

Config file

При ініціалізації проекту буде створено файл конфігу `playwright.config.ts(.js)`. Цей файл є серцем налаштувань вашого тестового середовища та визначає, як `Playwright` виконуватиме ваші тести.

Конфігураційний файл дозволяє централізовано керувати всіма аспектами тестування: від вибору браузерів до налаштування паралельного виконання. Правильна конфігурація значно підвищує ефективність тестування та спрощує підтримку проекту.

Структура базового конфігу

Початковий конфігураційний файл створюється автоматично і містить основні налаштування для старту роботи. Він імпортує необхідні типи з `@playwright/test` та експортує об'єкт конфігурації через функцію `defineConfig`.

`testDir`

Вказує папку з тестами

`use`

Загальні опції для всіх тестів

`projects`

Конфігурація для різних браузерів

Основні опції конфігурації

Playwright надає широкий спектр опцій для налаштування поведінки тестів. Кожна опція відповідає за конкретний аспект виконання тестів і може бути адаптована під вимоги проекту.

Опція **forbidOnly** особливо корисна в CI/CD пайплайнах, оскільки запобігає випадковому коміту тестів з `.only`, що могло б привести до виконання лише частини тест-съюту.

Налаштування **fullyParallel** дозволяє максимально прискорити виконання тестів, запускаючи їх одночасно, що критично важливо для великих проектів.

Ключові переваги

- Гнучке керування паралелізмом
- Автоматичні повторні спроби
- Множинні репортери
- CI/CD оптимізація

forbidOnly

Блокує запуск якщо є `.only` в тестах

fullyParallel

Паралельне виконання всіх тестів

projects

Множинні браузери та конфігурації

reporter

Формат звітів про тестування

retries

Кількість повторних спроб

workers

Максимум паралельних процесів

Фільтрування тестів



1

`testIgnore`

Glob-патерн для ігнорування файлів. Корисно для виключення допоміжних файлів або тест-даних.

2

`testMatch`

Glob-патерн для визначення тестових файлів. Дозволяє запускати лише специфічні групи тестів.

Playwright надає потужні механізми фільтрування тестових файлів через glob-патерни та регулярні вирази. Це дозволяє точно контролювати, які тести будуть виконані.

Опція **testMatch** визначає, які файли відповідають тестовим, а **testIgnore** дозволяє виключити певні файли або директорії з виконання.



Просунута конфігурація: Частина 1

Просунуті опції конфігурації надають детальний контроль над процесом тестування. Вони дозволяють налаштовувати глобальні хуки, управляти артефактами та встановлювати таймаути.

1

`outputDir`

Папка для всіх артефактів тестування: скріншотів, відео, трейсів та інших результатів виконання

2

`globalSetup`

Шлях до файлу з глобальними налаштуваннями, що виконуються перед усіма тестами

Просунута конфігурація: Частина 2



globalTeardown

Очищення після тестів



timeout

Дефолтний таймаут

Опція **globalTeardown** виконується після завершення всіх тестів і використовується для очищенння ресурсів, закриття з'єднань або видалення тимчасових даних.

Параметр **timeout** встановлює максимальний час виконання для кожного тесту в мілісекундах. Значення за замовчуванням — 30 секунд.

- **Важливо:** Детальніше про всі опції конфігурації можна почитати в офіційній документації Playwright. Правильне налаштування цих параметрів критично важливе для стабільності тест-сьюту.

Проекти в Playwright

Проект – це логічна група тестів, що виконуються з однаковою конфігурацією. Концепція проектів є однією з найпотужніших можливостей Playwright, що дозволяє організувати складні тестові сценарії.

Проекти дозволяють запускати один і той же набір тестів на різних браузерах, пристроях, з різними тайм-аутами або у різних середовищах. Це забезпечує комплексне покриття тестуванням без дублювання коду.



Крос-браузерне тестування

Запуск тестів на Chromium, Firefox, WebKit та інших браузерах одночасно



Тестування на пристроях

Емуляція мобільних пристройів та різних viewport'ів для responsive-дизайну



Множинні середовища

Розділення тестів для staging, production та інших оточень

Конфігурація проектів для браузерів

Playwright підтримує всі основні браузерні движки та дозволяє тестувати веб-додатки на різних платформах. Кожен проект може мати унікальні налаштування та використовувати попередньо визначені конфігурації пристрій.



Desktop Chrome

Chromium engine для desktop



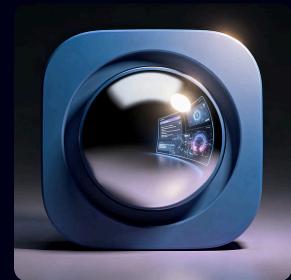
Firefox

Mozilla Firefox engine



WebKit

Safari browser engine



Mobile Chrome

Pixel 5 емуляція



Mobile Safari

iPhone 12 емуляція



Microsoft Edge

Edge browser testing

Використання `devices` об'єкта з `@playwright/test` забезпечує точну емуляцію реальних пристрій, включаючи `viewport`, `user agent` та інші параметри.

Запуск всіх проектів



За замовчуванням, Playwright виконує тести для всіх налаштованих проектів одночасно. Команда `npx playwright test` запускає повний тест-сьют на всіх браузерах та конфігураціях.

Worker'и автоматично розподіляють навантаження між проектами, оптимізуючи час виконання. У прикладі 7 тестів виконуються за допомогою 5 worker'ів паралельно.

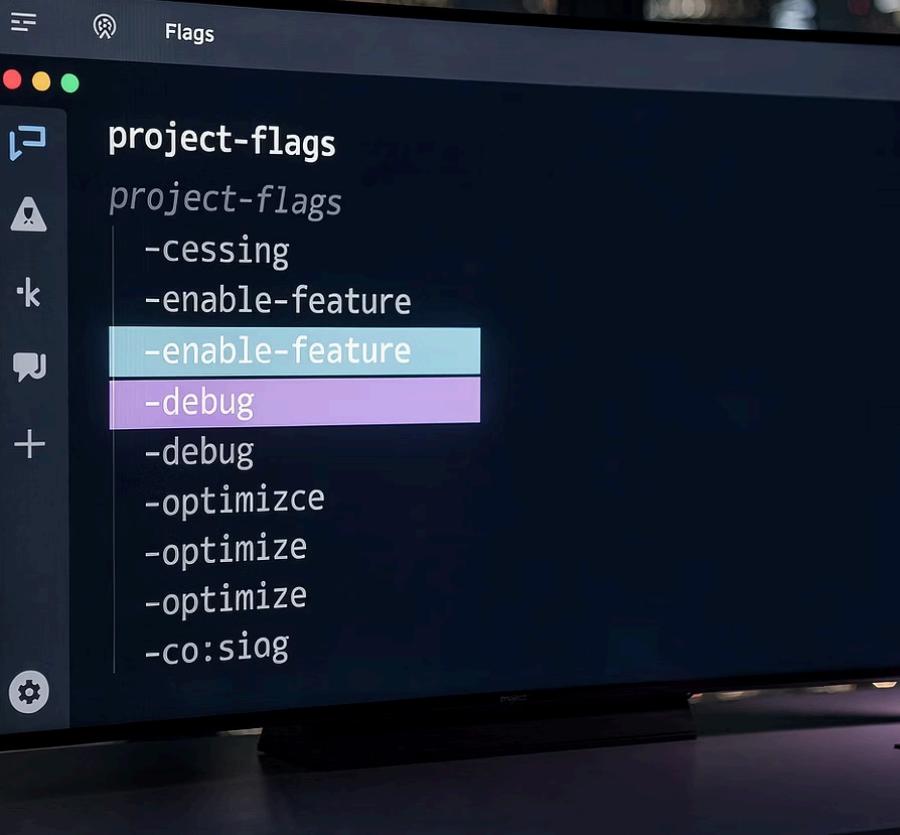
```
npx playwright test
```

Running 7 tests using 5 workers

- ✓ [chromium] › example.spec.ts:3:1 › basic test (2s)
- ✓ [firefox] › example.spec.ts:3:1 › basic test (2s)
- ✓ [webkit] › example.spec.ts:3:1 › basic test (2s)
- ✓ [Mobile Chrome] › example.spec.ts:3:1 › basic test (2s)
- ✓ [Mobile Safari] › example.spec.ts:3:1 › basic test (2s)
- ✓ [Microsoft Edge] › example.spec.ts:3:1 › basic test (2s)
- ✓ [Google Chrome] › example.spec.ts:3:1 › basic test (2s)

Запуск конкретного проекту

Для запуску тестів лише на певному браузері або конфігурації використовуйте флаг `--project`. Це значно прискорює процес розробки, коли потрібно протестувати зміни на конкретному браузері.



- 1
- 2
- 3

Вказуємо проект

Використовуємо `--project=назва`

Виконання тестів

Запускаються лише для обраного проекту

Швидкий результат

Економія часу під час розробки

```
npx playwright test --project=firefox
```

```
Running 1 test using 1 worker
```

```
✓ [firefox] › example.spec.ts:3:1 › basic test (2s)
```

Розділення тестів на проекти

Проекти можна використовувати для логічного розділення тестів за функціональністю, критичністю або іншими критеріями. Це дозволяє створювати гнучкі стратегії тестування.



- **Висновок:** Проекти надають високу гнучкість конфігурації, дозволяючи адаптувати тестове середовище під специфічні потреби додатку та команди розробки.