**TypeScript parte I – Evoluindo seu JavaScript**

1. **Aula 1 – Porque usar TypeScript:**
   1. A primeira coisa quando vamos codar em typescript é fazer a preparação do ambiente, ou seja, instalar node.js e depois dar um npm install no terminal dentro da pasta do projeto para instalar as dependências do server para rodar nossa aplicação.
      1. Npm run server: Starta o server quando rodar o comando no terminal dentro da pasta do projeto.
      2. No meu caso estou usando a extensão live server do VScode pois não funcionou de jeito nenhum.
   2. Negociação, modelagem e regras:
      1. #: variável privada, não se altera.
      2. Criamos uma classe de negociação para nossa aplicação onde possui as variáveis de data, quantidade e valor privadas com um # antes delas e um construtor que atribui esses valores ao ser criado.
      3. Teoricamente não poderíamos conseguir alterar os valores dessas variáveis diretamente, então fizemos o teste com uma atribuição direta fora do construtor.
      4. Ao fazer um console.log do antes e depois dessa atribuição direta, notamos que ao invés de alterar o valor da variável privada, ele criou uma nova variável com o valor novo atribuído:

// Módulo da negociação

export class Negociacao {

  #data;

  #quantidade;

  #valor;

  constructor(data, quantidade, valor) {

    this.#data = data;

    this.#quantidade = quantidade;

    this.#valor = valor;

  }

}

// Módulo do APP

import { Negociacao } from "./models/negociacao.js";

const negociacao = new Negociacao(new Date(), 10, 100);

console.log(negociacao)

negociacao.quantidade = 1000

console.log(negociacao)



* + 1. O módulo do app é o que está vinculado a nossa aplicação web.
  1. Finalização do modelo:
     1. Lembrando que se quiser visualizar uma variável privada, não conseguimos mesmo que tentemos colocar o # antes dela num console.log:

console.log(negociacao.#data)



* + 1. Se tentarmos visualizar sem o # ele dá como Undefined.
    2. Portanto precisamos criar getters para essas propriedades para que possamos visualizar elas, embora ainda não poderemos fazer alterações diretamente utilizando elas, pois são getters e não setters.