Building Blocks of JavaScript

Dr Harshad Prajapati
14 Nov 2023

What is JavaScript?

- JavaScript is a scripting (interpreted) language.
  - versatile,
  - high-level programming language.
- JavaScript has major role in web development.
  - JavaScript allows developers to add dynamic and interactive elements to websites, enhancing user experience.
  - Earlier, JavaScript was used for only front-end development.
  - Node.JS environment allows to create server side applications also using JavaScript as a programming language.

History of JavaScript?

- JavaScript is a scripting language created by Netscape.
- The original name for JavaScript was LiveScript.
  - The name was changed when Java became popular.
- Similar Script was created by Microsoft called JScript.
- European Computer Manufacturers Association (ECMA) provides standard for scripting languages.
- JavaScript is also called ECMAScript, but browser still refers it as JavaScript.

JavaScript vs Java

JavaScript

- Need browser to run and text editor to build programs.
- Variables are untyped.
- Has objects, but no class (class was added but syntactic sugar).
- Events and event handlers.
- Source code is interpreted.

Java

- Needs JRE to run and JDK to build programs.
- Variables are typed.
- Pure object oriented (objects and class).
- Events and event handlers.
- Source code is translated to byte code, which is run.

JavaScript and EcmaScript

- JavaScript is an implementation of the ECMAScript standard.
- The ECMAScript only defines
  - The syntax/characteristics of the language and
  - A basic set of commonly used objects such as Number, Date, Regular Expression, etc.
- Browsers typically support additional objects such as
  - Window,
  - Frame,
  - Form,
  - DOM,
  - Services.

Key Characteristics of JavaScript

- Statements in JavaScript resemble statements in Java:
  - Because both languages borrowed heavily from the C language.
- JavaScript is platform-independent.
  - Client-side JavaScript executes on the user's browser.
  - Server-side JavaScript executes on Node environment (V8 Engine).
    - V8 JavaScript Engine is open-source, developed by The Chromium Project for the Google Chrome Browser.
- JavaScript is Object Oriented.
- JavaScript is Event driven.
- JavaScript supports asynchronous execution and asynchronous programming.

Use Cases of JavaScript

- JavaScript is a versatile language.
- Web development:
  - Core language for creating dynamic and interactive web pages.
  - Modern frontend frameworks are based on that or support JavaScript.
- Server-side development:
  - Node.js allows using JavaScript for server-side scripting.
- Mobile App Development:
  - JavaScript is used in frameworks like React Native for building cross platform applications.
- Game Development:
  - Used in conjunction with HTML5 for browser-based games.

How to Run JavaScript in a Browser

- JavaScript code is included within <script> tags in HTML document:
- ```
  <script type="text/javascript">
      document.write("<h1>Hello World!</h1>") ;
  </script>
  ```
- Understanding of the code
  - We could use other script, type="text/jscript"
  - Script: javascript is the default.
  - Semicolon is optional. But, needed if we put two or more statements on the same line.
  ```

Example: Using JavaScript

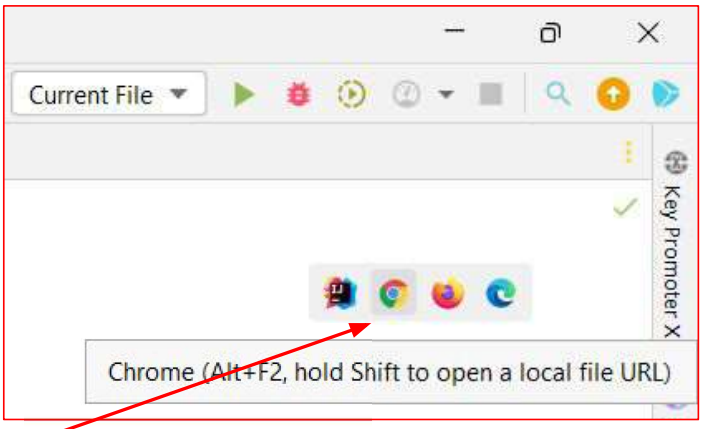
Create HTML File using IntelliJ Idea

javascriptdemo.html

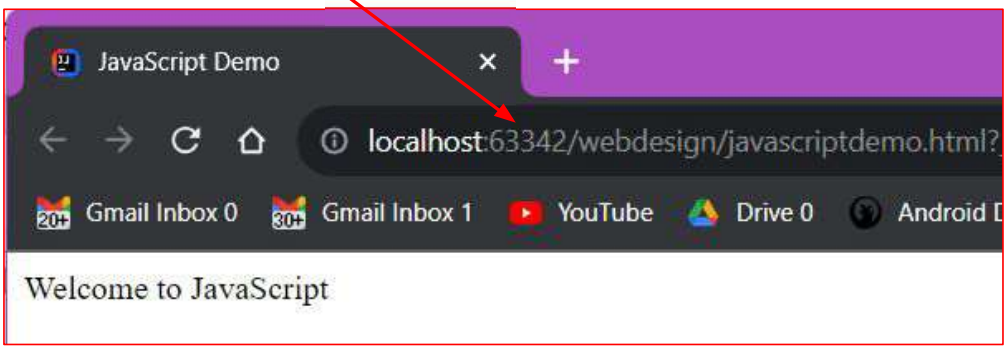
```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <title>JavaScript Demo</title>
6       <script>
7           document.write("Welcome to JavaScript");
8       </script>
9   </head>
10  <body>
11
12  </body>
13  </html>
```

JavaScript Code.
Default language is JavaScript
So type="text/javascript" is not required.

Run File in Browser

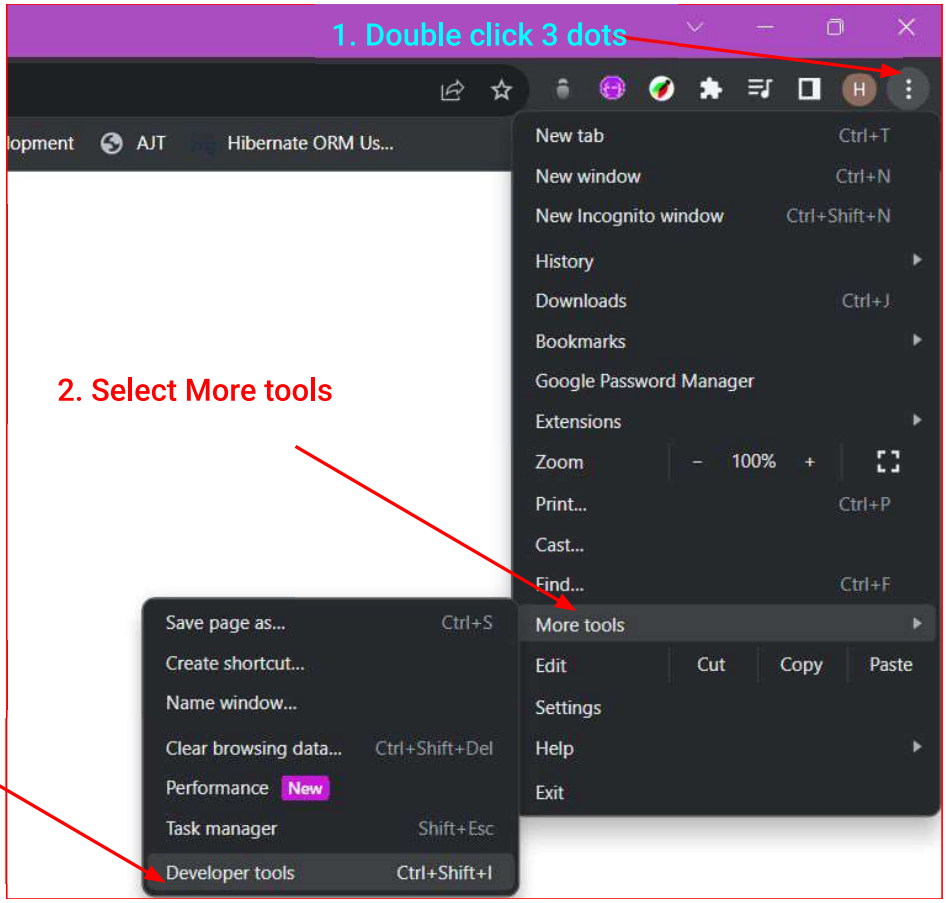


IntelliJ Idea allows to run HTML file directly from the IDE.
The IDE opens a port to serve HTML files.

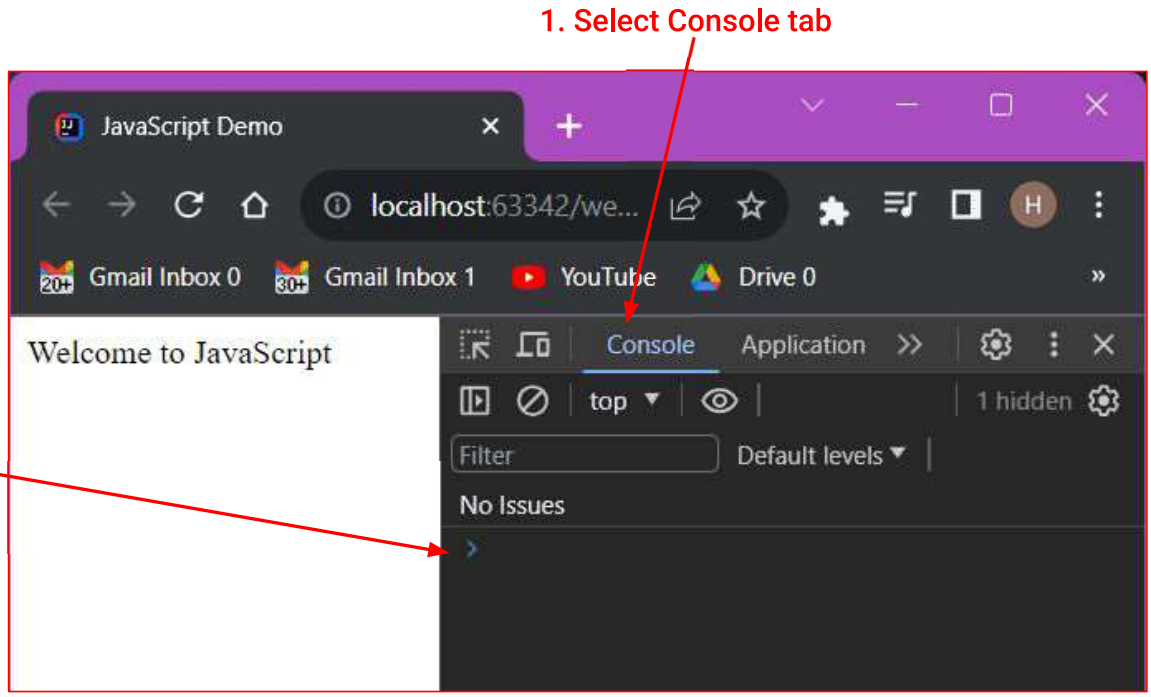


Writing JavaScript Code Directly in Browser

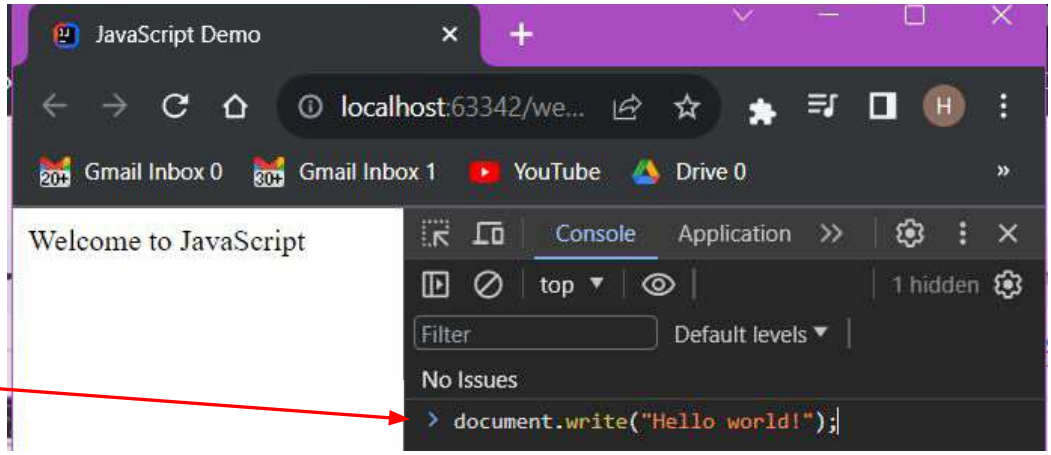
- Open Developer Tools in web browser:
- Two ways:
  - Using menu.
  - Ctrl + Shift + I (shortcut)



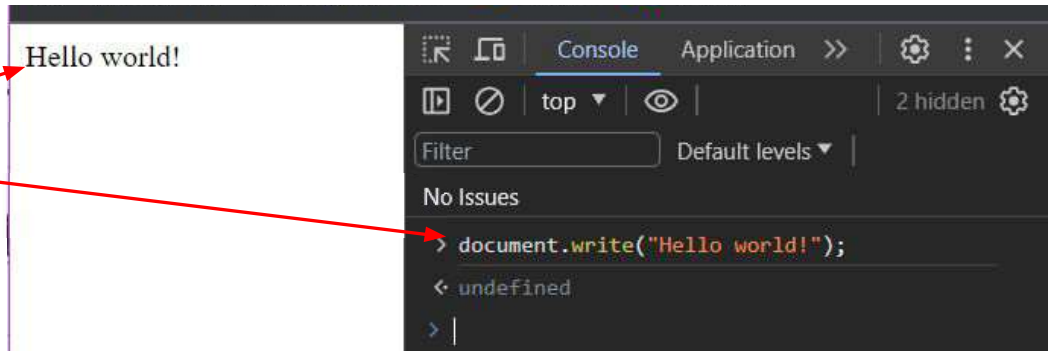
Write JavaScript Code Directly in Browser



1. Write JavaScript statement. Hit ENTER key.



2. Browser executes JavaScript and shows output.



Options to Associate JavaScript Code with Webpage

- We can write JavaScript code inline with form fields.
- We can place JavaScript code in <head> portion.
  - JavaScript functions should be defined in the <head>.
  - This make sure that the function is loaded before it is needed.
- We can place JavaScript code anywhere inside <body>.
- We can place JavaScript code in a separate .js file.
  - In HTML file, we write the following to use JavaScript available in a separate file:

    <script src="myjs.js"></script>
  - The .js file (myjs.js) does not require to include <script> element again.

JavaScript Language

- JavaScript is dynamically typed language.
- It's syntax is similar to Java language.
- JavaScript supports:
  - variables, arrays, objects.
  - control structures (if else, switch)
  - loop constructs (for while, for in, for of)
  - error handling using try catch.
  - inbuilt objects.
  - inbuilt functions.

Variables in JavaScript Language

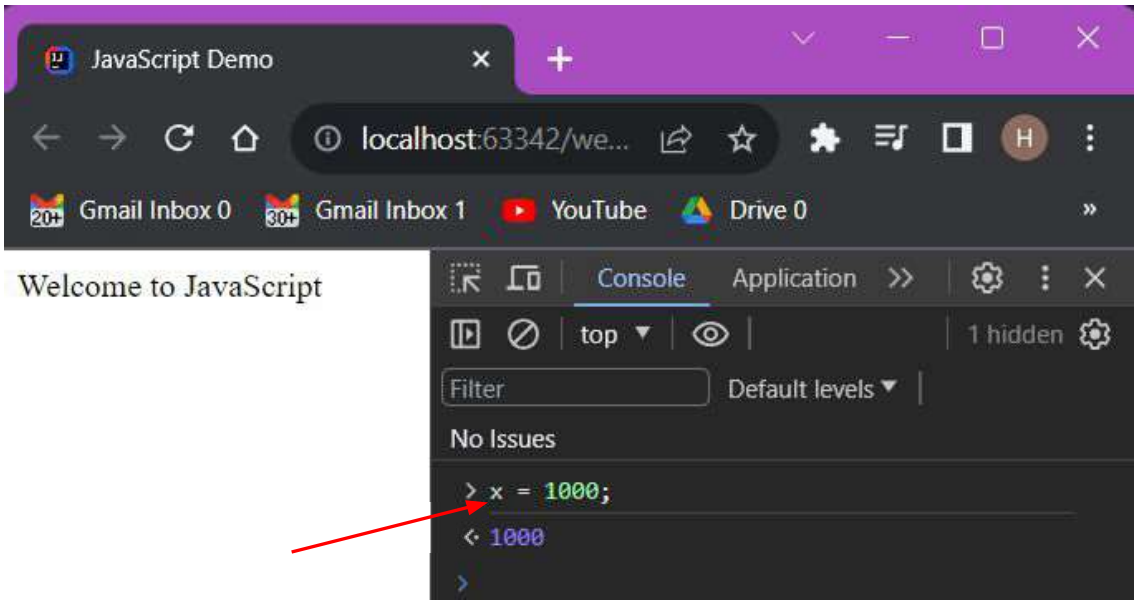
- JavaScript is dynamically typed language.
  - var n = "JavaScript";
    n = 1.5;
    n = 1;
- The word var is optional.
- Variables are not typed (they can hold values of any data type)
- Variable names are case sensitive.
- Variables names must begin with a letter or underscore.

Global Variables vs Local Variables in JavaScript

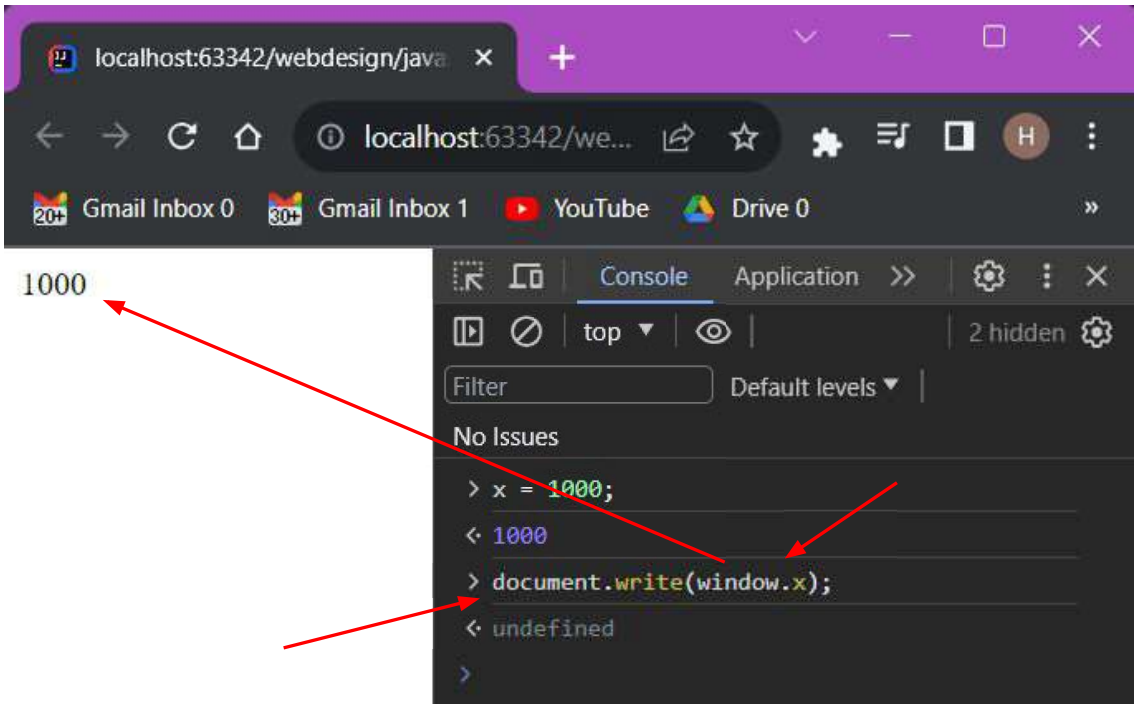
- Local Variables:
  - Variables declared within a function are local to that function
  - Local variables are accessible only within that function.
- Global variables:
  - Variables declared outside a function are global.
  - Global variables are accessible from anywhere on the page.
    - To access say variable x, we can write window.x.

Example: Variable

Declaring Global Variable



Using Global Variable



Datatypes in JavaScript

- Primitive data types:
  - Number: integer and floating-point numbers.
  - Boolean: true or false.
  - String: a sequence of characters.
- Composite data types (or Complex data types)
  - Object: a named collection of data.
  - Array: a sequence of values (an array is actually a predefined object)
- Special data types:
  - null: the only value is null – to represent nothing.
  - undefined: the only value is undefined – to represent the value of an uninitialized variable.

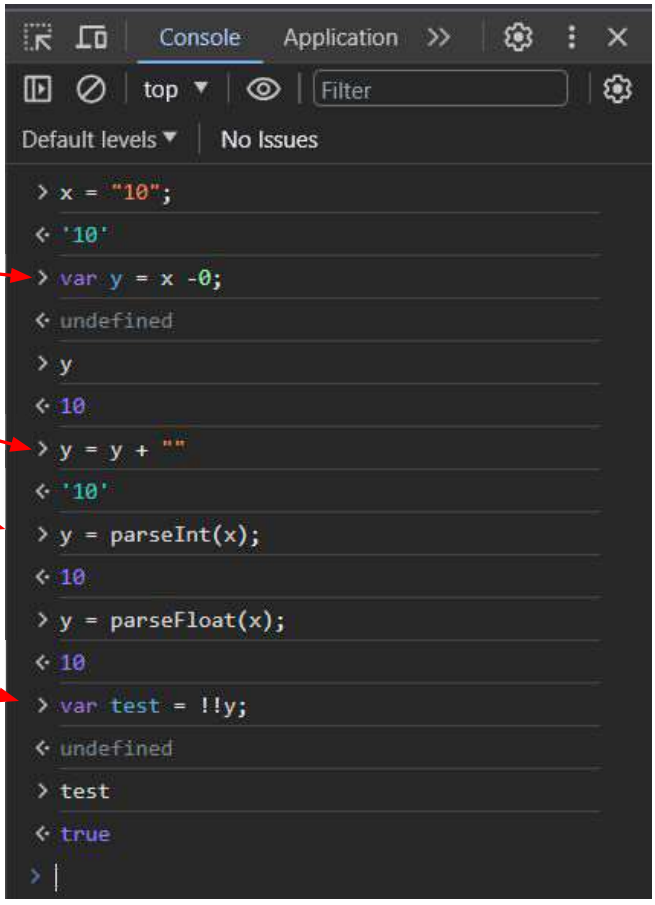
Boolean

- Booleans are either true or false.
- 0, "0", empty strings, undefined, null, and NaN are false.
- All other values are true.

Example: Datatype and Type Conversion

Type Conversion

- Converting a value to a number:
  - var numberVar = someVariable – 0;
- Converting a value to a string:
  - var stringVar = someVariable + "";
- Converting a string to a number:
  - parseInt("123");
    OR parseFloat("123.45");
- Converting a value to a boolean:
  - var boolVar = !!someVariable;



Important Operators

- The conditional operator (?:):
  - condition ? value_if_true : value_if_false
- Special equality test:
  - == and != try to convert their operands present on both the sides to the same type before performing the test.
  - === and !== do not convert operands to the same type.

Example: == and === Operators

Important Operators: == vs ===

- Type conversion is performed before comparison in use of == comparison.
  - var a = ("5" == 5); // true
- No implicit type conversion.
  - var b = ("5" === 5); // false
- var c = (5 === 5.0); // true
- var d = (true == 1); // true
  - (true is converted to 1)
- var e = (true == 2); // false
  - (true is converted to 1)
- var f = (true == "1") // true

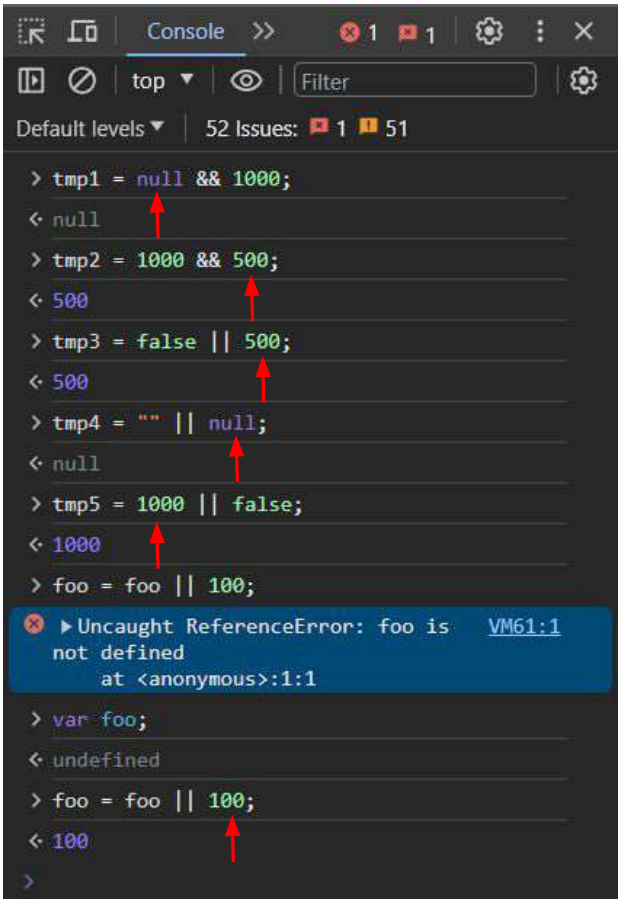
Important Operators: && and ||

- Important information: The && and || operators are heavily used in React while conditionally rendering.
- Usage of (firstThing && secondThing)
  - If the first thing is true then only perform the second thing.
  - Example, If API response has come (first thing), then render the response (second thing)
- Usage of (firstThing || secondThing)
  - If the first thing is false then only perform the second thing.
  - Example, if an API URL is not initialized, then initialize API URL.

Example: && and || Operators

Important Operators: && and ||

- tmp1 = null && 1000; // tmp1 is null
- tmp2 = 1000 && 500; // tmp2 is 500
- tmp3 = false || 500; // tmp3 is 500
- tmp4 = "" || null; // tmp4 is null
- tmp5 = 1000 || false; // tmp5 is 1000
- var foo;
  foo = foo || 100;
  // If foo is null, undefined, false, zero, NaN,
  // or an empty string are falsy values.



Example: typeof Operator

typeof Operator

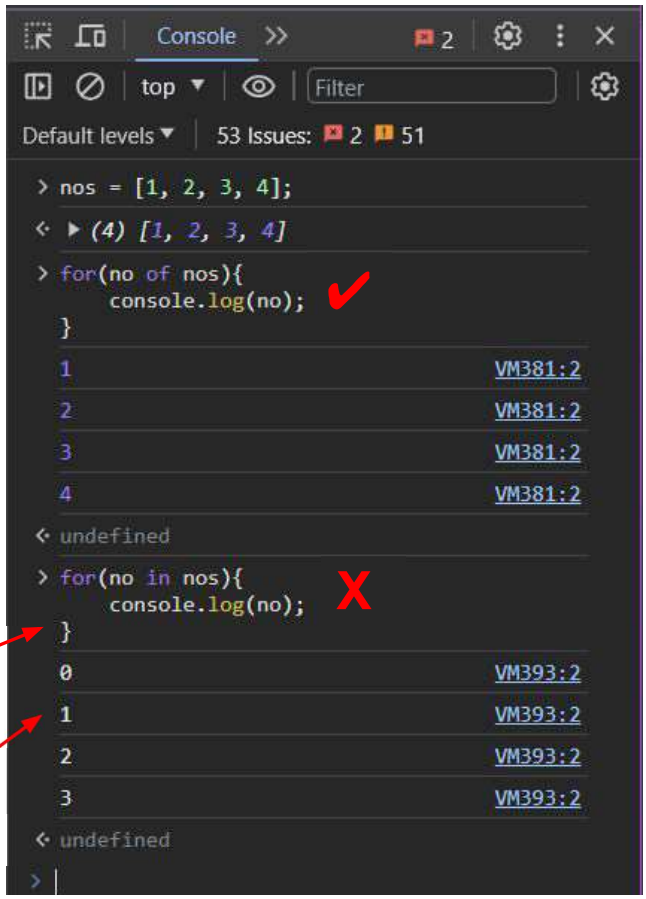
- The typeof operator (unary) tells the type of its operand.
  - Returns a string which can be number, string, boolean, object, function, undefined, and null.
- var x = "hello", y;
- typeof x;
- typeof y;
- typeof z;
- var a = [];
  - An array is internally stored as an object.

Example: Loop Constructs

Important Loop Constructs

- for … of loop (elements of array)
  - The for...of loop is used to iterate over iterable objects such as arrays, strings, maps, sets, etc.
- for … in loop (members in object)
  - The for...in loop is used to iterate over the enumerable properties of an object.
- Be careful to use the right loop construct when working with array.
  - The for…in provides keys.

These are keys (indexes) of the array



Important Loop Constructs

- While working with object, we can use only for … in loop construct.
- Object is not iterable, so for … of loop construct cannot be used.

Functions

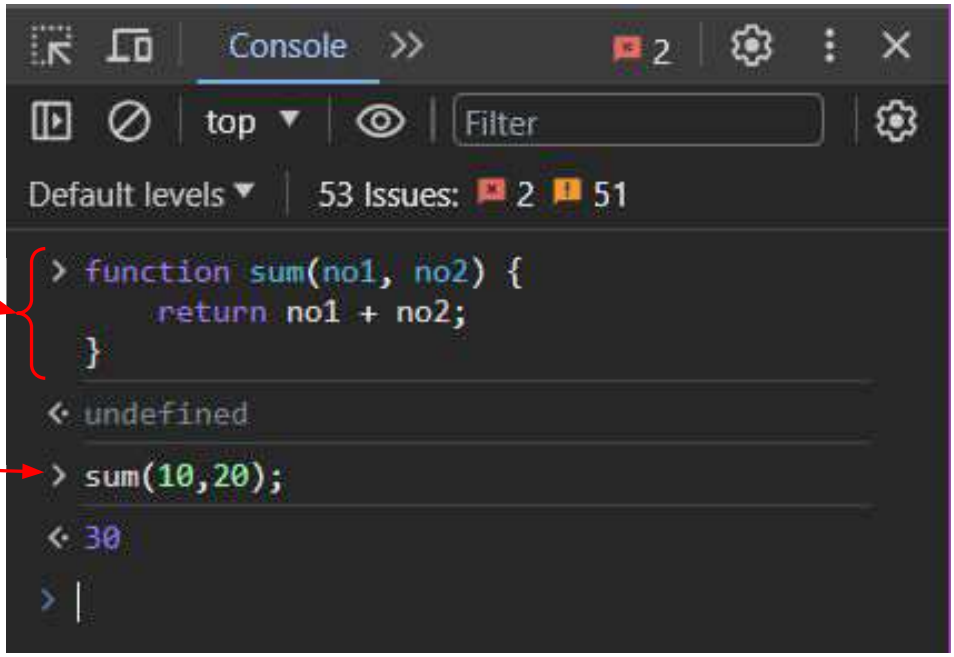
- Functions should be defined in the <head> of an HTML page, to ensure that they are loaded first.
- The syntax for defining a function is:
  function functionName(arg1, …, argN) { statements }
- The function may contain return value statements.
- Any variables declared within the function are local to it.
- The syntax for calling a function is just
  functionName(arg1, …, argN);
- Simple parameters are passed by value, objects are passed by reference.

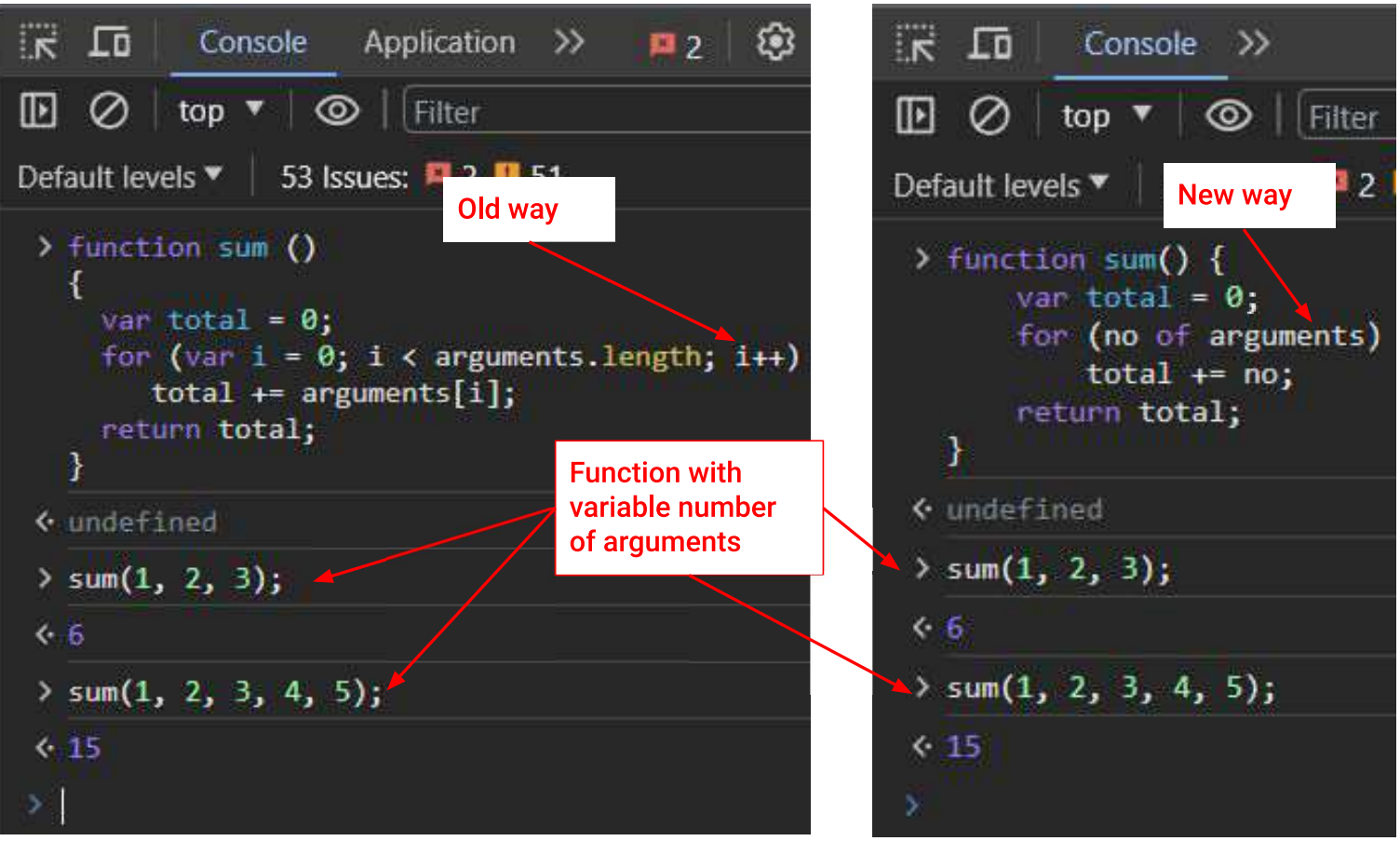
Example: Functions

Function with Fixed Number of Arguments

Define a function

Call a function

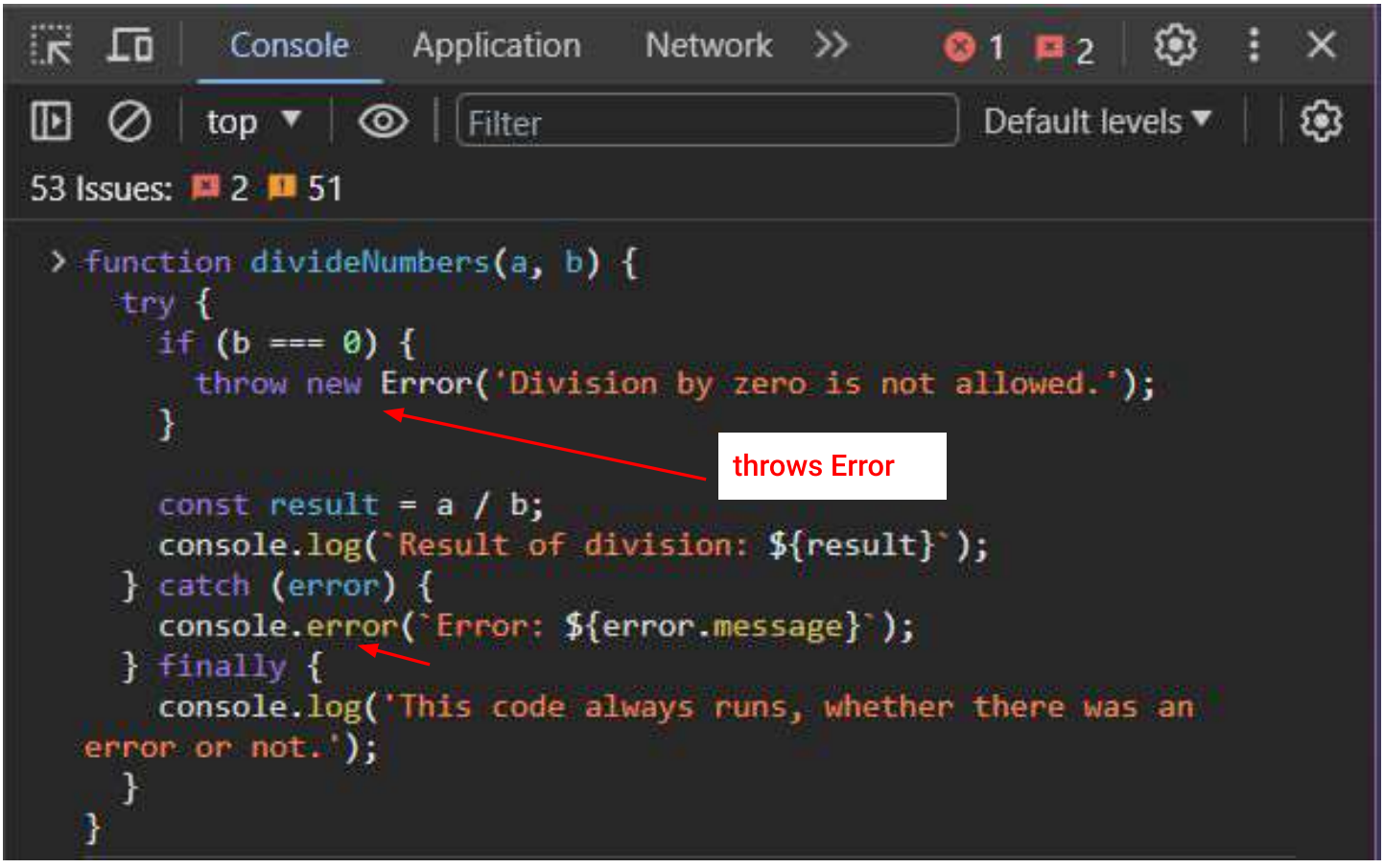




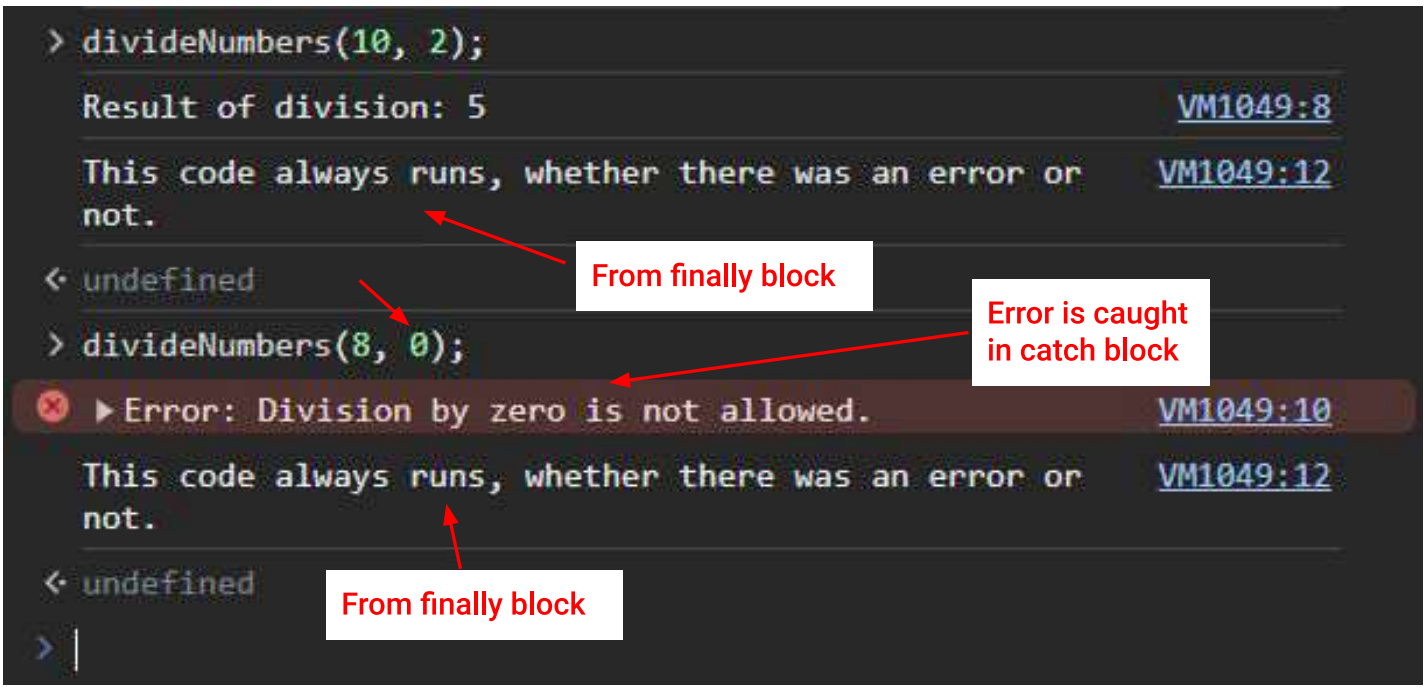
try catch finally for Exception Handling

- Exception handling in JavaScript is almost the same as in Java.
- The throw expression creates and throws an exception.
- try {
  // statements to try
  } catch (e) { // Notice: no type declaration for e
  // exception handling statements
  } finally { // optional, as usual
  // code that is always executed
  }

Example: try-catch-finally



Testing Exception Handling



The this keyword

- The this is a keyword and not a variable, so its value cannot be changed.
- In JavaScript, this refers to an object.
- To which object, this refers depends on how it is used.
  - In an object method, this refers to the object.
  - Alone, this refers to the global object.
  - In a function, this refers to the global object.
  - In a function, in strict mode, this is undefined.
  - In an event, this refers to the element that received the event.
  - Methods like call(), apply(), and bind() can refer this to any object.

References

- https://developer.mozilla.org/en-US/docs/Web/JavaScript