James Brown

2/4/2020

WEB-231

Richard Krasso

Assignment 9.4

Client Side Debugging

Developer tools have been critical in every technical job I’ve ever worked, and they are becoming even more critical as more and more applications are being built with dynamic JavaScript and third party APIs. In this one pager, I’m going to talk about three different features of Google Chrome’s developer tools. The features include using the JavaScript Console for debugging JavaScript, viewing and changing the DOM, and inspecting network activity.

Up first in this review is debugging JavaScript through the JavaScript Console. So, what is the JavaScript Console and how do you use it? The JavaScript console logs the majority of errors that occur when JavaScript is running on a web page. Kayce Basques, Technical Writer for Chrome DevTools, states that the first step in using this tool is to try to “reproduce the bug”. (Basques) During this step, you essentially try to recreate the issue by performing a set of steps.

Once you have the steps to reproduce identified, the next step in Basques recommendation is to “Get familiar with the Sources panel UI”. (Basques) In this step, you are essentially modifying CSS, viewing network requests, and other features within the DevTools console. We’ll talk more about those features later, but the next step once you’re comfortable with navigating the tools is to add breakpoints. A breakpoint is a point at which you specify the code to stop running. This is helpful when you are stepping through the code and want to see if the code runs up to a certain point. At this point, you’ve more than likely identified the cause of the error, so the next step would implement a fix. This final step involves editing the code and re-running the script to see if the issue is resolved.

In my opinion, I think being able to use the JavaScript Console to debug issue is one of the most critical skills I’ve had to use as a support Engineer at my job. A lot of services that my job offers revolve around dynamic JavaScript Web pages. It has also been very helpful in debugging other student’s weekly discussion code as the console generally tells you the exact line at which the error is occurring.

Another feature Chrome DevTools offers is editing the DOM. There are a few different ways that you can modify the DOM. The quickest way in my experience, is to locate the element or section of the page you want to edit, then right click and choose “Inspect”. This will open up the DevTools “Elements” panel. From there, you can edit any element in the DOM tree of the Elements panel by double clicking it. This feature is really helpful when you need to change the value of any given element.

The last feature I will talk about today is the Network tab. The Network tab logs and records all of HTTP network requests that are being made from the browser. This includes loading resources like jquery, google apis, etc. These logs are stored in a section called the “Waterfall Graph”. According to BitDegree.org, this section “provides a visual representation of how long the requests spent in queue, took to get a response, and download the resources”. (BitDegree) How is this helpful? If you have a website that is experiencing performance issues, it is more than likely due to resources that are taking a long time to load. This waterfall graph will show you which resources are taking the longest, which will help you with looking for opportunities for optimizing those specific resources.

I’ve only scratched the surface on what Chrome DevTools can do, but they are truly amazing tools.

Sources

Basques, K. (2019). Get Started with Debugging JavaScript in Chrome DevTools. Retrieved February 7, 2020, from https://developers.google.com/web/tools/chrome-devtools/javascript/#apply-fix

BitDegree. (2016, January 5). Why is My Website Slow: Using Chrome Network Tab Explained. Retrieved from https://www.bitdegree.org/learn/chrome-network-tab