

TS TypeScript – Simplificando Classes

TypeScript fornece um código de classe menos verboso que consiste em basicamente usar o constructor como “ponto central” da classe. Como assim?

Se eu declarar o modificador de acesso das minhas propriedades no construtor, eu não vou precisar referencia-las na minha classe, nem as atribuir no corpo do construtor, por baixo dos panos ele mesmo já vai fazer isso. O que **simplifica o código dessa maneira**:

```
export class Negociacao {  
  private data : Date;  
  private quantidade : number;  
  private valor : number;  
  private volume : number;  
  
  constructor(data : Date, quantidade : number, valor : number){  
    this.data = data;  
    this.quantidade = quantidade;  
    this.valor = valor;  
  }  
  
  getData(){  
    return this.data;  
  }  
}  
  
export class Negociacao {  
  constructor(  
    private data : Date,  
    private quantidade : number,  
    private valor : number  
  ){}  
  
  getData(){  
    return this.data;  
  }  
  
  getQuantidade(){  
    return this.quantidade  
  }  
}
```

Arrays e **ReadOnlyArrays** também podem ser simplificados:

```
export class Negociacoes {  
  private negociacoes : Array<Negociacao> = [];  
  
  lista(): ReadonlyArray<Negociacao>{  
    return this.negociacoes;  
  }  
}  
  
export class Negociacoes {  
  private readonly negociacoes : Negociacao[] = [];  
  
  lista(): readonly Negociacao[] {  
    return this.negociacoes;  
  }  
}
```

O **readOnly** consegue simplificar ainda mais algumas características da classe, por exemplo os getters, se eu transformar **meus atributos em public readOnly** eu **não preciso explicitamente declarar getters** e ainda sim eles vão manter o mesmo comportamento de só poderem ser vistos e não alterados:

```
export class Negociacao {  
  constructor(  
    private data : Date,  
    public readonly quantidade : number,  
    public readonly valor : number  
  ){}  
  
  getData() : Date {  
    const copiaData = new Date(this.data.getTime());  
    return this.data;  
  }  
  
  getVolume(){  
    return this.quantidade * this.valor;  
  }  
}  
  
export class Negociacao {  
  constructor(  
    private data : Date,  
    public readonly quantidade : number,  
    public readonly valor : number  
  ){}  
  
  getData(){  
    return this.data;  
  }  
  
  getVolume(){  
    return this.quantidade * this.valor;  
  }  
}
```

Alguns **atributos** ainda **precisam de uma lógica para funcionar corretamente**, como o **getter da Data** que devolve uma cópia, impedindo que o valor da data do objeto real seja modificado e o **getter de volume** que precisa de uma conta. Nesses casos podemos usar um getter padrão com código envolvido.