Reinventing the wheel: Lessons learned from my first year as an Automation Test Developer

Sam Simataa
sam.simataa@gmail.com

Abstract

Transitioning from manual software testing to automation testing seemed straightforward on paper, but as many know - it’s easier said than done. Nevertheless, having graduated from college with a technical degree, I decided to take up the challenge.

I was introduced to TestCafe\(^1\), a Node.js end-to-end test automation framework, and spent many hours learning and implementing what I was learning. Finding and analyzing bugs found by the automation framework gave me more insights on how to refine test scripts and what approaches to take when testing User Stories. With more practice, I began to understand the importance of organizing test code and increasing readability by using Page Model patterns. This quality of code can be parsed faster, and problematic sections can be demoed or explained more clearly. I’ve found this to also aid in nurturing healthier collaborations with other Quality Assurance (QA) team members and even Developers.

Reflecting on my first year in test automation, a profound sense of self-confidence has grown from the lessons learned. Therefore, in this talk I will share a process I’ve created for myself when learning new technologies that may seem overwhelming: Learn - Implement - Organize - Collaborate - Repeat. This cycle has allowed me to be a more vocal and open team member and continues to fuel my QA passions.

Biography

I am a new Automation Test Developer with little over 1 year under my belt, but with over 7 years exposure to the Information Technology (IT) world. I have mainly worked in the Manufacturing/Construction (Fastenal Company) and Healthcare industries (Preventice Technologies, Mayo Clinic). However, I have also done freelance QA project work through Testlio and tested software in several other industries including Entertainment, Automotive and Finance.

The following are presentations I have done before:

1. Ministry of Testing - UI Automation Week - TestBash().Online 2020
   1.1. Experience Report - Web UI Automation
   2.1. Experience Report - Shift Left Testing
   2.2. Experience Report - Sharing Your Testing Story
3. PyCon Namibia 2021
   3.1. Quality Over Quantity: An Introduction to Software Testing
As a QA professional, I enjoy getting to know and learn from people of all walks of life and advocating for the user experience. I also enjoy listening to Bach, Drake and A.R. Rahman; playing soccer, tennis and doing triathlons; and travelling, museums and good conversations.

1. Introduction

Having graduated from college with a major in Management Information Systems and minors in Computer Science and Music, I ventured into the world of IT and landed a position as a Support Analyst. Over time, debugging support calls and tickets evoked a sense of user advocacy in me and in time led me to become a Manual Tester. Even though I was satisfied by my passion in advocating for end users and finding bugs, over time I felt the need to dig deeper into QA, and eventually discovered test automation. This felt like a perfect fit: I would still retain my previous QA passions while making use of the technical aspects of my college degree.

I began this journey when I landed a position as an Automation Test Developer at Mayo Clinic in Rochester, MN. One of the chief attractions to them is their primary value: the needs of the patient comes first; this value spoke to my inner value of advocating for the end user. This buy-in would prove instrumental in me keeping the personal motto of “quality over quantity” in the initial days of my test automation journey.

2. Deciding on a Framework

The allure of User Interface (UI) Test Automation is quite powerful, as automation as a discipline has risen in demand. However, getting started can be an inordinate task. This is due to multiple resources and technologies that exist, and the plethora of tutorials as well. It can be overwhelming to start down a path and be exposed to many different learning paths and resources. Fortunately for me, I had a project to think about. Therefore, one way that I was able to overcome this was by creating some criteria to help decide which automation framework and tool would be good to use for the project. One of the decision frameworks I’ve enjoyed using is the Decision Matrix decision making framework, invented by Stuart Pugh. It allows for multidimensional evaluation of several contending options. This made choosing a framework efficient and relevant.

3. Growing in Knowledge

Once I was able to select a test framework, I followed the documentation and examples that were suggested by the TestCafe website first. This allowed me to learn and build in best practices as expected by the creators of the technology. In other words, it felt like building up muscle memory to be employed in the future. In my case, using the decision matrix led me to discover TestCafe - a Node.js end-to-end test automation framework. It was easy to set up and get going, and documentation really assisted in building that muscle memory.

Automating the UI portions of User Stories allowed me to practice implementing lasting change. I found myself trying to implement tests for a single User Story and would have to refactor when the next User Story came along. It’s as Aristotle once famously said “the more you know, the less you know you know”. However, I also came to find that the more you know, the more you know! It may sound redundant, but my experience supports this in that I was able to refactor code to catch valuable, albeit simple, bugs. Creating these valuable bug reports aided in Gap Analysis of parts that may have been missed in Development or Business Analysis. Just like a piano player
practicing a piece, the implementation coupled with the muscle memory allows for a deeper understanding of what is in front of you.

4. Code Organization and Readability

In an agile world, change is constant, and change is the nemesis of automation. As mentioned above, frustrations came with each User Story, but so did more opportunities to learn. I stumbled upon the concepts of Abstraction and Page Model patterns - allowing code to be organized and become reusable components by multiple test files. This allowed me to instill a “Don’t Repeat Yourself” (DRY) principle and approach in my test case design. The biggest gain to this was creating a natural sense of code readability. This would come in very handy when trying to demo code or asking for help when stuck.

5. Collaboration

I would like to believe that striving for higher code readability has helped in collaborations when either asking for help, trying to succinctly get to the core of an inquiry, or even when trying to propose new changes in the automation code. Creating such feedback loops really helped make me feel supported and competent, even at the beginning level.

6. Conclusion

Now reflecting on my first year in test automation, a profound sense of self-confidence has grown from these lessons learned. I then realized the following process for myself when learning new technologies that may seem overwhelming: Learn - Implement - Organize - Collaborate - Repeat. 
Learn - using resources to choose and understand a technology better, whether this be for test automation or otherwise; Implement – adding value to a [work] project by use the gained knowledge to find and develop creative solutions when automating UI portions of a User Story; Organize – working smarter and not harder by planning and creating reusable components that can benefit a project in the long run such as using Page Model patterns; Collaborate – sharing the ups and downs of your progress in order to garner credibility and create a sense of community and feedback loops; Repeat – challenges arise that may be outside the scope of the technology used, so repeating the four previous steps will help overcome some of the challenging barrier. An example could be integrating another technology or moving to a new technology all together.

This cycle has allowed me to be a more vocal and open team member and continues to fuel my QA passions and reminds me that as a QA professional, I should always strive for quality over quantity.

References
