Coding Bootcamp

JavaScript

TECHNOLOGY

Getting Started with JavaScript

simplilearn

Learning Objectives

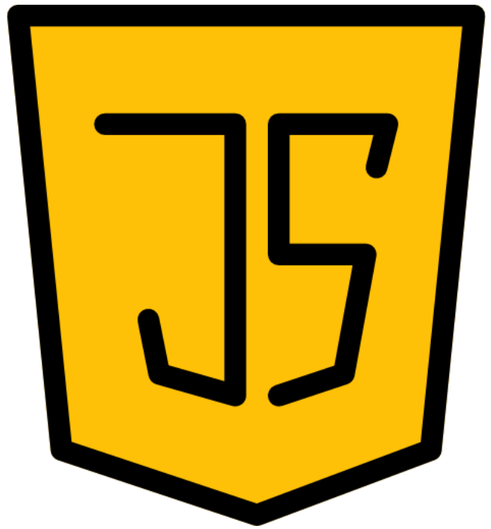
By the end of this lesson, you will be able to:

- Outline the basics of JavaScript for understanding its core concepts
- Identify and categorize the primitive types and write their syntax for accurate data manipulation
- Define objects in JavaScript for organizing complex data
- Demonstrate how arrays are used in JavaScript for managing collections of data
- List the methods used in arrays for efficient data operations

What Is JavaScript?

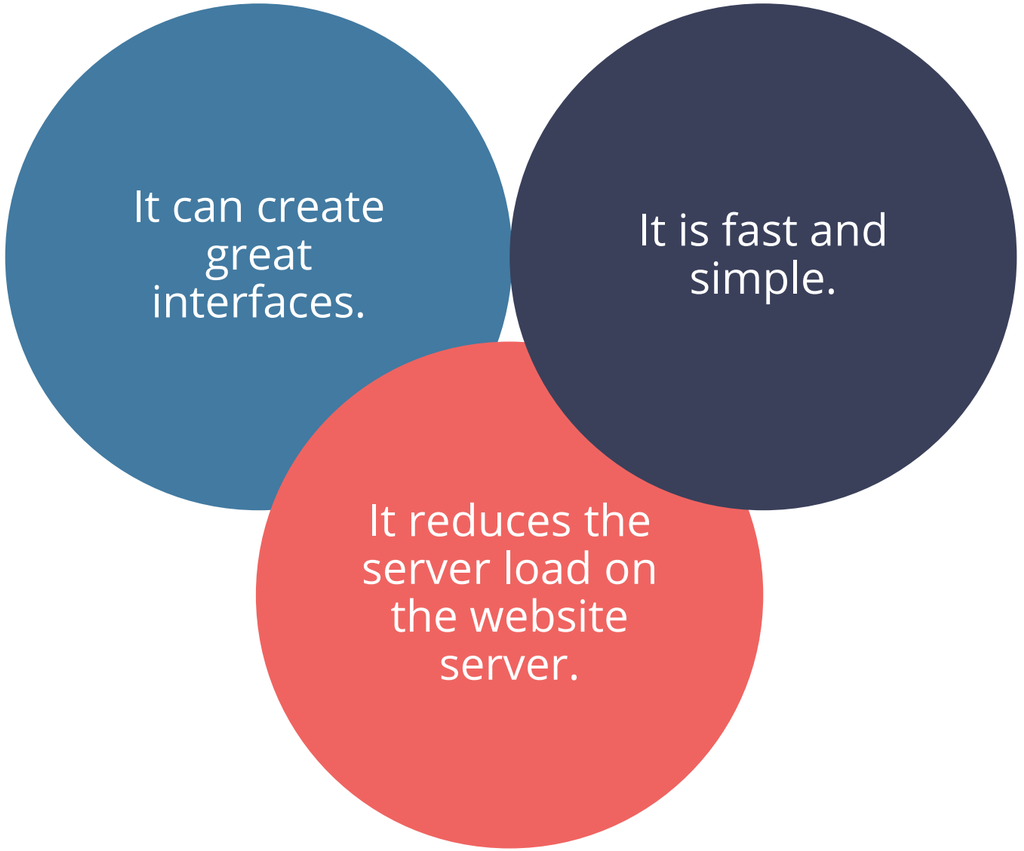
What Is JavaScript?

JavaScript is a lightweight, text-based programming language.



- It is employed on the client side as well as the server side.
- It enables developers to make web pages more interactive.
- It can be used to add interactive elements to website operations.

Advantages of JavaScript

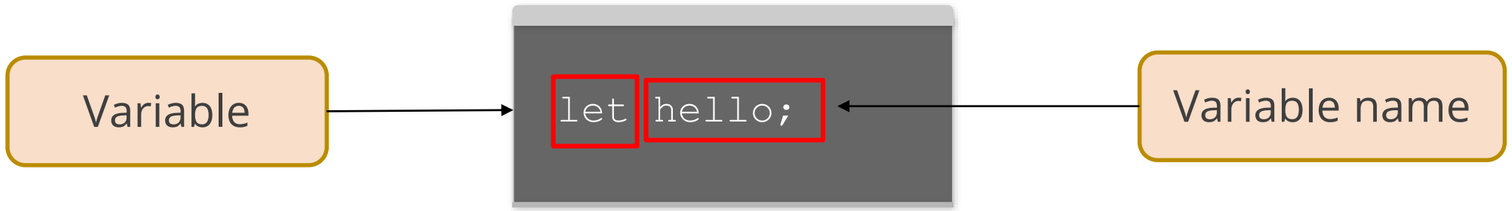


Variables

Variables

In JavaScript, a variable stores the data value that can be changed later.

Syntax:



Data in the variable hello is within the assignment operator(=).

```
let hello;
hello = ' how are you! ';
```

Variables

The string is saved in the variable's related region.

```
let hello;
hello = ` how are you! `;
alert (hello) ;
```

The variable declaration and assignment can be combined into a single line.

```
let hello = ` how are you! `;
alert (hello) ;
```

Variables

The value is changed in the variable.

```
let hello;
hello = ' how are you! ';
hello = 'I am fine' ;
alert (hello) ;
```

In older scripts, the **var** variable is used instead of **let**.

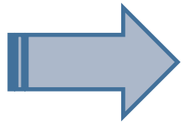
The var key term is identical to the let key term.

```
var hello = ' how are you! ';
```

Variables

Example:

```
<! DOCTYPE html>
<html>
<body>
  <script>
      let hello;
      hello = “Hello User!! Welcome to XYZ Page”;
Alert(hello);
</script>
</body>
</html>
```



Output:

```
127.0.0.1:5500 says
Hello User!! Welcome to
XYZ Page
```

Constants

Constants

In JavaScript, if users never want a variable to change, they can use the **const** keyword instead of the **let** keyword.

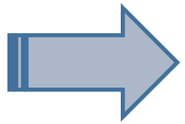
```
const avgHeight = "Average Height of Men is 177cm" ;
```

The variable is declared using the **const**, also called constants.

Constants

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      const avgHeight = "Average Height of Men is
177cm";
alert(avgHeight);
  </script>
</body>
</html>
```



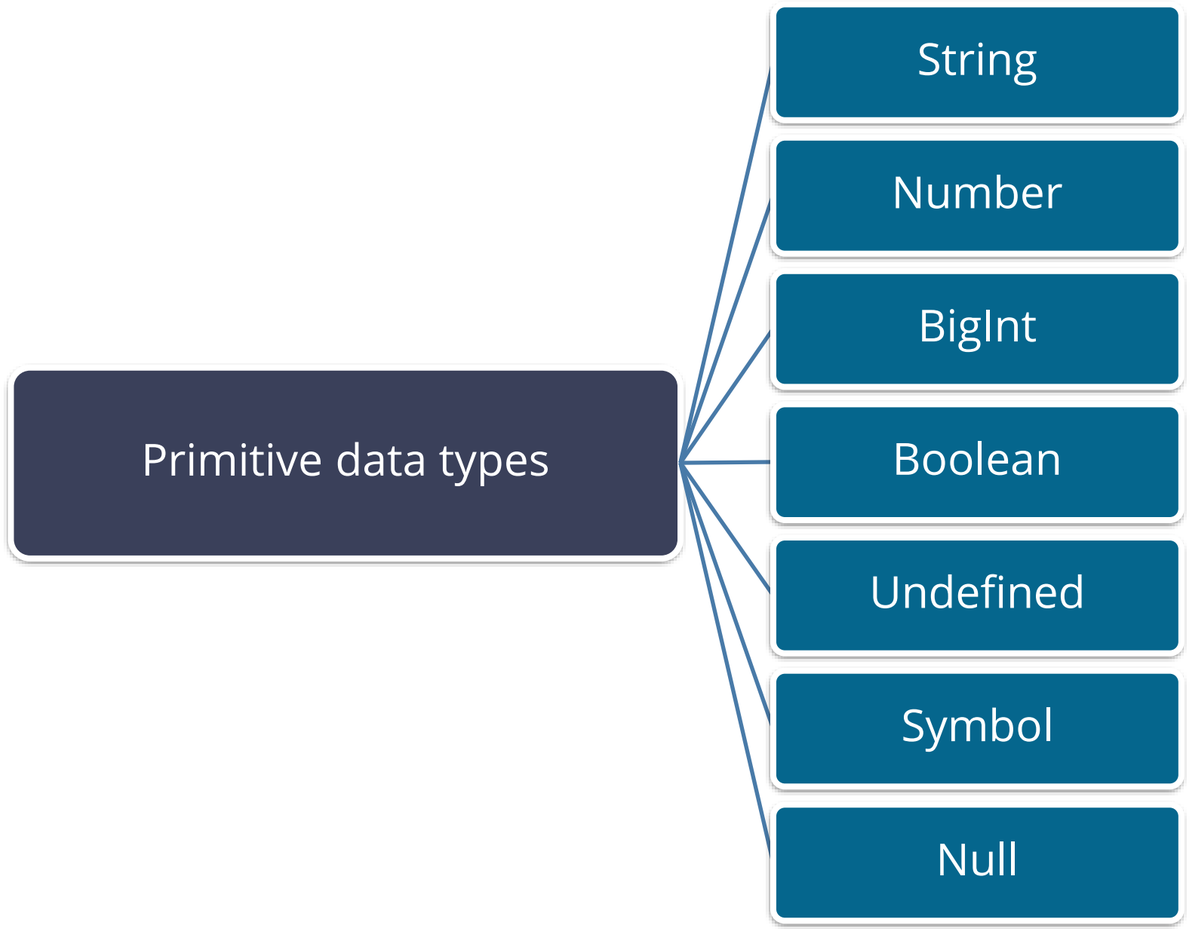
Output:

```
127.0.0.1:5500 says
Average Men Height is
177cm
```

Primitive Data Types

Primitive Data Types

In JavaScript, a primitive is data that is not an object and has no methods.



String

It is a series of characters in JavaScript.

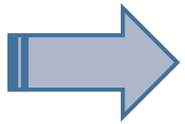
```
let std1Name = "Jack";
let std2Name = "Dave";
```

The string is written in double or single quotes.

String

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let std1Name = “Jack”;
      alert(std1Name);
  </script>
</body>
</html>
```



Output:

```
127.0.0.1:5500 says
Jack
```

Number

JavaScript has only one type of number. It contains only the numeric values.

The operations applied to numbers are:

Addition

Subtraction

Multiplication

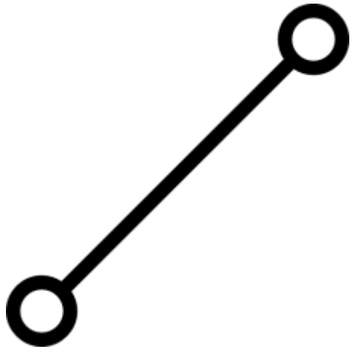
Division

Number

The number data types include special numeric values in addition to conventional numbers.

Infinity

It is obtained by dividing any number by zero or just the reference directly.
Example: alert (1/0)



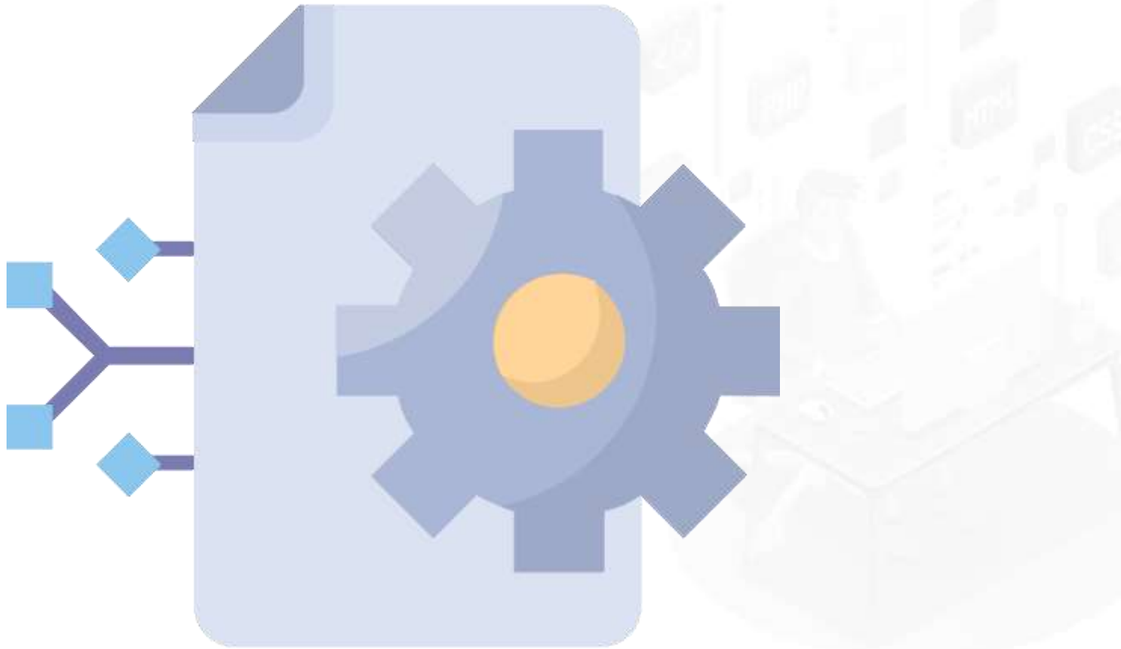
NaN

It represents the computational error.
Example: alert (not a number/2)

Number

The BigInt type represents integers of any length.

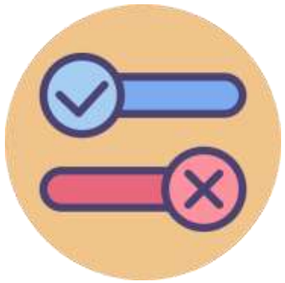
```
const bigInt = 3467898918230982109380109n;
```



Boolean

The boolean type in JavaScript has only two values: true or false.

```
let termsAndConditions = true;
let subscriptionForNewsLetter = false;
```

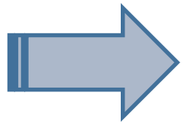


In this, one field is checked and the other is not. It gives the result in a true or false format.

Boolean

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let age = 19
      let isEligibleForDriving = age >= 18;
      alert(isEligibleForDriving);
  </script>
</body>
</html>
```



Output:

```
127.0.0.1:5500 says
True
```

Undefined

It appears when there is no value assigned to a variable.

```
let age;
Alert(age);
```

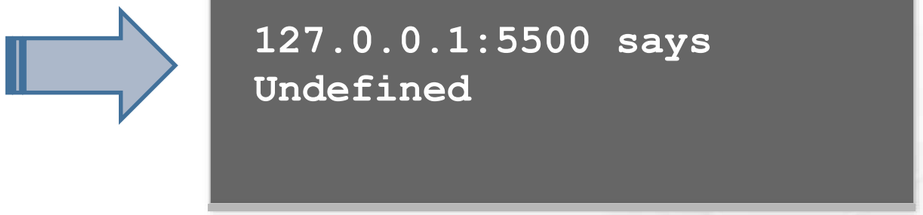
Although the variable is declared, no value is assigned to it.

Undefined

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let age;
      alert(age);
  </script>
</body>
</html>
```

Output:



Symbol

This data type defines a property that is private to the object.

To create a new symbol, the function **Symbol()** is used.

Note that every symbol is unique. Two symbols, even with the same key values, are not the same.

Null

This data creates its own separate type, which contains only the null values in it.

```
Let name = null;
```

Null basically is a unique value that means **nothing**, **empty**, or **nothing at all**.

Dynamic Typing

Dynamic Typing

In JavaScript, variables are not bound to a specific data type. A variable can hold any type of value and can change its type at runtime.

```
var s = 10;
```

```
S = "Hello, how are you"
```

S is changed into the string.

Objects

Objects

An object is a collection of key-value pairs where the keys are strings (or symbols) and the values can be of any type, including other objects, arrays, and functions.

Example

```
let car = new Object();
let car = { };
```


They are used to store and manage data in a structured way, enabling easy access and manipulation of related data.

Objects: Property

A property is a **key: value** pair in which the key is a string, and the value is any value.

```
let user = {
  name: "JACK",
  age: 24
}
```

Arrays

Arrays

An array in JavaScript is a type of variable that can hold more than one value in it.

There are two syntaxes for creating arrays:

```
let arr = new Array ();
let arr = [];
```

```
let colors = [ "green", "yellow", "blue", "white" "gray", "red"
];
```

Arrays

Alert function is used to display the array values.

```
alert( colors[0] );
alert( colors[1] );
alert( colors[2] );
alert( colors[3] );
alert( colors[4] );
alert( colors[5] );
```



```
color[1] = “black“;
```

Arrays

The element can be added by adding an index. The new element and its value will be added to the array at the seventh position.

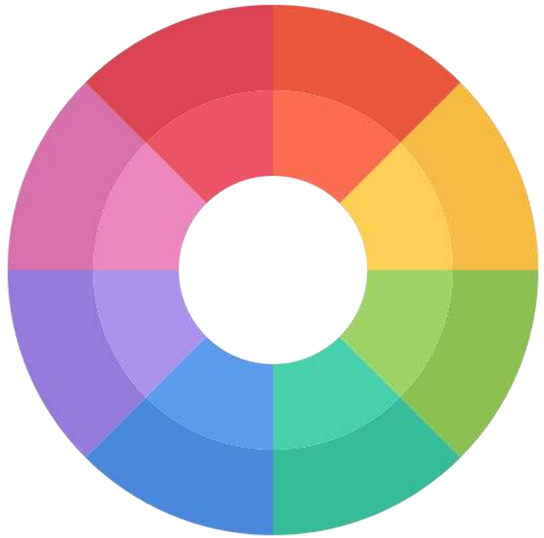
```
color[6] = “purple“;
```

To get the total count of an array index, **length** is used.

```
let colors = [ "green", "yellow", "blue", "white" "gray", "red"
];
Alert ( colors.length );
```

Arrays

Add the name of the array within the alert to display the whole array

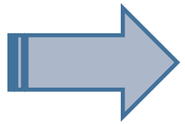


```
let colors = [ "green", "yellow", "blue", "white" "gray", "red"
];
alert (colors);
```

Arrays

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let colors = [
“green”,
“yellow”,
“blue”,
“white”,
“gray”,
“red”
      ]
      alert(colors);
  </script>
</body>
</html>
```



Output:

```
127.0.0.1:5500 says
green,yellow,blue,white,gr
ay,red
```

TECHNOLOGY

Methods in Arrays

Methods in Arrays

There are four common methods in an array:

- Push
- POP
- Shift
- Unshift

Push

It appends the part to the end of the array and adds an element at the end of the array.

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let colors = [
“green”,
“yellow”,
“blue”,
“white”,
“gray”,
“red”
      ]
      colors.push(“violet”);
      alert(colors);
  </script>
</body>
</html>
```

Output:

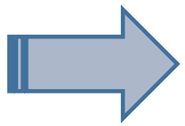
```
127.0.0.1:5500 says
green,yellow,blue,white,
gray,red,violet
```

POP

It takes an element from the end.

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let colors = [
"green",
"yellow",
"blue",
"white",
"gray",
"red"
      ]
      colors.pop();
      alert(colors);
  </script>
</body>
</html>
```



Output:

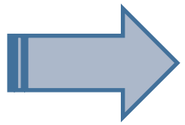
```
127.0.0.1:5500 says
green,yellow,blue,white,
gray,red
```

Shift

In this method, the first element of the array is extracted and returned in this procedure.

Example:

```
<!DOCTYPE html>
<html>
<body>
  <script>
      let colors = [ “red”, “yellow”, “green”];
      alert ( colors.shift () );
      alert(colors);
  </script>
</body>
</html>
```



Output:

```
This page says
red
```

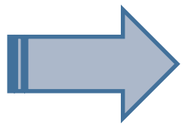
Unshift Method

It helps to add the element at the beginning of the array.

Example:

```
<!DOCTYPE html>
<html>
  <script>
      let colors = [ “red”, “yellow” ];
      colors.unshift(“green”);
      alert(colors);
  </script>
</html>
```

Output:



TECHNOLOGY

Functions

Functions

Functions are the program's main building blocks. The following are the aspects of a function:

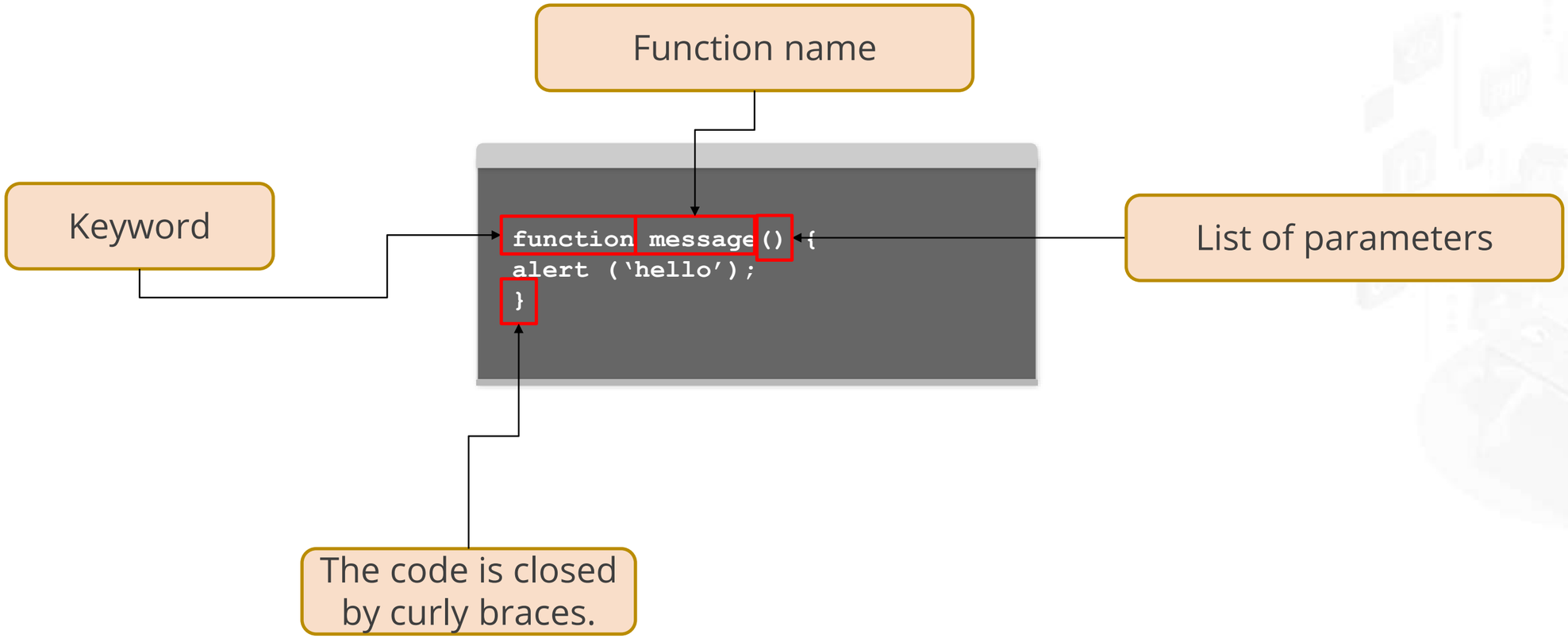
Input

Output

In JavaScript, a function is a procedure with a set of statements that performs a task or calculates a value.

Functions

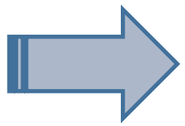
Before calling a function, you need to declare and define it.



Functions

Example:

```
<!DOCTYPE html>
<html>
  <script>
      function message()[
      alert("How are you");
      ]
      message();
  </script>
</html>
```



Output:

```
This page says
How are you
```

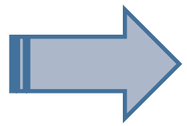
Local Variables

In a local variable, only the variables declared within the function are accessible by the function.

Example:

```
<!DOCTYPE html>
<html>
  <script>
      function message () {
      let msg = [ "Hii, how are you !" ];
      alert (msg);
      }
      message();
      alert(msg);
  </script>
</html>
```

Output:



```
This page says
Hii, how are you!
```

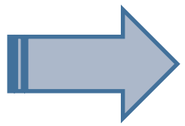
Outer Variables

In the outer variable, the variable declared outside the function is also accessible by the function.

Example:

```
<!DOCTYPE html>
<html>
  <script>
      let user = "Reet";
      function message () {
      let msg = [ 'Hello,' + user;
      alert (msg);
      }
      message();
</script>
</html>
```

Output:



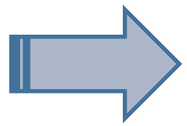
```
This page says
Hello, Reet
```

Parameters

They are used to pass the arbitrary data to the function.

Example:

```
<!DOCTYPE html>
<html>
  <script>
      function message (from, text) {
      alert (from + ':' +text);
      }
      message('John', 'hello');
</script>
</html>
```



Output:

```
This page says
John:hello
```

Writing and Executing JavaScript Program Using Node

Problem Statement:

You have been asked to demonstrate the process of writing and executing a JavaScript program using Node.js.

Outcome:

By completing this task, you demonstrated the process of writing and executing a JavaScript program using Node.js. You began by creating a JavaScript directory and then executed the JavaScript program within that directory using Node.js.

Note: Refer to the demo document for detailed steps:
01_Writing_and_Executing_JavaScript_Program_Using_Node

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed are:

1. Create a JavaScript directory
2. Execute the JavaScript program using Node.js

Key Takeaways

- JavaScript is a lightweight, text-based programming language.
- In JavaScript, a variable stores the data value that can be changed later.
- In JavaScript, if users never want a variable to change, they can use the **const** keyword instead of the **let** keyword.
- The type of variable used for declaration is not specified in JavaScript.
- A function is a procedure with a set of statements that performs a task or calculates a value.

TECHNOLOGY

Thank You

simplilearn