Introdução

Primeiros Passos

- criar uma pasta
- dentro da pasta criar um arquivo **.html**

Comandos Iniciais

- todos os comandos do JS deve ser rodado dentro da tag **<script>**
- a tag **<script>** deve ficar dentro de **<body>**
- JS não mais exige '**;' no final para fechar o comando**
- abrir uma janela de alerta:

```
window.alert('mensagem')
```

- abre uma janela de confirmação:

```
window.confirm('mensagem')
```

- abre uma janela de input:

```
window.prompt('Qual é seu nome?')
```

Comentários

- uma linha ⇒ usa-se //
- mais de uma linha ⇒ /* */

Declaração de variáveis

- **var** nome_da_variável = valor_da_variável
- também pode ser usado **let** e **const**
- para string podem ser usadas aspas simples, duplas ou crase
- podem começar com letra, $ ou _
- não podem começar com números
- é possível usar acentos e símbolos
- não podem conter espaços ou palavras reservadas
- variáveis de números inteiros e com casas decimais, são do tipo **number**
- tipos de **number**
  - Infinity
  - NaN ⇒ Not a Number
- para saber o tipo da variável:
  - **typeof** variável ou valor
- **null** é uma variável do tipo objeto no JS

Tratamento de Dados

- o comando window.prompt('Digite algo'), retorna sempre uma string
- para converter em números, podemos utilizar:
  - Number.parseint(n): para converter em inteiros
  - Number.parsefloat(n): para converter em reais
  - Number(n): o JS decide qual o tipo de número, automaticamente
  - lembrando que são comandos case sensitives
- convertendo de números para string:

- String(n)
- n.toString()

Formatando strings

```
var curso = 'JavaScript'
'Estou aprendendo' + curso // usando concatenação
`Estou aprendendo ${curso}` // usando template string
```

Comandos de strings

- string.length ⇒ tamanho da string
- string.toUpperCase(): tudo em maiúsculo
- string.toLowerCase(): tudo em minúsculo

Escrever na tela com template strings

- document.write()

```
<script>
    var nome = window.prompt('Digite seu nome: ')
    document.write(`${nome}, possui ${nome.length} letras.`)
  </script>
```

- também é possível mesclar com elementos html

```
<script>
    var nome = window.prompt('Digite seu nome: ')
    document.write(`Olá <strong>${nome}</strong>, seu nome possui ${nome.length} letras.</br>`)
    document.write(`Seu nome em maiúsculo é <strong>${nome.toUpperCase()}</strong>.</br>`)
    document.write(`Em minúsculo é <strong>${nome.toLowerCase()}</strong>.`)
  </script>
```

Formatação de números e padrão financeiro

- var_núm.toFixed(núm_de_casa_decimais)
- var_núm.replace('.', ',') ⇒ troca o ponto por vírgula

```
var n1 = 1455.5
n1.toFixed(2) => '1455.50'
n1.toFixed(2).replace('.', ',') => '1455,50'
```

- passando a string para moeda local

```
var n1 = 1455.5
n1.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'})
=> 'R$ 1.455,50'
```

Tipos de Operadores

Operadores de Incremento

- x = x * 1 ⇒ x *= 1
- x = x / 1 ⇒ x /= 1
- x = x ** 1 ⇒ x ** 1
- x = x + 1 ⇒ x += 1 ⇒ x ++ (incrementa 1)
- x = x - 1 ⇒ x -= 1 ⇒ x— (decrementa 1)

Operador de Identidade

- no JavaScript a igualdade:
  - 5 == '5' ⇒ True
- isso por que a linguagem não testa o tipo, mas só o caracter em si
- para testar o valor e o tipo existe um operador chamado:
  - **operador de identidade** ou
  - **operador de igualdade restrita**
- esse operador é representado por 3 sinais de igual '==='
- nesse caso 5 === '5' ⇒ False
- esse tipo de operador também serve para desigualdade
- representado por exclamação e dois sinais de igual ' ! = = '
- chamado de desigual restrito
- nesse caso 5 ≠= '5' ⇒ True

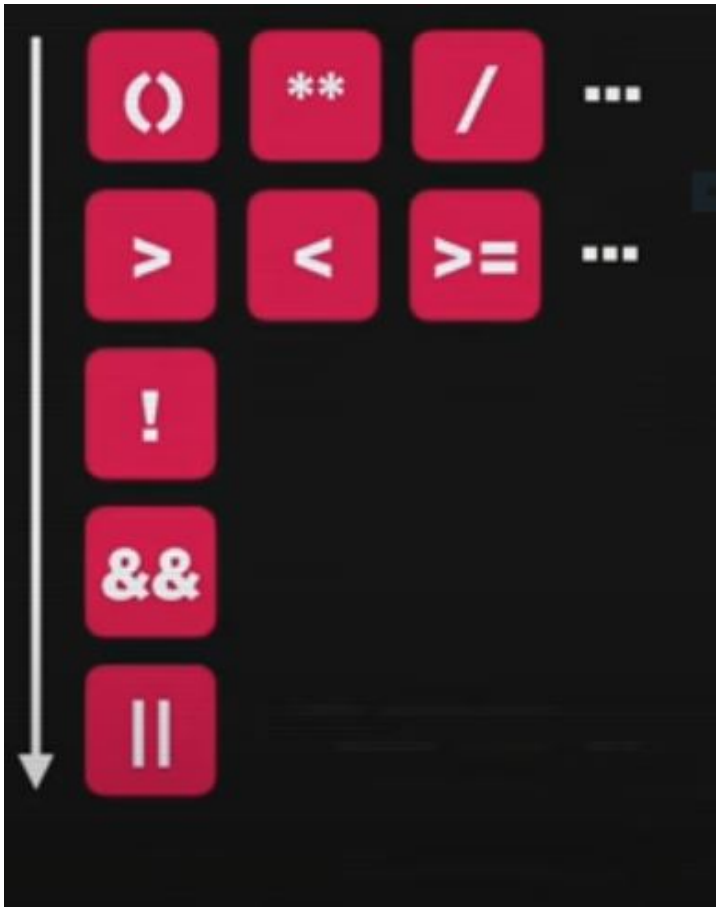
Operadores Lógicos

- ! ⇒ **negação**

    - operador unário, só tem um operando
- && ⇒ **conjunção** (e)
    - operador binário, dois valores lógicos, um de cada lado
- || ⇒ **disjunção** (ou)
    - operador binário, dois valores lógicos, um de cada lado

Ordem de precedência

- entre diferentes tipos de operadores a ordem é:
    - parênteses ⇒ aritmético ⇒ relacional ⇒ lógico
- caso haja mais de um operador lógico em uma mesma expressão, a ordem de execução é:
    - **não** ⇒ **e** ⇒ **ou**
    - ! ⇒ && ⇒ ||



Operador Ternário

- possui 3 partes
- possui os símbolos '?' e ':' na mesma expressão
- expressão composta por:
  - teste ? true : false
- pega o primeiro teste e aplica uma ação caso seja verdadeiro, ou outra ação caso seja falso
- Ex:
  - **média ≥ 7.0** ? **"Aprovado"** : **"Reprovado"**