Introduction

You'll learn about the topics this chapter contains, which include using arrays and array methods.

We'll cover the following

- What does this chapter include?

The ancient Greek poet Archilochus wrote, "A fox knows many things, but a hedgehog one important thing." The great historian Isaiah Berlin said all thinkers are either hedgehogs or foxes. I think the same is true of syntax.

As you've seen, `const` is a hedgehog. It can only do one thing—*make an unchanging declaration*. By only doing one thing, it makes your code *readable and predictable*. As you'll see in upcoming tips, array methods are all hedgehogs. They can do only one thing on an array. But they do it well, so you can safely predict outcomes without diving into the details.

For the most part, you want to stick with syntax that does one thing very well. But there are times when you need things to be flexible. An array is the ultimate fox because it can do many things. In fact, it can do almost anything you'd ever want for a collection of information. More importantly, many other collections use concepts that you'd most often associate with arrays.

For example, when you have a string, **'hedgehog'**, you have a lot of available actions you'd normally perform on arrays.

- You can get the size: `'hedgehog'.length` will return `8`.
- You can also pick out individual letters by index: `'hedgehog'[3]` will return `'g'`.

There are so many other methods that it would take too long to list them all.

These methods don't belong to arrays specifically (they rely on a property called `Iterator`), but they're most intuitively connected to arrays. When you study arrays carefully, you'll gain many insights into other data structures. Arrays know many things. They are foxes.

What does this chapter include?

In this chapter, you'll see that arrays are becoming better than ever. Not only are they a good choice for many data needs, but they have new syntax that reduces many common actions to one-liners while simultaneously reducing mutations that can cause subtle bugs. And pay attention—you'll see the same ideas in later tips.

To begin, you'll see how data can always be converted to arrays, including converting other collections (such as objects) to arrays when necessary. From there, you'll learn new syntax, such as `includes()`, to test existence in arrays and, crucially, the *spread operator* symbolized by three dots (`...`). The spread operator is so important in the modern use of arrays that the next two tips will explore how the spread operator changes how you use arrays in your code. Pay close attention—you'll see the spread operator in many future tips.

To keep code readable, you should stick with simple, predictable approaches (hedgehogs). But to make code flexible, you need arrays to move between structures. It's a tough balancing act, but you need both. Everything you do in JavaScript will be easier if you have a clear understanding of arrays.

Time to jump in and see how arrays provide a level of flexibility you won't find in most collections.