



# Embarcadero Delphi Conference

Programação Baseada  
em Regras com RTTI



*Mário Guedes*



# DESAFIOS DO DIA A DIA



**Embarcadero  
Delphi Conference**



- Não é de hoje que as soluções de desenvolvimento são cada vez mais complexas:
  - Plataformas diferentes
    - 32 bits; 64 bits; Web; Mobile ...
  - Sistemas operacionais diferentes
    - Windows; Mac; Linux; iOS; Android ...
  - Integração
    - Java; .NET; PHP; Python; Ruby ...
    - Sistemas legados; Telefonia; Governos; Web Services ...



Enfim, vivemos em um momento onde as ***Regras de Negócio*** são cada vez mais ***complexas e imprevisíveis*** e exigem ***respostas rápidas***

- A nova RTTI (Delphi 2010) nos dá a oportunidade de entregar  muito mais  com  menos código ;
- Código mais limpo: menos ruído;

*Mas muitos de nós pensamos que é para poucos e iniciados;*

*Mas não é.*



# PROGRAMAÇÃO BASEADA EM REGRAS



**Embarcadero  
Delphi Conference**



- É uma forma de abordar os desafios de desenvolvimento;
- Busca facilitar a rápida adaptação do software à uma mudança de regra de negócio;
- No Delphi, podemos usar a ***RTTI***, ***Generics*** e ***Customs Attributes*** para atingir este objetivo;
- Não substitui nenhum paradigma: *trata-se apenas de uma estratégia*;



**Embarcadero  
Delphi Conference**





- **Informação de Tipo em Tempo de Execução;**
- É o *framework* oferecido pelo Delphi para prover **Reflexão** de tipos;
- Por *tipo* entenda:
  - *Classes; Records; Ordinais; Interfaces; Primitivos (integer, string, ...) e etc.;*
- O próprio Delphi usa a RTTI: afinal ele está em tempo de execução e precisa, entre outras coisas, mostrar as propriedades dos objetos no Object Inspector;
- Desde o Delphi 2010 está muito mais simples de usar e muito mais poderoso;

- Introduzido no Delphi 2009, ***tipos genéricos*** ou ***tipos parametrizados*** nos permite criar “*moldes*” de classes e métodos;
- Proporciona baixo acoplamento de código;
- Extremamente útil para listas de objetos e arrays:

```
TList<TMinhaClasse>  
...  
TArray<TMinhaClasse>
```

- Usamos a notação:

```
<tipo>
```

```
function GerarLinha<T> (ADados: T): string;
```



- A partir do Delphi 2010 temos um novo recurso na RTTI que é a classe ***TCustomAttributes***;
- É uma maneira de atribuir uma “qualidade”, ou uma “informação” a qualquer elemento da programação:
  - *Classe, Record, Campos, Métodos, Parâmetros, etc;*



- É caracterizado por uma classe descendente de *TCustomAttributes* que por sua vez não implementa nada de especial;
- Ao atribuir a “qualidade” a um elemento usamos a notação:

```
[Nome_da_Classe(Parâmetros do Construtor)]
```

```
[TExemplo('Delphi XE 3')]
```

```
property Exemplo: string read FExemplo;
```

- A unidade ***System.RTTI*** possui todo o arsenal necessário para tirar proveito do recurso;
- A unidade é muito bem documentada, facilitando o entendimento;
- Oferece várias classes de *Reflexão*;
- Em geral, fazemos chamadas recursivas e em *loop* para conseguir as informações desejadas;



- É o tipo responsável por iniciar os recursos da RTTI;
- É um record, e não uma classe!
- Deve ser inicializado e finalizado:  

```
_ctx := TRTTIContext.Create;  
_ctx.Free;
```
- Possui métodos para retornar um TRTTIType:
  - GetType
  - GetTypes
  - FindType
  - GetPackages

- Provê informações sobre um *tipo*;
- Possui métodos para obter:
  - Campos
  - Métodos
  - Propriedades
- Podemos verificar o tipo à qual o TRTTIType se refere através da propriedade *TypeKind*

- Provê informações sobre uma propriedade;
- Entre outras informações oferecidas destaca-se:
  - IsReadable – *Indica se pode ser lido;*
  - IsWritable – *Indica se pode ser escrito;*
  - Visibility – *Indica o escopo de visibilidade;*
  - GetValue – *Permite ler um valor de uma determinada instância;*
  - SetValue – *Permite escrever um valor em uma determinada instância;*



- É o tipo de reflexão que representa o **valor** de um **propriedade** de uma determinado **tipo** em uma determinada **instância** ;
- Para determinar o tipo do valor podemos usar o método: `IsType<tipo>`
- Para recuperar o valor podemos usar o método: `AsType<tipo>`
- Parece com o *variant* mas não tem nada a ver;



- Toda classe de reflexão possui o método `GetAttributes` que retorna um `TArray<TCustomAttributes>`
- Devemos, então, varrer esses atributos e tomar as decisões pertinentes ao domínio do problema que esta sendo resolvido;



# DEMONSTRAÇÃO PRÁTICA



**Embarcadero  
Delphi Conference**



- Vamos imaginar um sistema que deve gerar um documento eletrônico para o Governo, como por exemplo o EFD-Pis/Cofins;
- Vamos focar no aspecto de geração das strings que compõe o tal documento;
- Decidimos que cada layout de linha será representado por uma classe;



- Toda linha tem um número de identificação com 3 algarismos;
- As informações que compõe cada linha são separadas por *pipe*;
- Datas são representadas por ddmmyyyy;
- Valores monetários não precisam de separador decimal;
- Importante observar a ordem das linhas e dos campos;



- Linha 000 – *Data de geração do documento*
- Linha 001 – *Informações da contabilidade*
  - Nome do contador: *máximo de 50 caracteres*
  - CRC do contador: *exatamente 10 caracteres*
- Linha 002 – *Dados da empresa*
  - Nome da empresa: *máximo de 50 caracteres*
  - CNPJ: *exatamente 14 caracteres*
- Linha 003 – *Vendas*
  - Data da venda: *formato data*
  - Nome do Cliente: *máximo de 50 caracteres*
  - Valor da compra: *formato monetário*
- Linha 999 – *Fechamento do documento*



- Será desenvolvido uma classe base que através dos *atributos personalizados* gerará a string corretamente:  
TGeraLinha
- Cada linha descrita no layout terá uma classe correspondente, sendo que cada campo imprimível será representado por uma propriedade publicada (escopo **published**)
- Cada propriedade que representa um campo imprimível terá os *atributos personalizados* necessários



- As classes de geração de linha serão instanciadas e enfileiradas em uma *lista especializada* nos permitindo gerar manipular as instâncias a qualquer momento;





No nosso cenário hipotético, identificamos algumas regras possíveis e suas características, a saber:

- TCodigoLinhaAttribute
- TOrdemImpressaoAttribute
- TStringVariavelAttribute
- TStringFixaAttribute
- TDataAttribute
- TMonetarioAttribute

- Surgir uma nova linha ou outra não for mais necessária?  
*Podemos criar novas classes e descartar as que não forem mais necessárias com esforço mínimo.*
- Mudarem a formatação dos campos como a data por exemplo?  
*Simplesmente vá no “código-ninja” e faça as adequações pertinentes.*
- Mudarem o tamanho de um campo ou a ordem dos campos de uma determinada linha?  
*Vá na classe em questão e mude os valores necessários nos atributos personalizados.*
- Surgir um novo tipo de formatação, como o Boolean por exemplo?  
*Simplesmente crie um novo atributo personalizado e adeque o “código-ninja”*



- Várias soluções podem aplicar este recurso:
  - ORM
  - Serialização de objetos
  - Protocolos de comunicação
  - Geração de documentos (EDI)
  - Classes Proxies de comunicação (Data Snap)
- Faz parte da estratégia criar um gerador de código, ou seja, um aplicativo que a partir de uma fonte (banco de dados, planilha, arquivo INI, etc.) gere os tipos automaticamente.



- RTTI (Run-time Type Information)  
Rodrigo Leonhardt  
<http://edn.embarcadero.com/br/article/41728>
- Rob's Technology Corner  
Robert Love  
<http://robstechcorner.blogspot.com.br/search/label/RTTI>
- Novidades no Delphi 2010  
Rodrigo Carreiro Mourão  
<http://www.devmedia.com.br/curso/novidades-no-delphi-2010/210>
- Eu Gosto do Delphi  
Mário Guedes  
<http://eugostododelphi.blogspot.com.br/search/label/rtti>



- Portal de Treinamentos e Vagas – <http://www.edobrasil.net/treinamentos>
- Embarcadero Developer Network - <http://edn.embarcadero.com>
- Diretório de MVP's - <http://www.embarcadero.com.br/mvp-directory>
- Documentação dos Produtos - <http://docs.embarcadero.com>
- CodeRage 7 - <http://www.embarcadero.com/coderage>
- YouTube - <http://youtube.com/user/embarcaderodobrasil>
- Twitter - <https://twitter.com/EmbarcaderoBR>  
<http://twitter.com/EmbarcaderoTech>
- Blogs: <http://blogs.embarcadero.com>
- Facebook:  
<https://www.facebook.com/pages/Embarcadero-Delphi-Brasil/399151510134179>
- [atendimento@embarcadero.com.br](mailto:atendimento@embarcadero.com.br)
- (11) 5643-1333

# Obrigado!



Embarcadero  
Delphi Conference

## Mário Guedes

[mario.guedes@arrayof.com.br](mailto:mario.guedes@arrayof.com.br)

<http://eugostododelphi.blogspot.com>

<http://br.linkedin.com/in/jmarioguedes>

<http://facebook.com/eugostododelphi>

<http://twitter.com/eugostododelphi>

