JavaScript Variables and Types

USING VARIABLES, LITERALS, AND ASSIGNMENTS

Barry Luijbregts
SOFTWARE ARCHITECT & DEVELOPER

@AzureBarry www.azurebarry.com

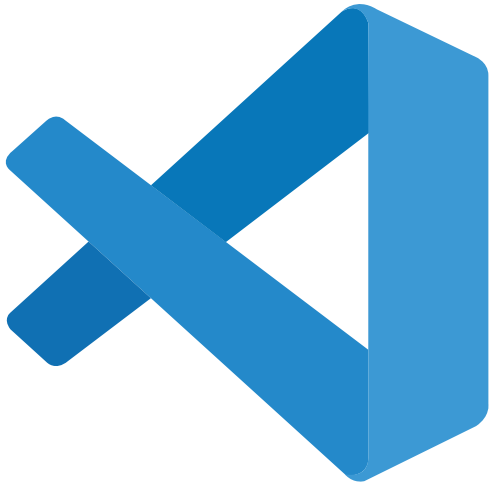
Introduction

How this course works

Demos, demos, demos...

How This Course Works

How This Course Works



Visual Studio Code

https://code.visualstudio.com

Basic knowledge of **VS Code** and **JavaScript web development**

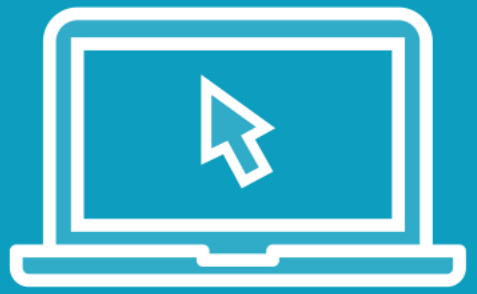
Demo code

Demos, demos, demos

Demo Code

https://github.com/bmaluijb/GetYourLoanApp

Demo



Using template literals

Using tagged template literals

Difference between let and const

Destructuring syntax

The Difference Between Let and Const

Block Scope

```
function MyFunction() {
        var x = 10;

        if (true) {
            var x = "Hello";
        }

        // x is "Hello" here
}
```

Block Scope

```
function MyFunction() {
        var x = 10;

        if (true) {
            let x = "Hello";
            // x is “Hello” here
        }

        // x is 10 here
}
```

Block Scope

```
function MyFunction() {
        var x = 10;

        if (true) {
            const x = "Hello";
            // x is “Hello” here
        }

        // x is 10 here
}
```

Block Scope

```
function MyFunction() {


        if (true) {
            const x = "Hello";
            // x is “Hello” here
        }

        console.log(x. length);
        // result in an error
}
```

Redeclaring Variables

```
function MyFunction() {

        var y = 2;
        let  y = 4; // not allowed

        if(true){
            var y = 4;
            let  y = 2; // not allowed
        }
}
```

Redeclaring Variables

```
function MyFunction() {

        let  y = 2;
        let  y = 4; // not allowed

        if(true){
            let  y = 4;
            let  y = 2; // not allowed
        }
}
```

Redeclaring Variables

```
function MyFunction() {

        if(true){
            let  y = 3; // allowed
        }
        if(true){
            let  y = 4; // allowed
        }
        if(true){
            let  y = 6; // allowed
            y = “Hello”;
        }
}
```

The Const Keyword

```
const  arr = [3, 4, 5];

arr = 3; // results in an error

arr = “Hello”; // results in an error

arr = null; // results in an error

arr[0] = 22; // allowed

var  arr = Object.freeze([3, 4, 5]);
```

Var vs. Let vs. Const

var

- No block scope
- Can be redeclared anywhere
- Can be used and reassigned anywhere

let

- Block scope
- Can **NOT** be redeclared within scope
- Can be reassigned within scope

const

- Block scope
- Can **NOT** be reassigned or redeclared.
- The value it references **CAN** be changed

Summary

Using template literals

- `Dear ${name}`
- Multiline

Using tagged template literals

- highlightText `Dear ${name}
- function highlighText(strings, ...values){}
- String.raw
- strings.raw

let and const

- let and const provide Block Scope variables
- let can be reassigned but not redeclared
- const can’t be reassigned or redeclared

Destructuring syntax

- var [a, b, c] = array;
- var {Id : a, Name: b} = object;

Where to Go Next

MDN web docs

- https://developer.mozilla.org/

W3Schools

- https://www.w3schools.com/js/

ES6: The Right Parts (Kyle Simpson)

- https://app.pluralsight.com/library/courses/es6-the-right-parts/