Introduction to JavaScript

Software Development Bootcamp

Topic

Variables

What Are Variables?

Variables are names for values in JavaScript.

- Used anywhere a value or expression is used
- Assigned using the assignment operator (=)
- Allow us to store and manipulate data

Why Use Variables?

Variables are essential for dynamic programming:

- Store and update information
- Make code more readable and maintainable
- Allow for reuse of values throughout your code

What Is Declaring A Variable?

Declaring a variable is like creating a labeled container:

- It tells JavaScript to reserve memory for a value
- It defines the name (label) we'll use to access that value
- It specifies how that value can be used (let, const)

Variable Declaration

Three key aspects of variable declaration:

- **Keyword**: let, const
- **Name**: A unique identifier for the variable
- **Optional value**: Assigning an initial value

```
let message = 'Hello, world!';
```

`let` Keyword

let creates variables that can be reassigned:

- Good for values that may change
- Helps prevent unintended variable overwriting

```
// Declaration of the message variable with the value 'Hello, world!'
let message = 'Hello, world!';
console.log(message) // Output is 'Hello, world!'

// Redefining the message variable
message = 'Hello JavaScript!';
console.log(message) // Output is 'Hello JavaScript!'
```

`const` Keyword

const creates variables that cannot be reassigned:

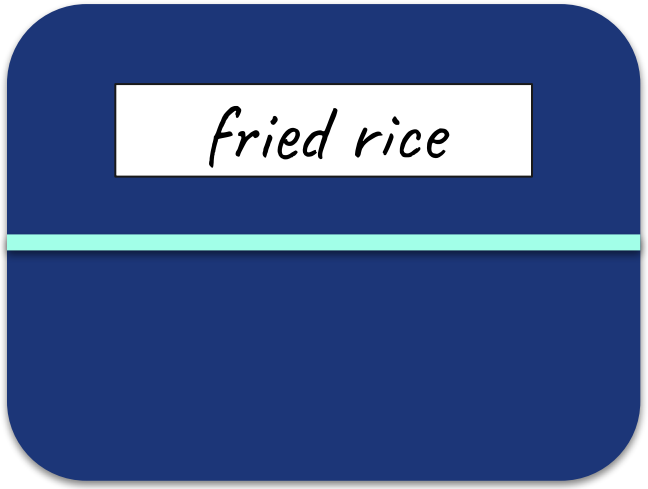
- Must be initialized when declared
- Good for values that should not change

```
// Declaration of the message variable with the value 'Hello, world!'
const message = 'Hello, world!';
console.log(message) // Output is 'Hello, world!'
// Redefining the message variable
message = 'Hello JavaScript!';
console.log(message) // Error: Assignment to constant variable
```

Labelling Leftovers

When you label your leftovers, you are creating a key/value pair.

Just like a variable.



Topic

Manipulating Data

What Are Data Types?

Data types are categories for different kinds of values in JavaScript:

- They define what operations can be performed on the data
- They determine how the data is stored in memory
- They help prevent errors by ensuring proper data handling

Why Use Data Types?

Data types are fundamental to programming languages for several reasons:

- They provide structure and meaning to data
- They determine how the computer interprets and processes information
- They help prevent errors and unexpected behavior in code

JavaScript's Data Types

Javascript has a number of data types, here are three to get you started.

Number: For numeric values (e.g., 42, 3.14)
String: For text (e.g., "Hello, world!")
Boolean: For true/false values

String Indexing

String indexing allows us to access individual characters in a string:

- Each character in a string has a numeric position (index)
- Indexing starts at 0 for the first character
- We can use square brackets [] to access characters by their index

```
let myString = 'Hello'
console.log(myString[0])
// output is 'H'
```

String Methods

String methods are built-in functions in JavaScript that can be used to manipulate and work with strings:

- They allow us to perform operations on strings
- They return new strings (strings are immutable in JavaScript)
- They provide powerful tools for text processing

Common String Methods

Here are some frequently used string methods in JavaScript:

- **length:** Returns the length of a string
- **toUpperCase():** Converts a string to uppercase
- **toLowerCase():** Converts a string to lowercase
- **trim():** Removes whitespace from both ends of a string
- **slice():** Extracts a portion of a string

String Methods In Action

Using String Methods helps you:

- Write more concise and readable code
- Efficiently manipulate text data
- Solve complex string related problems

```
let message = "  Hello, World!  ";
console.log(message.length);
// Output: 16
console.log(message.trim().toUpperCase
());
// Output: "HELLO, WORLD!"
console.log(message.slice(2, 7));
// Output: "Hello"
```

From Data Types to Operators

Data types and operators work together to manipulate data:

- **Data types** define what kind of information we have
- **Operators** define what we can do with that information

What Are Operators?

Operators are symbols that tell JavaScript to perform specific operations:

- Arithmetic operators: +, -, *, /, %
- Assignment operators: =, +=, -=, *=, /=
- Comparison operators: ==, ===, !=, !==, >, <, >=, <=
- Logical operators: &&, ||, !

Why Use Operators?

Operators allow us to:

- Perform calculations
- Assign and modify values
- Compare values
- Combine logical conditions

Building Blocks of Data Manipulation

- **Operators** act on data
- **Expressions** use operators to produce values
- **Statements** use expressions to perform actions

The Flow of Manipulation

1. Operators work on individual pieces of data
2. Expressions combine operators and values
3. Statements use expressions to change program state

Expression Example

- `'Hello, '` is a string
- `'world'` is another string
- The `+ operator`, when used with strings, concatenates them.

```
// string expressions
'Hello, ' + 'world'
// resolves to "Hello, world"
```

Statement Example

- **Declaration**: Creates a new variable 'age' and initializes it
- **Assignment**: Changes the value of the existing variable 'age'

```
// declaration statements
let age = 23
// assignment statements
age = 45
```

Expressions & Statements

- Write more efficient and readable code
- Debug issues more effectively
- Structure your programs logically

Manipulating Data

Data manipulation involves four key components:

- **Data Types**: Define the nature of the data
- **Operators:** Perform actions on the data
- **Expressions**: Combine data and operators to produce values
- **Statements**: Use expressions to perform actions

Data Types provide the foundation

- Numbers, strings, booleans, etc.

Operators act on data of specific types

- Arithmetic (+, -), comparison (>, <), etc.

Expressions use operators and data to create values

- `"Hello" + name`

Statements use expressions to perform actions

- `let x = 5 + 3`

Exercises

Combining Data Types