6a Intructions JS API pages 89 - 96 Conditionals

Nicholas Bruzzese

July 26, 2025

1 Lecture: Understanding Conditionals in JavaScript

Conditionals are a fundamental concept in JavaScript that allow a program to make decisions based on specific conditions. They act as control structures, determining which parts of the code are executed and when, based on whether a condition is true or false. This lecture explains conditionals in JavaScript, focusing on their structure and practical use, tailored for a young learner with some coding experience.

1.1 Embedding JavaScript in HTML

Before exploring conditionals, it is important to understand how JavaScript code is included in a web page. JavaScript is typically embedded within an HTML file using the `<script>` element. Unlike other HTML elements, such as `<p>` or `<h1>`, which display content on the page, the `<script>` element contains JavaScript code that the browser's JavaScript interpreter executes. For example:

```
    <script>
    var message = "Hello world!";
    console.log(message);
    </script>

```

In this code, a variable `message` is created with the value `"Hello world!"`, and `console.log(message)` prints that value to the browser's console. The console is a tool where developers can see output from their code, useful for testing and debugging. Unlike typing directly into the JavaScript console, where each line runs immediately, code in an HTML file runs from top to bottom when the page loads, and only outputs to the console if explicitly instructed, such as with `console.log`.

1.2 What Are Conditionals?

A conditional statement allows a program to execute certain code only if a specific condition is true. Think of it like making a choice: for example, if it is raining, take an umbrella; otherwise, leave it at home. In JavaScript, conditionals control the flow of a program by deciding which code to run based on whether a condition evaluates to `true` or `false`.

1.3 If Statements

The simplest form of conditional is the `if` statement. It checks whether a condition is true and, if so, executes a block of code called the body. The structure of an `if` statement is:

```
if (condition) {
  // Code to run if the condition is true
}

```

The condition must be an expression that evaluates to a Boolean value (`true` or `false`). For example:

```
var name = "Nicholas";
console.log("Hello " + name);
if (name.length > 7) {
  console.log("Wow, you have a REALLY long name!");
}

```

Here, the variable `name` is set to `"Nicholas"`. The condition `name.length > 7` checks if the length of the string `"Nicholas"` (which is 8 characters) is greater than 7. Since this is true, the console prints:

```
Hello Nicholas
Wow, you have a REALLY long name!
```

If the condition is false, the body of the `if` statement is skipped. For example, changing the name to `"Nick"` (4 characters) means `name.length > 7` is false, so only `Hello Nick` is printed, and the `if` statement's body is ignored.

1.4 If...Else Statements

An `if...else` statement extends the `if` statement by providing an alternative block of code to run if the condition is false. The structure is:

```
if (condition) {
  // Code to run if the condition is true
} else {
  // Code to run if the condition is false
}

```

For example:

```
var name = "Nicholas";
console.log("Hello " + name);
if (name.length > 7) {
  console.log("You have a REALLY long name!");
} else {
  console.log("Your name isn't very long");
}

```

If `name` is `"Nicholas"`, the condition `name.length > 7` is true, so the output is:

```
Hello Nicholas
You have a REALLY long name!
```

If `name` is changed to `"Nick"`, the condition is false, so the `else` body runs, producing:

```
Hello Nick
Your name isn't very long
```

The `if...else` statement ensures that one of the two blocks of code (either the `if` body or the `else` body) will always execute, depending on the condition.

1.5 Chaining If...Else Statements

Sometimes, a program needs to check multiple conditions. This is done by chaining `if...else` statements using the `else if` clause. The structure looks like:

```
    if (condition1) {
      // Code if condition1 is true
    } else if (condition2) {
      // Code if condition2 is true
    } else if (condition3) {
      // Code if condition3 is true
    } else {
      // Code if all conditions are false
    }

```

Each condition is checked in order, and the first true condition triggers its corresponding body. Once a true condition is found, the rest of the chain is skipped. If none of the conditions is true, the `else` body (if present) runs. For example:

```
    var lemonChicken = false;
    var beefWithBlackBean = true;
    var sweetAndSourPork = false;
    if (lemonChicken) {
      console.log("Great! I'm having lemon chicken!");
    } else if (beefWithBlackBean) {
      console.log("I'm having the beef...");
    } else if (sweetAndSourPork) {
      console.log("OK, I'll have the pork...");
    } else {
      console.log("Guess I'll go hungry!");
    }

```

In this case, since `beefWithBlackBean` is `true`, the output is:

```
I'm having the beef...
```

The program stops checking conditions after finding the first true one, so the `sweetAndSourPork` condition and the `else` block are ignored. If all conditions are false and there is no `else` block, no code in the chain executes. For example:

```
var lemonChicken = false;
var beefWithBlackBean = false;
var sweetAndSourPork = false;
if (lemonChicken) {
  console.log("Great! I'm having lemon chicken!");
} else if (beefWithBlackBean) {
  console.log("I'm having the beef...");
} else if (sweetAndSourPork) {
  console.log("OK, I'll have the pork...");
}

```

Here, nothing is printed because no conditions are true, and there is no `else` block.

1.6 Practical Example: Personalised Greetings

To illustrate how conditionals work, consider a program that prints a personalised greeting based on a `name` variable:

```
var name = "Alex";
if (name == "Alex") {
  console.log("Hello mate!");
} else {
  console.log("Hello stranger!");
}

```

Here, the condition `name == "Alex"` checks if the `name` variable equals `"Alex"`. Since it does, the output is:

```
Hello mate!
```

To handle multiple names, such as greeting a parent, you can use `else if`:

```
var name = "Dad";
if (name == "Alex") {
  console.log("Hi mate!");
} else if (name == "Dad") {
  console.log("Hi Dad!");
} else if (name == "Mum") {
  console.log("Hi Mum!");
} else {
  console.log("Hello stranger!");
}

```

If `name` is `"Dad"`, the output is:

```
Hi Dad!
```

This program checks each condition in order and prints the appropriate greeting. If `name` does not match `"Alex"`, `"Dad"`, or `"Mum"`, it prints `"Hello stranger!"`.

2 Summary

Conditionals in JavaScript are essential for controlling a program's flow. The `if` statement runs code only if a condition is true. The `if...else` statement provides an alternative action if the condition is false. Chaining `else if` clauses allows checking multiple conditions, with an optional `else` block to handle cases where no conditions are true. By embedding JavaScript in an HTML `<script>` element and using `console.log`, you can test and observe the results of your conditionals in the browser's console. These tools enable you to write programs that make decisions, making your code more dynamic and interactive.