



Objects Lab

- Due No Due Date
- Points 1
- Submitting a website url

 (<https://github.com/learn-co-curriculum/phase-0-intro-to-js-2-objects-lab>)  (<https://github.com/learn-co-curriculum/phase-0-intro-to-js-2-objects-lab/issues/new>)

Learning Goals

- Create an `Object`
- Perform operations on an `Object`

Introduction

We covered the concepts of `Object` s in JavaScript. Now it's time to put the concepts into practice.

If you haven't already, **fork and clone** this lab into your local environment. Navigate into its directory in the terminal, then run `code .` to open the files in Visual Studio Code.

Instructions

Follow the steps below, running `npm test` as you go to get additional information from the tests. Don't forget to run `npm install` first!

Let's say we are working on a program that will keep track of a company's employees. We want to store each `employee` as an `Object` . We're starting small, so to begin with we'll only keep track of the employee's name and street address.

To start, assign an `employee` variable to an `Object` containing `name` and `streetAddress` keys; you can use whatever values you like. Use literal syntax to create your `Object` . Various updates will be applied to this variable (destructively and non-destructively) in this lab.

Once you've initialized the `employee` `Object`, you'll need to create the following four functions:

- `updateEmployeeWithKeyAndValue()` : this function should take in three arguments: an `employee Object` , a `key` and a `value` . This function should not mutate the `employee` ; it should return a *new Object* that has an updated `value` for the `key` passed in. **Hint:** use the spread operator!
- `destructivelyUpdateEmployeeWithKeyAndValue()` : this function should work the same as `updateEmployeeWithKeyAndValue()` but it *should* mutate the `employee Object` passed in.
- `deleteFromEmployeeByKey()` : this function should take in a `employee Object` and a `key` . It should delete the property with that `key` from the `employee Object` . This should not mutate the original `employee Object` ; it should return a *new Object* that doesn't include the identified key-value pair. **Hint:** use the spread operator!
- `destructivelyDeleteFromEmployeeByKey()` : this function should work the same as `deleteFromEmployeeByKey()` but it *should* mutate the `employee Object` .

As you work on your functions, be sure to think about when to use dot notation vs. bracket notation.

After you have all the tests passing, remember to commit and push your changes up to GitHub, then submit your work to Canvas using CodeGrade.

Conclusion

In this lab, we practiced creating an `Object` and performing operations on it.

Resources

- [MDN: Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object)  [_ \(https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object\)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object)