Lesson Wednesday

Introduction to Programming (/introduction-to-programming) / JavaScript and Web Browsers (/introduction-to-programming/javascript-and-web-browsers) / Debugging in JavaScript: Using console.log()

Text Cheat sheet

Debugging Using `console.log()`

- `console.log()` allows us to log messages in the DevTools console. It can be helpful for making sure code is reached or checking the value of variables.
- `console` is a Web API and `log()` is a method of the `console` object. This object has other methods and properties that you can explore on your own time.
- Don't forget that the default behavior of a form submission event is to load a new page. We don't want to do that so we need to use `event.preventDefault()` to prevent that behavior.
- We can tell that our application has attempted to make a GET request to a server and reload the page if a `?` is added within the URL in our browser address bar.

- When you use multiple `console.log()` s, you should add a description so they are easier to track in the DevTools console. You can chain text and variables together with `+` or `,` . Here are two examples from the lesson:
  - `console.log("verbInput = " + verbInput);`
  - `console.log("targeting span in story =", document.querySelector("span#verb").innerTex);`
- If you add `console.log()` statements directly to your code, make sure to remove them after you are done. Otherwise, your code will look sloppy.
- While `console.log()` is a useful tool, don't overuse it. Adding breakpoints and other debugging tools are generally more effective. We'll learn about these in upcoming lessons.

Previous (/introduction-to-programming/javascript-and-web-browsers/other-ways-to-organize-ui-logic)
Next (/introduction-to-programming/javascript-and-web-browsers/practice-forms)
Lesson 56 of 75
Last updated March 24, 2023

disable dark mode

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