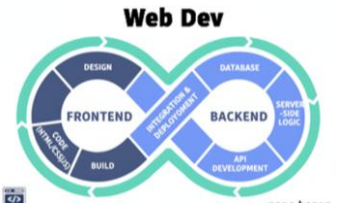


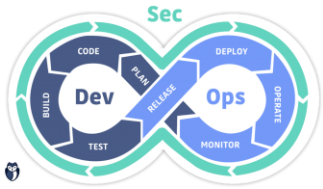


# Day 32

# “Web Development + Security”

## JavaScript Functions:

**What Is a Function in JavaScript?**

A function is a block of code designed to perform a specific task. You can define it once and reuse it as many times as you want.

Basic syntax:

function functionName(parameter1, parameter2, ...) {

// code to execute

return result; // optional

}

A very basic function:

```
JS main.js > ...
1   function nice(name){
2       console.log("Hello, " + name + "!");
3       console.log("Welcome to the program.");
4       console.log("I guess your name is " + name + ".");
5   }
6
7   nice("Aditya");

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node "e:\FullStackDevelopment\Day3
Hello, Aditya!
Welcome to the program.
I guess your name is Aditya.
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Another basic sum function:

```
9   function sum(a,b){
10      console.log(a+b);
11  }
12  sum(55,75);

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
130
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Another way to write the sum function: using the return keyword

```
15   //another way to write the same sum function
16   function add(a,b){
17       return a+b;
18   }
19   result = add(5,10);
20   console.log(result);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
15
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Adding more than one set of numbers:

```
22   //Another way to add the two numbers
23   function add(a,b){
24       return a+b;
25   }
26   result = add(5,10);
27   result1 = add(15,25);
28   result2 = add(100,200);
29   console.log(result);
30   console.log(result1);
31   console.log(result2);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
15
40
300
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Example: three parameters in the function

```
33  //Three paramenters function
34  function add(a,b,c=10){ //defaukt value of c is 10
35      return a+b+c;
36  }
37  result = add(5,10);
38  result1 = add(15,25,30); //here c will take 30 as value
39  console.log(result);
40  console.log(result1);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
25
70
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Example: arrow functions

```
42  //Arrow function
43  const func1 = (x) => {
44      console.log("Hey Aditya, " + x);
45  }
46
47  func1("How are you?");
48  func1(33);
49  func1(true);
50
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
Hey Aditya, How are you?
Hey Aditya, 33
Hey Aditya, true
PS E:\FullStackDevelopment\Day31-40\Day32>
```

## JavaScript Strings:

**What Is a String in JavaScript?**

A string is a sequence of characters enclosed in quotes — used to represent text.

**Ways to declare strings:**

| Type | Example | Description |
|---|---|---|
| Single quotes | 'Hello' | Common and simple |
| Double quotes | "Hello" | Common and flexible |
| Template literals (backticks) | `Hello` | Introduced in ES6, allows expressions & multiline strings |

A very basic example using the index:

Example: to get the string length

```
116   let a = "Hello";
117   console.log(a.length);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node main.js
5
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Example: template literals

```
119   //Template literals
120   let name = "Aditya";
121   console.log(`Hello, ${name}! Welcome to JavaScript.`);
122   //double quotes
123   console.log("Hello, " + name + "! Welcome to JavaScript.");
124   //single quotes
125   console.log('Hello, ' + name + '! Welcome to JavaScript.');
126   //Double quote inside single quote
127   console.log('He said, "Hello, ' + name + '! Welcome to JavaScript."');
128   //Single quote inside double quote
129   console.log("He said, 'Hello, " + name + "! Welcome to JavaScript.'");
130   //Both single and double quotes inside template literal
131   console.log(`He said, "It's great to see you, ${name}! Welcome to JavaScript."`);
132
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node .\main.js
Hello, Aditya! Welcome to JavaScript.
Hello, Aditya! Welcome to JavaScript.
Hello, Aditya! Welcome to JavaScript.
He said, "Hello, Aditya! Welcome to JavaScript."
He said, 'Hello, Aditya! Welcome to JavaScript.'
He said, "It's great to see you, Aditya! Welcome to JavaScript."
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Example: escape sequences

```
133  //Escape sequences
134  console.log("Hello,\nAditya!"); //New line
135  console.log("Hello,\tAditya!"); //Tab space
136  console.log("Hello, \"Aditya\"!"); //Double quote
137  console.log('Hello, \'Aditya\'!'); //Single quote
138  console.log("Hello, \\Aditya!"); //Backslash
139  console.log("Hello, \rAditya!"); //Carriage return
140  console.log("Hello, \bAditya!"); //Backspace
141  console.log("Hello, \fAditya!"); //Form feed
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node .\main.js
Hello,
Aditya!
Hello,  Aditya!
Hello, "Aditya"!
Hello, 'Aditya'!
Hello, \Aditya!
Aditya!
Hello,Aditya!
Hello,
Aditya!
PS E:\FullStackDevelopment\Day31-40\Day32>
```

Example: strings are immutable

```
143  //Strings are immutable
144  let str = "Hello";
145  str[0] = "J"; // ❌ Won't change
146  console.log(str); // Still "Hello"
147  str = "Jello"; // ✅ Reassigning works
148  console.log(str); // Now "Jello"
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\FullStackDevelopment\Day31-40\Day32> node .\main.js
Hello
Jello
PS E:\FullStackDevelopment\Day31-40\Day32>
```

--The End--