testcraft
by Perforce

AUTOMATION NOW

How to Automate Web Tests Without Hiring, Firing, or Waiting Forever

How to Automate Web Tests Without Hiring, Firing, or Waiting Forever

Accelerating release velocity? It just makes sense. Every new feature and enhancement drives opportunities to attract and delight more customers.

Bugs, however, don't delight anyone. Particularly if they tip over customers' online shopping carts or reject their online payments.

Unfortunately, the proliferation of devices and browsers has made it nearly impossible to say with confidence that these kinds of show-stopping bugs aren't smuggling themselves into any given release.

One way to increase confidence is to increase testing. But, for organizations relying on manual testers, more testing lengthens release cycles, and decreases opportunities to attract and delight customers.

The impact of this very problem generated a tsunami of codeless test automation tools in recent years. Initial entrances to the field were simple record and playback tools. The promise was high. Delivery fell short. The tools were buggy and tests often broke when code changed.

To truly automate testing, then, you had to refactor your team and hire an automation engineer or two who could script your Selenium tests. When those scripts break, which they do, they must be fixed. It takes a long time. It is costly. And it is unproductive.

What if you don't want to disturb your existing team? And you need automation ... now?

How to Automate Web Tests Without Hiring, Firing, or Waiting Forever

In this eBook, we'll introduce you to a next generation of codeless test automation. You'll learn how it helps teams get going quickly with test automation while avoiding record/playback issues, the need for high-cost coders, and the stifling inefficiencies of Selenium test maintenance. For teams that want to move faster with automation, the solution provides the tools they need to generate and maintain quality test scripts.

Let's get a better understanding of what codeless test automation is, what tests it can be used for, and how it can help your organization.

We'll introduce next-gen codeless testing in five sections:

Part One: What Is Next-Gen Codeless Test Automation?
Part Two: Going Codeless: Which Tests Are Best?
Part Three: What About Selenium? A Comparison
Part Four: Automating in the Cloud vs On-Premises
Part Five: 4 Steps to Starting With Codeless Automation

Part 1
What Is Next-Gen Codeless Test Automation?

Codeless test automation is a low-friction path to automation. With it, you can:

- Empower manual testers to automate tests.
- Relieve development teams of burdensome automation duties.
- Give everyone focus to work on what they're paid for: delivering business value.

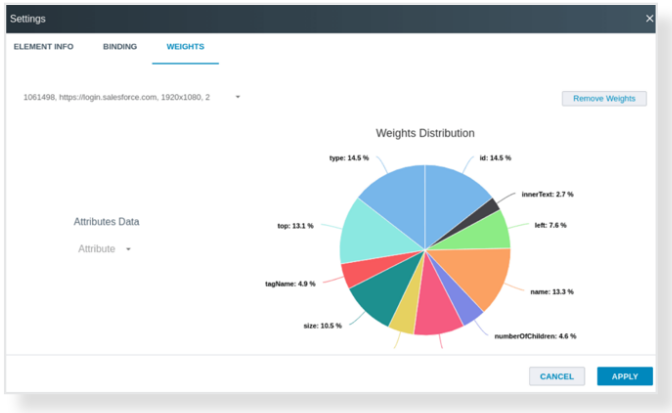
When we talk about the *next generation* of codeless test automation, we're addressing how the emerging capabilities of AI and ML are improving the quality and ease of automation.

For example, in TestCraft, AI is used to drastically reduce the number of flaky tests and minimize maintenance. The AI uses what's called a smart-binding algorithm that automatically heals 97% of the changes that make tests break. Scripts run continuously while fixing themselves without interrupting the team.

CANDIDATES FOR NEXT-GEN CODELESS

Business testers and other manual testers (even developers) are ideal candidates for this new kind of codeless test automation. Because while codeless testers don't need to know how to code, they do need a decent understanding of testing and have insight into the application, its business goals, and user pain points.

Importantly, the AI isn't replacing anyone. It only enhances the existing team.



AI/ML in TestCraft applies weights to various locators. So flaky tests heal themselves.

Part 2
Going Codeless: Which Tests Are Best?

For many organizations, codeless test automation is best applied to **continuous testing or regression testing**. These kinds are often repetitive and time-consuming, making them ideal for automation.

But there are several types of tests that can – and should – be written and executed codelessly. To figure that out, look around for:

- Manual tests that are hard to implement or require greater in-house technical skills.
- Flaky tests that are consistently inconsistent (e.g. pass and fail for the same configuration or result in false negatives).

FLAKY TESTS THAT CODELESS CAN FIX

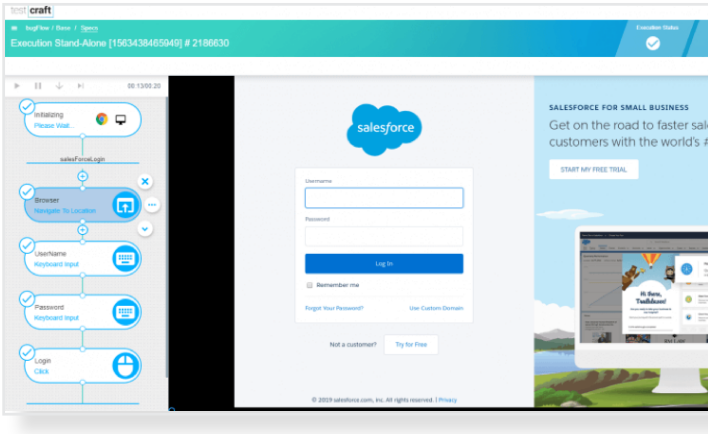
Typical flaky tests are those that look at web pop-ups for unconfigured image elements or the effects of mobile behaviors, such as certain devices being blocked by certain applications.

COMBINING CODED & CODELESS

Organizations that plan to use a combination of coded and codeless tests should make sure they work together. A good practice is to exclude the tests that will operate codelessly from the actual code itself. Apply the codeless test and then bring it back into the sequence.

Lastly, keep one thing in mind: **codeless automation will never completely replace all manual testing**. There will always be tests that need the human touch or close monitoring. Traditionally, these are tests with third-party dependencies and complex prerequisites for setting up.

Test on the left. Site on the right. Next-gen codeless lets users create tests visually, without code.



Part 3
What About Selenium? A Comparison

Selenium has been the de facto choice for automating web tests. So how does it compare to next-gen codeless automation tools for web applications?

If your organization already has dedicated resources for Selenium-scale automation, then it might be a better choice. However, many QA organizations have experienced challenges with Selenium and turned their gaze toward codeless Selenium tools.

	Selenium	Next-Gen Codeless for Web
Setup	Complex — start in weeks/months	Simple — immediate start
Personnel	Hire test engineers	Use existing team
Test Creation	6 hours/test — requires coding	1 hour/test — visual creation
Test Modification	Hours — requires coding	Minutes — during runtime
Maintenance	Heavy — tests break when app changes	Minimal — AI prevents 97% of tests from breaking
Test Execution	Local	Cloud

Part 4
Automating in the Cloud vs On-Premises

Codeless test automation was historically achieved with downloaded, on-premise technologies. By nature, this technology makes the users and IT (who need to make sure everything's installed properly and up to date) reach for an aspirin. On-premise codeless test automation technology encounters issues around speed, scalability, and the new demands of work-from-home collaboration.

SCALABILITY

Managing a Selenium grid locally as new browser versions are released is close to impossible, far from inexpensive, and, in most cases, causes big delays in test executions. Also, scaling a grid with hundreds of permutations locally is flaky, unreliable, and difficult compared to a cloud SaaS solution, which offers unlimited scalability. There's also little setup required for the team. With a cloud-based solution, they can meet strict project timeline requirements and expand test coverage.

SPEED

Cloud testing solutions also enjoy the benefits of strong machines, robust infrastructure, and networking that can boost test automation execution time and performance significantly. This contributes to the overall performance and speed of testing. Local solutions fall short in supporting multiple geographies that are, in many cases, necessary in web and mobile-web testing.

WFH COLLABORATION

Modern testing is all about collaboration. And digital collaboration has never been more essential than it is today, when many teams are or could be working from home. When comparing a collaborative cloud environment to several local desktops, it is noticeably clear that the benefit of cross-team collaboration disappears or declines significantly. Sharing test data and test scenarios is essential to reducing overhead in recreating tests.

The cloud empowers multiple users to make the most of existing resources and see a teammate's executions on demand.

Part 5
4 Steps to Starting With Codeless Automation

If you want to improve the quality of your web application through codeless automation, here are your four first steps to success.

1. GET BUY-IN

To get buy-in, you'll need to reframe your thinking and enter the perspectives of everyone involved — from management to practitioners.

What does ROI look like to each of your stakeholders? Make it real.

For management, the ROI can be best framed in terms of cost savings, product quality, and risk mitigation. How much time can you save? How much better is the coverage? Take a high priority page. What's the cost of downtime if a new feature release breaks something you didn't even know you should have tested?

For the daily practitioners, help them understand that the inconvenience is temporary, that they will be able to take the repetitive and boring tasks out of their day, and that having experience with test automation will only bring good things to their career opportunities.

There's ROI across the organization. Test automation means faster release velocity which means more value for customers. Those customers will encounter far fewer surprises from bugs, human errors, or escaped defects because test automation also gives you greater quality coverage across your development lifecycle.

2. DECIDE WHAT TO AUTOMATE & WHO WILL DO IT

Any new process comes with a learning curve. For teams just starting with codeless test automation, there is a broad range of things to wrap their minds around: from figuring out which tests to automate to deciding who will do the automating.

Do you have the resources and skills you need? Is there an existing testing team? Do you have budget to hire an SDET to help code the changes? Or do you need to move forward with an existing team of manual testers? All are important questions to answer.

Part 5

3. PICK A CODELESS TEST AUTOMATION TOOL

For those entering the world of codeless test automation, getting the tooling right is essential. There are many kinds of tools, from legacy and soon-to-be-obsolete record and playback tools all the way to codeless test automation tools using AI.

Pick what best serves your team's purpose now but don't forget the long-term. Is it easy to use? Does it work in all browsers? Does it have good support? Does it integrate with CI/CD tools? Tooling needs to address your challenges, serve your strengths, and integrate with your current and future needs.

4. START SMALL, FAIL SMALL, LEARN FAST

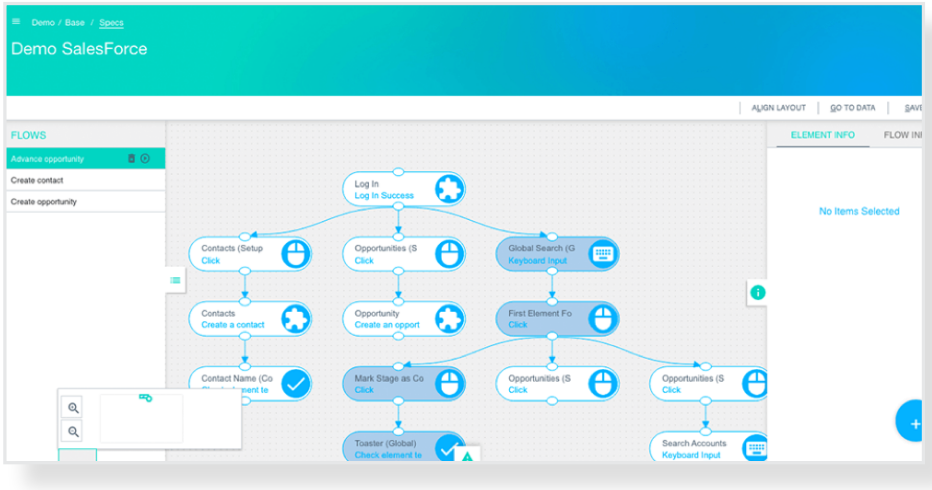
The glimmer of full-on test automation is bright. But don't be blinded by planning too much too soon. Starting small is a best practice for nearly any and every business initiative. It holds especially true for codeless test automation.

Give yourself small goals. Automate one test case. Work out any kinks and bring that wisdom forward.

It's much easier to replicate and scale small successes than it is to isolate exactly what broke amid the infinite variables and dependencies common to a large rollout.

Pro Tip: Build your tests modularly, so that you can save time by reusing components. It's also easier to fix changes. If the app changes materially, you know exactly what area of a test to change and need only make the change once.

With next-gen codeless, reusable test flows can be written by dragging and dropping elements.



Part 5

NOW IS THE TIME FOR AUTOMATION

Automation is inevitable. Organizations that continue to delay are only holding themselves back. To enter the world of test automation quickly, look to the next generation of codeless test automation tools that offer fast implementation, improved productivity through AI/ML, and fewer testing bottlenecks.

Teams that take a smart approach to codeless test automation will ultimately benefit from stabilized pipelines, better collaboration between teams, and increased test automation coverage.

Now that you've learned all about codeless test automation and what it can do for you, be sure to check out TestCraft. It's a next-generation codeless testing platform that lets you create tests up to six times faster.

>> Request a TestCraft free trial.
>> Watch a TestCraft demonstration.

ABOUT TESTCRAFT BY PERFORCE

TestCraft (now a Perforce company) is a Selenium-based codeless test automation platform for companies seeking a SaaS solution to streamline and scale development of high-quality web products. With TestCraft, testers of any skill level can quickly build and reuse tests while AI-based self-healing technology reduces maintenance overhead time by preventing tests from breaking. With TestCraft, organizations can:

- Improve productivity & agility by automating time-intensive manual tests.
- Cut maintenance time with AI that automatically fixes 97.4% of changes that occur.
- Drag and drop elements to intuitively create reusable test flows.
- Find and fix issues faster, with less effort, through detailed reports.

To find out how TestCraft can add value to your team's test creation and execution process, visit TestCraft.io.