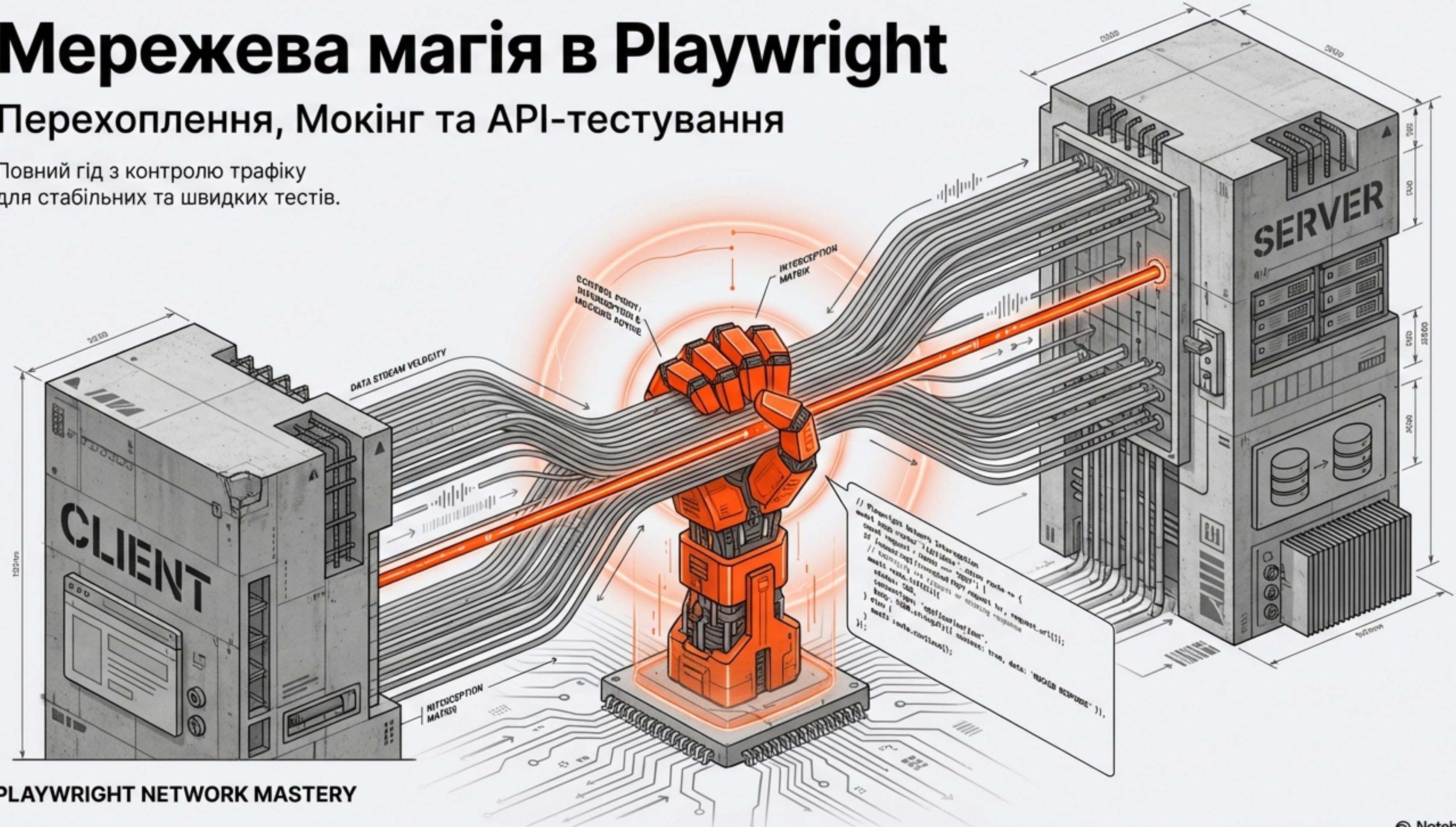


Мережева магія в Playwright

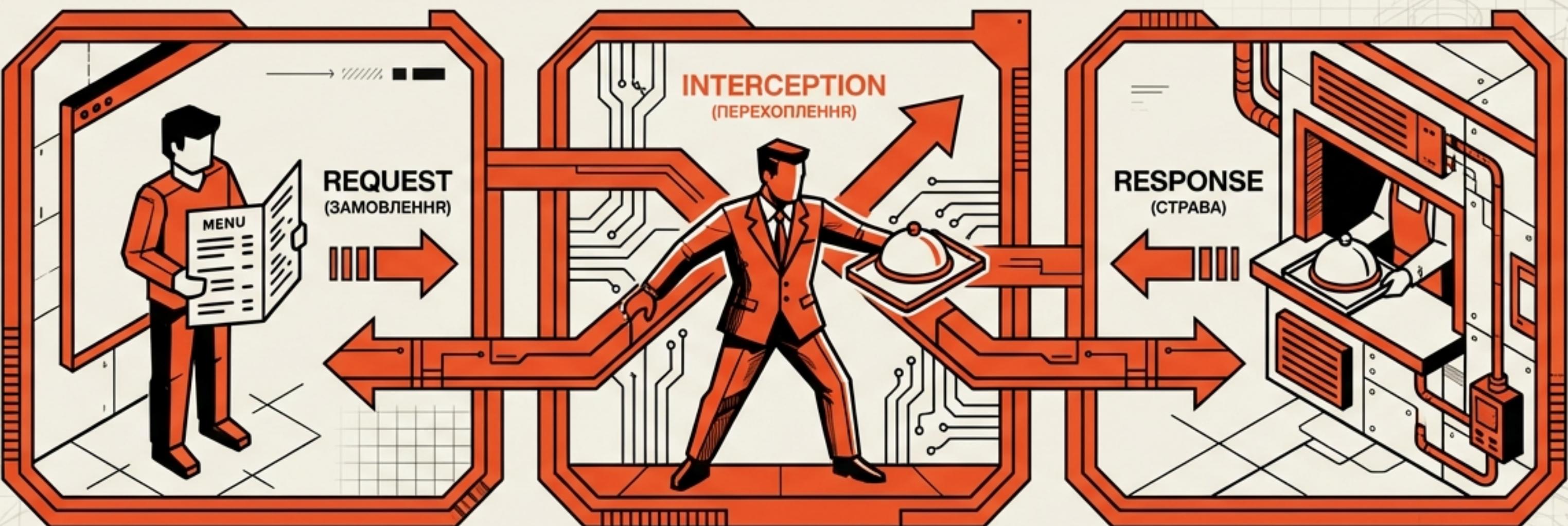
Перехоплення, Мокінг та API-тестування

Повний гід з контролю трафіку
для стабільних та швидких тестів.



PLAYWRIGHT NETWORK MASTERY

Анатомія обміну даними: Ресторанна аналогія



Request:

Повідомлення від браузера до сервера.

Interception:

Можливість «підслухати», змінити або заблокувати.

Response:

Відповідь сервера (дані, статус).

Пасивне спостереження: Рентгенівський зір

Не можна виправити те, що не бачиш.

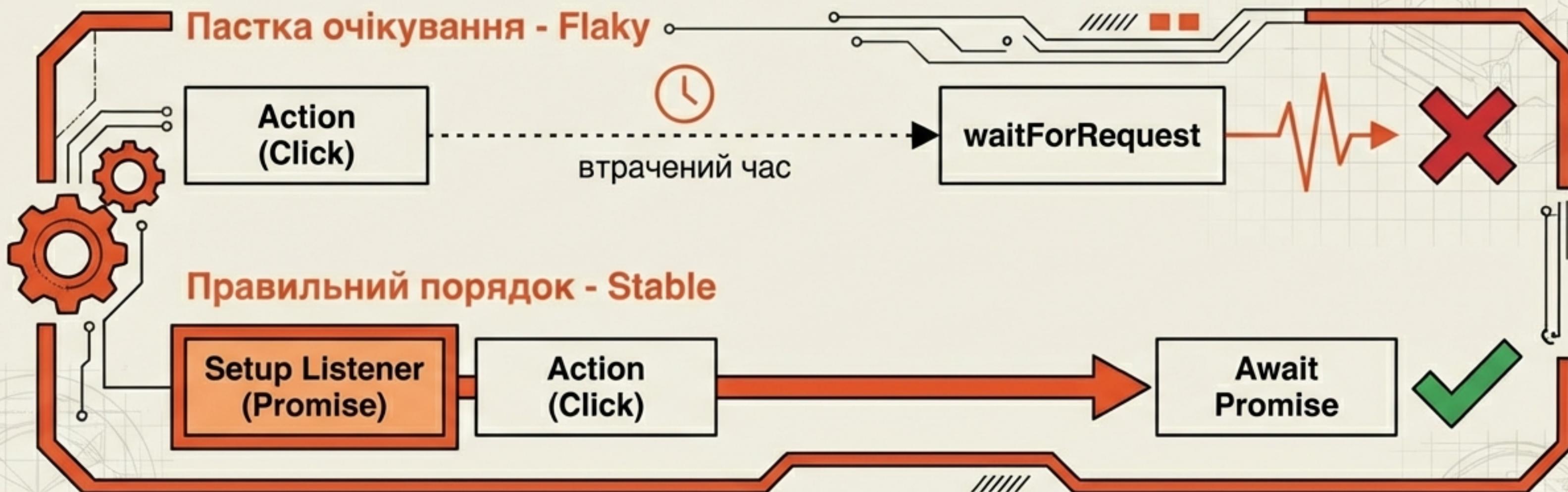
```
>> GET /api/user  
<< 200 OK /api/user
```

```
// Слухаємо вихідні запити  
page.on('request', request =>  
    console.log('>>', request.method(), request.url()));  
  
// Слухаємо вхідні відповіді  
page.on('response', response =>  
    console.log('<<', response.status(), response.url()));
```



Логування
без втручання

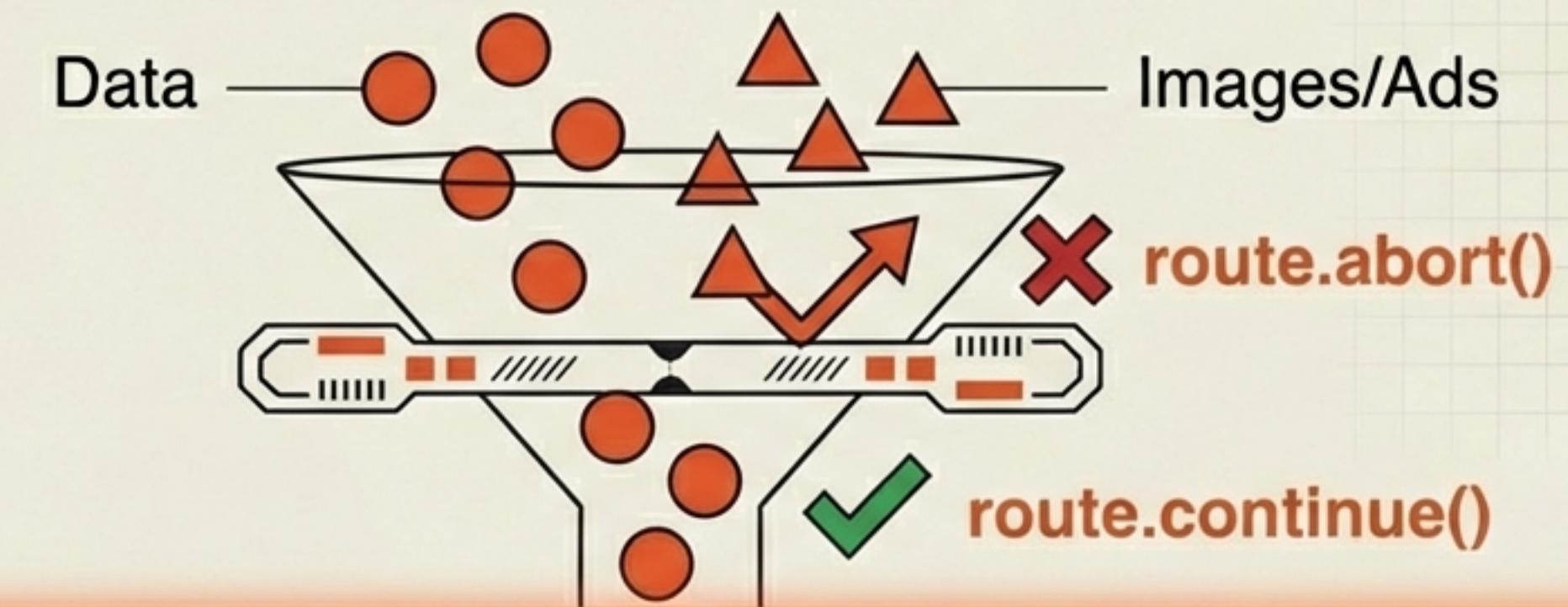
Синхронізація: Чекайте на мережу, а не на UI



waitForRequest / waitForResponse: Надійніше за UI.

Glob Patterns: `*/api/**` (будь-яка частина), `*/*.jpg` (розширення).

The Gatekeeper: Блокування та Фільтрація

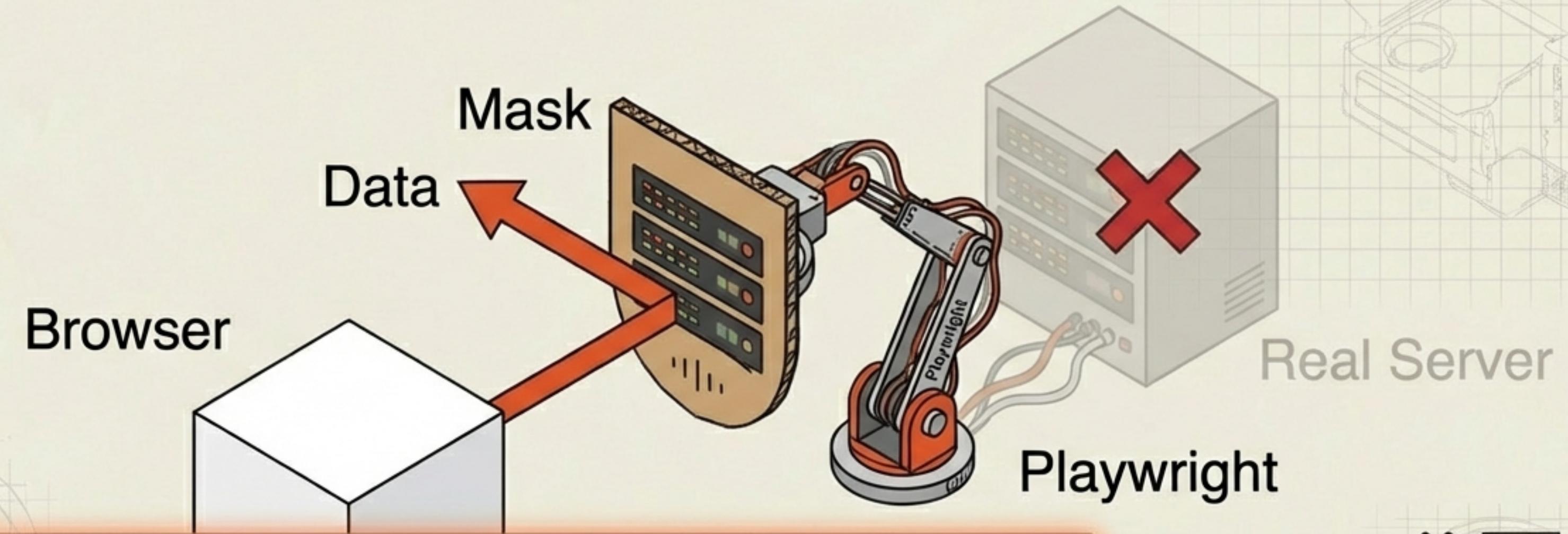


```
// Блокування зображень для швидкості
await page.route('**/*.{png,jpg,jpeg}', route => route.abort());
```

- **route.abort()**: Пришвидшення тестів (блокування картинок, трекерів).
- **route.continue()**: Модифікація заголовків на льоту.



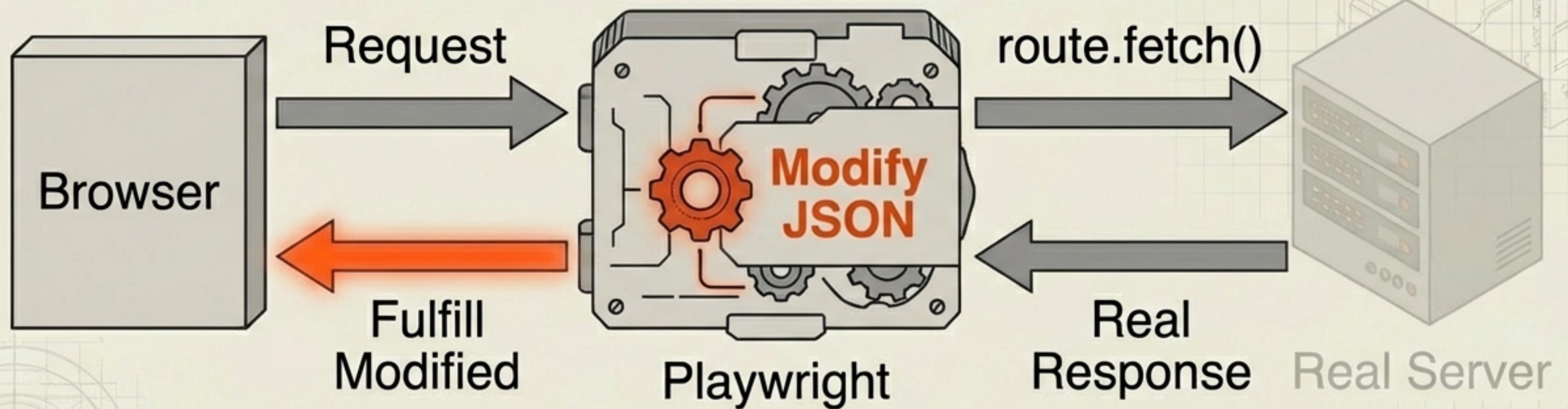
Ілюзіоніст: Основи Mocking



```
await page.route('**/api/login', route => {
  route.fulfill({ status: 200,
    contentType: 'text/plain',
    body: 'accept' });
});
```

Переваги:
Незалежність від
бекенду, миттєва
швидкість.

Розширені підміни: Модифікація реальних даних



```
const response = await route.fetch();
const json = await response.json();
json.role = 'admin'; // Зміна даних
await route.fulfill({ response, json });
```

Обережно: Типові помилки Mocking

1



Зависання

Забули `fulfill()`
або `continue()`.

2



Запізнення

Реєстрація маршруту
ПІСЛЯ `page.goto`.

3

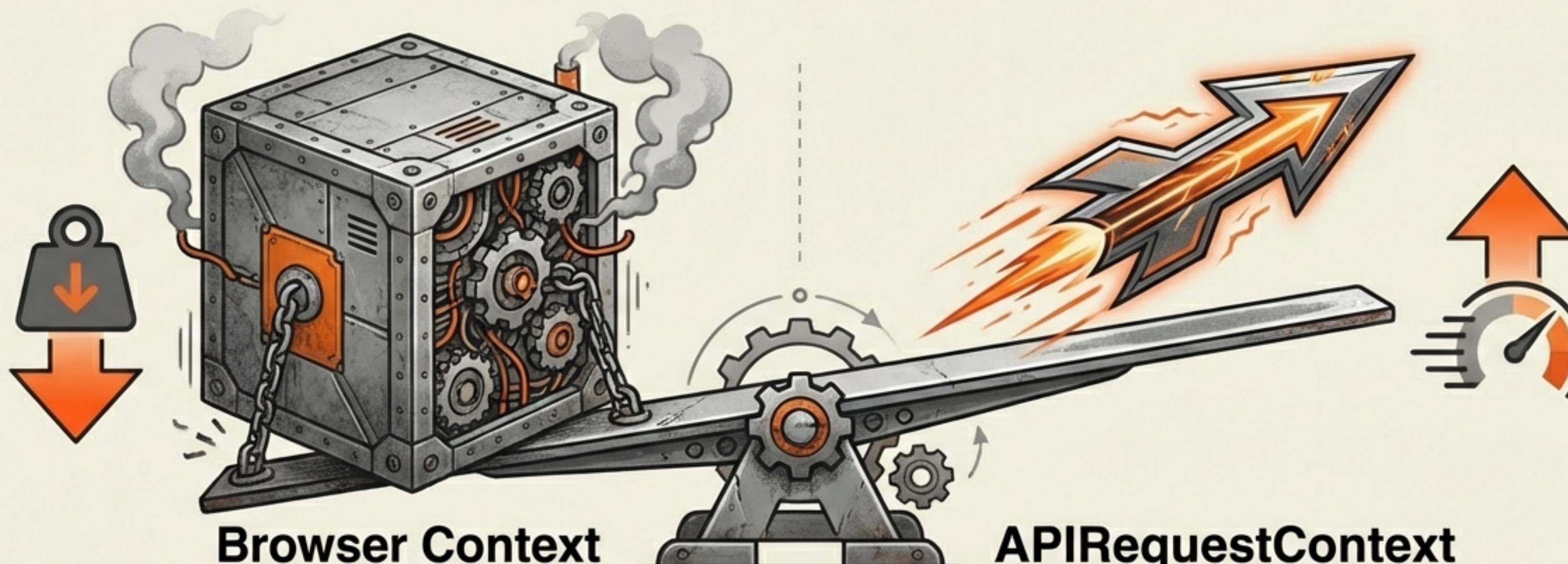


Конфлікти

Перший зареєстрований
 маршрут перемагає.

Правило: Реєструйте маршрути до початку навігації.

За межами браузера: Чисте API-тестування



Browser Context
(UI + Network)

APIRequestContext

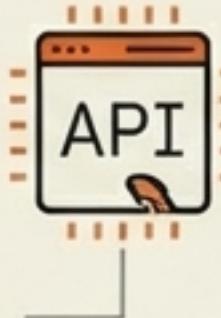


Спільні Cookies з браузером.
Для змішаних тестів.

Shared Context (`page.request`)

Повна ізоляція.
Для чистих API-тестів.

Global Context (`playwright.request`)



Інструментарій: Методи та Дані

GET

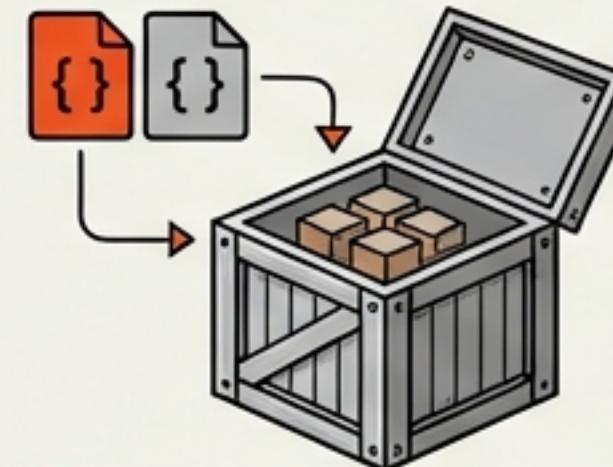
`https://api.example.com/resource?`

`params: { id: 123 }`
(Query String)



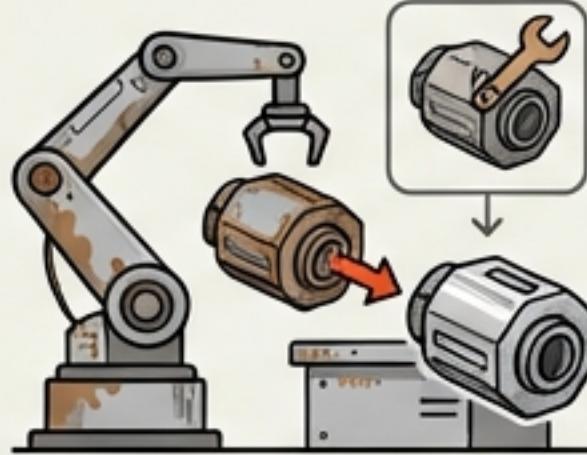
POST

`data: { json }`
(Body)



PUT/PATCH

Оновлення або
Заміна ресурсу



DELETE

Видалення
ресурсу

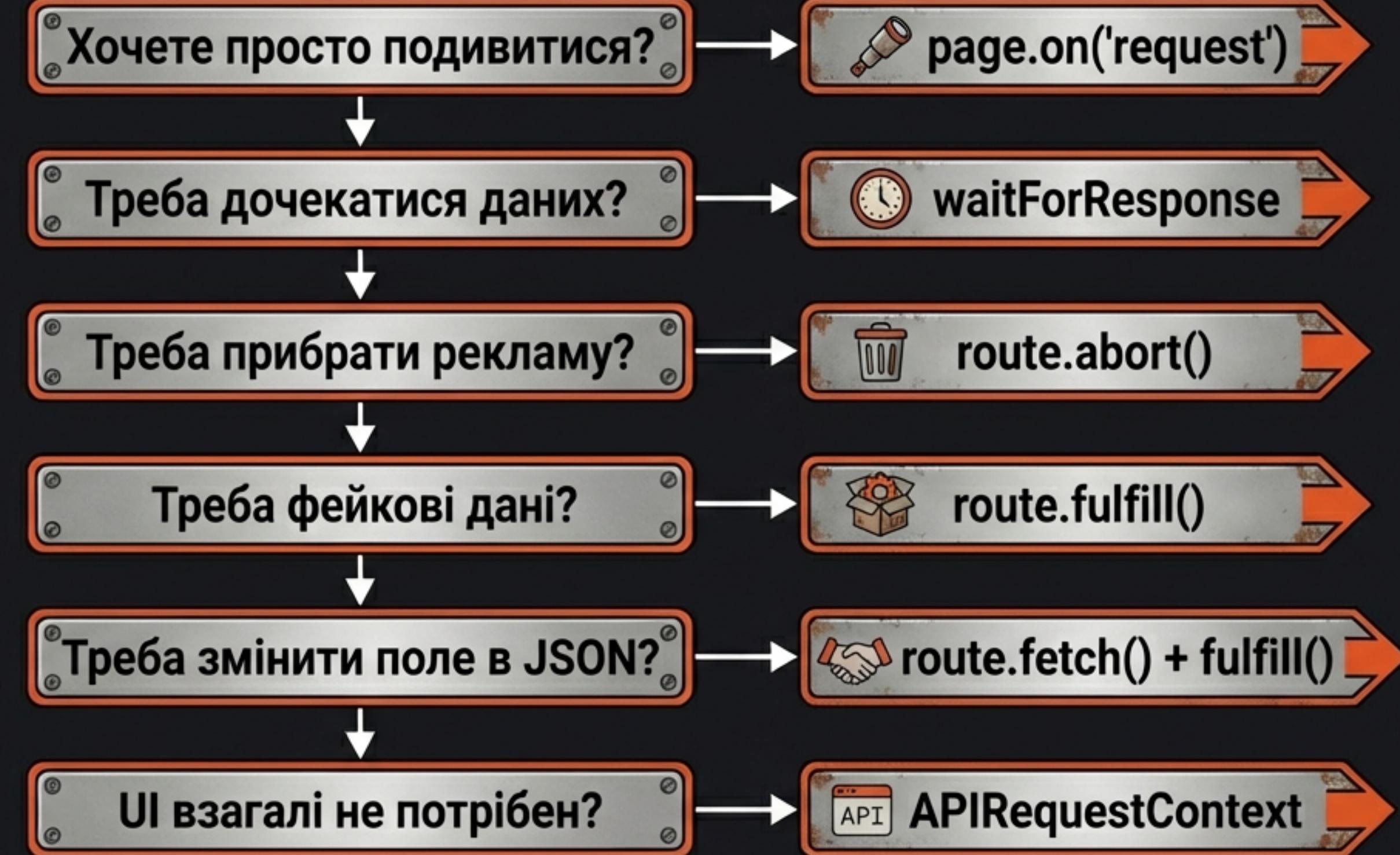


params = URL Query (`?id=1`) vs **data = Request Body (JSON)**

Гібридна стратегія: API як Setup для E2E

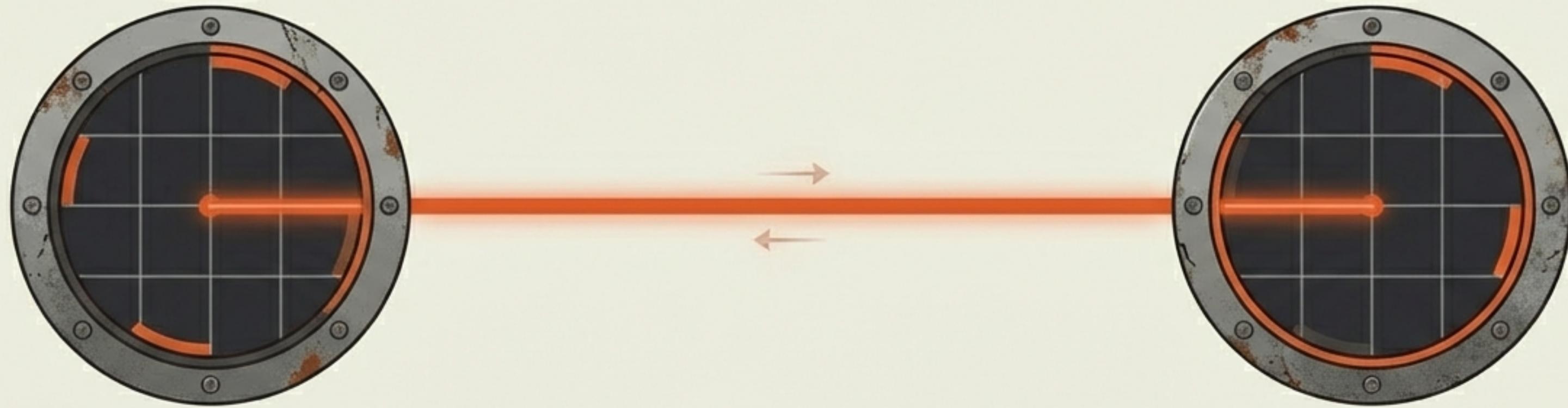


Матриця рішень: Що обрати?



Ви контролюєте мережу

Playwright надає інструменти – ви будете стратегією.



Експериментуйте з гібридними підходами для максимальної швидкості.