



**Sonu Madheshiya**

# **Playwright: Mastering the Essentials - Part 5**

**Mastering Playwright Actions : A Comprehensive Guide**

***Prepared By:***

***Sonu Madheshiya***  
***Automation Tester***



**<https://www.linkedin.com/in/sonumadheshiya/>**



# Playwright : Mastering the Essentials - Part 5

## Mastering Playwright Actions : A Comprehensive Guide

Playwright provides a robust set of tools to interact with web applications, allowing you to simulate a wide range of user actions. This guide covers text input, form interactions, mouse clicks, keyboard shortcuts, file uploads, and more, to help you master Playwright actions.

### Text Input

Playwright's `locator.fill()` method is designed to simplify text input into form fields and editable elements. This method focuses the element and triggers an input event with the provided text.

#### Examples:

##### Text Input:

```
await page.getByRole('textbox').fill('Sonu Madheshiya');
```

##### Date Input:

```
await page.getByLabel('Birth date').fill('2000-02-02');
```

##### Time Input:

```
await page.getByLabel('time').fill('09:15');
```

##### Local Datetime Input:

```
await page.getByLabel('Local time').fill('2020-03-02T05:15');
```

## Checkboxes and Radio Buttons

To manage checkboxes and radio buttons, use the `locator.setChecked()` method. This method allows you to check or uncheck these elements efficiently.



## Examples:

### Check a Checkbox:

```
await page.getByLabel('I agree').check();
```

### Assert Checked State:

```
expect(page.getByLabel('Subscribe to newsletter')).toBeChecked();
```

### Select a Radio Button:

```
await page.getByLabel('Option').check();
```

## Select Options

Use `locator.selectOption()` to handle `<select>` elements. You can specify options by value or label and select multiple options if needed.

## Examples:

### Single Selection by Value:

```
await page.getByLabel('Color').selectOption('green');
```

### Single Selection by Label:

```
await page.getByLabel('Color').selectOption({ label: 'Green' });
```

### Multiple Selections:

```
await page.getByLabel('Colors').selectOption(['red', 'green', 'blue']);
```



## Mouse Clicks

Playwright supports a variety of mouse click interactions, including single, double, right, and more specialized clicks.

### Examples:

#### Generic Click:

```
await page.getByRole('button').click();
```

#### Double Click:

```
await page.getByText('Item').dblclick();
```

#### Right Click:

```
await page.getByText('Item').click({ button: 'right' });
```

#### Shift + Click:

```
await page.getByText('Item').click({ modifiers: ['Shift'] });
```

#### Ctrl + click on Windows and Linux or Meta + click on macOS :

```
await page.getByText('Item').click({ modifiers: ['ControlOrMeta'] });
```

#### Hover Over Element:

```
await page.getByText('Item').hover();
```

#### Click at Position:

```
await page.getByText('Item').click({ position: { x: 0, y: 0 } });
```



**Forcing Clicks:** If an element is covered or not actionable, you can force the click:

```
await page.getByRole('button').click({ force: true });
```

**Programmatic Click:** Trigger a click event directly:

```
await page.getByRole('button').dispatchEvent('click');
```

## Type Characters

To simulate typing characters one by one, use `locator.pressSequentially()`. This method emulates real keyboard input by generating all necessary keyboard events.

**Example:**

```
await page.locator('#area').pressSequentially('Sonu Madheshiya Automation Tester');
```

## Keys and Shortcuts

Playwright supports various key presses and keyboard shortcuts.

**Examples:**

**Hit Enter:**

```
await page.getByText('Submit').press('Enter');
```

**Control + Right Arrow:**

```
await page.getByRole('textbox').press('Control+ArrowRight');
```

**Press \$:**

```
await page.getByRole('textbox').press('$');
```



### Shift + A:

```
await page.locator('#name').press('Shift+A');
```

### Shift + Arrow Left:

```
await page.locator('#name').press('Shift+ArrowLeft');
```

**Shortcuts:** Modifier keys such as Shift, Control, Alt, and Meta can be used in combinations:

### Select All:

```
await page.getByRole('textbox').press('Control+A'); // On Windows/Linux
await page.getByRole('textbox').press('Meta+A');    // On macOS
```

### Copy:

```
await page.getByRole('textbox').press('Control+C'); // On Windows/Linux
await page.getByRole('textbox').press('Meta+C');    // On macOS
```

### Paste:

```
await page.getByRole('textbox').press('Control+V'); // On Windows/Linux
await page.getByRole('textbox').press('Meta+V');    // On macOS
```

## Upload Files

Manage file uploads using `locator.setInputFiles()`. This method supports single and multiple file uploads, directory uploads, and uploading from memory buffers.

### Examples:

#### Select One File:

```
await page.getByLabel('Upload file').setInputFiles(path.join(__dirname, 'myfile.pdf'));
```





### Select Multiple Files:

```
await page.getByLabel('Upload files').setInputFiles([
  path.join(__dirname, 'file1.txt'),
  path.join(__dirname, 'file2.txt'),
]);
```

### Upload Directory:

```
await page.getByLabel('Upload directory').setInputFiles(path.join(__dirname, 'mydir'));
```

### Upload Buffer from Memory:

```
await page.getByLabel('Upload file').setInputFiles({
  name: 'file.txt',
  mimeType: 'text/plain',
  buffer: Buffer.from('this is test')
});
```

### Handling Dynamic Input Elements:

```
const fileChooserPromise = page.waitForEvent('filechooser');
await page.getByLabel('Upload file').click();
const fileChooser = await fileChooserPromise;
await fileChooser.setFiles(path.join(__dirname, 'myfile.pdf'));
```

### Focus Element

To handle dynamic focus events, use `locator.focus()` to focus on a specific element.

#### Example:

```
await page.getByLabel('Password').focus();
```

### Drag and Drop

Perform drag-and-drop operations using `locator.dragTo()` or manual drag-and-drop techniques.


Examples:

Using `dragTo()`:

```
await page.locator('#item-to-be-dragged').dragTo(page.locator('#item-to-drop-at'));
```

Manual Dragging:

```
await page.locator('#item-to-be-dragged').hover();
await page.mouse.down();
await page.locator('#item-to-drop-at').hover();
await page.locator('#item-to-drop-at').hover(); // Second hover
await page.mouse.up();
```



**Note:** To reliably trigger the dragover event, you may need to perform the drag operation twice:

```
await page.locator('#item-to-be-dragged').hover();
await page.mouse.down();
await page.locator('#item-to-drop-at').hover();
await page.locator('#item-to-drop-at').hover(); // Second hover
await page.mouse.up();
```

## Scrolling

While Playwright generally handles scrolling automatically, manual scrolling can be done for precise control.

Examples:

**Automatic Scrolling:** *Here Scroll automatically so that button is visible and then click.*

```
await page.getByRole('button').click();
```

Manual Scrolling:

```
await page.getText('Footer text').scrollIntoViewIfNeeded();
```





<https://www.linkedin.com/in/sonumadheshiya/>

### Using Mouse Wheel:

```
await page.getByTestId('scrolling-container').hover();  
await page.mouse.wheel(0, 10);
```

### Programmatic Scrolling:

```
await page.getByTestId('scrolling-container').evaluate(e => e.scrollTop += 100);
```

---

**Stay tuned with [Sonu Madheshiya](#) on LinkedIn for more interesting content on automation testing.**