

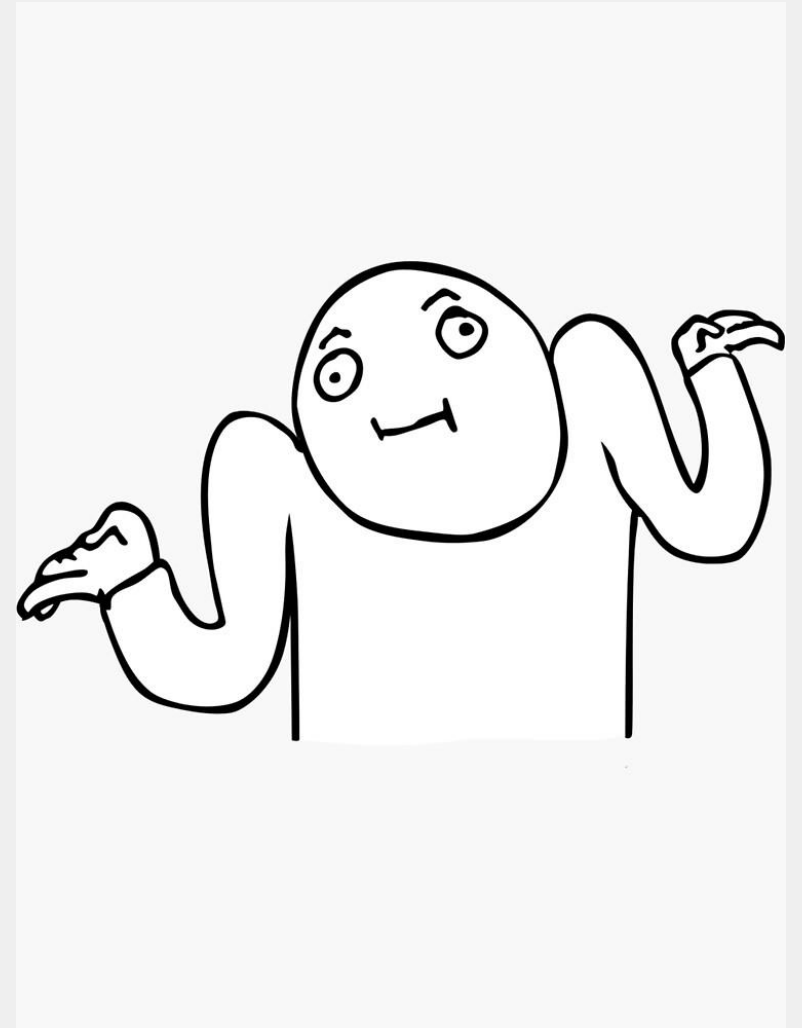
# Desenvolvendo para interfaces – Por onde começar!

{Felipe carvalho

A palestra abordará a utilização de interfaces, mostrando de uma forma simples, destacando sua importância, implementação e benefícios.



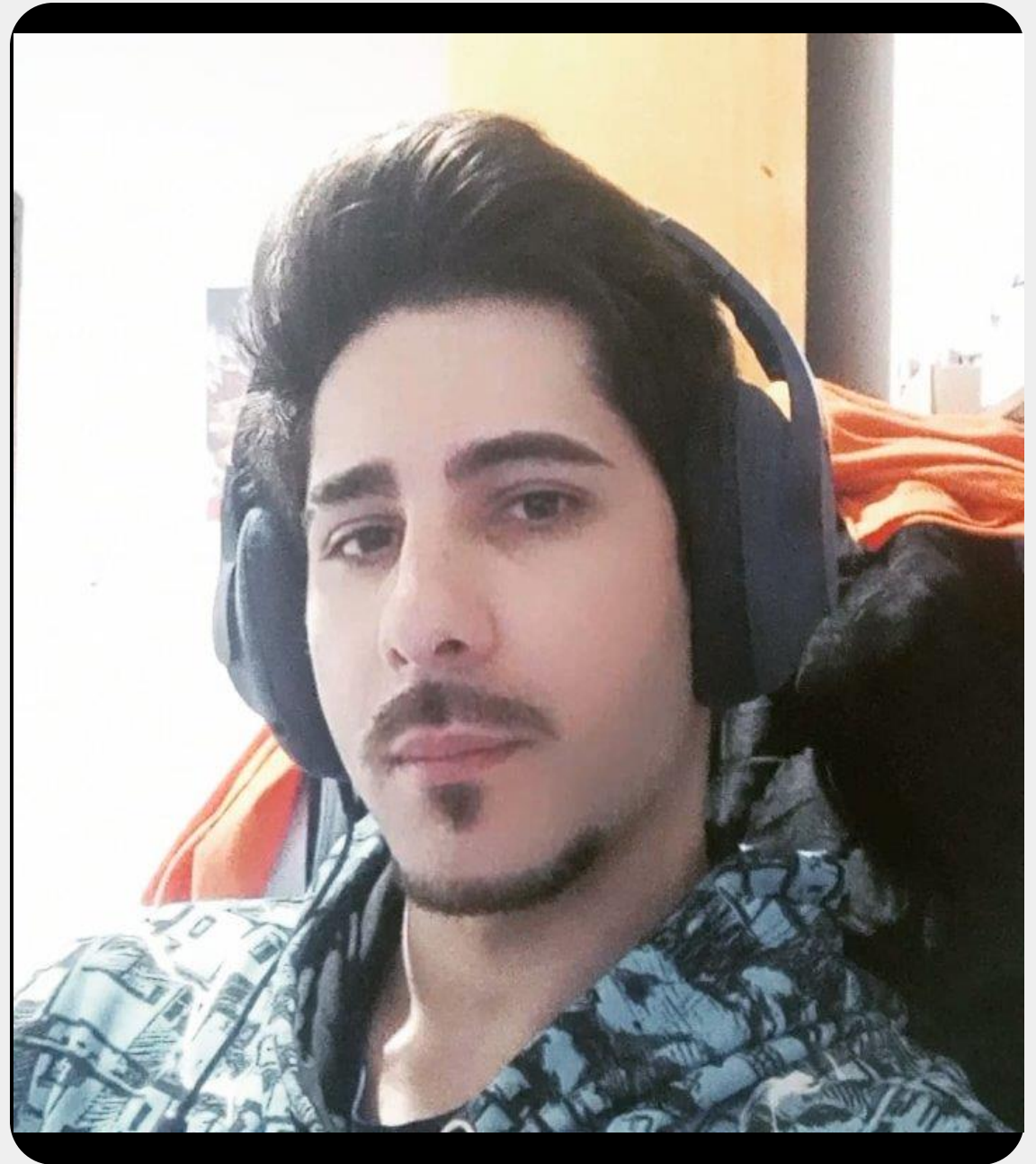
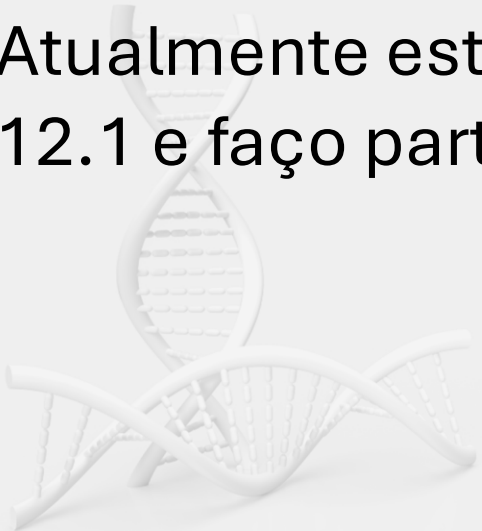
**Mas afinal quem é esse  
que está falando?**



# Quem sou?

Me chamo Felipe, trabalho com Delphi há pelo menos 9 anos. Passando por algumas versões como o famoso e destemido Delphi 7, xe2, xe7, Tokyo e outros...

Atualmente estou utilizando o 12.1 e faço parte do time da TMR.



# Tópicos

- O que são Interfaces;
- Contador de referência;
- Como criar e passar a utilizar;
- Benefícios;





**Agora chega de enrolação e vamos ao que interessa!**



# Mas afinal o que é uma interface?



# Ao fazermos uma pesquisa



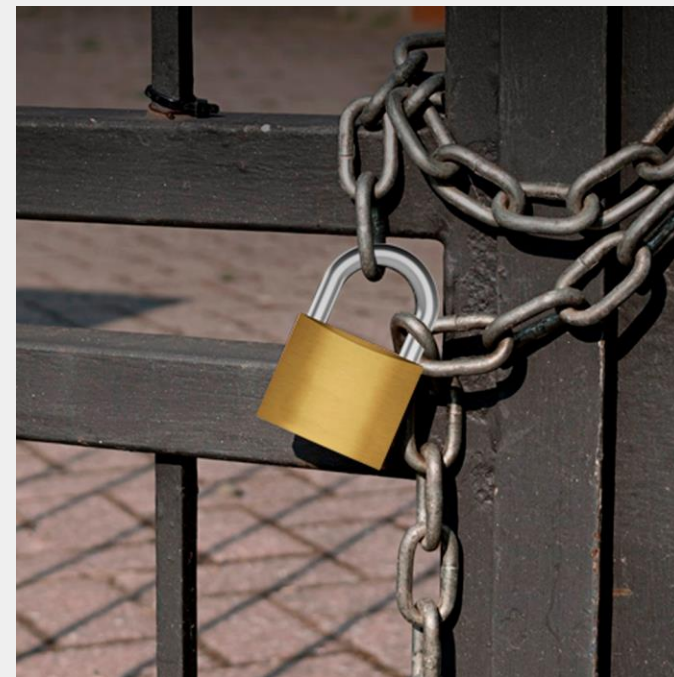


# No Wikipédia encontramos uma definição interessante

"...Uma camada de abstração onde permite que os programas utilizem os recursos do sistema sem conhecer os seus detalhes de **implementação**, esse esquema isola e protege..."







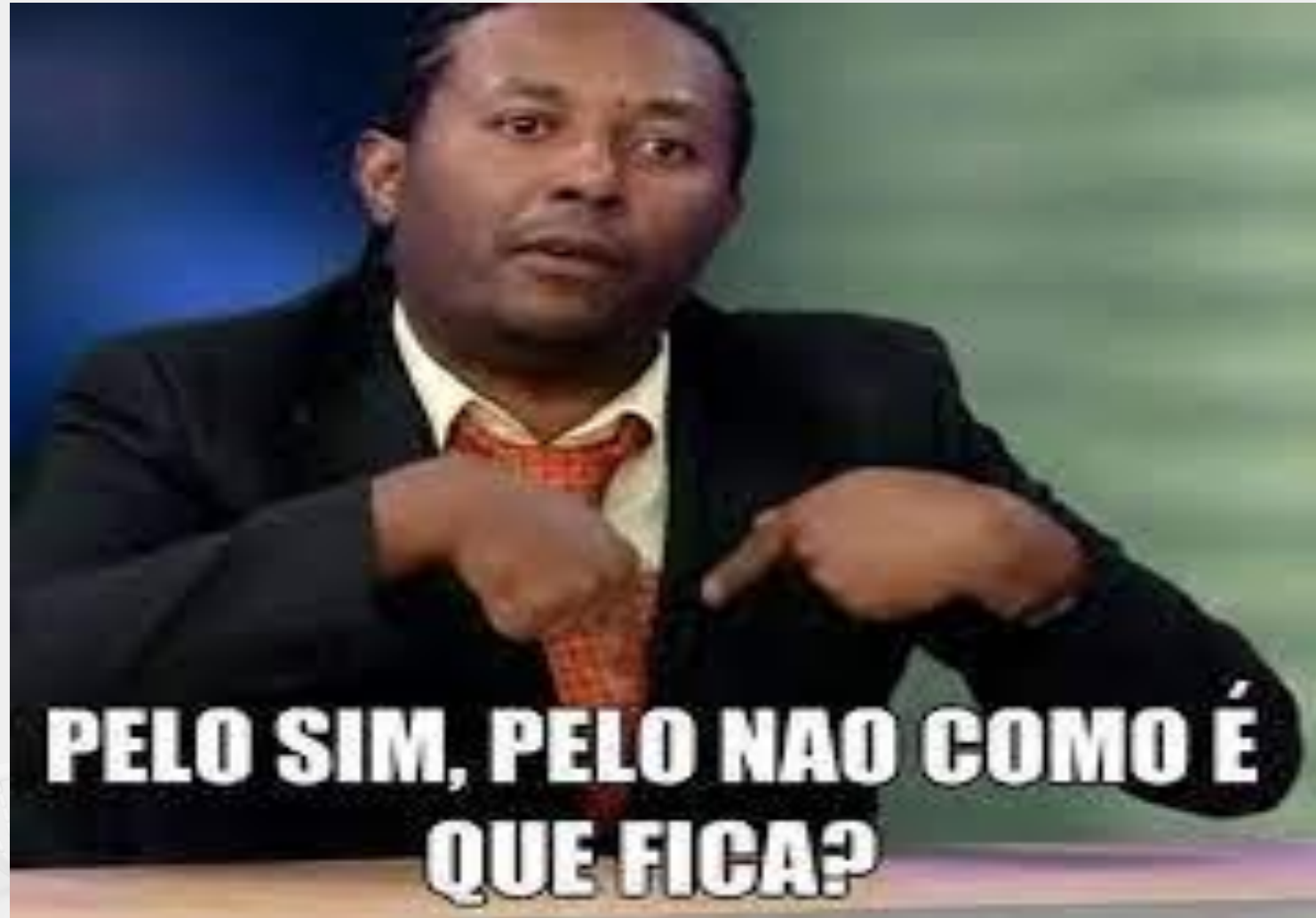
# Então podemos entender como...



São ótimas para definir "contratos" em que sejam estritos e permitindo que classes que não estejam relacionadas implementem o conjunto de métodos, sem impor uma hierarquia de herança.



**Tá, mas onde tudo isso se encaixa?**



# Bora vê então





# Aqui temos o exemplo de uma interface

```
1 unit Empresa.Intf;  
2 |  
3 interface  
4 |  
5 type  
6 IEmpresa = interface  
7     ['{2109ED53-D8A3-40E8-9CCB-F1D4B25F06FE}']  
8  
9     function GetNome: string;  
10    procedure SetNome(const aValue: string);  
11    function GetLocalidade: string;  
12    procedure SetLocalidade(const aValue: string);  
13    function GetCNPJ: string;  
14    procedure SetCNPJ(const aValue: string);  
15  
16    property Nome: string read GetNome write SetNome;  
17    property Localidade: string read GetLocalidade write SetLocalidade;  
18    property CNPJ: string read GetCNPJ write SetCNPJ;  
19 end;  
20 |  
21 implementation  
22 |  
23 end.
```

```
3 interface
4
5 type
6 IEmpresa = interface
7     ['{2109ED53-D8A3-40E8-9CCB-F1D4B25F06FE}']
8
9     function GetNome: string;
10    procedure SetNome(const aValue: string);
11    function GetLocalidade: string;
12    procedure SetLocalidade(const aValue: string);
13    function GetCNPJ: string;
```

```
interface
type
IEmpresa = interface
    ['{2109ED53-D8A3-40E8-9CCB-F1D4B25F06FE}']

    function GetNome: string;
    procedure SetNome(const aValue: string);
    function GetLocalidade: string;
    procedure SetLocalidade(const aValue: string);
    function GetCNPJ: string;
```

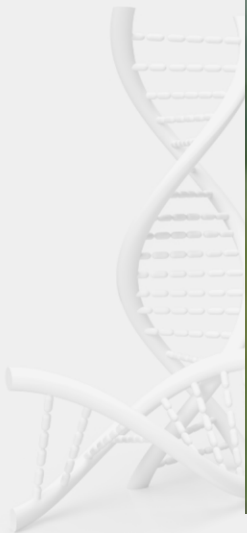


```
interface
type
IFempres - interface
    ['{2109ED53-D8A3-40E8-9CCB-F1D4B25F06FE}']
    function GetNome: string;
    procedure SetNome(const aValue: string);
    function GetLocalidade: string;
    procedure SetLocalidade(const aValue: string);
    function GetCNPJ: string;
```

- "...Para gerar uma GUID existe o atalho **Ctrl+Shift+G** ..."
- "...Um GUID é um valor binário que utilizada para identificação exclusivamente de uma interface. Contendo um GUID, é possível usar a consulta de interface para obter referências às suas implementações..."



# Lembra da relação do contrato e suas cláusulas?





# Aqui conseguimos ter uma visualização mais clara



```
4
5 type
6   IEmpresa = interface
7     ['{2109ED53-D8A3-40E8-9CCB-F1D4B25F06FE}']
8
9     function GetNome: string;
10    procedure SetNome(const aValue: string);
11    function GetLocalidade: string;
12    procedure SetLocalidade(const aValue: string);
13    function GetCNPJ: string;
14    procedure SeCNPJ(const aValue: string);
15
16    property Nome: string read GetNome write SetNome;
17    property Localidade: string read GetLocalidade write SetLocalidade;
18    property CNPJ: string read GetCNPJ write SeCNPJ;
19  end;
20
21 implementation
22
```

# Tá... mas qual a vantagem?

- ❑ Flexibilidade e Extensibilidade: Facilita a substituição de componentes.
- ❑ Segurança e Encapsulamento: Proteção contra mudanças internas.
- ❑ Testabilidade: Facilita a criação de testes unitários.



**Era só isso???**



## Calma que tem mais...

- Um dos motivos pelos quais fazemos a utilização de interfaces é sua praticidade na hora de fazer a liberação da memória da classe a qual está fazendo sua implementação.

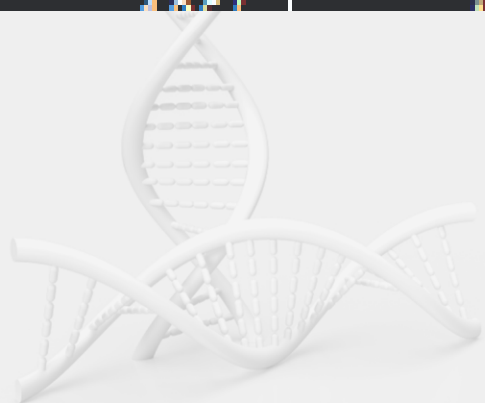




# Pra isso vamos nos apoiar na

## TInterfacedObject

```
1058 □ TInterfacedObject = class(TObject, IInterface)
1059     {$IFDEF AUTOREFCOUNT}
1060     private const
1061         objDestroyingFlag = Integer($800000000);
1062         function GetRefCount: Integer; inline;
1063     {$ENDIF}
```



# Vamos ao contador de referência

"...AddRef e \_Release rastreiam o tempo de vida de um objeto **incrementando** a contagem de referência no objeto quando uma referência de interface é passada para um cliente e **destruirão** o objeto quando essa contagem de referência for zero..."


## Memory Management of Interface Objects

*Go Up to [Using Interfaces](#)*

One of the concepts behind the design of interfaces is ensuring the lifetime management of the objects that implement them. The `_AddRef` and `_Release` methods of `IInterface` provide a way to implement this lifetime management. `_AddRef` and `_Release` track the lifetime of an object by incrementing the reference count on the object when an interface reference is passed to a client, and will destroy the object when that reference count is zero.

<https://docwiki.embarcadero.com/>



 TInterfacedObject Search for a method

```
1052 {$ENDIF}
1053
1054 { TInterfacedObject provides a threadsafe default implementation
1055   of IInterface. You should use TInterfaceObject as the base class
1056   of objects implementing interfaces. }
1057
1058 TInterfacedObject = class(TObject, IInterface)
1059 {$IFDEF AUTOREFCOUNT}
1060   private const
1061     objDestroyingFlag = Integer($80000000);
1062   function GetRefCount: Integer; inline;
1063 {$ENDIF}
1064   protected
1065   {$IFDEF AUTOREFCOUNT}
1066     [Volatile] FRefCount: Integer;
1067     class procedure __MarkDestroying(const Obj); static; inline;
1068   {$ENDIF}
1069     function QueryInterface(const IID: TGUID; out Obj): HRESULT; stdcall;
1070     function _AddRef: Integer; stdcall;
1071     function _Release: Integer; stdcall;
1072   public
1073   {$IFDEF AUTOREFCOUNT}
1074     procedure AfterConstruction; override;
1075     procedure BeforeDestruction; override;
1076     class function NewInstance: TObject; override;
1077     property RefCount: Integer read GetRefCount;
1078   {$ENDIF}
1079   end;
1080   {$IFDEF SYSTEM_HPP_DEFINES_OBJECTS}
1081   {$NODEFINE TInterfacedObject}           { defined in systobj.h }
1082   {$ENDIF}
1083
```

Mas agora chega de enrolação e bora criar  
**~~gambia...~~ código**





## Links de conteúdos de apoio

- ❑ [https://docwiki.embarcadero.com/RADStudio/Athens/en/Object\\_Interfaces\\_\(Delphi\)](https://docwiki.embarcadero.com/RADStudio/Athens/en/Object_Interfaces_(Delphi))
- ❑ <https://devspace.tuttoilmondo.com.br/desenvolvimento/interfaces-em-delphi/>
- ❑ <https://www.youtube.com/watch?v=NmNLdNV0d8>

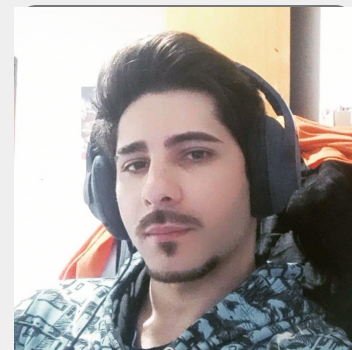


# Embarcadero Conference 2024

Inovação faz parte do nosso DNA!




Quer me ver na  
**#ECON25?**  
Acesse o QRCode  
e avalie minha palestra!




**Felipe Carvalho**

 [@felpcarvalho](https://www.instagram.com/felpcarvalho)

 [felpcarvalho](https://www.linkedin.com/in/felpcarvalho)

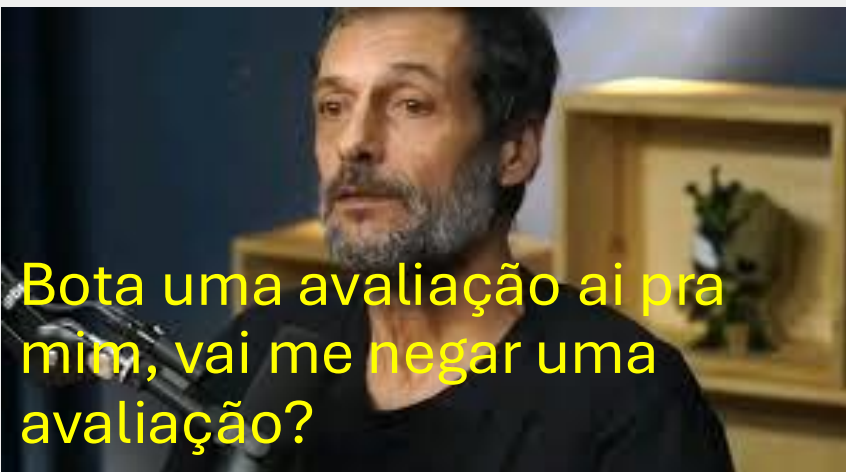
 [felicarvalho@outlook.com](mailto:felicarvalho@outlook.com)

 [\(11\) 9 9999 9999](tel:(11)999999999)

 [@SouTMRti](https://www.instagram.com/@SouTMRti)  
www.tmrri.com.br





# Não esqueça de avaliar ou a maldição do spam irá acontecer com VOCÊ!!!!!!!!!!!!!!




**Felipe Carvalho**

 [@felpcarvalho](https://www.instagram.com/felpcarvalho)

 [felpcarvalho](https://www.linkedin.com/felpcarvalho)

 [felicarvalho@outlook.com](mailto:felicarvalho@outlook.com)

 [\(11\) 9 9999 9999](tel:(11)999999999)

