Can I use Selenium for performance testing?

When releasing a new version of the website, we want to make sure that the new version does not perform slower than existing code. If we do not pay attention to the website performance, it will not take a long time for our fast and new website to become slow and unpleasant to use.

PROBLEM

Technically, we can use Selenium for performance testing. However, just because we can technically do something, a question should always be nagging us at the back of our mind: should we do it? Using Selenium for performance testing is similar to using a butter knife to cut a steak; effective but unpleasant. The following factors make Selenium unsuitable for performance testing:

- **Inaccurate**: When testing page load performance, we are not testing the server response times but page load times. Even if our website is fully loaded, some asset from a third-party application might still be loading. Timing the page load times will give inflated results.
- **Testing JavaScript performance**: Selenium is not the best tool to use to test the performance of JavaScript on the page. First of all, it is too big and clumsy to give any results worth noting. Since JavaScript is technically a single-threaded language, that means that our site's scripts will have to share resources with any third-party JavaScript loaded in the browser. The slower than expected results might be coming from something outside of our control.
- **Testing asset performance**: If we cannot reliably test JavaScript performance and the performance of complete page loads, maybe we can test the asset loading. Selenium will not be able to tell you accurately how fast a certain image or video downloaded and rendered in the browser. Browser performance depends on so many factors outside of the current page, such as the amount of windows currently open, available RAM, disk usage, and CPU usage by background processes. Not to mention the asset caching performed by the browser and the caching performed by a network caching proxy that your IT department installed for the whole company. It is

close to impossible to get consistent download and render time results without using Selenium.
- **Testing server load**: By far, testing the server load with Selenium is the least helpful experience. In order to generate noticeable load on the web server we need dozens, if not hundreds of Selenium instances, and even then, it might not be enough.

In conclusion, Selenium is a terrible solution for performance testing. There are simpler and better solutions available.

POSSIBLE SOLUTIONS

Depending on the type of performance testing we are trying to accomplish, there are specialized tools to accomplish just about any goal. For example, to put a server under heavy load and record the response times from it, JMeter is a great tool. It simulates user behavior by recording the HTTP interactions your browser makes as you normally browse the website and replays these interactions with thousands of concurrent requests.

NOTE

For more information on the JMeter project, visit at http://jmeter.apache.org/.

To test the performance of the JavaScript on a given page, Google provides the V8 Benchmark Suite (http://v8.googlecode.com/svn/data/benchmarks/v7/run.html).

If we want to have a cross-browser solution for checking what assets are slow to load, YSlow (http://yslow.org/) is a great tool for that. Furthermore, most modern browsers provide a built-in test suite for JavaScript and asset performance.

Finally, if none of the tools mentioned satisfy your needs, we can write our own scripts. Using Ruby or Bash, we can write a simple script that makes HTTP requests against different API endpoints or assets and records the time it took to complete a purchase request or for an image to download. At the end of the day, a simple shell script will provide a much more accurate performance report than a Selenium test.