

Tutorial de Sikuli para Selenium Automatización

¿En qué está Sikuli? Selenium?

Sikuli es una herramienta de automatización de pruebas basada en GUI de código abierto. Se utiliza principalmente para interactuar con elementos de páginas web y manejar ventanas emergentes. Sikuli utiliza la técnica de “Reconocimiento de imágenes” y “GUI de control” para interactuar con elementos de páginas web y ventanas emergentes. En Sikuli, todos los elementos web se toman como imágenes y se almacenan dentro del proyecto.

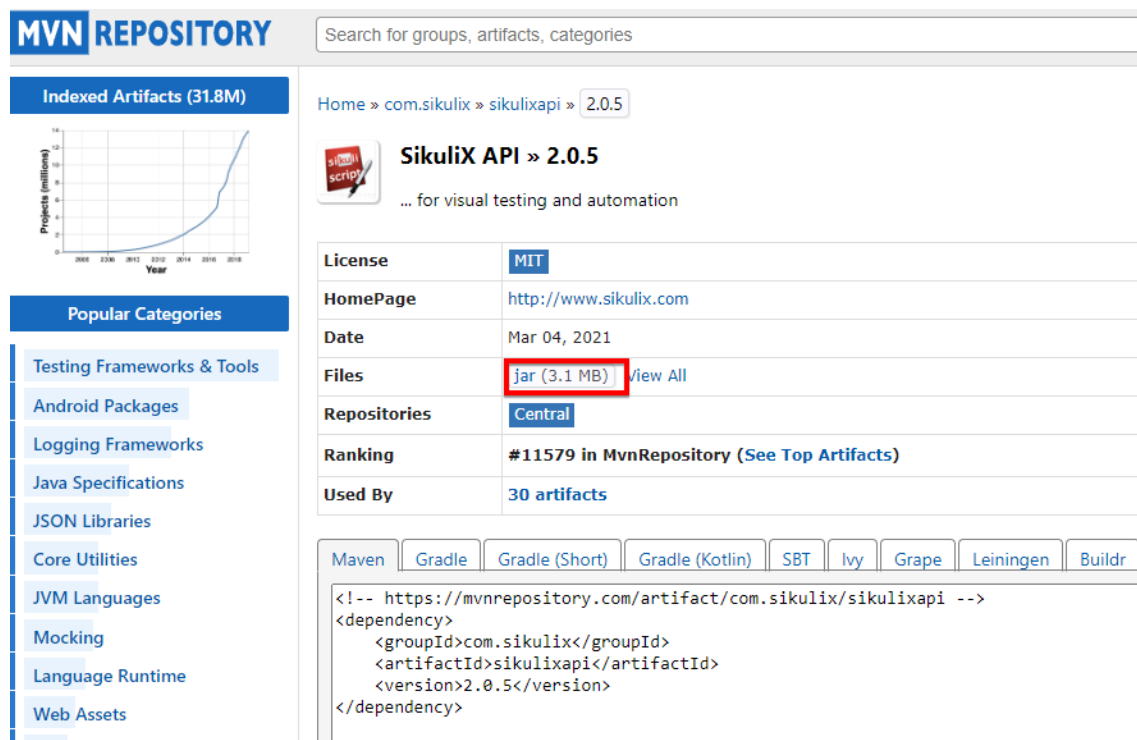
Cómo utilizar Sikuli con Selenium controlador web

Sikuli se puede integrar con Selenium webdriver utilizando el archivo JAR de Sikuli.

La siguiente secuencia es la lista de pasos para configurar Sikuli con Selenium webdriver.

Paso 1) Descargar el archivo JAR de Sikuli desde la siguiente URL y extraiga el contenido del archivo ZIP a una carpeta.

<https://mvnrepository.com/artifact/com.sikulix/sikulixapi/2.0.5>



Indexed Artifacts (31.8M)

Home » com.sikulix » sikulixapi » 2.0.5

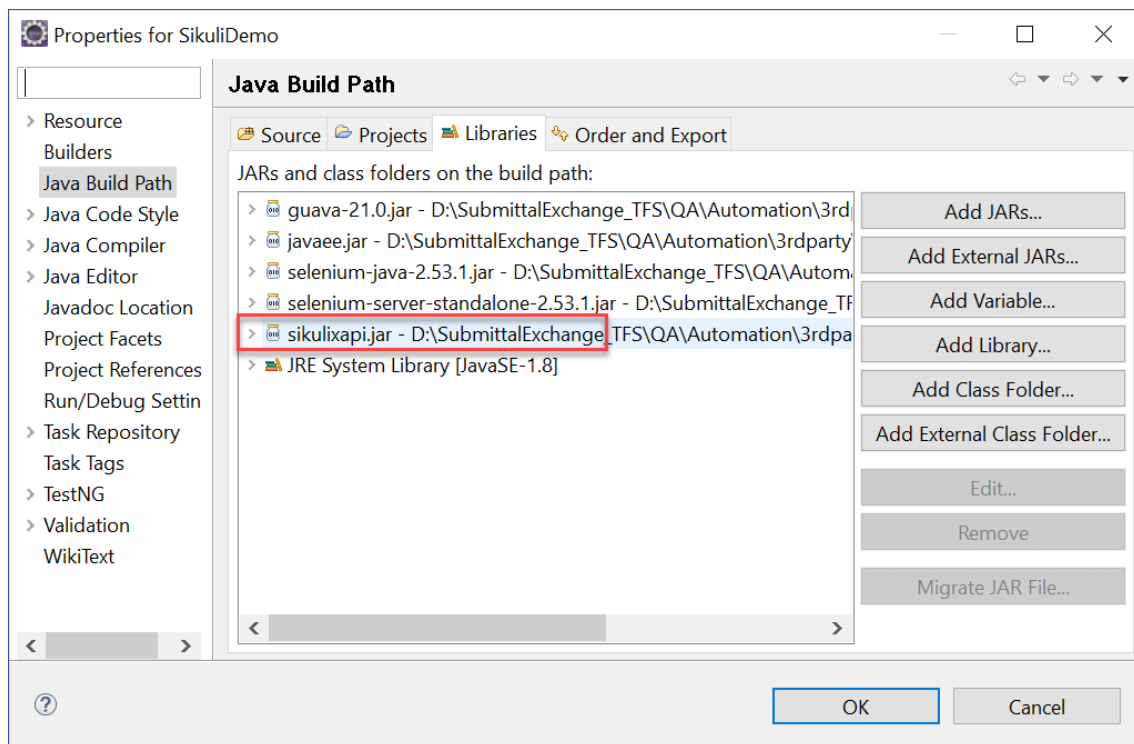
SikuliX API » 2.0.5
... for visual testing and automation

License	MIT
HomePage	http://www.sikulix.com
Date	Mar 04, 2021
Files	jar (3.1 MB) View All
Repositories	Central
Ranking	#11579 in MvnRepository (See Top Artifacts)
Used By	30 artifacts

Maven Gradle Gradle (Short) Gradle (Kotlin) SBT Ivy Grape Leiningen Buildr

```
<!-- https://mvnrepository.com/artifact/com.sikulix/sikulixapi -->
<dependency>
  <groupId>com.sikulix</groupId>
  <artifactId>sikulixapi</artifactId>
  <version>2.0.5</version>
</dependency>
```

Paso 2) Cree un nuevo proyecto JAVA en Eclipse y agregue el archivo JAR a la ruta de compilación, junto con los archivos jar de selenium usando clic derecho en el proyecto -> Ruta de compilación -> Configurar ruta de compilación



Una vez que haya agregado el archivo JAR a la ruta de compilación del proyecto, se pueden usar las clases proporcionadas por Sikuli.

Clase de pantalla en Sikuli

Método	DESCRIPCIÓN	Sintaxis
Haga clic	Este método se utiliza para hacer clic en un elemento en la pantalla usando el nombre de la imagen como parámetro.	<code>Pantalla s = nueva pantalla(); s.click("QA.png");</code>
doble clic	Este método se utiliza para hacer doble clic en un elemento. Acepta el nombre de la imagen como parámetro.	<code>Pantalla s = nueva pantalla(); s.doubleClick("QA.png");</code>
Tipo	Este método se utiliza para proporcionar valor de entrada a un elemento. Acepta el nombre de la imagen y el texto que se enviará como parámetros.	<code>s.type("QA.png","TEXTO");</code>
Hover	Este método se utiliza para pasar el cursor sobre un elemento. Acepta el	<code>s.hover("QA.png");</code>

Método	DESCRIPCIÓN	Sintaxis
	nombre de la imagen como parámetro.	
Encuentre	Este método se utiliza para encontrar un elemento específico en la pantalla. Acepta el nombre de la imagen como parámetro.	s.find("QA.png");

La clase Screen es la clase base para todos los métodos proporcionados por Sikuli. La clase Screen contiene métodos predefinidos para todas las operaciones que se realizan comúnmente en los elementos de la pantalla, como hacer clic, hacer doble clic, proporcionar entrada a un cuadro de texto, pasar el mouse sobre un elemento, etc. A continuación, se incluye una lista de los métodos que se utilizan comúnmente en la clase Screen.

Clase de patrón en Sikuli

La clase de patrón se utiliza para asociar el archivo de imagen con atributos adicionales para identificar de forma única el elemento. Toma la ruta de la imagen como parámetro.

Patrón p = nuevo Patrón("Ruta de la imagen");

Los siguientes son los métodos más comúnmente utilizados de la clase Patrón.

Método	DESCRIPCIÓN	Sintaxis
obtener nombre de archivo	Devuelve el nombre del archivo contenido en el objeto Patrón.	Patrón p = nuevo patrón("D:\Demo\QA.png"); Nombre de archivo de cadena = p.GetFileName();
similares	Este método devuelve un nuevo objeto Pattern con similitud establecida en un valor específico. Acepta el valor de similitud entre 0 y 1 como parámetro. Sikuli busca todos los elementos que se encuentran dentro del rango de similitud especificado y devuelve un nuevo objeto de patrón.	Patrón p1 = p.similar(0.7f);
Exacto	Este método devuelve un nuevo objeto de patrón con similitud establecida en 1. Solo busca una coincidencia exacta del elemento especificado.	Patrón p1 = p.exact();

Ejemplo de código para cargar archivos usando Sikuli

El siguiente código explica el uso de Sikuli para cargar archivos en Firefox.

```

package com.sikuli.demo;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.sikuli.script.FindFailed;
import org.sikuli.script.Pattern;
import org.sikuli.script.Screen;

import org.openqa.selenium.chrome.ChromeDriver;

public class SikuliDemo {

    public static void main(String[] args) throws FindFailed {

        System.setProperty("webdriver.chrome.driver", "D:\\chromedriver.exe");
        String filepath = "D:\\Guru99Demo\\Files\\";
        String inputFilePath = "D:\\Guru99Demo\\Files\\";
        Screen s = new Screen();
        Pattern fileInputTextBox = new Pattern(filepath + "FileTextBox.PNG");
        Pattern openButton = new Pattern(filepath + "OpenButton.PNG");
        WebDriver driver;

        // Open Chrome browser
        driver = new ChromeDriver();
        driver.get("https://demo.guru99.com/test/image_upload/index.php");

        // Click on Browse button and handle windows pop up using Sikuli
        driver.findElement(By.xpath("//*[ @id='photoimg']")).click();
        s.wait(fileInputTextBox, 20);
        s.type(fileInputTextBox, inputFilePath + "Test.docx");
        s.click(openButton);

        // Close the browser
    }
}

```

```

        driver.close();
    }
}

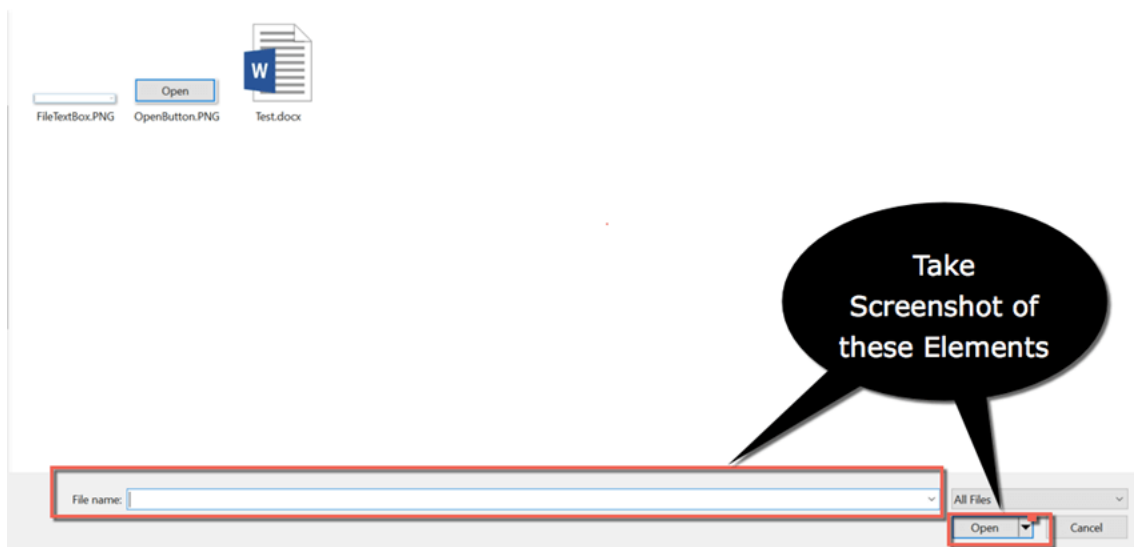
```

Explicación del código:

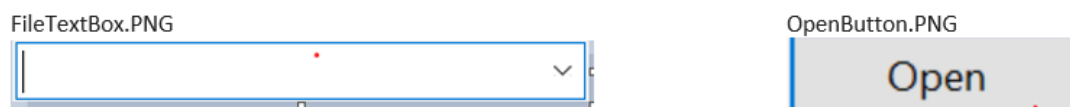
Paso 1) La primera declaración implica configurar la ruta ejecutable del controlador para Chrome.

```
System.setProperty("webdriver.chrome.driver", "D:\\ chromedriver.exe");
```

Paso 2) Utilice una herramienta de captura de pantalla como Snipping Tool para tomar capturas de pantalla de la ventana emergente 'FileText'Box' y botón 'Abrir'.



Así es como debería verse tu captura de pantalla: –



Las imágenes para el cuadro de texto de entrada de archivos de Windows y el botón Abrir se almacenan en 'FileText'Box.PNG' y 'OpenButton.PNG'.

Sikuli utiliza la técnica de Reconocimiento de Imágenes para reconocer elementos en la pantalla. Encuentra elementos en la pantalla basándose únicamente en sus imágenes.

Ejemplo: si desea automatizar la operación de apertura del Bloc de notas, debe almacenar la imagen de un icono del escritorio para el Bloc de notas en un archivo PNG y realizar una operación de clic en él.

En nuestro caso, reconoce el cuadro de texto de entrada del archivo y abre el botón Windows ventana emergente que utiliza las imágenes almacenadas. Si la resolución de la pantalla cambia desde la captura de la imagen hasta la ejecución del script de prueba, el comportamiento de Sikuli sería inconsistente. Por lo tanto, siempre es recomendable ejecutar el script de prueba con la misma resolución con la que se capturan las imágenes.

El cambio en el tamaño de píxel de las imágenes hará que Sikuli lance una excepción FindFailed.

Paso 3) Las siguientes declaraciones incluyen la creación de objetos para las clases Pantalla y Patrón. Crea un nuevo objeto de pantalla. Establezca la ruta del archivo que desea cargar como parámetro para el objeto Patrón.

```
Screen s = new Screen();
```

```
Pattern fileInputTextBox = new Pattern(filepath + "FileTextBox.PNG");
```

```
Pattern openButton = new Pattern(filepath + "OpenButton.PNG");
```

Paso 4) Las siguientes declaraciones implican abrir el navegador Chrome con la URL: https://demo.guru99.com/test/image_upload/index.php

```
driver = new ChromeDriver();
```

```
driver.get("https://demo.guru99.com/test/image_upload/index.php");
```

La URL anterior es una aplicación de demostración para demostrar la funcionalidad de carga de archivos.

Paso 5) Haga clic en el botón elegir archivo usando la siguiente declaración

```
driver.findElement(By.xpath(".*[@id='photoimg']")).click();
```

Paso 6) Espere a que aparezca la ventana emergente. El método de espera se utiliza para controlar la demora asociada con la apertura de la ventana emergente después de hacer clic en el botón Explorar.

```
s.wait(fileInputTextBox, 20);
```

Paso 7) Escriba la ruta del archivo en el cuadro de texto del archivo de entrada y haga clic en el botón Abrir

```
s.type(fileInputTextBox, inputFilePath + "Test.docx");
```

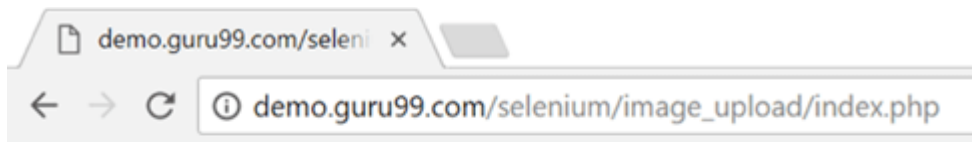
```
s.click(openButton);
```

Paso 8) Cerrar el navegador

```
driver.close();
```

Salida:

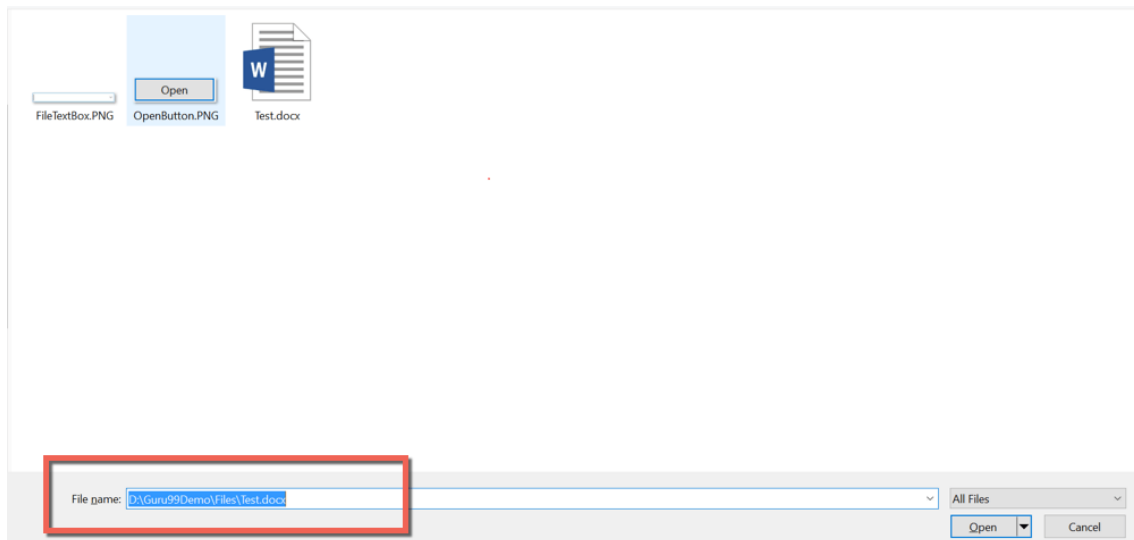
Inicialmente, el script abre el navegador Chrome.



Upload your File

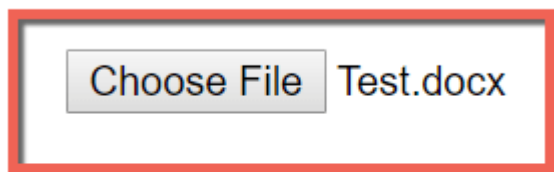
Choose File No file chosen

Haga clic en el botón "Seleccionar archivo" y aparecerá la ventana emergente de archivos de Windows. Ingrese los datos en el cuadro de texto de entrada de archivo y haga clic en el botón "Abrir".



La siguiente pantalla se muestra una vez que se completa la carga del archivo y se cierra el navegador.

Upload your File



File Upload Successful

Conclusión

Sikuli se utiliza para gestionar objetos flash en una página web y ventanas emergentes con facilidad. Sikuli se utiliza mejor cuando los elementos de la interfaz de usuario no

cambian con frecuencia. Debido a esta desventaja, desde una perspectiva de pruebas de automatización, Sikuli tiene menos preferencia en comparación con otros marcos como Robot y AutoIT.