



Embarcadero Conference

DROP Core + Fundamentos + Novidades

Carlos Agnes & Guilherme Dücker

Embarcadero

Conference



Apresentação

Carlos Agnes

- `type TTatu = TCarlosAgnes;`
- Bacharel em Ciência da Computação;
- Embarcadero MVP;
- Delphi Master Developer Certified;
- Desenvolvedor há 19 anos;
- Grupo Agros.



Apresentação

Guilherme Dücker:

- Tecnólogo e Análise e Desenvolvimento de Sistemas;
- Desenvolvedor há 8 anos;
- Grupo Agros.



DROP Core + Fundamentos + Novidades

Objetivos:

- Fugir do “mais do mesmo”;
- Novidades da versão 1.4;
- Sim, vai ter um pouco de ORM.



Aquasoft DROP: uma rápida introdução!

- No resumo, é uma biblioteca de classes;
- Totalmente Open Source;
 - <https://github.com/AquasoftTi/AqDrop>
- Compatível com Delphi XE4 .. Delphi Tokyo (Rio);
 - Servers (DataSnap, RADServer, etc.);
 - Desktop;
 - Apps;
- É mais conhecido por ser um framework ORM!
- Versão 1.4:
 - Resultado de um ano de melhorias;
 - Publicação prevista para 05/11/2018;

DROP Core

Embarcadero

Conference

AqDrop.Core.Calendar

- Série de classes para gerenciar eventos...

```
procedure TfrmExemplos.TesteEventoCalendario;  
var  
    lVouParaAAcademia: TAqRecurrentCalendarEvent;  
begin  
    lVouParaAAcademia := TAqRecurrentCalendarEvent.Create;  
    lVouParaAAcademia.StartDate := Date;  
    lVouParaAAcademia.StartTime := TTime.EncodeTime(20, 0, 0, 0);  
    lVouParaAAcademia.EndTime := TTime.EncodeTime(22, 0, 0, 0);  
    lVouParaAAcademia.WeeklyRecurrence.DaysOfTheWeek :=  
        [TAqWeekDay.Monday .. TAqWeekDay.Friday];  
  
    if lVouParaAAcademia.ActiveAt(Date + 7 + TTime.EncodeTime(21, 0, 0, 0)) then  
    begin  
        ShowMessage('Acho que não vou...');  
    end;  
end;
```




AqDrop.Core.Collections

- Collections turbinadas:
 - Diversos tipos de listas;
 - Dicionários;
 - Todas collections são baseadas em interfaces;
 - Funcionalidades extras como...

AqDrop.Core.Collections

```
function TAqRtti.GetType(pType: PTypeInfo): TRttiType;  
var  
    lTypeFound: Boolean;  
begin  
    FMultiReadExclusiveWriteSynchronizer.BeginRead;  
    try  
        lTypeFound := FTypesCache.TryGetValue(pType, Result);  
    finally  
        FMultiReadExclusiveWriteSynchronizer.EndRead;  
    end;  
  
    if not lTypeFound then  
    begin  
        FMultiReadExclusiveWriteSynchronizer.BeginWrite;  
        try  
            if not FTypesCache.TryGetValue(pType, Result) then  
            begin  
                Result := FContext.GetType(pType);  
                FTypesCache.Add(pType, Result);  
            end;  
        finally  
            FMultiReadExclusiveWriteSynchronizer.EndWrite;  
        end;  
    end;  
end;
```

AqDrop.Core.Collections

- Criando o dicionário do DROP com suporte a locker...

```
function TAqRtti.GetType(pType: PTypeInfo): TRttiType;  
begin  
    Result := FTypesCache.GetOrCreate(pType,  
        function: TRttiType  
        begin  
            Result := FContext.GetType(pType);  
        end);  
end;
```

AqDrop.Core.Generics.Converters

```
TAqTypeConverters = class
  strict private
  Private fields and method
  public
    constructor Create;

    procedure RegisterConverter<TFrom, TTo>(const pConverter: TFunc<TFrom, TTo>);

    function Convert(const pFrom: TValue; const pToType: PTypeInfo): TValue; overload;
    function Convert<TTo>(const pFrom: TValue): TTo; overload;
    function Convert<TFrom, TTo>(const pFrom: TFrom): TTo; overload;

  Register Standard Converters

    class procedure InitializeDefaultInstance;
    class procedure ReleaseDefaultInstance;

    class property Default: TAqTypeConverters read GetDefaultInstance;
end;
```

AqDrop.Core.Generics.Converters

```
procedure TfrmExemplos.AtualizarEdit<T>(const pEdit: TEdit; const pValor: T);  
begin  
    pEdit.Text := TAqTypeConverters.Default.Convert<T, string>(pValor);  
end;  
  
procedure TfrmExemplos.AtualizarSpinEdit<T>(const pSpinEdit: TSpinEdit; const pValor: T);  
begin  
    pSpinEdit.Value := TAqTypeConverters.Default.Convert<T, Integer>(pValor);  
end;
```


AqDrop.Core.Helpers.TObject

```
[-] TClasseA = class
    strict private
        FSomenteA: Integer;
        FAeB: string;
    end;

[-] TClasseB = class
    strict private
        FAeB: Integer;
        FSomenteB: Integer;
    end;

[-] procedure TfrmExemplos.TesteClone;
begin
    FObjetoDaClasseB := FObjetoDaClasseA.CloneTo<TClasseB>;
end;
```



Helpers para RTTI

- TAqRtti: cache para tipos e concentrador do RttiContext;
- Cache de *fields by offset*;
- GetAttribute & HasAttribute;
- TRttiType.GetDataTypes
 - Helper para tipos de dados desambiguados;
- TRttiMember
 - Métodos e propriedades unificadas para TRttiField e TRttiProperty

AqDrop.Core.InterfacesObject

- TAqInterfacesObject
- TAqARCOObject
- TAqDelegatedInterface

```
TNovaClasse = class(TMinhaArvoreQueNaoImplementaInterface, IInterface)
strict private
    FInterfaceDelegada: IAqDelegatedInterface;

    property InterfaceDelegada: IAqDelegatedInterface read FInterfaceDelegada implements IInterface;
public
    constructor Create;
end;
```



AqDrop.Core.ExecutionQueue

- Enfileiramento de tarefas / *jobs*:
 - Execução síncrona;
 - Execução assíncrona;

AqDrop.Core.Helpers.TArray

- Funções para varrer arrays:
 - Todos os itens;
 - Condição de parada;

```
procedure TfrmExemplos.TesteItemNegativo(const pArray: TArray<Integer>);  
var  
    lIndiceEncontrado: Integer;  
begin  
    if TAqArray<Integer>.SearchItem(pArray,  
        function(pItem: Integer): Boolean  
        begin  
            Result := pItem < 0;  
        end, lIndiceEncontrado) then  
    begin  
        ShowMessage(Format('O array possui ao menos um item negativo (posição %d).',  
            [lIndiceEncontrado]));  
    end;  
end;  
end;
```




Entre várias outras funcionalidades

- Implementação de alguns *patterns*;
- Bloqueador de chamadas recursivas;
- Autômatos;
- Interface simplificada para chamadas REST;
 - Obrigado Cesar Romero;
- Diversos *helpers* para tipos primitivos;

DROP ORM

Embarcadero

Conference



DROP ORM

***Framework* de Mapeamento Objeto-Relacional**

Exemplo de Mapeamento

```
[AqTable]
TPessoa = class
strict private
  [AqAutoIncrementColumn('ID')]
  FID: Integer;
  FNome: string;
  FDocumento: Int64;
  FEndereco: string;

  [AqColumn('COMPLEMENTO', [caNullIfEmpty])]
  FComplemento: string;
public
  property ID: Integer read FID;
  property Nome: string read FNome write FNome;
  property Documento: Int64 read FDocumento write FDocumento;
  property Endereco: string read FEndereco write FEndereco;
  property Complemento: string read FComplemento write FComplemento;
end;
```

```
lListaPessoas := FORMManager.Get<TPessoa>;
```



Um pouco mais sobre o ORM do DROP

- Suporte a 7 SGBDs;
- Via DBX ou FireDAC;
- Suporte ao sistema nativo de auto-incremento do SGBD;
- Mapeamento altamente dinâmico;
- Nível 4 na relação BD x Modelo de dados;
- Camada de abstração de SQLs;
- Camada de classes base com suporte a cache de dados.

Novidades na Abstração de SQLs

```
lListaPessoas := FORMManager.Get<TPessoa>;
```

```
lListaPessoas := FORMManager.Get<TPessoa>(
  procedure(pSelect: IAqDBSQLSelect)
  begin
    pSelect.Offset := 100;
    pSelect.Limit := 50;
  end);
```



Saída para Firebird

```
select first 50 skip 100  
  Pessoa.ID,  
  Pessoa.Nome,  
  Pessoa.Documento,  
  Pessoa.Endereco,  
  Pessoa.COMPLEMENTO  
from  
  Pessoa
```

Saída para Oracle

```
select * from
( select Pessoa.ID,
      Pessoa.Nome,
      Pessoa.Documento,
      Pessoa.Endereco,
      Pessoa.COMPLEMENTO,
      rownum innerrownum
  from Pessoa ) encapsultadedsource
where ( innerrownum > 100 and   innerrownum <= 150 )
```

Novo suporte à relação mestre/detalhe

```
[AqTable]
TPessoa = class
strict private
  [AqAutoIncrementColumn('ID')]
  FID: Integer;
  FNome: string;
  FDocumento: Int64;
  FEndereco: string;

  [AqColumn('COMPLEMENTO', [caNullIfEmpty])]
  FComplemento: string;

  FFilhos: TObjectList<TFilho>;
public
  property ID: Integer read FID;
  property Nome: string read FNome write FNome;
  property Documento: Int64 read FDocumento write FDocumento;
  property Endereco: string read FEndereco write FEndereco;
  property Complemento: string read FComplemento write FComplemento;
end;
```

```
[AqTable]
TFilho = class
strict private
  [AqAutoIncrementColumn('ID')]
  FID: Integer;
  [AqDetailKey('ID_PESSOA')]
  FIDPessoa: Integer;

  FNome: string;
public
  property ID: Integer read FID;
  property IDPessoa: Integer read FIDPessoa;
  property Nome: string read FNome write FNome;
end;
```



AqDrop.DB.Base.Detail

TAqDBDetailList<T: TAqDBDetail>:

- Classe da camada de classes base;
- Suporte a *lazy loading*.

Precisa da própria classe de gerenciamento de detalhes?

- É possível ensinar o DROP a usá-las!
- *How to* na própria unit AqDrop.DB.Base.Detail.



Mais novidades no ORM

- Mais opções na sintaxe fluente da abstração de SQLs;
- SQLSetup;
- Callback para leitura de dados extras da consulta;



E outros recursos nem tão novos...

- Objeto de conexão *threadsafe*;
 - Sensível a contexto;
- Suporte a Adapters e Solvers customizados;
- Cache de objetos:
 - octNone;
 - octOwnsObjects;
 - octCloned;



Roadmap

- Mais opções para customização de SQLs;
- Reaproveitamento de objetos carregados parcialmente;
- Mais opções ou facilitadores para LiveBindings;
- *Wizards* para criação de mapeamentos;
- Documentação e demos;



Reforçando

<https://github.com/AquasoftTi/AqDrop>

Versão 1.4 dia 05/11/2018

OBRIGADO

Nos vemos na trilha Delphi
da TDC Porto Alegre

Embarcadero

Conference

OBRIGADO



Carlos Agnes
tatu@tatur.com
@tatur

Guilherme Dücker
guilherme.ducker@agro1.inf.br

Embarcadero

Conference





Embarcadero Conference