COMPARISON OF TWO JAVACRIPT DEBUGGING TOOLS

(Pratik Krishnamurthy and Catharina Vijay)

Research questions

In this paper, we are comparing two popular JavaScript debugging tools: Chrome Developer Tools and Firefox Developer Tools. Though there are a lot of features that participants can use in the two tools, we are interested in only three specific features in each tool. We will investigate the tools based on only these features and we try to find the answers to the following research questions:

RQ 1. Which features do the users like in the two tools?
In this question, we try to find which one of the three features in each tool our participants find the most helpful. To answer this research question, we make use of data collected from our Post-Session questionnaire.

RQ 2. How do the two debugging tools differ from each other based on the time taken to fix a bug?
To answer this research question, we use data like the start time (time when the user starts the debugging task) and the end time (time when the user is done with the debugging task). Using these two times, we get the total time that the user spends using the tool assigned to him/her, on a particular debugging task.

Method

a) Participants

We recruited 30 human subjects for debugging JavaScript tasks using Chrome Developer Tools and 30 subjects for the debugging the same tasks using Firefox Developer Tools. We decided on this number because we needed enough data to do a proper statistical analysis. Since our study was not based on gender differences, we were open to having participants of any sex. The age criteria for selection was that the participants had to be 18 years or older.

Subjects were recruited by emailing the grad and the undergrad mailing lists, the developers in the Business Solutions Group, the developers at the OSU's Central Web Services and our colleagues who could know someone fit for the study. While recruiting, we ensured that our participants' background fit our inclusion criteria. We decided to pay our participants an amount of $20 for participating in our study. We preferred people who were invested in the study as they

would make a conscious effort to do well in the tasks. We ensured that we had each participant's informed consent before starting the study.

We randomized the participants and the tasks for the study. To aid randomization, we used an online randomizer to assign participants to either the Chrome Developer Tools group or the Firefox Developer Tools group. For confidentiality reasons, we did not paste the name of our participants into the online randomizer. The procedure that we followed to randomize is listed below:

1. Based on the order in which people signed up for our study, we assigned them a key in the form of numbers starting from 1 and ending at 60.
2. Then using the Randomizing tool on Random.org, we obtained a randomized ordering of the keys.
3. We then considered the first 30 random keys and assigned them to Group A and the rest to Group B. Based on the keys, we were easily able to find the person assigned to each group.

The inclusion criteria for our participants were:

- No experience with JavaScript. (but less than 6 months of experience acceptable)
- Total programming experience <= 2 year
- No previous experience with Chrome developer tools, Firefox developer tools or FireBug extension to Firefox.

b) Study Design

We conducted a live, controlled study where we were able to closely control variables and eliminate as much noise as possible. Also, Mechanical Turks was not a good option for us because people might have different versions of Firefox or Chrome on their computers and it would be difficult for us to evaluate the study and control external variables. We had two separate sessions for people using the Chrome and Firefox Developer tools. The order of the sessions was also randomized.

We made a conscience decision to conduct a "between subjects" study because Chrome Developer Tools is similar to Firefox Developer Tools and we wanted to eliminate any kind of "Learning Effect".

c) Materials

We used a background and post-study questionnaire to gather different kinds of information from the participants. The background questionnaire was used to collect background information about our participants. We asked them questions about their programming experience, debugging experience and whether they had previous experience with Chrome Developer Tools or Firefox Developer Tools. We used three, 21 point Likert scales to measure how comfortable the participants were with programming and debugging. The other questions were aimed to check if participants fit into the inclusion criteria that we had set earlier.

In the post-study questionnaire, the participants were asked to rate their thoughts on the features of the tool that they used. The helpfulness was measured with a 7 point Likert scale. It also asked the participants how comfortable they were with using the debugging tool.

A copy of the questionnaires can be found in the Appendix section towards the end of the paper.

d) Procedure

Experiment roles

We had a person assigned to walk the participants through the tutorial and another person who was the driver. We also had three other helpers during the study to help supervise the 30 participants in each session. After the tutorial portion of the study was done, the driver and the tutorial reader switched roles to become helpers. We ensured that we had a ratio of 6:1 for the participant : helpers. The helper's jobs was to walk around the lab and police the other participants and make sure that they didn't get distracted. The helpers kept the participants on track and answered questions that the participants had. Special care was taken to ensure that each helper only answered questions about the material that was covered in the tutorial. The helpers were instructed to not accidentally help the participants by giving them extra information.

Preparation before the start of the study:

1) Before the start of the study, all the computers in the lab were set with the JavaScript code that the participants would debug and the logging mechanism. Each system had a backup copy of the tasks in case the participant wanted to start their task over again. We also had three systems in standby with the environment already set up in case one of our other laptops crashed. We did not allow the participants to use the internet because we wanted them to debug the tasks and not look up the solutions on the internet.

2) We had all the background questionnaires laid out in front of the computers ready for the participants to fill out once they were seated in front of the computers.

3) We handed out badges to the participants with their participant number on it once they checked-in at the door. The computers and the background questionnaires were also numbered and one of our helpers escorted the participant to the computer that had their participant number on it. This system made it easy for us keep track of all the participants and all material that we gathered.

4) We also labelled the JavaScript files that participants worked on to keep everything organized. For example: For a participant from Group A with participant number 3 working on task number 2, the JavaScript file was named as 'A3 - Task 2.html'. Each task questions was present in the JavaScript file that participants had to debug. It had the participant ID followed by the actual question to help us analyze the recorded screen easily later.

5) Each participant saw a blank screen in front of them until the session started. The helpers ensured that the participants were not fiddling with the system in front of them.

Background Questionnaire

Once the participants were seated and ready, we asked them to fill out the background questionnaire. Our helpers then went around and collected the questionnaires and verified that the participants had answered all the questions. Once all the questionnaires were collected, we moved on to the tutorial.

Tutorial

During the tutorial, the driver had a screen up on the projector and we required that participants had the same screen in front of them. The driver was visually guiding the participants and this really helped us set the pace of the tutorial. The participants also used the projector screen to get back on track when they were lost. During that time, our helpers were walking around the room to see how all the participants were coping.

Tasks

After we got through the tutorial successfully, the participants were asked to start their practice tasks. We required the participants to do one practice task and three actual study tasks. Participants were given 10 minutes for the practice task and 15 minutes for each of the actual study tasks. We wanted to give our participants enough time to do their tasks so as not to pressurize them with less time. The appropriate time was calculated by observing how long our pilot study subjects took to complete a task. At the end of 15 minutes, the participants were instructed to move on to the next task.

During the time that the participants were working on their tasks, the helpers were walking around the lab ensuring that the participants were not browsing the internet or checking their phones. The helpers also answered a few questions that the participants had.

A few of the participants were done with their task before the allotted time but they were asked to sit at their desks without disturbing the other participants till it was time to move on to the next task. This was to ensure that the other participants did not rush through their task because they thought that they could leave.

We had instructed the participants not to browse the web or play with their phones. We were a little strict with how the participants spent their free time because we ran the risk of introducing noise into the study.

Data

As mentioned in the above sections, the background questionnaire asked participants about their general programming experience, JavaScript experience and experience using the tools in the study. In short, the background questionnaire was geared more towards ensuring that the participants fit into our inclusion criteria.

The post session questionnaire asked the participants to rate helpfulness of each feature that they used in the study and their comfort level with the tool. The helpfulness of each feature was

measured using 7-point Likert scale. We assigned numerical values ranging from 1 for “Not helpful” to 7 for “Very Helpful” to each of the boxes in the Likert scale.

Other than questionnaires, we used Camtasia Studio to log the activity of each user. The data that we collected were the times when the participants started their tasks to the times when they ended their tasks. From this data, we found the time that the users took to complete each task.

Results

After finishing our experiment consisting of 60 participants (30 in each of the two treatments), we conducted statistical analyses and found the following evidences to answers our research questions.

RQ1: What features did the users like in the tool that they used?
For each tool, we considered each feature as a “treatment” and did a within-subject study to answer this particular question. To investigate how useful the participants found each feature in a tool, we compared the helpfulness indicated by the participants in the post-session questionnaire for each feature. As described in Table 1, an ANOVA contrast (*R2=0.2235*) against ‘Sources’ feature showed a significant difference between the means of the helpfulness of the other two features. For example, participants found ‘Sources’ feature more helpful compared to the other features. Similarly, in Table 2, an ANOVA contrast (*R2=0.8966*) against ‘Debugger’ feature showed a significant difference between the means of the helpfulness of the other two features. For example, participants found ‘Debugger’ feature more helpful compared to the other features

	Mean (p-value for contrast with Sources)			**dF**	**F**	**P**
	Console	**Elements**	**Sources (Intercept)**			
Helpfulness (max 7)	6 (0.0499)	5 (3.3e-06)	6.67 (N/A)	87	12.52	1.661e-05

Table 1: ANOVA contrast results (against Sources) by features for Chrome Developer Tools.

	Mean (p-value for contrast with Debugger)			**dF**	**F**	**P**
	Console	**Inspector**	**Debugger (Intercept)**			
Helpfulness (max 7)	2.5 (<2e-16)	2 (<2e-16)	4 (N/A)	87	377	<2.2e-16

Table 2: ANOVA contrast results (against Debugger) by features for Firefox Developer Tools.

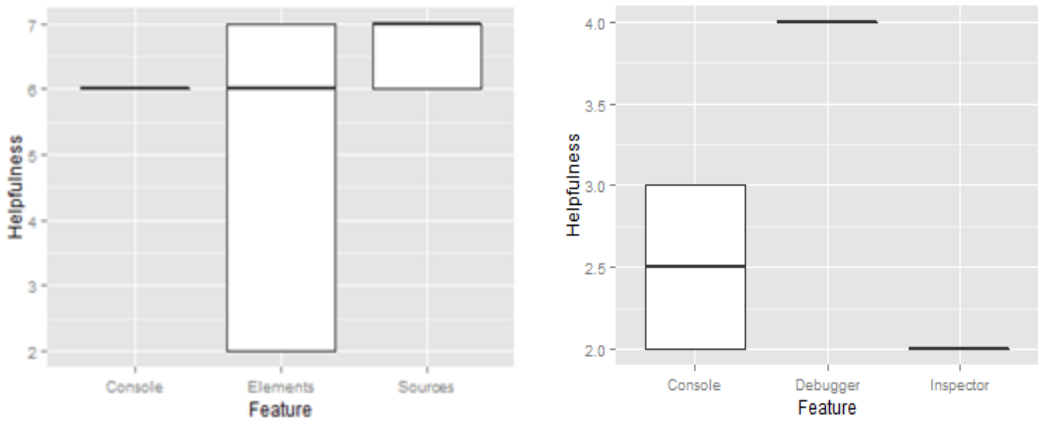


Figure 1: (L) Comparison of Helpfulness of each feature in Chrome Developer Tools and (R) Comparison of Helpfulness of each feature in Firefox Developer Tools.

Figure 1 suggests that the median of 'Sources' feature was greater than the other Chrome Developer Tools features. Similarly, the median of 'Debugger' feature was greater than the other features in Firefox Developer Tools.

RQ2: How do the two debugging tools differ based on the time taken to fix a bug?
To answer this research question, the data collected from the Camtasia Studio log activity and in particular, the time (in seconds) that the participants in each treatment took to finish their tasks was used. The study setup was between-subjects and the experimental design consisted of two treatments (each tool as a treatment). The participants were asked to do the same tasks in both treatments, but the orders of the tasks were randomized. The Welch Two Sample t-test (*t=2.4516, dF = 54.792, p-value = 0.01744, mean of Chrome treatment =444.6667, mean of Firefox treatment =412.5000*) performed using the times it took the participants to finish the tasks showed a significant difference between the times it took to debug the tasks using the two tools.

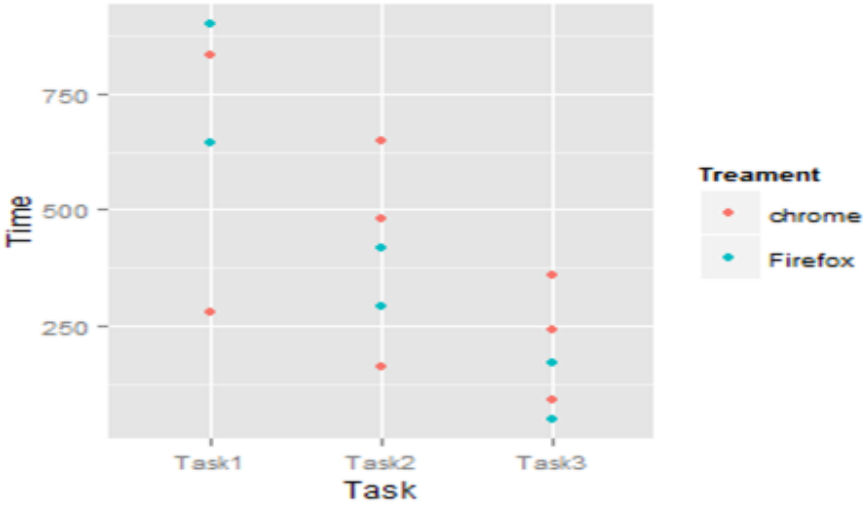


Figure 2: Comparison of Time (in seconds) and the tasks for the two treatments.

Figure 2 gives the distribution of time it took for participants to debug their tasks in both the tools.

Discussion

We summarize the answers to our research questions as follows:

RQ1: Features in the two treatments that the users liked:
Participants in the Chrome Developer Tools treatment found the 'Sources' feature most helpful while the participants in the Firefox Developer Tools treatment preferred 'Debugger' feature most helpful.
RQ2: Difference in the two debugging tools based on the time taken to fix bugs:
There was a significant difference between the times it took to debug tasks using the two tools. Participants using Chrome Developer Tools took significantly more time to debug the tasks.

Threats to Validity:

The internal threats that we identified for our study were the complexity of the tasks, inadequate questions in the questionnaires and the time given to participants for the tasks.

The external threats in our study were the experience level of participants, having the participants decide when they were done with their tasks and participants not fitting into our criteria well.

The reliability threat in our study was the fact that there could be human errors while properly noting the time taken by the participants for each task. Since the tools in our study are popular and widely used by people throughout, there cannot be a doubt of reliability considering these two tools.

Appendix

A. <u>Background questionnaire:</u>

1. How many years of programming **experience** do you have?

   _____ Years

2. How comfortable are you with programming?

Not comfortable Very comfortable

3. How many years of debugging **experience** do you have?

   _____ Years

4. How comfortable are you with debugging?

Not comfortable Very comfortable

5. How many years of JavaScript debugging **experience** do you have?

   _____ Years

6. How comfortable are you with JavaScript debugging?

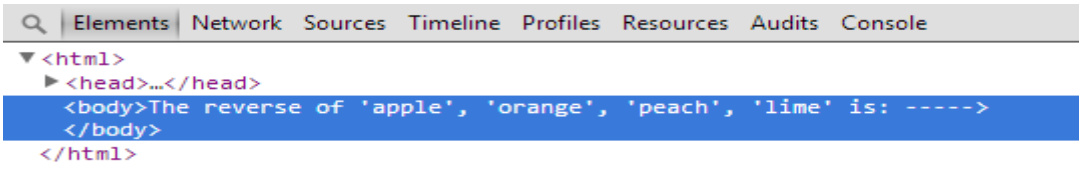
Not comfortable Very comfortable

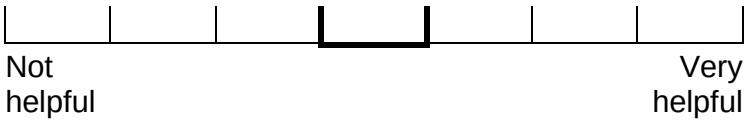
7. Do you have **experience** working with any of the following tools: Chrome Developer Tools, Firefox Developer Tools, Firebug extension for Firefox

- ☐ Yes Which one? _____
- ☐ No

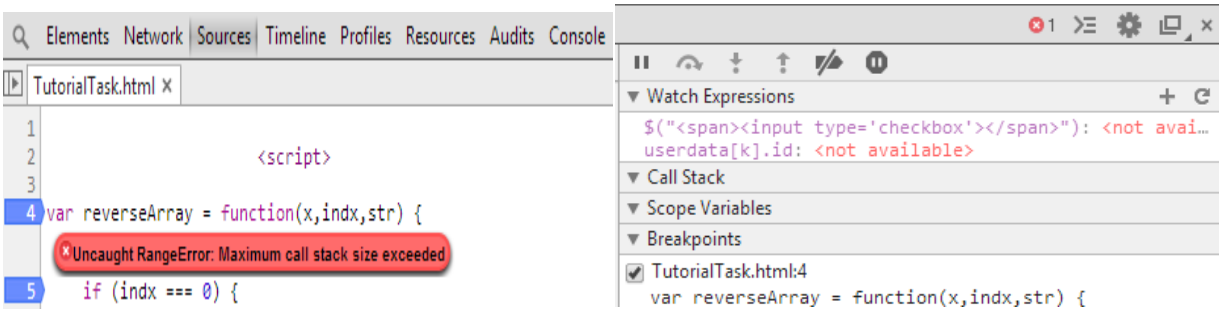
Post Session Questionnaire (Chrome Developer Tools participants):

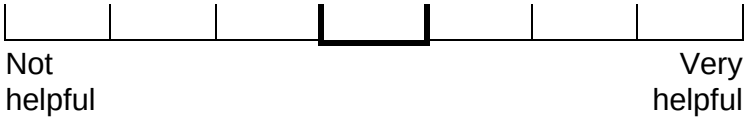
1. Rate your thoughts on the ‘Elements’ feature of the tool.



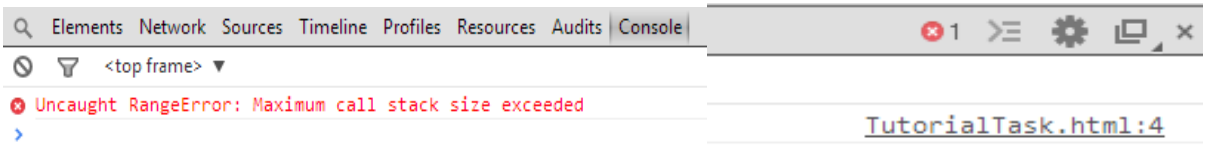


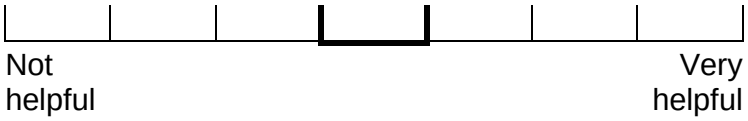
2. Rate your thoughts on the ‘Sources’ feature of the tool.



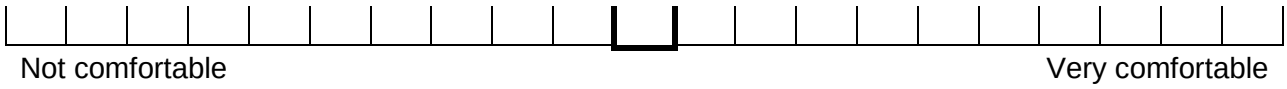


3. Rate your thoughts on the ‘Console’ feature of the tool.

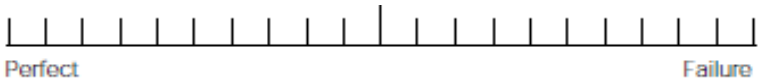




4. How comfortable were you with using the debugging tool?

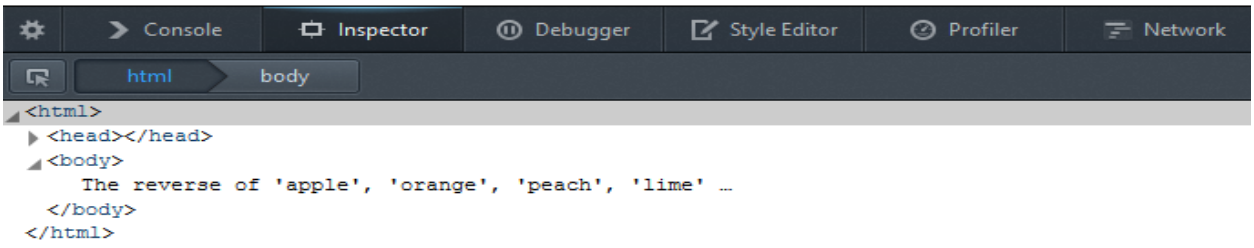


5. How successful were you in accomplishing what you were asked to do?



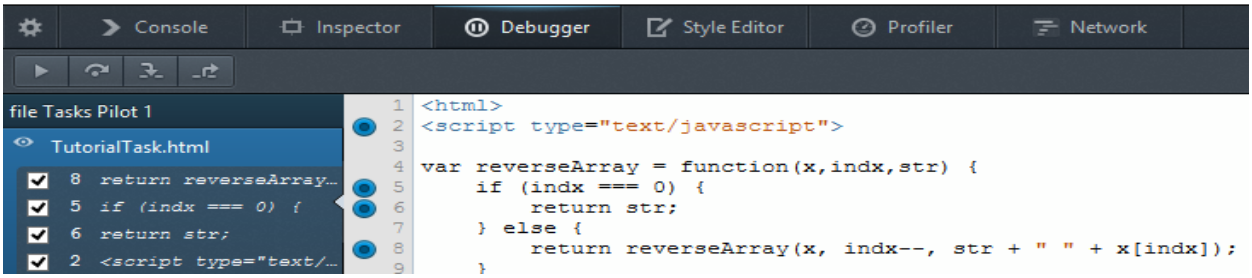
Post Session Questionnaire (Firefox Developer Tools participants):

1. Rate your thoughts on the ‘Inspector’ feature of the tool.



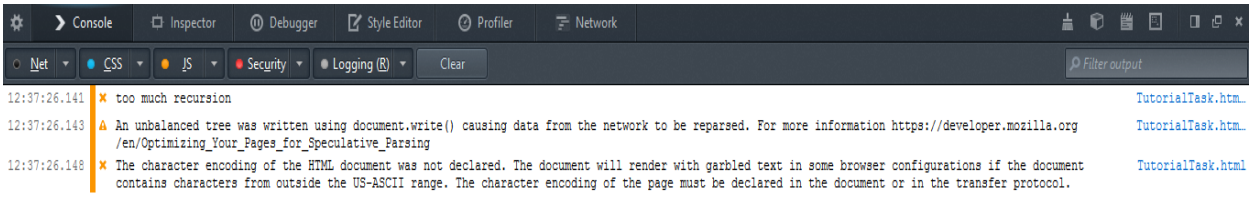
Not helpful — Very helpful

2. Rate your thoughts on the ‘Debugger’ feature of the tool.



Not helpful — Very helpful

3. Rate your thoughts on the ‘Console’ feature of the tool.



Not Helpful — Very helpful

4. How comfortable were you with using the debugging tool?

Not comfortable Very comfortable

5. How successful were you in accomplishing what you were asked to do?

Perfect Failure