

[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](#)
- 
- Playwright Test
- Timeouts

On this page

# Timeouts

## Introduction

Playwright Test has multiple configurable timeouts for various tasks.

Timeout	Default	Description
		Timeout for each test, includes test, hooks and fixtures:
		<b>SET DEFAULT</b>
Test timeout	30000 ms	<code>config = { timeout: 60000 }</code>
		<b>OVERRIDE</b>
		<code>test.setTimeout(120000)</code>
		Timeout for each assertion:
		<b>SET DEFAULT</b>
Expect timeout	5000 ms	<code>config = { expect: { timeout: 10000 } }</code>
		<b>OVERRIDE</b>
		<code>expect(locator).toBeVisible({ timeout: 10000 })</code>

## Test timeout

Playwright Test enforces a timeout for each test, 30 seconds by default. Time spent by the test function, fixtures, `beforeEach` and `afterEach` hooks is included in the test timeout.

Timed out test produces the following error:

```
example.spec.ts:3:1 > basic test =====
Timeout of 30000ms exceeded.
```

The same timeout value also applies to `beforeAll` and `afterAll` hooks, but they do not share time with any test.

## Set test timeout in the config

```
playwright.config.ts
import { defineConfig } from '@playwright/test';

export default defineConfig({
  timeout: 5 * 60 * 1000,
});
```

API reference: [testConfig.timeout](#).

## Set timeout for a single test

```
import { test, expect } from '@playwright/test';

test('slow test', async ({ page }) => {
  test.slow(); // Easy way to triple the default timeout
  // ...
});

test('very slow test', async ({ page }) => {
  test.setTimeout(120000);
  // ...
});
```

API reference: [test.setTimeout\(\)](#) and [test.slow\(\)](#).

## Change timeout from a beforeEach hook

```
import { test, expect } from '@playwright/test';

test.beforeEach(async ({ page }, testInfo) => {
  // Extend timeout for all tests running this hook by 30 seconds.
  testInfo.setTimeout(testInfo.timeout + 30000);
});
```

API reference: [testInfo.setTimeout\(\)](#).

## Change timeout for beforeAll/afterAll hook

beforeAll and afterAll hooks have a separate timeout, by default equal to test timeout. You can change it separately for each hook by calling [testInfo.setTimeout\(\)](#) inside the hook.

```
import { test, expect } from '@playwright/test';

test.beforeAll(async () => {
  // Set timeout for this hook.
  test.setTimeout(60000);
});
```

API reference: [testInfo.setTimeout\(\)](#).

## Expect timeout

Web-first assertions like `expect(locator).toHaveText()` have a separate timeout, 5 seconds by default. Assertion timeout is unrelated to the test timeout. It produces the following error:

```
example.spec.ts:3:1 > basic test =====
Error: expect(received).toHaveText(expected)

Expected string: "my text"
```

```
Received string: ""
Call log:
  - expect.toHaveText with timeout 5000ms
  - waiting for "locator('button')"
```

## Set expect timeout in the config

```
playwright.config.ts
import { defineConfig } from '@playwright/test';

export default defineConfig({
  expect: {
    timeout: 10 * 1000,
  },
});
```

## Global timeout

Playwright Test supports a timeout for the whole test run. This prevents excess resource usage when everything went wrong. There is no default global timeout, but you can set a reasonable one in the config, for example one hour. Global timeout produces the following error:

```
Running 1000 tests using 10 workers

514 skipped
486 passed
Timed out waiting 3600s for the entire test run
```

You can set global timeout in the config.

```
// playwright.config.ts
import { defineConfig } from '@playwright/test';

export default defineConfig({
  globalTimeout: 60 * 60 * 1000,
});
```

API reference: [testConfig.globalTimeout](#).

## Advanced: low level timeouts

These are the low-level timeouts that are pre-configured by the test runner, you should not need to change these. If you happen to be in this section because your test are flaky, it is very likely that you should be looking for the solution elsewhere.

Timeout	Default	Description
		Timeout for each action:
Action timeout	no timeout	<b>SET DEFAULT</b> config = { use: { actionTimeout: 10000 } } <b>OVERRIDE</b> locator.click({ timeout: 10000 })

Timeout	Default	Description
Navigation timeout	no timeout	Timeout for each navigation action: <b>SET DEFAULT</b> config = { use: { navigationTimeout: 30000 } } <b>OVERRIDE</b> page.goto('/', { timeout: 30000 })
Global timeout	no timeout	Global timeout for the whole test run: <b>SET IN CONFIG</b> config = { globalTimeout: 60*60*1000 }
beforeAll/afterAll timeout	30000 ms	Timeout for the hook: <b>SET IN HOOK</b> test.setTimeout(60000)
Fixture timeout	no timeout	Timeout for an individual fixture: <b>SET IN FIXTURE</b> { scope: 'test', timeout: 30000 }

## Set timeout for a single assertion

```
import { test, expect } from '@playwright/test';

test('basic test', async ({ page }) => {
  await expect(page.getByRole('button')).toHaveText('Sign in', { timeout: 10000 });
});
```

## Set action and navigation timeouts in the config

```
playwright.config.ts
import { defineConfig } from '@playwright/test';

export default defineConfig({
  use: {
    actionTimeout: 10 * 1000,
    navigationTimeout: 30 * 1000,
  },
});
```

API reference: [testOptions.actionTimeout](#) and [testOptions.navigationTimeout](#).

## Set timeout for a single action

```
import { test, expect } from '@playwright/test';

test('basic test', async ({ page }) => {
  await page.goto('https://playwright.dev', { timeout: 30000 });
  await page.getByText('Get Started').click({ timeout: 10000 });
});
```

## Fixture timeout

By default, [fixture](#) shares timeout with the test. However, for slow fixtures, especially [worker-scoped](#) ones, it is convenient to have a separate timeout. This way you can keep the overall test timeout small, and give the slow fixture more time.

- TypeScript
- JavaScript

```
import { test as base, expect } from '@playwright/test';

const test = base.extend<{ slowFixture: string }>({
  slowFixture: [async ({}, use) => {
    // ... perform a slow operation ...
    await use('hello');
  }, { timeout: 60000 }]
});

test('example test', async ({ slowFixture }) => {
  // ...
});

const { test: base, expect } = require('@playwright/test');

const test = base.extend({
  slowFixture: [async ({}, use) => {
    // ... perform a slow operation ...
    await use('hello');
  }, { timeout: 60000 }]
});

test('example test', async ({ slowFixture }) => {
  // ...
});
```

API reference: [test.extend\(\)](#).

[Previous](#)  
[Sharding](#)

[Next](#)  
[TypeScript](#)

- [Introduction](#)
- [Test timeout](#)
  - [Set test timeout in the config](#)
  - [Set timeout for a single test](#)
  - [Change timeout from a beforeEach hook](#)
  - [Change timeout for beforeAll/afterAll hook](#)
- [Expect timeout](#)
  - [Set expect timeout in the config](#)
- [Global timeout](#)
- [Advanced: low level timeouts](#)
  - [Set timeout for a single assertion](#)
  - [Set action and navigation timeouts in the config](#)
  - [Set timeout for a single action](#)
- [Fixture timeout](#)

Learn

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

## Community

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

## More

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

Copyright © 2024 Microsoft