

# Introducing JavaScript objects

In JavaScript, most things are objects, from core JavaScript features like arrays to the browser [APIs](#) built on top of JavaScript. You can even create your own objects to encapsulate related functions and variables into efficient packages and act as handy data containers. The object-based nature of JavaScript is important to understand if you want to go further with your knowledge of the language, therefore we've provided this module to help you. Here we teach object theory and syntax in detail, then look at how to create your own objects.

## Prerequisites

Before starting this module, you should have some familiarity with [HTML](#) and [CSS](#). You are advised to work through the [Introduction to HTML](#) and [Introduction to CSS](#) modules before starting on JavaScript.

You should also have some familiarity with JavaScript basics before looking at JavaScript objects in detail. Before attempting this module, work through [JavaScript first steps](#) and [JavaScript building blocks](#).

**Note:** If you are working on a computer/tablet/other devices where you are not able to create your own files, you could try out (most of) the code examples in an online coding program such as [JSBin](#) or [Glitch](#).

## Guides

### [Object basics](#)

In the first article looking at JavaScript objects, we'll look at fundamental JavaScript object syntax, and revisit some JavaScript features we've already looked at earlier on in the course, reiterating the fact that many of the features you've already dealt with are in fact objects.

### [Object prototypes](#)

Prototypes are the mechanism by which JavaScript objects inherit features from one another, and they work differently from inheritance mechanisms in classical object-oriented programming languages. In this article, we explore how prototype chains work.

### [Object-oriented programming](#)

In this article, we'll describe some of the basic principles of "classical" object-oriented programming, and look at the ways it is different from the prototype model in JavaScript.

## [Classes in JavaScript](#)

JavaScript provides some features for people wanting to implement "classical" object-oriented programs, and in this article, we'll describe these features.

## [Working with JSON data](#)

JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax, which is commonly used for representing and transmitting data on the web (i.e., sending some data from the server to the client, so it can be displayed on a web page). You'll come across it quite often, so in this article, we give you all you need to work with JSON using JavaScript, including parsing the JSON so you can access data items within it, and writing your own JSON.

## [Object building practice](#)


In previous articles we looked at all the essential JavaScript object theory and syntax details, giving you a solid base to start from. In this article we dive into a practical exercise, giving you some more practice in building custom JavaScript objects, which produce something fun and colorful — some colored bouncing balls.

# Assessments

## [Adding features to our bouncing balls demo](#)

In this assessment, you are expected to use the bouncing balls demo from the previous article as a starting point, and add some new and interesting features to it.

## See also

[Learn JavaScript](#) 

An excellent resource for aspiring web developers — Learn JavaScript in an interactive environment, with short lessons and interactive tests, guided by automated assessment. The first 40 lessons are free, and the complete course is available for a small one-time payment.

## Help improve MDN

Was this page helpful to you?



[Learn how to contribute.](#)

This page was last modified on Mar 5, 2024 by [MDN contributors](#).

