



[www.frontendgenius.com](http://www.frontendgenius.com)

# Debugging in JavaScript

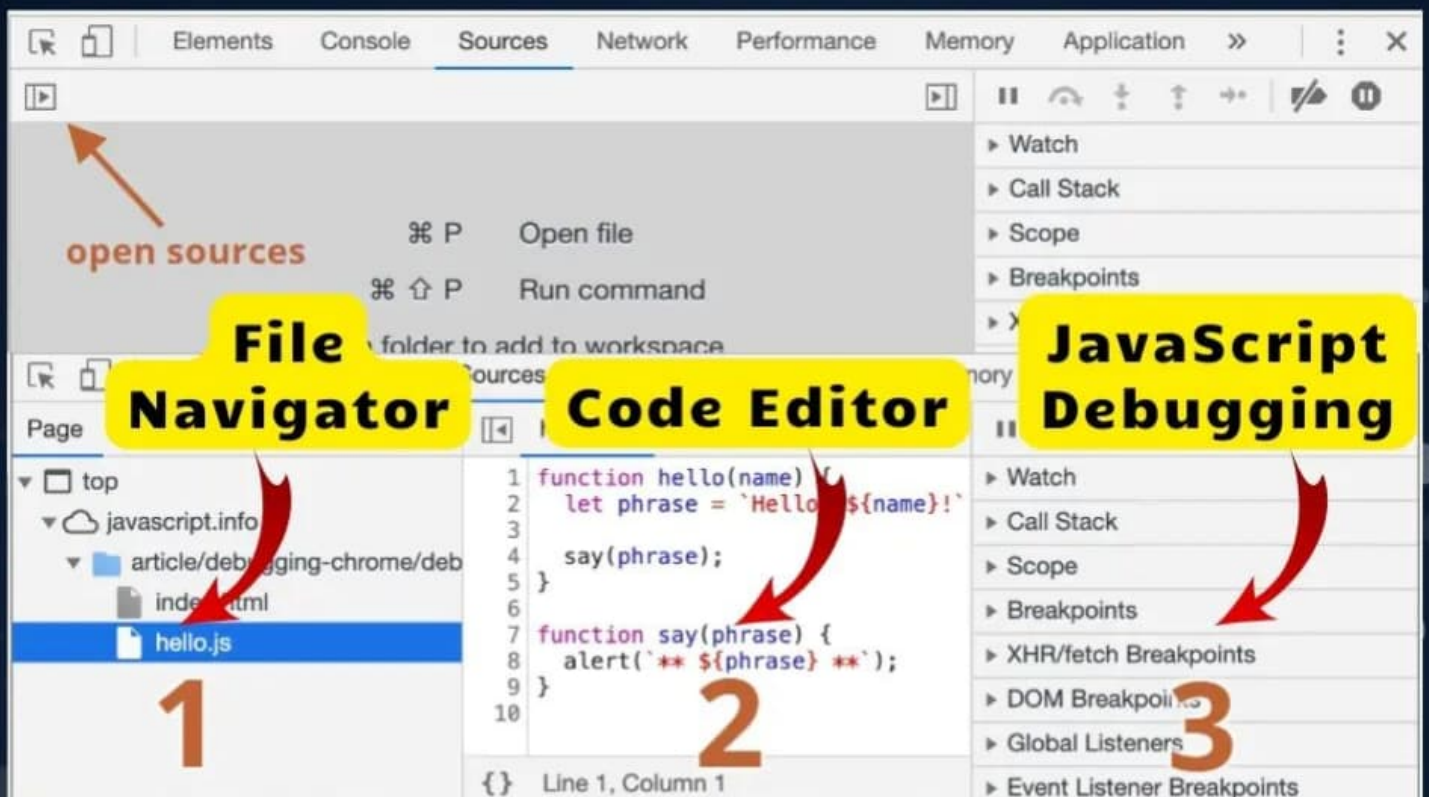


@frontendgenius

# The "Sources" panel

Your Chrome version may look a little bit different, but it still should be obvious what's there.

1. Open the example page in Chrome.
2. Turn on developer tools with F12 (Mac: Cmd+Opt+I).
3. Select the Sources panel.

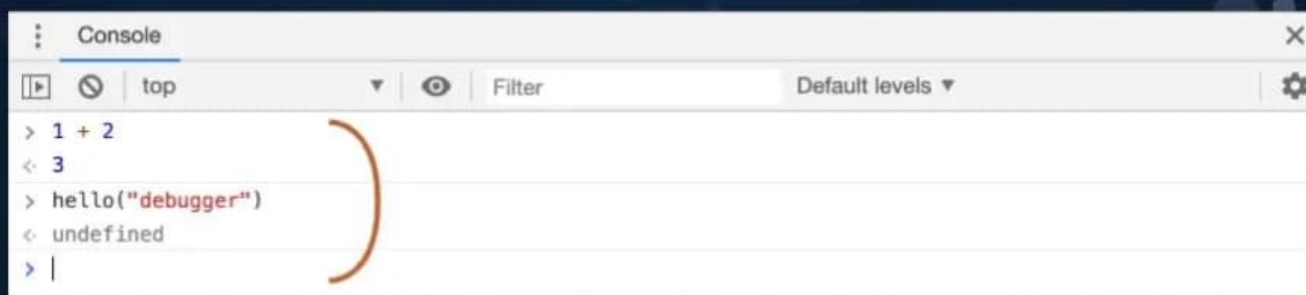


# Console

If we press Esc, then a console opens below. We can type commands there and press Enter to execute.

After a statement is executed, its result is shown below.

For example, here  $1+2$  results in 3, while the function call `hello("debugger")` returns nothing, so the result is undefined:



```
Console
[ ] [ ] top [ ] Filter Default levels [ ]
> 1 + 2
< 3
> hello("debugger")
< undefined
> |
```

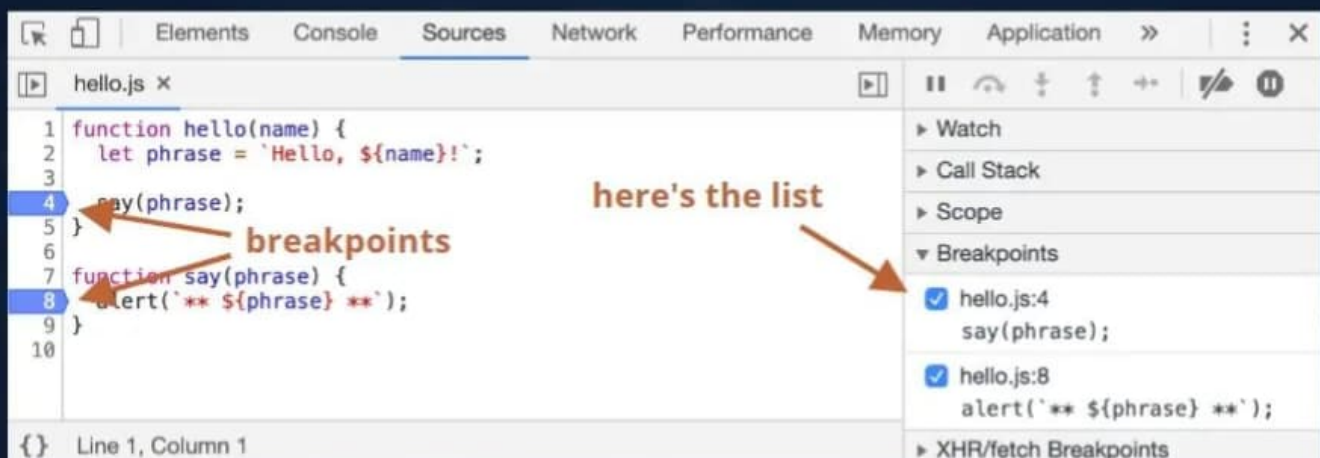


# Breakpoints

In hello.js, click on line number 4. Yes, right on the 4-digit, not on the code.

Congratulations! You've set a breakpoint. Please also click on the number for line 8.

It should look like this (blue is where you should click):





# The command "debugger"

We can also pause the code by using the debugger command in it, like this:

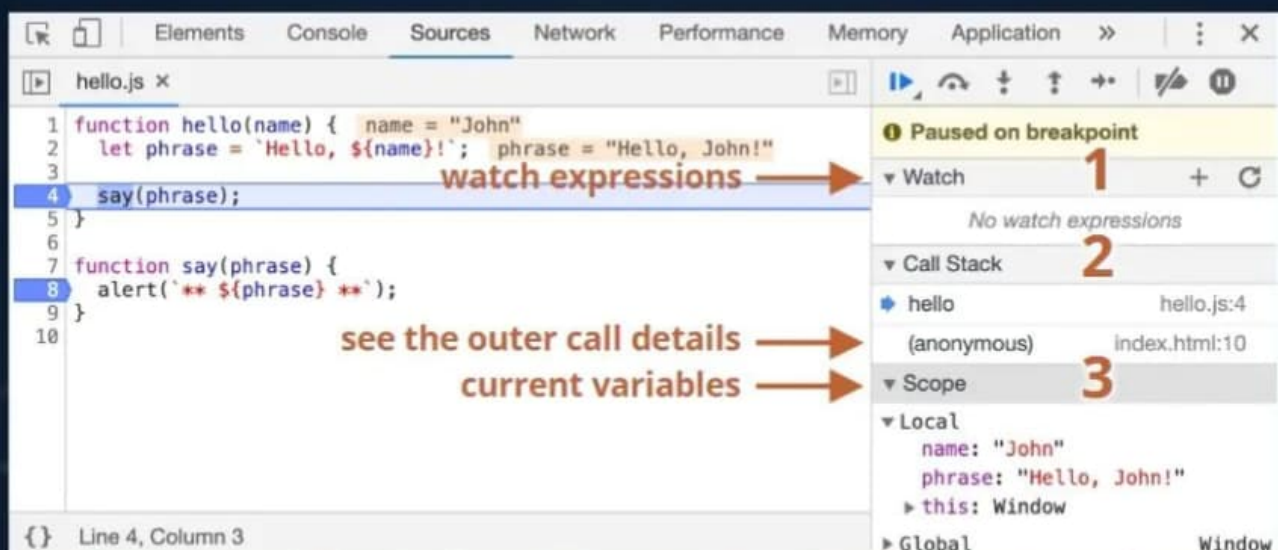
```
1 function hello(name) {  
2   let phrase = `Hello, ${name}!`;  
3  
4   debugger; // <-- the debugger stops here  
5  
6   say(phrase);  
7 }
```



# Pause and look around

In our example, `hello()` is called during the page load, so the easiest way to activate the debugger (after we've set the breakpoints) is to reload the page. So let's press F5 (Windows, Linux) or Cmd+R (Mac).

As the breakpoint is set, the execution pauses at the 4th line:



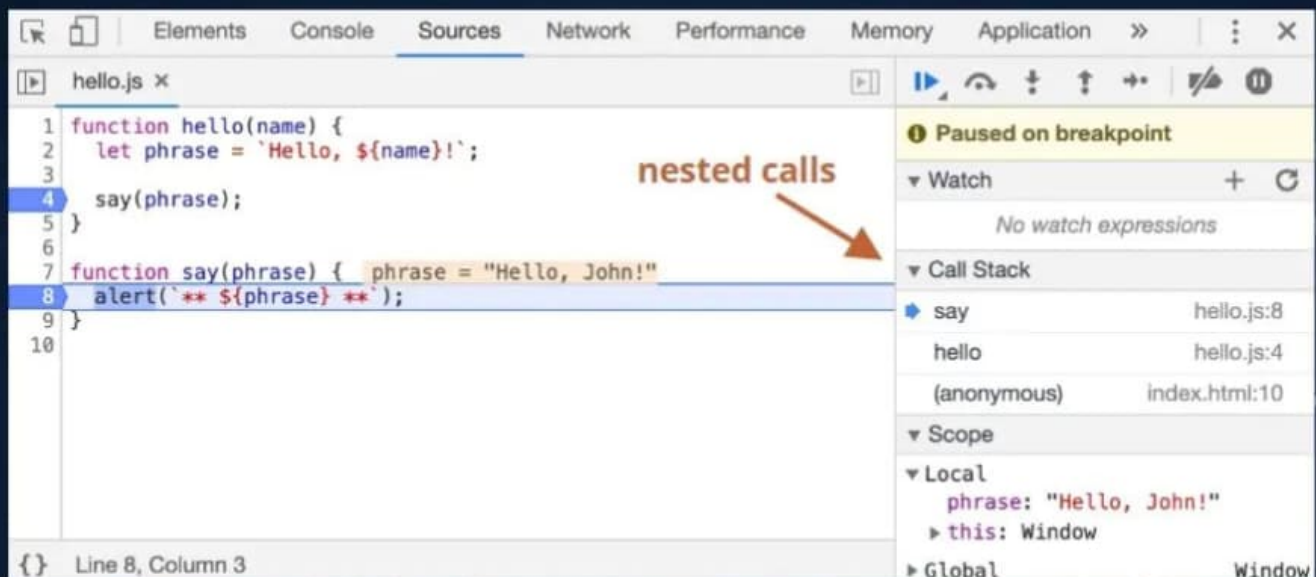


# Tracing the execution



Now it's time to trace the script. There are buttons for it at the top of the right panel. Let's engage them.

- **"Resume": continue the execution, hotkey F8.** Resumes the execution. If there are no additional breakpoints, then the execution just continues and the debugger loses control. Here's what we can see after a click on it:



# Logging

To output something to console from our code, there's `console.log` function. For instance, this outputs values from 0 to 4 to console:

```
1 // open console to see
2 for (let i = 0; i < 5; i++) {
3   console.log("value,", i);
4 }
```

Regular users don't see that output, it is in the console. To see it, either open the Console panel of developer tools or press `Esc` while in another panel: that opens the console at the bottom.

If we have enough logging in our code, then we can see what's going on from the records, without the debugger.