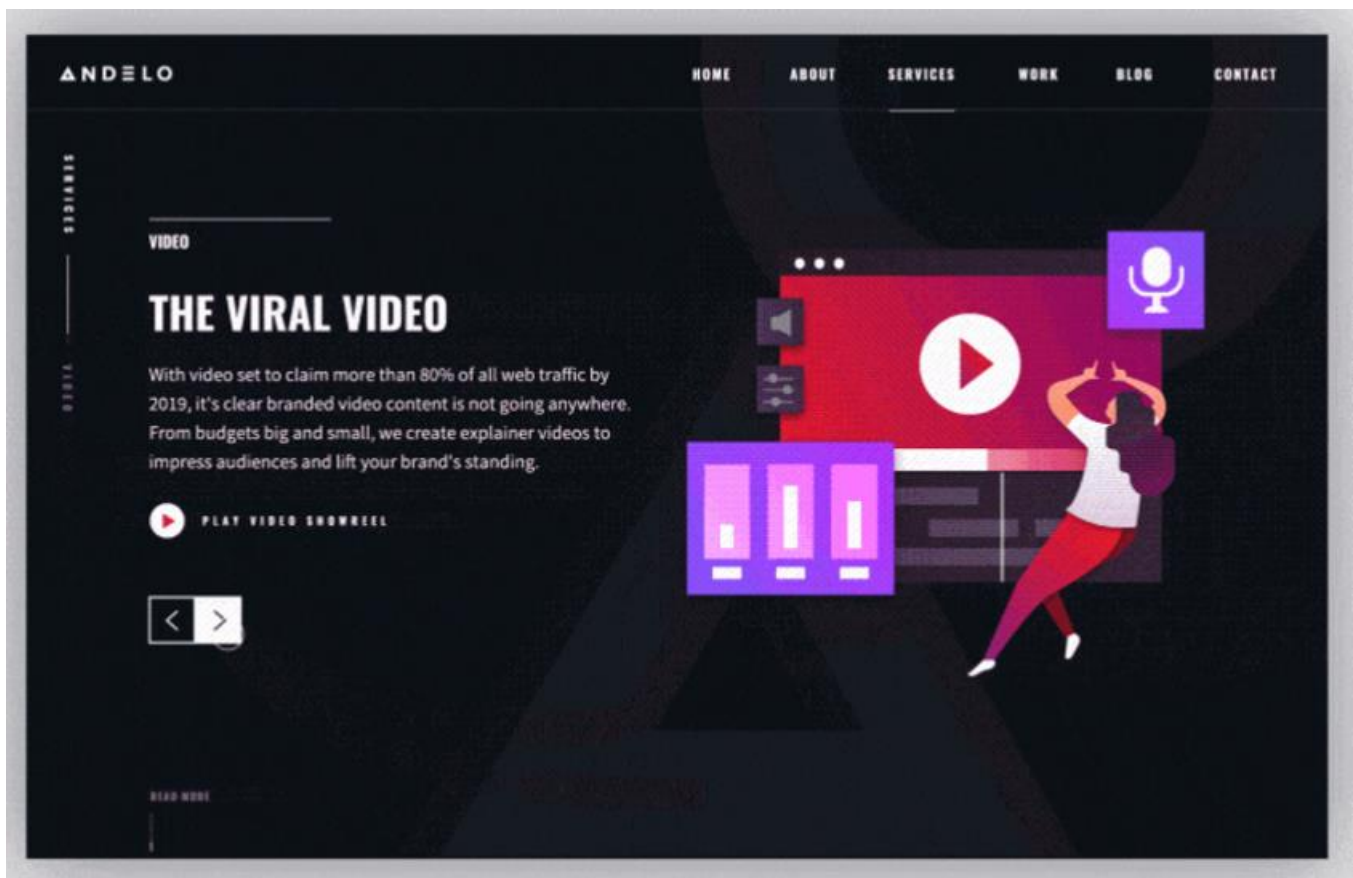


Module 3: Introduction to JavaScript

Video 3.4: Programming in JavaScript

Introduction

You now know HTML and CSS, which help you structure a web page and style it beautifully. One thing remains: making the page interactive. You know that JavaScript helps with that. Facebook, Amazon, and Google all rely on JavaScript to provide a seamless user experience. Therefore, you need to understand how JavaScript works.



What Is Programming?

Essentially, programming is how you tell machines what to do and when in order to perform tasks. A task can be something simple, like calculating the sum of two numbers, or something complex, like rendering a web page.

Programming as a Recipe

Programming is similar to cooking. A recipe for a chocolate cake might have steps, such as mixing the ingredients, setting the oven temperature, and baking. In a program, each statement is a step.

- `var ingredients = [{name: "flour", quantity: 100}, {name: "cocoa": quantity: 20}];` could represent gathering your ingredients in a cooking recipe.
- `var flour = ingredients.find((ingredient) => ingredient.name === 'flour');` represents finding the ingredient named flour.
- `flour.quantity = flour.quantity - 5;` represents adding five units of flour to the mix and subtracting those units from the inventory.



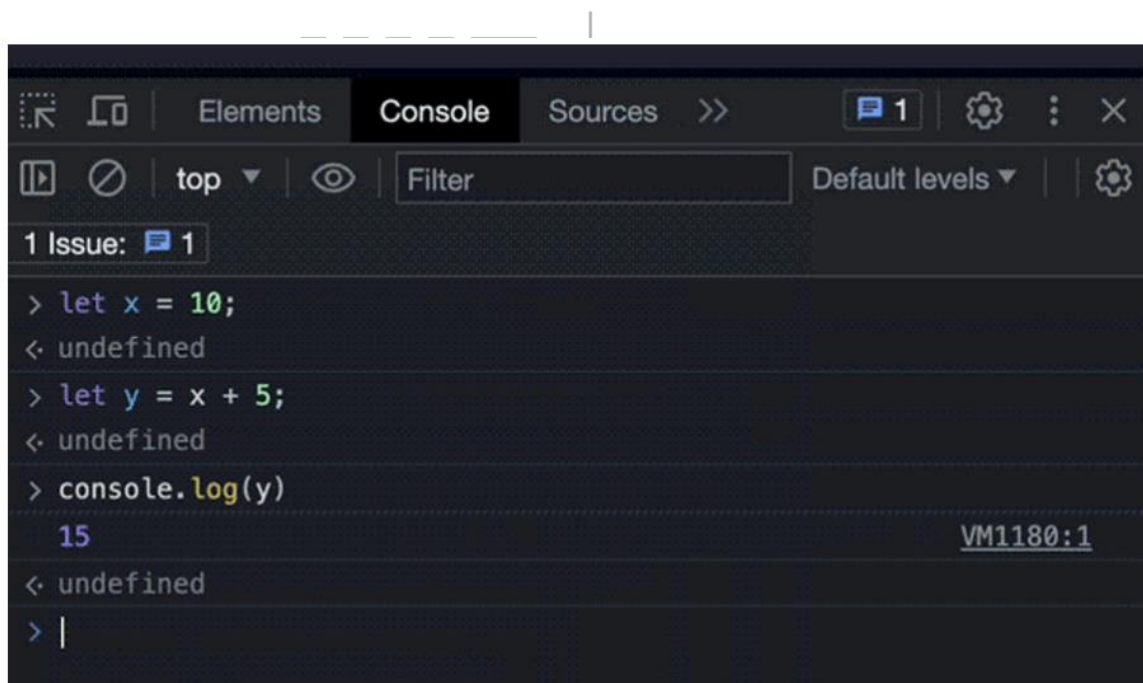
Statements in JavaScript

A statement in JavaScript is equivalent to a single step or instruction in a recipe. Common JavaScript statements might include variable assignments such as `var x = 10;` or function calls such as `console.log("Hello, world");` The semicolon at the end of a statement is similar to a period at the end of a sentence; it signifies the end of a complete thought or action.

Execution of Statements

Code statements in JavaScript are executed in sequence, similar to how you would follow the steps in a recipe in a specific order.

Here's an example code:

A screenshot of a web browser's developer console, specifically the 'Console' tab. The console shows a sequence of JavaScript commands and their outputs. The first command is `> let x = 10;`, which returns `< undefined`. The second command is `> let y = x + 5;`, which also returns `< undefined`. The third command is `> console.log(y)`, which returns `15`. The console interface includes a toolbar with icons for opening the console, pausing, and filtering, as well as a 'Filter' input field and 'Default levels' dropdown. The output '15' is highlighted in blue, and the source location 'VM1180:1' is visible on the right side of the output line.

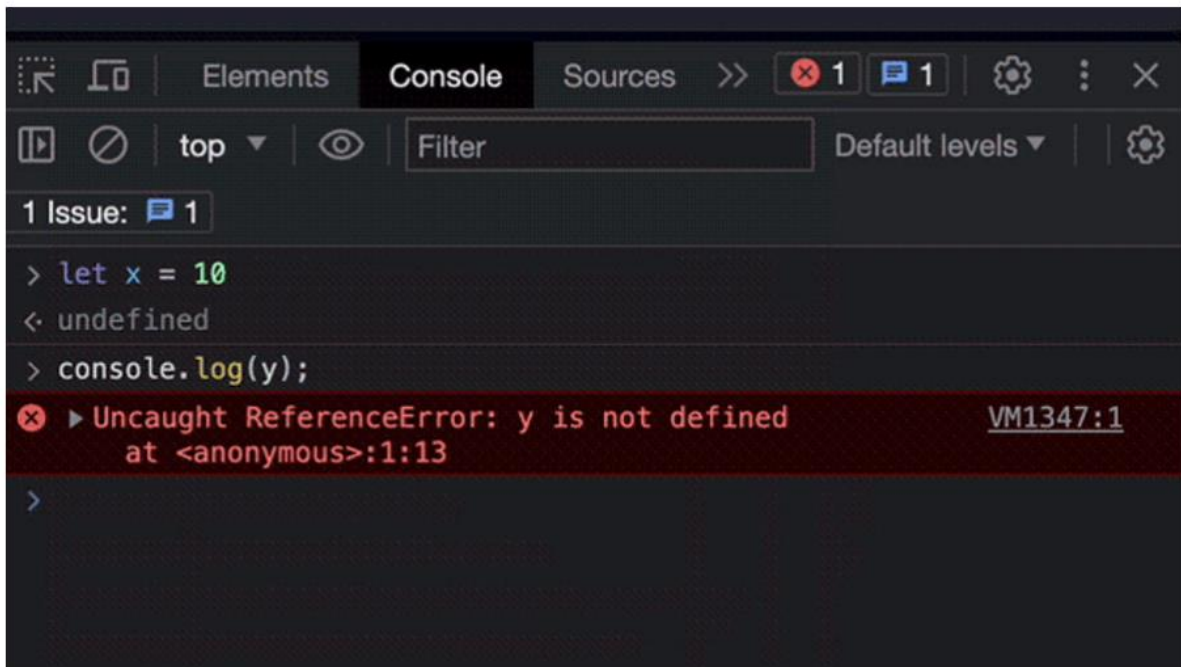
```
> let x = 10;
< undefined

> let y = x + 5;
< undefined

> console.log(y)
15
< undefined

> |
```

If you change the order of the statements, you get an error or an undesired result.



Control Structures



Say there's an option to add some nuts to your cake. You can choose to add them or you can skip that step and proceed. Similarly, in programming, some statements are either skipped or executed based on conditions. These statements are inside "conditionals," which make up one of the control structures. Loops are examples of a control structure that allows you to change the flow of execution.

The Core of Programming

Programs have four programming pillars. Consider a login page:

- **Input:** A user types their email and password.
- **Process:** The program checks if the credentials are correct.
- **Output:** A welcome message or an error message is displayed.
- **Storage:** User session data might be saved for use in future interactions.

Example:

```
var email = prompt("Enter email"); // Input
var password = prompt("Enter password"); // Input
if (email === "user@email.com" && password === "1234") { //
  Process
    alert("Login successful"); // Output
    sessionStorage.setItem("user", email); // Storage
}
```

Summary

- Programming, particularly in JavaScript, consists of writing a series of statements that are executed in sequence or as specified by the control structures.
- This is similar to a recipe: you gather the ingredients (input), perform the steps (process), taste the food (output), and maybe store some for later (storage).