Co-Surfer

Web Automation Application With Selenium

Developer : Doğukan ÖZDER

Project Consultant : Serkan DEVRİM

Introduction:

Co-Surfer is an automation test software developed using WPF and Selenium, designed to automate repetitive tasks on web applications. With Co-Surfer, you can accelerate your testing process by automating various actions on any desired website. It offers a range of functionalities such as clicking, reading, writing, right-clicking, tab navigation, URL navigation, and waiting. Co-Surfer allows you to specify elements using XPath, Id, Name, Classname, or tagname, enabling seamless interaction with different web elements.

Features:

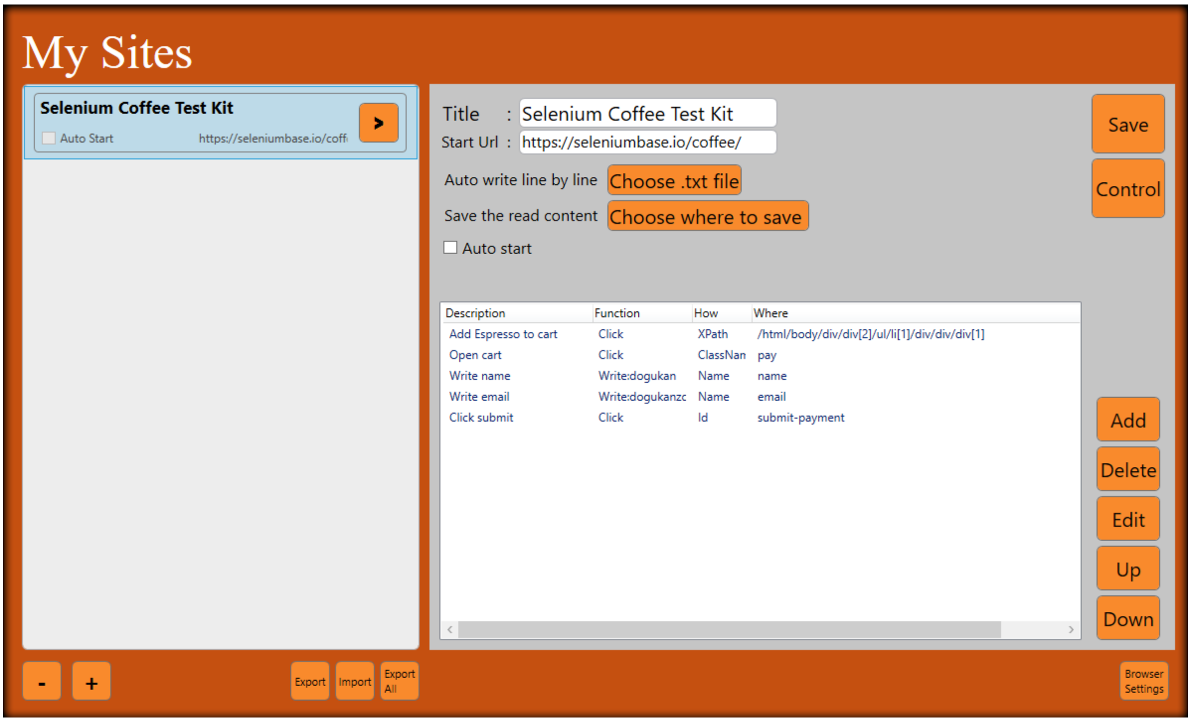
- **Automated Interaction:** Co-Surfer automates a wide range of actions on web pages, including clicking on elements, reading text from web elements, writing text, right-clicking, changing tabs, navigating to different URLs, and waiting for elements to load.
- **Element Identification:** Co-Surfer allows you to identify web elements using XPath, Id, Name, Classname, or tagname. This flexibility enables you to interact with various elements on web pages, making it easier to target specific elements and perform desired actions.
- **Dynamic Element Handling:** Co-Surfer is designed to handle dynamic web pages where the structure may change over time or pop-ups may appear. It provides robust error handling mechanisms to adapt to these changes and continue the automation process seamlessly.
- **Rapid Data Entry:** Co-Surfer excels at rapidly filling in data, making it ideal for tasks that involve entering large amounts of information into web forms or fields. By automating this process, Co-Surfer significantly reduces the time required for repetitive data entry tasks.
- **Flexible Configuration:** Upon launching Co-Surfer, you can easily select the desired website to automate. This flexibility allows you to utilize Co-Surfer on any web application of your choice, streamlining your testing efforts across different projects and websites.

Future Enhancements:

The development of Co-Surfer is an ongoing process, and there are several exciting features planned for future releases:

- **Scrolling Functionality:** Co-Surfer will incorporate a scrolling feature, enabling automated scrolling of web pages. This addition will allow you to automate interactions with elements that are not initially visible on the screen.
- **Multi-Site Support:** Co-Surfer will provide the capability to open and automate multiple websites simultaneously. This enhancement will enable more efficient testing across different web applications and improve productivity.
- **Enhanced Error Handling:** Co-Surfer will further enhance its error handling mechanisms to gracefully handle unexpected scenarios, ensuring smooth execution even in complex web application environments.
- **Advanced Reporting:** Co-Surfer will offer comprehensive reporting features, providing detailed logs and insights on the automation process. These reports will assist in identifying potential issues, debugging automation scripts, and enhancing overall testing efficiency.
- **Integration with Testing Frameworks:** Co-Surfer will provide integrations with popular testing frameworks, enabling seamless incorporation of automated tests into existing test suites and workflows.
- **Analysis with Artificial Intelligence:** Co-Surfer will be able to analyze the read data and convert it into graphically or visually meaningful data.

Co-Surfer is an invaluable tool for testers and developers looking to automate their testing efforts and increase productivity. With its user-friendly interface, robust automation capabilities, and planned future enhancements, Co-Surfer is set to become an essential asset in the arsenal of automation testers.



Selenium Coffee Test Kit

□ Auto Start https://seleniumbase.io/coff

Title : Selenium Coffee Test Kit

Start Url : https://seleniumbase.io/coffee/

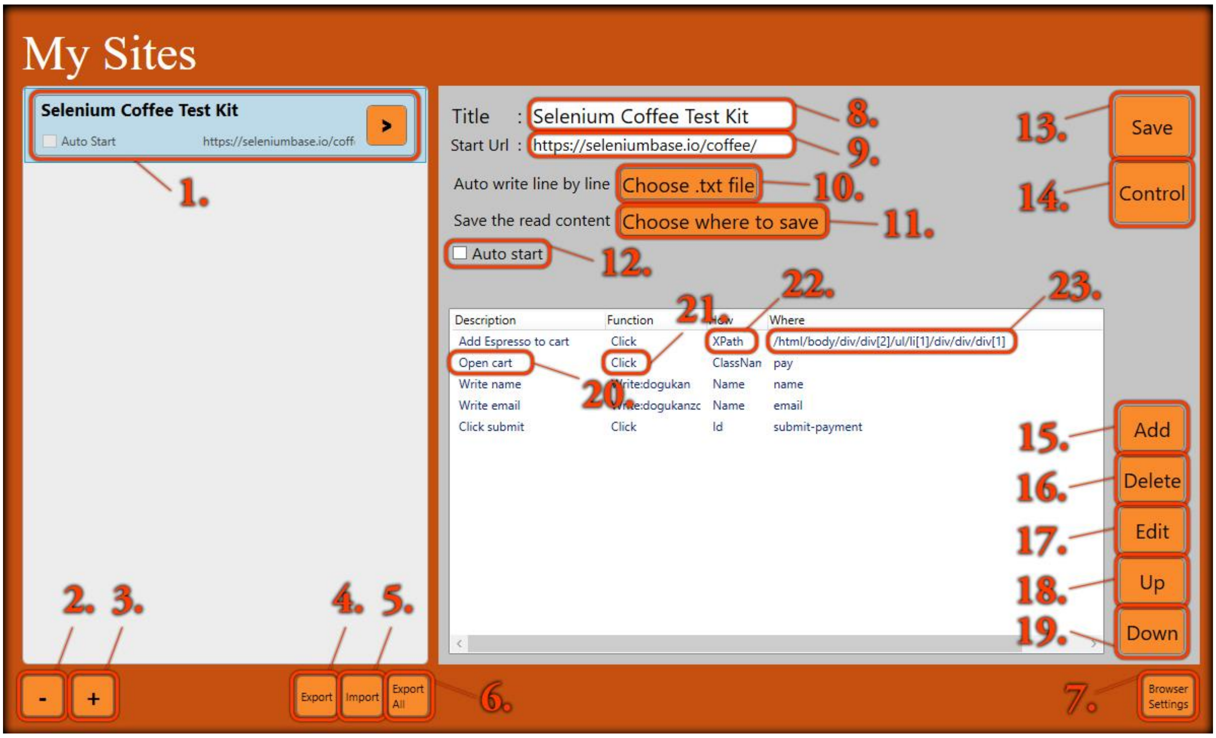
Auto write line by line Choose .txt file

Save the read content Choose where to save

□ Auto start

Description	Function	How	Where
Add Espresso to cart	Click	XPath	/html/body/div/div[2]/ul/li[1]/div/div/div[1]
Open cart	Click	ClassNan	pay
Write name	Write:dogukan	Name	name
Write email	Write:dogukanzc	Name	email
Click submit	Click	Id	submit-payment

Manual:



My Sites

Selenium Coffee Test Kit

☐ Auto Start https://seleniumbase.io/coffi

Title : Selenium Coffee Test Kit

Start Url : https://seleniumbase.io/coffee/

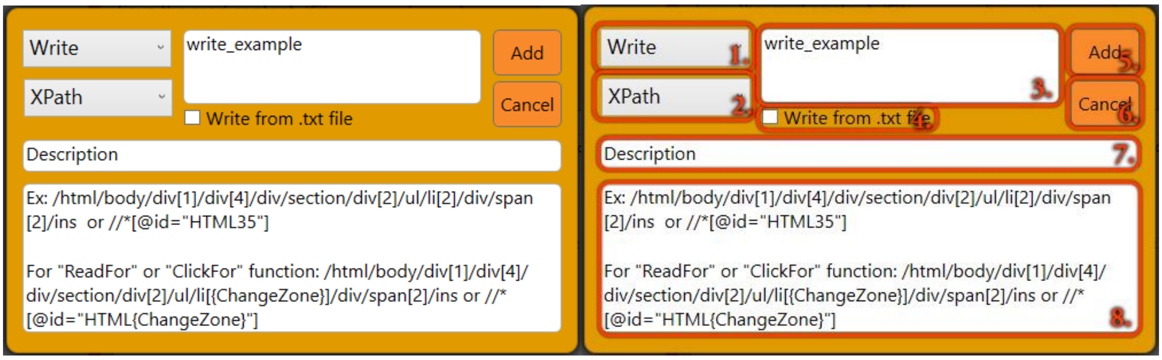
Auto write line by line Choose .txt file

Save the read content Choose where to save

☐ Auto start

Description	Function	How	Where
Add Espresso to cart	Click	XPath	/html/body/div/div[2]/ul/li[1]/div/div/div[1]
Open cart	Click	ClassNan	pay
Write name	Write:dogukan	Name	name
Write email	Write:dogukanzc	Name	email
Click submit	Click	Id	submit-payment

1. The added sites are listed in this field and the method information is displayed.
2. Removes the selected site from the list.
3. Allows you to add a new site.
4. Allows you to save the selected site in Excel format.
5. Allows you to add sites registered in Excel format to the program.
6. It allows you to save all sites at once in the form of an Excel file.
7. Opens the window to set the browser and driver location where the sites will be opened.
8. The title of the site.
9. The starting url of the site.
10. If there is a file to be written line by line, it allows you to select its location.
11. If a value is to be read using the Read function, it allows you to select the place where the content will be saved.
12. If you want the site to run automatically when the program is opened, you can mark it.
13. Adds the created site to the My Sites list.
14. It will help you to check by trying the entered steps.
15. Adds path.
16. Delete selected path.
17. Edit selected path.
18. Moves the selected path up one.
19. Moves the selected path down one.
20. Shows the description information of the path.
21. It shows the function of the path.
22. Indicates which element type the function will use.
23. It shows the element path.



write_example

Ex: /html/body/div[1]/div[4]/div/section/div[2]/ul/li[2]/div/span[2]/ins or //*[@id="HTML35"]

For "ReadFor" or "ClickFor" function: /html/body/div[1]/div[4]/div/section/div[2]/ul/li[{ChangeZone}]/div/span[2]/ins or //*[@id="HTML{ChangeZone}"]

1. Allows you to select a function.
   a. **Click:** Clicks on the selected element.
   b. **Read:** Reads the text contained in the selected element.
   c. **ReadFor:** Reads the text contained in the selected ordered elements.
   d. **Write:** Writes the entered value to the selected element.
   e. **Go to url:** Navigates to the entered address.
   f. **Go back:** Navigates to the previous address.
   g. **Go forward:** Navigates to the next address.
   h. **Wait:** It waits until the entered millisecond, or if -1 is entered, it waits until the user approves.
   i. **RightClick:** Right-clicks on the selected element.
   j. **Change Tab:** Switches to the selected tab.
   k. **ClickFor:** Clicks on the selected ordered elements.
2. It allows you to choose how the selected function will reach the element.
   a. **XPath**
   b. **Id**
   c. **Name**
   d. **ClassName**
   e. **TagName**
3. Enter the value to be written for the Write function.
4. The value to be written for the Write function is marked if it is from the .txt file.
5. Adds the path with the entered values.
6. Cancels the path insertion process.
7. Allows you to briefly add a description for the path.
8. Allows you to write the content that the selected function needs.

Keywords:

Automation testing, WPF, C#, Selenium, Web automation, User interface, Efficiency, Productivity, Element identification, Error handling, Speed, Performance.