How & When: Going from Manual to Automated Testing
Contents

3 | Introduction
4 | Uses of Manual Testing
5 | Moving Towards Automation
6 | Choosing an Automation Tool
8 | Steps for Moving to Automation
9 | An Automation Strategy to Take You Into the Future
Introduction

As development teams strive to push new code and features out to customers faster than ever, many are curious about the advantages of automation.

But those development teams may not know where to start. If you’re considering a shift to automation, or don’t know where to begin, keep reading. We’ll help you learn more about when to automate, and how to do it successfully.
Manual Testing is performed step-by-step by a human. Rather than using automated computer scripts, it requires someone to go through the application with a mouse, keyboard, or fingertip (for mobile devices).

In this way, the manual tester acts as the end user to advocate for quality. Whether visual or functional, this uses human observation to find any overlooked bugs.

**Skills required:** Manual testers should have a working knowledge of the application. But no coding knowledge is necessary.

**Manual testing is good for:**
- One-off and ad-hoc testing
- Exploratory testing
- Visual testing
- Usability testing
- Debugging

Automated testing uses tools, scripts and software to perform test cases by repeating predefined manual actions.

The biggest draw of automated testing is that it’s significantly faster and allows multiple tests to run at once. This allows larger test coverage across environments and larger code coverage during the testing phase. Running many tests side by side like this is often referred to as parallel testing.

New browsers, operating systems, and devices hit the market constantly. But with automation, we have a faster way to repeat the same test cases against all sorts of configurations. Without having to manually repeat the same test automated testing also reduces human error.

**Skills required:** To automate, you’ll need a test automation tool. There are various options, and depending on which you pick, you’ll likely need to learn new skills such as programming languages and frameworks. We’ll cover that more as we continue.
If you find a lot of bugs and defects when manually testing the application, your process likely isn’t suited for automated testing just yet.

Remember, automated testing only tests what’s explicitly predefined in the script, so the application should have a standard UI. Testers also need a solid understanding of the application’s strengths and weaknesses so that nothing falls through the cracks.

So when is automation more efficient? Its usefulness becomes clear when developers need to commit code often, or when testers must work to ship quickly on a stable product. It’s also helpful because automated testing lets the team test before, during, and after deploying web changes.

If your team members find themselves integrating often and adding regressions to requirements, it may be time to add automation. You’ll reduce time spent on repetitive tasks and increase code coverage, making it easier to meet deadlines.

When is the right time to automate?

- Regression tests
- Repetitive functional tests
- User states
- Tests prone to human error
- Browser testing / parallel testing
- Data-driven testing

Summing up: Automation becomes a great option when you need to repeat multiple test configurations.
Choosing an Automation Tool

There are a few key areas of functionality you'll want to evaluate when it comes to selecting a test automation tool. Base this decision on your organization's technologies and projects, as well as the systems and resources already in place.

There are also many other open source and paid add-ons that you may want to invest in. They make automation easier, faster, and more enjoyable for testers. Many test automation frameworks, for example, are free and much easier to implement than building one from scratch.
Here are some more popular automation tools:

**Record and Replay**
- Requires no scripting knowledge: simply point, click, and re-run your test
- Ideal for individuals with little-to-no programming knowledge, non-technical roles, and manual testers
- Lightweight method to fill in the gaps of your manual testing or Selenium strategy
- Solution for smaller test cases
- Can help teams where members outside of QA take part in some testing

**Scripting**
- Your options include one of the most popular open source automation tools known as Selenium
- Selenium WebDriver allows users to automate test scripts written in any language, across different browsers and operating systems.
- Requires programming knowledge, framework knowledge, and learning Selenium commands
- Good for technical experts that know the ins and outs of writing a script, or teams with the resources and time to learn
- Robust solution for end-to-end testing
- Largest reach of automation speed and code coverage
- Selenium can pair with third-party cloud tools to maintain a Selenium grid

**Cloud Tools**
- Cloud tools such as CrossBrowserTesting give you added capabilities on top of Selenium
- One of the major benefits is not having to host your own device lab, or having to maintain your own Selenium grid
- Test in parallel (run tests at the same time)
- Less maintenance, cost, and time spent on the upkeep of real devices and virtual machines
Steps for Moving to Automation

1. Begin with a **business goal** where automation will prove useful
2. Start with **small test cases** that are easy to maintain & reusable
3. **Identify** which high-priority tasks are taking up the most time
4. **Decide** which browsers, operating systems, and devices you need to test
5. Know what not to automate and the **limitations of your tool**
6. Get the **entire team on board** and encourage more cross-team communication
7. Make the effort for **continuous learning**
Automation is not a one-and-done task. It takes maintenance to code tests, update scripts, and create reusable suites. Not to mention the experience to identify when automation will help.

You’ll also need to refine tests over time to meet the changing needs of your team. To be successful, you have to spend time maintaining and creating new ones as needed. And as noted above, as you adopt automation, it’s important to remember the ongoing need for manual testing.

It’s impossible to automate everything. And even if you could, you shouldn’t. There’s a time and place for automation as much as there is for manual testing. And of course, automated testing doesn’t mean manual testing should be disregarded. It’s a way to supplement time-consuming, error-prone tasks so that you have more time to execute new tests.

The trick to being successful with automation is knowing how and when. It takes a healthy dose of human observation, curiosity, and insight. If you can recognize where automation helps vs. when you should stick with manual testing, you’ll be well on your way to successfully adding it to your process.
Make the **step to automation** easy.

Run manual, visual and Selenium tests in the cloud on 2,050+ real desktop and mobile browsers. It's easy to implement, and shows exactly how your website will look on any device.

[Check Out CrossBrowserTesting](#)