

```
    }, m => {  
      lit(" ")  
      {  
        ect":  
        {  
          ents.has(a[1])){  
            send("connected");  
            id = a[1];  
            {  
              id = a[1]  
              ents.set(a[1], {client: {position:  
                send("connected")  
            }  
            id = Math.random().toString().slice  
            = id;  
            ts.set(id, {client: {position: {  
              }  
            }  
          }  
        }  
      }  
    }  
  }  
}
```

# Introduction to JavaScript

JavaScript is a powerful programming language that has revolutionized the way we interact with the web. It enables developers to add dynamic functionality, interactivity, and animations to websites, making browsing experiences more engaging and user-friendly.

A large, stylized 'JS' logo is positioned on the left side of the slide. The 'J' is a solid dark grey, and the 'S' is a dark grey outline. The background of the slide is a dark grey-blue, and the left edge features a vertical yellow bar.

# What is JavaScript?

## 1 Client-Side Scripting

JavaScript is primarily used as a client-side scripting language, running in web browsers to enhance the user experience.

## 2 Versatile Language

In addition to web development, JavaScript can also be used for server-side programming, mobile app development, and more.

## 3 Interpreted Language

JavaScript is an interpreted language, meaning it is executed line by line without the need for compilation.

# JavaScript in the Browser

## DOM Manipulation

JavaScript allows developers to dynamically access and modify the Document Object Model (DOM), enabling changes to web page content and structure.

## Event Handling

JavaScript handles user interactions, such as clicks, mouse movements, and form submissions, allowing for responsive and dynamic user experiences.

## Web APIs

JavaScript can leverage various Web APIs, such as the Geolocation API, the Canvas API, and the Web Storage API, to add advanced features to web applications.

# JavaScript Syntax

## Variables

JavaScript uses keywords like `var`, `let`, and `const` to declare variables, which can store different data types.

## Control Flow

JavaScript uses conditional statements (if-else, switch), loops (for, while, do-while), and other control flow structures to execute code based on various conditions.

## Functions

Functions in JavaScript are first-class citizens, meaning they can be assigned to variables, passed as arguments, and returned from other functions.

## Objects

JavaScript is an object-oriented language, allowing developers to create and manipulate complex data structures using objects, properties, and methods.

test ECMAScript standard defines nine


ives

ng  
er

nt  
ean  
fined  
ol

x  
-  
ct  
tion



Made with 

@flowforfrank

# Variables and Data Types

## 1 Primitive Types

JavaScript has several built-in primitive data types, including numbers, strings, booleans, null, and undefined.

## 2 Dynamic Typing

JavaScript is a dynamically typed language, meaning variables can hold any type of data, and types can change at runtime.

## 3 Variable Scope

The scope of a variable in JavaScript is determined by where it is declared, affecting its accessibility and lifetime.

# Operators and Expressions



## Arithmetic

JavaScript supports various arithmetic operators, such as `+`, `-`, `*`, `/`, and `%`.



## Assignment

Assignment operators, like `=`, `+=`, `-=`, allow you to assign values to variables.



## Comparison

Comparison operators, like `==`, `!=`, `>`, `<`, `>=`, `<=`, enable logical comparisons.



## Logical

Logical operators, such as `&&`, `||`, and `!`, combine or negate boolean expressions.

if condition  
is true

is False

**Code block**

# Control Flow and Conditional Statements

1

## If-Else

The if-else statement allows you to execute different code blocks based on a condition.

2

## Switch

The switch statement enables you to execute different code blocks based on multiple conditions.

3

## Loops

Loops, like for, while, and do-while, allow you to repeatedly execute a block of code.

# Functions

Function Declaration

Defines a named function, using the ``function`` keyword.

Function Expression

Assigns an anonymous function to a variable.

Arrow Functions

Provides a concise syntax for defining functions.

Parameters and Arguments

Functions can accept input values (parameters) and use them in their code.

Return Values

Functions can return values, which can be used elsewhere in the code.