1️⃣ JavaScript Data Types

JavaScript data types tell us **what kind of value** a variable stores.

Main Types:

- **Number** – numbers
- **String** – text
- **Boolean** – true / false
- **Undefined** – declared but no value
- **Null** – empty value
- **BigInt** – very large numbers
- **Symbol** – unique values
- **Object** – collection of data

Example:

```
let age = 20;          // Number

let name = "John";     // String

let isPass = true;     // Boolean

let x;                 // Undefined

let y = null;          // Null
```

2️⃣ JavaScript typeof Operator

Use:

Used to **check the data type** of a variable.

Syntax:

```
typeof variable
```

Examples:

```
typeof 10        // "number"

typeof "Hello"   // "string"

typeof true      // "boolean"

typeof null      // "object" (JavaScript bug)
```

typeof undefined // "undefined"

📌 Mostly used for **debugging** and **checking input types**.

3️⃣ JavaScript toString() Method

Use:

Converts a **number or value into a string**.

Syntax:

```
value.toString()
```

Examples:

```
let num = 123;

let str = num.toString();


console.log(str);      // "123"

console.log(typeof str); // "string"
```

Use Case:

When you want to **display numbers as text**.

4️⃣ JavaScript Type Conversion

Use:

Changing **one data type into another**.

There are **two types**:

✅ Implicit Type Conversion (Automatic)

JavaScript converts the type **by itself**.

```
let x = 10 + "5";
```

Result:

```
"105"
```

Number becomes string automatically.

✅ Explicit Type Conversion (Manual)

Programmer converts the type.

String → Number

Number("123"); // 123

parseInt("10"); // 10

Number → String

String(100); // "100"

(100).toString();

Boolean → Number

Number(true); // 1

Number(false); // 0

Type Conversion Table (Easy)

From	To	Example
Number	String	String(10)
String	Number	Number("10")
Boolean	Number	Number(true)
Number	Boolean	Boolean(1)

Summary (1 line each)

- **Data Types** → Type of value stored
- **typeof** → Check data type
- **toString()** → Convert to string
- **Type Conversion** → Change one type to another

1️⃣ JavaScript Errors – Introduction

A **JavaScript error** happens when something goes wrong in the code,
due to **wrong syntax, wrong logic, or runtime problems**.

Common Reasons:

- Wrong spelling
- Missing brackets or quotes
- Using undefined variables
- Dividing by zero (logic error)

Example:

```
console.log(x); // x is not defined
```

2️⃣ JavaScript Silent Errors

What are Silent Errors?

Silent errors are errors that **do not show any message**,
but the program **does not work as expected**.

They are also called **logical errors**.

Example:

```
let x = 10;

let y = "5";


let result = x + y;

console.log(result);
```

Output:

```
105
```

No error message, but result is wrong logically.

3️⃣ JavaScript Error Statements

JavaScript provides **error handling statements** to manage errors.

try...catch

Used to **handle runtime errors**.

```
try {
  let x = y + 10;   // y is not defined
} catch (error) {
  console.log("Error occurred");
}
```

finally

Runs **always**, whether error occurs or not.

```
try {
  console.log("Try block");
} catch {
  console.log("Error");
} finally {
  console.log("Always executed");
}
```

throw

Used to **create custom errors**.

```
let age = 15;


if (age < 18) {
  throw "Age must be 18 or above";
}
```

---

4 JavaScript Error Object

When an error occurs, JavaScript creates an **Error object** with useful information.

Properties:

- **name** → error type
- **message** → error description

Example:

```
try {
 let x = y + 1;
} catch (err) {
 console.log(err.name);    // ReferenceError
 console.log(err.message); // y is not defined
}
```

Common Error Types:

- ReferenceError
- TypeError
- SyntaxError
- RangeError

5️⃣ JavaScript Debugging

What is Debugging?

Debugging means **finding and fixing errors** in code.

Common Debugging Methods:

1. console.log()

```
let x = 10;
console.log(x);
```

2. debugger

Stops execution for inspection.

```
let x = 5;
debugger;
x = x + 5;
```

3. Browser Developer Tools

- Press **F12**
- Open **Console**
- See errors and logs

Summary

- **JS Errors** → Mistakes in code
- **Silent Errors** → No message, wrong output
- **Error Statements** → try, catch, throw, finally
- **Error Object** → Stores error details
- **Debugging** → Finding and fixing errors
