Playwright Visual Testing PDF

Playwright visual testing compares a new screenshot with a stored baseline using expect(page).toHaveScreenshot(). The first run creates the baseline, later runs fail on differences beyond a threshold. To keep results stable, mask dynamic areas, disable animations, fix the viewport, and generate baselines in the same environment as CI.

Key takeaways

- The first run saves a baseline image; later runs compare against it.
- Differences are controlled by thresholds such as maxDiffPixels.
- Mask or hide dynamic content such as dates, ads and random images.
- Baselines differ between operating systems and browsers, so generate them where CI runs.
- Review updated snapshots in pull requests like any code change.

What is Playwright visual comparison testing?

Visual testing in Playwright is the automated comparison of screenshots against approved baseline images to detect unintended changes in layout, colour or appearance of a page or component.

What does visual testing catch?

Functional tests confirm that a button works. They do not notice that its text now overlaps the logo, that a colour changed, or that a section moved off screen. Visual tests compare how the page looks with an approved image, so unplanned appearance changes show up as failures in the pull request.

How do I write a basic visual test?

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

test('home page looks right', async ({ page }) => {
  await page.goto('/');
  await expect(page).toHaveScreenshot('home.png', { fullPage: true });
});
```

On the first run the test fails and saves `home.png` as the expected image. Check the image by eye, commit it and later runs compare against it.

How do I update baselines?

When a design change is intended, run `npx playwright test --update-snapshots`, view the changed images and commit them with the code change. Reviewers should open the images in the pull request. Never update snapshots blindly to turn a build green.

What options control the comparison?

Option	Purpose
`maxDiffPixels`	Allowed number of differing pixels
`maxDiffPixelRatio`	Allowed share of differing pixels
`threshold`	Colour difference tolerance per pixel
`animations: 'disabled'`	Stop animations during capture
`mask: [locators]`	Cover dynamic areas with a box
`clip`	Capture only a region
`fullPage`	Capture the whole scrollable page
Start strict and relax only for a stated reason.	

How do I handle dynamic content?

Mask clocks, user names, random avatars and ads: `mask: [page.locator('.timestamp')]`. Mock the network so data is fixed. Use a fixed date with the clock API where available. Wait for fonts and images to load by asserting a visible element first. Without these steps, tests will flicker.

Why do snapshots differ between machines?

Fonts, rendering engines and operating systems draw text slightly differently, so a baseline made on a laptop may fail on Linux in CI. Playwright adds the browser and platform name to snapshot file names. Generate and update baselines in the same container image or runner type used for CI, or run visual tests only in the pipeline.

Should I snapshot whole pages or components?

Component-level screenshots with `expect(locator).toHaveScreenshot()` are smaller and less sensitive to unrelated changes. Whole-page images are useful for key templates. Choose a limited set of stable, valuable pages, rather than screenshotting everything, because every image has a review cost.

How do I keep the suite maintainable?

Name snapshots clearly, group them in a folder, limit the viewport sizes to those you support, and remove obsolete images. Document how to regenerate baselines in the README. Track how often

visual tests fail for real versus noise and adjust.

What are the limits of visual testing?

It shows that something changed, not whether it matters, and it does not replace functional or accessibility tests. Large pages with frequent content changes produce noise. Combine it with design review and consider dedicated visual review tools if your team needs approval workflows.

How do I review visual changes in a pull request?

Open the old and new images side by side, along with the diff image that Playwright produces on failure. Ask whether the change matches the design request, whether other pages were affected and whether the change looks right on every supported viewport. Add a short note in the pull request describing which snapshots changed and why, so reviewers do not need to guess. Treat unexplained differences as defects until proven otherwise.

Which pages deserve visual tests first?

Choose pages where appearance carries business value and which change rarely: the home page, the pricing page, the login form, a product card and the checkout summary. Avoid pages dominated by live data. Start with five to ten snapshots across two viewports, measure how often they produce noise, and expand only when the suite remains calm. A small, trusted visual suite is more useful than a large one people learn to ignore.

How do I combine visual and accessibility checks?

Run visual comparison for layout, then add role-based locators and an accessibility scan for names, contrast and keyboard order. The visual test says the page looks the same; the accessibility checks say it still works for assistive technology. Together they catch a wider range of regressions than either approach alone, and both can run in the same pipeline without much extra effort.

What is a short checklist before enabling visual tests?

Fix the viewport, disable animations, mock dynamic data, decide the allowed difference, choose the baseline environment and write down how to update images.

Frequently asked questions

How does Playwright visual comparison work?

It takes a screenshot and compares it to a saved baseline image, failing the test if differences exceed the allowed threshold.

How do I update snapshots after an intended change?

Run the tests with the update snapshots flag, review the changed images and commit them together with the code change.

Why do screenshots fail only in CI?

Fonts and rendering differ across operating systems, so create baselines in the same environment that runs the tests in the pipeline.

How do I deal with dates and ads?

Mask those areas, mock network data and disable animations so that the captured page is the same on each run.

Can visual tests replace functional tests?

No. They detect appearance changes only, so use them alongside functional and accessibility checks.

Sources and further reading

- Playwright documentation
- Playwright visual comparisons
- Playwright test assertions

About this PDF

These PDFs are study material for learning and revision. They are not a certificate, a degree or a promise of a job, and technology changes, so check official documentation for the latest details. Spot an error? Write to info@uncodemy.com and the trainer team will review it.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550