



PLAY RIGHT WITH PLAYWRIGHT

EASY UI TESTING

ELIZAVETA RAGOZINA

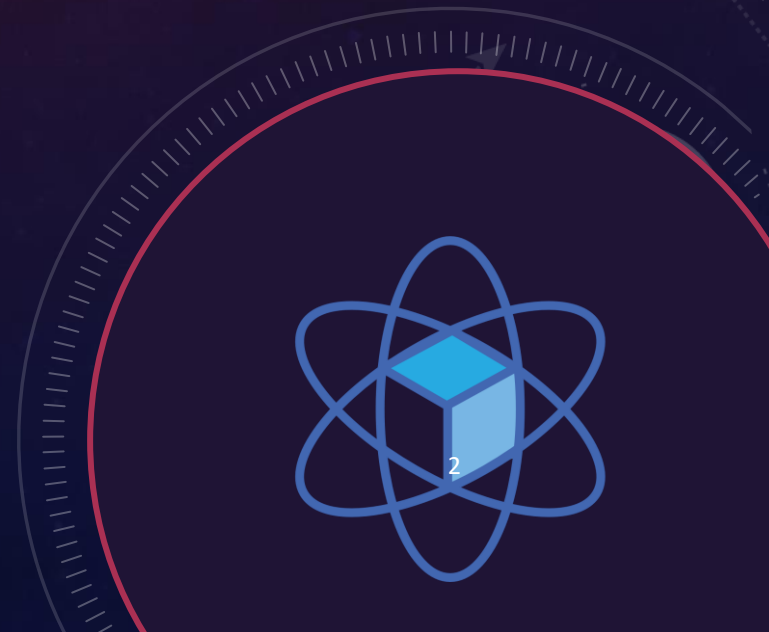
CERN IT LIGHTNING TALKS #24

ONCE UPON A TIME...

- We did time consuming manual testing of user flows
- Prone to errors

BUT THEN...

- We decided to automate
- Selected Playwright



ADVANTAGES OF PLAYWRIGHT



**Cross-
browser
support**



**Reliable and
fast**

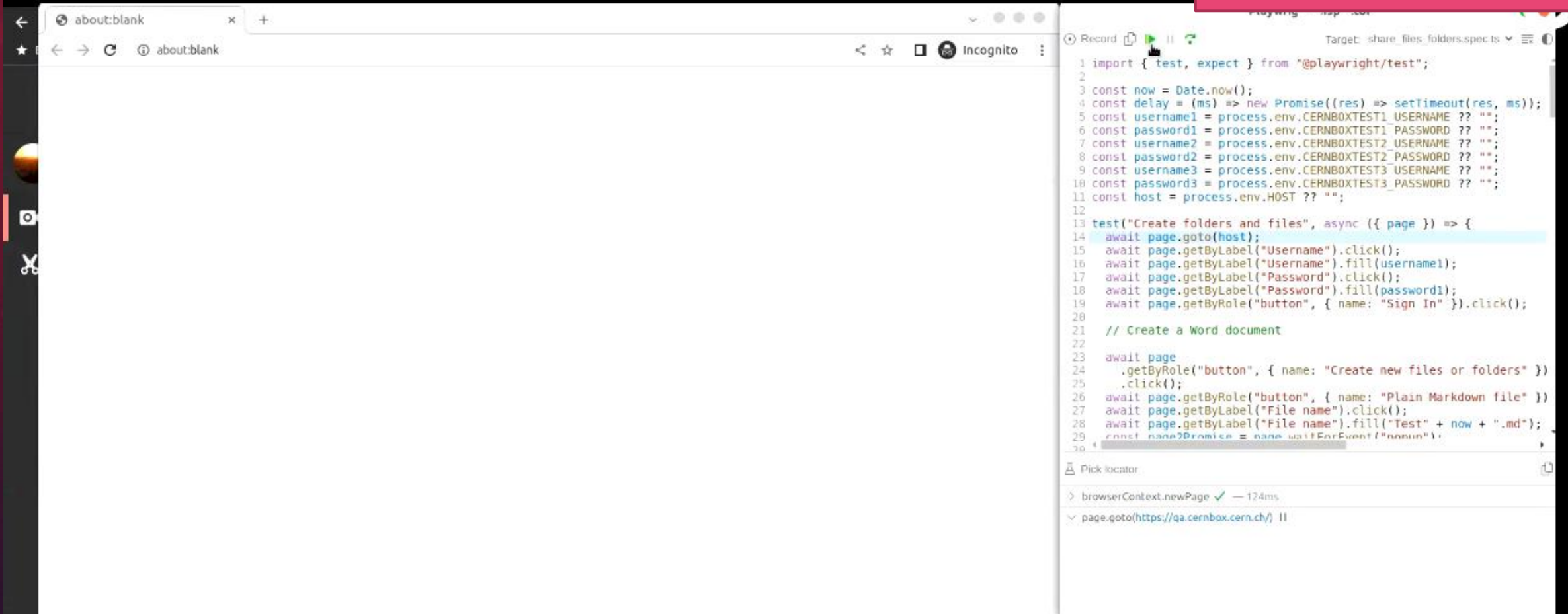


**Language
flexibility**



**Easy setup
and
maintenance**



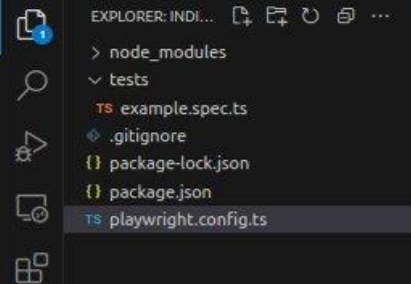


The image shows a side-by-side view of a web browser and a code editor. The browser on the left is Incognito mode, displaying an 'about:blank' page. The code editor on the right shows a Playwright test script for CERNBox. The script imports 'test' and 'expect' from '@playwright/test', sets up environment variables for usernames and passwords, and defines a test case 'Create folders and files'. The test case performs actions like navigating to the host, logging in, and creating a new file.

```
1 import { test, expect } from "@playwright/test";
2
3 const now = Date.now();
4 const delay = (ms) => new Promise((res) => setTimeout(res, ms));
5 const username1 = process.env.CERNBOXTEST1_USERNAME ?? "";
6 const password1 = process.env.CERNBOXTEST1_PASSWORD ?? "";
7 const username2 = process.env.CERNBOXTEST2_USERNAME ?? "";
8 const password2 = process.env.CERNBOXTEST2_PASSWORD ?? "";
9 const username3 = process.env.CERNBOXTEST3_USERNAME ?? "";
10 const password3 = process.env.CERNBOXTEST3_PASSWORD ?? "";
11 const host = process.env.HOST ?? "";
12
13 test("Create folders and files", async ({ page }) => {
14   await page.goto(host);
15   await page.getByLabel("Username").click();
16   await page.getByLabel("Username").fill(username1);
17   await page.getByLabel("Password").click();
18   await page.getByLabel("Password").fill(password1);
19   await page.getByRole("button", { name: "Sign In" }).click();
20
21   // Create a Word document
22
23   await page
24     .getByRole("button", { name: "Create new files or folders" })
25     .click();
26   await page.getByRole("button", { name: "Plain Markdown file" })
27     .click();
28   await page.getByLabel("File name").fill("Test" + now + ".md");
29   const pagePromise = page.waitForEvent("popup");
30 }
```

The code editor also shows a 'Pick locator' dropdown menu with the following options:

- browserContext.newPage ✓ — 124ms
- page.goto(https://qa.cernbox.cern.ch/) II



```
TS playwright.config.ts > [Ⓜ] default
1 import { defineConfig, devices } from '@playwright/test';
2
3 export default defineConfig({
4   testDir: './tests',
5   /* Run tests in files in parallel */
6   fullyParallel: true,
7   /* Fail the build on CI if you accidentally left test.only in the source code. */
8   forbidOnly: !!process.env.CI,
9   /* Retry on CI only */
10  retries: process.env.CI ? 2 : 0,
11  /* Opt out of parallel tests on CI. */
12  workers: process.env.CI ? 1 : undefined,
13  /* Reporter to use. See https://playwright.dev/docs/test-reporters */
14  reporter: 'html',
15  /* Shared settings for all the projects below. See https://playwright.dev/docs/api/c
16  > use: {
17    },
18  },
19
20  /* Configure projects for major browsers */
21  projects: [
22    {
23      name: 'chromium',
24      use: { ...devices['Desktop Chrome'] },
25    },
26
27    {
28      name: 'firefox',
29      use: { ...devices['Desktop Firefox'] },
30    },
31
32  ],
33
34  },
35
```

- What's Installed
- How to run the example test
- How to open the HTML test report

Installing Playwright

Get started by installing Playwright using npm or yarn. Alternately, you can use the VS Code extension.

npm **yarn** **pnpm**

```
npm init playwright@latest
```

Run the install command and select the following to complete the installation:

- Choose between TypeScript or JavaScript
- Choose the name of your Tests folder (default is 'tests')

THE BASICS

Locators

- CSS selectors
- `getByRole`,
`getByText`, ...

```
const locator = page.getByRole('checkbox',  
{ name: 'Subscribe' })
```

Actions

- `click`, `push`,
`focus`, `hover`,
`type`, ...

```
await locator.click();
```

Assertations

- `toBeVisible`,
`toBeEnabled`,
`toHaveCount`,
`toContainText`, ...

```
await expect(locator).toBeVisible();
```

LET'S WRITE A SIMPLE TEST FOR INDICO

USING THE CODE GENERATOR MAGIC

The screenshot shows a webpage for an IT Lightning Talks session. At the top, it says "IT Lightning Talks: session #24" with a date of Friday 9 Jun 2023, 10:00 to 11:00 in Europe/Zurich. The location is 31/3-004 - IT Amphitheatre (CERN). The speakers listed are Pedro Ferreira, Sebastian Lopienski, Hugo Gonzalez Labrador, and Alberto Di Meglio. A description explains that IT Lightning Talks (ITLT) are short presentations on computing technology or IT department topics, with a link to a wiki page. Below this is a videoconference section with a "Please log in" button. The main part of the page is a list of talks, each with a time slot, title, description, and speaker. The talks are: 10:00-10:03 Welcome (Hugo Gonzalez Labrador, Pedro Ferreira, Sebastian Lopienski); 10:03-10:10 How to build your mech keyboard (Javier Ferrer); 10:10-10:17 Ditching Vim for Kakoune (Robert Vasek); 10:17-10:24 Query your Cloud Infrastructure Interactively with Steampipe! (Jack Henschel); 10:24-10:31 Malware Intelligence analysis for better Windows protection @ CERN (Roman Sumailov); 10:31-10:38 Play right with Playwright: Easy UI testing (Elizaveta Ragozina); and 10:38-10:45 Open Source It Right (with REUSE and SPDX) (Giacomo Tenaglia). Each talk has a duration icon (clock) and a time value (3m, 7m, etc.).

IT Lightning Talks: session #24

Friday 9 Jun 2023, 10:00 → 11:00 Europe/Zurich
31/3-004 - IT Amphitheatre (CERN)
Pedro Ferreira (CERN) · Sebastian Lopienski (CERN) · Hugo Gonzalez Labrador (CERN) · Alberto Di Meglio (CERN)

Description IT Lightning Talks (ITLT) are short presentations on any topic related to computing technology or to the IT department. See more here: <https://twiki.cern.ch/IT/LightningTalks/>

Videokonferenz [IT Lightning Talks: session #24](#) [Please log in](#)

- 10:00** — 10:03 **Welcome** 🕒 3m
Speakers: Hugo Gonzalez Labrador (CERN), Pedro Ferreira (CERN), Sebastian Lopienski (CERN)
- 10:03** — 10:10 **How to build your mech keyboard** 🕒 7m
What if you want to build your own mechanical keyboard instead of buying one? For starters, you are guaranteed have more fun! In this talk I will give you some tips to avoid frustrations, and some info on tools and materials you will need.
Speaker: Javier Ferrer (CERN)
- 10:10** — 10:17 **Ditching Vim for Kakoune** 🕒 7m
Story of a Vim user switching a text editor, bringing back opinions and experiences.
Speaker: Robert Vasek (CERN)
- 10:17** — 10:24 **Query your Cloud Infrastructure Interactively with Steampipe!** 🕒 7m
Cloud infrastructures tends to have lots of moving pieces: containers, loadbalancers, virtual machines, databases etc. Steampipe is a tool that allows querying all these pieces through a single interface with SQL. Are you tired of writing brittle Bash and JQ scrips? Then this is the tool for you!
Speaker: Jack Henschel (CERN)
[Steampipe docs](#) [Steampipe plugins](#)
- 10:24** — 10:31 **Malware Intelligence analysis for better Windows protection @ CERN** 🕒 7m
Threatray is a company that proposes new way to identifying malware on the Windows PCs – by scanning memory of running processes, identifying "suspicious" regions and analyzing them to identify the malware, also in retrospect! CERN Security Team and Threatray found a mutually beneficial relationship to improve security posture of our Windows devices and put in place another mechanism for protecting us from possible threats.
Speaker: Roman Sumailov (CERN)
- 10:31** — 10:38 **Play right with Playwright: Easy UI testing** 🕒 7m
This talk showcases how Playwright can simplify UI testing for developers. Playwright is an open-source tool that allows developers to automate UI tests in multiple browsers and devices, making the testing process faster and more efficient.
Speaker: Elizaveta Ragozina (CERN)
- 10:38** — 10:45 **Open Source It Right (with REUSE and SPDX)** 🕒 7m
Do you care about properly licensing your code, or you just put it "out there in gitsomething"? We clarify a couple of key concepts on Open Source and present a set of guidelines and specifications you can adopt to help humans and machines interact with your code.
Speaker: Giacomo Tenaglia (CERN)

```
eli@eli-ZenBook-UX534FAC-UX533FAC: ~/dev/indico-ui-tests
eli@eli-ZenBook-UX534FAC-UX533FAC:~/dev/indico-ui-tests$ npx playwright codegen https://indico.cern.ch/event/1280138/
```

```
'synax=1121103.48&basemap=plan&mode=2D' );
```

```
'synax=1121108.94&basemap=plan&mode=2D' );
```

8

You have Docker installed on your system. Do you want to install the recommended extensions from Microsoft for it?

Install Show Recommendations

ADJUST GENERATED CODE

```
tests > TS example.spec.ts > ...
1 import { test, expect } from '@playwright/test';
2
3 test('test', async ({ page }) => {
4   await page.goto('https://indico.cern.ch/event/1280138/');
5   await page.locator('span').filter({ hasText: 'IT Lightning Talks: session #24' }).click();
6   await page.getByRole('button', { name: '<' }).click();
7   await page.getByText('Copied to clipboard').click();
8 });
```

Improve selectors,
Add comments, assertions

```
tests > TS example.spec.ts > test("Copy zoom meeting url") callback
1 import { test, expect } from "@playwright/test";
2
3 test("Copy zoom meeting url", async ({ page }) => {
4   await page.goto("https://indico.cern.ch/event/1280138/");
5
6   // Expand conference information
7   await page
8     .locator("span")
9     .filter({ hasText: "IT Lightning Talks: session #24" })
10    .click();
11
12   // Copy zoom url
13   // await page.getByRole('button', { name: '<' }).click();
14   await page.locator(".event-details-content .js-copy-to-clipboard").click();
15
16   // await page.getByText("Copied to clipboard").click();
17   expect(page.getByText("Copied to clipboard")).toBeVisible();
18 });
19
```

TEST REPORTS

Q	All 3	Passed 1	Failed 1	Flaky 0	Skipped 1	Total time: 23.3s
example.spec.ts						
✗ Copy zoom meeting url firefox						10.7s
example.spec.ts:3						
✓ Copy zoom meeting url chromium						9.8s
example.spec.ts:3						
Copy zoom meeting url webkit						0
example.spec.ts:3						

Copy zoom meeting url

example.spec.ts:3 10.7s

firefox

✗ Run

Errors

```
Error: expect(received).toBeVisible()
Call log:
- expect.toBeVisible with timeout 5000ms
- waiting for getByText('Copied to clipboard')

15 |
16 | // await page.getByText('Copied to clipboard').click();
> 17 | expect(page.getByText("Copied to clipboard")).toBeVisible();
    |                                             ^
18 |
19 | });

at /home/eli/dev/indico-ui-tests/tests/example.spec.ts:17:49
```

Test Steps

- ✓ Before Hooks 2.6s
- ✓ page.goto(https://indico.cern.ch/event/1280138/) — example.spec.ts:4 8.9s
- ✓ locator.click(span >> internal:has-text="IT Lightning Talks: session #24") — example.spec.ts:10 199ms
- ✓ locator.click(.event-details-content .js-copy-to-clipboard) — example.spec.ts:14 285ms
- ✗ expect.toBeVisible — example.spec.ts:17 26ms
- ✓ After Hooks 494ms

MORE FEATURES

- Parallelization
- Testing of APIs
- Testing browser extensions
- Mocking different network conditions
- ...

<https://playwright.dev>



The background is a gradient of dark red to dark blue. On the left side, there are several overlapping circular elements. A prominent one is a large circle with a scale around its perimeter, marked with numbers from 40 to 260 in increments of 10. Other circles contain smaller concentric circles, some with arrows indicating rotation. The right side of the image is filled with a bokeh effect of small, out-of-focus circles in various shades of blue and purple.

DO YOU HAVE ANY
QUESTIONS?