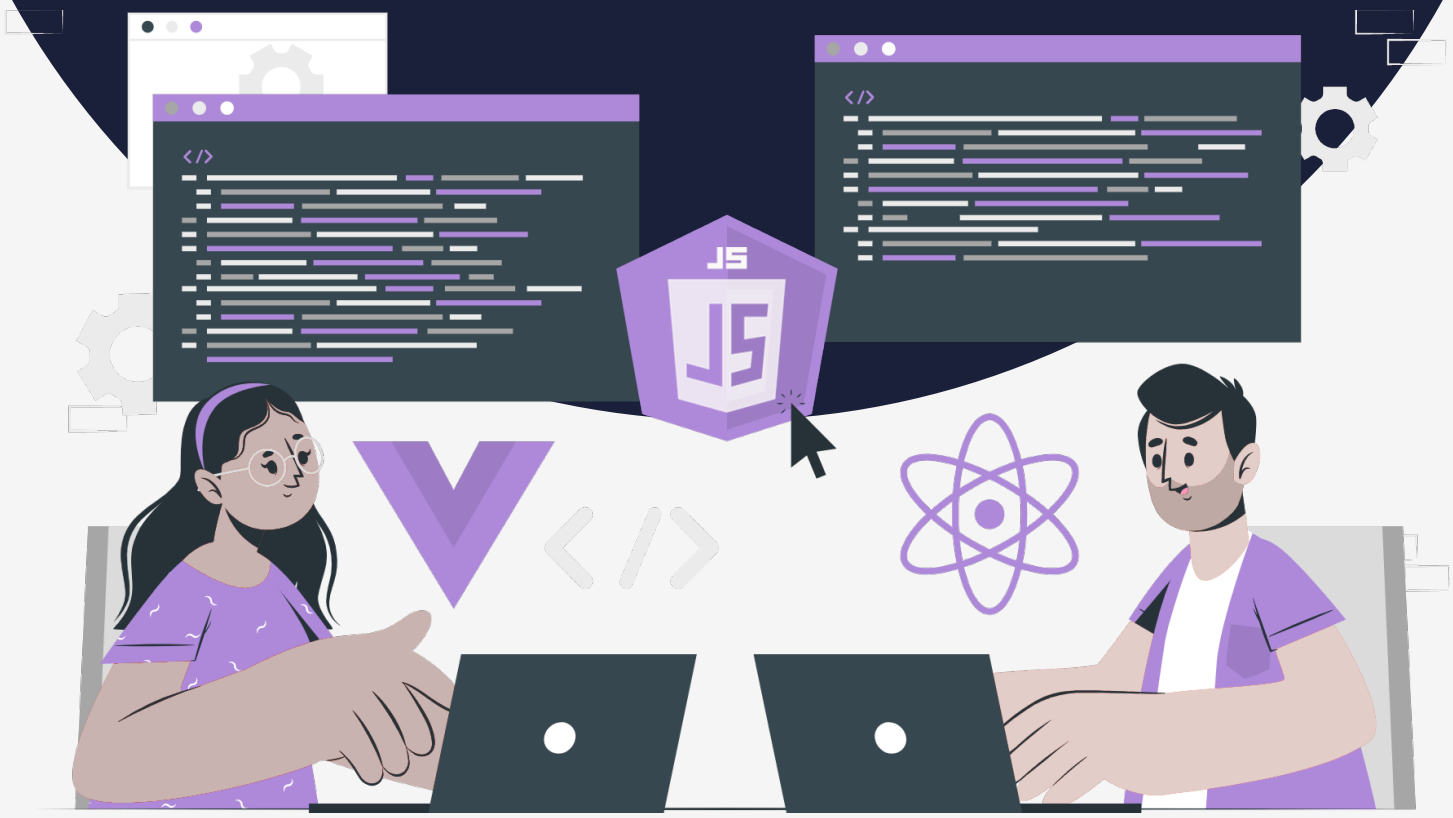
Lesson:

Switch Case



PW SKILLS

Topics Covered :

1. Introduction to switch case.
2. Break statement.
3. Syntax of switch case.
4. Implementation of switch case.
5. When to use switch statements or if/else statements.

Since we have been talking about conditionals from past lectures. Let's look at a real-life scenario, assume it's lunchtime and you walk to your favorite restaurant. The attendant offers you the menu. On the menu are different delicious items made for special people like you. You go through the menu and choose one or more meals from the menu and have yourself a good lunch. That is what switch statements help us do in JavaScript.

Switch statements allow you to execute different blocks of code based on different conditions (cases) being matched. It is a more efficient way to use multiple if/else checks.

We can have many case statements but unlike if statements, they are executed from the first matched value until a break is specified. It checks only for equality in the values. So, in order to come out of the switch statement we need to specify a break or the condition must match the default case.

A break statement is used to stop the execution of a loop or switch statement before it has completed all of its iterations or cases. We will look into the break statement in further lectures.

Switch statements use strict comparisons and for the code to be able to execute the values must be the same type.

Let's look at the syntax of the switch statement.

```
switch (expression) {

  case condition1:
    // code to execute;
    break;

  case condition2:
    // code to execute;
    break;

  case condition3:
    // code to execute;
    break;

  default:
  // default code;
}
```

Let's compare the switch syntax with the if statement.

```
switch (expression) {
    case condition1: // if (expression === condition1 then execute this block)
    // code to execute;
    break;

    case condition2: // if (expression === condition2 then execute this block)
    // code to execute;
    break;

    case condition3: // if (expression === condition3 then execute this block)
    // code to execute;
    break;

    default: // if (expression === none of the previous conditions then execute this block)
    // default code;
}
```

Let's now implement a switch statement considering an example. We represent our weekday in both number notation and text notation like 1 for Sunday, 2 for Monday, and so on. Let's take the number notation as input and provide text notation as output using the switch statement.

In this case, the only condition is that the number notation of the day must be between 1 to 7. If any other input is given then it will be considered invalid input.

Let's look at the code.

```
// Input
var day = 5; // Output: Thursday

switch (day) {
  case 1: // if (day === 1) then execute this block)
    console.log("Sunday");
    break;

  case 2: // if (day === 2) then execute this block)
    console.log("Monday");
    break;

  case 3: // if (day === 3) then execute this block)
    console.log("Tuesday");
    break;

  case 4: // if (day === 4) then execute this block)
    console.log("Wednesday");
    break;

  case 5: // if (day === 5) then execute this block)
    console.log("Thursday");
    break;
```

```
  case 6: // if (day === 6) then execute this block)
    console.log("Friday");
    break;

  case 7: // if (day === 7) then execute this block)
    console.log("Saturday");
    break;

  default: // if (expression === none of the previous conditions then execute this block)
    console.log("Day doesn't exist");
}
```

This code will output "Thursday" to the console because the value of the variable "day" is 5.
As we have passed the day to the switch statement, the day matches case 5 in the switch statement. When this case is executed, it will run the code block associated with it, which is console.log("Thursday").

The break statement at the end of the case ensures that the code execution exits the switch statement after the matching case has been executed, so the code in the other cases and the default block will not be executed.

When to use switch statements or if/else statements?

- if/else conditional branches are great for variable conditions that result in a Boolean, whereas switch statements are great for fixed data values.
- In a situation where more than one choice is preferred, the switch is a better choice than an if/else statement.
- Considering the speed of execution it is advised if the number of cases is more than 5 use a switch, otherwise, you may use if-else statements.
- Switch is more readable and looks much cleaner when you have to combine cases.