Things You Can Do With Chrome Developer Tools

The developer tools that come with Google Chrome are very powerful indeed. With it you can change CSS stylings, debug JavaScript, and step through your code one piece at a time. But there are some other things you may not know about. Swagat Kumar Swain wrote an article entitled, “Things You Probably Didn’t Know You Could Do with Chrome’s Developer Console.” The article has a list of ten little-known tools that are a part of the Chrome Developer Tools (“CDT”). I am going to discuss four of them.

First, with CDT, you have the ability to select DOM elements. This is handy when you don’t have access to jQuery in the DOM (Swain). You essentially use the same selectors as you would in jQuery – tag name, class, and id (Swain). You would use either one dollar sign or two dollars signs, depending on whether you were trying to retrieve the first element matching your selector or all elements matching your selector (Swain). And you can also specify a certain element’s position and retrieve that one element (Swain). Once you’ve selected an element, you can live-edit it (Basques). However, edits you make to the DOM are not permanent changes. Once you reload the page all the changes are erased (Basques). You need to actually open your source code in an editor to make and save the changes permanently.

Second, you can find events associated with an element in the DOM (Swain). You pick an element and use the command getEventListeners($(‘selector’)) to get all the elements that are associated with it (Swain). You can then review each one. You can also check if one particular event is associated with a certain element (Swain). You can also explore the properties of the different event listener objects and see the event listener details (Kearney, Copes).

Third, you can also listen monitor specific events (Swain; Kearney, Copes). Monitoring the events allows you to log the events in the console as soon as their fired (Swain). You can log all the events bound to an element, or just a particular event bound to the element (Swain). You can then pass that event as an argument to a function (Swain). Further, you can set up your own requirements and use that to log multiple events, and you can even pass those events into an array (Swain).

Fourth, you can calculate the time it takes for a particular block of code to execute (Swain). This is done using the methods console.time and console.timeEnd. Figuring out time of execution is important as a measure of performance. According to Peter Bengtsson, a slow website loading can lead to people leaving your site for someone else’s (Bengtsson). While Bengtsson was not discussing CDT or the console.time and console.timeEnd methods per se, he makes a good point about slow loading, and timing your code execution is very helpful when tweaking your code to speed things up.

The four tools I have discussed are just a small part of what you can do with CDT. It is definitely worth your while to read up on other things you can do with CDT, especially as it continues to grow and add new features. Even more important is using these tools. With a little practice, these tools can definitely speed up your development while reducing errors associated with such development.

Resources

Swagat Kumar Swain. (2016, Sep 21). Things You Probably Didn’t Know You Could Do With Chrome’s Developer Console (article). Retrieved from <https://medium.freecodecamp.org/10-tips-to-maximize-your-javascript-debugging-experience-b69a75859329>.

Kayce Basques. (Last updated 2019, Jan 29). Edit the DOM (article). Retrieved from <https://developers.google.com/web/tools/chrome-devtools/inspect-styles/edit-dom>.

Meggin Kearney, Flavio Copes. (Last updated 2019, Jan 29). Monitor Events (article). Retrieved from <https://developers.google.com/web/tools/chrome-devtools/console/events>.

Peter Bengtsson. (2015, Sep 2). Measuring JavaScript Functions’ Performance (article). Retrieved from <https://www.sitepoint.com/measuring-javascript-functions-performance/>.