1. Introduction

* Lesson 1 details how to create, access, and modify objects.
* Lesson 2 examines how JavaScript functions are first-class functions.
* Lesson 3 illustrates JavaScript's abstractions over traditional classes and inheritance.

Go from linear to object oriented programming.

const mixedArray = [6, 'banana', false, ['apple', 'banana', 'orange', 'grape', 'lychee']];

The object is one of the most important data structures in JavaScript. After all, you're currently taking an entire course on object-oriented programming!

const car = {

color: 'red',

year: 1992,

isPreOwned: true

};

const menu = {

name: 'Salted Caramel Ice Cream',

price: 2.95,

ingredients: ['butter', 'ice cream', 'salt', 'sugar']

};

Object have key/value pairs, curly braces, and are unordered.

In JavaScript, an object is an unordered collection of properties. Each property consists of a key/value pair, and can reference either a primitive (e.g., strings, numbers, booleans, etc.) or another object. Unlike elements in an array, which are accessed by a numeric index, properties in objects are accessed by their key name using either square bracket notation or dot notation.

1. Create and Modify Properties

// Using literal notation:

const myObject = {};

// Using the Object() constructor function:

const myObject = new Object();

Recall that since objects are mutable, not only can we modify existing properties (or even add new ones) -- we can also delete properties from objects.

With a few exceptions, properties in objects are mutable.

1. Invoking Object Methods

A method is a property of an object whose value is a function. Methods are called on objects in the following format: object.method().

const myArray = [ function alerter() { alert('Hello!'); } ];

myArray[0]();

const chameleon = {

color: 'green',

changeColor: function() {

if (this.color === 'green') {

this.color = 'pink';

} else {

this.color = 'green'; }}};

1. Beware of Globals

chameleon.lookAround();

The chameleon object is left of the dot. Therefore, inside the .lookAround() method, this will refer to the chameleon object!

When a regular function is invoked, the value of this is the global window object.

The keywords var, let, and const are used to declare variables in JavaScript. var has been around since the beginning of the language, while let and const are significantly newer additions (added in ES6).

Counterintuitively, though, global variables and functions are not ideal. There are actually a number of reasons why, but the two we'll look at are:

* Tight coupling - In tight coupling, pieces of code are joined together in a way where changing one unintentionally alters the functioning of some other code:
* Name collisions - A name collision occurs when two (or more) functions depend on a variable with the same name.

1. Extracting Properties and Values

Support for Object.keys() and Object.values()

const triangle = {

type: 'polygon',

sides: 3,

sumOfAngles: 180,

equilateral: true,

equiangular: true

};

Object.keys(triangle);

1. Lesson Summary