JavaScript Syntax and Variables

JavaScript Syntax

JavaScript syntax refers to the **rules and structure** used to write JavaScript programs correctly.

Basic Syntax Rules

```
// Declaring variables
let x = 5;
let y = 6;

// Computing values
let z = x + y;

// This is a comment
```

JavaScript Values

JavaScript has two main types of values:

1. **Literals** – Fixed values
2. **Variables –** Values stored in containers

JavaScript Literals

Literals are fixed values written directly in the code.

Number Literals

```
10.50
1001
```

String Literals

Strings are written inside quotes:

"John Doe"

'John Doe'

JavaScript Keywords

Keywords are reserved words that have special meanings in JavaScript.

Example:

let x = 5;

const fname = "John";

⚠ Keywords are **case-sensitive**
let is valid, but LET or Let is not.

JavaScript Variables

Variables are **containers used to store data values**.

Example:

let x;

x = 6;

JavaScript Identifiers

Identifiers are names given to variables.

Rules:

- Must start with a letter, _, or $
- Can contain digits after the first character
- Cannot be JavaScript keywords
- Case-sensitive

Valid examples:

```
let firstName;

let _age;

let $price;
```

JavaScript Operators

Operators are used to perform operations.

Assignment Operator

```
let x = 5;
```

Arithmetic Operators

```
let sum = x + y;

let product = 5 * 10;
```

JavaScript Expressions

An expression is a combination of values, variables, and operators that produces a result.

Examples:

```
(5 + 6) * 10

x * 10

"John" + " " + "Doe"
```

JavaScript Case Sensitivity

JavaScript is case-sensitive.

```
let lastName = "Doe";

let lastname = "Peterson";
```

These are **different variables**.

JavaScript Naming Conventions (Camel Case)

- Hyphen case (not allowed): first-name
- Underscore: first_name
- Pascal Case: FirstName
- Camel Case (recommended): firstName

JavaScript Statements

Statements are instructions executed by JavaScript.

Example:

let x = 5;

let y = 6;

let z = x + y;

JavaScript Programs

A JavaScript program is a **sequence of statements** executed by the browser in order.

Semicolons

Semicolons separate statements.

let a = 5;

let b = 6;

let c = a + b;

Semicolons are optional but **highly recommended**.

JavaScript White Space

JavaScript ignores extra spaces.

```
let person = "Hege";

let person="Hege";
```

Best practice:

```
let x = y + z;
```

Line Length & Line Breaks

Avoid long lines. Break lines after operators.

```
document.getElementById("demo").innerHTML =
"Hello Dolly!";
```

JavaScript Code Blocks

Code blocks group statements using { }.

```
function myFunction() {
  document.getElementById("demo1").innerHTML = "Hello";
  document.getElementById("demo2").innerHTML = "How are you?";
}
```

JavaScript Comments

Single-Line Comment

```
// This is a comment

let x = 5; // Assign value
```

Multi-Line Comment

```
/*
This is a multi-line comment
used to explain code
*/
```

JavaScript Variables (Detailed)

Ways to Declare Variables

Modern JavaScript:

- let
- const

Old (Not Recommended):

- var
- Automatic declaration

Using let

```
let x = 5;
let y = 6;
let z = x + y;
```

- Block scoped
- Cannot be redeclared
- Can be reassigned

Using const

```
const price1 = 5;
const price2 = 6;
```

```
let total = price1 + price2;
```

- Block scoped
- Cannot be redeclared
- Cannot be reassigned
- Must be initialized

Using var (Not Recommended)

```
var x = 5;

var y = 6;

var z = x + y;
```

Problems with var:

- No block scope
- Can be redeclared
- Hoisted

JavaScript Data Types

Basic data types:

```
const pi = 3.14;

let name = "John";

let answer = 'Yes';
```

One Statement, Multiple Variables

```
let person = "John", car = "Volvo", price = 200;
```

Assignment Operator

= assigns values, it is not a comparison operator.

```
x = x + 5;
```

Constant Arrays

```
cars = ["Audi"];
```

Constant Objects

```
car = {type:"Volvo"};
```
