Javascript

Variáveis, Constantes,
Tipos de Dados e Conversões

Variáveis

Uma variável em Javascript pode ser declarada com a instrução **let**, conforme trecho de código abaixo:

```
let mensagem;

mensagem = "Olá mundo!";
```

Também é possível inicializar a variável na mesma linha da declaração:

```
let mensagem = "Olá mundo!";
```

Variáveis

- Restrições de nomes:
  - O nome de uma variável pode conter letras, números e os caracteres $ e _.
  - O nome de uma variável não pode ser iniciado por um dígito numérico.

- Uma variável precisa ser declarada com let antes de ser utilizada.

- O escopo de uma variável declarada com let é o do bloco de comando em que a mesma foi declarada.

Variáveis

- Existe uma forma obsoleta de declaração de variável que utiliza a instrução **var**, porém ao declarar uma variável com **var** a mesma **não terá escopo de bloco**, ou seja, uma variável declarada dentro de um if poderá ser utilizada fora do mesmo.
- Variáveis declaradas com var também podem ser declaradas após a sua utilização e isso acontece devido do Javascript deslocar a declaração de todas as variáveis para o início.
- Devido a estes problemas, **evite ao máximo declarar variáveis com var**.

Constantes

Uma constante em Javascript pode ser declarada com a instrução const, conforme trecho de código abaixo:

```
const mensagem = "Olá mundo!";
```

Quando a constante é utilizada como apelido para facilitar a lembrança de valores difíceis de serem guardados, por convenção, ela deve ser nomeada em letras maiúsculas e cada palavra deve ser separada pelo caractere _. Exemplo:

```
const COR_AZUL = "#00F";
```

Tipos de Dados

- Em Javascript temos os seguintes tipos de dados:
  - String
  - Number
  - BigInt
  - Boolean
  - null
  - undefined
  - Object
  - Symbol

Tipos de Dados

- String
  - O tipo string em Javascript é adequado para armazenar cadeias de caracteres.
  - Strings em Javascript precisam estar entre aspas
    - Existem 3 tipos de aspas em Javascript:

```
const mensagem = "Olá mundo!";
```

```
const mensagem = 'Olá mundo!';
```

```
const mensagem = `Olá mundo!`;
```

  - Não existem diferenças entre aspas simples e aspas duplas, porém o terceiro tipo permite a interpolação de expressões na string, sendo que para isso é necessário utilizar a expressão entre chaves posicionadas após o caractere $.

Tipos de Dados

- Exemplo de interpolação de strings:

```
const x = 122;
const mensagem = `Testando ${x+1}`; //Conterá o valor: Testando 123
```

Tipos de Dados

- Number
  - O tipo number representa números que podem ser inteiros ou ponto flutuante.
  - Existem os operadores +, -, * e /
  - Existem valores especiais que este tipo de dados pode gerar:
    - Infinity
      - Valor que é maior que qualquer número
    - -Infinity
      - Valor que é menor que qualquer número
    - NaN
      - Resultado de uma operação matemática incorreta.
      - A presença de qualquer valor NaN em uma expressão fará que o valor resultante da expressão também seja NaN.

```
const numero = "Dez" / 2; //Conterá o valor: NaN
```

Tipos de Dados

- BigInt
  - O tipo bigint deve ser utilizado quando é necessário representar inteiros muito grandes de forma precisa.
  - A letra “n” após uma constante indica que a mesma é do tipo bigint.
  - Exemplo de declaração de constante do tipo bigint:

```
const numero = 2345678901234456n;
```

Tipos de Dados

- Boolean
  - O tipo boolean deve ser utilizado quando é necessário armazenar somente um dos seguintes valores: true ou false.

```
const enviarCopia = true;
const sairSemSalvar = false;
```

- Null
  - O tipo null que representa “nada”, “vazio” ou “valor desconhecido”.

```
const obj = null;
```

Tipos de Dados

- Undefined
  - O tipo undefined que representa que o valor de uma variável não foi inicializado.

```
let variavel; //undefined
```

- Symbol
  - Tipo utilizado para criar identificadores únicos para objetos.

Tipos de Dados

- Object
  - Tipo de dado que representa um objeto, que pode ser utilizado para representar entidades mais complexas.

```
let produto = { descricao: "Suco X", preco: 10.67 };
```

- Objetos podem ter a suas propriedades alteradas de duas formas:

```
produto.descricao = "Suco XX";
produto.["preco"] = 16.7;
```

typeof

- A função typeof permite conhecer o tipo de uma variável.
  - Exemplo 1:

```
let variavel;
variavel = "Olá mundo!";
alert(typeof(variavel));
```

typeof

- A função typeof permite conhecer o tipo de uma variável.
  - Exemplo 2:

```
let variavel;
variavel = 123;
alert(typeof(variavel));
```

typeof

- A função typeof permite conhecer o tipo de uma variável.
  - Exemplo 3:

```
let variavel;
variavel = true;
alert(typeof(variavel));
```

Conversão de Tipos Primitivos

- Para fazer a conversão de tipo, basta utilizar as funções:
  - String
    - Para converter o tipo passado por parâmetro para o tipo string.
  - Number
    - Para converter o tipo passado por parâmetro para o tipo number.
  - Boolean
    - Para converter o tipo passado por parâmetro para o tipo boolean.

Conversão de Tipos Primitivos

- Exemplo de quando deve ser feita a conversão:

```
let numero1 = "5";
let numero2 = 2;
let soma = numero1 + numero2;
alert(`${soma} -> ${typeof(soma)}`);
```

Conversão de Tipos Primitivos

- Exemplo de quando deve ser feita a conversão:

```
let numero1 = "5";
let numero2 = 2;
let soma = Number(numero1) + numero2;
alert(`${soma} -> ${typeof(soma)}`);
```

Conversão Numérica

- Valores que resultam em `NaN`:
  - undefined
  - "abc123"
  - "123abc"
  - "teste"
- Valores que resultam em `0`:
  - null
  - false
  - "" (string vazia)
  - " " (espaços)
- Valores que resultam em `1`:
  - true
- Valores que resultam em `123`:
  - " 123 "
  - " 123"
  - "123 "

Fim de material

Dúvidas?