* - [Instructor] As you start working with JavaScript, you'll quickly notice these files tend to get really large and kind of hard to work with .
* You can see here, this is a fairly straightforward script that isn't too much going on but you can already see it's getting really long .
* And there's a lot of pieces that rely on other pieces to be in place and figuring out how everything is wired together requires a lot of scrolling up and down .
* To solve this problem, we now have something called JavaScript modules .
* **JavaScript modules allow us to break pieces out of a JavaScript file** and place them in a separate file and then import them back into the original file again .
* In the exercise files four, 02\_04, you can see this in action .
* So here we have script .
* js .
* And at the very top of script .
* js it says import backpack from backpack .
* js .
* If I opened backpack .
* js, you'll see here we have a constant called backpack .
* That's what's being imported .
* And at the very bottom, it says export default backpack .
* So what's happening here is in backpack .
* js, we've defined a constant called backpack and exported it saying to the browser this entity called backpack, in this case an object, can be used by any other file if it's imported into that other file .
* Then in script .
* js, we import that as that entity, the object, and use it just as if it was inside this file .
* Now if you compare script .
* js from 02\_03 with script .
* js from 02\_04, you'll see the only thing I've done here is taken this constant backpack and one of its helper functions, broken them out of this file and placed them into its own file .
* Everything else is exactly the same because thanks to the JavaScript modules, backpack, this object because it's been imported, works as if it was sitting inside script .
* js .
* It's just sitting in a separate file .
* Now to get this to work in practice, you have to do one more thing .
* Inside index .
* html, you need to tell the browser, first of all there are two files now .
* And second of all, these two files on our modules, they may rely on each other and it's important that they're both loaded before things work properly .
* Interestingly, ***when you set the type attributes to module, both of these automatically get deferred .***
* So they will only run after everything else has happened to ensure that the browser has all the modules available before anything gets rendered out .
* JavaScript modules is a fairly advanced and very new feature but there's a reason why I'm showing it to you .
* If you work with react or view or anything else, this is standard practice .
* In fact, the module thinking came from these JavaScript frameworks .
* And anytime you go into a JavaScript framework, you'll see this import and export all the time .
* So now you know exactly what it is .
* And you also know you can actually do this in straight up vanilla JavaScript .
* You don't have to have a JavaScript framework .
* There is **one caveat** to doing this though and it's an important one .
* So let me show you .
* If I go to the exercise files for the previous movie, 02\_03, you'll remember that the backpack object is sitting inside the script .
* js file .
* And that means if I type in backpack here, I can get to the backpack object directly from the console .

Graphical user interface

Description automatically generated with medium confidence

* It's available to me .
* However, if I go to 02\_04, the output is still the same .
* But if I try to call in backpack there is no object called backpack here .

Graphical user interface, text, application

Description automatically generated

* That's because the backpack object is **only available in the context of script .**
* **js .**
* It is scoped only to this file in the current state, not to the browser .
* So that means I've taken the backpack object out of the equation where the browser is concerned .
* So I can't work with it the same way I did before .
* This is important to keep in mind, because it tells you what type of things you should be putting in modules and what types of things you should be putting in the main file .
* So for example, putting an object in a module like I've done here, probably not a good idea .
* But later on in the course, I'll show you how putting a class, so effectively a template to create objects in a separate module, makes a lot of sense because that way you can reference it whenever you need it and it doesn't have to clutter up your code .
* Modules are cutting edge JavaScript .
* And like I said, you'll encounter them all the time in JavaScript frameworks, but they also work in vanilla JavaScript if you just set it up correctly .