WEB TESTING

The Power of Parameterized Locators in Automation

Suraiya 10 Feb 2025 3 195 1 Share

Parameterized locators provide flexibility and efficiency in automation testing by allowing dynamic identification of elements based on varying conditions. Instead of hardcoding locators, we can create reusable templates that accept parameters at runtime, making our test scripts more scalable
and maintainable.

Advantages of Parameterized Locators

- **Dynamic:** We can interact with any element by just passing the required parameters (like button text or product name).
- **Scalable:** As the website grows and new elements are added, we don't need to create new locators; just pass the right parameters.
- **Clean & Reusable:** Parameterized locators reduce redundancy in our test code, making it more concise and easier to maintain.
- **Maintainable:** If the structure of a page changes, we only need to update the locator function, rather than changing multiple instances of locators in our tests.

Example: Parameterized Locators for the QABrains Website

Let's assume the QABrains website has several buttons or elements whose identifiers can change, like buttons with dynamic **text** or **links** for different services. Instead of writing individual locators for each button or element, we'll create parameterized locators to interact with elements based on dynamic attributes.

1. Generic Locator for Any Button or Link Based on Dynamic Text

On a website, links or buttons text like "Home", "Discussion", "Tags", "About Us", "Sign In" often change. Instead of creating separate locators for each, we can use a parameterized locator to handle
them dynamically.

```
public By getLinkByText(String linkText) {
    return By.xpath("//nav//*[self::a or self::button] [normalize-space()='" + linkText
+ "']");
}
```

Usage Example:

To interact with a **"Discussion"** link:

```
By discussionLink = getLinkByText("Discussion");
WebElement discussionElement = driver.findElement(discussionLink);
discussionElement.click();
```

(e.g., "Selenium Training" or "Web Automation").

```
public By getProductByName(String productName) {
       return    By.xpath("//section[contains(@class,    'product-list')]//h2[normalize-
space()='" + productName + "']");
}
```

Usage Example:

To locate a **"Selenium Training"** product:

```
By seleniumTraining = getProductByName("Selenium Training");
WebElement seleniumProduct = driver.findElement(seleniumTraining);
seleniumProduct.click();
```

3. Handling Dynamic Input Fields Based on Labels

We can create parameterized locators to find input fields based on their labels, which might change
(like "Username" or "Email").

```
public By getInputFieldByLabel(String labelText) {
     return   By.xpath("//label[normalize-space()='"   +   labelText   +   "']/following-
sibling::input       |       //label[normalize-space()='"       +       labelText       +
"']/ancestor::div[contains(@class, 'form-group')]//input");
  }
```

Usage Example:

To locate the **"Username"** input eld:

```
By usernameField = getInputFieldByLabel("Username");
WebElement usernameElement = driver.findElement(usernameField);
usernameElement.sendKeys("test user name");
```

4. Dynamic Table Cell Locator

In some cases, we may need to interact with cells in a table, where the row or column numbers are dynamic.

```
public By getTableCellLocator(int row, int column) {
    return By.xpath("//table//tr[" + row + "]//td[" + column + "]");
}
```

Usage Example:

To access a cell in the **5th row** and **7th column**:

```
By cellLocator = getTableCellLocator(5, 7);
WebElement cellElement = driver.findElement(cellLocator);
System.out.println(cellElement.getText());
```

These parameterized locators help make our tests **flexible**, **scalable**, and **maintainable** for dynamic
websites like QABrains.

Comments (1)

Sorted by Newest

Wasif Zaman • 1mo ago • Likes 0

Thanks for sharing your thoughts. Informative insight!

REPLY

Suraiya Author • 1mo ago • Likes 1

Thank you

REPLY

Related Blogs

WEB TESTING 0 0 288

Comprehensive Guide to Testing Cloud-Based Applications and Services for Functionality...

In today's digital-first world, cloud computing has become the backbone of mod

Habiba
18 Mar 2025

WEB TESTING 0 0 317

Stress Testing in Web Testing: Ultimate Guide to Performance Optimization

Stress testing is a crucial part of web application testing that evaluates a sys

Abu Hasan
09 Mar 2025

Popular Tags

View All

Popular Post

Can a Software Tester Become a Game Tester? Here's What You Need t...

As the gaming industry continues to grow, fueled by innovations in virtual reali

Understanding Java Object-Oriented Programming (OOP) Concepts

Java is a powerful and widely used programming language known for its versatilit

Essential Bugs to Check for in Game Testing: A Guide for Beginners

Game testing is crucial to ensure a smooth, engaging, and bug-free experience fo

JMeter: Short technique for Generating an HTML load test report using...

Pre-requisites:Install Java:Java Version: "1.8.0_291" or higher (minimum require

View All

Popular Discussion

01 Top Software Testing Interview Questions and Expert Tips from QA Leaders

02 AI tools for QA engineer

03 What is SQL?

04 Appium, WebDriver

05 What are the most effective strategies you've found for balancing speed and...

View All

QA Brains is the ultimate QA community to exchange knowledge, seek advice, and engage in discussions that enhance Quality Assurance testers' skills and expertise in software testing.

QA Topics

Web Testing

Interview Questions

Game Testing

See more →

Quick Links

Discussion

About Us

Terms & Conditions

Privacy Policy

Follow Us

For Support

support@qabrains.com