

Testování webů pro mírně pokročilé

Jakub Vokoun

jakub.vokoun@wftech.cz

Testing Humor 🐛

Murphy's Law of Testing: Anything that can go wrong will go wrong... but only after you deploy to production.

Q: How do you know if someone writes tests?

A: Don't worry, they'll tell you... repeatedly!

"It works on my machine!"

— Famous last words before production deployment

whoami (1)

- linuxák skoro již 20 let
- "liný DevOpsák, který automatizuje, co potká"

Motivace

- automatizace testů webových aplikací
- testy v reálném prohlížeči
- zátěžové testy
- CI/CD integrace

Kategorie testování

Testování	O co v něm jde
Load testing	Sledování chování při zátěži
Stress testing	Co ještě infrastruktura zvládne
Soak testing	Kontinuální load testing + monitorování
Spike testing	Load testing s náhlým zvýšením/snížením zátěže
Scalability testing	Jak systém škáluje s rostoucími požadavky
Configuration testing	Sledujeme vliv konfigurace při zátěži

Playwright

- jednotné API pro všechny podporované prohlížeče:
 - Chromium
 - WebKit
 - Firefox
- cross-platform:
 - Windows, Linux, nebo macOS
 - lokálně nebo CI
 - headless nebo headed

Playwright

- API pro několik rozšířených jazyků:
 - TypeScript
 - JavaScript
 - Python
 - .NET
 - Java

Playwright

NodeJS - instalace

```
npm init playwright@latest
```

```
npx playwright test
```

Playwright

NodeJS - ukázkový test

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

test('has title', async ({ page }) => {
  await page.goto('https://playwright.dev/');
  await expect(page).toHaveTitle(/Playwright/);
});

test('get started link', async ({ page }) => {
  await page.goto('https://playwright.dev/');
  await page.getByRole('link', { name: 'Get started' }).click();
  await expect(page.getByRole('heading', { name: 'Installation' })).toBeVisible();
});
```

Playwright

Python - instalace

```
pip install pytest-playwright
```

```
playwright install
```

Playwright

Python - ukázkový test

```
import re
from playwright.sync_api import Page, expect

def test_has_title(page: Page):
    page.goto("https://playwright.dev/")
    expect(page).to_have_title(re.compile("Playwright"))

def test_get_started_link(page: Page):
    page.goto("https://playwright.dev/")
    page.get_by_role("link", name="Get started").click()
    expect(page.get_by_role("heading", name="Installation")).to_be_visible()
```

Playwright

Úvod

Test začíná navštívením stránky

```
page.goto("https://www.linuxdays.cz/")
```

Pro interakci se stránkou se využívá lokátor

```
# Vytvoření lokátoru  
link = page.get_by_role("link", name="zde")  
  
# Interakce  
link.click()
```

Playwright

Lokátor

Akce	Popis	Akce	Popis
<code>locator.check()</code>	zaškrtnutí checkboxu	<code>locator.click()</code>	kliknutí na prvek
<code>locator.uncheck()</code>	odškrtnutí checkboxu	<code>locator.hover()</code>	hover
<code>locator.fill()</code>	vyplnění inputu	<code>locator.focus()</code>	focus
<code>locator.press()</code>	stisknutí klávesy	<code>locator.setInputFiles()</code>	výběr souboru
<code>locator.select_option()</code>	výběr ze selectu		

Playwright

Aserce

expect volání	expect volání
expect(locator).to_be_checked()	expect(locator).to_be_enabled()
expect(locator).to_be_visible()	expect(locator).to_contain_text()
expect(locator).to_have_attribute()	expect(locator).to_have_count()
expect(locator).to_have_text()	expect(locator).to_have_value()
expect(page).to_have_title()	expect(page).to_have_url()

Playwright

API ukázky

```
# Vyplnění inputu
page.get_by_role("textbox").fill("Ahoj vespolek...")

# Checkbox podle labelu
page.get_by_label("Souhlasím se zasíláním newsletteru").check()

# Select & label
page.get_by_label("Velikost").select_option(label="M")

# Drag & drop
page.locator("#item-to-be-dragged").drag_to(page.locator("#item-to-drop-at"))
```

Playwright

API ukázky

```
# Přihlášení
page = context.new_page()
page.goto("https://github.com/login")

page.get_by_label("Username or email address").fill("username")
page.get_by_label("Password").fill("password")
page.get_by_role("button", name="Sign in").click()

# Testy...
page.goto("https://github.com/jakubvokoun/my-secret-repo")
```

Playwright

API ukázky

```
# alert(), confirm(), ...
page.on("dialog", lambda dialog: dialog.accept())
page.get_by_role("button").click()
```

```
# Stažení souboru
with page.expect_download() as download_info:
    page.get_by_text("Stáhnout").click()
download = download_info.value

download.save_as("/tmp/" + download.suggested_filename)
```

Playwright

API ukázky

```
# Emulace zařízení
from playwright.sync_api import sync_playwright, Playwright

def run(playwright: Playwright):
    iphone_13 = playwright.devices['iPhone 13']
    browser = playwright.webkit.launch(headless=False)
    context = browser.new_context(
        **iphone_13,
    )

    with sync_playwright() as playwright:
        run(playwright)
```

Playwright

API ukázky

```
# Eventy
def print_request_sent(request):
    print("Request sent: " + request.url)

def print_request_finished(request):
    print("Request finished: " + request.url)

page.on("request", print_request_sent)
page.on("requestfinished", print_request_finished)
page.goto("https://wikipedia.org")

page.remove_listener("requestfinished", print_request_finished)
page.goto("https://www.openstreetmap.org/")
```

Playwright

API ukázky

```
# Screenshoty
page.screenshot(path="screenshot.png")
page.screenshot(path="screenshot.png", full_page=True)
```

```
# Video
context = browser.new_context(record_video_dir="videos/")
# Testy...
context.close()
```

Playwright

Další užitečné funkce a vlastnosti

Generování testů

```
playwright codegen demo.playwright.dev/todomvc
```

Debug testů

```
pytest --headed --browser firefox
```

Od funkcionálních testů k výkonostním

- **Playwright:** testuje co aplikace dělá
- **k6:** testuje *jak rychle* to dělá

Nyní se podíváme na zátěžové testování s k6...

k6

Co je k6?

- moderní nástroj pro zátěžové testování
- open-source, vyvinutý firmou Grafana Labs
- testy se píší v JavaScriptu (ES6)
- zaměřeno na vývojáře a DevOps týmy

k6

Klíčové vlastnosti

- **Performance:** optimalizováno pro vysokou propustnost
- **Developer-centric:** testy jako kód
- **CI/CD integrace:** snadná integrace do pipeline
- **Cloud & on-premise:** flexibilní nasazení
- **Rich metrics:** detailní metriky a reporting

k6

Instalace

```
# Linux (Ubuntu/Debian) - moderní způsob
curl -s https://dl.k6.io/key.gpg | sudo gpg --dearmor -o /usr/share/keyrings/k6-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/k6-archive-keyring.gpg] https://dl.k6.io/deb stable main" | sudo tee /etc/apt/sources.list.d/k6.list
sudo apt update
sudo apt install k6
```

```
# macOS
brew install k6
```

```
# Docker
docker run --rm -i grafana/k6 run - <script.js
```

k6

Základní test

```
import http from 'k6/http';
import { check } from 'k6';

export const options = {
  vus: 10, // virtuální uživatelé
  duration: '30s',
};

export default function () {
  let response = http.get('https://www.linuxdays.cz/');

  check(response, {
    'status je 200': (r) => r.status === 200,
    'response time < 500ms': (r) => r.timings.duration < 500,
  });
}
```

k6

Pokročilé scénáře - konfigurace

```
import http from 'k6/http';
import { check, sleep } from 'k6';

export const options = {
  stages: [
    { duration: '2m', target: 100 }, // ramp up
    { duration: '5m', target: 100 }, // stabilní zátěž
    { duration: '2m', target: 200 }, // zvýšení zátěže
    { duration: '1m', target: 0 },    // ramp down
  ],
  thresholds: {
    http_req_duration: ['p(95)<500'], // 95% požadavků pod 500ms
    http_req_failed: ['rate<0.1'],    // méně než 10% chyb
  },
};
```

k6

Pokročilé scénáře - testovací funkce

```
export default function () {  
  let response = http.get('https://api.example.com/users');  
  check(response, {  
    'status 200': (r) => r.status === 200,  
  });  
  sleep(1);  
}
```

Demo

Playwright ukázky

- **Headed režim:** vidíme prohlížeč v akci
- **Headless režim:** rychlejší běh bez GUI
- **Docker Compose:** kontejnerizované testovací prostředí
- **GitLab CI:** automatické spuštění testů při každém commitu

k6 + monitoring stack

- **k6:** generování zátěže
- **InfluxDB:** ukládání metrik v čase
- **Grafana:** vizualizace výsledků testů

Praktické použití

Kdy použít Playwright?

- **E2E testy** pro kritické uživatelské scénáře
- **Regresní testování** po každém release
- **Cross-browser testování** (Chrome, Firefox, Safari)
- **Visual regression testing** (screenshoty)
- **API testování** v kombinaci s UI testy

Praktické použití

Kdy použít k6?

- **Performance testing** před produkčním nasazením
- **Capacity planning** - kolik serverů potřebujeme?
- **SLA verification** - plníme požadavky na rychlost?
- **Monitoring** - pravidelné kontroly výkonu
- **CI/CD integrate** - automatická kontrola výkonu

Reference

- Playwright: <https://playwright.dev/>
- k6: <https://k6.io/>
- k6 dokumentace: <https://k6.io/docs/>
- Playwright dokumentace: <https://playwright.dev/docs/>