



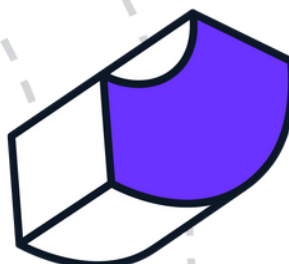
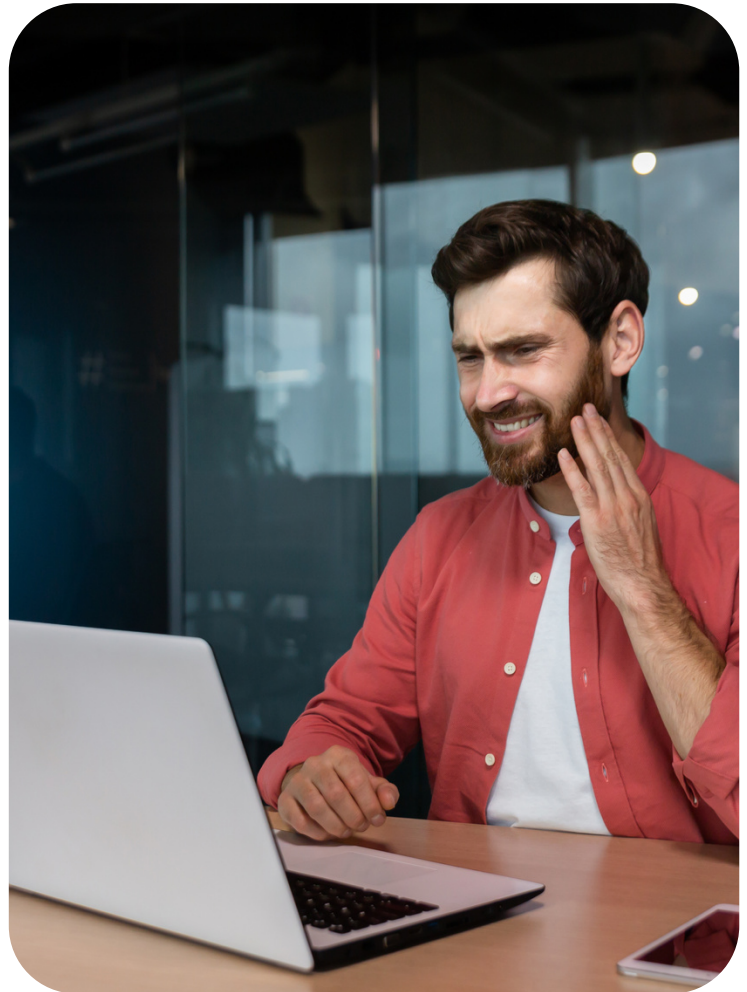
Toothache, Headaches & Manual Testing



Toothache, Headaches & Manual Testing

Painful medical conditions and manual testing may not initially appear to have much in common, but while neither will kill you, they both impact your productivity and the world would be a more pleasant place without either of them.

Science has yet to identify a global panacea to headaches and toothache, however, we can imagine a point in the future where their frequency and severity will be significantly reduced. But can we envisage a future without manual testing or one where the burden on your company is similarly transformed?



Let's first try to understand why manual testing persists?

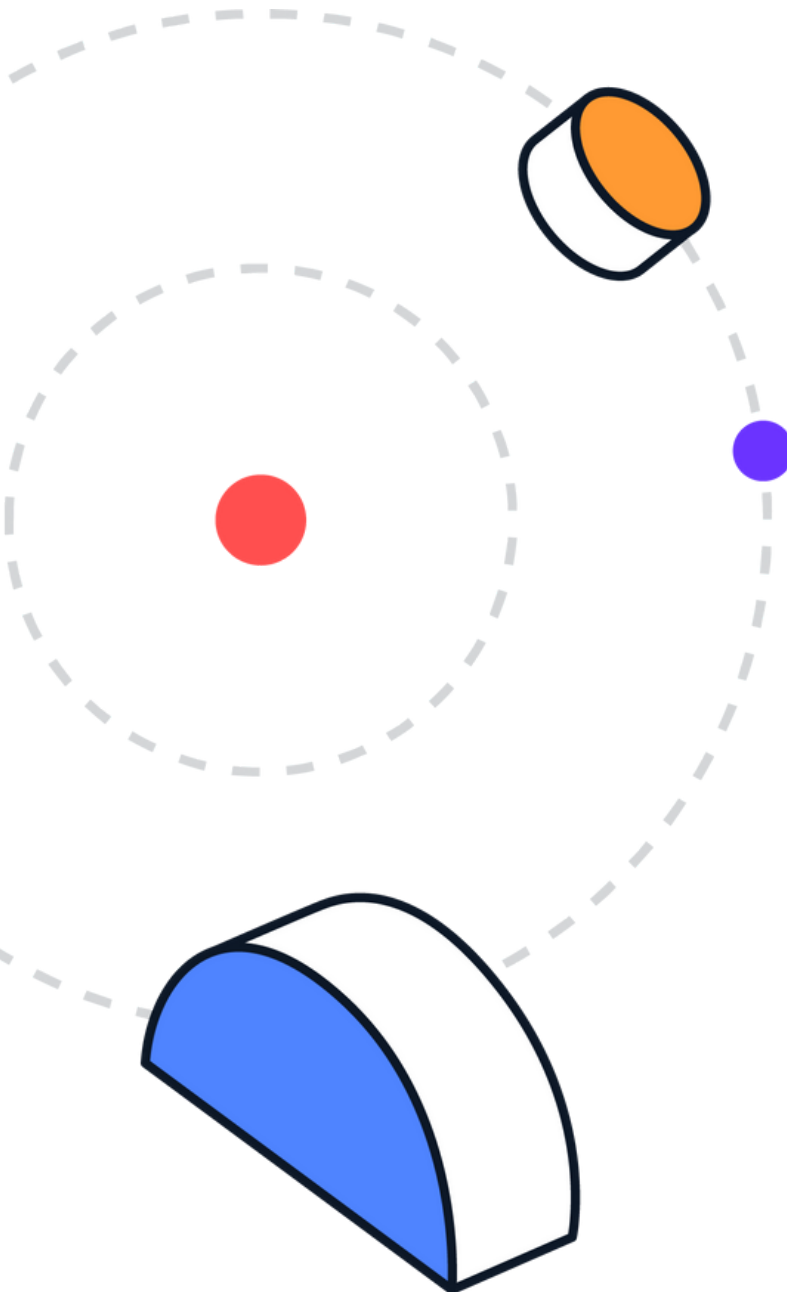
Automated testing is not new, but it is an activity that requires planning, takes time and resources to develop and will not be immediately available. Manual testing, by contrast offers instant gratification, requiring no technical skills and only limited planning. We all know the downsides, but instant gratification is highly addictive.

But, before we simply classify manual testing as an addiction, are there any other reasons why this activity persists? Is it, in fact, a necessary evil?

Our company, Original Software, offers [TestDrive](#) which is probably the most productive test automation product on the planet. But even we cannot claim that you can automate a one-off test, as quickly as you can execute it manually.

So, are there many instances of one-off tests or low-frequency tests where test automation cannot offer a positive ROI?

Of course, there are!

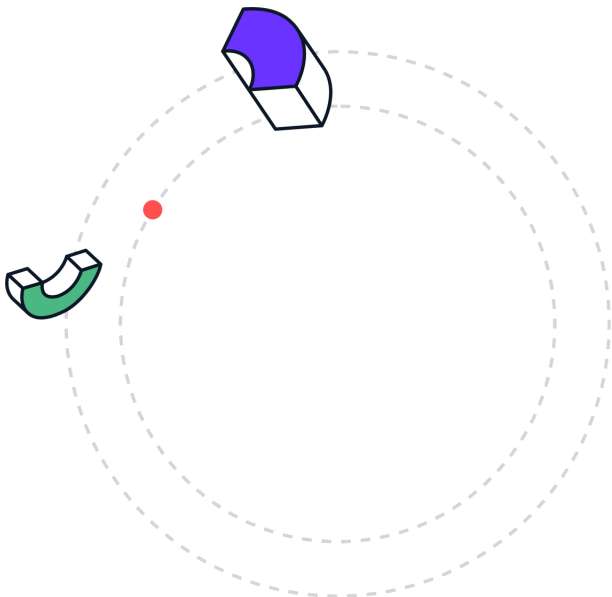


Why does Manual Testing get a bad press?

Aside from the rarely changed areas of your applications where there is only an infrequent need to test, there is the massive area of first-touch testing – you've purchased a new application, or it's been delivered from development, and you need to take it on its first test drive. Does it meet the business need, can a business user be productive with it, and is it even stable? These are questions that must be addressed before it is sensible to invest in automation. So, for these reasons alone manual testing will always be with us.

Now we've established that manual testing is here to stay, why does it get such bad press?

It gets bad press because it deserves it. Manual testing takes too long, places a load on scarce resources, produces uneven test coverage and is associated with uneven feedback and indifferent defect reporting.



That's a pretty damning indictment, but it gets worse.

It is a fact that a single test cycle doesn't find all the defects and that a single fix cycle doesn't address all the reported defects correctly. So, the manual testing cycle must be repeated and repeated and repeated. That leads to user fatigue and disenchantment. Not good in any circumstances, but horrendous when you consider the net effect is that the worst testing occurs just before you go live.

User acceptance testing is another manual testing activity, and it is hard to envisage a time when subject matter experts will embrace test automation. And surely, they shouldn't have to. By the time an application enters the UAT phase, the goal must be that all technical defects have been eliminated. That still leaves functional defects where, despite all the preceding efforts, the application cannot support the business processes as currently defined.



And we cannot ignore another aspect of UAT.

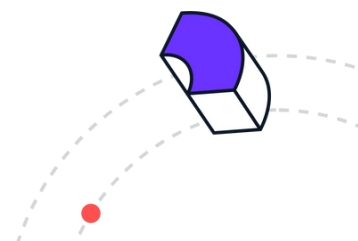
UAT performs a crucial role in exposing key users to the new application for the first time. So UAT can be as much about training as it is about testing.

It is clear that for many reasons manual testing will be with us for the foreseeable future. But does it have to so manual? Can anything be done about its poor reputation? Can manual testing be made more productive and dynamic?

Yes it can!

There are numerous ways to make manual testing easier, cutting down on time and tester fatigue.

Start by looking at stages where automation could be added, perhaps in simple, repetitive tasks, where human intervention isn't needed. By identifying where manual tests can be reduced, any unnecessary duplication of tests can be eliminated.



But, the bottlenecks!



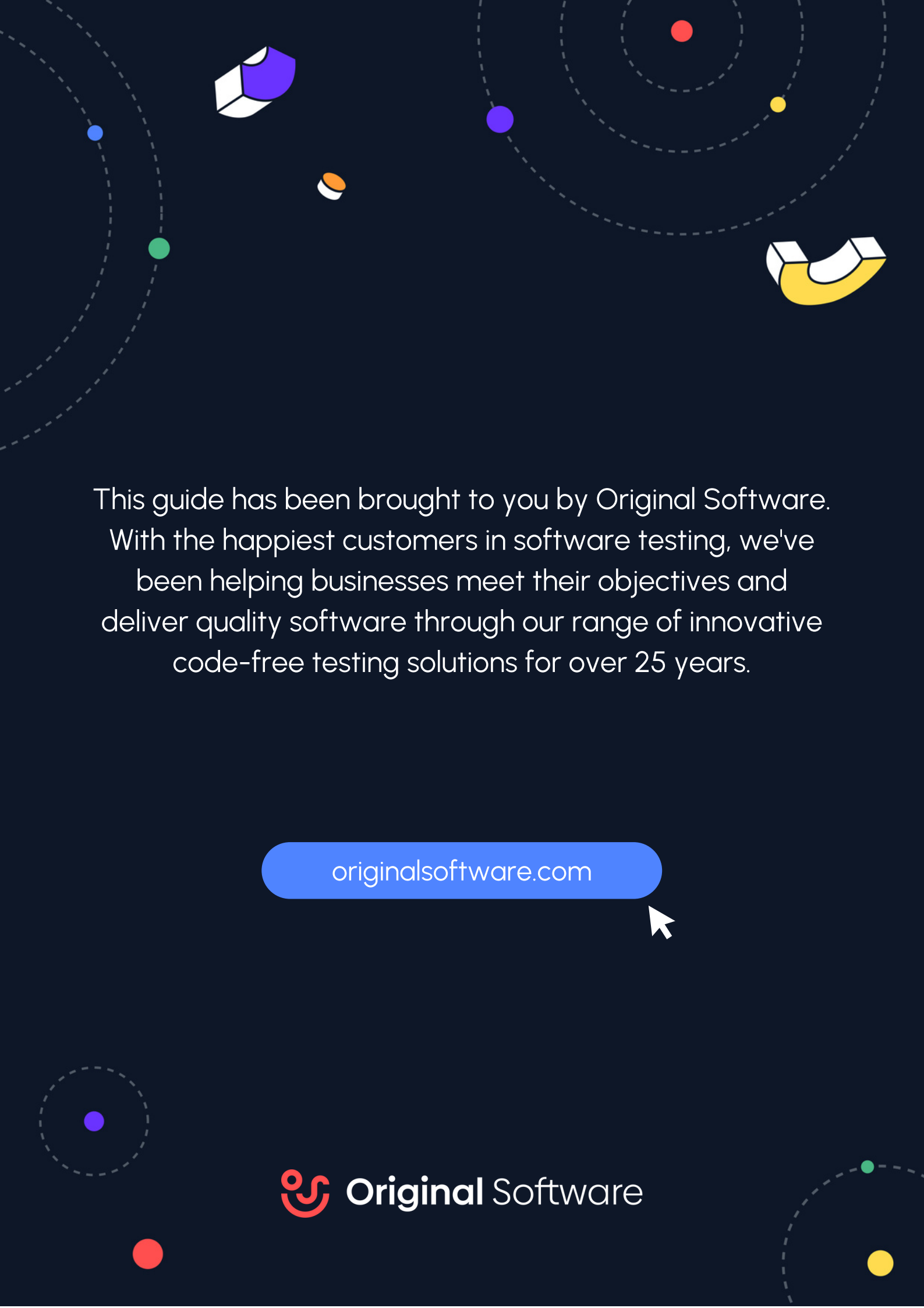
Unfortunately, relying on manual testing for the more repetitive stages of software testing can cause bottlenecks, with testers often experiencing fatigue and boredom that slows them down. It can also be difficult to keep track of and replicate any bugs found. Software such as TestAssist can help, as they build an audit trail for compliance, so there's no need to carry out screen captures or keep logs.

It can also help make poor testers better testers. They provide instant feedback for agile mock-up prototyping and enable the production of training guides and process documentation for free.

This all makes for a compelling case to consider using manual testing tools. Automating some of the simpler repetitive tasks will enable the re-allocation of resources to more complex areas manual work is required.

If you'd like to know more and discover how manual testing can be made easier and used as an entry point into rapid test automation, let's have a chat.





This guide has been brought to you by Original Software. With the happiest customers in software testing, we've been helping businesses meet their objectives and deliver quality software through our range of innovative code-free testing solutions for over 25 years.

originalsoftware.com

 **Original Software**