

## TS TypeScript – Enumerated

Imagina que eu tenha uma **regra de negócio** que me diz que devo apenas cadastrar negociações se as mesmas foram realizadas em um dia útil da semana. É bem simples fazer isso na verdade, basta eu usar os métodos de date que retornam essa informação e **fazer uma checagem antes de cadastrar**:

```
public adiciona(){
    const negociacao = this.criaNegociacao();

    if(!this.ehDiaUtil(negociacao.getData())){
        this.mensagemView.atualizar("Só podem ser cadastradas negociações que foram feitas em dias úteis")
        return ;
    }

    this.negociacoes.adiciona(negociacao)
    this.limparFormulario();
    this.atualizaView();
}

private ehDiaUtil(data: Date) : boolean {
    return data.getDay() > 0 && data.getDay() < 6
}
```

### Negociações

Só podem ser cadastradas negociações que foram feitas em dias úteis

Data  
19/06/2022

Quantidade  
2

Valor  
120

Incluir

Beleza, agora pensando em quesito de legibilidade e entendimento de negócio, quando eu vejo uma comparação do tipo "`getDay() > 0`" eu preciso saber que o `getDay()` retorna um número de 0 a 6 que representa um dia da semana.

Da pra melhorar isso com a utilização de **Enum**, **enums** são classes que contém valores imutáveis, como os dias da semana são valores imutáveis eu posso declarar um **enum de dias da semana** onde cada dia tem seu valor, a partir daí minhas comparações podem ficar mais legíveis e ainda fica a possibilidade de usar em qualquer lugar do projeto:

```
export enum DiasDaSemana {
    DOMINGO = 0,
    SEGUNDA = 1,
    TERCA = 2,
    QUARTA = 3,
    QUINTA = 4,
    SEXTA = 5,
    SABADO = 6
}
```

```
private ehDiaUtil(data: Date) : boolean {
    return data.getDay() > 0 && data.getDay() < 6
}
```

```
private ehDiaUtil(data: Date) : boolean {
    return data.getDay() > DiasDaSemana.DOMINGO && data.getDay() < DiasDaSemana.SABADO
}
```

## TS TypeScript – Parâmetros opcionais

Minhas views tem templates, esse meu **template é um HTML injetado via innerHTML**, e se eu quiser impedir que **esses templates conttenham scripts**? Eu posso **fazer isso filtrando a tag script e removendo ela**, show de bola (frameworks geralmente já fazem esse trabalho):

```
atualizar(model : T) : void {
    let template = this.template(model);
    if(this.removerTagScript){
        template = template.replace('<script>[\s\S]*?</script>', '')
    }
    this.elemento.innerHTML = template;
}
```

Mas pode ser que eu também não queira remover, então vai ser um recurso **opcional** da minha View, e isso é possível graças a um recurso do TypeScript que é utilizar **"?" no parametro que pode ou não ser opcional**, ***\*\*parâmetros opcionais apenas podem ser tratados como opcionais se forem os ultimos parametros\*\****:

```
constructor(seletor : string, remover? : boolean){  
  this.elemento = document.querySelector(seletor);  
  if(remover){  
    this.removerTagScript = remover;  
  }  
}
```