

[Skip to main content](#)



[PlaywrightDocsAPI](#)

[Node.js](#)

- [Node.js](#)
- [Python](#)
- [Java](#)
- [.NET](#)

[Community](#)

Search⌕

- [Getting Started](#)
  - [Installation](#)
  - [Writing tests](#)
  - [Generating tests](#)
  - [Running and debugging tests](#)
  - [Trace viewer](#)
  - [Setting up CI](#)
- [Getting started - VS Code](#)
- [Release notes](#)
- [Canary releases](#)
- [Playwright Test](#)
  - [Test configuration](#)
  - [Test use options](#)
  - [Annotations](#)
  - [Command line](#)

- [Emulation](#)
  - [Fixtures](#)
  - [Global setup and teardown](#)
  - [Parallelism](#)
  - [Parameterize tests](#)
  - [Projects](#)
  - [Reporters](#)
  - [Retries](#)
  - [Sharding](#)
  - [Timeouts](#)
  - [TypeScript](#)
  - [UI Mode](#)
  - [Web server](#)
- [Guides](#)
  - [Library](#)
  - [Accessibility testing](#)
  - [Actions](#)
  - [Assertions](#)
  - [API testing](#)
  - [Authentication](#)
  - [Auto-waiting](#)
  - [Best Practices](#)
  - [Browsers](#)
  - [Chrome extensions](#)
  - [Clock](#)
  - [Components \(experimental\)](#)
  - [Debugging Tests](#)
  - [Dialogs](#)
  - [Downloads](#)
  - [Evaluating JavaScript](#)
  - [Events](#)
  - [Extensibility](#)
  - [Frames](#)
  - [Handles](#)
  - [Isolation](#)
  - [Locators](#)
  - [Mock APIs](#)
  - [Mock browser APIs](#)
  - [Navigations](#)
  - [Network](#)
  - [Other locators](#)
  - [Page object models](#)
  - [Pages](#)
  - [Screenshots](#)
  - [Visual comparisons](#)
  - [Test generator](#)
  - [Trace viewer](#)
  - [Videos](#)
  - [WebView2](#)
- [Migration](#)

- [Integrations](#)
- [Supported languages](#)
- 
- Guides
- Dialogs

On this page

# Dialogs

## Introduction

Playwright can interact with the web page dialogs such as [alert](#), [confirm](#), [prompt](#) as well as [beforeunload](#) confirmation. For print dialogs, see [Print](#).

## alert(), confirm(), prompt() dialogs

By default, dialogs are auto-dismissed by Playwright, so you don't have to handle them. However, you can register a dialog handler before the action that triggers the dialog to either [dialog.accept\(\)](#) or [dialog.dismiss\(\)](#) it.

```
page.on('dialog', dialog => dialog.accept());  
await page.getByRole('button').click();
```

NOTE

[page.on\('dialog'\)](#) listener **must handle** the dialog. Otherwise your action will stall, be it [locator.click\(\)](#) or something else. That's because dialogs in Web are modals and therefore block further page execution until they are handled.

As a result, the following snippet will never resolve:

WARNING

WRONG!

```
page.on('dialog', dialog => console.log(dialog.message()));  
await page.getByRole('button').click(); // Will hang here
```

NOTE

If there is no listener for [page.on\('dialog'\)](#), all dialogs are automatically dismissed.

## beforeunload dialog

When [page.close\(\)](#) is invoked with the truthy `runBeforeUnload` value, the page runs its unload handlers. This is the only case when [page.close\(\)](#) does not wait for the page to actually close, because it might be that the page stays open in the end of the operation.

You can register a dialog handler to handle the `beforeunload` dialog yourself:

```
page.on('dialog', async dialog => {
  assert(dialog.type() === 'beforeunload');
  await dialog.dismiss();
});
await page.close({ runBeforeUnload: true });
```

## Print dialogs

In order to assert that a print dialog via [window.print](#) was triggered, you can use the following snippet:

```
await page.goto('<url>');

await page.evaluate('(() => {window.waitForPrintDialog = new Promise(f =>
window.print = f);})();})();');
await page.getByText('Print it!').click();

await page.waitForFunction('window.waitForPrintDialog');
```

This will wait for the print dialog to be opened after the button is clicked. Make sure to evaluate the script before clicking the button / after the page is loaded.

[Previous](#)  
[Debugging Tests](#)

[Next](#)  
[Downloads](#)

- [Introduction](#)
- [alert\(\), confirm\(\), prompt\(\) dialogs](#)
- [beforeunload dialog](#)
- [Print dialogs](#)

Learn

- [Getting started](#)
- [Playwright Training](#)
- [Learn Videos](#)
- [Feature Videos](#)

Community

- [Stack Overflow](#)
- [Discord](#)
- [Twitter](#)
- [LinkedIn](#)

More

- [GitHub](#)
- [YouTube](#)
- [Blog](#)
- [Ambassadors](#)

Copyright © 2024 Microsoft