camelCase vs kebab-case

Experiment 2, Experimentation & Evaluation 2024

Abstract

This study explores whether the style of identifiers—camelCase or kebab-case—impacts reading performance. Using a custom web-based application, 24 participants completed 20 tasks requiring them to locate specific identifiers from lists containing distractors, under controlled conditions. Participants were categorized by programming experience and age to evaluate potential variability. The study utilized a single-factor, within-subject design, ensuring that each participant was exposed to both identifier formats across the 20 randomized tasks. Performance was measured based on the time taken to correctly identify the target identifier. Statistical analysis revealed no significant difference in reading speed between the two identifiers. This finding supports the Null Hypothesis, suggesting that identifier styles does not significantly influence reading efficiency.

1. Introduction

The focus of our experiment is to investigate whether people read identifiers faster when written in camelCase or kebab-case.

The motivation behind this study is to determine whether using a specific separator in composed identifiers can speedup code reading.

To answer the research question, we conducted a within-subject experiment where participants identified camelCase and kebab-case identifiers in randomized tasks. Reading times were measured to compare performance across the two styles, with results analyzed to test the null hypothesis.

Hypotheses:
Null Hypothesis (H0): There is no statistically significant difference in reading speed when using camelCase compared to kebab-case for composed identifiers.

2. Method

The following subsections provide all the essential details required to replicate the experiment accurately.

2.1 Variables

The independent variable (i.e., the variable manipulated during the experiment) is the style of the composed identifiers.

Independent variable	Levels
Style	CamelCase or kebab-case.

The dependent variable (i.e., what is measured in the experiment) is the time taken to select the matching identifier.

Dependent variable	Measurement Scale
Time	Ratio scale (in ms)

The control variables (i.e., what is kept constant during the experiment) are the following:
the number of tasks per participant, the number of distractor identifiers per task and the experimental environment.

2.1 Control variable	Fixed Value
Tasks	20
Distractors	3
Environment	Web application

The blocking variables (i.e., measured potential sources of variability used to partition the experimental units into blocks, but are not part of the hypothesis) are the following:
the participants' programming experience and age.

Blocking variable	Levels
Programming experience	High: intermediate or advanced Low: beginner
Age	High: 40 years or older Low: under 40 years

2.2 Design

Type of Study (check one):

☐ **Observational Study**	☐ **Quasi-Experiment**	☐ **Experiment**

Number of Factors (check one):

☐ **Single-Factor Design**	☐ **Multi-Factor Design**	☐ Other

Between vs. Within (check one):

☐ **Between Group Design** (independent measures)	☐ **Within Subject Design** (repeated measures)	☐ Other

The experiment we designed is neither an Observational Study nor a Quasi-Experiment, as we are not looking at a phenomenon in a systematic and scientifically rigorous way in its environment, and we have complete control over manipulation of the independent variables. Therefore, it is an Experiment.
Moreover, the study employs a Single-Factorial Design, as it includes one independent variable, and a Within Subject Design, since each participant is exposed to all experimental conditions.

2.3 Participants

The participants in the experiment consisted of 24 individuals with an average age of 36 years (range: 19–68 years) and diverse professional backgrounds.
The participants were categorized into groups based on two key criteria: programming experience and age. Regarding programming experience, participants were divided into two groups: those with low experience (beginner) and those with high experience (intermediate or advanced). Similarly, participants were also categorized by age into two groups: low age (under 40 years old) and high age (40 years or older).
Recruitment was primarily conducted through direct contact with personal connections, including classmates, parents, and other students without expertise in informatics.
As the study followed a Within-Subject Design, all participants were exposed to all experimental conditions and completed the entire set of tasks using the tool.

2.4 Apparatus and Materials

The experiment utilized several key tools and components.

The programming environment was based on React version 18.3.1, a JavaScript library designed for building user interfaces. Papa Parse (version 5.4.1) was employed to facilitate the conversion of data into CSV format for analysis.

Time measurements were handled using the `Date.now()` function, a built-in JavaScript function that provides millisecond-level precision.

The execution platform for the experiment was a browser environment, specifically Google Chrome version 131.0.6778.139. Data storage and management were facilitated through React's Context API, which ensured consistent capture and availability of experimental results for export as CSV files.

2.5 Procedure

The experiment is conducted using a web-based tool designed to guide a participant through a series of tasks. Each session involves a single participant, with the total duration varying based on the time the participant requires to complete all tasks.

The session begins with the participant accessing a welcome page on the tool. This page provides clear instructions about the experiment's goal and an overview of the tasks the participant is expected to perform. The participant is informed that their task is to identify matching identifiers from a list and that their performance, in terms of time, is recorded.

After reading the instructions, the participant completes a short form to provide demographic information, including age and years of programming experience.

Following the demographic data collection, the participant proceeds to the task execution phase. The participant is presented with a total of 20 tasks, evenly and randomly distributed between camelCase and kebab-case identifiers. For each task, a short sentence consisting of two to four words, such as "move south," is displayed. Afterward, a list of four identifiers is shown, one of which corresponds to the given sentence while the other three serve as distractors. For example, after displaying "move south," the identifiers might include "move-source," "move-south" (the correct answer), "more-south," and "mover-sound." The participant is required to select the correct identifier as quickly and accurately as possible.

The tool automatically records the time taken by the participant to make each correct selection, measured in milliseconds. Once all 20 tasks are completed, the tool displays a completion message to the participant. The results, including task performance data and demographic information, are stored in a CSV file for later analysis.

3. Results

3.1 Visual Overview

This subsection presents a visual summary of the experimental results. To ensure clarity and insight, the data is organized into a series of box plots and density plots that highlight the reading performance for `camelCase` and `kebab-case` across the entire population and all collected samples. By including the plots for the entire population, the visualizations not only highlight the influence of blocking variables (programming experience and age) but also provide a comprehensive view of the overall trends and outcomes of the experiment.

The figures are as follows:

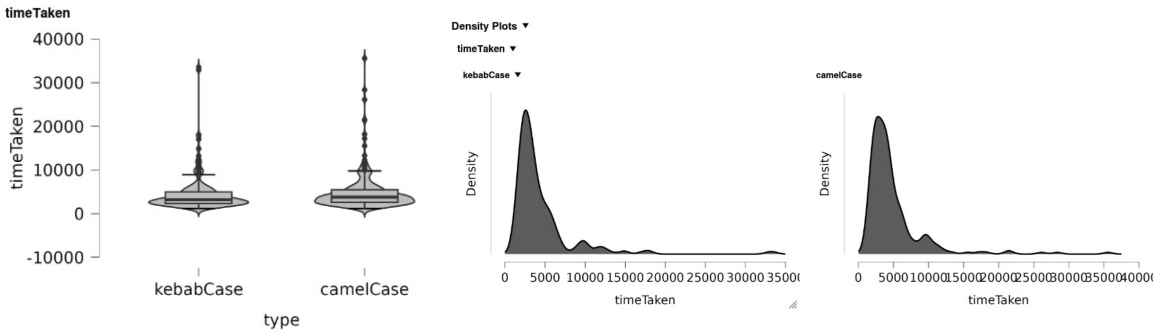


Figure 1: Comparison of `camelCase` and `kebab-case` performance across all participants.

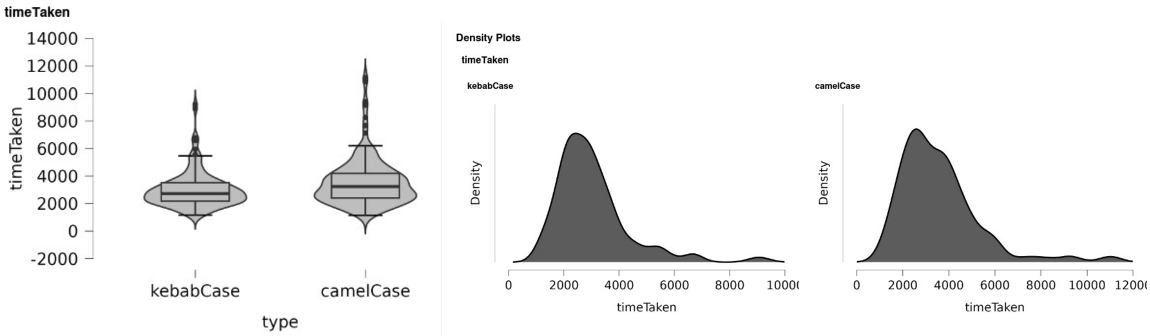


Figure 2: Comparison of `camelCase` and `kebab-case` performance among participants with high programming experience.

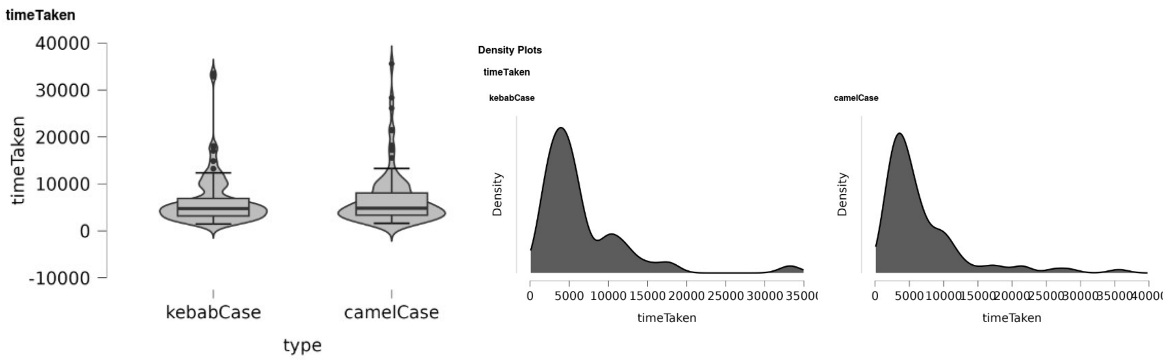


Figure 3: Comparison of `camelCase` and `kebab-case` performance among participants with low programming experience.

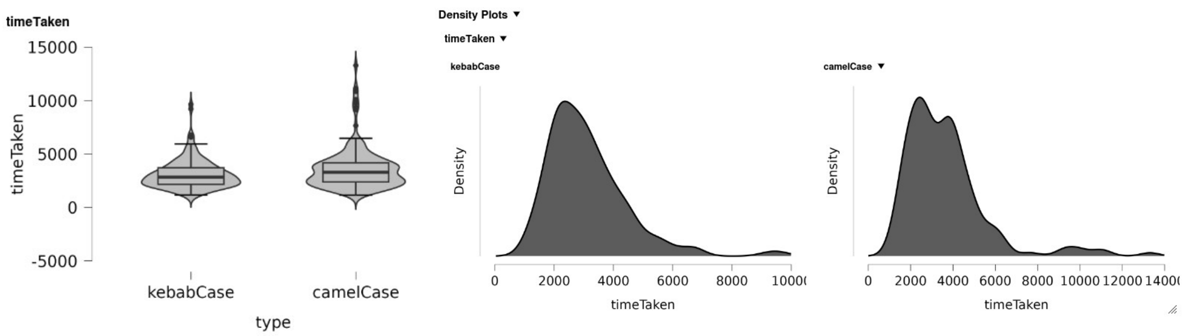


Figure 4: Comparison of `camelCase` and `kebab-case` performance among participants under 40 years of age (low age).

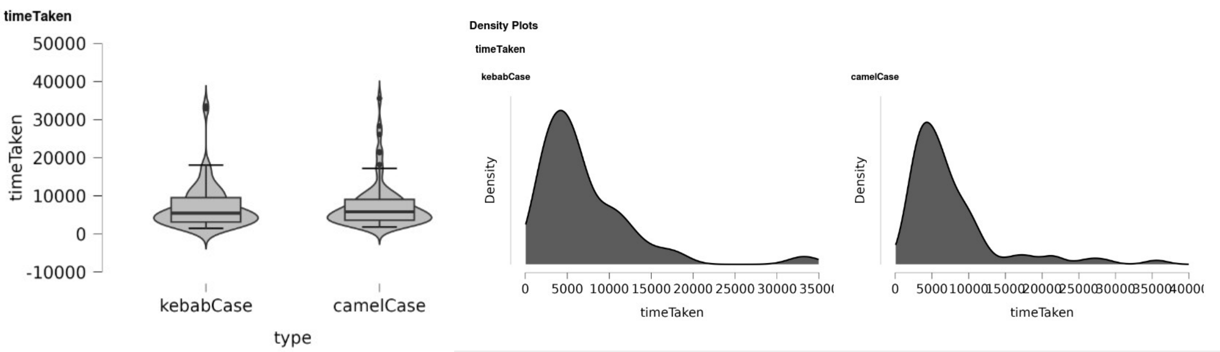


Figure 5: Comparison of `camelCase` and `kebab-case` performance among participants aged 40 and above (high age).

3.2 Descriptive Statistics

The following section provides a detailed overview of the descriptive statistics computed for the experiment. These statistics, calculated for both the entire population and the individual samples, include the minimum, first quartile (Q1), median, third quartile (Q3), maximum, mean, and standard deviation.

The minimum is the smallest value in a data set. The first quartile (also known as Q1) is the value below which 25% of the data falls. The median is the middle value of the data (when it is ordered from smallest to largest). The third quartile (also known as Q3) is the below which 75% of the data falls, and the maximum is the highest value in the set.

The mean, or average, is calculated by summing all the values in the data set and dividing by the total number of observations, offering an overall indication of the data's center. Finally, the standard deviation measures the spread or dispersion of the data around the mean, indicating how much the individual values deviate from the average.

The computed descriptive statistics are reported below:

Descriptive Statistics

	timeTaken	
	kebabCase	camelCase
Valid	240	240
Median	3182.500	3779.000
Mean	4423.637	4890.942
Std. Deviation	3988.792	4274.349
Minimum	1162.000	1151.000
Maximum	33552.000	35590.000
25th percentile	2302.250	2579.000
50th percentile	3182.500	3779.000
75th percentile	4961.750	5460.250

Table 1: Descriptive statistics for `camelCase` and `kebab-case` performance across all participants.

Descriptive Statistics ▼

	timeTaken	
	kebabCase	camelCase
Valid	140	140
Median	2724.500	3235.500
Mean	3016.364	3576.750
Std. Deviation	1339.246	1755.238
Minimum	1162.000	1151.000
Maximum	9228.000	11171.000
25th percentile	2174.250	2405.500
50th percentile	2724.500	3235.500
75th percentile	3523.000	4191.750

Table 2: Descriptive statistics for `camelCase` and `kebab-case` performance among participants with high programming experience.

Descriptive Statistics

	timeTaken	
	kebabCase	camelCase
Valid	100	100
Median	4753.500	4818.500
Mean	6393.820	6730.810
Std. Deviation	5400.966	5824.018
Minimum	1452.000	1615.000
Maximum	33552.000	35590.000
25th percentile	3186.000	3362.250
50th percentile	4753.500	4818.500
75th percentile	6901.250	8065.000

Table 3: Descriptive statistics for `camelCase` and `kebab-case` performance among participants with low programming experience.

Descriptive Statistics

	timeTaken	
	kebabCase	camelCase
Valid	160	160
Median	2850.000	3299.500
Mean	3114.706	3626.981
Std. Deviation	1360.012	1958.808
Minimum	1162.000	1151.000
Maximum	9661.000	13300.000
25th percentile	2179.750	2387.750
50th percentile	2850.000	3299.500
75th percentile	3728.250	4191.750

Table 4: Descriptive statistics for `camelCase` and `kebab-case` performance among participants under 40 years of age (low age).

Descriptive Statistics

	timeTaken	
	kebabCase	camelCase
Valid	80	80
Median	5442.500	5819.500
Mean	7041.500	7418.863
Std. Deviation	5831.088	6151.697
Minimum	1452.000	1840.000
Maximum	33552.000	35590.000
25th percentile	3166.750	3640.250
50th percentile	5442.500	5819.500
75th percentile	9533.250	9067.250

Table 5: Descriptive statistics for `camelCase` and `kebab-case` performance among participants aged 40 and above (high age).

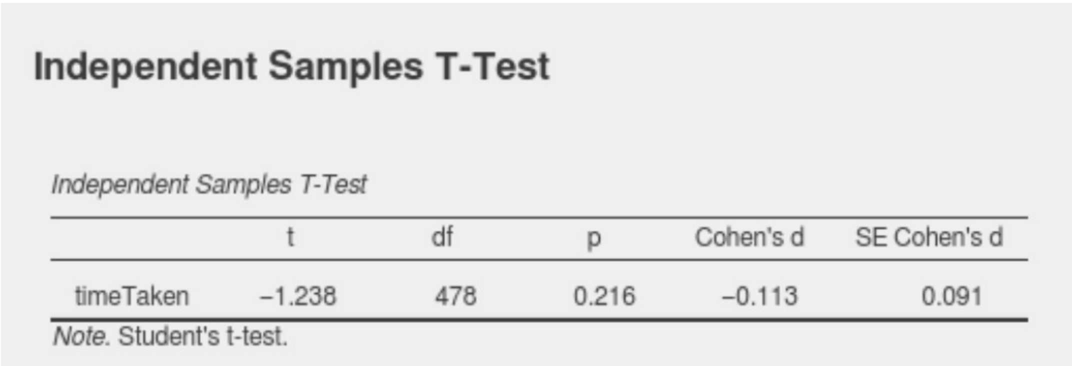
3.3 Inferential Statistics

The following section presents a detailed overview of the inferential statistics computed using the Independent Samples T-Test. These statistics, calculated exclusively for the entire population, include the t-statistic (t), degrees of freedom (df), and p-value (p). In addition to the computed inferential statistics, the descriptive measures Cohen’s d and the standard error of Cohen’s d (SE Cohen’s d) are also included as associated descriptive statistics.

The t-statistic (t) measures the size of the difference between the two groups (camelCase and kebab-case) relative to the variation in the data. A larger absolute value of t suggests a greater difference between the groups. The degrees of freedom (df) indicate the amount of information available for estimating the population parameters, which is influenced by the sample size. The p-value (p) represents the probability of observing the current data under the assumption that the null hypothesis is true.

Cohen’s d quantifies the size of the difference between the groups in terms of standard deviations, offering an indication of the practical significance of the difference. The standard error of Cohen’s d (SE Cohen’s d) reflects the precision of this effect size estimate, with smaller values indicating greater confidence in the measure.

The computed inferential statistics, along with the associated descriptive measures Cohen’s d and SE Cohen’s d, are presented below:



Independent Samples T-Test

Independent Samples T-Test

	t	df	p	Cohen's d	SE Cohen's d
timeTaken	−1.238	478	0.216	−0.113	0.091

Note. Student's t-test.

Table 1: Inferential statistics for `camelCase` and `kebab-case` performance across all participants.

4. Discussion

4.1 Compare Hypothesis to Results

The hypothesis states that there is no statistically significant difference in reading speed between camelCase and kebab-case for composed identifiers. The results of the inferential analysis on the entire population support the null hypothesis, as the p-value exceeds 0.05, confirming that there is no significant difference in reading times between the two styles. This suggests that the choice of text format does not have a systematic impact on reading performance across the entire population.

Moreover, the low value of Cohen's d (Cohen's d = -0.113) further reinforces the conclusion that any observed differences in reading performance within the population are not meaningful.

However, the descriptive statistics provided additional context. The mean and median reading times for CamelCase were consistently higher than those for kebab-case across the entire population and all the samples. In addition, CamelCase displayed slightly greater variability in reading times, as indicated by its larger standard deviation and interquartile range compared to kebab-case.

Based on these results, the findings from the inferential analysis supports the hypothesis that reading performance doesn't differ significantly between the two text styles. Although descriptive statistics suggest a potential disadvantage for CamelCase, however these trends were not strong enough to reach statistical significance at the population level.

4.2 Limitations and Threats to Validity

While the findings of this study provide useful insights, several limitations and potential threats to validity must be acknowledged. Internal validity could have been influenced by task familiarity, as participants might have improved their performance due to learning effects over the course of the 20 tasks. Despite randomizing the task order, this learning effect may still have played a role. Additionally, the time measurements, recorded using the JavaScript `Date.now()` function, might have been affected by system performance or browser processing variations, introducing potential inaccuracies. Furthermore, even though participants were given detailed instructions, there remains the possibility that they did not fully understand the tasks or the importance of prioritizing accuracy over speed, which could have skewed the results.

External validity is also a concern due to the sample demographics. The participant pool consisted of only 24 individuals with diverse but non-representative backgrounds, making it difficult to generalize the findings to larger or more specific populations, such as professional developers. The categorization of participants into two broad experience groups may have oversimplified the complexity of programming expertise, potentially masking subtler differences.

Construct validity raises additional concerns. The experiment assumed that reading speed for composed identifiers in a controlled environment accurately reflects real-world programming scenarios. However, developers typically encounter identifiers within broader contexts that could influence their interpretation and performance.

To address these limitations, future studies should consider increasing the sample size and ensuring the participant pool is more representative. Timing mechanisms with greater accuracy and reliability could minimize the risk of system-induced delays. Tasks could be designed to better reflect real-world programming scenarios, incorporating contextual cues. Expanding the range of programming expertise categories would capture finer granularity and provide a more nuanced understanding of participant performance.

4.3 Conclusions

The results support the null hypothesis, indicating no statistically significant differences in reading speed between camelCase and kebab-case styles. This suggests that the choice of identifier format does not systematically affect readability, regardless of the reader's programming experience or age.

While the descriptive statistics hint at a slight advantage for kebab-case, the inferential statistics confirm that this difference is not statistically significant, reinforcing the conclusion that neither format offers a meaningful performance advantage.

Appendix

See our GitHub repository to check for extra materials and raw data.