



EPUB/PDF

[Home](#) → [The JavaScript language](#) → [An introduction](#)

📅 5th April 2021

# Developer console

Code is prone to errors. You will quite likely make errors... Oh, what am I talking about? You are *absolutely* going to make errors, at least if you're a human, not a [robot](#).

But in the browser, users don't see errors by default. So, if something goes wrong in the script, we won't see what's broken and can't fix it.

To see errors and get a lot of other useful information about scripts, "developer tools" have been embedded in browsers.

Most developers lean towards Chrome or Firefox for development because those browsers have the best developer tools. Other browsers also provide developer tools, sometimes with special features, but are usually playing "catch-up" to Chrome or Firefox. So most developers have a "favorite" browser and switch to others if a problem is browser-specific.

Developer tools are potent; they have many features. To start, we'll learn how to open them, look at errors, and run JavaScript commands.

## Google Chrome

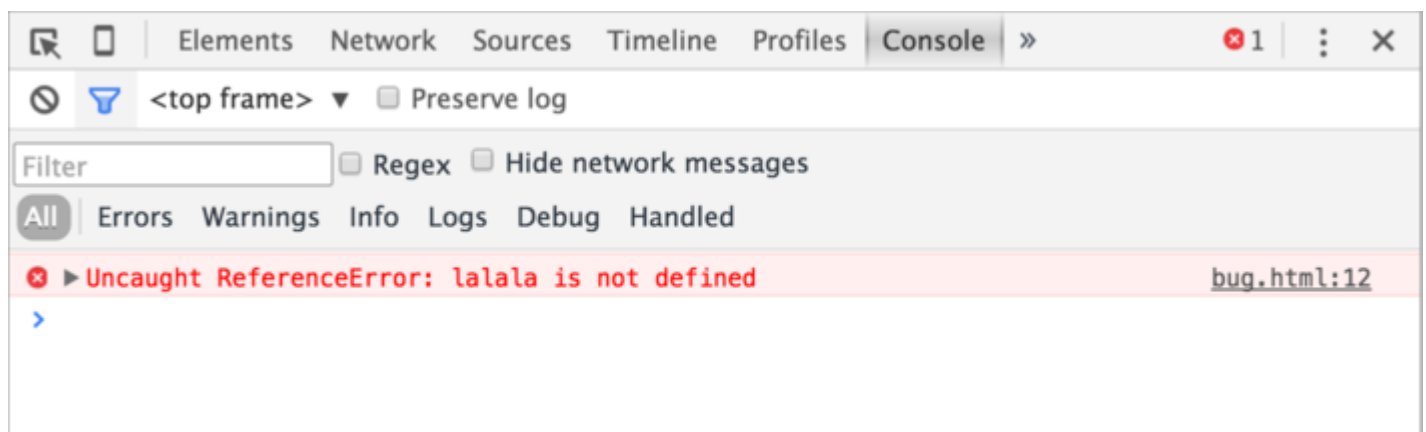
Open the page [bug.html](#).

There's an error in the JavaScript code on it. It's hidden from a regular visitor's eyes, so let's open developer tools to see it.

Press **F12** or, if you're on Mac, then **Cmd+Opt+J**.

The developer tools will open on the Console tab by default.

It looks somewhat like this:



The exact look of developer tools depends on your version of Chrome. It changes from time to time but should be similar.

- Here we can see the red-colored error message. In this case, the script contains an unknown “lalala” command.
- On the right, there is a clickable link to the source `bug.html:12` with the line number where the error has occurred.

Below the error message, there is a blue `>` symbol. It marks a “command line” where we can type JavaScript commands. Press `Enter` to run them.

Now we can see errors, and that’s enough for a start. We’ll come back to developer tools later and cover debugging more in-depth in the chapter [Debugging in the browser](#).

### **i** Multi-line input

Usually, when we put a line of code into the console, and then press `Enter`, it executes.

To insert multiple lines, press `Shift+Enter`. This way one can enter long fragments of JavaScript code.

## Firefox, Edge, and others

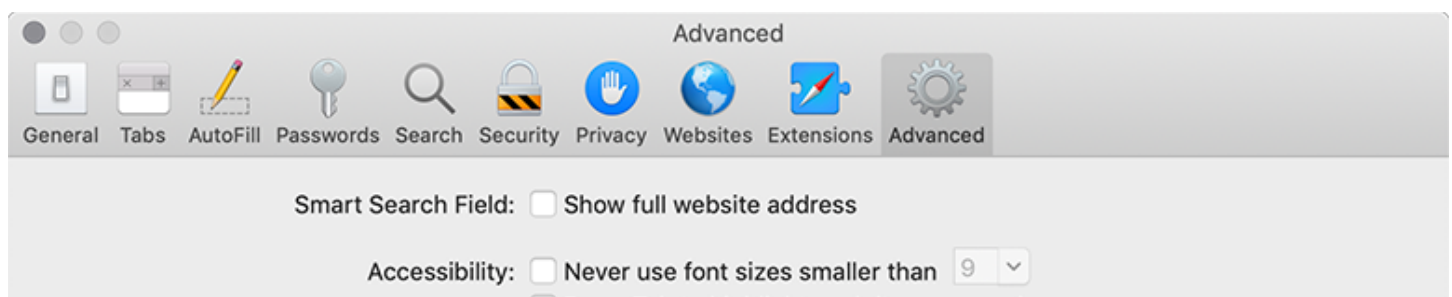
Most other browsers use `F12` to open developer tools.

The look & feel of them is quite similar. Once you know how to use one of these tools (you can start with Chrome), you can easily switch to another.

## Safari

Safari (Mac browser, not supported by Windows/Linux) is a little bit special here. We need to enable the “Develop menu” first.

Open Preferences and go to the “Advanced” pane. There’s a checkbox at the bottom:



Now `Cmd+Opt+C` can toggle the console. Also, note that the new top menu item named “Develop” has appeared. It has many commands and options.

## Summary

- [Developer tools](#) allow us to see errors, run commands, examine variables, and much more.
- They can be opened with `F12` for most browsers on Windows. Chrome for Mac needs `Cmd+Opt+J`, Safari: `Cmd+Opt+C` (need to enable first).

Now we have the environment ready. In the next section, we'll get down to JavaScript.



Previous lesson

Next lesson



Share



[Tutorial map](#)

## Comments

- If you have suggestions what to improve - please [submit a GitHub issue](#) or a pull request instead of commenting.
- If you can't understand something in the article – please elaborate.
- To insert few words of code, use the `<code>` tag, for several lines – wrap them in `<pre>` tag, for more than 10 lines – use a sandbox ([plnkr](#), [jsbin](#), [codepen](#)...)