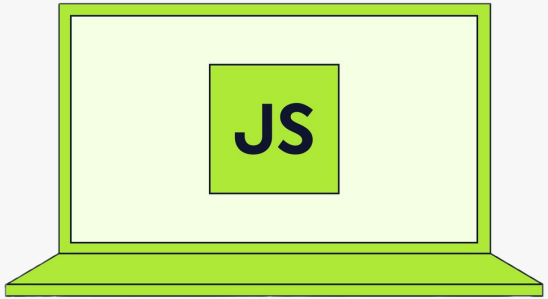
Newton School of Technology

The Complete Javascript Course



@newtonschool

Lecture 19: Basics of Javascript

-Rahul Kumar

JS

Table of Contents

Newton School of Technology

Let’s Start

What is Javascript?

Newton School of Technology

Animate elements and make pages dynamic.

JAVASCRIPT

The language that makes web pages come alive. 🤯

Run back-end code with Node.js.

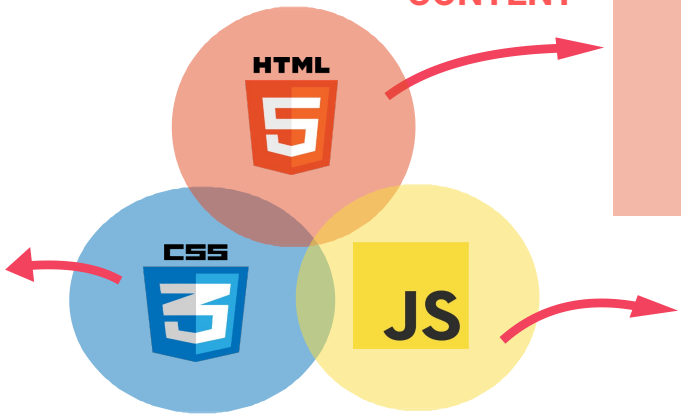
Create web, mobile, or desktop apps.

Fetch and process data dynamically.

Javascript Role in Web Development

Newton School of Technology

Javascript adds logic to our web pages:-



CONTENT

NOUNS

```
<p>Data</p>
```

means "***paragraph***"

ADJECTIVES

```
p {color: red}
```

means "the paragraph text is ***red***"

VERBS

```
p.hide();
```

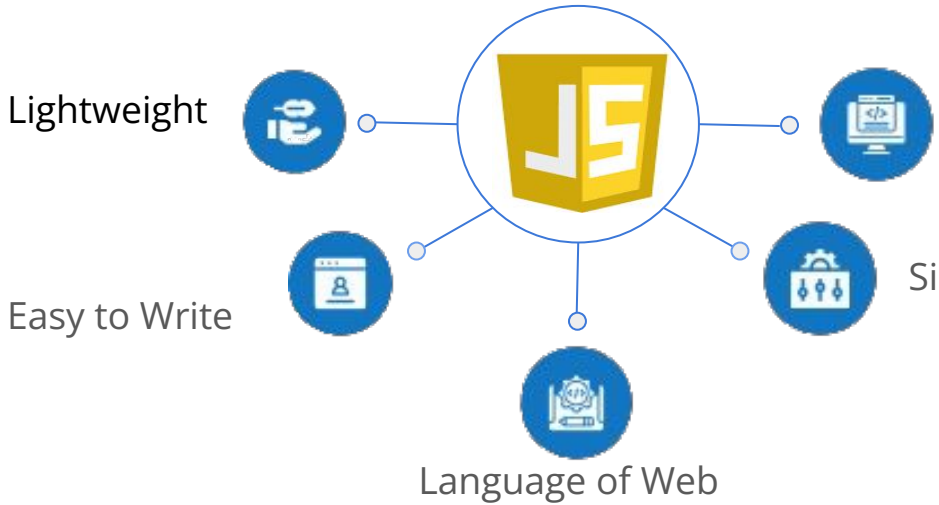
means "***hide*** the paragraph"

PRESENTATION

PROGRAMMING LANGUAGE: BUILDING WEB APPLICATIONS

Features of Javascript

JavaScript is a programming language that makes websites interactive, like adding buttons, animations, and live updates.



JavaScript is single-threaded, meaning it processes one task at a time in a sequential order.

Comparing with other languages

Newton School of Technology

Let’s compare javascript with other languages:-

	JavaScript	C#	Java	Ruby	Python
Front-End	✓	✗	✗	✗	✗
Back-End	✓	✓	✓	✓	✓
Mobile Apps	✓	✓	✓	✓	✓
Desktop Apps	✓	✓	✓	✓	✓
Easy to learn	✓	✗	✗	✓	✓

Newton School of Technology

The Origin of JavaScript

A Browser Ahead of Its Time

In the 1990s, Netscape Navigator changed the web. It was the first widely-used browser, capable of beautifully displaying HTML and CSS

NetScape Browser

Early HTML/CSS website

A World of Static Websites

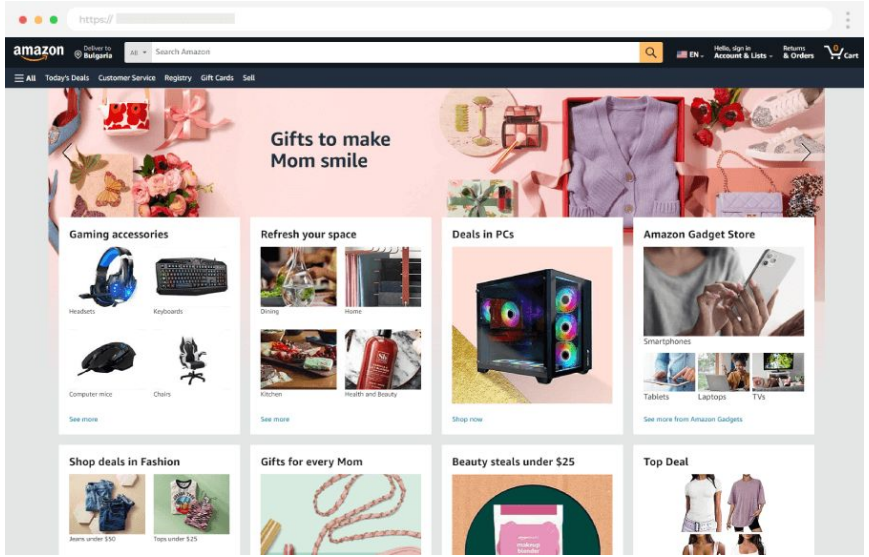
But there was a problem — web pages were static. They couldn't respond dynamically to user actions

Brainchild: Brendan Eich

To solve the issue, Brendan Eich was tasked with creating a Java-like scripting language for the web, and amazingly, he accomplished it in just 10 days!

JavaScript Revolutionizes the Web

With Brendan Eich's creation of JavaScript, websites transformed from static pages into dynamic, interactive experiences.



How to run Javascript code?

Newton School of Technology

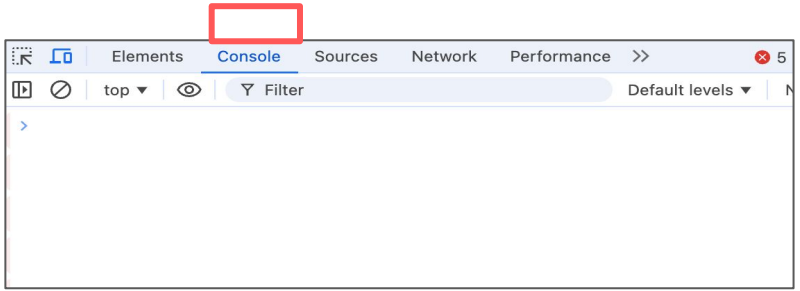
You can run JavaScript directly in a browser using the Developer Console or by embedding it in an HTML file with <script> tags.

1. Developer Console
2. Embedding in HTML(<script>...</script> tags)

Running javascript in Developer Console

Open the Developer Console in Chrome on Mac using Cmd + Option + J. Type your JavaScript code in the Console tab and press Enter to execute it.

While on any webpage, Press Cmd + Option + J




```
> console.log('hello world')
  hello world                VM244:1
< undefined
```

In Console you can write javascript code

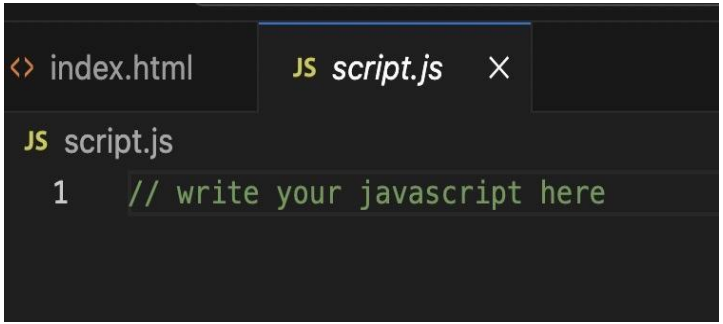
Run javascript: script tag

Newton School of Technology

We use script tag to either embed code inside it or link code:-

Embedding Code

```
<script src="script.js"></script>
```

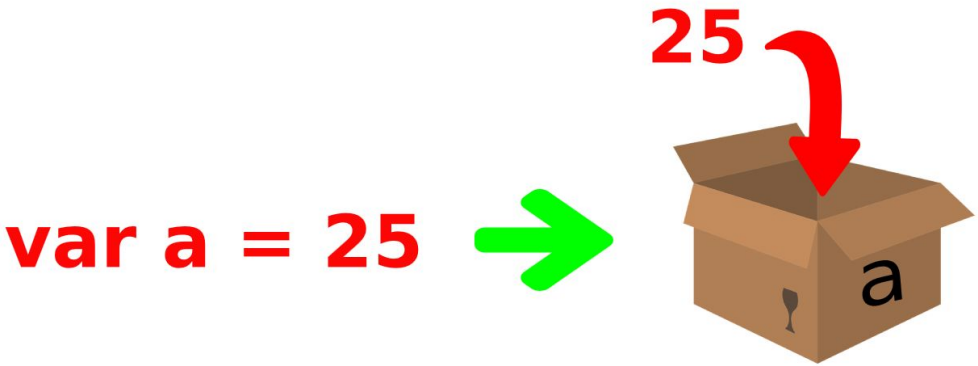


Linking External Code

Let’s Start Learning Javascript Syntax

Variables in Javascript

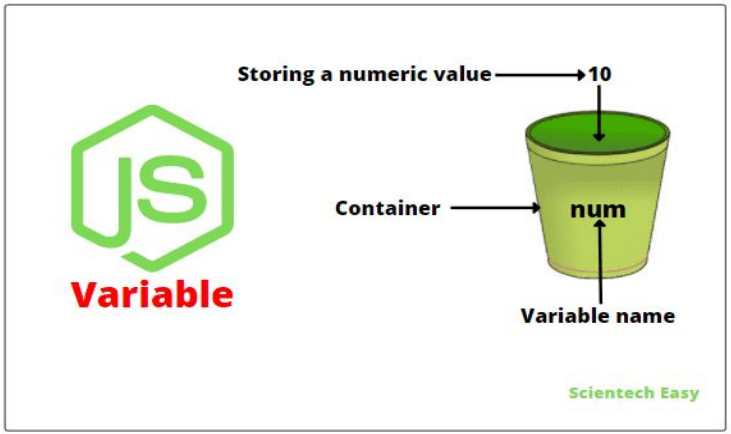
Variables in JavaScript are containers for storing data values. They allow developers to label and reference data in their programs,



Storing Values in Javascript

JavaScript provides three keywords to declare variables: **let**, **var**, and **const**.

These can be used to declare and store values.

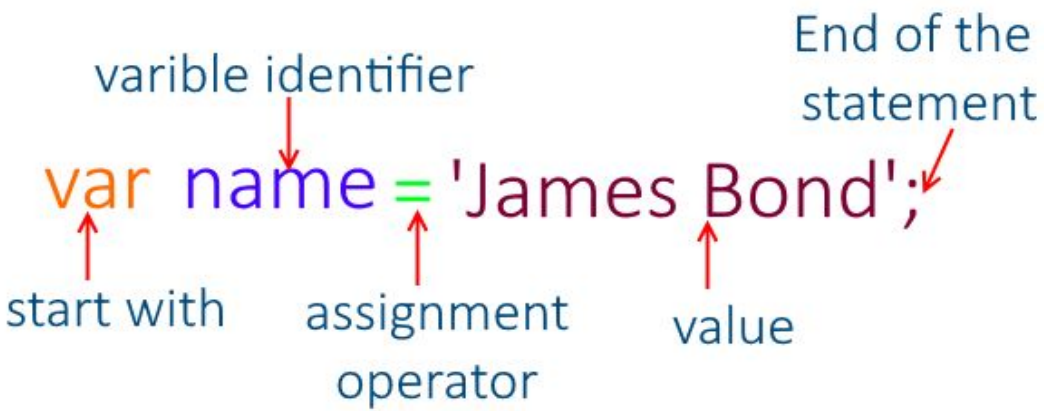



```
1  let num1 = 10;
2  var num2 = 20;
3  const num3 = 30;
```

Syntax for variable declaration

Variables in JavaScript are declared using let, const, or var, followed by a variable name.

Keyword used to declare a variable:-



Newton School of Technology

Similarly we can declare for let and const

Variables in JavaScript are declared using let, const, or var, followed by a variable name, let is an better alternative of var

```
let firstName = "Alice"; //can be reassigned
firstName = "Bob"; //reassignment is allowed

const age = 25; //can't be reassigned
```

Once assigned, the value of a const variable cannot be changed or reassigned.

Newton School of Technology

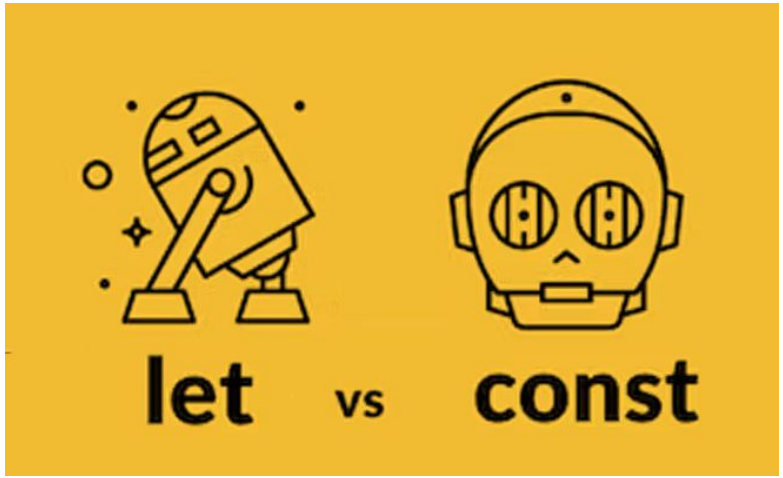
Difference between var, let and const,

Feature / Property	var	let	const
Re-declaration (in same scope)	✅ Allowed	❌ Not allowed	❌ Not allowed
Re-assignment	✅ Allowed	✅ Allowed	❌ Not allowed (value must stay the same)
Default Initialization	undefined	No default, must declare before use	No default, must declare before use
Mutability of value	Can change	Can change	Cannot re-assign primitive, but object properties can change
Best Use Case	Legacy code (not recommended)	Variables that may change	Constants / fixed references

Newton School of Technology

Understood!! But what about let and const

let lets you change the value later, while const keeps the value fixed once set.




```
// Using `let`
let firstName = "Alice";
// Declare and assign a value

firstName = "Bob"; // Reassign a new value

// Using `const`
const lastName = "Smith";
// Declare and assign a value
console.log(lastName); // Output: Smith

// Attempt to reassign a new value to `const`
// TypeError: Assignment to constant variable
```

Newton School of Technology

When you write code, how do you know what your program is doing?

Newton School of Technology

“When you write code, how do you know what your program is doing?”

Developers use the **console** to debug, inspect, and log output while writing & testing code.

What is the Console?

Newton School of Technology

- A tool built into browsers (and Node.js).
- Allows you to **log information**, **debug code**, and **inspect variables/values**.
- You write commands like `console.log()` to send output to the console.

Why Use the Console?

- ✅ Debug your code
- ✅ See intermediate values while developing
- ✅ Test snippets quickly
- ✅ Display warnings or errors clearly

Newton School of Technology

Types of Console Methods

Method	What it does	Example
console.log()	Print general info/output	**console.log("Hello World")**
console.error()	Print error messages (red in browser)	**console.error("Something went wrong")**
console.warn()	Print warnings (yellow in browser)	**console.warn("This is a warning")**
console.info()	Similar to log, often styled differently	**console.info("Info message")**
console.table()	Display array/object as a table	**console.table([{name: "A"}, {name: "B"}])**
console.clear()	Clears the console	**console.clear()**

Activity on console

Understanding console.log in javascript

console.log() is a JavaScript method that outputs messages or data, helping developers debug and monitor code execution.

```
> let firstName = "Alice
<· undefined
> console.log(firstName)
  Alice
```

Understanding console.log in javascript

Understanding console.log in javascript

Newton School of Technology

Here:

- The first argument is the string, starting with **%c** (to apply CSS to the text).

- The second argument is the CSS styles that apply to that log.

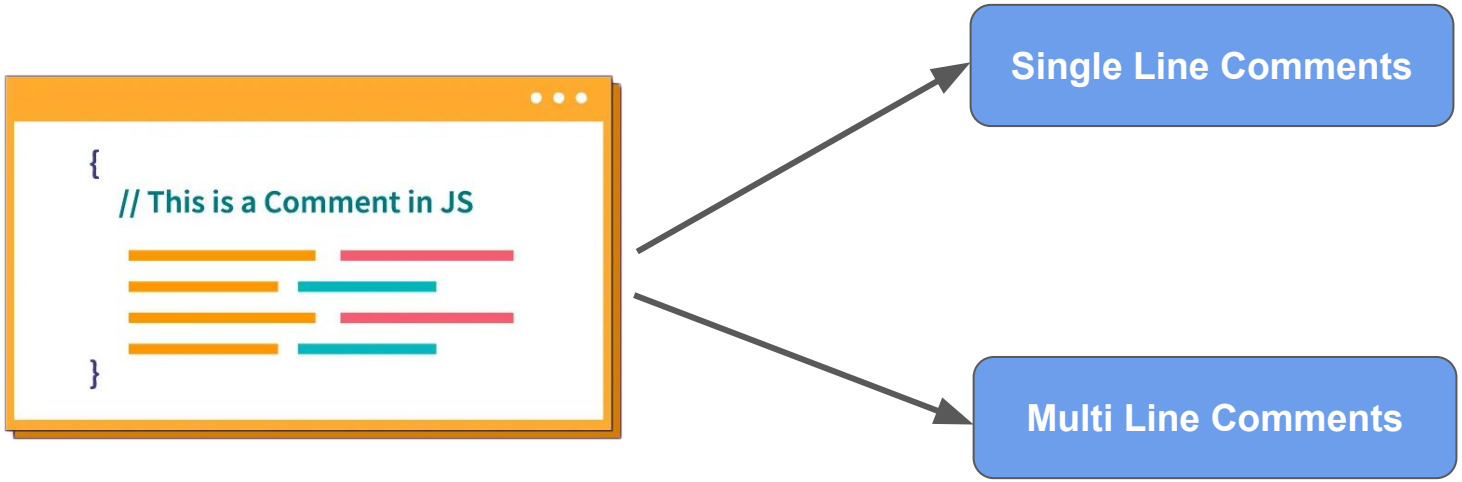
Comments: The Footnotes of Code

Comments are like notes in a book, offering explanations without affecting the code. Always comment your code, not just for others, but for your future self too.

Comments helps you understand the code better later one.

Types of Comments

You can use either single line comments or multi-line comments in javascript.



Comments: Single Line

Single-line comments are used to add brief explanations or notes within a single line of code.

Single line comments start with // and end within the line.

```
1  // This is a single-line comment
2  let x = 5;
   // This is another single-line comment after a statement
3
4  console.log(x); // Prints the value of x
```

Comments: Multi-Line

Multi-line comments are used to add longer explanations or comment out multiple lines of code, enclosed between /* and */.

```
/*
  This function calculates the total of cart items.
  If the total exceeds $100, a 10% discount is applied.
*/
function calculateTotal(cart) {
    let total = 0;
    for (let item of cart) total += item.price;
    return total > 100 ? total * 0.9 : total;
}
```

Use multi-line comments for longer explanations, commenting out blocks of code, or providing detailed documentation.

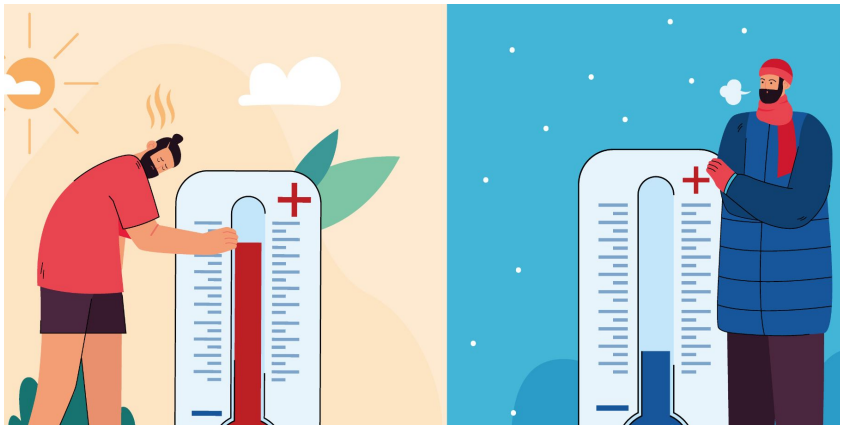
Newton School of Technology

Some Real World Examples

Temperature Changes Over a Day

The temperature of a room or a city changes throughout the day.

In Programming:

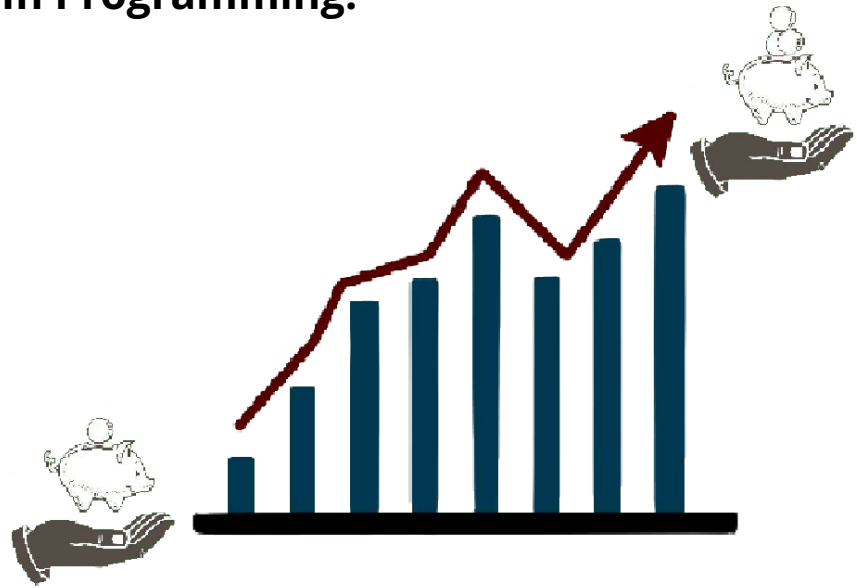


```
let temperature;
// Declare a variable to store the temperature

// Morning temperature
temperature = 20; // Assign morning temperature

// Afternoon temperature
temperature = 28;
// Update with afternoon temperature

// Evening temperature
temperature = 22;
// Update with evening temperature
```

Bank Account Balance

Your bank account balance fluctuates with deposits and withdrawals.

In Programming:



```
let bankBalance = 1000;
// initial balance in rupees
console.log(bankBalance); // output : 1000

// you make a deposit of $500
bankBalance += 500;
// Adding money to the account
console.log(bankBalance); // output : 1500

// you withdraw $300
bankBalance -= 300;
// subtracting money from the account
console.log(bankBalance); // output : 1200
```

Speed of a Car

A car’s speed changes depending on driving conditions and actions.

In Programming:

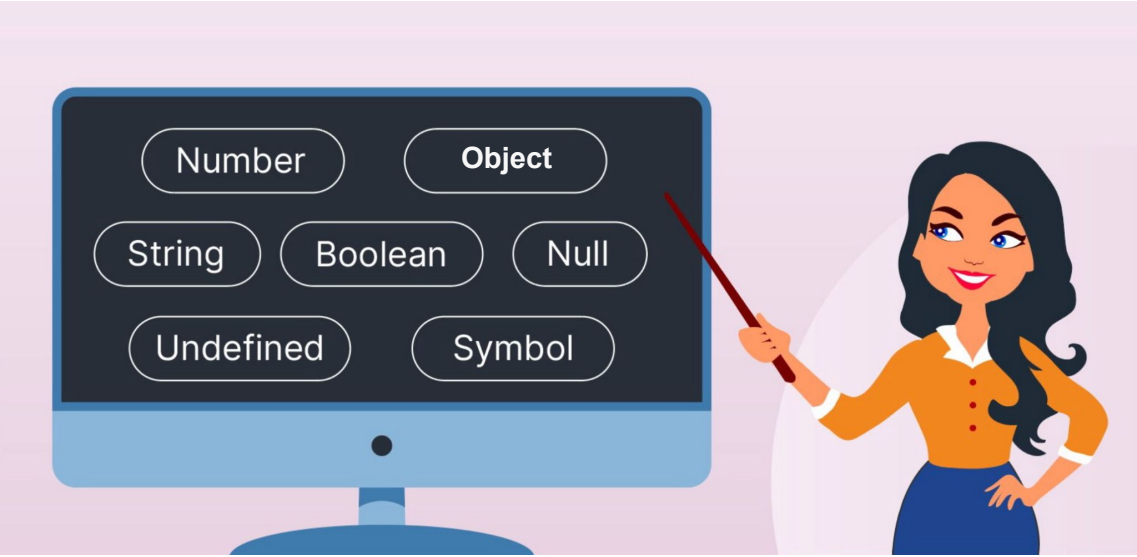

```
// Initial speed
carSpeed = 0;
// The car is stationary

// Speed after starting
carSpeed = 60;
// The car is now moving

// Speed on the highway
carSpeed = 100;
// The car speeds up on the highway

```

In Class Questions

What are Data Types in javascript?

Newton School of Technology

In JavaScript, **data types** are the classification of data values that tell the type of data being dealt with.



Data types indicate whether data is a number, string, boolean, etc.

Don't worry if you're unfamiliar with them yet.

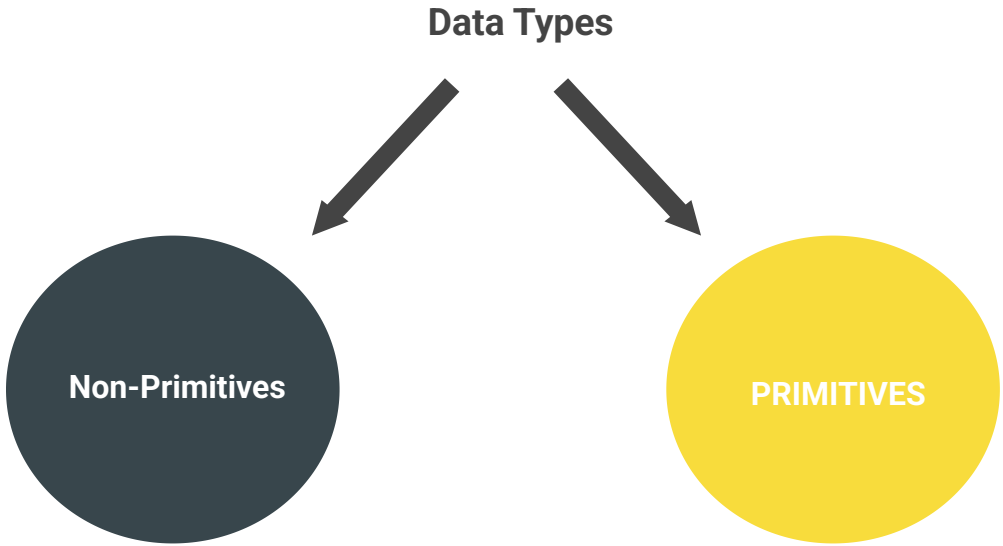
Newton School of Technology

Primitive and Non-Primitive Data Types

Categories of Data Types

Newton School of Technology

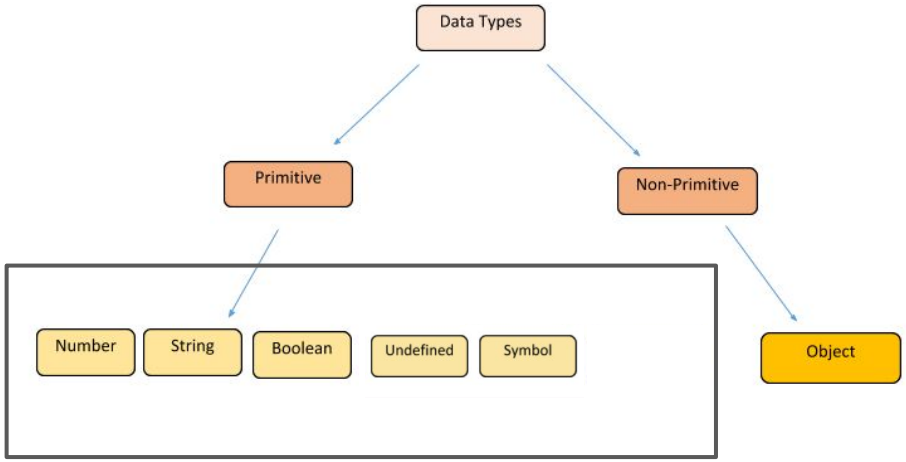
In JavaScript, data types are primitives (simple, immutable) and non-primitives (complex, mutable, and multi-valued).



Let’s start with primitives

Newton School of Technology

Primitive data types in JavaScript are simple, immutable values like numbers, strings, booleans, symbols.



Declaring Primitive Data types

Newton School of Technology

Primitive data types in JavaScript are simple, immutable values like numbers, strings, booleans, undefined, null etc. Let's have a look:-

```
// Primitive Data Types in JavaScript

// Number
let age = 25; // Integer
console.log("Age:", age); // Output: Age: 25

// String
let firstName = "John"; // Sequence of characters
console.log("First Name:", firstName);
// Output: First Name: John

// Boolean
let isActive = true; // Logical value
console.log("Is Active:", isActive);
// Output: Is Active: true
```

In JavaScript, the data type of a variable is implicitly determined by the value assigned to it, such as a `Number` for numbers, a `String` for text, and a `Boolean` for true/false values

Declaring Primitive Data types

Newton School of Technology

Primitive data types in JavaScript are simple, immutable values like numbers, strings, booleans, symbols.

```
1  // Primitive Data Types in JavaScript
2
3  // Undefined
4  let userLocation;
   // Variable declared but not assigned
5  console.log("User Location:", userLocation);
   // Output: User Location: undefined
6
7  // Null
8  let car = null; // Intentional absence of a value
9  console.log("Car:", car); // Output: Car: null
```

If a variable has no value, it's automatically undefined. To intentionally keep it empty, assign it null.

Sounds overwhelming! Don’t worry

Newton School of Technology

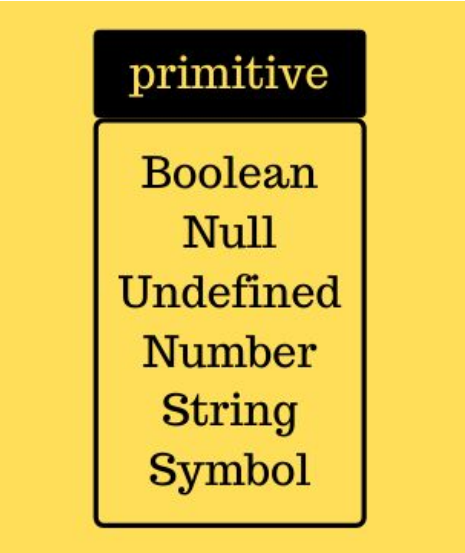
We all find it difficult when we start but it gets easier as we go along

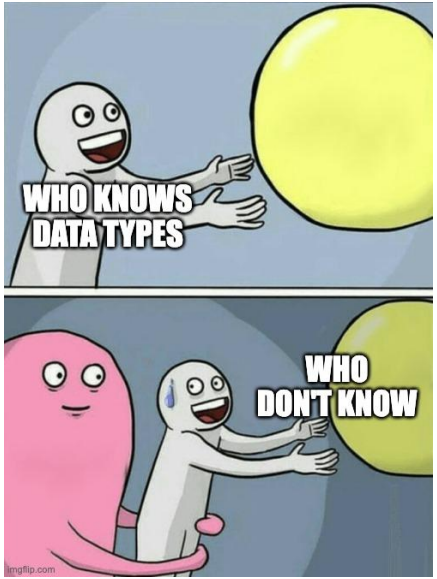
Ask questions when you're unsure, code when things feel confusing, and remember—the more you code, the better you'll understand!

Understanding Data types

Newton School of Technology

Today, we begin our journey through data types, starting with the most common and progressing to the more complex and powerful.

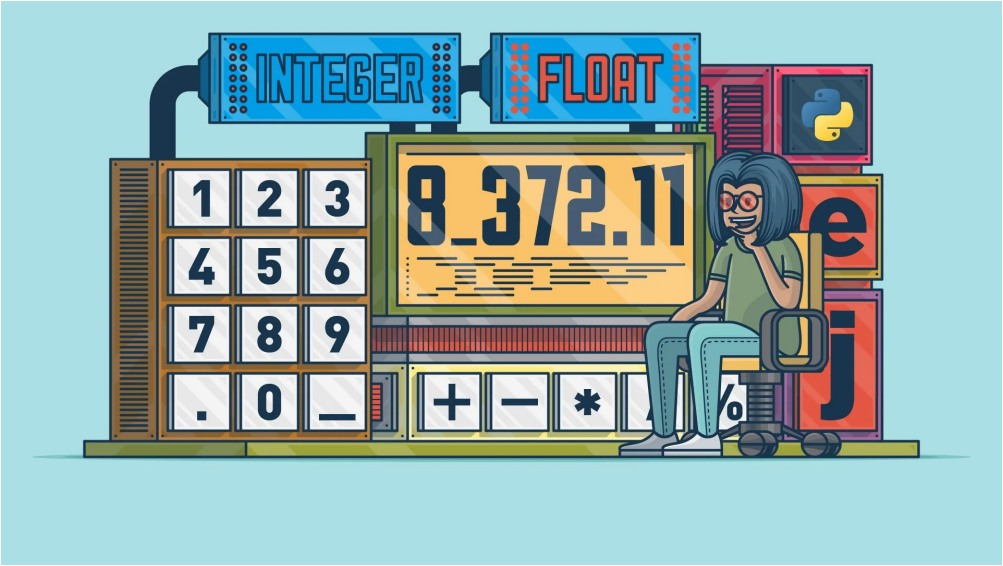




Meet the Numbers

Newton School of Technology

They represent quantities, from counting objects to calculating complex formulas. It could be either simple integers or decimals.



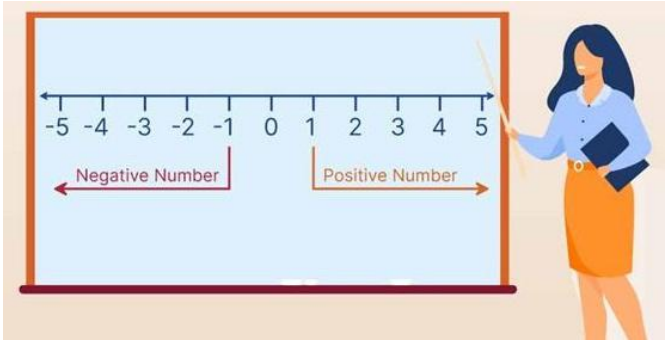
They could either be integers or fractions.

Declaration and Assignment

Newton School of Technology

Let’s declare and assign some numbers:-

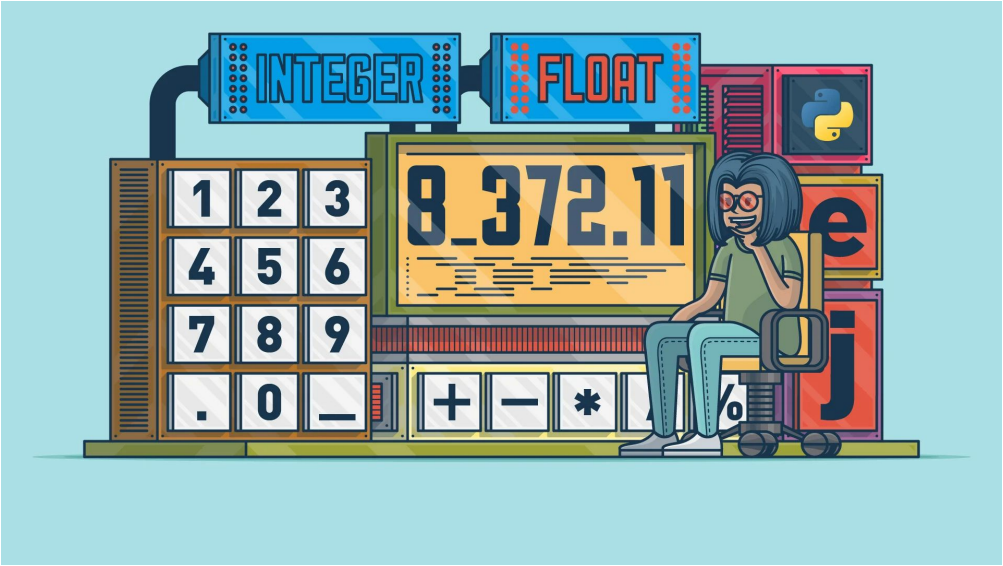
```
let intNum = 10;      // Positive integer
let negInt = -5;      // Negative integer
let floatNum = 3.14; // Positive float
let negFloat = -2.5; // Negative float
```



Enter the Strings

Newton School of Technology

They are the storytellers of JavaScript. They carry text, allowing us to write messages, descriptions, and even dialogue for our characters.



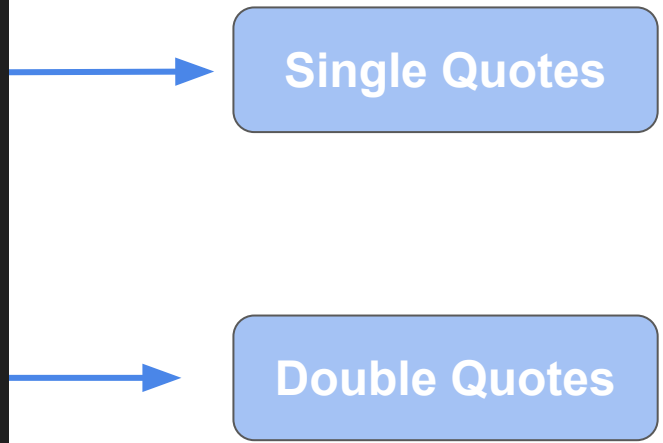
They could either be integers or fractions.

Declaration and Assignment

Newton School of Technology

Let’s declare and print some strings:-

```
1  // Using Single Quotes ('')
2  let singleQuoteString = 'Hello, world!';
3  console.log(singleQuoteString);
4  // Output: Hello, world!
5
6  // Using Double Quotes ("")
7  let doubleQuoteString = "Hello, world!";
8  console.log(doubleQuoteString);
9  // Output: Hello, world!
```



Declaration and Assignment

Newton School of Technology

Let’s learn how to use backticks and how it is different from ‘...’ and “...”

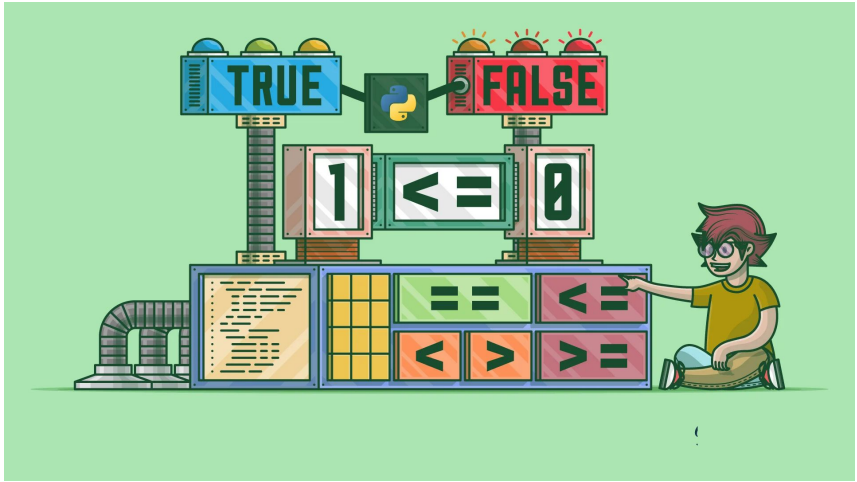
```
// Using Backticks (``)
let backtickString = `Hello, world!`;
console.log(backtickString);
// Output: Hello, world!

// Multi-line string using backticks
let multiLineString = `This is a string
that spans multiple
lines.`;
console.log(multiLineString);
// Output:
// This is a string
// that spans multiple
// lines.
```

Backticks Quotes

boolean

Newton School of Technology

Think of it as a switch—only two states: true or false.




```
1  let isSunny = true;
2  let isRaining = false;
```

Assigning boolean values

null

Newton School of Technology

This represents "nothing" or "empty". You can assign it intentionally when a variable is meant to have no value.


```
let car = null;
// This variable is explictly set to have no value

console.log(car);
// Output: null
```

undefined

Newton School of Technology

In JavaScript, undefined means a variable has been declared but has not been assigned a value yet. It also represents the default value of uninitialized variables.

undefined

defined

```
// Undefined example
let userAge;
console.log("userAge:", userAge);
// Output: undefined
// (variable is declared but not assigned a value)

// Defined example
let userName = "John Doe";
console.log("userName:", userName);
// Output: "John Doe"
// (variable is assigned a string value)
```

undefined vs defined

Any difference in null & undefined??

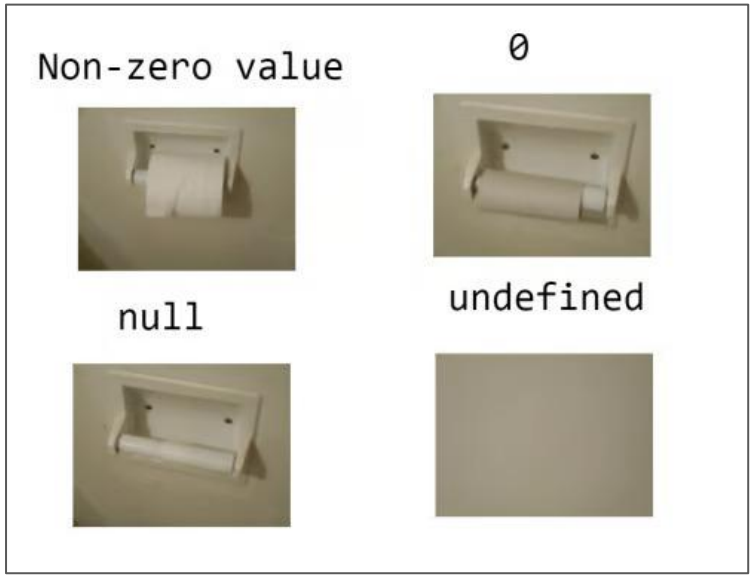
Newton School of Technology

Even though they look similar but they are not same at all. You have fallen in same trap like everyone else:-

Understanding differences

Newton School of Technology

Let’s understand difference between zero, null and undefined



undefined	null
has not been assigned	could be assigned
typeof undefined	typeof object

Truthy and Falsy values

Newton School of Technology

Any value which is not falsy is considered true.

Falsy Expressions
False
NaN
Undefined
Null
Empty String (“”/ ‘’)
0

Apart from these values all the values are considered true.

How to check truthy/falsy values??

Newton School of Technology

We can check truthy/falsy values using Boolean function.

```
let value = "Hello";
// Example truthy value

console.log(Boolean(value));
// Output: true

let anotherValue = "";
// Example falsy value
console.log(Boolean(anotherValue));
// Output: false
```

Here we used:-

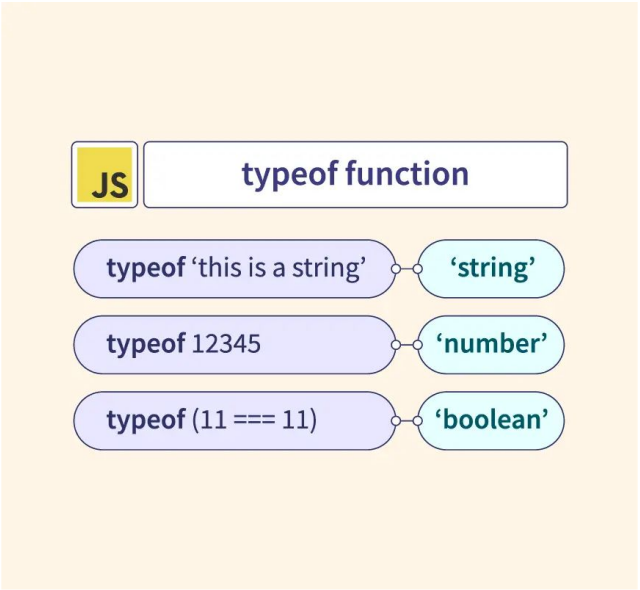
1. “Hello”: Evaluated as true
2. “”: Evaluated as false

Try few other examples by your own.

How to check data type: typeof

Newton School of Technology

The typeof operator is used to determine the type of a given variable or value. It returns a string indicating the type




```
// Declaring and assigning values to variables
let greeting = "This is a string";  // String
let num = 12345;                    // Number
let isEqual = 11 === 11;
// Boolean (True or False)

// Using typeof to check the type of each variable
console.log("Type of greeting:", typeof greeting);
// Output: string
console.log("Type of num:", typeof num);
// Output: number
console.log("Type of isEqual:", typeof isEqual);
// Output: boolean
```

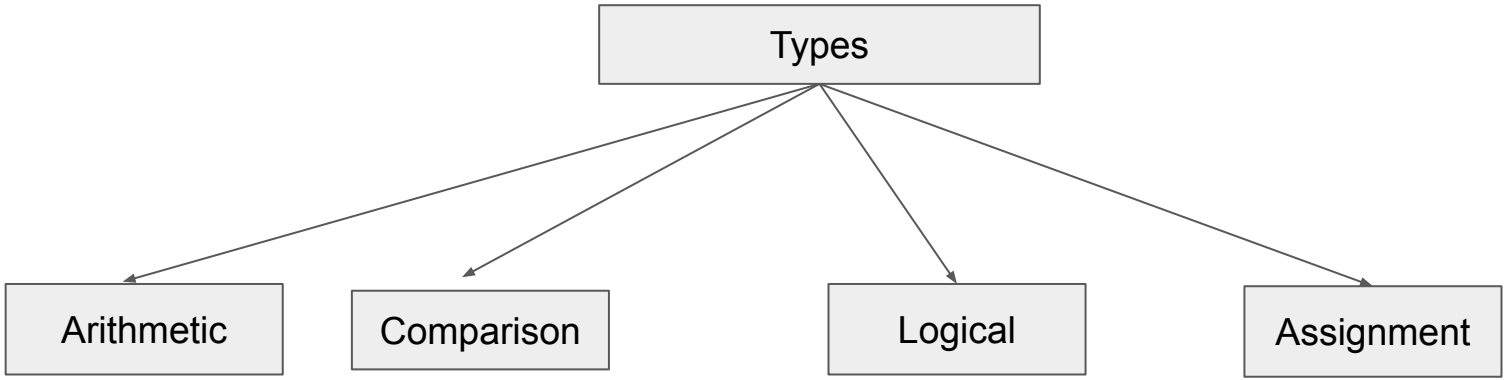
Newton School of Technology

Operators

What are operators?

Newton School of Technology

In JavaScript, **operators** are special symbols used to perform operations on values



Arithmetic Operators

Newton School of Technology

Arithmetic operators in JavaScript are used to perform mathematical operations on numbers.

Arithmetic Operator	Name	Example
+	Addition	a + b
-	Subtraction	a – b
*	Multiplication	a * b
/	Division	a / b
%	Modulus	a % b
++	Increment Operator	a++
--	Decrement Operator	a--

Arithmetic Operators: Example

Newton School of Technology

Here’s are few examples of arithmetic operations in JavaScript:

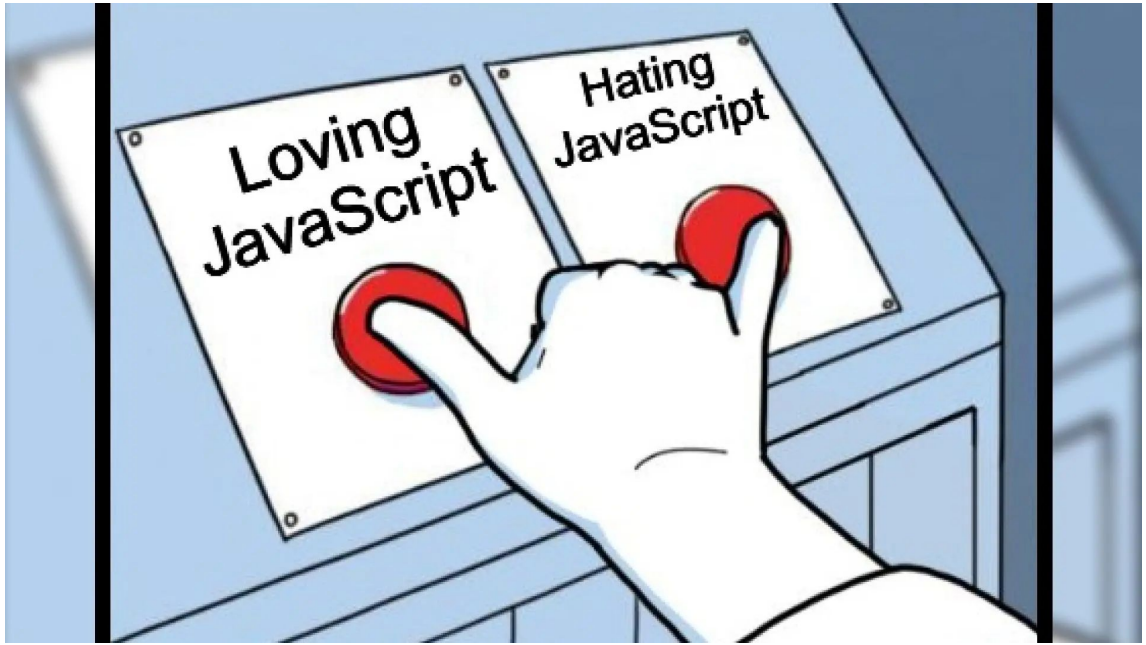
```javascript
// Declaring two variables
let num1 = 10;  // First number
let num2 = 5;   // Second number

// Addition
let addition = num1 + num2;
console.log("Addition: num1 + num2 =", addition);
// Output: 15

// Subtraction
let subtraction = num1 - num2;
console.log("Subtraction: num1 - num2 =", subtraction);
// Output: 5

// Multiplication
let multiplication = num1 * num2;
console.log("Multiplication: num1 * num2 =", multiplication);
// Output: 50
```

When you see your younger sibling struggling with basic algebra

Now how are you feeling??

Newton School of Technology

You must be having love-hate relationship now. Don’t worry, you will feel at home more you practice:-



Newton School of Technology

Some Real World Examples

Calculating Discounts and Special Offers

Newton School of Technology

When shopping, it's crucial to calculate the best price after discounts. For example, if an item costs $100 and has a 20% discount.

```
// Original price of the item
let originalPrice = 100;

// Discount percentage
let discountPercentage = 20;

// Calculate discount amount
let discountAmount = (originalPrice * discountPercentage) / 100;

// Calculate final price after discount
let finalPrice = originalPrice - discountAmount;

console.log("Original Price: $", originalPrice);
// Output: 100
console.log("Discount Amount: $", discountAmount);
// Output: 20
console.log("Final Price After Discount: $", finalPrice);
// Output: 80
```

Using arithmetic operators to determine the discount amount and the final price.

Job Offers: Making Right Choice

Newton School of Technology

To choose the better job offer, you can use comparison operators to evaluate salaries.

```
// Salary offers from two job companies
let offerA = 60000;  // Job offer A salary
let offerB = 75000;  // Job offer B salary

// Compare salaries using comparison operators
let isOfferAGreater = offerA > offerB;
// Checks if offer A is greater than offer B
let isOfferBGreater = offerB > offerA;
// Checks if offer B is greater than offer A

// Display the best offer
if (offerA > offerB) {
  console.log("Choose Offer A for a higher salary.");
} else if (offerB > offerA) {
  console.log("Choose Offer B for a higher salary.");
} else {
  console.log("Both offers are the same.");
}
```

Using a greater-than and smaller than to help you quickly see which offer provides a higher salary.

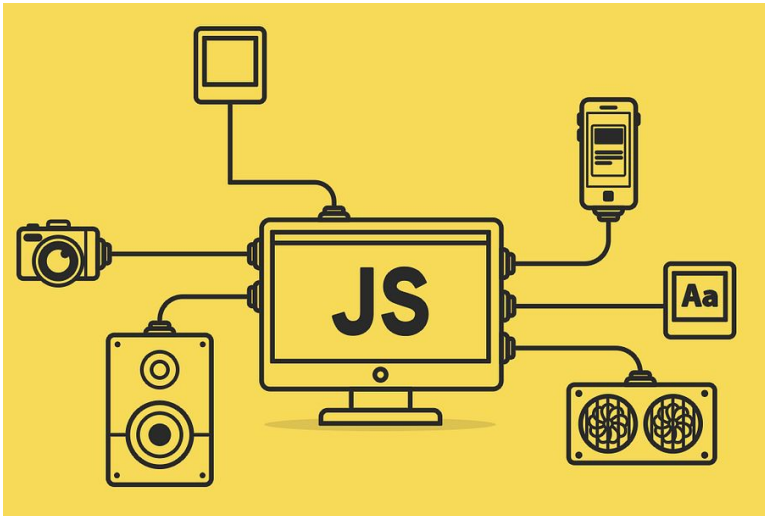
Practice! Practice! And become boss

Newton School of Technology

More you practice more you find easy way outs and more you will become confident in writing javascript code. Practice is the key...

In Class Questions

Did you know?



JavaScript powers over 98% of websites, making it most popular programming language! 🌐✨

Newton School of Technology

References

1. **MDN Web Docs - JavaScript**: Comprehensive and beginner-friendly documentation for JavaScript.
   https://developer.mozilla.org/en-US/docs/Web/JavaScript
2. **Eloquent JavaScript**: A free online book covering JavaScript fundamentals and advanced topics.
   https://eloquentjavascript.net/
3. **JavaScript.info**: A modern guide with interactive tutorials and examples for JavaScript learners.
   https://javascript.info/
4. **freeCodeCamp JavaScript Tutorials**: Free interactive lessons and coding challenges to learn JavaScript.
   https://www.freecodecamp.org/learn/

Newton School of Technology

Thanks for watching!