Java and Selenium: Automating Test is it always better?

Maria Emanuela Arico
Ettlingen

Schlüsselworte
Selenium, Automated Tests, Tests

Test and Software Development

In Software Development, the test phase is one of the most important once. Testing all the parts of our software is a must: it is impossible to have a reliable and stable software without it. The goal of the test phase is to detect bugs and failures in the software in order to fix them.

1. Automated Test Vs Manual Test

In the last years, the topic of the automated test is getting always more "trendy", opposed to the manual test. The Automated Software Testing is a sum of processes that execute some pre-defined and automatic scripts in order to have ways to test all the parts of one software before releasing it in production environment (also in case of bug fixing, to verify that the application works correctly in its totality). This is possible with the use of a particular framework, that running the scripts, allows us to execute them as we want (for example several times, at a specific time or with specific values).
Manual Tests are done by a physical person, called QA (Quality Assurance), that tests the application manually by inserting several times different values in it in order to test if all the features of the apps work correctly. Both ways have several pros and cons.
First of all, let's evaluate the costs of both: for an automated test we will need scripts, software and tools, and someone that develops these. This means investing money on the recruitment of professional figures able to develop and configure these tests in our system. For the manual tests, we will need a fixed QA that will carry out the tests manually during all the development process as well as for the maintenance of the application itself. Another important point to evaluate is the type of test we need. If we need a repetitive test we have to consider that this will take a lot of time and effort to be carried out by a person: the QA will have to spend a lot of time to repeat the same tests multiple times. An automated script would do this kind of work in a little time and without a physical person investing his time. It will execute the iterations faster than a person, since it is executed by a tool. On the other hand, we have to consider that a manual test does not need someone to develop a script to execute it: this is an advantage in case of specific tests, that can be done directly by the QA, and it will not be needed to develop a hoc-based script for it.
Another point to be evaluate is the human approach given by the QA: manual tests will allow us to have a human perspective during the test, giving us the possibility to improve the user-friendliness and the usability of the application, a plus that an automated test cannot give us. Another point to be evaluated is the reliability of the tests: an automated test is human-mistake-free: a script will carry out always the same action, avoiding the risk of the QA forgetting a specific test or inserting as input a wrong text. But a script will never act or think like a person: a QA can for sure test better the user-experience of the application, try to think and simulate what could be, for example, a specific final-user behaviour. A script that is already predefined cannot modify its behaviour during the test to try to simulate a real-user acting.

Selenium for Automated Test

In the big world of the automated test, one tool is dominating the scenes of the GUI test: Selenium.

Selenium is one of the most popular and appreciated web testing tool, specifically to automate all tests to be carried out on the graphic interface, for many different platforms. It is open source and supports all the most used browsers. Selenium is based on an own language script called Selenese, although the scripts can be written, with the use of several APIs, in the most used and known languages (as Java, PHP, Python…), as well as they can be integrated directly in our project. The scripts can test directly the correct behaviour of the graphic interface.
This tool is great in case of test of applications that are updated, fixed and improved often: we will always have a way to test easily and fast if our application is working as expected after any possible modification made on our source code, without having a QA that has to test again manually all the parts of the application that could have been affected by the edition that took place.

Is Selenium the future of the software testing process?

One first reason of the extreme grow of the use of Selenium in the last years is for sure given by the open source nature of the tool: it is free and can be used and adapted to several situations (from the big customers to the beginner QAs). We have also the really good and wide compatibility of the tool with the most famous and used web browsers: with Firefox we have also the possibility to register and playback the scripts, without developing them, and execute them as much as we want. The compatibility of the tool with the most used coding languages, helped also the expansion of this framework. Another good point of this tool is that is already compatible with the modern mobile web applications, for the two main mobile operating systems, iOS and Android. We must consider in our analysis about the future of the testing phase the growing relevance that mobile developing is having in the last years, and this will for sure give a huge possibility to this tool to be the future for still many years of the automated software testing phase.

Can automatic test (and Selenium) replace completely the manual one?

This is a very important question. A lot of customers try to have only automated test: it is easy to understand why, they need just a first investment to develop them and after they do not need any other human action, as long as they keep the behaviour of the application as it is. This means a lot of money saved because there is no need to hire someone that has to test manually the behaviour of the application in every bug fix or new releases.
But having an automated test for all the final-behaviour is almost impossible: who is writing the script cannot calculate all the possible cases that could take place in an interactive application. Let's try to imagine this problem with an example. We have a normal web page in which you can do a registration to have the credentials to log inside our application. We will have several text forms, and we can develop scripts in order to test if the application works fine everywhere, if it is giving back the correct error code in case of mistakes in the insert of the text, if it saves correctly the given information and so on. But we will never be able to test all the possible user behaviours in an automated script and we will never be able to execute, for example a specific procedure of actions to be taken in order to test a reported bug or if a specific bugfix is working correctly. Developing many scripts for all these test cases is expensive, useless and takes long, especially if they are tests that we will not be needed anymore. As we said at the beginning, an automated test cannot report or give feedback about the usability of the application, and we will not have the "human perspective" that at the end is needed to improve an application that was designed to be used by humans.
For this reason, it is correct to say: automation is not always the best option.
After assimilating this concept, it is important to understand when the advantage of the a automated script must be chosen.

We will try to analyse and define what are the best cases and conditions in which it is more convenient to use either manual or automated tests, keeping always in mind that most likely the best way will always be using a combination of both, especially if the software to be tested is being further developed or is constantly object of bugfixing.

When is it better to use manual tests?

In general, it is better to use a manual approach of the testing phase in all the cases where we need a "human perspective":

- When we need to test the "usability" and the "user-friendliness" of the application
- When we have to test a specific function or behaviour of the application
- When we have to check a specific test-case, that needs to be tested only a few times, and it does not really need the development of an hoc-script
- In case it is needed to evaluate new features or functions of the application
- When "human feedback" about one or more functions is required
- A general rule could be defined as: using a manual test is better when something not strictly related to the "pure" functionality of the application has to be tested, and something needs to be judged by a "human eye".

When is it better to use automated tests?

In general, it is better to use an automated approach of the testing phase in all the case we need to test something more than once, fast and in a reliable way:

- When we have to carry out many repetitions of the test, that are executed faster by a tool
- When we are going to test a functionality that has to be tested in every released update of the application
- In case of "performance tests" and "stress tests", so when we have to simulate a big amount of users using the application at the same time
- When functionality of loading or processing of the application tests are required.
- In general, in all the case in which we have several repetitions and in all the cases in which it is not needed a "human point of view".

Kontaktadresse:
Maria Emanuela Arico
esentri AG
Pforzheimerstraße, 132
D-76275 Ettlingen

Telefon: +49 (0) 7243 / 354 90 0
Mobil: +49 (0) 7243 / 354 90 63
E-Mail emanuela.arico@esentri.com
Internet: www.esentri.com