If you're exploring your tooling options for test automation, it doesn't have to mean a suite of costly proprietary tools. Plenty of open source testing tools have become viable options. This eGuide explains why you should consider open source tools for your testing needs—including for the specific situations of testing in production, DevOps, accessibility testing, and security testing—as well as considerations you should take into account when selecting your tools.

In this Open Source Testing Tools eGuide

Why Selenium Should Be Your UI Test Tool
Selecting a testing tool is hard work. If you look on vendor websites, you'll get marketing material promising the world. If you look on forums, you'll mostly get people trying to solve their own problems. Justin Rohrman tells you why you might choose Selenium as your UI testing tool, based on real experience with real software projects—rather than a marketing page.

Solving Production Issues Using Testing Tools
Standard web-monitoring tools can ping webpages and verify that they're responding, but they don't alert you to an issue. But you can use the technology in load testing to monitor your sites by running an interactive script that can detect issues and generate emails as needed. It runs constantly like a silent sentry, never sleeping or taking a vacation, improving your sites' reliability.

Why You Need Continuous Testing in DevOps
DevOps is more than adopting the right set of tools; it's a cultural shift that incorporates testing at each stage of the agile project lifecycle. Continuous testing is key to unlocking this culture change because it weaves testing activities into every part of the software design, development, and deployment processes, which helps everyone involved communicate more, collaborate better, and innovate faster.

What Testers Need in Their Accessibility Testing Toolkits
A software tester's accessibility testing toolkit should contain various tools, both to help testers "walk in the shoes" of their users and to quickly flag obvious problems and expose accessibility features (or a lack of them). High performance is only achievable with human skill, but these tools will help you uncover potential issues and make your product a better user experience for a wider audience.

Using Open Source Tools for Security Testing
Performing a series of security tests before deployment of your application has become paramount. But that doesn't have to mean a suite of costly tools. Plenty of open source security testing tools have become viable options. Here's why you should consider open source tools for your different types of security testing.

How Testers Can Use Docker to Shift Left and Automate Deployments
Docker has several advantages over virtual machines: It's easier to deal with, starts up faster, and requires fewer resources. Using Docker also can give testers more confidence in their releases. Developers use the same environment that will be used in production, which streamlines code delivery and shifts QA left.

Balancing Process and Tools
The limits of a tool may lead us to realize that we are not working as effectively as we can, and often, changing a tool is part of the solution. But there are good and bad ways to select a tool and how you use it. In particular there are risks when you focus first on tools before considering the problem.

Insight from the Industry

Additional Resources

COPYRIGHT 2019

Why Selenium Should Be Your UI Test Tool

By Justin Rohrman

Selecting a testing tool is hard work. If you look on vendor websites, you'll get marketing material promising the world and then a call from their sales staff. If you look on forums, you'll mostly get people trying to solve problems relating to their product: "Why won't SuperUITester++ click Save on my Flash application when there are 4 other modals open?"

Let's take a look at why you might choose Selenium as your UI testing tool with information based on real experience with real software projects—rather than a marketing page.

Why the UI

There are lots of different places you can begin testing software. I like to think of them as different layers in a product, just like sedimentary layers archaeologists dig through when unearthing ancient cities. You want to use the most appropriate tool for each layer of software you are looking at: smaller units, seeing how methods work together, fully formed APIs, simple scenarios in the UI, and complex product usage. All the layers are important; the question is how much of each you want.

The API and everything below that will give you a feel for code quality and some basic functionality. Testing the UI will help you know things from a different perspective: the user's.

Why Selenium

The WebDriver object triggers real events in the browser: mouse clicks, button clicks, entering text, and events from the keyboard. Think of each step as a building block. Stacked together, they can enable a technical team to do some powerful things. Here are a few of the more common reasons for using WebDriver.

Building Automated Checks

Probably the most common reason people elect to use the Selenium suite of tools is to drive a specific set of commands and check status—to see if the user is logged in, if the book goes into the shopping cart, or if a transaction processes.

This includes checks to make sure buttons or labels are present on a page, data that you created saved correctly, procedural aspects

COPYRIGHT 2019