Well-known Symbols

name clashes occurring in objects and how to avoid them using Symbols

When using symbols as identifiers for objects, we don't have to set up a global registry of available identifiers. We also save the creation of a new identifier, as all we need to do is create a `Symbol()`.

The same holds for external libraries.

In the next lesson, we'll talk about some well-known symbols and how they change the behavior of your code.

Well-known Symbols

There are some well-known symbols defined to access and modify internal JavaScript behavior. For example, you can redefine built-in methods, operators, and loops. Redefining standard language behavior usually confuses other developers. Ask yourself, will this skill move you forward in your career?

We will not focus on well-known symbols in this section. If there is a valid use case for it, I will signal it in the corresponding lesson. Otherwise, I suggest staying away from manipulating the expected behavior of your code.

Before learning newer concepts, let's solve some exercises.