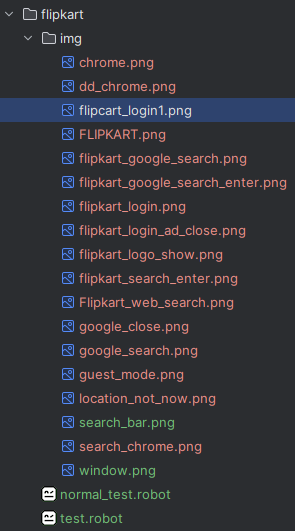
If you take a flipkart website to Login

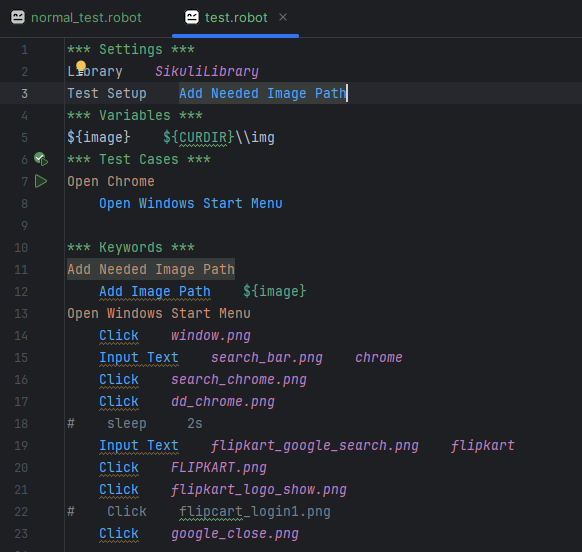
# 1.2 Difference Robot Framework vs Sikuli Library

In Normal Robot Framework



In Sikuli





| **Feature** | **Sikuli** | **Selenium** |
| --- | --- | --- |
| Based on | Image Recognition | HTML DOM |
| Platform | Desktop/Web (via screen) | Web apps only |
| Speed | Slower | Faster |
| Use case | Games, Flash, Desktop apps | Web testing |

Example: If you take a screenshot of a “Login” button (login\_button.png), sikuli can find and click it on the screen.

# 1.3 Why Use Sikuli?

Sikuli is perfect when other tools can’t access the UI element directly.

| **Tool** | **Language** | **Image Matching** | **Best For** |
| --- | --- | --- | --- |
| SikuliX | Java/Python | ✅ Yes | GUI/Desktop/Hybrid apps |
| PyAutoGUI | Python | ✅ Yes | Simple automation, cross-platform |
| AutoIt | AutoIt Script | ✅ Yes (basic) | Windows desktop automation |
| OpenCV | Python/C++ | ✅ (manual) | Custom vision detection |
| TestComplete | Script-based | ✅ Yes | Enterprise-grade GUI testing |
| Eggplant | Proprietary | ✅ Yes | Enterprise cross-platform |

AutoItLibrary, WhiteLibraary, Pywinauto + Robot Framework

When Sikuli Use

* Desktop Applications

No DOM in web apps(id, class, tag)

Sikuli can see UI via screenshot

* Game automation

No standard element accesseverything is graphical

* Image-based verification

Handle image

Why Use Robot Framework with sikuli?

Robot Framework is a keyword-driven test automation framework. It makes test scripts readable and easy to organize.

Why combine it with sikuli?

Sikuli by itself image-based scripting (Java/Python)

\*\*\*Test Cases\*\*\*

Click Image

Click login.png

# 1.4 Robot Framework vs Sikuli

| **Feature** | **Robot Framework** | **Sikuli** | **Best For** |
| --- | --- | --- | --- |

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Test automation framework | Visual automation  tool | Combine both! |

|  |  |  |  |
| --- | --- | --- | --- |
| Web Testing | ✅ (via Selenium) | ❌ | Use Robot + Selenium |

|  |  |  |  |
| --- | --- | --- | --- |
| Desktop (complex) | ✅ (via WhiteLibrary or Sikuli) | ✅ | Use Robot + Sikuli |

|  |  |  |  |
| --- | --- | --- | --- |
| Image-based | ❌ | ✅ | Sikuli wins here |

|  |  |  |  |
| --- | --- | --- | --- |
| Structure & Reports | ✅ | ❌ | Robot wins here |

Summary for Sikuli

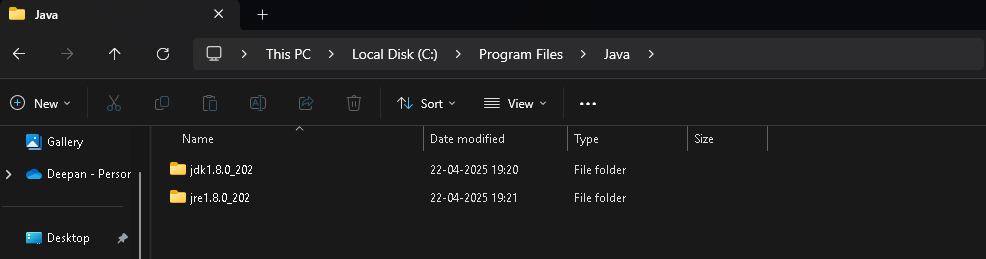
* When you need to automate image-based GUIs
* When don’t exist traditional locator (id, path) we use sikuli image based process

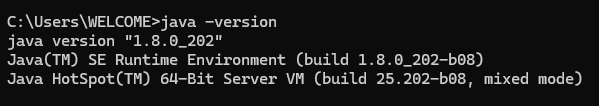
# 2. Install Sikuli

pip install robotframework-sikulilibrary

Install Java

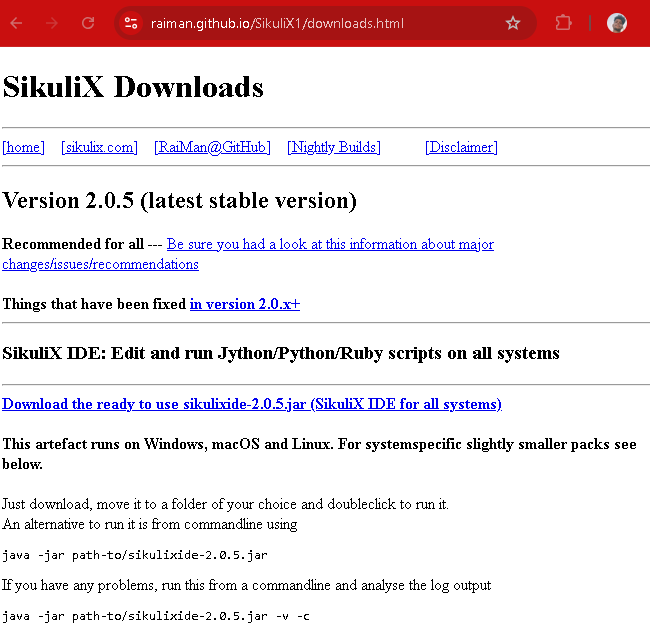






Download SikuliX

Go To SikuliX official download



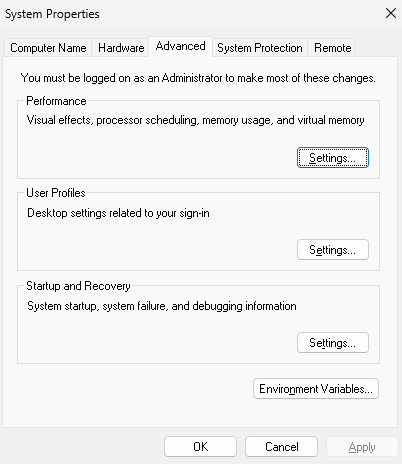
Download

**sikulixide-2.0.5.jar** or latest version

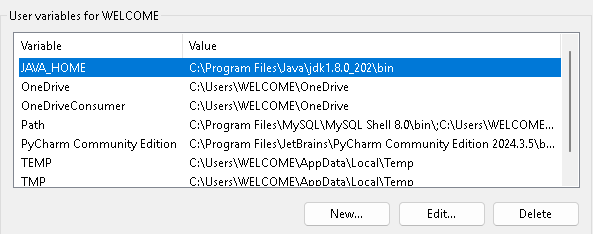
Win+s

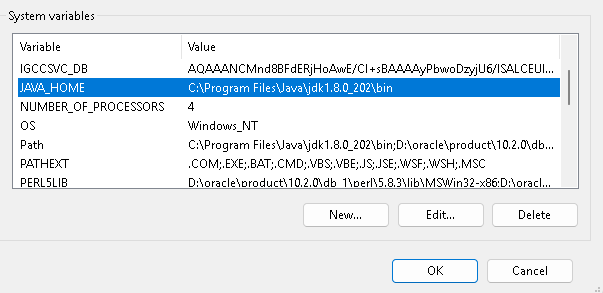
Edit the system environment variable

Click Environment Variable



File path add or create in Both JAVA\_HOME and PATH





Click OK.

# 3. Basic Keywords or Commands in SikuliLibrary

These commands are used in .robot files when you include Library SikuliLibrary

1. Set Image Path

Sets the folder where your images (screenshot) are stored.

2. Click Image

Clicks on the image on the screen

3. Wait For Image

Waits until the image appears on the screen

4. Type Text

Types text into the currently focused input field

5. Double Click Image

6. Paste Text

7. Image Should Exists / Not Exist

Verify if image is found

8. Hover Image

Move mouse pointer over the images

9. Drag and drop

10. Capture Screenshot

11. Set Capture Folder

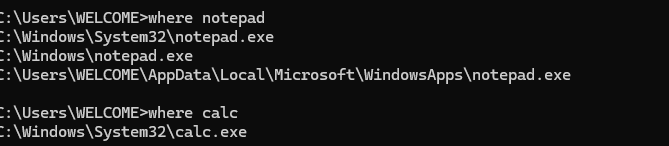
### 12. Pro Tip: Use Relative Paths

To keep your test project portable:

robot

Copy code

Set Image Path ${CURDIR}/images



\*\*\* Settings \*\*\*  
Library *SikuliLibrary*Library *Process*\*\*\* Variables \*\*\*  
${calc} *C://Windows//System32//calc.exe*${task} *C://Windows//System32//taskkill.exe*\*\*\* Test Cases \*\*\*  
Add Two Numbers In Calculator  
#directly give the path clac.exe not work to give dynamically in variable section  
 Open Application ${calc}  
 Sleep *5s* Click *D://Program//SIKULI//SikuliX//Pratice//Calculator//images//eight.png* Sleep *5s* Click *D://Program//SIKULI//SikuliX//Pratice//Calculator//images//plus.png* Click *D://Program//SIKULI//SikuliX//Pratice//Calculator//images//eight.png* Click *D://Program//SIKULI//SikuliX//Pratice//Calculator//images//equal.png* Sleep *2s* Run Process ${task} */f /im calc.exe*

Main Keyword

#### 🖱️ Interaction Keywords

* Click
* Double Click
* Right Click
* Mouse Down
* Mouse Up
* Hover
* Drag And Drop

#### ⌨️ Typing and Input

* Type
* Paste
* Input Text

#### 🔍 Searching and Waiting

* Wait Until Screen Contains
* Wait Until Screen Does Not Contain
* Wait
* Exists
* Find
* Region Exists

#### 🖼️ Screenshot and Region

* Capture Screenshot
* Set Screenshot Directory
* Get Region
* Highlight

#### 🔁 Utility and Settings

* Sleep
* Set Bundle Path
* Set OCR Text Read
* Set Wait Time After Action