JavaScript Essentials

Conditionals

Table of Contents

Lesson Objectives

- Understand the needs to make decisions and carry out actions accordingly depending on different inputs, scenario
- Understand how conditional statements work in JavaScript
- Able to use if…else, switch, ternary operator with ease to make decisions

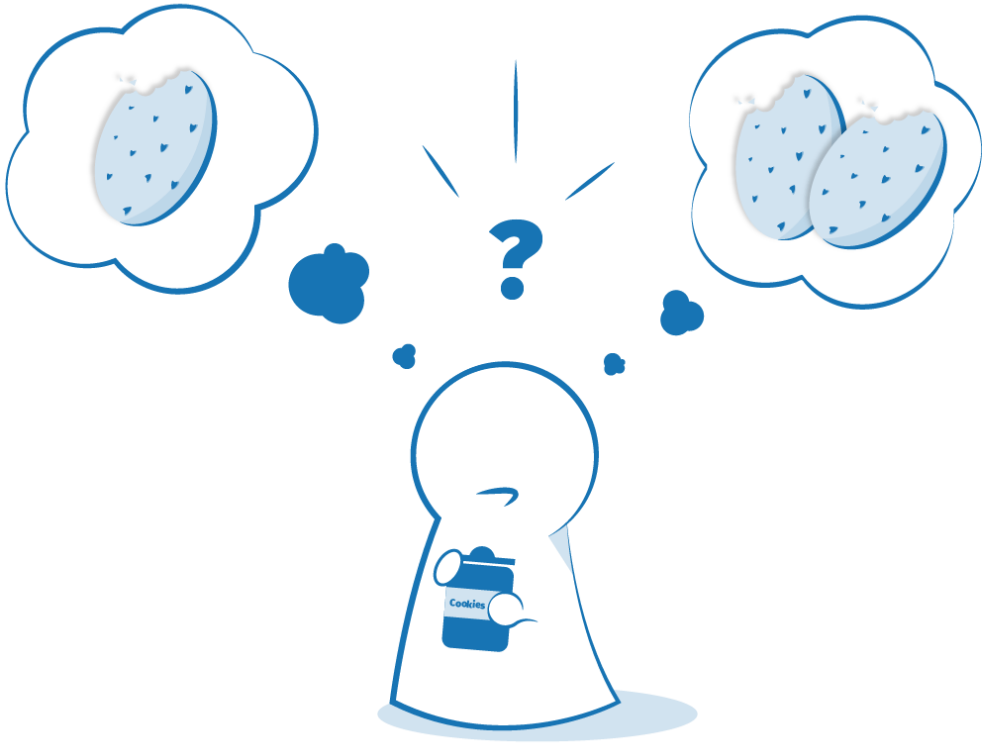
Section 1

Overview – What is a condition?

Overview – What is a condition?

- In any programming language, the code needs to make decisions and carry out actions accordingly depending on different inputs.
- **For example:** in a game, if the player's number of lives is 0, then it's game over.
- In a weather app, if it is being looked at in the morning, show a sunrise graphic; show stars and a moon if it is nighttime.

Overview – What is a condition?



Overview – Summary

- Conditional plays a big role in any Programming language
- It helps to make decisions and carry out actions accordingly depending on different inputs
- JavaScript provide such conditionals or control flow mechanism

Section 2

if…else statement

if…else statement – Basic syntax

- Let's look at by far the most common type of conditional statement you'll use in JavaScript — the humble if...else statement.
- Basic if...else syntax looks like the following in pseudocode:

```
1 if (condition) {
2   code to run if condition is true
3 } else {
4   run some other code instead
5 }
```

if…else statement – Basic syntax

- You should note that you don't have to include the else and the second curly brace block — the following is also perfectly legal code:

```
1 if (condition) {
2   code to run if condition is true
3 }
4
5 run some other code
```

if…else statement – Basic syntax

- As a final point, you may sometimes see if...else statements written without the curly braces, in the following shorthand style:

```
1 if (condition) code to run if condition is true
2 else run some other code instead
```

- **Not recommended**

if…else statement – Basic syntax

- Example 1: The parent might say "Hey sweetheart! If you help me by going and doing the shopping, I'll give you some extra allowance so you can afford that toy you wanted."

if…else statement – Basic syntax

- The last example provided us with **two** choices, or outcomes — but what if we want **more than two**?
- There is a way to chain on extra choices/outcomes to your if...else — using **else if**. Each extra choice requires an additional block to put in between if() { ... } and else { ... }
- Example 2: Checking weather

if…else statement – Tip on condition

- Comparison operators are used to test the conditions inside our conditional statements
  1. === and !== : test if one value is identical to, or not identical to, another.
  2. < and > : test if one value is less than or greater than another.
  3. <= and >= : test if one value is less than or equal to, or greater than or equal to, another.

if…else statement – Nesting if…else

- It is perfectly OK to put one if...else statement inside another one — to nest them.

```
if (choice === 'sunny') {
  if (temperature < 37) {
    para.textContent =
      'It is nice and sunny outside t
  } else if (temperature >= 37) {
    para.textContent = 'It is SUPER H
  }
} else if (choice === 'rainy') {
```

if…else statement – Logical operators

- If you want to test multiple conditions without writing nested if...else statements, logical operators can help you. When used in conditions, the first two do the following:
  1. && — AND
  2. || — OR

if…else statement – Logical operators

- To give you an **AND** example, the previous example snippet can be rewritten to this:

```
if (choice === 'sunny' && temperature < 37) {
  para.textContent =
    'It is nice and sunny outside today. Wear shorts!
} else if (choice === 'sunny' && temperature >= 37) {
  para.textContent = 'It is SUPER HOT. Dont go out.';
} else if (choice === 'rainy') {
```

if…else statement – Logical operators

- Let's look at a quick **OR** example:

```
1 | if (iceCreamVanOutside || houseStatus === 'on fire') {
2 |   console.log('You should leave the house quickly.');
3 | } else {
4 |   console.log('Probably should just stay in then.');
5 | }
```

if…else statement – Logical operators

- The last type of logical operator, **NOT**, expressed by the ! operator, can be used to **negate** an expression. Let's combine it with **OR** in the above example:

```
1 | if (!(iceCreamVanOutside || houseStatus === 'on fire')) {
2 |   console.log('Probably should just stay in then.');
3 | } else {
4 |   console.log('You should leave the house quickly.');
5 | }
```

if…else statement – Logical operators

- You can combine as many logical statements together as you want, in whatever structure

```
if ((x === 5 || y > 3 || z <= 10) && (loggedIn || userName === 'Steve')) {
  // run the code
}
```

if…else statement – Logical operators

- Common mistake (wrong syntax):

```
1 | if (x === 5 || 7 || 10 || 20) {
2 |   // run my code
3 | }
```

if…else statement – Logical operators

- This is logically not what we want!
- To make this work you've got to specify a complete test either side of each **OR** operator:

```
1 | if (x === 5 || x === 7 || x === 10 ||x === 20) {
2 |   // run my code
3 | }
```

if…else - Summary

- **if…else** is probably the most common type of conditional statement in JavaScript
- Basic syntax: **if …else**
- You can have as much **else…if** as you need
- You can nest as much **if…else** as you want (**Not recommended)**
- Make use of Logical operators to strike for cleaner and easy to understandab code
- 3 type of logical operators: **AND, OR, NOT**

Section 3

switch statement

switch statement – Problem with if…else

- if...else statements do the job of enabling conditional code well, but they are not without their downsides.

```
var field;
// check field
if (fied === 'id') {
  // code
} else if (field === 'name') {
  // code
} else if (field === 'age') {
  // code
} else if (field === 'address') {
  // code
} else if (field === 'clazz') {
  // code
} // more else if
```

switch statement – For the rescue

- In such a case, switch statements are your friend
- It takes a single expression/value as an input,
- Look through a number of choices until they find one that matches that value
- Executing the corresponding code that goes along with it.

switch statement – For the rescue

- Here's some more pseudocode, to give you an idea:

```
switch (expression) {
  case choice1:
    run this code
    break;

  case choice2:
    run this code instead
    break;

  // include as many cases as you like

  default:
    actually, just run this code
}
```

switch statement – For the rescue

- Convert previous example to switch:

```
var field;
// check field
switch (field) {
  case 'id': {
    // code
    break;
  }
  case 'name': {
    // code
    break;
  }
  case 'age': {
    // code
    break;
  }
  default: {
    // code
  }
}
```

Practice 1: switch statement

switch statement - Summary

- When you get a large number of choices think about **switch**
- Syntax for switch is: switch (value) { case: }
- **default** case is optional but it’s recommended to include one
- Do not forget the **break** keywork if you don’t want your program to behave strangely

Section 4

Ternary operator

Ternary operator – Syntax

- The ternary or conditional operator is a small bit of syntax that tests a condition and returns one value/expression if it is true, and another if it is false
- The pseudocode looks like this:

```
( condition ) ? run this code : run this code instead
```

Ternary operator – Example

- So let's look at a simple example:

```
var isLoggedin = false;

var greeting = isLoggedin ? 'Hello user, you are logged in' : 'Hello stranger';
greeting;
"Hello stranger"
```

Practice 2: ternary operator

Ternary operator - Summary

- Useful in some situations, and can take up a lot less code than an if...else block if you simply have two choices that are chosen between via a true/false condition.
- Is an expression not a statement so it can be anywhere that expects an expression, for example: on the right side of = operator, an item an array, condition in if…else or switch, in template string and logical operators and much more

Thank you

Q&A