a1qa Test automation 101 User Guide

a1 qa | Software Testing Company

Test automation 101
The complete guide
eBook

Contents

1 Overview of test automation
2 To automate or not to automate?
3 What's the difference?
4 Common automation tests
5 Specialized test automation from a1qa
6 Choosing the right test automation tool
7 5 benefits of test automation
8 5 steps to kickstart test automation
9 5 top tips for optimization
10 Future-proof your testing
11 Documents / Resources
11.1 References
12 Related Posts

Overview of test automation

What is it, and how does it work?

The goal of automated testing isn’t to replace manual testing completely — it’s to automate the monotonous, repetitive tasks that take up time. The key is knowing when to automate. Learn more

In the age of agile and emerging technology, our world is becoming increasingly digital and fast-paced. Users want smarter products with more exciting new features and quick system updates — and leadership teams are eager to give it to them to stay ahead of the competition. To keep on track, QA testing often falls into the hands of the developers, taking up precious time for innovating and improving the software. On top of shorter development cycles and tighter release deadlines, they now face the unique challenge of having to complete complex testing tasks outside their specialization. As a result, the traditional and hands-on approach to testing is quickly becoming obsolete, with many organizations turning to automation to free up development time and speed to market.
Test automation is an empowering tool, but weeding through the onslaught of offers and information out there can be overwhelming. This little book of hacks is designed to help you build a powerful automation strategy and optimize your existing processes with ease.

To automate or not to automate?

Learn what tests should be automated and when to stick with manual testing.

Automate your…

Repetitive tests
Ex: Testing the same features repeatedly

Time-consuming tests
Ex: Checking functionality after modification

Continuous testing
Ex: Continually checking for defects early and often

Manually test your…

User interface (UI)
Ex: Checking button visibility on mobile devices

User experience (UX)
Ex: Ensuring usability with a target group

Exploratory testing
Ex: Investigation and discovery without test cases

What’s the difference?

Find out how manual and automated tests are implemented and executed.

Automated

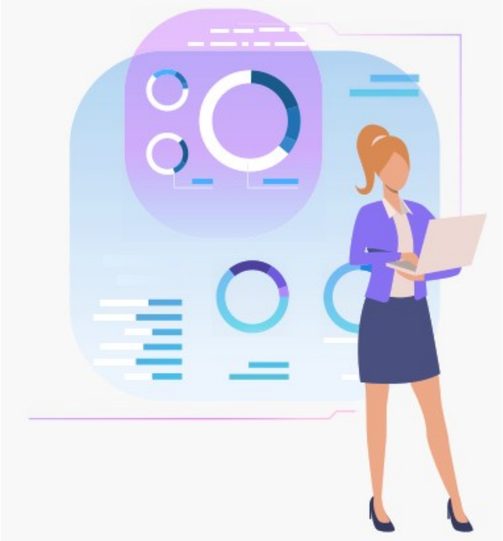
- Executed by an automation tool
- Multiple tests done in parallel
- Uses coded scripts
- Scripts are stored for reuse
- Increased test coverage
- Reports automatically generated

Manual

- Executed by a QA analyst
- Tests done one by one
- Data fields entered individually
- Actions must be repeated
- Limited to certain devices and OSs
- Reports manually written

Common automation tests

See which test cases are often automated to avoid repetitive or time-consuming tasks.



Unit Tests the individual components of an application **Smoke** Checks the stability of the build **Black box** Searches for incorrect or missing functions	**Integration** Integrates and tests application modul **Functional** Ensures all functions meet expectatior **Regression** Checks that existing features function response to code changes

Specialized test automation from a1qa

Discover a new approach to test automation with our specialized test cases.

Performance Checks system stability, processing power, and efficiency	**Usability** Identifies weak spots in usability and develops improvements	**Cybersecurity** Assesses security and vulnerability to boost protection	**Compatibility** Ensures cross-browser and platfo compatibility

Choosing the right test automation tool

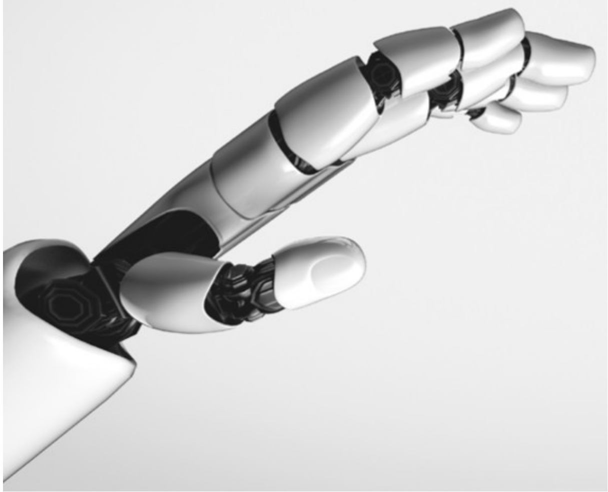
You have your choice of tools out there, but it’s important to find a scalable solution suited to your system.

Quality Automation from a1qa: Develop, execute, and analyze automated and manual tests in one single, customized interface.
Explore the continuous testing toolkit

Common test automation tools:

- Selenium: Web browser testing
- Appium: Mobile app testing
- Cucumber: Behavior-driven development testing
- Ranorex: Desktop, web-based and mobile testing
- TestComplete: Automated UI testing
- Microfocus UFT: End-to-end functional testing
- Apache JMeter: Functional and performance testing
- Tosca: Continuous testing

5 benefits of test automation

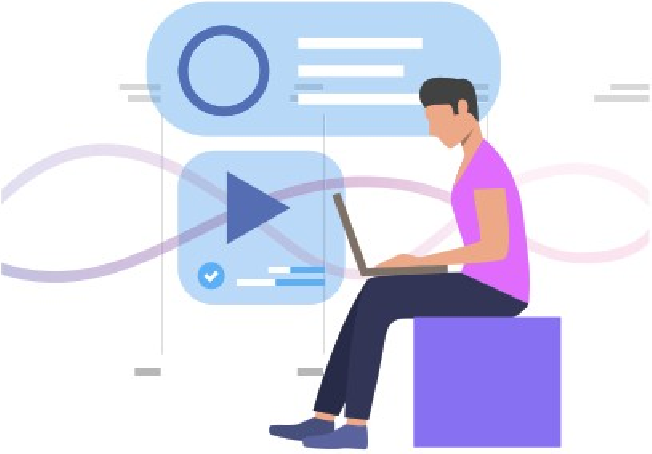


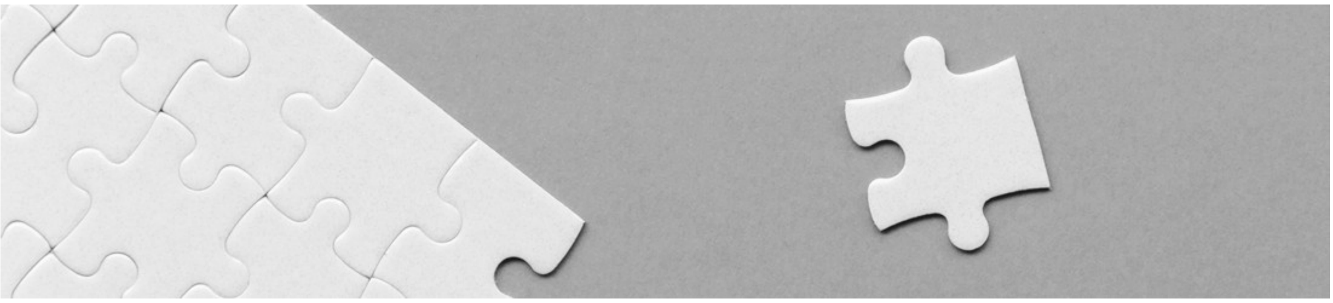
Need to convince your leadership team that it's time to automate?
Find out how your software development process could benefit from working better, faster, and smarter with test automation.

1. Low TCO, high ROI

   Automation means less demand for specialized QA members, fewer manual tasks, and more time for discovery and innovation — lowering your TCO and boosting ROI.
2. Faster time-to-market

   With the ability to quickly run continual tests, new or special features can be developed, iterated, and validated at a faster rate, speeding up their release time.





3. Improved product quality

   Running consistently accurate tests eliminates human error and detects earlier in the testing cycle, improving overall functionality and enhancing the customer experience.
4. Increased productivity

   When manual repetitive tasks are automated, your QA team is awarded more time and energy to focus on value-adding tasks, such as testing a user experience and performing exploratory checks.

5. Optimized test coverage

   Free from the limitations of manual testing, automation allows you to create and add new scripts to your suite, giving you the power to test new features or complex applications without worry

5 steps to kickstart test automation

Not sure where to start?
Follow these simple steps to design a holistic strategy and set up the right tools and processes successfully.

1. Create a solid strategy

   Before you hit the ground running, outline your vision and testing scope. Keep in mind that automation is ideal for long-term projects, bringing in the highest value for those from 6-months and onward. Which test cases would deliver the greatest benefits to the business through automation? Consider tests that have been difficult to manage, demand burdensome updates, or take up a lot of valuable time and effort to run. Set realistic goals and make sure your architects know what to automate and how to get started.
2. Find the right tool for you

   Web or mobile? Java or Ruby? With so many tools out there, it's best to take note of your software's operating system and platform, and your QA testing team's skills and capabilities. Can they understand and manage a more advanced programming language, or is there a simpler tool that covers all the features you want? A single interface for all your testing needs, like Aquality Automation, is a great IT solution.
3. Build high-quality test data

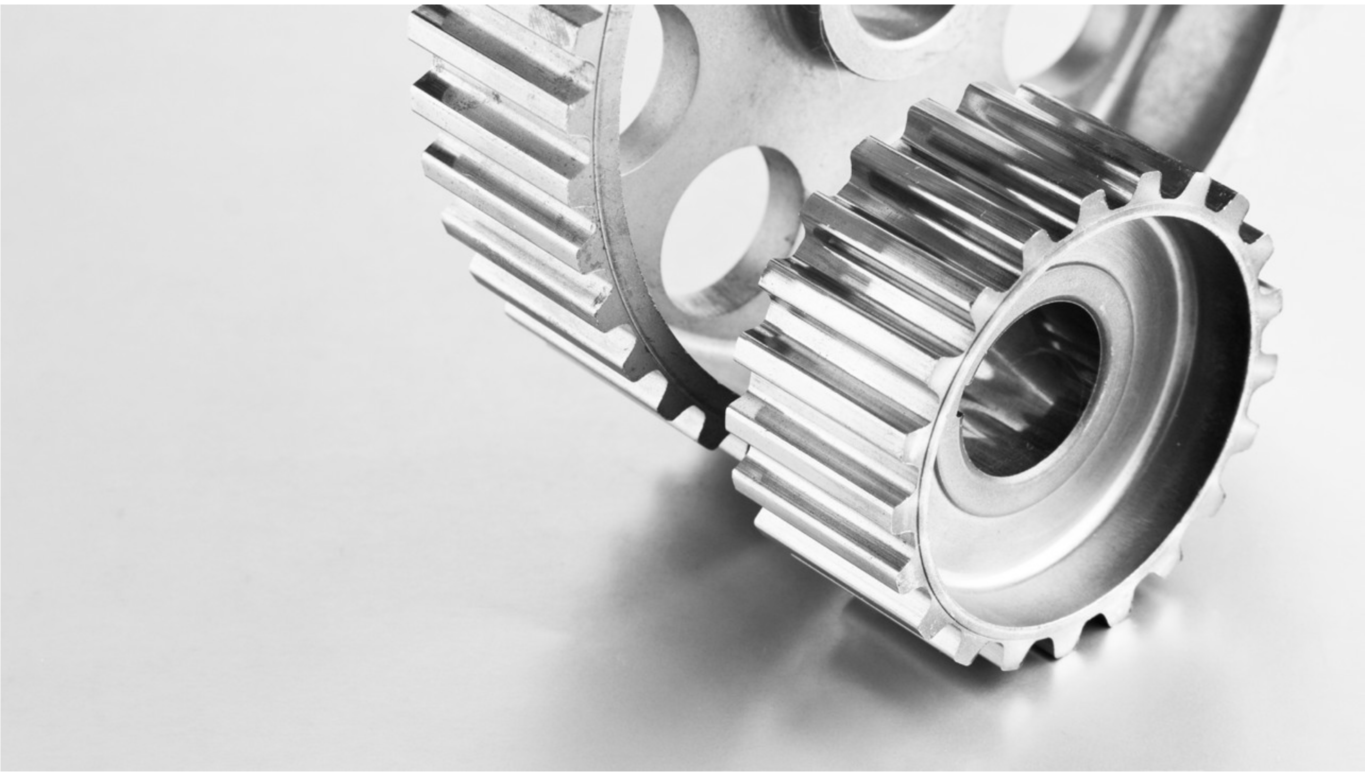
   Now that you've chosen your testing tool, get your data ready to go. To write good, reusable test scripts, your tool will need to be supplied with high-quality data before execution. Make sure your data hasn't been corrupted and is up to date. Include invalid inputs to test negative results and unsupported formats to cover all testing areas — and if your data is too large, invest in a data automation tool to save time.
4. Share the load

   Effective testing demands building, writing, and maintaining scripts in order to find all possible defects in your software, and this can only be done as a shared effort across your quality assurance team. Bring in different skill sets and embrace varying skill levels — everyone will have a role to play here. Divide testing into smaller tasks and map out clear frameworks, implementation requirements, and test case specifications.

5. Stay flexible and adaptable

   Like your software, your testing process will have to adapt to changing user needs. As your interface evolves, changes will move things around in your test cases, affecting your results. To avoid having to completely redo your testing process in the future, find an automation tool that allows for easy updates or label control captions around an easy-to-remember naming system — not objects that may be relocated by a changing interface.

5 top tips for optimization



Once you're all set up, what can you do to ensure success?
Explore proven ways to boost implementation or streamline your existing automation processes.

1. Keep it simple

   Depending on the needs and capabilities of your engineers, it's a good idea to start with small tests. (See Know when to automate). This will help you quickly identify where you may need to make improvements to your test data or processes. Once your small test cases have proved successful, you can move more tests to automation. An added benefit is that shorter tests are easier to label, maintain, and reuse than larger, more complex cases.
2. Clean up your code

   Just as you'll set guidelines to routinely conduct maintenance on your automation tool, have your test automation team agree on how to check your code for obsolete test cases and unused data. Best practice to keeping it clean and up to date is by checking it on a daily basis. The last thing you want to do is confuse your automation tool by supplying old or corrupt coding.
3. Stop repeating yourself

   The goal of automated testing is to make things easier and less time-consuming than manually testing — so why spend time rewriting the same test code over and over again? If that code can be used in multiple test steps, write it once and turn it into a test library. With easy access to reusable test scripts, your test library will help you quickly create code for new features and update tests according to changes to functionality.

4. Energize your team

   Testing automation is a team effort that demands support from your QA team, software developers, and stakeholders. Before you begin, include a Definition of Done to clearly define your goals, scope, and roles. Set guidelines for reporting results on a regular basis to leadership and how to identify action items. Armed with clearly defined ways of working, your team will feel confident and motivated to succeed.

5. Find the right approach

   Take your time to find the right approach for your team. If an assessment of skill levels has identified gaps in coding script knowledge, con- sider taking on codeless testing as an alternative, such as keyword-driven testing. This simple approach creates a series of keywords around a specified action, so that even non-testers can help create hundreds of robust automated tests.

Future-proof your testing

As technology advances, users will continue to demand more intuitive software at a faster rate, which means shorter development cycles. **To keep up, you simply can’t afford to lose the time, energy, and resources on manually testing repetitive or time-consuming tasks that could be automated.** Adopting a solid testing automation strategy and best-of-breed tools will ensure you stay ahead of the curve and deliver higher quality products as a future-focused organization.

We hope this little book of hacks will help you get started with ease or optimize your existing automated processes. If you find that your resource or capacity for automation is limited, get in touch with the a1qa team of test automation experts.

About a1qa

As a leading software QA and testing vendor, a1qa brings 17 years’ experience across multiple industries, with expertise in a wide range of services — from consulting and training to full-cycle QA testing. Our proprietary R&D for test automation offers a customized framework to meet your business needs and exceed your expectations.

Need professional QA support?

Get in touch with an experienced QA service provider to get a test automation solution tailored to your systems and operations.

a1
qa

Software
Testing
Company

[a1qa.com](a1qa.com)
start@a1qa.com
United States
tel.: +1 720 207 5122
United Kingdom

tel.: +44 208 816 7320

Follow us

Documents / Resources

	a1qa Test automation 101 [pdf] User Guide Test automation 101, Test automation

References

- Pure-play Software Testing Company – a1qa