Variáveis no JavaScript

Quando estávamos estudando portugol vimos que as variáveis são usadas para criarmos referências na memória, nelas atribuímos valores e fazemos manipulações de dados a partir delas, no JavaScript é exatamente a mesma coisa, o que muda é apenas como criamos as variáveis, temos 3 palavrinhas chave para a criação de variáveis, num primeiro momento será um pouco confuso e é 100% normal que seja, fique tranquilo se não entender de primeira vista.

---

var

Essa é a maneira mais "antiga" de criar variáveis no JavaScript, "var" vem de "variable", ou seja, variável mesmo como podemos deduzir, não há segredo nenhum em sua sintaxe, veremos exemplos a seguir:

```
var variavel1;
variavel1 = 'algum valor';


var variavel2 = 'algum valor';
```

Nas 2 primeiras linhas nós declaramos a variável, em seguida atribuímos um valor a essa variável, temos essa possibilidade com o var, na última linha já fizemos tudo de uma vez, a declaração e a atribuição de um valor, caso necessário podemos "reatribuir" outro valor nessa variável criada com a palavra var:

```
var variavel2 = 'algum valor';
variavel2 = 'outro valor agora';
```

E a partir disso a variavel2 agora carrega o valor 'outro valor agora'.

Quando declaramos uma variável com var ela tem 2 situações:

- Quando declaramos ela no escopo global;
- Quando declaramos ela no escopo de função;

Como assim escopo global? Escopo de função?

O escopo global é o que está fora de alguma função ou bloco de código como loops por exemplo, funções (veremos em breve) tem um escopo delimitado por { chaves }, dentro dessas chaves é o escopo da função, portanto se usarmos var para declarar uma variável dentro de uma função aquela variável só vale dentro daquela função, porém se declaramos no escopo global essa variável vale em qualquer local do código.

```
// Declaração da função no escopo global (fora de uma função ou algum outro escopo)
var nome = "Lucas";

// Declaração da função
function imprimeNome() {
    // Aqui entre as chaves é o escopo da função
    console.log(`Meu nome é: ${nome}`);
}

// A chamada da função criada anteriormente
imprimeNome();
```

Agora outra situação é a seguinte:

```
function imprimeNome() {
    // Declaração da variável dentro do escopo da função
    var nome = "Lucas";
}

// Chamada da função
imprimeNome();

// Tentativa de uso da função, porém com erro
// pois a variável que estamos tentando usar aqui
// só vale dentro do escopo em que foi definida
console.log(`Meu nome é: ${nome}`);
```

O erro que recebemos é **ReferenceError: nome is not defined**, pois a variável nome foi definida dentro do escopo da função.

---

let

A **let** declara variáveis que "funcionam" dentro de um escopo de bloco ou expressão na qual é usada. Ao contrário da variável var, ela não declara uma variável globalmente ou localmente para uma função inteira, independente do escopo do bloco em si. Com ela não criamos uma propriedade no objeto global, por exemplo:

```
let nome1 = 'Clóvis'
var nome2 = 'Clóvis'

console.log(this.nome1);
// Retorna: undefined
console.log(this.nome2);
// Retorna: Clóvis
```

Perceba que temos diferentes resultados, pois, como dito, a let não cria uma propriedade no objeto global assim como a var.

---

const

Usar **const** para declarar variáveis é o mais recomendado, pois após declarado e atribuído um valor, esse valor não pode ser modificado, ao contrário de **var** e **let** a **const** não pode ser iniciada sem a atribuição de um valor.

```
const nome = "Lucas";
```

O seu escopo é idêntico ao de **let**, sempre que possível utilize **const** em seus projetos.