JavaScript M3-4 Notes

Very often when you write code, you want to perform different actions based on different conditions.
You can do this by using conditional statements in your code.
Use if to specify a block of code that will be executed if a specified condition is true.

```
if (condition) {
  statements
}
```

The statements will be executed only if the specified condition is true.
Example:

```
var myNum1 = 7;
var myNum2 = 10;
if (myNum1 < myNum2) {
  alert("JavaScript is easy to learn.");
}
```

Use the else statement to specify a block of code that will execute if the condition is false.

```
if (expression) {
  // executed if condition is true
}
else {
  // executed if condition is false
}
```

The example below demonstrates the use of an if...else statement.

```
var myNum1 = 7;
var myNum2 = 10;
if (myNum1 > myNum2) {
  alert("This is my first condition");
}
else {
  alert("This is my second condition");
}
```

Try It Yourself

The above example says:
- If myNum1 is greater than myNum2, alert "This is my first condition";
- Else, alert "This is my second condition".

The browser will print out the second condition, as 7 is not greater than 10.
You can use the else if statement to specify a new condition if the first condition is false.

Example:

```
var course = 1;
if (course == 1) {
  document.write("<h1>HTML Tutorial</h1>");
} else if (course == 2) {
  document.write("<h1>CSS Tutorial</h1>");
} else {
  document.write("<h1>JavaScript Tutorial</h1>");
}
```

The final else block will be executed when none of the conditions is true.

Let's change the value of the course variable in our previous example.

```
var course = 3;
if (course == 1) {
  document.write("<h1>HTML Tutorial</h1>");
} else if (course == 2) {
  document.write("<h1>CSS Tutorial</h1>");
} else {
  document.write("<h1>JavaScript Tutorial</h1>");
}
```

In cases when you need to test for multiple conditions, writing if else statements for each condition might not be the best solution.
The switch statement is used to perform different actions based on different conditions.

Syntax:

```
switch (expression) {
 case n1:
   statements
   break;
 case n2:
   statements
   break;
 default:
   statements
```

}
Consider the following example.

```
var day = 2;
switch (day) {
  case 1:
    document.write("Monday");
    break;
  case 2:
    document.write("Tuesday");
    break;
  case 3:
    document.write("Wednesday");
    break;
  default:
    document.write("Another day");
}
// Outputs "Tuesday"
```

When JavaScript code reaches a break keyword, it breaks out of the switch block. This will stop the execution of more code and case testing inside the block.
Usually, a break should be put in each case statement.