

Lab 2: JavaScript Coin Flipper

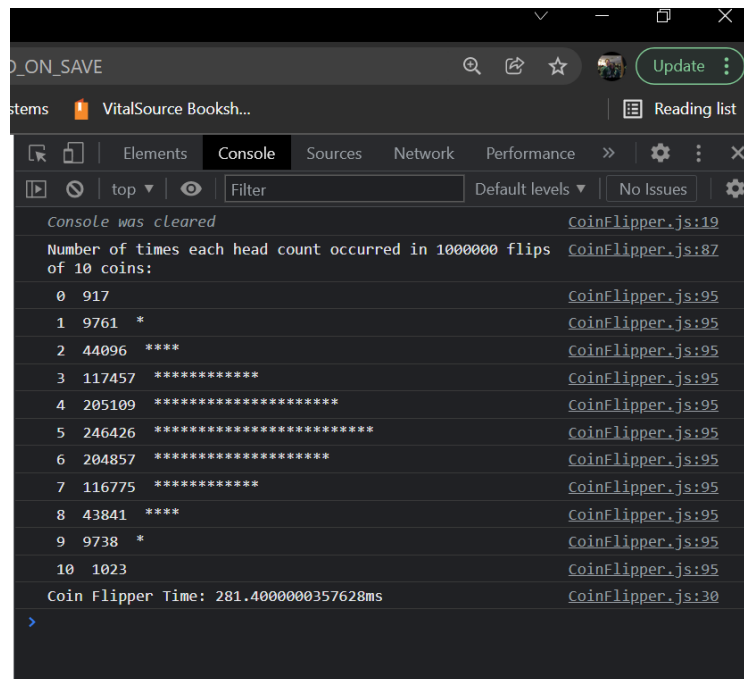
Author: Nigel Nelson
Course: SE2840/021

Introduction:

This lab acts as an exercise in creating a JavaScript application in a web browser. The application collects input from a user for the number coins to flip and the number of times to flip those coins. The application then executes for the specified number of coin flips, and measures how long it takes to accomplish this. Finally, the application outputs to the console a histogram detailing how many times of each possible number of head flips was observed in the total number of coin flips and also the time it took to execute the coin flips. Students were provided a functional version of this program implemented in Java, and challenged to convert it to the JavaScript equivalent. Students were also provided stub JavaScript and HTML files that were to be used to mirror the functionality of the Java Coin Flip application. After the completion of the JavaScript implementation, experiments were carried out to analyze the performance of the Java implementation vs the JavaScript implementation, and the effect that different browsers had on the performance of the JavaScript implementation.

Results:

Chrome Screen Shot:

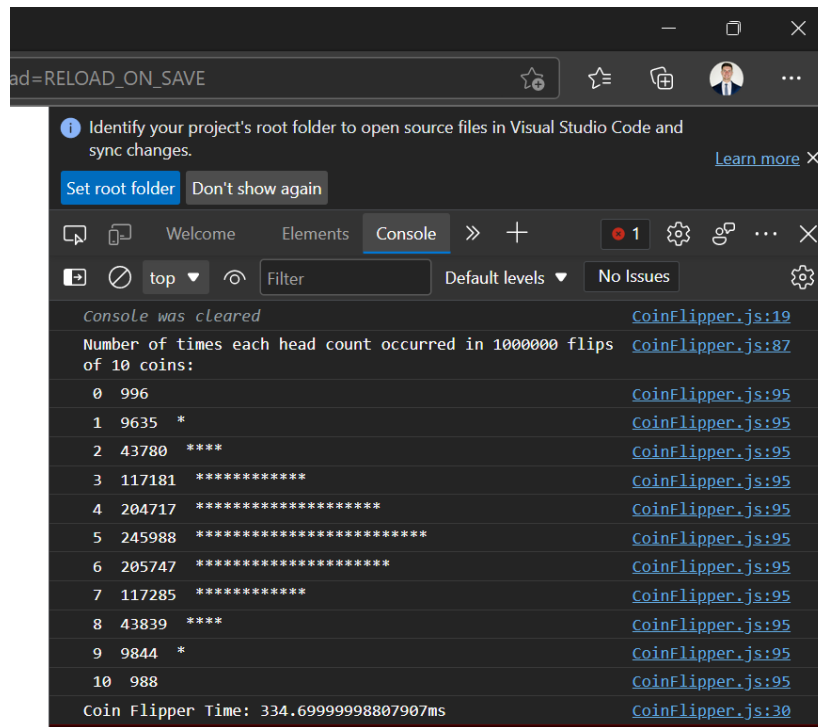


```
Console was cleared
Number of times each head count occurred in 1000000 flips of 10 coins:
0  917
1  9761 *
2  44096 ****
3  117457 *****
4  205109 *****
5  246426 *****
6  204857 *****
7  116775 *****
8  43841 ****
9  9738 *
10 1023
Coin Flipper Time: 281.4000000357628ms
```

Edge Screen Shot:

Lab 2: JavaScript Coin Flipper

Author: Nigel Nelson
Course: SE2840/021



```
ad=RELOAD_ON_SAVE

Identify your project's root folder to open source files in Visual Studio Code and sync changes.
Set root folder Don't show again

Welcome Elements Console 1
top Filter Default levels No Issues

Console was cleared CoinFlipper.js:19
Number of times each head count occurred in 1000000 flips of 10 coins: CoinFlipper.js:87
0 996 CoinFlipper.js:95
1 9635 * CoinFlipper.js:95
2 43780 **** CoinFlipper.js:95
3 117181 ***** CoinFlipper.js:95
4 204717 ***** CoinFlipper.js:95
5 245988 ***** CoinFlipper.js:95
6 205747 ***** CoinFlipper.js:95
7 117285 ***** CoinFlipper.js:95
8 43839 **** CoinFlipper.js:95
9 9844 * CoinFlipper.js:95
10 988 CoinFlipper.js:95
Coin Flipper Time: 334.69999998807907ms CoinFlipper.js:30
```

Times for recording for flipping 10 coins 1,000,000 times:

Java Coin Flipper implementation:

Trial Number	Execution Time (ms)
1	312
2	268
3	407
4	476
5	381
6	418
7	357
8	289
9	510
10	490
Average:	390.8

Lab 2: JavaScript Coin Flipper

Author: Nigel Nelson
Course: SE2840/021

JavaScript Coin Flipper implementation:

Trial Number	Execution Time (ms)
1	286.800
2	270.400
3	300.400
4	268.300
5	293.600
6	262.400
7	290.100
8	314.200
9	244.200
10	301.5
Average:	281.05

Analysis:

1. Was JavaScript faster or slower than Java? Why do you think this is the case?
 1. The JavaScript implementation was faster than the Java implementation. This may be due to the fact that the browser was able to store the script in cache, and as such gain a performance advantage. This could also be due to the fact that the JavaScript engine used, V8 for Chrome, may have better optimizations for loops and array storage, which is the biggest component of this program, than the optimizations offered by the JVM used.
2. What browsers did you use in your evaluation? Which one was the fastest, slowest, etc? Why do you think this is the case?
 1. I used Chrome and Microsoft Edge. Chrome was marginally faster than Microsoft edge, usually by 10ms or less. This is likely due to differences in the JavaScript engines used, where the Chromes V8 engine may have better loop or storage optimizations than Microsoft Edge's Chakra engine. The explanation for the marginal difference is likely due to the trivial nature of this program, where performance differences may be better seen in a more complex application.

Lab 2: JavaScript Coin Flipper

Author: Nigel Nelson
Course: SE2840/021

Summary:

From this lab I learned much about JavaScript, a programming language which prior to this lab I was completely unfamiliar with. My biggest generalized take away is that JavaScript is essentially a combination of Java and Python, making it very easy to learn. The brackets and end of line semi colons was syntactically very similar to Java, while the weak typing and interpreted nature was very similar to Python. In addition, I learned how to gather user input using the “prompt()” pop-up, and display alerts using the “alert()” function. I also learned that JavaScript’s execution is browser dependent and can have performance differences across browsers. Lastly, I learned how the selection of programming languages is a crucial phase of designing an application if performance is of the utmost importance, as for this Coin Flip application the JavaScript implementation was ~100ms faster on average than the Java version.

Suggestions:

- It would be great if the instructions could be presented in a pdf as I like going through and checking off requirements as I go.
- A list of resources for analyzing performance differences between Java and JavaScript would helpful as I struggled to find good articles that helped me understand why my JavaScript implementation may have been faster than the Java Implementation.

Highlights of the Lab:

- I really enjoyed being provided a Java implementation. It allowed me to see similarities and differences between Java and JavaScript that I believe allowed me to learn quicker.
- I appreciated the hints and tips sections of the lab write-up, it exposes me to functions that I may have otherwise not encountered.