Lesson Weekend

Introduction to Programming (/introduction-to-programming) / Arrays and Looping (/introduction-to-programming/arrays-and-looping) / Debugging in JavaScript: Using a linter

Text

Sometimes you won't be able to find a syntax error no matter how hard you look. Or your code may work correctly but still have an error. For instance, maybe you forgot to scope a variable with `let` or `const` or maybe you forgot a semicolon.

Most developers use a **linter** to check their code for errors. Linters are very helpful both for fixing errors as they arise and for writing clean code.

There are advantages and disadvantages to beginning coders using linters. It's important to develop a good eye for code, including finding errors. If you rely on a linter too much early on, you may not develop this skill as fully. On the other hand, many errors are genuinely hard for beginners to find and a linter can save us a lot of frustration by showing us exactly where the error is. Finally, a linter is a bit like an automated mentor — as you write code, the linter will let us know if something doesn't look right, just like a real-life mentor teaching us how to code.

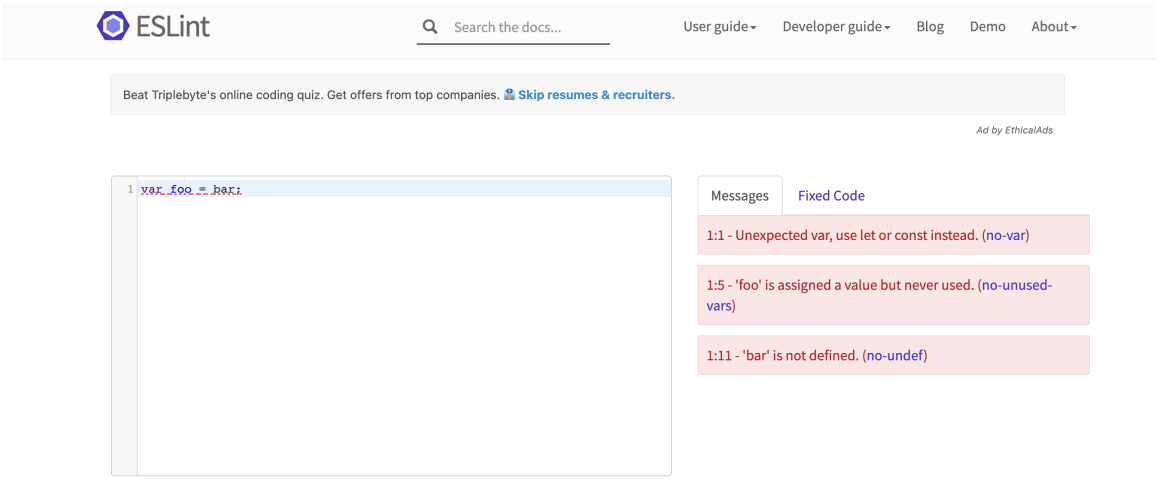
Debugging with a Linter

There are several approaches to linting. One is to add an extension via VS Code. Another is to bundle a development environment to include a linter that automatically lints our code. We will be doing the latter in Intermediate JavaScript. The tool we'll use is called ESLint. Both of the above approaches involve some configuration that might feel a bit overwhelming this early in the program. Also, we'd rather you try finding problems in your code *before* using a linter — and the above approaches will lint as you go.

Fortunately, there is also a browser-based demo of ESLint here: ESLint Demo (https://eslint.org/play/#eyJ0ZXh0IjoidmFyIGZvbyA9IGJhcjsiLCJvcHRpb25zIjp7InBhcnNlck9wdGlvbnMiOnsiZWNtYVZlcnN

Make sure you use this exact link so you are using the configuration we've set. We'll discuss the configuration further later in this lesson.

If we take a look, we'll see the following:



The window on the left is where we can insert any code we want to lint. The code that's already there (`var foo = bar;`) is the default code that ESLint provides. We can replace it by pasting or typing any code we want into the window.

The window on the right shows any errors in the code. As we can see, there are several even with the default code provided!

- First, the example uses `var` . We don't want that! ESLint tells us to use `let` or `const` instead. Note the `1:1` at the beginning of the error. That's the exact line and character where the error begins. In this case, it's the first character of line one. That's a really nice thing about ESLint — it can pinpoint an error when our eyes are too tired to see it.
- Second, we see an `assigned a value but never used` . This just means a variable has been declared but our code doesn't use it. If this were a full code snippet, this would be a bad thing; it would mean the variable is unnecessary and should be removed. However, if you paste in a smaller snippet of code that doesn't include all the variables that you're using, you can ignore the error.
- Third, we get an `is not defined` error. That's because `bar` is never defined. This can be an actual problem in the code (such as if we mistyped the variable name), but once again, if you're pasting in a snippet of code, it's possible you didn't include the snippet where the variable was originally defined.

You can try experimenting with the tool yourself. For instance, if you delete the semicolon at the end, a `Missing semicolon` error will pop up.

Configuration

You can click on the *Rules Configuration* tab to see all the potential rules and configurations you can have with ESLint. We recommend not looking too closely at it, though, because there's an overwhelming amount of stuff. Instead, we'll quickly go over the configuration we've set up. **Remember, the configuration is set up based on the link we provide above.** If you go to the page without the customized link, you won't have our custom configuration.

We've added the following custom configurations:

- Use ES2015 (a specific subset of JavaScript that came out in 2015). Technically, we are using ES5 for most of our code in the first four sections of the program (ES5 came out before ES2015). However, `let` and `const` came out with ES2015.
- Use `let` and `const` , not `var` .
- Require semicolons.
- Recognize the browser environment.

While you are welcome to update the configuration if you wish, we don't recommend it. And once again, actually configuring a linter can add an additional layer of confusion for beginners. A linter is a tool to help you fix errors in your code. If you find that you are trying to debug the linter itself, or trying to understand how the linter is working, you are focusing on the wrong thing! Your focus right now should be on learning JavaScript and web browser tools. You are not required to use a linter but it can be a helpful tool. At this point, we recommend trying to write, debug, and fix code without help from this linter.

You should ideally only check your code with this linter for two reasons:

- You're stumped by an error that you can't solve on your own.
- Your code is working fine but you want to make sure it's actually error-free code (for instance, you might not be sure about the placement of semicolons).

If your code is error-free, there will be a *Lint-free!* message in the right-hand pane. That doesn't necessarily mean your code works as expected or that it's well-written code. However, it does mean that your code is free of syntax errors and any bad practices that we've configured ESLint to be on the lookout for (such as missing semicolons).

Previous (/introduction-to-programming/arrays-and-looping/adding-and-removing-html-elements)
Next (/introduction-to-programming/arrays-and-looping/mentorship-lunch-talk)

Lesson 11 of 50
Last updated February 28, 2023

disable dark mode

Epicodus (http://www.epicodus.com)

© 2023 Epicodus (http://www.epicodus.com/), Inc.