Chapter 1

Unit Classes_C

1.1 Visão Geral

`TJSON_BaseObject Classe`

`TNSComponent Classe`

`TClass Classe`

`Message`

`CloneComponent`

`StrJSonToJSONObject`

`JSONObjectToStrJSon`

`StrJSonToArrays`

`ArraysToJSONValue`

`IsValidPtr`

`DISCARD`

1.2 Classes, Interfaces, Objetos e Registros

TJSON_BaseObject Classe

Hierarquia

TJSON_BaseObject > TObject

Métodos

ObjectToJSON

Declaração `public class function ObjectToJSON<T : class>(myObject: T): TJSONValue;`

JSONToObject

Declaração `public class function JSONToObject<T : class>(json: TJSONValue): T;`

TNSComponent Classe

Hierarquia

TNSComponent > TComponent

Propriedades

Alias

Declaração `published property Alias : AnsiString Read GetAlias Write SetAlias;`

Path

Declaração `public property Path : AnsiString Read _Path Write SetPath;`

ID_Dynamic

Declaração `public property ID_Dynamic : AnsiString Read GetID_Dynamic;`

RecPosition

Declaração `public property RecPosition: Longint Read _RecPosition Write SetRecPosition;`

CurrentRecord

Declaração `public property CurrentRecord: Longint Read _CurrentRecord Write SetCurrentRecord;`

Procedure_GlobalStr

Declaração `public property Procedure_GlobalStr: AnsiString read GetProcedure_GlobalStr write _Procedure_GlobalStr;`

Command

Declaração `published property Command: Integer Read _Command Write Set_Command;`

Module

Declaração `published property Module: Byte Read GetModule Write SetModule;`

HelpCtx_StrModule

Declaração `public property HelpCtx_StrModule: AnsiString read GetHelpCtx_StrModule write _HelpCtx_StrModule;`

HelpCtx_StrCommand

Declaração `public property HelpCtx_StrCommand: AnsiString read GetHelpCtx_StrCommand write _HelpCtx_StrCommand;`

HelpCtx_StrCommand_Topic

Declaração `public property HelpCtx_StrCommand_Topic: AnsiString read GetHelpCtx_StrCommand_Topic write _HelpCtx_StrCommand_Topic;`

HelpCtx_StrCurrentModule

Declaração `public property HelpCtx_StrCurrentModule: AnsiString read GetHelpCtx_StrCurrentModule write _HelpCtx_StrCurrentModule;`

HelpCtx_StrCurrentCommand

Declaração `public property HelpCtx_StrCurrentCommand: AnsiString read GetHelpCtx_StrCurrentCommand write _HelpCtx_StrCurrentCommand;`

HelpCtx_StrCurrentCommand_Opcao

Declaração `public property HelpCtx_StrCurrentCommand_Opcao: AnsiString read GetHelpCtx_StrCurrentCommand_Opcao write _HelpCtx_StrCurrentCommand_Opcao;`

HelpCtx_StrCurrentCommand_Topic

Declaração `public property HelpCtx_StrCurrentCommand_Topic: AnsiString read GetHelpCtx_StrCurrentCommand_Topic write _HelpCtx_StrCurrentCommand_Topic;`

HelpCtx_StrCurrentCommand_Topic_Content_Run

Declaração `public property HelpCtx_StrCurrentCommand_Topic_Content_Run: TEnum_HelpCtx_StrCurrentCommand_Topic_Content_run read GetHelpCtx_StrCurrentCommand_Topic_Content_Run write SetHelpCtx_StrCurrentCommand_Topic_Content_Run;`

Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File

Declaração `public property Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File: Boolean read _Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File write Set_Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File;`

HelpCtx_StrCurrentCommand_Topic_Content

Declaração `public property HelpCtx_StrCurrentCommand_Topic_Content: AnsiString read GetHelpCtx_StrCurrentCommand_Topic_Content write SetHelpCtx_StrCurrentCommand_Topic_Content;`

HelpCtx_Hint

Declaração `public property HelpCtx_Hint : AnsiString read GetHelpCtx_Hint write _HelpCtx_Hint;`

HelpCtx_Historico

Declaração `public property HelpCtx_Historico : AnsiString read GetHelpCtx_Historico write _HelpCtx_Historico;`

HelpCtx_Porque

Declaração `public property HelpCtx_Porque : AnsiString read GetHelpCtx_Porque write _HelpCtx_Porque;`

HelpCtx_Onde

Declaração `public property HelpCtx_Onde : AnsiString read GetHelpCtx_Onde write _HelpCtx_Onde;`

HelpCtx_Como

Declaração `public property HelpCtx_Como : AnsiString read GetHelpCtx_Como write _HelpCtx_Como;`

HelpCtx_Quais

Declaração `public property HelpCtx_Quais : AnsiString read GetHelpCtx_Quais write _HelpCtx_Quais;`

InstanceClass

Declaração `published property InstanceClass : TComponentClass Read GetInstanceClass;`

OkCreate

Declaração `public property OkCreate : Boolean Read _okCreate Default false;`

Owner_NSComponent

Declaração `public property Owner_NSComponent : TNSComponent read _Owner_NSComponent write SetOwner_NSComponent;`

RecordSelected

Declaração `public property RecordSelected : boolean read GetRecordSelected Write SetRecordSelected;`

FieldSelected

Declaração `public property FieldSelected : boolean read GetFieldSelected Write SetFieldSelected default false;`

HTMLContent

Declaração `public property HTMLContent : AnsiString Read GetHTMLContent;`

OnHTMLTag

Declaração `public property OnHTMLTag : Boolean Read _OnHTMLTag Write SetOnHTMLTag;`

HTMLDoc

Declaração `public property HTMLDoc : tStrings Read GetHTMLDoc Write SetHTMLDoc;`

HTMLFile

Declaração `public property HTMLFile : TFileName read GetHTMLFile Write SetHTMLFile;`

PageProducer

Declaração `public property PageProducer : TPageProducer read _PageProducer write _PageProducer;`

RecordAltered

Declaração `public property RecordAltered : Boolean read _RecordAltered write Set_RecordAltered;`

FieldAltered

Declaração `public property FieldAltered : Boolean read Get_FieldAltered write Set_FieldAltered;`

KeyAltered

Declaração `public property KeyAltered : Boolean read Get_KeyAltered write Set_KeyAltered;`

Appending

Declaração `public property Appending : Boolean read Get_Appending write Set_Appending;`

Append

Declaração `public property Append : Boolean Read _Append write SetAppend;`

RecordLimit

Declaração `public property RecordLimit : longint read Get_RecordLimit;`

Campos

Protected

Declaração `public const Protected _EditViewHelpCtx_Ok_Create_File_HTML : Boolean;`

State

Declaração `public State: Int64;`

_HTMLContent

Declaração `public _HTMLContent: AnsiString;`

Métodos

QueryInterface

Declaração `public function QueryInterface(const IID: TGUID; out Obj): Integer; stdcall;`

_AddRef

Declaração `public function _AddRef: Integer; stdcall;`

_Release

Declaração `public function _Release:  Integer; stdcall;`

Owner_Component

Declaração `public Function Owner_Component:TComponent;`

GetAlias

Declaração `protected Function GetAlias:AnsiString; Virtual;`

SetAlias

Declaração `protected Procedure SetAlias(Const aAlias:AnsiString); Virtual;`

SetPath

Declaração `protected Procedure SetPath(Const aPath:AnsiString); virtual;`

GetID_Dynamic

Declaração `protected Function GetID_Dynamic:AnsiString;`

GetCurrentField

Declaração `public Function GetCurrentField:Pointer; overload; Virtual;`

GetCurrentField

Declaração `public Function GetCurrentField(FieldNum:Longint):Pointer; overload; Virtual;`

TabIndex

Declaração `protected Function TabIndex:Longint; Virtual;`

GetAcao

Declaração `protected Function GetAcao():AnsiString; Virtual;`

DoOnHTMLTag_tgLink

Declaração
```
protected procedure DoOnHTMLTag_tgLink(Sender:  TObject; const
 TagString:  String;TagParams:  tStrings; var ReplaceText:  String); Virtual;
```

DoOnHTMLTag_tgImage

Declaração
```
protected procedure DoOnHTMLTag_tgImage(Sender:  TObject; const
 TagString:  String;TagParams:  tStrings; var ReplaceText:  String); Virtual;
```

DoOnHTMLTag_tgTable

Declaração
```
protected procedure DoOnHTMLTag_tgTable(Sender:  TObject; const
 TagString:  String;TagParams:  tStrings; var ReplaceText:  String); Virtual;
```

DoOnHTMLTag_tgImageMap

Declaração
```
protected procedure DoOnHTMLTag_tgImageMap(Sender:  TObject; const
 TagString:  String;TagParams:  tStrings; var ReplaceText:  String); Virtual;
```

DoOnHTMLTag_tgObject

Declaração
```
protected procedure DoOnHTMLTag_tgObject(Sender:  TObject; const
 TagString:  String;TagParams:  tStrings; var ReplaceText:  String); Virtual;
```

DoOnHTMLTag_tgEmbed

Declaração `protected procedure DoOnHTMLTag_tgEmbed(Sender: TObject; const TagString: String;TagParams: tStrings; var ReplaceText: String); Virtual;`

DoOnHTMLTag_tgCustom

Declaração `protected procedure DoOnHTMLTag_tgCustom(Sender: TObject; const TagString: String;TagParams: tStrings; var ReplaceText: String); Virtual;`

DoOnHTMLTag

Declaração `public procedure DoOnHTMLTag(Sender: TObject; Tag: TTag; const TagString: String;TagParams: tStrings; var ReplaceText: String);`

GetHelpCtx_Path

Declaração `protected function GetHelpCtx_Path: AnsiString; Virtual;`

GetHelpCtx_Doc_HTML

Declaração `public function GetHelpCtx_Doc_HTML: AnsiString; Virtual;`

ExecViewHelpCtx_F1

Declaração `public function ExecViewHelpCtx_F1:Word; Virtual;`

ExecViewHelpCtx_Alt_F1

Declaração `public function ExecViewHelpCtx_Alt_F1:Word; Virtual;`

ExecViewHelpCtx_Crtl_F1

Declaração `public function ExecViewHelpCtx_Crtl_F1:Word; Virtual;`

ExecViewHelpCtx

Declaração `public function ExecViewHelpCtx:Word;`

EditViewHelpCtx

Declaração `protected function EditViewHelpCtx:Word; Virtual;`

EventAvail

Declaração `protected function EventAvail:  Boolean; Virtual;`

GetEvent

Declaração `protected procedure GetEvent(var Event:  TEvent); Virtual;`

PutEvent

Declaração `protected procedure PutEvent(var Event:  TEvent); Virtual;`

GetOwner

Declaração `protected function GetOwner:  TPersistent; override;`

SetRecordAltered

Declaração `protected Function SetRecordAltered(Const aRecordAltered: Boolean):Boolean; Virtual;`

ChangeMadeOnOff

Declaração `protected procedure ChangeMadeOnOff(const aValue:Boolean); Virtual;`

SetRecPosition

Declaração `protected Procedure SetRecPosition(aRecPosition : Longint ); Virtual;`

SetCurrentRecord

Declaração `protected Procedure SetCurrentRecord(aCurrentRecord : Longint ); Virtual;`

GetProcedure_GlobalStr

Declaração `protected Function GetProcedure_GlobalStr:AnsiString; virtual;`

Set_Command

Declaração `protected Procedure Set_Command(a_Command : Integer); Virtual;`

SetCommand

Declaração `public Procedure SetCommand(aModule:Byte;aCommand : Integer;AStrModule,aStrCommand:AnsiString);`

GetModule

Declaração `protected Function GetModule: Byte; Virtual;`

SetModule

Declaração `protected Procedure SetModule(aModule : Byte ); Virtual;`

GetHelpCtx_StrModule

Declaração `protected Function GetHelpCtx_StrModule:AnsiString; virtual;`

GetHelpCtx_StrCommand

Declaração `protected Function GetHelpCtx_StrCommand: AnsiString; Virtual;`

GetHelpCtx_StrCommand_Topic

Declaração `protected Function GetHelpCtx_StrCommand_Topic: AnsiString; virtual;`

GetHelpCtx_StrCurrentModule

Declaração `protected Function GetHelpCtx_StrCurrentModule:AnsiString; virtual;`

GetHelpCtx_StrCurrentCommand

Declaração `protected Function GetHelpCtx_StrCurrentCommand: AnsiString; Virtual;`

GetHelpCtx_StrCurrentCommand_Opcao

Declaração `protected Function GetHelpCtx_StrCurrentCommand_Opcao: AnsiString; virtual;`

GetHelpCtx_StrCurrentCommand_Topic

Declaração `protected Function GetHelpCtx_StrCurrentCommand_Topic: AnsiString; virtual;`

GetHelpCtx_StrCurrentCommand_Topic_Content_Run

Declaração `protected Function GetHelpCtx_StrCurrentCommand_Topic_Content_Run: TEnum_HelpCtx_StrCurrentCommand_Topic_Content_run; virtual;`

SetHelpCtx_StrCurrentCommand_Topic_Content_Run

Declaração `protected Procedure SetHelpCtx_StrCurrentCommand_Topic_Content_Run(wHelpCtx_StrCurrentCommand_Topic_Content_Run: TEnum_HelpCtx_StrCurrentCommand_Topic_Content_run); virtual;`

Get_Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File

Declaração `protected Function Get_Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File:Boolean; Virtual;`

Set_Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File

Declaração `protected procedure Set_Ok_HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File(a_Ok_HelpCtx_StrCurrentCommar Boolean); Virtual;`

GetHelpCtx_StrCurrentCommand_Topic_Content

Declaração `protected Function GetHelpCtx_StrCurrentCommand_Topic_Content: AnsiString; virtual;`

SetHelpCtx_StrCurrentCommand_Topic_Content

Declaração `protected Procedure SetHelpCtx_StrCurrentCommand_Topic_Content(wHelpCtx_StrCurrentCommand_Topic_Content:AnsiStrin virtual;`

GetHelpCtx_Hint

Declaração `protected Function GetHelpCtx_Hint: AnsiString; VIRTUAL;`

GetHelpCtx_Historico

Declaração `protected Function GetHelpCtx_Historico: AnsiString; VIRTUAL;`

GetHelpCtx_Porque

Declaração `protected Function GetHelpCtx_Porque: AnsiString; VIRTUAL;`

GetHelpCtx_Onde

Declaração `protected Function GetHelpCtx_Onde: AnsiString; VIRTUAL;`

GetHelpCtx_Como

Declaração `protected Function GetHelpCtx_Como: AnsiString; VIRTUAL;`

GetHelpCtx_Quais

Declaração `protected Function GetHelpCtx_Quais: AnsiString; VIRTUAL;`

DoBeforeCreate

Declaração `public Procedure DoBeforeCreate(); Virtual;`

DoAfterCreate

Declaração `public Procedure DoAfterCreate; Virtual;`

GetSelf

Declaração `public Function GetSelf: TNSComponent;`

BeforeOpen

Declaração `public Function BeforeOpen(Const APath,AAlias : tString):Boolean; Virtual;`

AfterOpen

Declaração `public Function AfterOpen:Boolean; Virtual;`

Create

Declaração `public Constructor Create(AOwner: TComponent); Overload; override;`

CCreate

Declaração `public Function CCreate(wInstanceClass : TComponentClass):TNSComponent; overload; Virtual;`

CloneComponent

Declaração `public function CloneComponent(): TComponent; Virtual;`

Destroy

Declaração `public Destructor Destroy; Override;`

GetState

Declaração `public function GetState(Const AState: Longint): Boolean; Virtual;`

SetState

Declaração `public Function SetState(Const AState: Int64; Const Enable: boolean):Boolean; overload; Virtual;`

Abort_Create

Declaração `public PROCEDURE Abort_Create; Virtual;`

IsDB

Declaração `public function IsDB:Boolean; Virtual;`

IsTable

Declaração `public Function IsTable:ITable; Virtual;`

IsField

Declaração `public Function IsField:IHTML_Base; Virtual;`

IsGroup

Declaração `public Function IsGroup:Boolean; Virtual;`

IsFrame

Declaração `public Function IsFrame:IFrame; Virtual;`

IsDialog

Declaração `public Function IsDialog:Boolean; Virtual;`

IsInputText

Declaração `public function IsInputText:IInputText; Virtual;`

IsInputButton

Declaração `public function IsInputButton:IInputButton; Virtual;`

IsInputRadio

Declaração `public function IsInputRadio:IInputRadio; Virtual;`

IsInputCheckbox

Declaração `public function IsInputCheckbox:IInputCheckbox; Virtual;`

isInputPassword

Declaração `public function isInputPassword:IInputPassword; Virtual;`

isInputHidden

Declaração `public function isInputHidden:IInputHidden; Virtual;`

IsSelect

Declaração `public function IsSelect:ISelect; Virtual;`

IsComboBox

Declaração `public function IsComboBox:Boolean; Virtual;`

IsMultiCheckBoxes

Declaração `public function IsMultiCheckBoxes:Boolean; Virtual;`

IsListBox

Declaração `public function IsListBox:Boolean; Virtual;`

IsStaticText

Declaração `public function IsStaticText:Boolean; Virtual;`

IsLabel

Declaração `public function IsLabel:Boolean; Virtual;`

IsWindow

Declaração `public function IsWindow:Boolean; Virtual;`

IsHistoryWindow

Declaração `public function IsHistoryWindow:Boolean; Virtual;`

IsHistory

Declaração `public function IsHistory:Boolean; Virtual;`

IsScroller

Declaração `public function IsScroller:Boolean; Virtual;`

IsGrid

Declaração `public function IsGrid:Boolean; Virtual;`

IsScrollBar

Declaração `public function IsScrollBar:Boolean; Virtual;`

HandleEvent

Declaração `public procedure HandleEvent(var Event:  TEvent); Virtual;`

ClearEvent

Declaração `public procedure ClearEvent(var Event:  TEvent); Virtual;`

ClearEvents

Declaração `public Procedure ClearEvents;`

Top_Owner_NSComponent

Declaração `public Function Top_Owner_NSComponent:TNSComponent;`

SetOwner_NSComponent

Declaração `protected Procedure SetOwner_NSComponent(aOwner_NSComponent : TNSComponent); Virtual;`

GetRecordSelected

Declaração `protected Function GetRecordSelected: boolean; Virtual;`

SetRecordSelected

Declaração `protected Procedure SetRecordSelected(a_RecordSelected : boolean); Virtual;`

SetFieldSelected

Declaração `protected Procedure SetFieldSelected(a_FieldSelected : boolean); Overload; Virtual;`

GetFieldSelected

Declaração `protected Function GetFieldSelected: boolean; Overload; Virtual;`

GetHTMLContent

Declaração `protected Function GetHTMLContent: AnsiString; Virtual;`

SetHTMLFile

Declaração `protected Procedure SetHTMLFile(Const aHTMLFile: TFileName ); Overload; Virtual;`

SetHTMLFile

Declaração `public Procedure SetHTMLFile(); Overload; Virtual;`

SaveHTMLContentToFile

Declaração `public Function SaveHTMLContentToFile(FileNameDest:AnsiString):Integer; overload; Virtual;`

SaveHTMLContentToFile

Declaração `public Function SaveHTMLContentToFile:Integer; overload; Virtual;`

CreateHTML

Declaração `protected function CreateHTML: AnsiString; Overload; Virtual;`

Set_RecordAltered

Declaração `protected Procedure Set_RecordAltered(aSetRecordAltered:Boolean); VIRTUAL;`

Get_FieldAltered

Declaração `protected Function Get_FieldAltered:Boolean; VIRTUAL;`

Set_FieldAltered

Declaração `protected Procedure Set_FieldAltered(aFieldAltered:Boolean); VIRTUAL;`

Get_KeyAltered

Declaração `protected Function Get_KeyAltered:Boolean; VIRTUAL;`

Set_KeyAltered

Declaração `protected Procedure Set_KeyAltered(aKeyAltered:Boolean); VIRTUAL;`

Get_Appending

Declaração `protected Function Get_Appending:Boolean; VIRTUAL;`

Set_Appending

Declaração `protected Procedure Set_Appending(aAppending:Boolean); VIRTUAL;`

SetAppend

Declaração `protected Procedure SetAppend(aAppend:Boolean); Virtual;`

Get_RecordLimit

Declaração `protected Function Get_RecordLimit:  longint; overload; Virtual;`

TClass Classe

Hierarquia

TClass > TPersistent

Propriedades

OkCreate

Declaração
```
public property OkCreate :  Boolean Read _okCreate Write _okCreate
 Default false;
```

Alias

Declaração `public property Alias :  AnsiString Read GetAlias Write SetAlias;`

ClassName

Declaração `public property ClassName : String read GetClassName ;`

Campos

State

Declaração `public State: Longint;`

Métodos

GetSelf

Declaração `public Function GetSelf: TClass;`

Create

Declaração `public CONSTRUCTOR Create; Virtual;`

Free

Declaração `public PROCEDURE Free;`

Abort_Create

Declaração `public PROCEDURE Abort_Create; Virtual;`

Destroy

Declaração `public DESTRUCTOR Destroy; Override;`

GetState

Declaração `public function GetState(Const AState: Longint): Boolean; Virtual;`

SetState

Declaração `public Function SetState(Const AState: Int64; Const Enable: boolean):Boolean; overload; Virtual;`

1.3 Funções e Procedimentos

Message

Declaração `function Message(Receiver: TNSComponent; What, Command: Word;InfoPtr: Pointer): Pointer;`

CloneComponent

Declaração `function CloneComponent(aComponent: TComponent): TComponent;`

StrJSonToJSONObject

Declaração `Function StrJSonToJSONObject(StrJSon: String):TJSONObject;`

JSONObjectToStrJSon

Declaração `Function JSONObjectToStrJSon(aJSONObject :TJSONObject): String;`

StrJSonToArrays

Declaração `procedure StrJSonToArrays(StrJSon: String;Var aNames,aValues: TArrayOpenVariant);`

ArraysToJSONValue

Declaração `Function ArraysToJSONValue(Const aNames,aValues:  TArrayOpenVariant ):TJSONValue;`

IsValidPtr

Declaração `FUNCTION IsValidPtr( aClass:TNSComponent):BOOLEAN ; Overload;`

DISCARD

Declaração `PROCEDURE DISCARD(Var AClass);`

Chapter 2

Program fpmake

2.1 Uses

- `fpmkunit`

Chapter 3

Program httpproject1

3.1 Uses

- `fphttpapp`
- `unit1`(??)

Chapter 4

Unit mi.rtl

4.1 Uses

- mi.rtl.ApplicationAbstract(??)
- mi.rtl.Class_Of_Char(??)
- mi.rtl.Types(??)
- mi.rtl.Consts(??)
- mi.rtl.files(??)
- mi.rtl.Consts.StrError(??)
- mi.rtl.Consts.StringListBase(??)
- mi.rtl.Consts.StringList(??)
- mi.rtl.objects.types(??)
- mi.rtl.Objects.Consts(??)
- mi.rtl.Objects.Consts.Logs(??)
- mi.rtl.Objects.Consts.Mi_MsgBox
- mi.rtl.Objects.Consts.ProgressDlg_If(??)
- mi.rtl.Objects.Methods(??)
- mi.rtl.objects.Methods.dates(??)
- mi.rtl.Objects.Methods.Exception(??)
- mi.rtl.Objects.Methods.Paramexecucao(??)
- mi.rtl.Objects.Methods.Paramexecucao.Application(??)

- mi.rtl.Objects.Methods.StreamBase(??)
- mi.rtl.Objects.Methods.StreamBase.Stream(??)
- mi.rtl.Objects.Methods.StreamBase.Stream.FileStream(??)
- mi.rtl.Objects.Methods.StreamBase.Stream.MemoryStream(??)
- mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMemory(??)
- mi.rtl.Objects.Methods.Collection(??)
- mi.rtl.Objects.Methods.Collection.FilesStreams(??)
- mi.rtl.Objects.Methods.System(??)
- mi.rtl.Objects.Methods.Collection.SortedCollection(??)
- mi.rtl.Objects.Methods.Collection.SortedCollection.StringCollection(??)
- mi.rtl.Objects.Methods.Collection.Sortedcollection.Stringcollection.Collectionstring(??)
- mi.rtl.Objects.Methods.Collection.SortedCollection.StrCollection(??)
- mi.rtl.Objects.Methods.Db.Tb_Access(??)
- mi.rtl.Objects.Methods.Db.Tb__Access(??)
- mi.rtl.Objects.Methods.Db.Tb___Access(??)
- mi.rtl.Objectss(??)
- mi_rtl_ui_types(??)
- mi_rtl_ui_consts
- mi_rtl_ui_methods(??)
- mi_rtl_ui_DmxScroller_Buttons(??)
- mi_rtl_ui_custom_application(??)
- mi_rtl_ui_Dmxscroller(??)
- mi_rtl_ui_dmxscroller_form(??)
- LazarusPackageIntf

Chapter 5

Unit mi.rtl.ApplicationAbstract

5.1 Uses

- `Classes`
- `SysUtils`
- `CustApp`

5.2 Visão Geral

`TApplicationAbstract Classe`

5.3 Classes, Interfaces, Objetos e Registros

TApplicationAbstract Classe

Hierarquia

TApplicationAbstract > TCustomApplication

Chapter 6

Unit mi.rtl.Class_Of_Char

6.1 Uses

- `Classes`
- `SysUtils`

6.2 Visão Geral

`TClass_Of_Char Classe`

6.3 Classes, Interfaces, Objetos e Registros

TClass_Of_Char Classe

Hierarquia

TClass_Of_Char > TObject

Descrição

A class `TClass_Of_Char` é usada na tabela de caracter para conversão das letras com acentos

Propriedades

Asc_Ingles

Declaração `public property Asc_Ingles :  AnsiChar Read _Asc_Ingles;`

Asc_GUI

Declaração `public property Asc_GUI : AnsiString Read _Asc_GUI;`

Asc_HTML

Declaração `public property Asc_HTML : AnsiString Read _Asc_HTML;`

Métodos

Create

Declaração `public Constructor Create(aAsc_Ingles : AnsiChar; aAsc_GUI : AnsiString; aAsc_HTML : AnsiString);`

Chapter 7

Unit mi.rtl.Consts

7.1 Descrição

- A Unit `mi.rtl.Consts` reúne as constantes globais usados pelo pacote `mi.rtl(??)`. Esta unit foi testada nas plataformas: win32, win64 e linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **13/11/2021** : Classe criada
      - **/16/11/2021** :
      - Em **TConsts.Initialization** executar:
      - **System.FileMode** := `TConsts.FileMode(??)`;
      - Motivo: O mapa de Bits **System.FileMode** não permite acesso compartilhado.
      - **15/12/2021**
      - Criado a constante Identification = TIdentification.
      - **31/12/2021**
      - Criado a constante NRec e NRecAux para manter a compatibilidade com o passado.
      - **24/06/2022
      - Criar constante FldLink

7.2 Uses

- `Classes`
- `SysUtils`

- process
- mi.rtl.types(??)

7.3 Visão Geral

TConsts Classe

7.4 Classes, Interfaces, Objetos e Registros

TConsts Classe

Hierarquia

TConsts > TTypes(??) > TComponent

Descrição

A classe TConsts declara todas as constantes globais do pacote MarIcarai

Campos

ListaDeMsgErro

Declaração `public const ListaDeMsgErro :  TTypes.PSItem = nil;`

Descrição Pilha com tStrings de erros.

accNormal

Declaração `public const accNormal = 0;`

Descrição A constante accNormal (Const AccNormal = 0;) é um mapa de bits usado para identificar o bit do campo TDmxFieldRec.access(??) que informa se que o campo pode ser editado.

- **EXEMPLO**
  - Como usar o mapa de bits accNormal para saber se o campo pode ser editado.

```
with pDmxFieldRec^ do
  If (access and accNormal <> 0)
  then begin
         ShowMessage(Format('O campo %s pode ser editado'),[CharFie
       end;
```

accReadOnly

Declaração `public const accReadOnly = $1;`

Descrição A constante `accReadOnly` (Const ReadOnly = 1;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o campo é somente para leitura.

- **EXEMPLO**
  - Como usar o mapa de bits ReadOnly para saber se o campo não pode ser editado.

```
with pDmxFieldRec^ do
  If (access and ReadOnly <> 0)
  then begin
         ShowMessage(Format('O campo %s não pode ser editado'),[Cha
       end;
```

accHidden

Declaração `public const accHidden = $2;`

Descrição A constante `accHidden` (Const `accHidden` = 2;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o mesmo é invisível.

- **EXEMPLO**
  - Como usar o mapa de bits `accHidden` para saber se o campo é invisível.

```
with pDmxFieldRec^ do
  If (access and accHidden <> 0)
  then begin
      ShowMessage(Format('O campo %s está invisível'),[CharFieldNam
      end;
```

accSkip

Declaração `public const accSkip = $4;`

Descrição A constante `accSkip` (Const `accSkip` = 4;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o campo pode receber o focus.

- **EXEMPLO**
  - Como usar o mapa de bits `accSkip` para saber se o campo não pode receber o focus.

```
with pDmxFieldRec^ do
  If (access and accSkip <> 0)
  then begin
         ShowMessage(Format('O campo %s não pode receber o focus'),
       end;
```

accDelimiter

Declaração `public const accDelimiter = $8;`

Descrição A constante `accDelimiter` informa que o campo é delimitador de campos no Template.

accExternal

Declaração `public const accExternal = $10;`

accSpecA

Declaração `public const accSpecA = $20;`

accSpecB

Declaração `public const accSpecB = $40;`

accSpecC

Declaração `public const accSpecC = $80;`

fldStr

Declaração `public const fldStr = 'S';`

Descrição A constante `fldStr` (Const `fldStr` = 'S') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'S' após o caractere **"\"** representa no buffer do formulário um tipo ShortString que só aceita caractere maiúsculo.

- **EXEMPLO**
  - Representação de um string de 10 dígitos em um buffer de 11 bytes onde o byte zero contém o tamanho da string;

```
Const
  Nome := '\SSSSSSSSSS'
```

fldS

Declaração `public const fldS = fldStr;`

fldSTR_Minuscula

Declaração `public const fldSTR_Minuscula = 's';`

Descrição A constante `fldSTR_Minuscula` (Const `fldSTR_Minuscula` = 's') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 's' após o caractere "\" representa no buffer do formulário um tipo ShortString que só aceita caractere minúscula.

- **EXEMPLO**
  - Representação de um string de 10 dígitos em um buffer de 11 bytes onde o byte zero contém o tamanho da string;

```
Const
  Nome := '\sssssssss' //paulosergi
  Nome := '\Sssssssss' //Paulo serg
```

fldSMi

Declaração `public const fldSMi = fldSTR_Minuscula;`

fldSTRNUM

Declaração `public const fldSTRNUM = '#';`

Descrição A constante `fldSTRNUM` (Const `fldSTRNUM` = '#') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '#' após o caractere "\" representa no buffer do formulário um tipo ShortString que só aceita caractere numérico.

- **EXEMPLO**
  - Representação de um string de 11 dígitos em um buffer de 12 bytes onde o byte zero contém o tamanho da string;

```
Const
  telefone := '\(##) # ####-####' //85 9 9702 4498
```

fldSN

Declaração `public const fldSN = fldSTRNUM;`

fldAnsiChar

Declaração `public const fldAnsiChar = 'C';`

Descrição A constante `fldAnsiChar` (Const `fldAnsiChar` = 'C') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'C' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere maiúsculo.

- **EXEMPLO**
  - Representação de um AnsiString de 10 dígitos em um buffer de 11 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

```
Const
  Nome := '\CCCCCCCCCC'; //PAULO SÉRG
```

fldAC

Declaração `public const fldAC = fldAnsiChar;`

fldAnsiChar_Minuscula

Declaração `public const fldAnsiChar_Minuscula = 'c';`

Descrição A constante `fldAnsiChar_Minuscula` (Const `fldAnsiChar`(??) = 'c') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'c' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere minúsculo.

- **EXEMPLO**

– Representação de um AnsiString de 10 dígitos em um buffer de 11 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

Const

```
Nome := '\ccccccccc'; //paulo Sérg
Nome := '\Ccccccccc'; //Paulo Sérg
```

fldACMi

Declaração `public const fldACMi = fldAnsiChar_Minuscula;`

fldAnsiCharNUM

Declaração `public const fldAnsiCharNUM = '0';`

Descrição A constante `fldAnsiCharNUM` (Const `fldAnsiChar(??)` = '0') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '0' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere numérico ['0'..'9']] .

- **EXEMPLO**

  – Representação de um AnsiString de 11 dígitos em um buffer de 12 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

  Const

  ```
  telefone := '\(00) 0 0000-0000' //85 9 9702 4498
  ```

fldACN

Declaração `public const fldACN = fldAnsiCharNUM;`

fldAnsiCharVAL

Declaração `public const fldAnsiCharVAL = 'N';`

Descrição A constante `fldAnsiCharVAL` (Const `fldAnsiChar(??)` = '0') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '0' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere numérico ['0'..'9']] com formatação dbase.

- **EXEMPLO**
  - Representação de um AnsiString de 11 dígitos em um buffer de 12 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

Const

```
telefone := '\(NN) N NNNN-NNNN' //85 9 9702 4498
```

fldBYTE

Declaração `public const fldBYTE = 'B';`

Descrição byte Field

fldSHORTINT

Declaração `public const fldSHORTINT = 'J';`

Descrição shortint Field

fldSmallWORD

Declaração `public const fldSmallWORD = 'W';`

Descrição `word(??)` Field NortSoft

fldSmallInt

Declaração `public const fldSmallInt = 'I';`

Descrição `integer`(**??**) Field NortSoft

fldLONGINT

Declaração `public const fldLONGINT = 'L';`

Descrição `longint`(**??**) Field

fldRealNum

Declaração `public const fldRealNum = 'R';`

Descrição `real`(**??**) number Field (uses `TRealNum`(**??**))

fldRealNum_Positivo

Declaração `public const fldRealNum_Positivo = 'r';`

Descrição `real`(**??**) number Field positive (uses `TRealNum`(**??**))

fldBoolean

Declaração `public const fldBoolean = 'X';`

Descrição A constante `fldBoolean` (`fldBoolean` = 'X') indica que o campo é do tipo byte e só pode ter dois valores.

- **NOTA**
  - Valores possíveis:
    - 0 - False; não

* 1 = True; sim

- A forma de editá-los deve ser com o componente checkbox.

- **EXEMPLO**

```
Resourcestring
  tmp_Aceita = '\X Aceita o contrato +ChFN+'Aceita_contrato'+CharHint+'Aceita o
  Template = tmp_Aceita+' Aceita os termos do contrato
```

fldHexValue

Declaração `public const fldHexValue = 'H';`

Descrição hexadecimal numeric entry

CharUpperlimit

Declaração `public const CharUpperlimit = ^U ;`

Descrição Limite superior do campo (Somente 1 a 255)

fldENUM

Declaração `public const fldENUM = ^E;`

Descrição enumerated Field

fldBLOb

Declaração `public const fldBLOb = ^M;`

Descrição A constante `fldBLOb` indica que o campo é não formatado podendo ser um Record, porém a edição do mesmo será feito por outros meios.

- **NOTA**

- Para informar ao buffer do registro que o campo é `fldBLOb`, a função **CreateBlobField** é necessário.
- A **class function** `TUiMethods.CreateBlobField`**(??)(Len:** `integer`**(??); AccMode,Default: byte) : DmxIDstr;** reserva espaço para o mesmo.
- Pendência: Preciso criar um exemplo de uso deste tipo de informação.

FldRadioButton

Declaração `public const FldRadioButton = 'K';`

Descrição O tipo do campo `FldRadioButton` é um campo tipo `TCluster`(**??**) e é representado no Template em um controle TRadioButton

- **NOTAS**
  - Um Template pode conter vários campos do tipo cluster e o mesmo é identificado após a sequencia \K? onde ? indica que a informação pertence ao campo ?
    - Exemplo:
      - SEXO
      - \Ka Masculino
      - \Ka Feminino
      - \Ka Indefinido
  - Os campos clusteres possuem o mesmo número do campo e na primeira ocorrência contém o nome do campo na lista `pDmxFieldRec`(**??**).
- **EXEMPLO**

```
Result :=
  NewSItem('   SEXO ',
  NewSItem('    \Ka Masculino',
  NewSItem('    \Ka Feminino',
  NewSItem('    \Ka Indefinido',
  NewSItem('   ESTADO CIVIL ',
  NewSItem('    \Kb Solteiro',
  NewSItem('    \Kb Casado',
  NewSItem('    \Kb Divorciado',
  nil))))))))
```

FldDbRadioButton

Declaração `public const FldDbRadioButton = 'k';`

Descrição O tipo do campo `FldDbRadioButton` é um campo tipo String e é representado no Template em controle TDbRadioButton

- **NOTAS**
  - Um Template pode conter vários campos do tipo DbRaidoButton e o mesmo é identificado após a sequencia \k? onde ? indica que a informação pertence ao campo ?
    - Exemplo:
      - SEXO
      - \ka Masculino
      - \ka Feminino
      - \ka Indefinido
  - Os campos DbRadioButton possuem o mesmo número do campo e na primeira ocorrência contém o nome do campo na lista `pDmxFieldRec`(**??**).
  - O motivo pelo qual `FldDbRadioButton` foi criado é que o banco de dados do freepascal reconhece esse tipo como string com o nome do caption selecionado.
  - O tamanho da string deve ser o tamanho da maior string da lista de opções.

fldZEROMOD

Declaração `public const fldZEROMOD = 'Z';`

Descrição zero modifier

fldCONTRACTION

Declaração `public const fldCONTRACTION = '‘';`

Descrição A constante `fldCONTRACTION` omite da visão do usuário a parte do campo que não precisa ser mostrado, ou seja: limita a parte visível do texto permitindo scroll lateral do mesmo.

fldAPPEND

Declaração `public const fldAPPEND = ^G;`

Descrição A constante `fldAPPEND` é usada para concatenar duas listas do tipo `PSItem`(**??**).

- A constante `fldAPPEND` é necessário porque DmxScroller trabalha com string curta e a mesma tem um tamanho de 255 caracteres, onde o tamanho está na posição 0.
- Como usar a constante `fldAPPEND`:
  - A função **CreateAppendFields** retorna a constante `fldAPPEND` mais o endereço da string a ser concatenada.
    - **EXEMPLO**

```
procedure Template : ShortString;
  Var
    S1,s2,Template : TString;
begin
  S1 := ' Nome do Aluno....: \sssssssssssssssssssssss
  s2 := ' Endereço do aluno: \sssssssssssssssssssssss
  result := S1+CreateAppendFields(s2);
end;
```

    - **NOTA**
      - A contante `fldAPPEND` foi criada porque o projeto inicial foi para turbo pascal e ambiente console.
      - A versão atual podemos usar AnsiString visto que o limite do mesmo é a memória.
      - Para usar AnsiString é necessário converter para `PSitem`(**??**) com a função: **StringToSItem**.
      - **EXEMPLO:**

```
function TMI_UI_InputBox.DmxScroller_Form1Ge
begin
  with DmxScroller_Form1 do
  begin
    if _Template  <> ''
    then Result := StringToSItem(_Template,

//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 80

    else result := nil;
  end;
end;
```

fldSItems

Declaração `public const fldSItems = ^I;`

Descrição link to chain of `TSItem`(**??**) Templates

fldExtended

Declaração `public const fldExtended = 'E';`

Descrição `Real`(**??**) 10 bytes

fldReal4

Declaração `public const fldReal4 = 'O';`

Descrição `Real`(**??**) 4 Byte positivos e negativos

fldReal4Positivo

Declaração `public const fldReal4Positivo = 'o';`

Descrição `Real`(**??**) 4 Byte positivos

fldReal4P

Declaração `public const fldReal4P = 'P';`

Descrição P = `Real`(**??**) de mostrado x por 100 positivos e negativos

fldReal4PPositivo

Declaração `public const fldReal4PPositivo = 'p';`

Descrição P = `Real`(**??**) de mostrado x por 100 positivos

FldLink

Declaração `public const FldLink = ^L;`

Descrição A constante `FldLink` indica que o campo contem um campo com 255 posições que contém um endereço para um página html ou não:

- **LINKS POSSÍVEIS:**
  - ^L+1 = Endereço de uma página na web a ser acessada pelo browser.
  - ^L+2 = Nome de uma ação da lista actionItens.

FldlinkUrl

Declaração `public const FldlinkUrl = ^L+'1';`

Descrição Endereço de uma página na web a ser acessada pelo browser.

FldlinkAction

Declaração `public const FldlinkAction = ^L+'2';`

Descrição Nome de uma ação da lista actionItens.

fldData

Declaração `public const fldData = 'D';`

Descrição D = TipoData DD/DD/DD

fld_LData

Declaração `public const fld_LData = 'd' ;`

Descrição d = TDataTime;Guarda a data compactada 'dd/dd/dd'

fldLData

Declaração `public const fldLData = #1 ;`

Descrição #1 = TDataTime;Guarda a data compactada '##/##/##'

FldSData

Declaração `public const FldSData = '##/##/##';`

fldLHora

Declaração `public const fldLHora = #2 ;`

Descrição #2 = `Longint`(**??**);Guarda a hora compactada ##:##:##

FldSHora

Declaração `public const FldSHora = '##:##:##';`

fld_LHora

Declaração `public const fld_LHora = 'h';`

Descrição h = `Longint`(**??**);Guarda a hora compactada hh:hh:hh

FldOperador

Declaração `public const FldOperador = #3;`

Descrição #3 = Byte indica que o campo é um operador matemático

FldDateTimeDos

Declaração `public const FldDateTimeDos = #4 ;`

Descrição #4 = `Longint`(**??**);Guarda a data e hora compactada ##/##/## ##:##:## e o ano não pode ser menor que 1980.

FldSDateTimeDos

Declaração `public const FldSDateTimeDos = '##/##/## ##:##:##';`

CharShowPassword

Declaração `public const CharShowPassword = ^W;`

Descrição Usado para omitir os caracteres que estão sendo digitados em qualquer tipo de campo

ChSP

Declaração `public const ChSP = CharShowPassword;`

Descrição A contante `ChSP` é igual `CharShowPassword`(**??**).

CharShowPasswordChar

Declaração `public const CharShowPasswordChar = '*';`

Descrição Caractere a ser mostrado quando `CharShowPassword`(**??**) em fldField for igual = ^W

CharExecAction

Declaração `public const CharExecAction = ^T;`

Descrição A contante `CharExecAction` é usado para associar ao campo atual uma classe **TAction**.

- **NOTA**
  - O interpretador de Templates associa a ação do Template ao corrente campo.

ChEA

Declaração `public const ChEA = CharExecAction;`

Descrição A contante `ChEA` é igual `CharExecAction`(**??**).

CharLupa_Left

Declaração `public const CharLupa_Left = '';`

CharLupa_Right

Declaração `public const CharLupa_Right = '';`

Char_Seta_para_Cima

Declaração `public const Char_Seta_para_Cima = '';`

Char_Seta_para_Baixo

Declaração `public const Char_Seta_para_Baixo = '';`

Char_Seta_para_direita

Declaração `public const Char_Seta_para_direita :  AnsiString = '';`

Char_Seta_para_esquerda

Declaração `public const Char_Seta_para_esquerda :  AnsiString = '';`

Char_Seta_Back

Declaração `public const Char_Seta_Back = '';`

Char_Seta_End

Declaração `public const Char_Seta_End = '';`

Char_Seta_On

Declaração `public const Char_Seta_On = '';`

Char_Seta_Em_breve_Flecha

Declaração `public const Char_Seta_Em_breve_Flecha = '';`

Char_Seta_Top

Declaração `public const Char_Seta_Top = '';`

Char_Seta_Cicle

Declaração `public const Char_Seta_Cicle = '';`

Char_Bandeira_triangular

Declaração `public const Char_Bandeira_triangular = ' ';`

Char_Ponto_Interrogacao

Declaração `public const Char_Ponto_Interrogacao = '';`

Char_Ponto_Exclamacao

Declaração `public const Char_Ponto_Exclamacao = '';`

Char_Dedo_Direita

Declaração `public const Char_Dedo_Direita = '';`

Char_Proxima_Faixa

Declaração `public const Char_Proxima_Faixa = '';`

Char_AvancoRapido

Declaração `public const Char_AvancoRapido = '';`

Char_Retrocesso_Rapido

Declaração `public const Char_Retrocesso_Rapido = '';`

Char_Ultima_Faixa

Declaração `public const Char_Ultima_Faixa = '';`

Char_GoBof

Declaração `public const Char_GoBof = Char_Proxima_Faixa;`

Char_Next

Declaração `public const Char_Next = Char_AvancoRapido;`

Char_Prev

Declaração `public const Char_Prev = Char_Retrocesso_Rapido;`

Char_GoEof

Declaração `public const Char_GoEof = Char_Ultima_Faixa;`

Char_Refresh

Declaração `public const Char_Refresh = Char_Seta_Cicle;`

CharFieldName

Declaração `public const CharFieldName = ^B;`

Descrição A constante `CharFieldName` informa o nome do campo no Template. O nome do campo é passado após ^B e o mesmo não pode conter espaço em branco.

- **EXEMPLO DE USO**

```
NewSitem( Nome do produto:   SSSSSSSSSSSSSSS‘SSSSSSS^BNome_do_Produto,nil);
```

ChFN

Declaração `public const ChFN = CharFieldName;`

Descrição A constante `ChFN` é igual a `CharFieldName`(**??**), foi criada para facilitar seu uso.

CharListComboBox

Declaração `public const CharListComboBox = ^C;`

Descrição A contante `CharListComboBox` indica que o campo corrente possuem uma lista de opções do mesmo tipo campo.

- **EXEMPLO DE USO**

```
NewSItem(’ Dia de vencimento: \sssssssssss’+ChFN+’Dia’+
          CreateOptions(2,NewSItem(’Dia 10’,
                                   NewSItem(’Dia 15’,
                                   NewSItem(’Dia 20’,
                                   NewSItem(’Dia 25 e 26’,
                                  nil)))))+
          CharHint+’O template do campo deve ser do tamanho do maior item da li
          ’  dias ’,
        nil);
```

- **NOTA**
  - O template do campo deve ser do tamanho do maior item da lista.

ChLCB

Declaração `public const ChLCB = CharListComboBox;`

Descrição A contante `ChLCB` é igual a `CharListComboBox`(**??**)

TypeDate

Declaração
```
public const TypeDate = '\
 ZB'+^F+^U+AnsiChar(31)+#0+'/'+'ZB'+^U+AnsiChar(12)+#0+'/'+'ZB'+#0+^F;
```

Descrição the same Date Field with a Day/Month/Year sequence

_TypeDate

Declaração
```
public const _TypeDate = '\
 ZB'+^F+^U+AnsiChar(31)+#0+'/'+'ZB'^U+AnsiChar(12)+#0+'/'+'ZB'+^F;
```

TypeHora

Declaração
```
public const TypeHora = '\
 ZB'+^F+^U+AnsiChar(24)+#0+':'+'ZB'^U+AnsiChar(60)+#0+':'+'ZB'^U+AnsiChar(60)+#0+^F;
```

FldMemo

Declaração `public const FldMemo = 'M';`

TypeMemo

Declaração
```
public const TypeMemo =
 '\ZB'+^F+#0'sssssssss'#0'ZZZZZZZL'#0'ZZZZW'#0'ZZZZW'+#0+^F;
```

Descrição Usado em conjunto com `FldBLob`(**??**)

CTypeReal

Declaração `public const CTypeReal = [fldRealNum,fldReal4,fldReal4P,fldRealNum_Positivo,fldExtended];`

CTypeAnsiChar

Declaração `public const CTypeAnsiChar = [fldAnsiChar,fldAnsiChar_Minuscula,fldAnsiCharVAL];`

CTypeString

Declaração `public const CTypeString = [fldSTRNUM,fldSTR,fldSTR_Minuscula];`

CTypeInteger

Declaração `public const CTypeInteger = [fldENUM,fldBOOLEAN,fldBYTE,fldSHORTINT,fldSmallWORD,fldSmallInt,fldLONGINT,FldRadioButton];`

CTypeDate

Declaração `public const CTypeDate = [fldData,fldLData,fld_LData,FldDateTimeDos];`

CTypeHour

Declaração `public const CTypeHour = [fldLHora,fld_LHora];`

CTypeBlob

Declaração `public const CTypeBlob = [FldMemo,fldBLOb];`

CTypeOperator

Declaração `public const CTypeOperator = [FldOperador];`

CTypeKnown

Declaração `public const CTypeKnown : AnsiCharSet = CTypeReal + CTypeAnsiChar + CTypeString + CTypeInteger + CTypeDate + CTypeHour + CTypeBlob + CTypeOperator;`

efSync

Declaração `public const efSync = 0;`

efAsync

Declaração `public const efAsync = 1;`

SW_SHOWNORMAL

Declaração `public const SW_SHOWNORMAL : integer = ord(swoShow);`

Password_Admin

Declaração `public const Password_Admin : string = '123456';`

Descrição 0 = Indica uso normal do produto; 1= Indica que a `Password_admin` esta logado

Admin_Logado

Declaração `public const Admin_Logado : SmallWord = 0;`

FileModeDenyALL

Declaração `public const FileModeDenyALL : Boolean = False;`

Descrição Indica se o arquivo e exclusivo. Usado em Set_FileModeDenyALL

FlushBuffer

Declaração `public const FlushBuffer : Boolean = true;`

Descrição

- A constante `FlushBuffer` dar opção para usar cache de disco ou não.
  - **NOTA**
    - Se **True** então executa executa **FlushDOSFile** apos atualização dos arquivos.

FlushBuffer_Disk

Declaração `public const FlushBuffer_Disk : Boolean = False;`

Descrição

- A constante `FlushBuffer_Disk` é usado para indicar a banco de dados MarIcarai se deve usar cache de disco ou não.
  - **NOTA**
    - Se **True** executa executa **SysFileFlushBuffers** após atualização dos arquivos.

FlushBuffer_Disk_Transaction

Declaração `public const FlushBuffer_Disk_Transaction : Boolean = False;`

Descrição

- A constante `FlushBuffer_Disk_Transaction` é usado para indicar ao banco de dados **MarIcarai** se deve usar cache de disco ou não.
  - **NOTAS**
    - False = habilita cache de gravação das transações
    - True = Desabilita cache de gravação das transações

OkTempoDeTentativas

Declaração `public const OkTempoDeTentativas : Boolean = true;`

Descrição

- A constante `OkTempoDeTentativas` habilita o loop `TempoDeTentativas`(**??**) nas leitura e escritas ao arquivo.

TempoDeTentativas

Declaração `public const TempoDeTentativas : Longint = 30;`

Descrição

- A constante `TempoDeTentativas` é o tempo em segundos de tentativos nos processos de abertura, leitura e gravação de arquivos.

UAnsiChar

Declaração `public const UAnsiChar : AnsiChar = ' ';`

Descrição Último caractere digitado

TeclaF

Declaração `public const TeclaF : SmallInt = 0;`

Descrição Usado em readKey para capturar as Teclas Alt, Ctrl, Shift etc.

Identification

Declaração

```
public const Identification: TIdentification = (
 Id_branch : 0;
 Id_user : 1;
 UserName : '';
 FullUserName : '';
 password : ''; );
```

Descrição

- A constante `Identification` é usada para manter os dados do usuário logado ao sistema.
  - Id_branch : 0; //Número da filial do usuário logado
  - Id_user : 1; // Número do usuário Logado;
  - UserName : 'PauloSSPacheco'; // Nome do usuário logado
  - FullUserName : ''; //: Nome completo do usuário logado
  - password : ''; //: Password do usuário logado

TimeCmTime

Declaração `public const TimeCmTime :  Longint = 10;`

Descrição Número da filial do usuário logado Número do usuário Logado; Nome do usuário logado Nome completo do usuário logado Password do usuário logado

CTRL_SLEEP_ENABLE

Declaração `public const CTRL_SLEEP_ENABLE : Boolean = True;`

Descrição

- A constante pública global `CTRL_SLEEP_ENABLE` indica se o sistema deve executar a aplicação central caso a rotina atual tiver em loop aguardando alguma ação.
  - Exemplo: Tentando abrir um arquivo onde o mesmo se encontra dentro de uma transação.

FORMS_APPLICATION_PROCESS_MESSAGES

Declaração `public const FORMS_APPLICATION_PROCESS_MESSAGES : Boolean = false;`

Descrição

- A contante `FORMS_APPLICATION_PROCESS_MESSAGES` indica se deve ou não executar a aplicação principal;
  1. True : Processa as mensagem da aplicação gráfica quando necessário;
  2. False : ignora.

FORMS_APPLICATION_SHOW_MODAL

Declaração `public const FORMS_APPLICATION_SHOW_MODAL : Boolean = false;`

Descrição

- Se a constante `FORMS_APPLICATION_SHOW_MODAL`=**True** indica que `FORMS_APPLICATION_PROCESS_` e o método **TForm_VCL_DmxEditor.ShowModal** está em execução.

HANDLE_INVALID

Declaração `public const HANDLE_INVALID = high(THandle);`

Descrição

- A constante `HANDLE_INVALID` é usada para checar se um handle de um dispositivo é válido ou não

LF

Declaração `public const LF = #10;`

CR

Declaração `public const CR = #13;`

New_Line

Declaração `public const New_Line = ;`

Descrição

- A constantes `New_Line` é usado em **writeln** para passar a linha.

fmOpenRead

Declaração `public const fmOpenRead = SysUtils.fmOpenRead ;`

Descrição

- Abre um arquivo com acesso somente leitura.
  - Mapa de bits : 0 = Bit 0 desligado.
  - REFERÊNCIA:

    `fmopenread` (https://www.freepascal.org/docs-html/rtl/sysutils/fmopenread.ht

fmOpenWrite

Declaração `public const fmOpenWrite = SysUtils.fmOpenWrite ;`

Descrição

- Abre um arquivo com acesso somente gravação.
  - Mapa de bits : 1 = Bit 0 ligado
  - REFERÊNCIA:

    `fmopenwrite` (https://www.freepascal.org/docs-html/rtl/sysutils/fmopenwrite.ht

fmOpenReadWrite

Declaração `public const fmOpenReadWrite = SysUtils.fmOpenReadWrite;`

Descrição

- Abre um arquivo com acesso de leitura e gravação
  - Mapa de bits : 10 = Bit 1 ligado
  - REFERÊNCIA:

    `fmopenreadwrite` (https://www.freepascal.org/docs-html/rtl/sysutils/fmopenreadwri

fmShareCompat

Declaração `public const fmShareCompat = SysUtils.fmShareCompat ;`

Descrição

- A Constante `fmShareCompat` é usada na abertura do arquivo indicando no modo de compatibilidade com o DOS
  - Mapa de bits: 0 = bit 0(zero) desligado
  - REFERÊNCIA:

    `fmshareexclusive`(**??**) (https://www.freepascal.org/docs-html/rtl/sysutils/fmsharecompat

fmShareExclusive

Declaração `public const fmShareExclusive = SysUtils.fmShareExclusive;`

Descrição

- Flag usado para tornar acesso ao arquivo no modo exclusivo.
  - **NOTA**
    - Binário: 10000 = Bit 4 ligado
    - As contantes usadas para abertura de arquivo da **unit SysUtils** é totalmente diferente das constantes usadas na **unit system**, por isso o exemplo abaixo não funciona.
  - **REFERÊNCIA**:

`fmshareexclusive` (https://www.freepascal.org/docs-html/rtl/sysutils/fmshareexclusi

  - EXEMPLO:

```
function TFormTests.fTest_Reset(Var f : file  ):longint;
Begin
  AssignFile(f,'./doc/index.html');

{$i-}
Reset(f);
{$i+}
Result := IoResult;
  If Result <> 0
  then ShowMessage('Error: '+ErrorMessage(result));
end;

procedure TFormTests.Button_Test_ResetClick(Sender: TObject);
  Var
    f1,f2 : file;
begin
  fileMode := fmOpenReadWrite or fmShareExclusive  or fmShareCompa

  ShowMessage(IntToStr(fileMode));

  if fTest_Reset(f1) = 0
  Then fTest_Reset(f2);
```

```
  closeFile(f1);
  closeFile(f2);
end;
```

fmShareDenyWrite

Declaração `public const fmShareDenyWrite = SysUtils.fmShareDenyWrite;`

Descrição

- Bloqueie o arquivo para que outros processos possam apenas ler.
  - Mapa de Bit: 100000 = Bit 5 ligado.
  - REFERÊNCIA:

`fmsharedenywrite` (https://www.freepascal.org/docs-html/rtl/sysutils/fmsharedenywr

fmShareDenyRead

Declaração `public const fmShareDenyRead = SysUtils.fmShareDenyRead ;`

Descrição

- Bloqueie o arquivo para que outros processos não possam ler.
  - Mapa de bits: 110000 = Bit 4 e 5 ligado.
  - REFERÊNCIA:

`fmsharedenyread` (https://www.freepascal.org/docs-html/rtl/sysutils/fmsharedenyrea

fmShareDenyNone

Declaração `public const fmShareDenyNone = SysUtils.fmShareDenyNone ;`

Descrição

- Não bloqueie o arquivo.
  - Mapa de bits: 1000000 = Bit 6 ligado
  - REFERÊNCIA:

`fmsharedenynone` (https://www.freepascal.org/docs-html/rtl/sysutils/fmsharedenyno

GLOBAL_RIGHTS

Declaração `public const GLOBAL_RIGHTS = 0;`

Descrição

- A constante `GLOBAL_RIGHTS` é usada em FileCreate se o sistema for linux.
  - **REFERÊNCIA**
    - https://www.gnu.org/software/libc/manual/html_node/Permission Bits.html

faReadOnly

Declaração `public const faReadOnly = SysUtils.faReadOnly ;`

Descrição

- O atributo `faReadOnly` indica que o arquivo é somente para leitura.
  - **REFERÊNCIA**
    - https://www.freepascal.org/docs-html/rtl/sysutils/fareadonly.html
    - https://www.freepascal.org/docs-html/rtl/sysutils/findfirst.html
  - **NOTA**
    - Usado em TSearchRec e FindFirst

faDirectory

Declaração `public const faDirectory = SysUtils.faDirectory ;`

Descrição

- O atributo `faDirectory` indica que o arquivo é um diretório.
  - **REFERÊNCIA**
    - https://www.freepascal.org/docs-html/rtl/sysutils/fadirectory.html
    - https://www.freepascal.org/docs-html/rtl/sysutils/findfirst.html
  - **NOTA**

        * Usado em TSearchRec e FindFirst

    - **EXEMPLO**

```
procedure TFormTests.Button_GetInfoFileClick(Sender: TObject);

  function GetInfoFile(FileName:string ; out info : TSearchRec): In

  begin
     Result := FindFirst(FileName,faDirectory,Info);
     if Result = 0
     then Begin
            ShowMessage('O Diretório '+fileName+' encontrado.');
          end
     else begin
            ShowMessage('O diretório '+fileName+' não encontrado.')
          end;
  end;

  var
   Info: TSearchRec;
   err : integer;
begin
  err := GetInfoFile(ExpandFileName('..'),info);
  if err = 0 then
  Begin
    showMessage(intToStr(info.Attr));
    FindClose(Info);
  end;
end;
```

faNormal

Declaração `public const faNormal = SysUtils.faNormal ;`

Descrição

- O atributo `faNormal` indica que é uma arquivo normal.

    - **REFERÊNCIA**

        * https://www.freepascal.org/docs-html/rtl/sysutils/fanormal.html

    - **NOTA**

        * Usado em FindFirst para indicar que arquivos normais devem ser incluídos no resultado.

faAnyFile

Declaração `public const faAnyFile = SysUtils.faAnyFile ;`

Descrição

- O atributo `faAnyFile` indica que corresponder a qualquer arquivo
  - **REFERÊNCIA**
    - https://www.freepascal.org/docs-html/rtl/sysutils/faanyfile.html
  - **NOTA**
    - Use este atributo na chamada FindFirst para localizar todos os arquivos correspondentes.

faArchive

Declaração `public const faArchive = SysUtils.faArchive ;`

Descrição

- O atributo `faArchive` indica que o bit do arquivo está definido.
  - **REFERÊNCIA**
    - https://www.freepascal.org/docs-html/rtl/sysutils/faarchive.html
  - **NOTA**
    - Significa que o arquivo tem o conjunto de bits de arquivo. Usado em TSearchRec e FindFirst

fsFromBeginning

Declaração `public const fsFromBeginning = SysUtils.fsFromBeginning;`

Descrição

- O mapa de bits `fsFromBeginning` indica ao `TFiles.FileSeek(??)` que o deslocamento é relativo ao primeiro byte do arquivo. Esta posição é baseada em zero. ou seja, o primeiro byte está no deslocamento 0 (zero).

fsFromCurrent

Declaração `public const fsFromCurrent = SysUtils.fsFromCurrent ;`

Descrição

- O mapa de bits `fsFromCurrent` indica ao `TFiles.FileSeek(??)` que o deslocamento é relativo à posição atual.

fsFromEnd

Declaração `public const fsFromEnd = SysUtils.fsFromEnd ;`

Descrição

- O mapa de bits `fsFromEnd` indica ao `TFiles.FileSeek(??)` que o deslocamento é relativo ao final do arquivo. Isso significa que o deslocamento só pode ser zero ou negativo neste caso.

ArquivoNaoEncontrado2

Declaração `public const ArquivoNaoEncontrado2 = 002;`

Descrição * *

PathNaoEncontrado

Declaração `public const PathNaoEncontrado = 003;`

muitosArquivosAbertoSimultaneamente

Declaração `public const muitosArquivosAbertoSimultaneamente = 004;`

AcessoNegado5

Declaração `public const AcessoNegado5 = 005;`

Seek_fora_da_faixa_do_arquivo

Declaração `public const Seek_fora_da_faixa_do_arquivo = 007;`

ErroDeMemoria

Declaração `public const ErroDeMemoria = 008;`

ErroFormatoInvalido

Declaração `public const ErroFormatoInvalido = 011;`

NaoPodeExecutarTrocaDeNomesEntreDiscos

Declaração `public const NaoPodeExecutarTrocaDeNomesEntreDiscos = 017;`

ArquivoNaoEncontrado18

Declaração `public const ArquivoNaoEncontrado18 = 018;`

DiscoProtegidoContraEscrita

Declaração `public const DiscoProtegidoContraEscrita = 019;`

UnidadeDesconhecida

Declaração `public const UnidadeDesconhecida = 020;`

DriveNaoEstaPronto

Declaração `public const DriveNaoEstaPronto = 021;`

ErroDeDadosCRC

Declaração `public const ErroDeDadosCRC = 023;`

Falha_Geral

Declaração `public const Falha_Geral = 031;`

AcessoNegado32

Declaração `public const AcessoNegado32 = 032;`

ErroViolacaoDeLacre

Declaração `public const ErroViolacaoDeLacre = 033;`

MudancaDeDiscoInvalida

Declaração `public const MudancaDeDiscoInvalida = 034;`

Campo_nao_existe_no_registro_do_arquivo

Declaração `public const Campo_nao_existe_no_registro_do_arquivo = 037;`

Tipo_em_memoria_incompativel_com_o_tipo_do_campo_no_arquivo

Declaração
```
public const
 Tipo_em_memoria_incompativel_com_o_tipo_do_campo_no_arquivo = 038;
```

Erro_de_sintaxe_na_expressao

Declaração `public const Erro_de_sintaxe_na_expressao = 039;`

Tipos_de_campos_incompativeis

Declaração `public const Tipos_de_campos_incompativeis = 040;`

Tipos_de_campo_nao_conhecido

Declaração `public const Tipos_de_campo_nao_conhecido = 041;`

Campo_em_duplicidade_na_estrutura_da_tabela

Declaração `public const Campo_em_duplicidade_na_estrutura_da_tabela = 42;`

Arquivo_ja_existe

Declaração `public const Arquivo_ja_existe = 080;`

NaoPodeCriarDiretorio

Declaração `public const NaoPodeCriarDiretorio = 082;`

ParametroInvalido

Declaração `public const ParametroInvalido = 087;`

ErroDeLeituraEmDisco

Declaração `public const ErroDeLeituraEmDisco = 100;`

ErroDeGravacaoEmDisco

Declaração `public const ErroDeGravacaoEmDisco = 101;`

ErroArquivoFechado

Declaração `public const ErroArquivoFechado = 103;`

ErroArquivoFechadoParaEntrada

Declaração `public const ErroArquivoFechadoParaEntrada = 104;`

ErrorArquivoFechadoParaSaida

Declaração `public const ErrorArquivoFechadoParaSaida = 105;`

Formato_numerico_invalido_ou_incompativel

Declaração `public const Formato_numerico_invalido_ou_incompativel = 106;`

DiscoCheio

Declaração `public const DiscoCheio = 107;`

ErroDeEscritaNoDispositivoDeSaidaImpressora

Declaração `public const ErroDeEscritaNoDispositivoDeSaidaImpressora = 160;`

ErroFaltaHardware

Declaração `public const ErroFaltaHardware = 162;`

Err_Division_by_zero

Declaração `public const Err_Division_by_zero = 200;`

ErrorNaChecagemDeFaixa

Declaração `public const ErrorNaChecagemDeFaixa = 201;`

Objeto_Nao_Inicializado

Declaração `public const Objeto_Nao_Inicializado = 210;`

Chamada_a_um_Metodo_Abstrato

Declaração `public const Chamada_a_um_Metodo_Abstrato = 211;`

Stream_Registration_error

Declaração `public const Stream_Registration_error = 212;`

Collection_Index_Out_of_range

Declaração `public const Collection_Index_Out_of_range = 213;`

ErrorTentativa_de_abrir_um_arquivo_aberto

Declaração `public const ErrorTentativa_de_abrir_um_arquivo_aberto = 217;`

Erro_Tentativa_de_excluir_um_registro_excluido

Declaração `public const Erro_Tentativa_de_excluir_um_registro_excluido = 218;`

Erro_Tentativa_de_ler_um_registro_excluido

Declaração `public const Erro_Tentativa_de_ler_um_registro_excluido = 219;`

Erro_outro_usuario_da_rede_alterou_o_registro

Declaração `public const Erro_outro_usuario_da_rede_alterou_o_registro = 220;`

Estrutura_da_tabela_esta_danificada

Declaração `public const Estrutura_da_tabela_esta_danificada = 221;`

Tentativa_de_gravar_em_um_registro_compartilhado_sem_que_o_mesmo_esteja_travado

Declaração
```
public const
Tentativa_de_gravar_em_um_registro_compartilhado_sem_que_o_mesmo_esteja_travado
= 222;
```

AppCli_Evento_Executado_Por_Outro_Processo

Declaração `public const AppCli_Evento_Executado_Por_Outro_Processo = 223;`

AppCLi_Svr_Api_Nao_Instalado

Declaração `public const AppCLi_Svr_Api_Nao_Instalado = 224;`

TTransaction_Commit_esperado

Declaração `public const TTransaction_Commit_esperado = 225;`

Erro_Expression_is_not_valid

Declaração `public const Erro_Expression_is_not_valid = 226;`

Erro_Many_Parenthesis

Declaração `public const Erro_Many_Parenthesis = 227;`

Erro_Many_operators

Declaração `public const Erro_Many_operators = 228;`

Erro_Operador_aritmetico_esperado

Declaração `public const Erro_Operador_aritmetico_esperado = 229;`

Err_CalcVal_Not_Ready_Number

Declaração `public const Err_CalcVal_Not_Ready_Number = 230;`

REC_TOO_LARGE

Declaração `public const REC_TOO_LARGE = 231;`

REC_TOO_SMALL

Declaração `public const REC_TOO_SMALL = 232;`

KeyTooLarge

Declaração `public const KeyTooLarge = 233;`

RecSizeMismatch

Declaração `public const RecSizeMismatch = 234;`

KeySizeMismatch

Declaração `public const KeySizeMismatch = 235;`

MemOverflow

Declaração `public const MemOverflow = 236;`

ArqIndexInconsistente

Declaração `public const ArqIndexInconsistente = 237;`

Descrição turbo access. Erros Db e DaAccess

O_gerente_de_transacoes_esta_inativo

Declaração `public const O_gerente_de_transacoes_esta_inativo = 238;`

Erro_Excecao_inesperada

Declaração `public const Erro_Excecao_inesperada = 239;`

Acesso_negado_ao_arquivo_por_falta_de_autorizacao_de_seu_superior_imediato

Declaração
```
public const
Acesso_negado_ao_arquivo_por_falta_de_autorizacao_de_seu_superior_imediato =
240;
```

Registro_nao_localizado

Declaração `public const Registro_nao_localizado = 241;`

O_Evento_OnEnter_Retornou_falso

Declaração `public const O_Evento_OnEnter_Retornou_falso = 242;`

Descrição > Erro retornados nas buscas de registros

O_Evento_OnExit_Retornou_falso

Declaração `public const O_Evento_OnExit_Retornou_falso = 243;`

Attempt_to_insert_record_without_is_selected

Declaração `public const Attempt_to_insert_record_without_is_selected = 244;`

Descrição Erro gerado ao tentar incluir um registro sem que o mesmo não esteja no modo appending

attempt_to_edit_a_record_not_selecting

Declaração `public const attempt_to_edit_a_record_not_selecting = 245;`

LastError

Declaração `public const LastError : SmallInt = 0;`

Descrição

- Após uma chamada ao sistema operacional a variável publica global `LastError` guarda **0 (zero)** se sucesso ou o **código do erro** se houve fracasso.
  - A função `LastError` é atualizada em SetResult.

TaStatus

Declaração `public const TaStatus : SmallInt = 0;`

Descrição

- A variável pública global `TaStatus` indica o status da última operação de acesso ao banco de dados Turbo Access.
  - Nota: Sua função é semelhante a `LastError(??)`;

OK

Declaração `public const OK : Boolean = True;`

Descrição

- A variável pública global `OK` indica se houver erro na última ação.
  - Nota: Atualizada em **SetResult** onde `OK`=**true** houve sucesso e `OK`=**false** houve fracasso.

FileMode

Declaração `public const FileMode : word = fmOpenReadWrite ;`

Descrição A constante pública `FileMode` guarda o modo padrão de abertura dos arquivos;

- **NOTAS**:
  - Usada em **FileOpen** e **FileCreate**.
  - O mapa de bits usado `FileMode` é inicializado com:
    - Const `FileMode` : `word`(**??**) = `fmOpenReadWrite`(**??**);
- **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_test_FileCreateExecute(Sender: TObject);

  var
    aHandle,aHandle2,aHandle3 :  TMI_ui_types.THandle;
    err:integer;
    s : AnsiString;
begin
  with TMI_ui_types do
  begin
    err := FileCreate('text.txt',fileMode, ShareMode ,aHandle);
    if err = 0
    then begin
           SysMessageBox('Arquivo text.txt criado na pasta corrente.','Action_t
           s := ExpandFileName('text.txt');

           FileMode := fmOpenReadWrite;
           ShareMode := fmShareCompat or fmShareDenyNone;
```

```
                err := FileOpen(s,FileMode, shareMode,aHandle2);
                if err = 0
                Then begin
                       SysMessageBox('Arquivo text.txt aberto com o modo fmOpenReadW
                       FileClose(aHandle2);
                     end
                else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreat

                ShareMode := fmShareCompat or fmShareExclusive;
                err := FileOpen(s,fileMode,ShareMode ,aHandle3);
                if err = 0
                Then begin
                       SysMessageBox('Arquivo text.txt aberto com o modo fmOpenReadW
                       FileClose(aHandle3);
                     end
                else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreat


                FileClose(aHandle);
              end
              else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreateF
    end;
  end;
```

FileModeAnt

Declaração `public const FileModeAnt :  Word = 0;`

Descrição

- A function **SetFileMode** salva a variável `FileModeAnt` atual antes de modificar `fileMode`(??).;

ShareMode

Declaração `public const ShareMode :  Cardinal = fmShareCompat or fmShareDenyNone;`

Descrição A constante pública `ShareMode` guarda o modo padrão de compartilhamento na abertura dos arquivos;

- **NOTAS**:

- Usada em **FileOpen** e **FileCreate**.
- O mapa de bits usado `ShareMode` é inicializado com:
  - Const `ShareMode` : cardinal = `fmShareCompat`(**??**) or `fmShareDenyNone`(**??**);

- **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_test_FileCreateExecute(Sender: TObject);

  var
    aHandle,aHandle2,aHandle3 :  TMI_ui_types.THandle;
    err:integer;
    s : AnsiString;
begin
  with TMI_ui_types do
  begin
    err := FileCreate('text.txt',fileMode, ShareMode ,aHandle);
    if err = 0
    then begin
           SysMessageBox('Arquivo text.txt criado na pasta corrente.','Action_t
           s := ExpandFileName('text.txt');

           FileMode := fmOpenReadWrite;
           ShareMode := fmShareCompat or fmShareDenyNone;

           err := FileOpen(s,FileMode, shareMode,aHandle2);
           if err = 0
           Then begin
                  SysMessageBox('Arquivo text.txt aberto com o modo fmOpenReadW
                  FileClose(aHandle2);
                end
           else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreat

           ShareMode := fmShareCompat or fmShareExclusive;
           err := FileOpen(s,fileMode,ShareMode ,aHandle3);
           if err = 0
           Then begin
                  SysMessageBox('Arquivo text.txt aberto com o modo fmOpenReadW
                  FileClose(aHandle3);
                end
           else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreat
```

```
            FileClose(aHandle);
          end
          else SysMessageBox(TStrError.ErrorMessage(err),'Action_test_FileCreateE
      end;
    end;
```

MaxDirSizeFat32

Declaração `public const MaxDirSizeFat32 = 65534;`

MaxDirSizeNTFS

Declaração `public const MaxDirSizeNTFS = 2294967295;`

MaxDirSizeLinux

Declaração `public const MaxDirSizeLinux = MaxDirSizeNTFS;`

Descrição

- Máximo de pastas dos sistemas de arquivo do linux é limitada ao espaço em disco.

MaxDirSize

Declaração `public const MaxDirSize :  word = MaxDirSizeLinux;`

Auto_Add_Line_Default

Declaração `public const Auto_Add_Line_Default:Boolean = false;`

Descrição

- A contante `Auto_Add_Line_Default` é usada na construção de formulários de entrada automaticamente.
  - **NOTA**
    - **true** o formulário de entrada de dados insere uma linha em branco automaticamente.

Comma

Declaração `public const Comma : char = ',';`

Descrição
- Separador de milhar nas mascaras internas ao campo.

showComma

Declaração `public const showComma : char = '.';`

Descrição
- Separador de números na visualização }

DecPt

Declaração `public const DecPt : char = '.';`

Descrição
- Ponto decimal usado nas mascaras internas ao campo.

showDecPt

Declaração `public const showDecPt : Char = ',';`

Descrição
- `Char`(**??**) decimal point display }

CloseParenthesis

Declaração `public const CloseParenthesis = ')';`

OpenParenthesis

Declaração `public const OpenParenthesis = '(';`

CharDelimiter_0

Declaração `public const CharDelimiter_0 = #0;`

Descrição A constante `CharDelimiter_0` indica qua a sequncia seguite é um campo de dados

CharDelimiter_1

Declaração `public const CharDelimiter_1 = '\';`

Descrição A constante `CharDelimiter_1` indica qua a sequncia seguite é um campo de dados

CharDelimiter_2

Declaração `public const CharDelimiter_2 = '|';`

Descrição A constante `CharDelimiter_2` separa nome da tabela do nome do campo

CharDelimiter_3

Declaração `public const CharDelimiter_3 = ' ';`

CharShowzeroes

Declaração `public const CharShowzeroes = ^Z;`

Descrição A constante `CharShowzeroes` inicializa o registro com zeros

CharFillvalue

Declaração `public const CharFillvalue = ^V;`

Descrição Se o campo for numérico, preencha com '#0'(`AccNormal`(**??**)) se for alfanumérico, preencha com ' ' `AccNormal`(**??**)

CharAccHidden

Declaração `public const CharAccHidden = ^H;`

Descrição A contante `CharAccHidden` torna o campo invisível

ChAH

Declaração `public const ChAH = CharAccHidden;`

CharAccSkip

Declaração `public const CharAccSkip = ^S;`

Descrição A constante `CharAccSkip` indica que o campo não pode receber o foco.

ChAS

Declaração `public const ChAS = CharAccSkip;`

Descrição A constante `ChAS` é igual a `CharAccSkip`(**??**)

CharAccReadOnly

Declaração `public const CharAccReadOnly = ^R;`

Descrição A constante *`CharAccReadOnly`** informa que o tipo de acesso ao campo é somente para leitura e não pode ser editado.

ChARO

Declaração `public const ChARO = CharAccReadOnly;`

Descrição A constante `ChARO` é igual a `CharAccReadOnly`(**??**)

CharAllZeroes

Declaração `public const CharAllZeroes = ^A;`

Descrição A constante `CharAllZeroes` avisa para iniciar com #0 todos os campos

CharProviderFlag

Declaração `public const CharProviderFlag = ^P;`

Descrição O caractere de controle `CharProviderFlag` é usado pelo método **TUiDmxScroller_sql.CreateTables** para indicar que o caractere seguinte tem um sinalizador usado para criar tabelas no banco de dados.

- **SINALIZADORES**
  - 0 = **pfInUpdate** : As alterações no campo devem ser propagadas para o banco de dados..
  - 1 = **pfInWhere** : O campo deve ser usado na cláusula WHERE de uma instrução de atualização no caso de upWhereChanged.
  - 2 = **pfInKey** : Campo é um campo chave e usado na cláusula WHERE de uma instrução de atualização.
  - 3 = **pfHidden** : O valor deste campo deve ser atualizado após a inserção.
  - 4 = **pfRefreshOnInsert** : O valor deste campo deve ser atualizado após a inserção.
  - 5 = **pfRefreshOnUpdate** : O valor deste campo deve ser atualizado após a atualização.
  - 6 = **pfInKeyPrimary** : Campo é um campo chave primária e usado na cláusula WHERE de uma instrução de atualização.
  - 7 = **pfInAutoIncrement** : Campo é um campo autoincremental e usado em uma instrução de atualização.
- **NOTAS**
  - O campos com access = ^S automaticmanente o atributo MIProviderFlag terá [pfHidden]

- O valor defaults de MiProviderFlags := [pfInUpdate,pfInWhere];
- **Campos de chave primária**
    - Ao atualizar registros, TSQLQuery precisa saber quais campos compõem a chave primária que pode ser usada para atualizar o registro e quais campos devem ser atualizados: com base nessas informações, ele constrói um comando SQL UPDATE, INSERT ou DELETE.
    - A construção da instrução SQL é controlada pela propriedade UsePrimaryKeyAsKey e pelas propriedades ProviderFlags .
    - A propriedade Providerflags é um conjunto de 3 sinalizadores:
        - pfInkey : O campo faz parte da chave primária
        - pfInWhere : O campo deve ser utilizado na cláusula WHERE das instruções SQL.
        - pfInUpdate : Atualizações ou inserções devem incluir este campo. Por padrão, ProviderFlags consiste apenas em pfInUpdate .
    - **REFERÊNCIA**

Working_With_TSQLQuery (https://wiki.freepascal.org/Working_With_TSQLQu

CharpfInUpdate

Declaração `public const CharpfInUpdate = ^P'0';`

Descrição As alterações no campo devem ser propagadas para o banco de dados..

CharpfInWhere

Declaração `public const CharpfInWhere = ^P'1';`

Descrição O campo deve ser usado na cláusula WHERE de uma instrução de atualização no caso de upWhereChanged.

CharPfInKey

Declaração `public const CharPfInKey = ^P'2';`

Descrição Campo é um campo chave e usado na cláusula WHERE de uma instrução de atualização.

CharPfHidden

Declaração `public const CharPfHidden = ^P'3';`

Descrição O valor deste campo deve ser atualizado após a inserção.

CharPfRefreshOnInsert

Declaração `public const CharPfRefreshOnInsert = ^P'4';`

Descrição O valor deste campo deve ser atualizado após a inserção.

CharPfRefreshOnUpdate

Declaração `public const CharPfRefreshOnUpdate = ^P'5';`

Descrição O valor deste campo deve ser atualizado após a atualização.

CharPfInKeyPrimary

Declaração `public const CharPfInKeyPrimary = ^P'6';`

Descrição Campo é um campo chave primária e usado na cláusula WHERE de uma instrução de atualização.

CharPfInKeyPrimaryAutoIncrement

Declaração `public const CharPfInKeyPrimaryAutoIncrement = ^P'7';`

Descrição Campo é um campo autoincremental e usado em uma instrução de atualização.

CharForeignKey

Declaração `public const CharForeignKey = ^F ;`

Descrição Produz um erro indicando que a exclusão ou atualização criaria uma violação de restrição de chave estrangeira. Se a restrição for adiada, esse erro será produzido no momento da verificação da restrição se ainda existirem linhas de referência. Esta é a ação padrão.

CharFk_No_Action

Declaração `public const CharFk_No_Action = ^F'0' ;`

Descrição Produz um erro indicando que a exclusão ou atualização criaria uma violação de restrição de chave estrangeira. Isso é o mesmo que, **NO ACTION** exceto que o cheque não é adiável.

CharFk_Restrict

Declaração `public const CharFk_Restrict = ^F'1' ;`

Descrição Exclua todas as linhas que fazem referência à linha excluída ou atualize os valores das colunas de referência para os novos valores das colunas referenciadas, respectivamente.

CharFk_Cascade

Declaração `public const CharFk_Cascade = ^F'2' ;`

Descrição Defina a(s) coluna(s) de referência como nula.

CharFk_Set_Null

Declaração `public const CharFk_Set_Null = ^F'3' ;`

Descrição A contante `CharFk_Set_Null` defina a(s) coluna(s) de referência para seus valores padrão. (Deve haver uma linha na tabela referenciada que corresponda aos valores padrão, se eles não forem nulos, ou a operação falhará.

CharFk_Set_Default

Declaração `public const CharFk_Set_Default = ^F'4' ;`

CharHint

Declaração `public const CharHint = '^';`

Descrição O A constante `CharHint` é usado para documentar o campo e indica que todo o texto até o próximo caractere de controle será o conteúdo do campo HelpCtx_Hint.

- **EXEMPLO** "'pascal

  Resourcestring `tmp_Alunos_Idade(??)` = '\BB'+`ChFN(??)`+'idade'+`CharUpperlimit(??)`+#64+
  `CharHint`+'A idade do aluno. Valores válidos 1 a 64'+ `CharHintPorque(??)`+'Este campo é necessário para que se agrupe o alunos baseado em sua faixa etária'+ `CharHintOnde(??)`+'Ele será usado pelo coordenador ao classificar a turma';

  `tmp_Alunos_Matricula(??)` = \IIII'+`ChFN(??)`+'matricula'+`CharHint`+'A matricula do aluno é um campo sequencial e calculado ao incluir o registro';

  `tmp_Alunos(??)` = ' Idade: '+`tmp_Alunos_Idade(??)`+`lf(??)`+ ' Matricula: '+`tmp_Alunos_Matr`

ChH

Declaração `public const ChH = CharHint;`

CharHintPorque

Declaração `public const CharHintPorque = '^0';`

Descrição A contante `CharHintPorque` informa que todo texto até o próximo delimitador contém informações para o campo HelpCtx_Porque

CharHintOnde

Declaração `public const CharHintOnde = '^1';`

Descrição A contante `CharHintOnde` informa que todo texto até o próximo delimitador contém informações para o campo HelpCtx_Onde

Delimiters

Declaração `public const Delimiters : AnsiCharSet = [CharDelimiter_0, CharDelimiter_1, CharDelimiter_2, CharDelimiter_3, CharExecAction, CharFieldName, CharUpperlimit, CharAccHidden, CharAccSkip, CharAccReadOnly, CharAllZeroes, CharProviderFlag, CharForeignKey, CharHint, fldAPPEND, fldSItems, CharListComboBox];`

SinalDireita

Declaração `public const SinalDireita : Boolean = False;`

SinalDeMaisAtivo

Declaração `public const SinalDeMaisAtivo : Boolean = False;`

Descrição

- Mostra o sinal de + a direita dos campos numéricos

MaskIsNumber

Declaração `public const MaskIsNumber : TAnsiCharSet = [];`

Delta_Locate

Declaração `public const Delta_Locate : Longint = 100;`

Descrição

ConvertIdioma_Nil

Declaração `public const ConvertIdioma_Nil : TConvertIdioma = nil;`

Html_Nivel1

Declaração `public const Html_Nivel1 = '<font size="6">▪</font>';`

Html_Nivel2

Declaração `public const Html_Nivel2 = '<font size="5">▪</font>';`

Html_Nivel3

Declaração `public const Html_Nivel3 = '<font size="4">▪</font>';`

Html_Nivel4

Declaração `public const Html_Nivel4 = '<font size="2">▪</font>';`

Char_Nivel1

Declaração `public const Char_Nivel1 = Ansichar(254);`

Char_Nivel2

Declaração `public const Char_Nivel2 = Ansichar(207);`

Char_Nivel3

Declaração `public const Char_Nivel3 = Ansichar(248);`

Char_Nivel4

Declaração `public const Char_Nivel4 = Ansichar(250);`

Array_Of_Char

Declaração `public const Array_Of_Char : TArray_Of_Char = ((Asc_Ingles :'a';Asc_GUI :'á';Asc_HTML :'á'), (Asc_Ingles :'a';Asc_GUI :'â';Asc_HTML :'â'), (Asc_Ingles :'a';Asc_GUI :'à';Asc_HTML :'à'), (Asc_Ingles :'a';Asc_GUI :'ã';Asc_HTML :'ã'), (Asc_Ingles :'A';Asc_GUI :'Á';Asc_HTML :'Á'), (Asc_Ingles :'A';Asc_GUI :'À';Asc_HTML :'À'), (Asc_Ingles :'A';Asc_GUI :'Â';Asc_HTML :'Â'), (Asc_Ingles :'A';Asc_GUI :'Ã';Asc_HTML:'Ã'), (Asc_Ingles :'c';Asc_GUI :'ç';Asc_HTML :'ç'), (Asc_Ingles :'C';Asc_GUI :'Ç';Asc_HTML :'Ç'), (Asc_Ingles :'e';Asc_GUI :'é';Asc_HTML :'é'), (Asc_Ingles :'e';Asc_GUI :'ê';Asc_HTML :'ê'), (Asc_Ingles :'E';Asc_GUI :'É';Asc_HTML :'É'), (Asc_Ingles :'E';Asc_GUI :'Ê';Asc_HTML :'Ê'), (Asc_Ingles :'i';Asc_GUI :'í';Asc_HTML :'í'), (Asc_Ingles :'I';Asc_GUI :'Í';Asc_HTML :'Í'), (Asc_Ingles :'o';Asc_GUI :'ó';Asc_HTML :'ó'), (Asc_Ingles :'O';Asc_GUI :'Ó';Asc_HTML :'Ó'), (Asc_Ingles :'o';Asc_GUI :'õ';Asc_HTML :'õ'), (Asc_Ingles :'O';Asc_GUI :'Õ';Asc_HTML :'Õ'), (Asc_Ingles :'o';Asc_GUI :'ô';Asc_HTML :'ô'), (Asc_Ingles :'O';Asc_GUI :'Ô';Asc_HTML :'Ô'), (Asc_Ingles :'u';Asc_GUI :'ú';Asc_HTML :'ú'), (Asc_Ingles :'U';Asc_GUI :'Ú';Asc_HTML :'Ú'), (Asc_Ingles :'u';Asc_GUI :'ü';Asc_HTML :'ü'), (Asc_Ingles :'U';Asc_GUI :'Ü';Asc_HTML :'Ü'), (Asc_Ingles :'o';Asc_GUI :'º';Asc_HTML :'º'));`

PortaDaImpressora

Declaração `public const PortaDaImpressora : tString = 'prn';`

opcaoRedireciona

Declaração `public const opcaoRedireciona : AnsiChar = 'I';`

RedirecionaImpressora

Declaração `public const RedirecionaImpressora : boolean = false;`

redirecionaImpNul

Declaração `public const redirecionaImpNul : Boolean = False;`

NomeRedireciona

Declaração `public const NomeRedireciona : PathStr = 'C:\Maricarai.Lst';`

ApartirDeQuePagina

Declaração `public const ApartirDeQuePagina : Longint= 1;`

Descrição Caso `ApartirDeQuePagina` > 1 então redireciona para NUL todas as paginas dos relatórios ate que `ContaPagina`(**??**) seja = `ApartirDeQuePagina`

PaginaInicial

Declaração `public const PaginaInicial: Longint= 1;`

Descrição Pagina inicial na listagem

contalinha

Declaração `public const contalinha : Longint = 0;`

contaPagina

Declaração `public const contaPagina : Longint= 1;`

CmNulo

Declaração `public const CmNulo = 100 ;`

CmDbNextRec

Declaração `public const CmDbNextRec = CmNulo + 01;`

CmDbPrevRec

Declaração `public const CmDbPrevRec = CmNulo + 02;`

CmDbNextRecValid

Declaração `public const CmDbNextRecValid = CmNulo + 03;`

CmDbPrevRecValid

Declaração `public const CmDbPrevRecValid = CmNulo + 04;`

CmDbFindRec

Declaração `public const CmDbFindRec = CmNulo + 05;`

CmDbSearchRec

Declaração `public const CmDbSearchRec = CmNulo + 06;`

CmDbGoEof

Declaração `public const CmDbGoEof = CmNulo + 07;`

CmDbGoBof

Declaração `public const CmDbGoBof = CmNulo + 08;`

CmDbLocaliza

Declaração `public const CmDbLocaliza = CmNulo + 09;`

CmNewRecord

Declaração `public const CmNewRecord = CmNulo + 10;`

CmZeroizeRecord

Declaração `public const CmZeroizeRecord = CmNulo + 11;`

CmEvaluateRecord

Declaração `public const CmEvaluateRecord = CmNulo + 12;`

CmEditDlg

Declaração `public const CmEditDlg = CmNulo + 13;`

cmMyOK

Declaração `public const cmMyOK = CmNulo + 14;`

cmMyCancel

Declaração `public const cmMyCancel = CmNulo + 15;`

cmPrint

Declaração `public const cmPrint = CmNulo + 16;`

CmImport

Declaração `public const CmImport = CmNulo + 17;`

CmProcess

Declaração `public const CmProcess = CmNulo + 18;`

CmExecEndProc

Declaração `public const CmExecEndProc = CmNulo + 19;`

Descrição Usado para acessar a pesquisa associado ao campo

CmExecComboBox

Declaração `public const CmExecComboBox = CmNulo + 20;`

Descrição Usado para acessar a visao associada ao campo. Usado para visualizar CamposEnumerado e lista de forma geral

CmExecCommand

Declaração `public const CmExecCommand = CmNulo + 21;`

Descrição O comando vinculado ao campo focado e disparado para apliication.HanleEvent() se

CmCreate_Shortcut

Declaração `public const CmCreate_Shortcut = CmNulo + 22;`

CmVisualizar

Declaração `public const CmVisualizar = CmNulo + 23;`

CmExport_Stru

Declaração `public const CmExport_Stru = CmNulo + 24;`

CmExport

Declaração `public const CmExport = CmNulo + 25;`

CmInt

Declaração `public const CmInt = CmNulo + 29;`

TCmLivre

Declaração `public const TCmLivre = [CmVisualizar..CmInt];`

TCmCommands

Declaração `public const TCmCommands = [CmInt..255];`

TCmDb

Declaração
```
public const TCmDb = [CmDbNextRec , CmDbPrevRec , CmDbNextRecValid,
 CmDbPrevRecValid, CmDbFindRec , CmDbSearchRec , CmDbGoEof , CmDbGoBof ,
 CmDbLocaliza ];
```

TCmDbView

Declaração
```
public const TCmDbView = [cmMyOK , cmMyCancel , CmEditDlg ,
 CmEvaluateRecord , cmZeroizeRecord , CmNewRecord, CmProcess, cmPrint,
 CmExecEndProc, CmExecComboBox ];
```

TCmOutros

Declaração `public const TCmOutros =[cmPrint];`

CmNortSoft

Declaração `public const CmNortSoft = 50000;`

Descrição **

CmDbAddRec

Declaração `public const CmDbAddRec = CmNortSoft + 001;`

CmDbDeleteRec

Declaração `public const CmDbDeleteRec = CmNortSoft + 002;`

CmDbGetRec

Declaração `public const CmDbGetRec = CmNortSoft + 003;`

CmDbPutRec

Declaração `public const CmDbPutRec = CmNortSoft + 004;`

CmDbUpdateRec

Declaração `public const CmDbUpdateRec = CmNortSoft + 005;`

CmDbSearchTop

Declaração `public const CmDbSearchTop = CmNortSoft + 006;`

CmDbSearchKey

Declaração `public const CmDbSearchKey = CmNortSoft + 007;`

CmDbUsedRecs_Valid

Declaração `public const CmDbUsedRecs_Valid = CmNortSoft + 008;`

CmOkEscrevaParametrosDosRelatorios

Declaração `public const CmOkEscrevaParametrosDosRelatorios = CmNortSoft + 009;`

CmDbSelecionaIndice

Declaração `public const CmDbSelecionaIndice = CmNortSoft + 010;`

LivreCmVisualisa

Declaração `public const LivreCmVisualisa = CmNortSoft + 011;`

CmQuitInterno

Declaração `public const CmQuitInterno = CmNortSoft + 012;`

CmSobre

Declaração `public const CmSobre = CmNortSoft + 013;`

CmDbOnEnter

Declaração `public const CmDbOnEnter = CmNortSoft + 014;`

CmDbOnExit

Declaração `public const CmDbOnExit = CmNortSoft + 015;`

cmCores

Declaração `public const cmCores = CmNortSoft + 016;`

CmF7

Declaração `public const CmF7 = CmNortSoft + 017;`

CmDbLabel_DoubleClick

Declaração `public const CmDbLabel_DoubleClick = CmNortSoft + 018;`

cmDbView_DoubleClick

Declaração `public const cmDbView_DoubleClick = CmNortSoft + 019;`

CmDbOrdemCressante

Declaração `public const CmDbOrdemCressante = CmNortSoft + 020;`

CmDbOrdemDecrescente

Declaração `public const CmDbOrdemDecrescente = CmNortSoft + 021;`

CmDbSelecColunaAtual

Declaração `public const CmDbSelecColunaAtual = CmNortSoft + 022;`

CmMouseDownmbRightButton

Declaração `public const CmMouseDownmbRightButton = CmNortSoft + 023;`

CmReindex

Declaração `public const CmReindex = CmNortSoft + 024;`

CmCadastraImpressoraRede

Declaração `public const CmCadastraImpressoraRede = CmNortSoft + 025;`

CmInfoSystem

Declaração `public const CmInfoSystem = CmNortSoft + 026;`

cmPrintSemFormatar

Declaração `public const cmPrintSemFormatar = CmNortSoft + 027;`

CmDbDoBeforeInsert

Declaração `public const CmDbDoBeforeInsert = CmNortSoft + 028;`

CmDbDoBeforePost

Declaração `public const CmDbDoBeforePost = CmNortSoft + 029;`

CmDbDoBeforeDelete

Declaração `public const CmDbDoBeforeDelete = CmNortSoft + 030;`

CmDbDoAfterInsert

Declaração `public const CmDbDoAfterInsert = CmNortSoft + 031;`

CmDbDoAfterPost

Declaração `public const CmDbDoAfterPost = CmNortSoft + 032;`

CmDbDoAfterDelete

Declaração `public const CmDbDoAfterDelete = CmNortSoft + 033;`

CmTb_SelectRefCruzadaResume

Declaração `public const CmTb_SelectRefCruzadaResume = CmNortSoft + 034;`

CmTb_SelectSelect

Declaração `public const CmTb_SelectSelect = CmNortSoft + 035;`

CmTb_SelectResume

Declaração `public const CmTb_SelectResume = CmNortSoft + 036;`

CmRegistroValido

Declaração `public const CmRegistroValido = CmNortSoft + 037;`

CmCopyTo

Declaração `public const CmCopyTo = CmNortSoft + 038;`

CmCadastraImpressoraLocal

Declaração `public const CmCadastraImpressoraLocal = CmNortSoft + 039;`

CmSetAppending

Declaração `public const CmSetAppending = CmNortSoft + 040;`

CmStartTransaction

Declaração `public const CmStartTransaction = CmNortSoft + 041;`

CmCommit

Declaração `public const CmCommit = CmNortSoft + 042;`

CmRollback

Declaração `public const CmRollback = CmNortSoft + 043;`

CmOnCalcRecord_All

Declaração `public const CmOnCalcRecord_All = CmNortSoft + 044;`

CmTime

Declaração `public const CmTime = CmNortSoft + 045;`

cmEditaCores

Declaração `public const cmEditaCores = CmNortSoft + 046;`

cmSalvaCores

Declaração `public const cmSalvaCores = CmNortSoft + 047;`

cmHomePage

Declaração `public const cmHomePage = CmNortSoft + 048;`

CmDbPack

Declaração `public const CmDbPack = CmNortSoft + 049;`

FirstCmdNum

Declaração `public const FirstCmdNum = 4400;`

cmDMX

Declaração `public const cmDMX = FirstCmdNum;`

cmDMX_RollCall

Declaração `public const cmDMX_RollCall = cmDMX + 1;`

cmDMX_Ack

Declaração `public const cmDMX_Ack = cmDMX + 2;`

cmDMX_FieldAltered

Declaração `public const cmDMX_FieldAltered = cmDMX + 3;`

cmDMX_Draw

Declaração `public const cmDMX_Draw = cmDMX + 4;`

cmDMX_DrawData

Declaração `public const cmDMX_DrawData = cmDMX + 5;`

cmDMX_Lock

Declaração `public const cmDMX_Lock = cmDMX + 6;`

cmDMX_LockData

Declaração `public const cmDMX_LockData = cmDMX + 7;`

cmDMX_Unlock

Declaração `public const cmDMX_Unlock = cmDMX + 8;`

cmDMX_UnlockData

Declaração `public const cmDMX_UnlockData = cmDMX + 9;`

cmDMX_FixSize

Declaração `public const cmDMX_FixSize = cmDMX + 10;`

cmDMX_SetupRecord

Declaração `public const cmDMX_SetupRecord = cmDMX + 11;`

cmDMX_WrongKey

Declaração `public const cmDMX_WrongKey = cmDMX + 12;`

cmDMX_ZeroizeField

Declaração `public const cmDMX_ZeroizeField = cmDMX + 13;`

cmDMX_ZeroizeRecord

Declaração `public const cmDMX_ZeroizeRecord = cmDMX + 14;`

cmDMX_Enter

Declaração `public const cmDMX_Enter = cmDMX + 15;`

cmDMX_Left

Declaração `public const cmDMX_Left = cmDMX + 16;`

cmDMX_Right

Declaração `public const cmDMX_Right = cmDMX + 17;`

cmDMX_Home

Declaração `public const cmDMX_Home = cmDMX + 18;`

cmDMX_End

Declaração `public const cmDMX_End = cmDMX + 19;`

cmDMX_goto

Declaração `public const cmDMX_goto = cmDMX + 20;`

cmDMX_NextRow

Declaração `public const cmDMX_NextRow = cmDMX + 21;`

cmDMX_Up

Declaração `public const cmDMX_Up = cmDMX + 22;`

cmDMX_Down

Declaração `public const cmDMX_Down = cmDMX + 23;`

cmDMX_PgUp

Declaração `public const cmDMX_PgUp = cmDMX + 24;`

cmDMX_PgDn

Declaração `public const cmDMX_PgDn = cmDMX + 25;`

cmDMX_ScreenTop

Declaração `public const cmDMX_ScreenTop = cmDMX + 26;`

cmDMX_ScreenBottom

Declaração `public const cmDMX_ScreenBottom = cmDMX + 27;`

cmDMX_Top

Declaração `public const cmDMX_Top = cmDMX + 28;`

cmDMX_Bottom

Declaração `public const cmDMX_Bottom = cmDMX + 29;`

cmDMX_DoubleClick

Declaração `public const cmDMX_DoubleClick = cmDMX + 30;`

cmDMX_RecIndClicked

Declaração `public const cmDMX_RecIndClicked = cmDMX + 31;`

cmDMX_Reset

Declaração `public const cmDMX_Reset = cmDMX + 32;`

cmDMX_ScrollBarChanged

Declaração `public const cmDMX_ScrollBarChanged =cmDMX+33;`

cmDMX_InsertRec

Declaração `public const cmDMX_InsertRec = cmDMX + 34;`

cmPRN_NewPage

Declaração `public const cmPRN_NewPage = cmDMX + 40;`

cmPRN_EndPage

Declaração `public const cmPRN_EndPage = cmDMX + 41;`

cmPRN_SetOptions

Declaração `public const cmPRN_SetOptions = cmDMX + 42;`

cmPRN_LineFeed

Declaração `public const cmPRN_LineFeed = cmDMX + 43;`

cmPRN_FormFeed

Declaração `public const cmPRN_FormFeed = cmDMX + 44;`

cmPRN_Reset

Declaração `public const cmPRN_Reset = cmDMX + 45;`

cmUserScreen

Declaração `public const cmUserScreen = cmDMX + 51;`

cmToggleSound

Declaração `public const cmToggleSound = cmDMX + 52;`

cmToggleVideo

Declaração `public const cmToggleVideo = cmDMX + 53;`

cmBeep

Declaração `public const cmBeep = cmDMX + 54;`

cmChime

Declaração `public const cmChime = cmDMX + 55;`

cmPromptMsg

Declaração `public const cmPromptMsg = cmDMX + 56;`

cmBlinkMsg

Declaração `public const cmBlinkMsg = cmDMX + 57;`

cmDbMX_GetBuffer

Declaração `public const cmDbMX_GetBuffer = cmDMX + 58;`

cmDbMX_PutBuffer

Declaração `public const cmDbMX_PutBuffer = cmDMX + 59;`

DirectorySeparator

Declaração `public const DirectorySeparator :char = system.DirectorySeparator;`

Descrição A contante `DirectorySeparator` contém o caractere separador de diretório.

Lst

Declaração `public var Lst:  text ; static;`

onProcessMessages

Declaração `public const onProcessMessages : TOnProcedure = nil;`

Descrição

- O evento `onProcessMessages` é executado em CtrlSleep e deve ser iniciado para que possa processar as mensagens dos widgets que usão essa classe.

kbNoKey

Declaração `public const kbNoKey = 0;`

MessageBoxOff

Declaração `public const MessageBoxOff : Boolean = false;`

Descrição Se `MessageBoxOff` = true então não mostra o dialogo e torna o comando defaust

- Usada quando se quer despresar a ação do usuário e que ler os erros de um arquivo de erros. Normalmente deve ser usado nos programas controlados em linha de comando.

Métodos

CreateEnumField

Declaração
```
public class function CreateEnumField(ShowZ: boolean;
 AccMode,Default: LongInt;AItems: PSItem) : tString;
```

CreateTSItemFields

Declaração
```
public class function CreateTSItemFields(ATemplates: PSItem) :
 tString;
```

7.5 Constantes

SCmDbNextRec

Declaração `SCmDbNextRec = 'Próximo registro';`

SCmDbPrevRec

Declaração `SCmDbPrevRec = 'Registro Anterior';`

SCmDbNextRecValid

Declaração `SCmDbNextRecValid = 'Próximo registro válido';`

SCmDbPrevRecValid

Declaração `SCmDbPrevRecValid = 'Registro válido anterior';`

SCmDbFindRec

Declaração `SCmDbFindRec = 'Atualiza o registro atual';`

SCmDbSearchRec

Declaração `SCmDbSearchRec = 'SCmDbSearchRec';`

SCmDbGoEof

Declaração `SCmDbGoEof = 'Último registro';`

SCmDbGoBof

Declaração `SCmDbGoBof = 'Primeiro registro';`

SCmDbLocaliza

Declaração `SCmDbLocaliza = 'Localiza registro';`

SCmNewRecord

Declaração `SCmNewRecord = 'Novo registro';`

SCmZeroizeRecord

Declaração `SCmZeroizeRecord = 'Apaga o registro atual';`

SCmEvaluateRecord

Declaração `SCmEvaluateRecord = 'Grava o registro atual';`

SCmEditDlg

Declaração `SCmEditDlg = 'Edita o registro atual';`

ScmMyOK

Declaração `ScmMyOK = 'Ok';`

ScmMyCancel

Declaração `ScmMyCancel = 'Cancelar';`

ScmPrint

Declaração `ScmPrint = 'Imprimir';`

SCmImport

Declaração `SCmImport = 'Importar';`

SCmProcess

Declaração `SCmProcess = 'Processa';`

SCmExecEndProc

Declaração `SCmExecEndProc = 'SCmExecEndProc';`

Descrição Usado para acessar a pesquisa associado ao campo

SCmExecComboBox

Declaração `SCmExecComboBox = 'SCmExecComboBox';`

Descrição Usado para acessar a visao associada ao campo. Usado para visualizar CamposEnumerado e lista de forma geral

SCmExecCommand

Declaração `SCmExecCommand = 'SCmExecCommand';`

Descrição O comando vinculado ao campo focado e disparado para apliication.HanleEvent() se

SCmCreate_Shortcut

Declaração `SCmCreate_Shortcut = 'Cria atalho no desktop do windows';`

SCmVisualizar

Declaração `SCmVisualizar = 'Visualizar';`

SCmExport_Stru

Declaração `SCmExport_Stru = 'Exportar estrutura da tabela';`

Descrição Exporta a estrutura das consultas para o arquivo Schema.ini

SCmExport

Declaração `SCmExport = 'Exporta';`

Descrição Exporta a consulta seleciona para varios formatos de arquivos a serem implementados

SCmDbAddRec

Declaração `SCmDbAddRec = 'Adicionar registro';`

SCmDbDeleteRec

Declaração `SCmDbDeleteRec = 'Apagar registro selecionado';`

SCmDbGetRec

Declaração `SCmDbGetRec = 'Ler registro selecionado';`

SCmDbPutRec

Declaração `SCmDbPutRec = 'Gravar registro selecionado';`

SCmDbUpdateRec

Declaração `SCmDbUpdateRec = 'Atualizar registro selecionado caso tenha sido alterado';`

SCmDbSearchTop

Declaração `SCmDbSearchTop = 'Pesquisar primeira ocorrência a partir do topo da tabela';`

SCmDbSearchKey

Declaração `SCmDbSearchKey = 'Pesquisar primeira ocorrência a partir do inicio da tabela';`

SCmDbUsedRecs_Valid

Declaração `SCmDbUsedRecs_Valid = 'CmDbUsedRecs_Valid';`

SCmOkEscrevaParametrosDosRelatorios

Declaração `SCmOkEscrevaParametrosDosRelatorios = 'Escreva parâmetros dos relatórios';`

SCmDbSelecionaIndice

Declaração `SCmDbSelecionaIndice = 'Selecionar indice';`

SLivreCmVisualisa

Declaração `SLivreCmVisualisa = 'CmLivreCmVisualisa';`

SCmQuitInterno

Declaração `SCmQuitInterno = 'Quit interno';`

SCmSobre

Declaração `SCmSobre = 'Sobre';`

SCmDbOnEnter

Declaração `SCmDbOnEnter = 'CmDbOnEnter';`

SCmDbOnExit

Declaração `SCmDbOnExit = 'CmDbOnExit';`

ScmCores

Declaração `ScmCores = 'cmCores';`

SCmF7

Declaração `SCmF7 = 'Seleciona as opções para o campo selecionado';`

SCmDbLabel_DoubleClick

Declaração `SCmDbLabel_DoubleClick = 'CmDbLabel_DoubleClick';`

ScmDbView_DoubleClick

Declaração `ScmDbView_DoubleClick = 'cmDbView_DoubleClick';`

SCmDbOrdemCressante

Declaração `SCmDbOrdemCressante = 'Ordem cressante';`

SCmDbOrdemDecrescente

Declaração `SCmDbOrdemDecrescente = 'Ordem decrescente';`

SCmDbSelecColunaAtual

Declaração `SCmDbSelecColunaAtual = 'CmDbSelecColunaAtual';`

SCmMouseDownmbRightButton

Declaração `SCmMouseDownmbRightButton = 'CmMouseDownmbRightButton';`

SCmReindex

Declaração `SCmReindex = 'Cria indices dos arquivos';`

SCmCadastraImpressoraRede

Declaração `SCmCadastraImpressoraRede = 'Cadastra impressora da rede';`

SCmInfoSystem

Declaração `SCmInfoSystem = 'Informações do sistema';`

ScmPrintSemFormatar

Declaração `ScmPrintSemFormatar = 'cmPrintSemFormatar';`

SCmDbDoBeforeInsert

Declaração `SCmDbDoBeforeInsert = 'CmDbDoBeforeInsert';`

SCmDbDoBeforePost

Declaração `SCmDbDoBeforePost = 'CmDbDoBeforePost';`

SCmDbDoBeforeDelete

Declaração `SCmDbDoBeforeDelete = 'CmDbDoBeforeDelete';`

SCmDbDoAfterInsert

Declaração `SCmDbDoAfterInsert = 'CmDbDoAfterInsert';`

SCmDbDoAfterPost

Declaração `SCmDbDoAfterPost = 'CmDbDoAfterPost';`

SCmDbDoAfterDelete

Declaração `SCmDbDoAfterDelete = 'CmDbDoAfterDelete';`

SCmTb_SelectRefCruzadaResume

Declaração `SCmTb_SelectRefCruzadaResume = 'CmTb_SelectRefCruzadaResume';`

SCmTb_SelectSelect

Declaração `SCmTb_SelectSelect = 'CmTb_SelectSelect';`

SCmTb_SelectResume

Declaração `SCmTb_SelectResume = 'CmTb_SelectResume';`

SCmRegistroValido

Declaração `SCmRegistroValido = 'Registro válido';`

SCmCopyTo

Declaração `SCmCopyTo = 'Copiar para';`

SCmCadastraImpressoraLocal

Declaração `SCmCadastraImpressoraLocal = 'Cadastra impressora local';`

SCmSetAppending

Declaração `SCmSetAppending = 'CmSetAppending';`

SCmStartTransaction

Declaração `SCmStartTransaction = 'Inicia uma transação';`

SCmCommit

Declaração `SCmCommit = 'Confirma transação';`

SCmRollback

Declaração `SCmRollback = 'CmRollback';`

SCmOnCalcRecord_All

Declaração `SCmOnCalcRecord_All = 'Calcula todos os registros';`

SCmTime

Declaração `SCmTime = 'Time';`

ScmEditaCores

Declaração `ScmEditaCores = 'Edita cores';`

ScmSalvaCores

Declaração `ScmSalvaCores = 'Salva cores';`

ScmHomePage

Declaração `ScmHomePage = 'Gera documento no formato HTML do formulário atual';`

SCmDbPack

Declaração `SCmDbPack = 'Pack';`

sErr201

Declaração `sErr201 = 'Erro: %d - O valor %s está fora da faixa permitida para o campo. Faixa: [%d .. %d ]';`

Chapter 8

Unit mi.rtl.Consts.StrError

8.1 Descrição

-A unit `mi.rtl.Consts.StrError` implementa a classe `TStrError`(??) do pacote `mi.rtl`(??).

- **VERSÃO**:
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
    - 2021-12-02 -08:00 a 22:15 : Criado a unit `mi.rtl.Consts.StrError` e implementação da classe `TStrError`(??)
    - 2021-12-15 : Ajuste do método `TStrError.ErrorMessage4`(??);

8.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.Consts`(??)

8.3 Visão Geral

`TStrError Classe`

8.4 Classes, Interfaces, Objetos e Registros

TStrError Classe

Hierarquia

TStrError > `TConsts`(??) > `TTypes`(??) > TComponent

Descrição

A classe `TStrError` é usada para produzir texto informativo sobre o local onde o erro ocorreu.

Métodos

ErrorMessage

Declaração `public class function ErrorMessage(Const ErrorCode : SmallWord) : AnsiString; overload;`

Descrição A função `ErrorMessage` retorna um texto com o nome do erro passado por ErrorCode

- **PARÂMETRO**
  - **ErrorCode** - Número do erro
- **RETORNA**
  - Nome do erro
- **NOTA**
  - A versão abaixo é usada quando o sistema é windows, a mesma tem o nome do erro no linux e o nome do erro no windows separada com a palavra **ou**.
  - Quando a plataforma é linux a mensagem que aparece é os erros no linux.
- **LISTA DOS ERROS POSSÍVEIS**

ErrorMessage.inc (./units/include/windows/ErrorMessage.inc)

ErrorMessage.inc (./units/include/linux/ErrorMessage.inc)

```
0: Result := '000: Chamada inválida a função Result.';

{1..99 RESERVADO PARA ERROS DO DOS}
1: Result := 'EPERM 1: Operação não permitida';
2: Result := 'ENOENT 2 FileName ou diretório inexistente';
3: Result := 'ESRCH 3 Processo inexistente';
4: Result := 'EINTR 4 Chamada de sistema interrompida';
5: Result := 'EIO 5 Erro de entrada/saída';
6: Result := 'ENXIO 6 Endereço ou dispositivo inexistente';
7: Result := 'E2BIG 7 Lista de argumentos muito longa';
8: Result := 'ENOEXEC 8 Erro no formato exec';
9: Result := 'EBADF 9 Descritor de FileName inválido';
10: Result := 'ECHILD 10 Não há processos filhos';
11: Result := 'EAGAIN 11 Recurso temporariamente indisponível';
12: Result := 'ENOMEM 12 Não foi possível alocar memória';
13: Result := 'EACCES 13 Permissão negada';
14: Result := 'EFAULT 14 Endereço inválido';
15: Result := 'ENOTBLK 15 Dispositivo de bloco requerido';
16: Result := 'EBUSY 16 Dispositivo ou recurso está ocupado';
17: Result := 'EEXIST 17 FileName existe';
18: Result := 'EXDEV 18 Link entre dispositivos inválido';
19: Result := 'ENODEV 19 Dispositivo inexistente';
20: Result := 'ENOTDIR 20 Não é um diretório';
21: Result := 'EISDIR 21 É um diretório';
22: Result := 'EINVAL 22 Argumento inválido';
23: Result := 'ENFILE 23 Muitos FileNames abertos no sistema';
24: Result := 'EMFILE 24 Muitos FileNames abertos';
25: Result := 'ENOTTY 25 ioctl inapropriado para dispositivo';
26: Result := 'ETXTBSY 26 Área de texto ocupada';
27: Result := 'EFBIG 27 FileName muito grande';
28: Result := 'ENOSPC 28 Não há espaço disponível no dispositivo';
29: Result := 'ESPIPE 29 Procura ilegal';
30: Result := 'EROFS 30 Sistema de FileNames somente para leitura';
31: Result := 'EMLINK 31 Muitos links';
32: Result := 'EPIPE 32 Pipe quebrado';
33: Result := 'EDOM 33 Argumento numérico fora de domínio';
34: Result := 'ERANGE 34 Resultado numérico fora de alcance';
35: Result := 'EDEADLK 35 Evitado deadlock de recurso';
36: Result := 'ENAMETOOLONG 36 Nome de FileName muito longo';
37: Result := 'ENOLCK 37 Não há travas disponíveis';
38: Result := 'ENOSYS 38 Função não implementada';
39: Result := 'ENOTEMPTY 39 Diretório não vazio';
40: Result := 'ELOOP 40 Muitos níveis de links simbólicos';
41: Result := 'EWOULDBLOCK 41 Recurso temporariamente indisponível';
```

```
42: Result := 'ENOMSG 42 Não há mensagens do tipo desejado';
43: Result := 'EIDRM 43 Identificador removido';
44: Result := 'ECHRNG 44 Número do canal fora do intervalo';
45: Result := 'EL2NSYNC 45 Nível 2 não sincronizado';
46: Result := 'EL3HLT 46 Nível 3 parado';
47: Result := 'EL3RST 47 Nível 3 reiniciado';
48: Result := 'ELNRNG 48 Número de link fora da faixa';
49: Result := 'EUNATCH 49 Driver de protocolo não anexado';
50: Result := 'ENOCSI 50 Não há estrutura CSI disponível';
51: Result := 'EL2HLT 51 Parada de sistema nível 2';
52: Result := 'EBADE 52 Troca inválida';
53: Result := 'EBADR 53 Descritor de requisição inválido';
54: Result := 'EXFULL 54 Troca completa';
55: Result := 'ENOANO 55 Sem anode';
56: Result := 'EBADRQC 56 Código de requisição inválido';
57: Result := 'EBADSLT 57 Slot inválido';
58: Result := 'EDEADLOCK 35 Evitado deadlock de recurso';
59: Result := 'EBFONT 59 Formato do FileName fonte inválido';
60: Result := 'ENOSTR 60 Dispositivo não é um stream';
61: Result := 'ENODATA 61 Não há dados disponíveis';
62: Result := 'ETIME 62 Tempo expirado';
63: Result := 'ENOSR 63 Sem recursos de streams';
64: Result := 'ENOSR 64 Sem recursos de streams';
65: Result := 'ENOPKG 65 Pacote não instalado';
66: Result := 'EREMOTE 66 ObjectName é remoto';
67: Result := 'ENOLINK 67 Link foi cortado';
68: Result := 'EADV 68 Erro de aviso';
69: Result := 'ESRMNT 69 Erro de sr mount';
70: Result := 'COMM 70 Erro de comunicação ao enviar';
71: Result := 'EPROTO 71 Erro de protocolo';
72: Result := 'EMULTIHOP 72 Tentativa de Multi hop';
73: Result := 'EDOTDOT 73 Erro específico de RFS';
74: Result := 'EBADMSG 74 Mensagem inválida';
75: Result := 'EOVERFLOW 75 Valor muito grande para o tipo de dados definido';
76: Result := 'ENOTUNIQ 76 O nome não é único na rede';
77: Result := 'EBADFD 77 Descritor de FileName em mal estado';
78: Result := 'EREMCHG 78 Endereço remoto alterado';
79: Result := 'ELIBACC 79 Não foi possível acessar uma biblioteca compartilhada
80: Result := 'ELIBBAD 80 Acessando uma biblioteca compartilhado corrompida';
81: Result := 'ELIBSCN 81 Seção .lib corrompida em a.out';
82: Result := 'ELIBMAX 82 Tentando vincular em muitas bibliotecas compartilhada
83: Result := 'ELIBEXEC 83 Não foi possível executar uma biblioteca compartilha
84: Result := 'EILSEQ 84 Multi byte ou caractere largo inválido';
85: Result := 'ERESTART 85 Chamada de sistema interrompida deve ser reiniciada'
86: Result := 'ESTRPIPE 86 Erro de fluxos de pipe';
87: Result := 'EUSERS 87 Muitos usuários';
```

```
88: Result := 'ENOTSOCK 88 Operação socket em um FileName não-socket';
89: Result := 'EDESTADDRREQ 89 Endereço de destino requerido';
90: Result := 'EMSGSIZE 90 Mensagem muito longa';
91: Result := 'EPROTOTYPE 91 Tipo errado de protocolo para socket';
92: Result := 'ENOPROTOOPT 92 Protocolo não disponível';
93: Result := 'EPROTONOSUPPORT 93 Protocolo sem suporte';
94: Result := 'ESOCKTNOSUPPORT 94 Tipo socket sem suporte';
95: Result := 'EOPNOTSUPP 95 Operação sem suporte';
96: Result := 'EPFNOSUPPORT 96 Família de protocolo sem suporte';
97: Result := 'EAFNOSUPPORT 97 Família de endereços sem suporte pelo protocolo'
98: Result := 'EADDRINUSE 98 Endereço já em uso';
99: Result := 'EADDRNOTAVAIL 99 Não foi possível acessar o endereço requisitado

{100 A 149 ERROS DE ENTRADA E SAIDA (I/O)}
100: Result := '100: Erro ao ler o disco';//EndOfFile no delphi
101: Result := '101: Erro ao gravar no disco';//DiskFull no delphi
102: Result := '102: FileName não assinalado (falta ASSIGN) ';
103: Result := '103: FileName fechado';
104: Result := '104: O FileName fechado para entrada';
105: Result := '105: O FileName fechado para saída';
106: Result := '106: Formato numérico inválido ou incompatÍvel';
107: Result := '107: Disco cheio';
108..149:
     Result := '108..149: Reservado para I/O ';

{150..199 RESERVADO PARA ERROS CRITICOS}
150: Result := '150: Disco protegido';
151: Result := '151: UNIT desconhecida';
152: Result := '152: Drive não esta pronto';
153: Result := '153: Comando desconhecido';
154: Result := '154: Erro na CRC de dados';
155: Result := '155: Erro no drive solicitado pelo tamanho';
156: Result := '156: Erro no posicionamento de disco';
157: Result := '157: Tipo de meio desconhecido';
158: Result := '158: Setor não encontrado';
159: Result := '159: Impressora sem papel';
160: Result := '160: Erro de escrita no dispositivo de saída ( Impressora )';
161: Result := '161: Falta dispositivo de entrada ( Leitura )';
162: Result := '162: Falta hardware ( equipamento )';
163..199:
     Result := '163..199: RESERVADO PARA ERROS CRITICOS';

{200..255 RESERVADO PARA ERROS FATAL}
200: Result := '200: Divisão por zero';
201: Result := '201: Error na checagem de faixa';
202: Result := '202: Estouro no stack de memória';
```

```
203: Result := '203: Estouro no heap de memória';
204: Result := '204: Operação de pointer inválida';
205: Result := '205: Estouro em operação com ponto flutuante';
206: Result := '206: Erro de underflow com ponto-flutuante (Somente com 8087) '
207: Result := '207: Operação inválida com ponto flutuante';
208: Result := '208: Gerenciador de Overlay não instalado';
209: Result := '209: Erro da leitura no FileName de overlay';
210: Result := '210: ObjectName não inicializado';
211: Result := '211: Chamada a um MethodName abstrato';
212: Result := '212: Stream registration error';

213: Result := '213: Collection index out of range';
214: Result := '214: Collection overflow error';
215: Result := '215: Arithmetic overflow error';
216: Result := '216: General Protection fault';

{MarIcarai 217..255}
217: Result := '217: Tentativa de abrir um FileName aberto.';
218: Result := '218: Tentativa de excluir um registro excluido';
219: Result := '219: Tentativa de ler um registro excluído';
220: Result := '220: Outro usuário da rede alterou o registro';
221: Result := '221: Estrutura da tabela esta danificada';
222: Result := '222: Tentativa de gravar em um registro compartilhado sem que o

{ApCLiSvr.pas}
223: Result := '223: Evento executado por outro processo';
224: Result := '224: Servidor de API não instalado';

225: Result := '225: TTransaction.Commit esperado.';//TTransaction
226: Result := '226: Não é uma expressão válida';
227: Result := '227: Muitos parentese na expressão';
228: Result := '228: Muitos operadores na expressão';
229: Result := '229: Operador aritmético esperado na expressão';
230: Result := '230: O Número não pode ser lido na expressão';

REC_TOO_LARGE   :  Result  := 'Tamanho do registro em memória maior que o permi
REC_TOO_SMALL   :  Result  := 'Tamanho do Registro e muito longo';
KeyTooLarge     :  Result  := 'Tamanho da chave maior do que o maximo permitido
RecSizeMismatch :  Result  := 'Registro de dados incompatÍvel com a estrutura d
KeySizeMismatch :  Result  := 'Tamanho da pagina ou chave incompatÍvel com a es
MemOverflow     :  Result  := 'Não ha memória para os indice dos FileNames';
ArqIndexInconsistente
                :  Result  := 'FileName de indice inconsistente.';
238             :  Result := '238: O gerente de transacoes esta inativo'; //TTr
239             :  Result := '239: Excecao inesperada';
240: Result     := '240: Acesso negado ao FileName por falta de autorizacao de
```

```
241: Result     := '241: Registro não localizado'; // Erros retornados nas busc
242: Result     := '242: O evento OnEnter Retornou falso';
243: Result     := '242: O evento OnExit Retornou falso';
244
245             := '245 attempt_to_edit_a_record_not_selecting'
..
 254: Result := '238..254: RESERVADO PARA ERROS FATAIS';
255: Result := '255: ^C. Sistema abortado.';

ELSE Result := 'Erro indefinido Maior que 255';
```

- **REFERÊNCIAS**:
  - Lista de erros do Lazarus: https://wiki.lazarus.freepascal.org/RunError
  - FreePascal - acesso a FileNames. : https://www.freepascal.org/docs-html/rtl/system/ioresult.html
  - windows: https://docs.microsoft.com/pt-br/cpp/c-runtime-library/errno-constants?view=msvc-170
  - DOS: https://docs.microsoft.com/pt-br/cpp/c-runtime-library/errno-doserrno-sys-errlist-and-sys-nerr?view=msvc-170
  - Linux: https://man7.org/linux/man-pages/man3/errno.3.html

ErrorMessage

Declaração `public class function ErrorMessage(Const ErrorMsg : AnsiString) : AnsiString; overload;`

Descrição O método `ErrorMessage` receber uma mensagem e retorna a mesagem formatada.

ErrorMessage

Declaração `public class function ErrorMessage(const Sender: TObject; Const ErrorMsg : AnsiString) : AnsiString; overload;`

ErrorMessage

Declaração `public class function ErrorMessage(const Sender: TObject;Const aMethodName, aFileName, AFieldName:AnsiString;aMsg:AnsiString):AnsiString; Overload;`

ErrorMessage

Declaração `public class function ErrorMessage(const Sender: TObject;Const aMethodName, aFileName, AFieldName:AnsiString;aCodError:SmallInt):AnsiString; Overload;`

ErrorMessage8

Declaração `public class function ErrorMessage8(Const aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName, aMessage, aProcedure_or_Function :AnsiString):AnsiString; Overload;`

ErrorMessage7

Declaração `public class function ErrorMessage7(aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aCodError:SmallInt):AnsiString; Overload;`

ErrorMessage7

Declaração `public class function ErrorMessage7(aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aMessage:AnsiString):AnsiString; Overload;`

ErrorMessage6

Declaração `public class function ErrorMessage6(aModule, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aCodError:SmallInt):AnsiString; Overload;`

ErrorMessage5

Declaração `public class function ErrorMessage5(aModule, aUnit, aObjectName, aMethodName :AnsiString; aCodError:SmallInt):AnsiString; Overload;`

ErrorMessage5

Declaração `public class function ErrorMessage5(aModule, aUnit, aObjectName, aMethodName :AnsiString; aMsgError:AnsiString):AnsiString; Overload;`

ErrorMessage4

Declaração `public class function ErrorMessage4(Const aModule, aUnit, Procedure_or_function:AnsiString; aCodError:SmallInt):AnsiString; Overload;`

Descrição A class `ErrorMessage4` formata um texto com os parãmetros passado

ErrorMessage4

Declaração `public class function ErrorMessage4(Const aModule, aUnit, Procedure_or_function:AnsiString; aMessage:AnsiString):AnsiString; Overload;`

Descrição A class `ErrorMessage4` formata um texto com os parãmetros passado

Chapter 9

Unit mi.rtl.Consts.StringList

9.1 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.consts.StringListBase`(??)
- `mi.rtl.Consts`(??)

9.2 Visão Geral

`TMiStringList Classe`

9.3 Classes, Interfaces, Objetos e Registros

TMiStringList Classe

Hierarquia

TMiStringList > `TStringListBase`(??) > TStringList

Descrição

A class `TMiStringList` implementa a navegação como se tivesse navegando em arquivos usando os métodos `NextKey`(??),`Prevkey`(??) etc...

- **NOTA**
  - Usando quando quero manter uma lista de registros ordenada.

Métodos

AddKey

Declaração `public Function AddKey(WKey:String;wNr:Longint):Boolean;`

BofKey

Declaração `public Function BofKey: Boolean;`

Descrição Posiciona no inicio do bloco de registro do tipo default

LastKey

Declaração `public Function LastKey: Boolean;`

Descrição Posiciona no fin do bloco de registro do tipo default

EofKey

Declaração `public Function EofKey: Boolean;`

Descrição Posiciona no fin do bloco de registro do tipo default

PrevKey

Declaração `public Function PrevKey: Boolean;`

Descrição Posiciona no registro anterior do tipo default

Chapter 10

Unit mi.rtl.Consts.StringListBase

10.1 Descrição

-A unit `mi.rtl.Consts.StringListBase` implementa a classe `TStringListBase`(**??**) do pacote `mi.rtl`(**??**).

- **VERSÃO**:
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
    - **2022-01-25** -08:00 a 12:00 - Criado a unit `mi.rtl.Consts.StringListBase` e implementação da classe `TStringListBase`(**??**)
  - **2022-05-17**
    - T12 Criar método CopyFrom

10.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.Consts`(**??**)

10.3 Visão Geral

`TStringListBase Classe`

10.4 Classes, Interfaces, Objetos e Registros

TStringListBase Classe

Hierarquia

TStringListBase > TStringList

Propriedades

KeyMaster

Declaração `public property KeyMaster : String Read _KeyMaster Write SetKeyMaster;`

OkDestroy_Object

Declaração `public property OkDestroy_Object : Boolean Write _OkDestroy_Object;`

Descrição Redefini para poder deletar o objeto associando

Campos

Index_Currente

Declaração `protected Index_Currente: Integer;`

OkBof

Declaração `protected OkBof: Boolean;`

Descrição Inicio de arquivo

okEof

Declaração `protected okEof: Boolean;`

Descrição Fim de arquivo

Nr

Declaração `public Nr: PtrInt;`

Descrição Número do registro corrente

Métodos

Create

Declaração `public constructor Create(ALimit: longint;AOrdem:Boolean ); overload; virtual;`

ClearKey

Declaração `public Function ClearKey: Boolean;`

IndexOf

Declaração `public function IndexOf(const S: string): Integer; override;`

Descrição Redefini porque a instância anterior não funciona com caractere #254

Delete

Declaração `public procedure Delete(Index: Integer); override;`

Descrição Redefini para poder deletar o objeto associando

Destroy

Declaração `public destructor Destroy; override;`

Find

Declaração `public function Find(const S: string; Out Index:  Integer): Boolean; override;`

FindKey

Declaração `public Function FindKey(WKey:String):Boolean;`

NextKey

Declaração `public Function NextKey:  Boolean;`

SearchKey

Declaração `public Function SearchKey(WKey:String):Boolean;`

NewStr

Declaração `public Function NewStr(S : String):Boolean; overload;`

Append

Declaração `public Function Append(S : String):Boolean;`

AddSItem

Declaração `public procedure AddSItem(P : TConsts.PSItem; ConvertIdioma : TConsts.TConvertIdioma; OkDisposeSItems:Boolean); Overload;`

AddSItem

Declaração `public procedure AddSItem(P : TConsts.PSItem); Overload;`

Descrição Adiciona a lista passada por aSItem e desaloca a lista se OkDisposeSItems = true;

CloneSItems

Declaração `public Function CloneSItems(Const Items:`
`TConsts.PSItem):TConsts.PSItem;`

CopyTemplateFrom

Declaração `public Function CopyTemplateFrom(Const aTemplate:TConsts.tString):`
`TConsts.tString;`

PListSItem

Declaração `public Function PListSItem:  TConsts.PSItem;`

Chapter 11

Unit mi.rtl.files

11.1 Descrição

- A Unit `mi.rtl.files` contém as funções https://techlib.wiki/definition/wrapper.html (Wrapper) para os sistemas operacionais **Win32 Win64** e **Linux x86_64** reconhecidos pelo free pascal.
  - **OBJETIVO**:
    - Evitar de alterar todos os códigos escritos para a plataforma windows e por isso mantenho o mesmo comportamento do windows.
  - **VERSÃO**:
    - Alpha - 0.5.0.687
  - **NOTA**:
    - https://wiki.freepascal.org/Writing_portable_code_regarding_the_processor_architecture (Veja o link de como escrever código portátil em relação à arquitetura do processador?);
    - Só devo usar units https://man7.org/linux/man-pages/man2/syscalls.2.html (syscalls) do Linux ou https://docs.microsoft.com/pt-br/windows/win32/apiindex/windows-api-list (Windows) caso não encontre a mesma pronta nos projetos lazarus ou Free Pascal.
  - **REFERÊNCIA**
    - https://wiki.freepascal.org/Multiplatform_Programming_Guide (Guia de programação multiplataforma);

ystem': procedures and functions (https://www.freepascal.org/docs-html/rtl/system/index-5.html)

  - **CÓDIGO FONTE**:

*

– **HISTÓRICO**

* Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br

  · 2021-10-21 08:00 - Data em que essa unit `mi.rtl.files` foi criada.

  · 2021-11-02 15:42 - Escolha do projeto **pasdoc** para criar documento do pacote `mi.rtl(??)`.

  · 2021-11-04 08:37 - Implementação da função **SysFileOpen()**

  · 2021-11-04 14:00 - Implementação da função **SysSetResult()**

  · 2021-11-04 14:30 - Implementação da função **SetFileMode()**

  · 2021-11-04 15:30 - Documentar a unit `mi.rtl.files` e organizar sessão de constantes, variáveis e funções.

  · 2021-11-04 21:00 - Criar exemplo de uso das funções **SysFileOpen** e **SysFileClose**.

  · 2021-11-05 21:30 - Revisar documentação desta nas funções: **SysSetResult**, ...

  · 2021-11-12 08:56 - Procurar bug da função **SysFileOpen** na máquina windows.

  · Eureca. Resolvi o problema da função **SysFileOpen**.

  · O problema da função **SysFileOpen** estava na forma como no windows a função SysUtils.fileOpen trabalha.

  · Caso ocorra um erro a função **SysUtils.fileOpen** retorna high(THandle).

  · Para corrigir precisei modificar a função **SysSetResult**.

  · 2021-11-12 16:56 - Documentar a unit `mi.rtl.files` e criar a função **CopyFile()**.

  · 2021-11-12 18:05 - Criar a função SysFileSetSize para truncar o arquivo e documenta-la.

  · 2021-11-13

  · Criado a class `TFiles(??)` herdade de `TConsts(??)` com propósito em encapsular as funções de acesso ao sistema operacional.

  · Criar método **TFiles.ErrorMessage()**;

  · Criar método `TFiles.SetLastError(??)()`;

  · Criar método `TFiles.SetResult(??)()`

  · Criar método `TFiles.CopyFile(??)()`

  · Criar método `TFiles.SetFileMode(??)()`

· Criar método `TFiles.FileOpen(??)`**(5 parametro)**

· Criar método `TFiles.FileOpen(??)`**(3 parâmetro)**

· Criar método `TFiles.FileClose(??)()`

· Criar método `TFiles.FileTruncate(??)()`

· Criar método `TFiles.FileCreate(??)()`

· 2021-11-15

· O método `TFiles.FileCreate(??)()` não está obedecendo o mapa de bits FileMode() checar o porque:

· Solução:

· A função SysUtils.FileCreate precisa do fmCreate na criação do arquivo.

· Após criar o arquivo o mesmo deve ser fechado e aberto novamente com o mapa de bits **mode** e **shareMode** passado no parâmetro.

· Criar método `TFiles.FileSeek(??)()`

· Criar método `TFiles.FileRead(??)()`

· 2021-11-16

· O método SysFileSeek não gerar erro se o ponteiro do arquivo for inválido.

· Para contornar devo fazer a crítica se o ponteiro é maior que zero e menor que fileSize.

· Essa solução não atende porque não fileSeek não tem o nome do arquivo.

· Entendendo porque SysUtils.FileSeek não dar erro quando se tenta posicionar além do fim do arquivo:

· https://man7.org/linux/man-pages/man2/lseek.2.html

· No linux **lseek()** permite que o deslocamento do arquivo seja definido além do final do arquivo (mas isso não altera o tamanho do arquivo). Se os dados forem posteriormente escrito neste ponto, leituras subsequentes dos dados no gap (um "buraco") retorna bytes nulos ('\ 0') até que os dados sejam realmente escrito na lacuna.

· Criar método `TFiles.FileSize(??)()`

· Criar exemplo de uso do método `TFiles.FileSize(??)()`

· Criar exemplo de uso do método `TFiles.FileSeek(??)()`

· 2021-11-17

· 08:30 a 10:38 - Criar exemplo de uso do método `TFiles.FileRead(??)()`

· 11:36 a 11:48 - Criar método `TFiles.FileWrite(??)()`

· 11:50 a 12:21 - Criar exemplo de uso do método `TFiles.FileWrite(??)()` - Falta testar.

· 14:15 a 15:23 - Testar o exemplo de uso da função `TFiles.FileWrite(??)`. ok.

· 2021-11-21

· 10:30 a 11:10

· Criar classe método - `TFiles.FileExists(??)()`

· Criar classe método - `TFiles.DirectoryExists(??)()`

· 2021-11-22

· 10:00 a 10:06 - Criar classe método : `TFiles.CreateDir(??)()`

· 10:29 a 11:10 - Criar classe método : **TFiles.SysGetDriveType(aPath : AnsiString): TDriveType;**

· 11:11 a 12:02 - Criar classe método : **DuplicateHandle(hSourceHandle: LongInt;Var lpTargeTHandle: Longint) : Longint);

· 14:10 a 15:16 - Criar classe método : **FileFlushBuffers(Handle: THandle): Longint;**

· 15:44 a 17:32 - Criar classe método : **LockFile(_Handle:THandle; _LockStart, _LockLength: Int64): LongInt;**

· Não encontrei no linux o equivalente ao Windows.LockFile

· 15:44 a 17:32 - Criar classe método : **UnLockFile(_Handle:THandle; _LockStart, _LockLength: Int64): LongInt;**

· Não encontrei no linux o equivalente ao Windows.unLockFile

· 2021-12-01

· 11:42 a ??:?? : Implementar a função Is_TFileOpen

· 2021-12-02

· 20:00 a 22:15 : Implementei a classe `TStrError(??)`

· 20211230

· 15:30 a 16:10 : Criar método GetTempFileName

· 20220111

· 17:10 - Criar método shellExecute

11.2 Uses

- Classes
- dos
- SysUtils
- crt
- FileUtil
- mi.rtl.types(??)
- mi.rtl.Consts(??)
- mi.rtl.Consts.StrError(??)

11.3 Visão Geral

TFiles Classe

11.4 Classes, Interfaces, Objetos e Registros

TFiles Classe

Hierarquia

TFiles > TConsts(??) > TTypes(??) > TComponent

Descrição

no description available, TConsts description followsA classe TConsts declara todas as constantes globais do pacote MarIcarai

Métodos

IoResult

Declaração `public class Function IoResult:  Integer;`

Descrição A função IoResult captura o estados de system.ioResult e atualiza TaStatus(??)

CtrlSleep

Declaração `public class procedure CtrlSleep(Const Delay: Cardinal);`

Descrição

- A função `CtrlSleep` dar opção para que a aplicação execute outras tarefas caso ela se encontre em estado de espera.
  - **PARÂMETRO**
    - **Delay** - Tempo em milissegundo que deve aguardar**.

Set_CTRL_SLEEP_ENABLE

Declaração `public class Function Set_CTRL_SLEEP_ENABLE(Const aEnable: Boolean):Boolean;`

Descrição

- A função `Set_CTRL_SLEEP_ENABLE` habilita ou não a função `CtrlSleep(??)`.
  - **PARÂMETRO**
    - **aEnable**
      - Se **True** habilita `CtrlSleep`(**??**);
      - Se **false** desabilita o método `CtrlSleep`(**??**);
  - **RETORNA**
    - o valor anterior da variável `CTRL_SLEEP_ENABLE`(**??**);

ReadKey

Declaração `public class function ReadKey: AnsiChar;`

SetLastError

Declaração `public class procedure SetLastError(aCodeError: integer);`

Descrição

- A procedure `SetLastError` atualiza as variáveis globais `LastError(??)` e `OK(??)`
  - **PARÂMETROS**
    - aCodeError:`integer`(**??**) - Código do erro.
  - **EXEMPLO**

```
procedure tes_SetLastError();
Begin
  //Executa a procedure SetLastError

  SetLastError(2);
  showMessage(ErrorMessage(LastError));

end;
```

SetResult

Declaração `public class function SetResult(aHandle: THandle ; aSuccess: Boolean): Longint; overload;`

Descrição

- A função `SetResult` captura o último erro se o parâmetro **aSucesso=false** ou o **aHandle for inválido** e retorna **0 (zero)** se **aSucesso=true** e o *aHandle for válido.*
  - A função `SetResult` atualiza a variável global `LastError(??)` e a variável global `ok(??)`:
  - **Plataformas testadas**
    - win32
    - win64
    - linux

– **PARÂMETROS**

* **aHandle** - O handle do arquivo retornado pela ultima chamada ao sistema operacional.
* **aSucesso** - Recebe **true** se sucesso ou **false** se fracasso na última chamada ao sistema operacional.

– **RETORNA**

* O conteúdo da variável global `LastError(??)`.

– **NOTA**

* No windows, quando ocorre um erro o handle é igual = high(`THandle(??)`) por isso é necessário passar o handle do arquivo na chamada a `SetResult`().

CopyFile

Declaração `public class function CopyFile(lpExistingFileName, lpNewFileName: AnsiString; aExceptionOnError: boolean): Integer;`

Descrição

- A função `CopyFile` copia o arquivo passado por **alpExistingFileName** para o arquivo passo por **lpNewFileName**.

– **PARÂMETROS**

* **alpExistingFileName**:AnsiString - Nome do arquivo a ser copiado;
* **lpNewFileName**:AnsiString - Nome do arquivo destino da cópia;
* **aExceptionOnError**:boolean - **True** se o sistema deve gera exceção ou **false** se o sistema não deve gera exceção.

– **RETORNO**

* `Integer(??)` - Código do erro ou 0 (zero) se a cópia for feita com sucesso.
* Caso o parâmetro **aExceptionOnError** = true então a exceção deve ser tratada pela rotina que o chamou.

– **EXEMPLO**

```
procedure TFormTests.Button_tes_CopyFileClick(Sender: TObject);

  // Este procedimento faz duas cópia do arquivo index.html

  Var
    err : TFiles.integer;
Begin
  with TFiles do
    err := CopyFile('index.html','index.bak1',false);

  with TFiles do
  if err = 0
  Then showMessage('Copia 1 feita com sucesso.')
   else showMessage(ErrorMessage(err));

  with TFiles do
  try
    CopyFile('index.html','index.bak2',true);
    showMessage('Copia 2 feita com sucesso.') ;
  Except
    on E:Exception do
     ShowMessage(e.Message);
  end;
end;
```

SetFileMode

Declaração `public Class function SetFileMode(aFileMode:word):word;`

Descrição

- A função `SetFileMode` modifica o valor de `FileMode`(**??**) e retorna o valor do `FileMode`(**??**) anterior;

  – **PARÂMETROS**

    * **aFileMode** é o modo de abertura do arquivo.

  – **RETORNA**

    * O `FileModeAnt`(**??**);

– **NOTA**

* A variável pública `FileModeAnt`(**??**) é igual ao resultado desta função.

SetStateFileMode

Declaração `public class Function SetStateFileMode(Const AState: Longint; Const Enable: boolean):Boolean;`

Descrição

- Seta `FileMode`(**??**) e retorna o estado anterior do Mapa de bits passado por aState

GetStateFileMode

Declaração `public class function GetStateFileMode(Const AState: Longint): Boolean;`

Descrição

- Ler o estado do File Mode

FileOpen

Declaração `public class function FileOpen(const FileName: AnsiString; const Mode: Longint; const ShareMode : Cardinal; out Handle: THandle): Longint; Overload;`

Descrição

- **Abre o arquivo passado pelo parâmetro FileName**

– **PARÂMETROS**

* **FileName** - Nome do arquivo a ser aberto;
* **mode** - Modo de abertura. Valor possível veja `TFileMode`(**??**);
* **attribute** - Atributo de abertura do arquivo;
* **Flags** - flag de abertura do arquivo;
* **Handle** - Se tiver sucesso retorna nesta variável o número do arquivo aberto;

– **RETORNA**

  * `LongInt`(**??**) Zero se sucesso ou o código do erro se fracasso.

– **NOTA**:

  * Possiveis erros pode ser visto na função ErrorMessage();

– **EXEMPLOS DE USO**

```
procedure TFormTests.Button_Test_OpenFile_exclusive_modeClick(Sende

  procedure Test_OpenFile_exclusive(aFileName:AnsiString);
  Var
    Err : TFiles.integer;
    h,
    h1 : TFiles.THandle;
  Begin
    with TFiles do
      Err := FileOpen (aFileName, FmReadWrite or FmDenyALL  or fmSh

    with TFiles do
    if Err = 0
    Then Begin
           ShowMessage('Teste da função SysFileOpen retornou true')

           Err := FileOpen (aFileName,FmReadWrite or FmDenyALL  or
           if Err = 0
           Then Begin
                  FileClose(h1);
                  ShowMessage('Teste da função SysFileOpen retornou
                end
           else ShowMessage('Error: '+ErrorMessage(Err));

           FileClose(h);
    end
    else ShowMessage('Error: '+ErrorMessage(Err));

  End;

begin
  Test_OpenFile_exclusive('index.html');
end;
```

– **REFERÊNCIAS**

* https://www.freepascal.org/docs-html/rtl/sysutils/fileopen.html (FileOpen);
* `fmOpenRead`(**??**);
* `FileClose`(**??**);
* `THandle`(**??**).

FileOpen

Declaração `public Class function FileOpen(const FileName: AnsiString; out Handle: THandle): Longint; Overload;`

Descrição

- **Abre o arquivo passado pelo parâmetro FileName**

– **PARÂMETROS**

* FileName **Nome do arquivo a ser aberto;**
* **mode** - Modo de abertura;
* **Handle** - Se result = 0 o Handle contém o número do arquivo aberto, caso contrário, retorna `HANDLE_INVALID`(**??**)**;

– **RETORNA**

* 0 (zero) se sucesso ou o código do erro se fracasso;

– **NOTA**:

* Possiveis erros pode ser visto na função ErrorMessage();

– **EXEMPLO DE USO**

```
procedure TFormTests.TestSysOpenFileClick(Sender: TObject);
Var
  Err : TFiles.Longint;
  h : TFiles.THandle;
Begin
  with TFiles do
    Err := FileOpen ('index.html',fmOpenRead,h);

  with TFiles do
  if Err = 0 Then
  Begin
    FileClose(h);
    ShowMessage('Teste da função FileOpen retornou true')
  end
  else ShowMessage('Error: '+ErrorMessage(Err));
End;
```

- **REFERÊNCIAS**
  - https://www.freepascal.org/docs-html/rtl/sysutils/fileopen.html (FileOpen);
  - `fmOpenRead`(??);
  - `FileClose`(??);
  - `THandle`(??).

FileClose

Declaração `public Class function FileClose(Handle: THandle): Longint;`

Descrição

- A função `FileClose` fecha o arquivo passando por Handle.
  - **PARÂMETRO**
    - **Handle** - Número do arquivo aberto por `FileOpen`(??).
  - **RETORNA**
    - Zero de tiver sucesso ou o código do erro se não conseguir fechar o arquivo.
  - **REFERÊNCIAS**
    - https://www.freepascal.org/docs-html/rtl/sysutils/FileClose.html (FileClose);

FileTruncate

Declaração `public class function FileTruncate(Handle:THandle;NewSize:Int64 ): Longint;`

Descrição

- A função `FileTruncate` reduz o tamanho do arquivo para o tamanho passado pelo parâmetro **NewSize**.

  - **PARÂMETROS**

    * Handle:`THandle`(**??**) - Handle do arquivo a ser truncado.
    * NewSize:`Int64`(**??**) - Tamanho do arquivo a se truncado.

  - **RETORNO**

    * `Longint(??)` - 0(zero) se sucesso ou o código do erro se fracasso.

  - **REFERÊNCIA**

    Truncate file (https://www.freepascal.org/docs-html/rtl/sysutils/filetruncate.htm

  - **EXEMPLO**

```
procedure TFormTests.Button_tes_FileTruncateClick(Sender: TObject);
  // Este procedimento Trunca o arquivos 'index.html' para 100 byte
  Var
    aHandle: TFiles.THandle;
    NewSize: TFiles.word;
    err    : TFiles.integer;
Begin
  NewSize := 100;

  with TFiles do
    err := FileOpen('index.html',fileMode,aHandle);

  with TFiles do
  if  err = 0
  then begin
         //Executa a função SysFileTruncate
```

```
        err := FileTruncate(aHandle,NewSize);

        if err = 0
        then showMessage('O arquivo foi truncado para 100 bytes')
        else ShowMessage('Error: '+ErrorMessage(err));
      end
  else ShowMessage('Error: '+ErrorMessage(err));
end;
```

FileCreate

Declaração `public class function FileCreate(FileName: AnsiString; Mode: Longint; ShareMode : Cardinal; out Handle: THandle): Longint; overload;`

Descrição

- A função `FileCreate` cria um novo arquivo e retorna um identificador para ele ou código do erro houver fracasso.
  - **PARÂMETROS**
    - **FileName**: AnsiString - Nome do arquivo a ser criado;
    - **Mode**: `Longint`(**??**) - Modo de criação do arquivo. Veja `FileMode`(**??**) para mais informações;
  - **RETORNO**
    - `THandle`(**??**) - Handle do arquivo criado;
    - `LongInt`(**??**) - 0 (zero se sucesso ou o código do erro se fracasso.
  - **EXEMPLO**

```
procedure TFormTests.Button_test_FileCreateClick(Sender: TObject);

  function tes_FileCreate(FileName:AnsiString;out aHandle:THandle):
  Begin
    with TFiles do
      result := FileCreate(FileName,fmOpenReadWrite or fmShareExclu
  end;
```

```
var
  aHandle :  TFiles.THandle;
begin

  with TFiles do
  if tes_FileCreate('text.txt',aHandle) = 0
  then begin
         showMessage('Arquivo text.txt criado na pasta corrente.');
         FileClose(aHandle);
       end;
end;
```

DeleteFile

Declaração `public class function DeleteFile(const FileName :  AnsiString): SmallInt ;`

FileSize

Declaração `public class function FileSize( FileName:  string; out Count : int64):longint; overload;`

Descrição

- A função `FileSize` retorna o tamanho do arquivo em bytes.
  - **PARÂMETROS**
    - FileName: AnsiString - Nome do arquivo;
    - Count: `Int64`(**??**) - Número de bytes do arquivo passo do **FileName**.
  - **RETORNO**
    - `longint`(**??**) - 0 (zero) se sucesso ou o código do erro se houver fracasso;
    - **Count** - Número de bytes do arquivo.
  - **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_Test_FileSizeExecute(Sender: TObjec
    // Este procedimento obtem o tamanho do arquivo em bytes do ar
  var
    FileName:AnsiString;
    Count:Int64;
    err:Longint;
Begin
  with TFiles do
  begin
    FileName := 'index.html';
    err := FileSize(FileName,Count);
    if err <> 0
    Then showMessage(ErrorMessage(err))
    else showMessage('Tamanho do arquivo é: '+intToStr(Count));
  end;
end;
```

FileSize

Declaração `public class function FileSize( FileName:  string):int64; overload;`

FileSizes

Declaração `public class function FileSizes(Mask:  AnsiString;out aFileSize:Int64):  Longint; overload;`

Descrição O função `FileSizes` retorna em aFileSize a soma de todos os arquivos que satisfaça a mascara em path;

- **NOTA**
  - Se houver error retorna o código do error em `FileSize`(**??**)

FileSeek

Declaração `public class function FileSeek(const Handle:THandle; Const FOffset :  Int64; Origin:  LongInt; out NewPos:  Int64):  LongInt;`

Descrição

- A função `FileSeek` posiciona o ponteiro do arquivo na posição FOffSet começando da origem.
  - **PARÂMETROS**
    - **Handle:** `THandle(??)` - Handle do arquivo;
    - **FOffset:** `Int64(??)` - Ponteiro do arquivo a ser posicionado;
    - **Origin:** `LongInt(??)` - Origem do calculo da posição do arquivos pode ser:
      - `TConsts.fsFromBeginning(??)`;
      - `TConsts.fsFromCurrent(??)`;
      - `TConsts.fsFromEnd(??)` ;
    - **NewPos:** `Int64(??)` - Se tiver sucesso a função retorna neste parametro a nova posição do arquivo.
  - **RETORNO**
    - `LongInt(??)` - 0 (zero se sucesso ou código do erro se fracasso.
      - Em **NewPos** retorna o número da posição atual do arquivo.
    - **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_Test_FileSeekExecute(Sen
  // Este procedimento posiciona o cursor no final do a
   Var
    err : TFiles.integer;
    h   : TFiles.THandle;
    NRec,Count : TFiles.Int64;
Begin
  with TFiles do
  begin
    err := FileOpen('index.html',h);
    if (err = 0) and (fileSize('index.html',Count) = 0)
    Then Begin
           //Posiciona no final do arquivo
           err := FileSeek(h,Count,fsFromBeginning,NRec
```

```
            if err <> 0
            Then ShowMessage(ErrorMessage(err));
            FileClose(h);
          end
      else ShowMessage(ErrorMessage(err));
    end;
  end;
```

- **REFERÊNCIA**:

  `FileSeek` (https://www.freepascal.org/docs-html/rtl/sysutils/fileseek.html)

FileRead

Declaração `public class function FileRead(const Handle: THandle; out Buffer; const Count: Int64; out BytesRead: int64): LongInt;`

Descrição

- O método `FileRead` ler **Count** bytes do arquivo passado pelo **Handle** e retorna o número de bytes lidos em **BytesRead**

  - **PARÂMETROS**

    * **Handle:** `THandle(??)` - Handle do arquivo
    * **Out Buffer** - Buffer para onde os dados devem ser salvos;
    * **Count:** `Int64(??)` - Número de bytes a ser lido para o buffer;
    * **Out BytesRead**: `Int64`(??) - Número de Bytes lidos efetivamente.

  - **RETORNO**

    * `Longint(??)` - 0 (zero se sucesso ou o código do erro se fracasso;
    * Em **Buffer** os dados lidos do arquivo;
    * Em **BytesRead** Retorna o número de bytes lidos efetivamente.

* **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_Test_FileReadExecute(Sen
  // Este procedimento ler os últimos 10 bytes do arqui
  Const Size = 10;
  Var
    err  : TFiles.integer;
    h    : TFiles.THandle;
    NRec,
    Count : TFiles.Int64;
    s     : String[255];
    BytesLidos: Int64;
Begin
  with TFiles do
  begin

    err := FileOpen('index.html',h);
    if (err = 0)
    Then Begin
           err := fileSize('index.html',Count);
           if ( err = 0)
           Then begin
                  //Posiciona no final do arquivo
                  err := FileSeek(h,Count-Size-length(LF
                  if err <> 0
                  Then ShowMessage(ErrorMessage(err))
                  Else Begin
                         err := FileRead(h,s[1],Size+len
                         if (err = 0) and (BytesLidos =
                         Then Begin
                                s[0] := chr(Size);
                                ShowMessage('Bytes Lidos
                              end
                         else ShowMessage('Ponteiro do ar
                       end;

           end
           else ShowMessage(ErrorMessage(err));
           FileClose(h);
         end
    else ShowMessage(ErrorMessage(err));
  end;
end;
```

– **REFERÊNCIA**

`FileRead` (https://www.freepascal.org/docs-html/rtl/sysutils/fileread.html)

FileWrite

Declaração `public class function FileWrite(const Handle: THandle; const Buffer; const Count: Int64; out BytesWrites: int64): LongInt;`

Descrição

- O método `FileWrite` grava **Count** bytes do arquivo passado pelo **Handle** e retorna o número de bytes escritos em **BytesWrite**

  – **PARÂMETROS**

  * **Handle:** `THandle(??)` - Handle do arquivo
  * **Out Buffer** - Buffer de onde os dados devem ser escritos para o arquivo;
  * **Count:** `Int64(??)` - Número de bytes a ser escritos do Buffer para o arquivo;
  * **Out BytesRead**: `Int64(??)` - Número de Bytes efetivamente escritos.

  – **RETORNO**

  * `Longint(??)` - 0 (zero se sucesso ou o código do erro se fracasso;
  * Em **Buffer** os dados as ser escrito no arquivo;
  * Em **BytesRead** Retorna o número de bytes escritos efetivamente.
  * **EXEMPLO**

```
procedure TMi_Rtl_Tests.Action_Test_FileWriteExecute(Se

  // Este procedimento adiciona a sequência '-012345678

  Var
    Size : byte = 255;
    err  : TFiles.integer;
```

```
    h    : TFiles.THandle;

    NRec, Count : TFiles.Int64;

    s     : String[255];

    BytesLidos,BytesWrites: Int64;
Begin
  with TFiles do
  begin
    s :=  '-0123456789-0123456789'+LF;
    Size := length(s);

    if not FileExists('index.html')
    Then Err := FileCreate('index.html',fmOpenReadWrit
    else Err := FileOpen('index.html',h);

    if (err = 0)
    Then Begin
           err := fileSize('index.html',Count);
           if ( err = 0)
           Then begin
                  //Posiciona no final do arquivo - len
                  if Count >= size
                  then err := FileSeek(h,Count-length(I
                  else err := FileSeek(h,0,fsFromBeginni

                  if err = 0
                  Then Begin
                         //Acressenta string '-01234567
                         err:= FileWrite(h,s[1],length(
                         if (err = 0) and (BytesWrites
                         then Begin
                                ShowMessage('A sequênci
                              end
                         else begin
                                if (err <> 0)
                                Then ShowMessage(Error
                                else ShowMessage('Númer
                              end;
                       end
                  else Begin
                         ShowMessage(ErrorMessage(err))
                       end;
           end
           else ShowMessage(ErrorMessage(err));
```

```
        FileClose(h);
      end
    else ShowMessage(ErrorMessage(err));
  end;
end;
```

– **REFERÊNCIA**

FileWrites (https://www.freepascal.org/docs-html/rtl/sysutils/filewrite.html)

FileExists

Declaração `public class Function FileExists(Const FileName : AnsiString) : Boolean;`

Descrição

- A classe método `FileExists` checa se o arquivo passado no parâmetro existe.

DirectoryExists

Declaração `public class Function DirectoryExists(Const Directory : AnsiString) : Boolean;`

Descrição

- A classe método `DirectoryExists` checa se o diretório passado no parâmetro existe

CreateDir

Declaração `public class Function CreateDir(Const NewDir : AnsiString) : Boolean;`

Descrição

- A classe método `CreateDir` cria diretório passado no parâmetro

GetTempFileName

Declaração `public class function GetTempFileName(const Dir : string): string ;`

Descrição A classe método `GetTempFileName` retorna o nome de um arquivo temporário no diretório Dir.

- **NOTA**
  - Se Dir estiver vazio, o valor retornado por `GetTempDir`(**??**) será usado.
  - O Prefix será 'TMP'.
  - Em caso de erro, uma string vazia é retornada.

GetTempDir

Declaração `public class function GetTempDir(): string; overload;`

Descrição A classe function `GetTempDir` retorna o diretório temporário do sistema.

- **NOTA**
  - O nome retornado terminará com um caractere delimitador de diretório.
  - Não há garantia de que esse diretório exista ou seja gravável pelo usuário.

GetTempDir

Declaração `public class function GetTempDir(Const env:String;out path:PathStr):SmallInt; overload;`

Descrição A classe function `GetTempDir` retorna o diretório temporário do sistema.

- **PARÂMETROS**
  - **Const env** : Variável de ambiente que tenha que contenha a pasta de arquivos temporários.
  - **out path:`PathStr`(**??**) : Retorna a pasta dos arquivos temporários.
- **RETORNA**
  - SmallInt : Código do erro se houver ou zero (0) se conseguiu gerar o nome da pasta.

GetDriveType

Declaração `public class function GetDriveType(aPath : AnsiString): TDriveType; Overload;`

Descrição

- A função `GetDriveType` é usada para saber o tipo de dispositivo associado a pasta.
  - **PARÂMETRO**
    - **aPath** - A pasta dona do arquivo.
  - **RETORNA**
    - O tipo de dispositivo do tipo `TDriveType`(**??**).

DuplicateHandle

Declaração `public Class function DuplicateHandle(hSourceHandle: THandle;Var lpTargeTHandle: THandle) : Longint;`

Descrição

- A classe método `DuplicateHandle` duplica o handle do arquivo no windows e no linux essa função não funciona.

FileFlushBuffers

Declaração `public Class function FileFlushBuffers(aHandle: THandle): longint; overload;`

Descrição A classe function `FileFlushBuffers` descarrega o buffer do arquivo passado por aHandle

- **REFERÊNCIAS**

Linux (https://www.freepascal.org/docs-html/rtl/unix/fpfsync.html)

windows (https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-flushfilebuffers)

LockFile

Declaração `public class function LockFile(_Handle:THandle; _LockStart, _LockLength: Int64): LongInt;`

Descrição

- A função `LockFile` trava uma região do arquivo. -**NOTA**
  - O `lockfile` do linux não bloqueia região do arquivo e sim o arquivo todo.
  - **REFERÊNCIAS**

    Linux (https://www.freepascal.org/docs-html/rtl/unix/fpflock.html)

    windows (https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-lockfile)

UnLockFile

Declaração `public class function UnLockFile(_Handle:THandle; _LockStart, _LockLength: Int64): LongInt;`

Descrição

- A classe método `UnLockFile` destrava a região travada por `LockFile`(??).
  - **NOTA**
    - Funciona no linux mais não funciona do linux.

FindFiles

Declaração `public class procedure FindFiles(Mask : AnsiString; FileAttrs : Cardinal; var List : TStringList);`

Descrição

- A classe método `FindFiles` retorna uma lista de nomes de arquivos e diretórios que satisfazem os parâmetros: **Mask** e **FileAttrs**

– **EXEMPLO DE USO**

```
procedure TMi_Rtl_Tests.FormCreate(Sender: TObject);
begin
  ListFiles := TMiStringList.Create;
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.Action_test_FindFirstExecute(Sender: TObjec
  //Este procedimento ler os atributos da pasta: '.'

   function GetInfoFile(FileName:string;attribute : Cardinal; out i

  begin
     Result := FindFirst(ExpandFileName(FileName),attribute,Info);
     if Result = 0
     then Begin
            ShowMessage('O arquivo '+fileName+' contém o atributo:
          end
     else begin
            ShowMessage('O arquivo '+fileName+' não contém o atribu
          end;
  end;

  var
   Info: TSearchRec;
   err,i : integer;

   const FileAttrs : Cardinal = faHidden or
                                faReadOnly or
                                faSysFile or
                                faArchive or
                                faAnyFile or
                                faSymLink or
                                faDirectory ;


begin
  ListFiles.Clear;
  ListBox1.Clear;

  FileAttrs := 0;

  if CheckBox_faHidden.Checked
```

```
  then FileAttrs := faHidden;

  if CheckBox_faReadOnly.checked
  then FileAttrs := FileAttrs or faReadOnly;

  if CheckBox_faSysFile.checked
  then FileAttrs := FileAttrs or faSysFile;

  if CheckBox_faArchive.checked
  then FileAttrs := FileAttrs or faArchive;

  if CheckBox_faAnyFile.checked
  then FileAttrs := FileAttrs or faAnyFile;

  if CheckBox_faSymLink.checked
  then FileAttrs := FileAttrs or faSymLink;

  if CheckBox_faDirectory.checked
  then FileAttrs := FileAttrs or faDirectory;

  with TMI_ui_types do
    FindFiles(Edit1.Text,FileAttrs ,ListFiles );

  LabelCount.Caption := Format('ListFiles.Count %d',[ListFiles.Coun
  LabelCount.Show;
  if ListFiles.Count > 0
  then begin
          for i := 0 to ListFiles.Count-1 do
          begin
            ListBox1.Items.Add(ListFiles[i]);
          end;
       end;
end;


procedure TMi_Rtl_Tests.Edit1Change(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faAnyFileChange(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faArchiveChange(Sender: TObject);
```

```
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faDirectoryChange(Sender: TObject)
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faHiddenChange(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faReadOnlyChange(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faSymLinkChange(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;

procedure TMi_Rtl_Tests.CheckBox_faSysFileChange(Sender: TObject);
begin
  Action_test_FindFirstExecute(Self);
end;
```

GetCurrentDir

Declaração `public class function GetCurrentDir:  AnsiString;`

Descrição A classe método `GetCurrentDir` retorna o corrente pasta.

SetCurrentDir

Declaração `public class function SetCurrentDir( const NewDir : AnsiString):Boolean;`

Descrição A classe método `SetCurrentDir` define a pasta passado por **NewDir** como pasta corrente.
PARÂMETRO

- **NewDir** - Nome da pasta a ser definida.

RETORNA

- **TRUE** - Se sucesso
- **FALSE** - Se fracasso;

IsDirectory

Declaração `public class function IsDirectory( const Directory : AnsiString):Boolean;`

Descrição A classe método `IsDirectory` checa se a pasta passado por **Directory** é uma pasta válida.

PARÂMETRO

- **Directory** - Nome da pasta

RETORNA

- **TRUE** - Se a pasta existe
- **FALSE** - Se a pasta não existe;

FPrimeiroHandleLivre

Declaração `public class Function FPrimeiroHandleLivre: SmallInt;`

Descrição Retorna o numero de arquivos abertos no sistema operacional

- **NOTA**
  - TaStaus : Retorna o número do error se ouver

FlockFile

Declaração `public class function FlockFile(Handle : Thandle; modo : LongInt): LongInt ; overload;`

Descrição A class function `FlockFile` define ou remove um bloqueio no arquivo passado por Handle.

- **PARÂMETROS**
  - **MODE**
    - O modo pode ser uma das seguintes constantes:
      - LOCK_SH : define um bloqueio compartilhado.
      - LOCK_EX : define um bloqueio exclusivo.
      - LOCK_UN : desbloqueia o arquivo.
      - LOCK_NB : Isso pode ser OR junto com o outro. Se isso for feito, o aplicativo não bloqueia ao bloquear.
- **RETORNO**
  - `LONGINT(??)` : A função retorna zero se for bem-sucedida, um valor de retorno diferente de zero indica um erro.
- **REFERÊNCIA**
  - https://www.freepascal.org/docs-html/rtl/unix/fpflock.html

DiskFree

Declaração `public class function DiskFree(Partition:byte; out VrDiskFree :Int64):integer;`

Descrição A class function `DiskFree` retorna o espaço livre (EM BYTES) da partição passada por Partition

- PARÂMETRO:
  - 0 para a partição atual.
  - 1 para a primeira unidade de disquete.
  - 2 Para a segunda unidade de disquete.
  - 3 Para a primeira partição do disco rígido.
  - 4-26 Para todas as outras unidades e partições.

- REFERÊNCIA:
  - https://www.freepascal.org/docs-html/rtl/sysutils/diskfree.html
- OBSERVAÇÃO:
  - No Linux, e no Unix em geral, o conceito de disco é diferente do dos um, uma vez que o sistema de arquivos é visto como uma grande árvore de diretórios. Por esta razão, os `DiskFree` e DiskSize funções devem ser mimetizado utilizando nomes de arquivos que residem nas partições. Para obter mais informações, consulte AddDisk.
- EXEMPLO:

```
procedure testDiskFree;
   var
     VrDiskFree : int64;
begin
  WriteLn('TestDiskFree;');

  VrDiskFree := Diskfree(0);
  Writeln ('Free space of current disk: ',VrDiskFree, ' Bytes');

  VrDiskFree := (VrDiskFree div 1024);
  Writeln ('Free space of current disk: ',VrDiskFree,' KB)');

  VrDiskFree := (VrDiskFree div 1024);
  Writeln ('Free space of current disk: ',VrDiskFree,' MB)');

  VrDiskFree := (VrDiskFree div 1024);
  Writeln ('Free space of current disk: ',VrDiskFree,' GB)');

  VrDiskFree := (VrDiskFree div 1024);
  Writeln ('Free space of current disk: ',VrDiskFree,' TB)');
end;
```

ByteDrive

Declaração `public class Function ByteDrive(Const NomeArquivo:AnsiString) : Byte;`

Chapter 12

Unit mi.rtl.Objects.Consts

12.1 Descrição

- A Unit `mi.rtl.Objects.Consts` reune todos as contantes da unit **TObjects** globais usados pela class TObjects e suas descendências do pacote `mi.rtl`(??).
  - **NOTAS**
    - Esta unit foi testada nas plataformas: win32, win64 e linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **18/11/2021** 10:56 a ?? - Criar a unit mi.rtl.objects.Consts.pas
      - **19/11/2021** 20:35 a 21:22 - Conclusão da classe `TObjectsConsts`(??)
      - **13/12/2021** 21:00 a 22:10 - Documentar unidade.
  - **CÓDIGO FONTE**:
    -

12.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.objects.types`(??)
- `mi.rtl.consts.StringListBase`(??)

12.3 Visão Geral

`TObjectsConsts Classe`

12.4 Classes, Interfaces, Objetos e Registros

TObjectsConsts Classe

Hierarquia

TObjectsConsts > TObjectsTypes

Descrição

- A class `TObjectsConsts` usada para separar as contantes da unit **TObjects** do pacote `mi.rtl`(??).

Campos

OkZeraFGetMem

Declaração `public const OkZeraFGetMem :  Boolean = True;`

Descrição True zera a memória alocada por FGetMem

ErrorInfo

Declaração `public const ErrorInfo :  Integer = 0;`

Descrição Stream error info

stOk

Declaração `public const stOk = 0;`

Descrição

- No stream error

stError

Declaração `public const stError = -1;`

Descrição

- Access error

stCreateError

Declaração `public const stCreateError= -2;`

Descrição
- Initialize error

stReadError

Declaração `public const stReadError = -3;`

Descrição
- Stream read error

stWriteError

Declaração `public const stWriteError = -4;`

Descrição
- Stream write error

stGetError

Declaração `public const stGetError = -5;`

Descrição
- Get Class error

stPutError

Declaração `public const stPutError = -6;`

Descrição
- Put Class error

stSeekError

Declaração `public const stSeekError = -7;`

Descrição
- Seek error in stream

stOpenError

Declaração `public const stOpenError = -8;`

Descrição

- Error opening stream

StShareError

Declaração `public const StShareError = -9;`

Descrição

- Erro de compartilhamento

FmReadOnly

Declaração `public const FmReadOnly = fmOpenRead;`

Descrição 000

FmWriteOnly

Declaração `public const FmWriteOnly = fmOpenWrite;`

Descrição 001

FmReadWrite

Declaração `public const FmReadWrite = fmOpenReadWrite;`

Descrição 010

FmDenyALL

Declaração `public const FmDenyALL = fmShareCompat or fmShareExclusive;`

Descrição 0010000

FmDenyWrite

Declaração `public const FmDenyWrite = fmShareCompat or fmShareDenyWrite;`

Descrição 0100000

FmDenyRead

Declaração `public const FmDenyRead = fmShareCompat or fmShareDenyRead;`

Descrição 0110000

FmDenyNone

Declaração `public const FmDenyNone = fmShareCompat or fmShareDenyNone;`

Descrição 1000000

FmChildProcesses

Declaração `public const FmChildProcesses = $0080;`

Descrição 10000000

FmCreate

Declaração `public const FmCreate = $0100;`

Descrição 100000000

FmWait

Declaração `public const FmWait = $0200 ;`

Descrição 1000000000

FmMemory

Declaração `public const FmMemory = $0400 ;`

Descrição 10000000000 - Indica que o arquivo esta em tStreamemoria

FmMemory_Temp

Declaração `public const FmMemory_Temp = $0800 ;`

Descrição 100000000000 - Indica que o arquivo e temporario e esta em tStreamemoria

stOpenRead

Declaração `public const stOpenRead = FmReadOnly ;`

Descrição 000 - Read access only

stOpenWrite

Declaração `public const stOpenWrite = FmWriteOnly;`

Descrição 001 - Write access only

stOpen

Declaração `public const stOpen = fmOpenReadWrite;`

Descrição 010 - Read/write access

StCreate

Declaração `public const StCreate = FmCreate ;`

Descrição 100000000 - Create new file

IsApp_TV

Declaração `public const IsApp_TV : Boolean = False;`

Descrição

- A constante `IsApp_TV` indica se a aplicação gráfica é turbo vision.

IsApp_Console

Declaração `public const IsApp_Console :  Boolean = false;`

Descrição

- A constante `IsApp_Console` indica se a aplicação CGI deve ser compilado no modo console.

IsApp_Gui

Declaração `public const IsApp_Gui :  Boolean = false;`

Descrição

- A constante `IsApp_Gui` indica se a aplicação é gráfica independente de usar vcl ou não.

IsApp_ISAPI

Declaração `public const IsApp_ISAPI : Boolean = False;`

Descrição

- A constante `IsApp_ISAPI` indica se a aplicação web é compilada como dll deve ser executada em conjunto com browser.

IsApp_App_WS_Soap

Declaração `public const IsApp_App_WS_Soap :  Boolean = False;`

Descrição

- A constante `IsApp_App_WS_Soap` indica se a aplicação web é publicado com serviço XML com Protocolo Soap

IsApp_VCL

Declaração `public const IsApp_VCL : Boolean = false;`

Descrição

- A constante `IsApp_VCL` indica se a aplicação VCL pode ser mista console e gráfica.

IsApp_VCL_IE

Declaração `public const IsApp_VCL_IE : Boolean = False;`

Descrição

- A constante `IsApp_VCL_IE` indica se a aplicação gráfica usa os componentes VCL e webBrowser como entrada de dados.

IsApp_Cgi

Declaração `public const IsApp_Cgi :  Boolean = false;`

Descrição

- A constante `IsApp_Cgi` indica se a Aplicação é CGI.
  - **NOTA** -Ignora todo acesso do teclado e video do console usada como aplicações web, console ou GUI quando usado como pacote em tempo de designer.

IsApp_DSServerModule

Declaração `public const IsApp_DSServerModule :  Boolean = false;`

Descrição

- A constante `IsApp_DSServerModule` indica se a Aplicação App_DSServerModule.
  - **NOTA**
    - Ignora todo acesso ao teclado e video do console usada como aplicações servidor de dados.

coIndexError

Declaração `public const coIndexError = -1;`

Descrição Index out of range

coOverflow

Declaração `public const coOverflow = -2;`

Descrição Overflow

vmtHeaderSize

Declaração `public const vmtHeaderSize = 8;`

Descrição VMT header size

MaxBytes

Declaração `public const MaxBytes = MAX_BYTE;`

Descrição Maximum data size

MaxWords

Declaração `public const MaxWords = MAX_WORD;`

Descrição Max word data size

MaxSmallWords

Declaração `public const MaxSmallWords = MAX_SMALL_WORD;`

Descrição Max word data size

MaxPtrs

Declaração `public const MaxPtrs = MAX_ARRAY_PTR;`

Descrição Max ptr data size

MaxCollectionSize

Declaração `public const MaxCollectionSize = MaxPtrs;`

Descrição Max collection size

StreamTypes

Declaração `public const StreamTypes: PStreamRec = Nil;`

Descrição Stream types reg

AnsiChar_Control_Template

Declaração
```
public const AnsiChar_Control_Template : AnsiCharSet =
  [#0..#31,''];
```

Descrição `AnsiChar_Control_Template` : AnsiCharSet = [#0..#31,'',^a..^z,^A..^Z];

Okprocessing

Declaração `public const Okprocessing : Boolean = false;`

OkprocessingControlTime

Declaração `public const OkprocessingControlTime : Boolean = true;`

OkOkprocessingTime

Declaração `public const OkOkprocessingTime : Longint = 5 * 60;`

OkProcessingTime_Action

Declaração `public const OkProcessingTime_Action : TOkProcessingTime_Action = OkProcessingTime_Action_Password;`

OkOkprocessingClockBegin

Declaração `public const OkOkprocessingClockBegin : DWord = 0;`

OkTempoDeTentativas

Declaração `public const OkTempoDeTentativas : Boolean = true;`

Descrição Habilita `TempoDeTentativas`(**??**) nas leitura e escritas ao arquivo

TempoDeTentativas

Declaração `public const TempoDeTentativas : Longint = 10;`

Descrição TimeOut = Tempo em segundos de tentativos nos processos de abertura, leitura e gravacao de arquivos

Métodos

SetOkprocessing

Declaração `public Class Function SetOkprocessing(aOkprocessing : Boolean) : Boolean;`

FreeAndNil

Declaração `public class procedure FreeAndNil(var Obj);`

NewStr

Declaração `public class function NewStr(Const S: AnsiString): ptstring;`

Descrição -

NewSItem

Declaração `public class function NewSItem(const Str: tString; ANext: PSItem): PSItem;`

CloneSItems

Declaração `public class Function CloneSItems(Const Items: PSItem):PSItem;`

CopyTemplateFrom

Declaração `public class Function CopyTemplateFrom(Const aTemplate:tString): tString;`

Descrição A class function `CopyTemplateFrom` é necessario porque um Template pode ser uma lista de Strings, onde está lista pode ser inserida em um objeto e discartada ao destruir o objeto.

PushSItem

Declaração `public class Procedure PushSItem( Str: AnsiString; Var ANext: PSItem);`

Push_MsgErro

Declaração `public class Procedure Push_MsgErro( Str: AnsiString);`

Descrição Coloca uma string na pilha

Chapter 13

Unit mi.rtl.Objects.Consts.Logs

13.1 Descrição

- A Unit `mi.rtl.Objects.Consts.Logs` implementa a classe `TFilesLogs(??)` baseado na classe **TEventLog** da **FCL**
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **REFERÊNCIA**

TEventLog (https://www.freepascal.org/docs-html/current/fcl/eventlog/teventlog.html)

  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **08/12/2021**
      - 13:55 a 18:30 : Análise do projeto `mi.rtl.Objects.Consts.Logs`;
      - 20:30 a 22:40 : Criar a unit `mi.rtl.Objects.Consts.Logs` e a classe `TFilesLogs(??)`;
      - **09/12/2021**
      - 06:20 a 07:06 : Documentar a unit `mi.rtl.Objects.Consts.Logs`;
      - 09:02 a 12:15 : Documentar a classe `TFilesLogs(??)`.
      - 14:45 a 15:15 : Criar os métodos `TFilesLogs.Warning(??)` e documentar.
      - **13/12/2021**
      - 20:40 a ?? -Criar os métodos `TFilesLogs.info(??)` e documentar.

13.2 Uses

- Classes
- SysUtils
- dos
- mi.rtl.objects.consts(??)
- eventlog

13.3 Visão Geral

TFilesLogs Classe

13.4 Classes, Interfaces, Objetos e Registros

TFilesLogs Classe

Hierarquia

TFilesLogs > TObjectsConsts(??) > TObjectsTypes

Descrição

- A classe TFilesLogs é usada para registrar no arquivo **ParamStr(0)**+'.log' as mensagens de **Erros**, **Atenções** e **avisos** do sistema.
  - **NOTA**
    - A classe TObjectss(??) implementa a constante global **const TObjectss.Logs :** TFilesLogs **= nil;** que é inicializado em **mi.rtl.objectss.Initialization** e destruído em **mi.rtl.objectss.finalization**

Propriedades

fileLog

Declaração published property fileLog : TEventLog Read _fileLog;

FileName

Declaração public property FileName : string read GetFileName write SetFileName;

Descrição A propriedade FileName armazena o nome do arquivo de log necessário quando a propriedade LogType(??)=**ltFile**

Active

Declaração `public property Active : Boolean Read GetActive write SetActive;`

Descrição A propriedade `Active` ativa e desativa a gravação de mensagens no arquivo de log.

LogType

Declaração `public property LogType : TLogType Read GetLogtype Write SetlogType;`

Descrição

- A propriedade `LogType` redireciona o destino das mensagens de log.
  - Veja o tipo `TLogType(??)` para mais informações.

AppendContent

Declaração `public property AppendContent : Boolean Read GetAppendContent Write SetAppendContent;`

Descrição

- A propriedade `AppendContent` é usada para indicar ao sistema que o arquivo de log deve ser único ou acumulativo
  - **PARÂMETROS**
    - **True** : A abertura do arquivo de log é aberto com o comando **append**
    - **False** : A abertura do arquivo de log é aberto com o comando **rewrite**
    - **Código usado na abertura do arquivo**

```
if AppendContent and FileExists(FileName) then
   FileFlags := fmOpenWrite
else
   FileFlags := fmCreate;
```

RaiseExceptionOnError

Declaração `public property RaiseExceptionOnError : Boolean Read GetRaiseExceptionOnError Write SetRaiseExceptionOnError;`

Descrição

- A propriedade `RaiseExceptionOnError` informa ao sistema se deve gerar exceções quando houver erro ao gravar o log.

  - **Exemplo de uso de** `RaiseExceptionOnError`

```
procedure TEventLog.WriteFileLog(EventType : TEventType; const Msg
  Var
    S : String;
begin
  S:=FormatLogMessage(EventType, Msg)+LineEnding;
  try
    FStream.WriteBuffer(S[1],Length(S));
    S:='';
  except
    On E : Exception do
       S:=E.Message;
    end;

  If (S<>'') and RaiseExceptionOnError
  then  Raise ELogError.CreateFmt(SErrLogFailedMsg,[S]);

end;
```

Campos

EnableWriteIdentificao

Declaração `public const EnableWriteIdentificao : Boolean = true;`

Métodos

Create

Declaração `public constructor Create(aowner:TComponent); Override; Overload;`

Descrição O constructor `Create` cria uma instância da classe `TFilesLogs`(**??**) e inicializa as propriedade `FileName`(**??**) := **ParamStr(0)+'.log'**, `LogType`(**??**) := **ltFile**, *RaiseExceptionOnError*(**??**) := *true*, `AppendContent`(**??**) := **true** e `active`(**??**) := **true**.

destroy

Declaração `public destructor destroy; override;`

Warning

Declaração `public procedure Warning( const Fmt: string;Args: array of Const); overload;`

Descrição

- A procedure `Warning` é usada registrar mensagens do tipo **etWarning**.
  - **PARÂMETROS**
    - **Fmt** : String que será formatada com a função [**Format**](https://www.freepascal.org/docs-html/rtl/sysutils/forma
    - **Args** : Valores a serem usadas no função format(FMT,Args)
  - **REFERÊNCIA**

TEventLog.warning (https://www.freepascal.org/docs-html/fcl/eventlog/teventlog.warn

  - **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Action_test_Logs_WarningExecute(Sender: TOb
begin
  with Tobjectss do
    if Logs.Active then
    with Logs do
    begin
      Warning('Pouco espaço em disco. Tamanho livre = %s.', ['1GB']
    end;
end;
```

    - Arquivo de logs: **rtl.log**:

2021-12-09 14:58:00.588 `Warning`(**??**) Pouco espaço em disco. Tamanho livre = 1GB.

Warning

Declaração `public procedure Warning( const Msg:  string ); overload;`

Descrição

- A procedure `Warning` é usada registrar mensagens do tipo **etWarning**.
  - **PARÂMETROS**
    - **Msg** : String com a mensagem de atenção.
  - **REFERÊNCIA**

TEventLog.warning (https://www.freepascal.org/docs-html/fcl/eventlog/teventlog.warn

  - **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Button2Click(Sender: TObject);
begin
  with Tobjectss do
    if Logs.Active then
    with Logs do
    begin
      Warning('Senha inválida!');
    end;
end;
```

    - Arquivo de logs: **rtl.log**:

2021-12-09 14:58:00.588 `Warning` Senha inválida

Error

Declaração `public procedure Error(const Fmt:  String; Args:  array of const); overload;`

Descrição

- A procedure `Error` é usada registrar mensagens do tipo **etError**.
  - **PARÂMETROS**

* **Fmt** : String que será formatada com a função [**Format**](https://www.freepascal.org/docs-html/rtl/sysutils/forma
* **Args** : Valores a serem usadas no função format(FMT,Args)

– **REFERÊNCIA**

TEventLog.error (https://www.freepascal.org/docs-html/fcl/eventlog/teventlog.erro

– **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Action_test_Logs_ErrorExecute(Sender: TObjec
begin
  with Tobjectss do
    if Logs.Active then
    with Logs do
    begin
      Error('Exemplo de mensagem de erro. Número do erro = %d.', [5
    end;
end;
```

* Arquivo de logs: **rtl.log**:

2021-12-09 10:25:41.425 `Error(??)` Exemplo de mensagem de erro. Número do erro = 5.

Error

Declaração `public procedure Error(const Msg:  String); overload;`

Descrição

- A procedure `Error` é usada registrar mensagens do tipo **etError**.

– **PARÂMETROS**

* **Msg** : String com o nome do erro.

– **REFERÊNCIA**

TEventLog.error (https://www.freepascal.org/docs-html/fcl/eventlog/teventlog.erro

– **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Button2Click(Sender: TObject);
begin
  with TObjectss do
    if Logs.Active then
    with Logs do
    begin
      Error('Acesso ao arquivo negado.' );
    end;
end;
```

* Arquivo de logs: **rtl.log**:

2021-12-09 10:25:41.425 `Error` Acesso ao arquivo negado.

Info

Declaração `public procedure Info(const Fmt: String; Args: array of const); overload;`

Descrição

- A procedure `Info` é usada registrar mensagens do tipo **etInfo**.

– **PARÂMETROS**

* **Fmt** : String que será formatada com a função [**Format**](https://www.freepascal.org/docs-html/rtl/sysutils/forma
* **Args** : Valores a serem usadas no função format(FMT,Args)

– **REFERÊNCIA**

TEventLog.Info (https://www.freepascal.org/docs-html/current/fcl/eventlog/tevent

– **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Action_test_Logs_InfoExecute(Sender: TObject
begin
  with Tobjectss do
    if Logs.Active then
    with Logs do
    begin
      Info('A versão %s será lançada em breve.', ['Beta 3.5']);
    end;
end;
```

* Arquivo de logs: **rtl.log**:

2021-12-13 20:46:32.745 `Info(??)` A versão Beta 3.5 será lançada em breve.

Info

Declaração `public procedure Info(const Msg: String); overload;`

Descrição

- A procedure `Info` é usada registrar mensagens do tipo **etInfo**.
  - **PARÂMETROS**
    * **Msg** : String com mensagem informativa.
  - **REFERÊNCIA**

    TEventLog.Info (https://www.freepascal.org/docs-html/current/fcl/eventlog/tevent
  - **EXEMPLO**:

```
procedure TMi_Rtl_Tests.Action_Test_Logs_InfoExecute(Sender: TObject
begin
  with Tobjectss do
    if Logs.Active then
    with Logs do
    begin
      info('Sistema abortou de forma inesperada.' );
    end;
end;
```

* Arquivo de logs: **rtl.log**:

2021-12-13 20:46:32.745 `Info` Sistema abortou de forma inesperada.

_Write

Declaração `public Function _Write(Const S : AnsiString;Ln:Boolean):SmallInt;`

Flush_WiteBuffer

Declaração `public Function Flush_WiteBuffer:Boolean;`

WriteFErr_Ln_On_Off

Declaração `public Function WriteFErr_Ln_On_Off(Const S : AnsiString;aLn_On_Off:Boolean):SmallInt;`

WriteFErr

Declaração `public Function WriteFErr(Const S : AnsiString):SmallInt;`

WriteLnFErr

Declaração `public Function WriteLnFErr(Const S : AnsiString):SmallInt;`

WriteIdentificao

Declaração `public Procedure WriteIdentificao;`

LogError

Declaração `public PROCEDURE LogError(const Fmt: String; Args: array of const); overload;`

LogError

Declaração `public PROCEDURE LogError(CONST Msg:AnsiString ); overload;`

Chapter 14

Unit mi.rtl.Objects.Consts.ProgressDlg_If

14.1 Descrição

- A unit `mi.rtl.Objects.Consts.ProgressDlg_If` implementa a classe `TProgressDlg_If`(??) do pacote `mi.rtl`(??).
  - **VERSÃO**:
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - 2021-12-18
      - 14:42 15:30 - T12 Criar a unit `mi.rtl.Objects.Consts.ProgressDlg_If` e a classe `Tprogressdlg_if`(??)
      - 2021-12-21
      - 8:27 a xx - T21 documentar a classe `TProgressDlg_If`(??)

14.2 Uses

- Classes
- drivers
- DOS
- LazarusPackageIntf
- mi.rtl.objects.consts(??)

14.3 Visão Geral

TProgressDlg_If Classe

Register

14.4 Classes, Interfaces, Objetos e Registros

TProgressDlg_If Classe

Hierarquia

TProgressDlg_If > TObjectsConsts(??) > TObjectsTypes

Descrição

- A classe TProgressDlg_If é uma classe abstrata para comunicação com o usuário cujo a implementação deve ser feita nas plataformas: LCL, HTML e JavaScript.
  - **NOTA**
    - Só cria o dialogo se a posição chegar no delta(??).
  - Exemplo do uso de TProgressDlg_If

```
Var
  ProgressDlg_If : TProgressDlg_If;
Begin
  ProgressDlg_If := TProgressDlg_If.Create('Pesquisando registro',Alias,20);

  Repeat
    ProgressDlg_If.IncPosition;
  Until Not next;

  ProgressDlg_If.Free;
end.
```

Propriedades

Total

Declaração `published property Total: Longint Read _Total Write Set_Total;`

Descrição A propriedade `Total` é o `total` de elementos previstos na lista ao inicial o dialogo

Delta

Declaração `published property Delta: Longint Read _Delta Write Set_Delta;`

Descrição A propriedade `Delta` informado ao dialogo o intervalo no qual o dialogo precisa ser criado para que o usuário veja a previsão de termino.

Limit

Declaração `published property Limit: Longint Read _Limit Write Set_Limit;`

Descrição A propriedade `Limit` é o numero de linhas do controle que está sendo visualizado.

Title

Declaração `published property Title: AnsiString Read _Title Write Set_Title;`

Descrição A propriedade `Title` é usado no título do dialogo indicando a tarefa que está sendo executada.

observation

Declaração `published property observation: AnsiString Read _observation Write Set_observation;`

Descrição A propriedade `observation` é usado na barra de status do dialogo indicando qual o atalho aborta a operação

onCreate_ProgressDlg

Declaração `published property onCreate_ProgressDlg: TCreate_ProgressDlg Read _onCreate_ProgressDlg Write _onCreate_ProgressDlg;`

Descrição A propriedade `onCreate_ProgressDlg` deve ser implementado no pacote visual ou seja: Na interface do usuário que pode ser **LCL**, Javascript, tv32 etc..

onIncPosition_01

Declaração `published property onIncPosition_01: TIncPosition_01 Read _onIncPosition_01 Write _onIncPosition_01;`

Descrição A propriedade `onIncPosition_01` deve ser implementada na classe visual para incrementar aDelta na posição atual do processamento.

onIncPosition

Declaração `published property onIncPosition: TIncPosition Read _onIncPosition Write _onIncPosition;`

Descrição A propriedade `onIncPosition` deve ser implementada na classe visual para incrementar 1 na posição atual do processamento.

OnRedraw

Declaração `published property OnRedraw: TRedraw Read _OnRedraw Write _OnRedraw;`

Descrição A propriedade `OnRedraw` deve ser implementada na classe visual para atualizar a tela com a posição atual do processamento.

Campos

_TimeCurrent

Declaração `protected Var _TimeCurrent: Longint;`

_TimeBegin

Declaração `protected Var _TimeBegin:  Longint;`

_TimeForeseen

Declaração `protected Var _TimeForeseen:  Longint;`

_Percent

Declaração `protected Var _Percent:  SmallInt;`

Métodos

Set_Total

Declaração `protected procedure Set_Total(aTotal:  Longint); Virtual;`

Set_Delta

Declaração `protected procedure Set_Delta(aDelta:  Longint); Virtual;`

Set_Limit

Declaração `protected procedure Set_Limit(aLimit:  Longint); Virtual;`

Set_Title

Declaração `protected procedure Set_Title(aTitle:  AnsiString); Virtual;`

Set_observation

Declaração `protected procedure Set_observation(aobservation:  AnsiString); Virtual;`

Create_ProgressDlg

Declaração `public Procedure Create_ProgressDlg; overload; Virtual;`

Descrição A procedure `Create_ProgressDlg` deve ser anulada para implementar a criação do diálogo no pacote visual ou seja: Na interface do usuário que pode ser **LCL**, Javascript, tv32 etc..

RegisterOnEvents

Declaração `protected Procedure RegisterOnEvents; Virtual;`

Descrição A procedure `RegisterOnEvents` deve ser anulada para implementar os eventos desta classe caso a mesma não esteja registrada na IDE

Create

Declaração `public constructor Create(AOwner: TComponent); Overload; override;`

Descrição O constructor `Create` é necessário porque essa classe pode ser registrada da IDE

Create

Declaração `public constructor Create(aTitle : AnsiString; aobservation : AnsiString; aDelta, aTotal : longint ); Overload; Virtual;`

Descrição O constructor `Create` é usado para criar a classe sem a IDE

Destroy

Declaração `public Destructor Destroy; Override;`

Descrição O destructor `Destroy` é usado para destruir a classe.

IncPosition

Declaração `public procedure IncPosition(Const aDelta :longint); Overload; Virtual;`

Descrição A propriedade `IncPosition` deve ser anulada na classe visual para incrementar **aDelta** na posição atual do processamento.

IncPosition

Declaração `public procedure IncPosition; Overload; Virtual;`

Descrição A propriedade `IncPosition` deve ser anulada na classe visual para incrementar 1 na posição atual do processamento.

Redraw

Declaração `protected Procedure Redraw; Virtual;`

Descrição A propriedade `Redraw` deve ser anulada para implementar na classe visual para atualizar a tela com a posição atual do processamento.

SetPerc

Declaração `public procedure SetPerc(const aPosition :  longint);`

Descrição A procedure `SetPerc` é usado para informar ao dialogo a posição atual da contagem.

- **NOTA**
  - Calcula o percentual atual do processamento.

14.5 Funções e Procedimentos

Register

Declaração `procedure Register;`

14.6 Tipos

TProgressDlg_If_Class

Declaração `TProgressDlg_If_Class = Class of TProgressDlg_If;`

Chapter 15

Unit mi.rtl.objects.consts.strings

15.1 Uses

- Classes
- SysUtils

Chapter 16

Unit mi.rtl.Objects.Methods

16.1 Descrição

- A Unit `mi.rtl.Objects.Methods` implementa a classe `TObjectsMethods(??)` do pacote `mi.rtl(??)`.
  - **NOTAS**
    - Esta unit foi testada nas plataformas: win32, win64 e linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **19/11/2021** 21:25 a 23:15 Criar a unit mi.rtl.objects.TObjectsMethods.pas
      - **20/11/2021** 18:00 a 19:15 Criar a unit mi.rtl.objects.tStream.pas
  - **CÓDIGO FONTE**:
    -

16.2 Uses

- `Classes`
- `SysUtils`
- `System.UITypes`

- Process
- dos
- sqlDB
- SQLite3Conn
- PQConnection
- FpHttpClient
- mi.rtl.Class_Of_Char(??)
- mi.rtl.types(??)
- mi.rtl.Consts(??)
- mi.rtl.objects.consts(??)
- mi.rtl.objects.consts.MI_MsgBox
- mi.rtl.objects.consts.progressdlg_if(??)
- mi.rtl.objects.Consts.logs(??)
- mi.rtl.consts.StringList(??)

16.3 Visão Geral

TObjectsMethods Classe

16.4 Classes, Interfaces, Objetos e Registros

TObjectsMethods Classe

Hierarquia

TObjectsMethods > TObjectsConsts(??) > TObjectsTypes

Descrição

- A classe TObjectsMethods implementa os método de classe comum a todas as classes de TObjects do pacote mi.rtl(??).

Propriedades

Alias

Declaração public property Alias: AnsiString Read _Alias Write _Alias;

Campos

MI_MsgBox

Declaração `public const MI_MsgBox:  TMI_MsgBox = nil;`

_Logs

Declaração `public const _Logs :  TFilesLogs = nil;`

Descrição

- `Logs`(**??**) é inicializado em Initialization e destruído em finalization

Métodos

Logs

Declaração `public class function Logs:  TFilesLogs;`

SysMessageBox

Declaração `public class procedure SysMessageBox(Msg, Title:  AnsiString; Error:  Boolean);`

Descrição

- O método de classe `SysMessageBox` executa o diálogo `MessageBox`(**??**) do Lazarus;
  - **REFERÊNCIA**
    - https://wiki.lazarus.freepascal.org/Dialog_Examples#MessageBox

MessageBox

Declaração `public class function MessageBox(const Msg:  AnsiString):  Word; Virtual; overload;`

Descrição A classe `MessageBox` deve ser implementada no pacote mi.ui.

MessageBox

Declaração `public class function MessageBox(const aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TModalResult; Virtual; overload;`

Descrição O método `MessageBox` recebe 3 parâmetros. Criar um dialogo e retrona as opções escolhidas.

- **Exemplo de uso**

```
If MessageBox('O arquivo '+TMI_DataFile(DatF).nomeArq+' não existe.'+^M+
              ^M+
              'Cria o arquivo agora?'
               ,MtConfirmation,mbYesNoCancel,mbYes)= MrYes
Then begin
     end;
```

Abstracts

Declaração `public class procedure Abstracts;`

Descrição

- A classe método `Abstracts` encerra o programa com um erro de tempo de execução 211.
  - **NOTA**
    - Ao implementar um tipo de classe abstrato, chame Abstract nesses métodos Override que deve ser substituído em tipos descendentes. Isso garante que qualquer tentativa de usar instâncias do tipo abstrato de classe falhará.

RegisterError

Declaração `public Class procedure RegisterError;`

RegisterType

Declaração `public class procedure RegisterType(Var S: TStreamRec);`

Descrição

- A classe método `RegisterType` registra o tipo de classe fornecido com os fluxos do Free Vision, criando uma lista de objetos conhecidos. Streams só podem armazenar e retornar esses Tipos de classe.
  - Cada classe registrada precisa de um registro de stream único registro, do tipo TStreamRec.

LongMul

Declaração `public class function LongMul(X, Y: Integer): LongInt;`

Descrição

- A class function `LongMul` retorna o valor inteiro longo de valores inteiros X * Y.

LongDiv

Declaração `public class function LongDiv(X: LongInt; Y: Integer): Integer;`

Descrição A classe function `LongDiv` retorna o valor inteiro do inteiro longo X dividido pelo inteiro Y.

NNewStr

Declaração `public class procedure NNewStr(Var PS : ptstring;Const S : AnsiString);`

Descrição -

CallPointerLocal

Declaração `public class function CallPointerLocal(Func: codepointer; Frame: Pointer; Param1: pointer): pointer; inline;`

DisposeStr

Declaração `public class PROCEDURE DisposeStr( Var P: ptstring);`

IsValidPtr

Declaração `public class FUNCTION IsValidPtr( ADDR:POINTER):BOOLEAN ; overload;`

IsValidPtr

Declaração `public class FUNCTION IsValidPtr( const aClass:  Tobject):BOOLEAN ; overload;`

Name_Type_App_MarIcaraiV1

Declaração `public class Function Name_Type_App_MarIcaraiV1:AnsiString;`

Set_IsApp_VCL

Declaração `public class Function Set_IsApp_VCL(aIsApp_VCL:Boolean):Boolean;`

PopSItem

Declaração `public class Procedure PopSItem(Var Items:  PSItem);`

DISCARD

Declaração `public class PROCEDURE DISCARD(Var AClass); overload;`

DISCARD

Declaração `public class PROCEDURE DISCARD(Var AClass:TObject); overload;`

SetFlushBuffer_Disk

Declaração `public class Function SetFlushBuffer_Disk(Const aFlushBuffer_Disk : Boolean): Boolean;`

SetFlushBuffer

Declaração `public class Function SetFlushBuffer(Const aFlushBuffer : Boolean): Boolean;`

GetDosTicks

Declaração `public class Function GetDosTicks:DWord;`

Descrição

- returns ticks at 18.2 Hz, just like DOS

Seg_to_MillSeg

Declaração `public class Function Seg_to_MillSeg(aSegundos:Longint):DWord;`

Descrição A função `Seg_to_MillSeg` converte segundos para milisegundos.

- **NOTA**
  - 1 Milliseconds = 1/1000 segundos -> 1 segundo = 1000 Milliseconds

RunError

Declaração `public class Procedure RunError(Error:Word);`

Run_Error

Declaração `public class Procedure Run_Error(Error:Word;Procedimento_que_Executou:AnsiString);`

Alert

Declaração `public Class Procedure Alert(aTitle: AnsiString;aMsg:AnsiString);`

Descrição

- A procedure `Alert` executa um dialogo com botão **OK**

ShowMessage

Declaração `public Class procedure ShowMessage(const aMsg: string);`

Confirm

Declaração `public Class Function Confirm(aTitle: AnsiString;aPergunta:AnsiString):Boolean;`

Descrição

- A função `Confirm` executa um dialogo com os botões **OK** e **Cancel** fazendo uma pergunta.
  - **RETORNA:**
    - **True** : Se o botão **OK** foi pŕessionando;
    - **False** : Se o botão **Cancel** foi pŕessionando.

InputValue

Declaração `public Class function InputValue(const aTitle,aLabel: AnsiString;var aValue : Variant): TModalResult;`

Descrição O método `InputValue` ler um valor na tela e retorna em **aValue** o valor e em result retorna **MrOk** ou **MrCancel**

Prompt

Declaração `public Class Function Prompt(aTitle: AnsiString;aPergunta:AnsiString;Var aResult: Variant):Boolean;`

Descrição

- A função `Prompt` mostra um dialogo com dois botões **OK** e **Cancel** e um campo input solicitando que o usuário digite um valor.
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pŕessionando;
    - **False** : Se o botão **cancel** foi pŕessionando.
    - **aResult** : Retorna a string digitada no formulário;

InputPassword

Declaração `public Class Function InputPassword(aTitle: AnsiString;out aUsername:AnsiString;out apassword:AnsiString):Boolean; Overload;`

Descrição

- A função `InputPassword` mostra um dialogo solicitando o login do usuário e a senha e dois botões **OK** e **Cancel**
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pŕessionando;
    - **False** : Se o botão **cancel** foi pŕessionando.
    - **aUsername** : Retorna a string com nome do usuário.
    - **apassword** : Retorna a string com a senha do usuário.

InputPassword

Declaração `public Class Function InputPassword(aTitle: AnsiString;out apassword:AnsiString):Boolean; Overload;`

DisposeSItems

Declaração `public class procedure DisposeSItems(VAR AItems: PSItem); overload;`

DisposeSItems

Declaração `public class procedure DisposeSItems(Var AStrItems: PtString); overload;`

MaxItemStrLen

Declaração `public class function MaxItemStrLen(AItems: PSItem) : integer; Overload;`

MaxItemStrLen

Declaração `public class function MaxItemStrLen(PSItems: tString) : integer; Overload;`

conststr

Declaração `public class function conststr( i : Longint; Const a : AnsiChar) : AnsiString;`

centralizesStr

Declaração `public class function centralizesStr(Const campo : AnsiString; Const tamanho : Integer) : AnsiString;`

TypeFld

Declaração `public class Function TypeFld(Const aTemplate : tString;Var aSize : SmallWord):AnsiChar; overload;`

TypeFld

Declaração `public class Function TypeFld(Const aTemplate : ShortString):AnsiChar; overload;`

IStr

Declaração `public class Function IStr( Const I : Longint; Const Formato : tString) : tString; Overload;`

IStr

Declaração `public class Function IStr( Const I : Longint) : tString; Overload;`

IStr

Declaração `public class Function IStr( Const I : tString; Const Formato : tString) : tString; Overload;`

StrNum

Declaração `public class function StrNum(formato : AnsiString; buffer :Variant; Const Tipo : AnsiChar;Const OkSpc:Boolean) : AnsiString; overload;`

StrNum

Declaração `public class function StrNum(formato : AnsiString; buffer:Variant; Const Tipo : AnsiChar) : AnsiString; overload;`

IIF

Declaração `public class function IIF(Const Logica : Boolean; Const E1 , E2 : Boolean) : Boolean; overload;`

IIF

Declaração `public class function IIF(Const Logica : Boolean; Const E1 , E2 : AnsiChar) : AnsiChar; overload;`

IIF

Declaração `public class function IIF(Const Logica : Boolean; Const E1 , E2 : Longint ) : Longint; overload;`

IIF

Declaração `public class function IIF(Const Logica : Boolean; Const E1 , E2 : Extended ) : Extended; overload;`

IIF

Declaração `public class function IIF(Const Logica : Boolean; Const E1 , E2 : AnsiString ) : AnsiString ; overload;`

SIF

Declaração `public class Function SIF(Const Logica : Boolean; Const E1 , E2 : AnsiString ) : AnsiString ;`

MinL

Declaração `public class function MinL(Const a,b:Longint):Longint;`

MaxL

Declaração `public class function MaxL(Const a,b:Longint):Longint;`

NumToStr

Declaração `public class function NumToStr(Const formato : AnsiString;buffer:Variant; Const Tipo : AnsiChar;Const OkSpc:Boolean):AnsiString;`

InsertCrtlJ

Declaração `public class Function InsertCrtlJ(Const StrMsg:tString):tString;`

Create_Progress1Passo

Declaração `public class procedure Create_Progress1Passo(ATitle : tstring;Obs:tstring ; ATotal : Longint); Virtual;`

Set_Progress1Passo

Declaração `public class procedure Set_Progress1Passo(aNumber : Longint); Virtual;`

Destroy_Progress1Passo

Declaração `public class procedure Destroy_Progress1Passo; Virtual;`

LogError

Declaração `public class procedure LogError(const Fmt: String; Args: array of const); overload;`

LogError

Declaração `public class procedure LogError(const Msg:AnsiString ); overload;`

WideStringToString

Declaração `public class function WideStringToString(const ws: WideString): AnsiString;`

Set_FileModeDenyALLSalvaAnt

Declaração `public class Function Set_FileModeDenyALLSalvaAnt(Const ModoDoArquivo : Boolean;Var _FileModeDenyALLAnt:Boolean):Boolean;`

Set_FileModeDenyALL

Declaração `public Class Function Set_FileModeDenyALL(Const ModoDoArquivo : Boolean):Boolean;`

Sitems_MsgErro

Declaração `public class Function Sitems_MsgErro() : PSItem;`

Pop_MsgErro

Declaração `public class Function Pop_MsgErro:PSItem;`

Descrição Retire o ultimo string na pilha

SpcStrD

Declaração `public class function SpcStrD(Const campo : tString; Const Tam : Byte): tString;`

CentralizaStr

Declaração `public class function CentralizaStr(Const campo : AnsiString; Const tamanho : Integer) : AnsiString;`

spc

Declaração `public class function spc(Const campo:AnsiString;Const tam :Longint):AnsiString;`

NumberCharControl

Declaração `public class function NumberCharControl(s:AnsiString):Integer;`

Descrição O método `NumberCharControl` retorna o número de caracteres de controle da string s

StrAlinhado

Declaração `public class Function StrAlinhado(aStrMsg:tString;Colunas : byte;Const Alinhamento:TAlinhamento):tString;`

StringToSItem

Declaração `public class Function StringToSItem( StrMsg:AnsiString; Colunas : byte;Alinhamento:TAlinhamento):PSItem; virtual; overload;`

StringToSItem

Declaração `public class Function StringToSItem( StrMsg:AnsiString; Colunas : byte):PSItem; virtual; overload;`

SItemsLen

Declaração `public class function SItemsLen(S: PSItem) :  SmallInt;`

SItemToString

Declaração `public class Function SItemToString(Items:  PSItem):AnsiString;`

WriteSItems

Declaração `public class procedure WriteSItems(var S: TMiStringList;const Items: PSItem);`

Descrição A classe procedure `WriteSItems` retorna um TStringList com a lista passado por items

- **NOTA**
  - S : Deve ser passado não inicializado, ouseja deve ser NIL.

PSItem_ListaDeMsgErro

Declaração `public class Function PSItem_ListaDeMsgErro:PSItem; virtual;`

MessageError

Declaração `public class Procedure MessageError; virtual;`

String_ListaDeMsgErro

Declaração `public class Function String_ListaDeMsgErro(Separador:String):AnsiString; Overload;`

Dispose_ListaDeMsgErro

Declaração `public class Procedure Dispose_ListaDeMsgErro; virtual;`

Descrição A procedure `Dispose_ListaDeMsgErro` esvazia a pilha de mensagens de error caso as mensagen não tenhão sido tratadas antes de encerrar TMI_Application.

FMaiuscula

Declaração `public class Function FMaiuscula(str:AnsiString):AnsiString;`

AnsiString_to_USASCII

Declaração `public class function AnsiString_to_USASCII(const pText: string): string;`

Descrição A função `AnsiString_to_USASCII` remove os acentos do texto pText

- **REFERÊNCIA**
  - Exemplo completo: https://showdelphi.com.br/dica-funcao-para-remover-acentos-de-uma-string-delphi/

RemoveAccents

Declaração `public class function RemoveAccents(const str: String ): String;`

Descrição A class function `RemoveAccents` converte caracteres acentuados para caracteres não acentuados

- **POR QUE?**
  - Preciso que as chaves dos índices não tenha acentos para evitar confusão nas pesquisas.

String_Asc_GUI_to_Asc_Ingles

Declaração `public class Function String_Asc_GUI_to_Asc_Ingles(Const S: String): String;`

SGI

Declaração `public class Function SGI(Const S: String): String;`

String_Asc_GUI_to_Asc_HTML

Declaração `public class Function String_Asc_GUI_to_Asc_HTML(Const S: String): String;`

SGH

Declaração `public class Function SGH(Const S: String): String;`

Show_GetEnv_System

Declaração `public class Procedure Show_GetEnv_System;`

FGetMem

Declaração `public class Function FGetMem(Var Buff;Const TamBuff: Word) : Boolean;`

FFreeMem

Declaração `public class Procedure FFreeMem(Var Buff;Const TamBuff: Word) ;`

CGetMem

Declaração `public class Function CGetMem(Const BuffOriginal:Pointer ;Const TamBuff: Word):Pointer;`

Descrição Retorna um ponteiro para a memória alocada e este ponteiro aponta para uma copia dos dados passado por BuffOriginal

isfileopen

Declaração `public class function isfileopen(var f:file):boolean ; overload;`

isfileopen

Declaração `public class function isfileopen(var f:text):boolean ; overload;`

CloseLst

Declaração `public class Function CloseLst:SmallInt;`

RedirecionaParaImpressora

Declaração `public class Procedure RedirecionaParaImpressora;`

RedirecionaRelatorio

Declaração `public class Procedure RedirecionaRelatorio;`

ChangeSubStr

Declaração `public class Function ChangeSubStr(aSubStrOld : AnsiString; aSubStrNew : AnsiString; S: AnsiString ):AnsiString;`

Descrição Retorna S com o `tString`(**??**) Trocado

Alias_To_Name

Declaração `public class Function Alias_To_Name(AAlias : AnsiString):AnsiString;`

CreateGUID

Declaração `public class function CreateGUID():TString;`

Descrição A class método `CreateGUID` cria um novo valor de GUID (Globally Unique Identifier).

- **RETORNA**
  - **GUID** : Novo GUID se sucesso ou string vazia se fracasso.

SetExecAsync

Declaração `public class Function SetExecAsync(aExecAsync:Byte):Byte;`

GetExecAsync

Declaração `public class Function GetExecAsync():Byte;`

ShellScript

Declaração `public class function ShellScript(aCommand:String): String;`

Descrição O método `ShellScript` executa o shell do sistema operacional e retorna o Buffer da Tela

Pascal

```
program Project1;
  uses
    mi.rtl.objectss;

  var
    s : string;
begin
  s := TObjectss.ShellScript('ls *.res');
  if s <>''
  then WriteLn(s);
end.
```

ShellExecute

Declaração `public class function ShellExecute(Const lpOperation, FileName, Params, DefaultDir: AnsiString; ShowCmd: Integer): THandle; Overload;`

ShellExecute

Declaração `public class function ShellExecute(const FileName, Params, DefaultDir: AnsiString;ShowCmd: Integer): THandle; Overload;`

ShellExecute

Declaração `public class function ShellExecute(const FileName, Params: AnsiString): THandle; Overload;`

GetIpPub

Declaração `public class Function GetIpPub:String;`

Descrição A classe function `GetIpPub` retorna o ip publico da máquina local

StrToInt

Declaração `public class function StrToInt(aStr:String):Int64;`

BooleanToStr

Declaração `public class Function BooleanToStr(Const FieldData:Boolean):AnsiString;`

DelSpcED

Declaração `public class function DelSpcED(campo : Ansistring): AnsiString;`

Delspace

Declaração `public class function Delspace(campo : Ansistring):AnsiString;`

GetNameValid

Declaração `public class function GetNameValid(aName:AnsiString):AnsiString;`

IsNumber_Real

Declaração `public class Function IsNumber_Real(Const aTemplate : ShortString):Boolean;`

IsNumber

Declaração `public class Function IsNumber(Const aTemplate : ShortString):Boolean;`

IsData

Declaração `public class Function IsData(Const aTemplate : ShortString):Boolean;`

IsHora

Declaração `public class Function IsHora(Const aTemplate : ShortString):Boolean;`

HandleEvent

Declaração `public procedure HandleEvent(var Event: TEvent); Virtual;`

ClearEvent

Declaração `public procedure ClearEvent(var Event: TEvent); Virtual;`

Change_AnsiChar

Declaração `public class Function Change_AnsiChar(campo : AnsiString; Const AnsiChar_Font,AnsiChar_Dest : AnsiChar):AnsiString;`

DeleteMask

Declaração `public class Function DeleteMask(S : tString;ValidSet: AnsiCharSet):AnsiString; overload;`

DeleteMask

Declaração `public class function DeleteMask(S: ShortString;aMask:ShortString): AnsiString; overload;`

AddMask

Declaração `public class function AddMask(S: ShortString;aMask:ShortString): AnsiString;`

CreateDB_or_DropDB

Declaração `public class function CreateDB_or_DropDB(aConnectorType : TConnectorType; aHostname, aUserName, aPassword, aDataBaseName : String; okCreateDB:Boolean ):string;`

Descrição A função `CreateDB_or_DropDB` é usada para criar ou apagar um banco de dados

- **Banco de dados possíveis:**
  - PostreSQL;
  - SqlLite;
- **Retorna**
  - True : Conseguiu criar o banco de dados;
  - False : Error na ação
    - Error possiveis:
      - Banco de dados já existe quando se quer criar;

- Banco de dados não existe quando se quer apagar;
- Banco de dados usado por outro usuário.

- **EXEMPLO**

```
// Cria banco de dados maricarai no postgresSql
Procedure TForm1.Button2sqlPQConectionClick ( Sender : TObject ) ;
  var
    s : String;
begin
  s := CreateDB_or_DropDB(PostgresSQL,'127.0.0.1',
                                      'postgres',
                                      'masterkey',
                                      'maricarai',
                                      true);
  if s = ''
  then ShowMessage('Banco de dados maricarai foi criado no postgresSql')
  else ShowMessage(s);
End;

// Apaga banco de dados maricarai no SqLite3
Procedure TForm1.Button2sqlPQConectionClick ( Sender : TObject ) ;
  var
    s : String;
begin
  s:= CreateDB_or_DropDB(PostgresSQL,'127.0.0.1',
                                      'SqLite3',
                                      'masterkey',
                                      'maricarai',
                                      false);
  if s = ''
  then ShowMessage('Banco de dados maricarai foi apagado SqLite3')
  else ShowMessage(s);
End;
```

StrNumberValid

Declaração `public class function StrNumberValid(S: AnsiString): AnsiString;`

Descrição O método `StrNumberValid` remove as mascaras do número e retorna somente números

CheckRanger

Declaração `public class function CheckRanger(S : AnsiString; aHigh, aLow: Int64; out aErr:Integer): Int64;`

Descrição o classe método `CheckRanger` checa se s está entre aHigh e aLow retorna zero se houver erro e em aErr o código do erro.

IntValid

Declaração `public class function IntValid(S : AnsiString; TypeCode:AnsiChar):Boolean;`

Descrição O método `IntValid` retorna **TRUE** se o parâmetro S for número inteiro ou **FALSE** caso contrário.

ShowHtml

Declaração `public class Procedure ShowHtml(URL:string); virtual;`

Descrição O método `ShowHtml` Executa o browser padrão do sistema operacional.

Chapter 17

Unit mi.rtl.Objects.Methods.Collection

17.1 Descrição

- A Unit `mi.rtl.Objects.Methods.Collection` implementa a classe `TCollection(??)` do pacote `mi.rtl(??)`.
  - **NOTAS**
    - Esta unit foi testada nas plataformas: linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **20/11/2021** 10:12 a 22:49 : Criada a classe `mi.rtl.Objects.Methods.Collection`

17.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.objects.Methods(??)`
- `mi.rtl.objects.methods.StreamBase.Stream(??)`

17.3 Visão Geral

`TCollection Classe`

17.4 Classes, Interfaces, Objetos e Registros

TCollection Classe

Hierarquia

TCollection > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

- A class `TCollection` implementa coleções no pacote `mi.rtl`(??).

Propriedades

Count

Declaração `published property Count: Sw_Integer Read _Count write _Count;`

Descrição Item `count`

Campos

Items

Declaração `public var Items: PItemList;`

Descrição Item list pointer

State

Declaração `public var State: Longint;`

Limit

Declaração `public Limit: Sw_Integer;`

Descrição Item `limit count`(??)

Delta

Declaração `public Delta: Sw_Integer;`

Descrição Inc `delta` size

Status

Declaração `public Status: Integer;`

Descrição `TCollection`(**??**) `status`

ErrorInfo

Declaração `public ErrorInfo: Integer;`

Descrição `TCollection`(**??**) `error`(**??**) info

Métodos

Create

Declaração `public constructor Create(ALimit, ADelta: Sw_Integer); overload; virtual;`

Destroy

Declaração `public destructor Destroy; Override;`

IndexOf

Declaração `protected function IndexOf(Item: Pointer): Sw_Integer; Virtual;`

GetItem

Declaração `protected function GetItem(Var S: tStream ): Pointer; Virtual;`

Insert

Declaração `protected procedure Insert(Item: Pointer); Virtual;`

FreeItem

Declaração `protected procedure FreeItem(Item: Pointer); Virtual;`

SetLimit

Declaração `protected procedure SetLimit(ALimit: Sw_Integer); Virtual;`

Error

Declaração `protected procedure Error(Code, Info: Integer); Virtual;`

PutItem

Declaração `protected procedure PutItem(Var S: tStream ; Item: Pointer); Virtual;`

Create_Progress1Passo

Declaração `public procedure Create_Progress1Passo(ATitle : tstring;Obs:tstring ; ATotal : Longint); Virtual;`

Set_Progress1Passo

Declaração `public procedure Set_Progress1Passo(aNumber : Longint); Virtual;`

Destroy_Progress1Passo

Declaração `public procedure Destroy_Progress1Passo; Virtual;`

MessageBox

Declaração `public function MessageBox(const Msg: AnsiString): Word; Virtual;`

At

Declaração `public function At(Index: Sw_Integer): Pointer;`

LastThat

Declaração `public function LastThat(Test: TCallbackFunBoolParam): Pointer;`

FirstThat

Declaração `public function FirstThat(Test: Pointer): Pointer;`

Pack

Declaração `public procedure Pack;`

FreeAll

Declaração `public procedure FreeAll; Virtual;`

DeleteAll

Declaração `public procedure DeleteAll;`

Free

Declaração `public procedure Free(Item: Pointer);`

Delete

Declaração `public procedure Delete(Item: Pointer);`

AtFree

Declaração `public procedure AtFree(Index: Sw_Integer);`

AtDelete

Declaração `public procedure AtDelete(Index: Sw_Integer);`

ForEach

Declaração `public procedure ForEach(Action: Pointer);`

AtPut

Declaração `public procedure AtPut(Index: Sw_Integer; Item: Pointer);`

AtInsert

Declaração `public procedure AtInsert(Index: Sw_Integer; Item: Pointer);`

Chapter 18

Unit mi.rtl.Objects.Methods.Collection.FilesStrea

18.1 Descrição

- A unit `mi.rtl.Objects.Methods.Collection.FilesStreams` implementa a classe `TFilesStreams`(??) do pacote `mi.rtl`(??).
  - **VERSÃO**:
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - 2021-12-18
      - 14:42 a .. - T12 Criar a unit `mi.rtl.Objects.Methods.Collection.FilesStreams` e a classe `TFilesStreams`(??)
      - 2021-12-20
      - 17:11 a 18:30 - T12 Criar a classe `TFilesStreams`(??)
      - 20:20 a 22:54 - T21 Criar exemplo de uso da classe `TFilesStreams`(??)

18.2 Uses

- Classes
- SysUtils
- mi.rtl.objects.consts.MI_MsgBox
- mi.rtl.objects.consts.progressdlg_if(??)
- mi.rtl.objects.methods.StreamBase.Stream(??)
- mi.rtl.objects.methods.StreamBase.Stream.FileStream(??)
- mi.rtl.objects.methods.Collection(??)

18.3 Visão Geral

TFilesStreams Classe

18.4 Classes, Interfaces, Objetos e Registros

TFilesStreams Classe

Hierarquia

TFilesStreams > TCollection(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

- A classe TFilesStreams é usada para armazenar todos os arquivos abertos pelo sistema para poder fecha-los caso o programa aborte inesperadamente.

  – EXEMPLO DE USO

```
procedure TMi_Rtl_Tests.TabSheet_TFilesStreamsEnter(Sender: TObject);
  var
   i,L : integer;
   s:AnsiString;

begin
  filesStreams.DeleteAll;
  StringGrid1.Clear;

  filesStreams.Mask := edit2.Text;
  StringGrid1.RowCount := filesStreams.Count+1;
```

```
  LabelCount2.Caption := Format('FilesStreams.Count %d',[filesStreams.Count]);
  LabelCount2.Show;
  L := 0;
  StringGrid1.Cells[0,l] := 'Seq';
  StringGrid1.Cells[1,l] := 'FileName';
  StringGrid1.Cells[2,l] := 'FileSize';
  inc(l);
  if filesStreams.Count > 0
  then begin
          for i := 0 to filesStreams.Count-1 do
          with filesStreams.FileByNum(i) do
          begin
            StringGrid1.Cells[0,l] := Format('%d',[l]);
            StringGrid1.Cells[1,l] := FileName;
            s := Format('%d',[FileSize(FileName)]);
            StringGrid1.Cells[2,l] := s ;
            inc(L);
          end;
       end;

end;

procedure TMi_Rtl_Tests.Edit2Change(Sender: TObject);
begin
  TabSheet_TFilesStreamsEnter(Self);
end;
```

Propriedades

Mask

Declaração `published property Mask :  AnsiString Read _mask write SetMask;`

Descrição

- A propriedade `Mask` é usada como filtro na função SysUtils.FindFirst

Métodos

SetMask

Declaração `protected Procedure SetMask(a_Mask :  AnsiString);`

Descrição

- O método `SetMask` é usado para filtrar os arquivo da pasta corrente do banco de dados.
  - **Nota**
    - Para compreender essa função é bom ler o exemplo: `TFiles.FindFiles`(**??**)

Create

Declaração `public CONSTRUCTOR Create;`

FileByNum

Declaração `public Function FileByNum(const Index:Longint):TFileStream;`

FileByName

Declaração `public Function FileByName(const aFileName:AnsiString):TFileStream;`

CopyFiles

Declaração `public Function CopyFiles(aPathDest:PathStr):Integer;`

DeleteFiles

Declaração `public Function DeleteFiles():Integer;`

Error

Declaração `public PROCEDURE Error(Code, Info:  Integer); Override;`

Create_Progress1Passo

Declaração `public Procedure Create_Progress1Passo(ATitle : tString;Obs:tString ; ATotal : Longint); Override;`

Set_Progress1Passo

Declaração `public Procedure Set_Progress1Passo(aNumero_Segundos_que_deve_esperar : Longint); Override;`

Destroy_Progress1Passo

Declaração `public Procedure Destroy_Progress1Passo; Override;`

Chapter 19

Unit mi.rtl.Objects.Methods.Collection.SortedCol

19.1 Descrição

- A Unit `mi.rtl.Objects.Methods.Collection.SortedCollection` implementa a classe `TSortedCollection`(??) do pacote `mi.rtl`(??).
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **30/11/2021**
      - 9:45 a 11:47 : Criada a unit `mi.rtl.Objects.Methods.Collection.SortedCollection` e a classe `TSortedCollection`(??)

19.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.objects.Methods`(??)
- `mi.rtl.objects.methods.StreamBase`(??)
- `mi.rtl.objects.methods.Collection`(??)

19.3 Visão Geral

`TSortedCollection Classe`

19.4 Classes, Interfaces, Objetos e Registros

TSortedCollection Classe

Hierarquia

TSortedCollection > `TCollection`(**??**) > `TObjectsMethods`(**??**) > `TObjectsConsts`(**??**) > TObjectsTypes

Descrição

- A class `TSortedCollection` implementa coleções ordenadas de objetos.
  - EXEMPLO DE USO

```
???
```

Propriedades

tstrings

Declaração `protected property tstrings[Index: Sw_Integer]: tstring Read Getstrings write Setstrings;`

Descrição =n specifies the maximum number of AnsiCharacters

Campos

Duplicates

Declaração `public Duplicates: Boolean;`

Métodos

Create

Declaração `public CONSTRUCTOR Create(ALimit, ADelta: Sw_Integer); overload; override;`

Descrição `Duplicates`(**??**) flag

KeyOf

Declaração `protected FUNCTION KeyOf(Item: Pointer): Pointer; Virtual;`

IndexOf

Declaração `protected FUNCTION IndexOf(Item: Pointer): Sw_Integer; Override;`

Compare

Declaração `protected FUNCTION Compare(Key1, Key2: Pointer): Sw_Integer; Virtual;`

Search

Declaração `protected FUNCTION Search(Key: Pointer; Var Index: Sw_Integer): Boolean; Virtual;`

Insert

Declaração `protected PROCEDURE Insert(Item: Pointer); Override;`

GetMaxLength

Declaração `protected Function GetMaxLength():Integer;`

Chapter 20

Unit mi.rtl.Objects.Methods.Collection.SortedCol

20.1 Descrição

- A Unit `mi.rtl.Objects.Methods.Collection.SortedCollection.StrCollection` implementa a classe `TStrCollection`(??) do pacote `mi.rtl`(??).
  - **NOTA**
    - A diferença de `tstringCollection`(??) para `TStrCollection`(??) é que a primeira é uma coleção de shortstring e a segunda é uma coleção de PByteArray usada para trabalhar com AnsiString;
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **30/11/2021**
      - 11:47 a 11:47 : Criada a unit `mi.rtl.Objects.Methods.Collection.SortedCollection.StrCollect` e a classe **TStCollection**
      - 14:00 a 14:20 : Documentar a unit

20.2 Uses

- Classes
- SysUtils
- mi.rtl.objects.Methods.Collection.SortedCollection(??)
- mi.rtl.objects.methods.Streambase.Stream(??)

20.3 Visão Geral

TStrCollection Classe

TUnSortedStrCollection Classe

20.4 Classes, Interfaces, Objetos e Registros

TStrCollection Classe

Hierarquia

TStrCollection > TSortedCollection(??) > TCollection(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

A classe TStrCollection implementa uma coleção de **AnsiString**

Métodos

Compare

Declaração `protected FUNCTION Compare(Key1, Key2: Pointer): Sw_Integer; Override;`

GetItem

Declaração `public FUNCTION GetItem(Var S: TStream): Pointer; Override;`

FreeItem

Declaração `protected PROCEDURE FreeItem(Item: Pointer); Override;`

PutItem

Declaração `protected PROCEDURE PutItem(Var S: TStream; Item:  Pointer); Override;`

TUnSortedStrCollection Classe

Hierarquia

TUnSortedStrCollection > `TStrCollection`(??) > `TSortedCollection`(??) > `TCollection`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

A classe `TUnSortedStrCollection` implementa uma coleção de **AnsiString** na ordem original de inserção das AnsiStrings

Métodos

Insert

Declaração `public PROCEDURE Insert(Item:  Pointer); Override;`

Chapter 21

Unit mi.rtl.Objects.Methods.Collection.SortedCol

21.1 Descrição

- A Unit `mi.rtl.Objects.Methods.Collection.SortedCollection.StringCollection` implementa a classe `tstringCollection(??)` do pacote `mi.rtl(??)`.
  - **NOTA**
    - A diferença de `tstringCollection(??)` para `TStrCollection(??)` é que a primeira é uma coleção de shortstring e a segunda é uma coleção de PByteArray usada para trabalhar com AnsiString;
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **30/11/2021**
      - 14:20 a 15:15 : Criada a unit `mi.rtl.Objects.Methods.Collection.SortedCollection.StringColl` e a classe `tstringCollection(??)`

21.2 Uses

- Classes
- SysUtils
- mi.rtl.objects.methods.Collection.SortedCollection(??)
- mi.rtl.objects.methods.StreamBase.Stream(??)

21.3 Visão Geral

TStringCollection Classe

TUnSortedStringCollection Classe

21.4 Classes, Interfaces, Objetos e Registros

TStringCollection Classe

Hierarquia

TStringCollection > TSortedCollection(??) > TCollection(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

A classe TStringCollection implementa uma coleção de **AnsiString**

Métodos

GetItem

Declaração `public FUNCTION GetItem(Var S: TStream): Pointer; Override;`

Compare

Declaração `public FUNCTION Compare(Key1, Key2: Pointer): Sw_Integer; Override;`

FreeItem

Declaração `public PROCEDURE FreeItem(Item: Pointer); Override;`

PutItem

Declaração `public PROCEDURE PutItem(Var S: TStream; Item:  Pointer); Override;`

TUnSortedStringCollection Classe

Hierarquia

TUnSortedStringCollection > `TStringCollection`(**??**) > `TSortedCollection`(**??**) > `TCollection`(**??**) > `TObjectsMethods`(**??**) > `TObjectsConsts`(**??**) > TObjectsTypes

Descrição

A classe `TUnSortedStringCollection` implementa uma coleção de **Shortstring** na ordem original de inserção dos ShortStrings

Métodos

Insert

Declaração `public PROCEDURE Insert(Item:  Pointer); Override;`

Chapter 22

Unit mi.rtl.Objects.Methods.Collection.Sortedcoll

22.1 Descrição

- A Unit `mi.rtl.Objects.Methods.Collection.Sortedcollection.Stringcollection.Collectionstring` implementa a classe `TCollectionString`(**??**) do pacote `mi.rtl`(**??**).
  - **NOTA**
    - Essa classe foi criada para transformar Lista `PSItem`(**??**) eem `TCollection`(**??**) de strings;
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **07/12/2021**
      - 08:00 a 10:07 : Criada a unit `mi.rtl.Objects.Methods.Collection.Sortedcollection.Stringcoll` e a classe `TCollectionString`(**??**)

22.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.objects.Methods`(??)
- `mi.rtl.objects.methods.Collection.Sortedcollection.StringCollection`(??)

22.3 Visão Geral

`TCollectionString Classe`

22.4 Classes, Interfaces, Objetos e Registros

TCollectionString Classe

Hierarquia

TCollectionString >

Descrição

omit if `TListBoxRec`(??) is defined else where

Propriedades

AnsiStrings

Declaração `public property AnsiStrings[Index: Sw_Integer]: AnsiString Read GetAnsiStrings;`

Descrição Ler a string sem os caracteres de controle

Campos

Ordem

Declaração `public Ordem:Boolean;`

Descrição

- Se True insere em `ordem` alfabética

FoundTesteCompleto

Declaração `public FoundTesteCompleto:Boolean;`

Métodos

Create

Declaração `public constructor Create(ALimit, ADelta: Sw_integer;AOrdem:Boolean); overload; virtual;`

CreateLista

Declaração `public constructor CreateLista(AOrdem:Boolean;aLista:tString;const aFoundTesteCompleto:Boolean);`

NewStr

Declaração `public function NewStr(S : AnsiString):Boolean;`

Append

Declaração `public function Append(S : AnsiString):Boolean;`

AddSItem

Declaração `public procedure AddSItem(P : PSItem;OkDisposeSItems:Boolean); Overload;`

AddSItem

Declaração `public procedure AddSItem(P : PSItem); Overload;`

PListSItem

Declaração `public function PListSItem:  PSItem;`

Get_Html_List

Declaração `public function Get_Html_List:AnsiString;`

Descrição Retorna Uma sequencia de <li>

Found

Declaração `public function Found(const akey:tString):Boolean;`

GetMaiorString

Declaração `public function GetMaiorString(Const aConjDespreze:AnsiCharSet;aIgnore_ShowWid:Boolean) :  Byte; Overload;`

GetMaiorString

Declaração `public function GetMaiorString(Const aConjDespreze:AnsiCharSet) : Byte; Overload;`

GetMaiorAnsiString

Declaração `public function GetMaiorAnsiString() :  Integer; Overload;`

Clone

Declaração `public function Clone:TCollectionString;`

Search

Declaração `public function Search(Key: Pointer; Var Index: Sw_Integer): Boolean; Override;`

FormatStr

Declaração `public procedure FormatStr(LengthMaxCol: Integer);`

FreeItem

Declaração `public procedure FreeItem(Item: Pointer); Override;`

WriteSItems

Declaração `public class procedure WriteSItems(var S: TCollectionString; Const Items: PSItem);`

CloneSItems

Declaração `public class function CloneSItems(Const Items: PSItem):PSItem;`

CopyTemplateFrom

Declaração `public class Function CopyTemplateFrom(Const aTemplate:tString): tString;`

22.5 Tipos

TStringCollection

Declaração `TStringCollection = mi.rtl.objects.methods.Collection.Sortedcollection.StringCollection.TStringCollection;`

Chapter 23

Unit mi.rtl.objects.Methods.dates

23.1 Descrição

-A Unit `mi.rtl.objects.Methods.dates` implementa a classe `TDates(??)`.

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
    - **27/01/99 05:12 a 08:00** - Retirei do Db_Generic e Db_Global todas as rotinas com referência a datas.
    - **27/01/99 09:05 a 09:30** - Debugar as rotinas de datas utilizando o ano 2000.
    - **27/01/99 09:30 a** - Todos os relatórios que necessitam de datas de início e fim do período devem ser inicializadas com as datas mínimas e máximas.
    - **15/12/21 13:50 a 14:30** - Criar classe mi.rtl.objects.Methods.dates.Tdates e adicionar as rotinas de datas do passado.

23.2 Uses

- `Classes`
- `SysUtils`

- `DateUtils`
- `dos`
- `mi.rtl.objects.Methods(??)`
- `mi.rtl.objects.consts.MI_MsgBox`

23.3 Visão Geral

`TDates Classe`

23.4 Classes, Interfaces, Objetos e Registros

TDates Classe

Hierarquia

TDates > `TObjectsMethods(??)` > `TObjectsConsts(??)` > TObjectsTypes

Descrição

- A classe `TDates` contém todas os métodos necessários para acessar data no formato de 3 bytes.
  - NOTA**
    - Formato de data e hora tratado por essa classe:
      - type `TypeData(??)` = Record dia:byte;mes:Byte;ano : byte; End;
      - type `TipoHora(??)` = record H,M,S,S100 : Word; end;
      - Type TData_e_Hora_Compactada = Longint;

Campos

AnoLimit

Declaração `public const AnoLimit :  Byte = 30;`

DataMinima

Declaração `public const DataMinima :  TypeData = (dia:1;mes:1 ;ano:_AnoLimit);`

DataMaxima

Declaração `public const DataMaxima : TypeData = (dia:31;mes:12;ano:_AnoLimit-1);`

ArrayStrMeses

Declaração `public const ArrayStrMeses : Array[TMeses] of string = ('', 'Janeiro', 'Fevereiro', 'Marco', 'Abril', 'Maio', 'Junho', 'Julho', 'Agosto', 'Setembro', 'Outubro', 'Novembro', 'Dezembro');`

StrDiaSemana

Declaração `public const StrDiaSemana : Array[0..6] of String[7] = ('Sabado ', 'Domingo', 'Segunda', 'Terca ', 'Quarta ', 'Quinta ', 'Sexta ');`

HoraInicial

Declaração `public const HoraInicial : TipoHora = (H : 0; M : 0; S : 0; S100 : 0 );`

DataSistOp

Declaração `public DataSistOp: TypeData; static;`

TempData

Declaração `public TempData: TypeData; static;`

Métodos

juliano

Declaração `public class function juliano(d,m,a : SmallInt) : TRealNum;`

DifeData

Declaração `public class function DifeData(diaAtual, mesAtual, anoAtual,diaAnterior , mesAnterior , anoAnterior : byte ) :Longint; Overload;`

DifeData

Declaração `public class function DifeData(Const DAntBuff,DAtuBuff:TypeData) : Longint; Overload;`

DifeData

Declaração `public class function DifeData(Const DatAnterior:TypeData;Const DatAtual : TypeData; Const Operador:AnsiChar; Const operando : Longint) : Boolean; Overload;`

somaData

Declaração `public class procedure somaData(var dia,mes:Byte; Var ano : SmallInt ; diasAsomar : Integer); Overload;`

somaData

Declaração `public class procedure somaData(var dia,mes,ano : byte ; diasAsomar : Integer); Overload;`

SomaData

Declaração `public class procedure SomaData(Var Buff:TypeData;Prazo: Integer) ; Overload;`

FSomaData

Declaração `public class function FSomaData(Buff:TypeData;Prazo: Integer):TString ; Overload;`

SomaDataEmMeses

Declaração `public class procedure SomaDataEmMeses(Const DataFont : TypeData; Const Meses : SmallInt; Var DataDest : TypeData);`

SomeDataPara

Declaração `public class procedure SomeDataPara(Const Buff1:TypeData;Var Buff2:TypeData ;Const Prazo: Integer) ;`

Dtjuliana

Declaração `public class function Dtjuliana(Var Buff) : TRealNum;`

moveData

Declaração `public class procedure moveData(var dataFonte, dataDestino);`

ConvNomeData

Declaração `public class function ConvNomeData(Const NomeArqFonte :String; NomeArqDestino : String; Const Mes,Ano : Byte; var Mensagem : String ) : String;`

StrDataMesAno

Declaração `public class function StrDataMesAno(Const mes,Ano:byte) : String;`

DiaMaxDoMes

Declaração `public class function DiaMaxDoMes(Const Mes : Byte;Ano : Integer) : Byte;`

FDiaSemana

Declaração `public class function FDiaSemana(Var BuffData) : Byte;`

FStrDiaSemana

Declaração `public class function FStrDiaSemana(Var data ) : String;`

StrMes

Declaração `public class function StrMes(Const mes : Word ) : String;`

StrData

Declaração `public class function StrData(Const Dia,mes,ano: Word ; Const ch :AnsiChar) : string;`

StringData

Declaração `public class function StringData(Const Buff :TypeData;Const Ch : AnsiChar) : String;`

GetDataSistOp

Declaração `public class function GetDataSistOp(Var Buff;const Separador:AnsiChar):String;`

FGetDataSistOp

Declaração `public class function FGetDataSistOp(const Separador:AnsiChar):String;`

GetDateSystem

Declaração `public class function GetDateSystem(const DateMask:TDateMask):String;`

GetHourSystem

Declaração `public class function GetHourSystem(const HourMask :THourMask):String;`

GetFTimeDos

Declaração `public class function GetFTimeDos:Longint; Overload;`

GetFTimeDos

Declaração `public class function GetFTimeDos(Var wSec1000:SmallInt):Longint; Overload;`

GetFTimeDos

Declaração `public class function GetFTimeDos(Var wSec100:Byte;Var wSec1000:SmallInt):Longint; Overload;`

GetFTimeDos_Valid

Declaração `public class function GetFTimeDos_Valid(aTime_UltimoAcesso:Longint;aMinutos_de_tolerancia_do_Ultimo_Acesso:byte):Boo`

PackDate

Declaração `public class function PackDate(Const Data:TypeData):Longint; Overload;`

PackDate

Declaração `public class function PackDate(Const Data:String; Const Mask: TDateMask):Longint; Overload;`

PackDate

Declaração `public class function PackDate(Const Data:String; Const Mask: TDateMask; TimePack:Longint):Longint; Overload;`

PackHour

Declaração `public class function PackHour(Const Hora:String; Const Mask: THourMask; TimePack:Longint):Longint; Overload;`

UnPackDate

Declaração `public class function UnPackDate(Const TimePack:Longint):TypeData;`

UnPackHora

Declaração `public class procedure UnPackHora(Const TimePack:Longint;Var Hora : TipoHora);`

PackHora

Declaração `public class procedure PackHora(Const Hora :  TipoHora; Var TimePack:Longint);`

PackDateHora

Declaração `public class procedure PackDateHora(Data:TypeData; Hora :TipoHora; Var TimePack:Longint);`

UnPackDateHora

Declaração `public class procedure UnPackDateHora(Const TimePack:Longint; Var Data:TypeData; Var Hora :TipoHora);`

StringTimeD

Declaração `public class function StringTimeD(Const TimePack:Longint;Const Ch : AnsiChar) : String;`

StringTimeH

Declaração `public class function StringTimeH(Const TimePack:Longint) : String;`

StringTimeHSemPonto

Declaração `public class function StringTimeHSemPonto(Const TimePack:Longint) : String;`

FIncAno

Declaração `public class function FIncAno(Ano:SmallInt) : Byte;`

FDecAno

Declaração `public class function FDecAno(Ano:SmallInt) : Byte;`

FAno

Declaração `public class function FAno(Ano:SmallWord) : SmallWord;`

FAno2Digito

Declaração `public class function FAno2Digito(Ano : SmallWord):Byte;`

FAnoDoIndex

Declaração `public class function FAnoDoIndex(Const Dia,Ano : byte):String;`

StrAno

Declaração `public class function StrAno(ano : SmallInt ) : String;`

StrToDate

Declaração `public class function StrToDate(aStrDate:String; Const Mask: TDateMask):PTypeData;`

DateToStr

Declaração `public class function DateToStr(Const aDate:TypeData;Const Mask: TDateMask) : String; Overload;`

DateToStr

Declaração `public class function DateToStr(Const aDate:Longint;Const Mask: TDateMask) : String; Overload;`

DateToDateTime

Declaração `public class function DateToDateTime(aDate:TypeData):`
`System.TDateTime; Overload;`

DateToDateTime

Declaração `public class function`
`DateToDateTime(aTimePack:Longint):System.TDateTime; Overload;`

DateTimeToDate

Declaração `public class function DateTimeToDate(aDateTime:TDateTime):TypeData;`
`Overload;`

DateTimeToDateStr

Declaração `public class function`
`DateTimeToDateStr(aDateTime:TDateTime):String; Overload;`

DateTimeToTimeStr

Declaração `public class function`
`DateTimeToTimeStr(aDateTime:TDateTime):String;`

DateTimeToDateTimeDos

Declaração `public class function`
`DateTimeToDateTimeDos(aDateTime:TDateTime):Longint; Overload;`

DateTimeDosToStr

Declaração `public class function`
`DateTimeDosToStr(aTimePack:Longint;Mask:TDateMask):String;`

StrToDateTime

Declaração `public class function StrToDateTime(aDataTime:String;Mask:TDateMask):TDateTime; Overload;`

StrToDateTimeDos

Declaração `public class function StrToDateTimeDos(aDataTime:String;Mask:TDateMask):Longint; Overload;`

StrToHora

Declaração `public class function StrToHora( aStrHora:String; Const Mask: THourMask):TipoHora;`

StrToHour

Declaração `public class function StrToHour(Const aStrHora:String; Const Mask: THourMask):Longint; Overload;`

StrToHour

Declaração `public class function StrToHour(Const aStrHora:String; Const Mask: THourMask;TimePack:Longint):Longint; Overload;`

HourToStr

Declaração `public class function HourToStr(Const aStrHora:Longint; Const Mask: THourMask;Const OkSpc : Boolean):String; Overload;`

HourToStr

Declaração `public class function HourToStr(Const Hora:TipoHora; Const Mask: THourMask;Const OkSpc : Boolean):String; Overload;`

HourToDateTime

Declaração `public class function HourToDateTime(Const aTimePack:Longint): TDateTime; Overload;`

str2jul

Declaração `public class function str2jul(DateStr:string): longint;`

jul2str

Declaração `public class function jul2str(JulDate:longint) :string;`

Julian

Declaração `public class function Julian( Year, Month, Day : Word ) : LongInt;`

LeapYear

Declaração `public class function LeapYear( Year : Word ) : Boolean ;`

DiaMaxMes

Declaração `public class function DiaMaxMes(Const DataAtual :TypeData) : byte;`

DateMask_to_Str

Declaração `public class function DateMask_to_Str(Const aDateMask : TDateMask ):String;`

Str_to_DateMask

Declaração `public class function Str_to_DateMask(aStrDate:String):TDateMask;`

HourMask_to_Str

Declaração `public class function HourMask_to_Str(Const aHourMask : THourMask ):String;`

ValidDate

Declaração `public class function ValidDate( aData : TypeData):Byte;`

ValidHour

Declaração `public class function ValidHour( H,M,S,S100 : Word):Byte;`

DifHoraEmSegundos

Declaração `public class function DifHoraEmSegundos(Const HAtu,HAnt : TipoHora ):Longint;`

DifHora_Retorne_TipoHora

Declaração `public class function DifHora_Retorne_TipoHora(Const HAtu,HAnt : TipoHora ):TipoHora; Overload;`

DifHora_Retorne_TipoHora

Declaração `public class function DifHora_Retorne_TipoHora(Const HAtu,HAnt : Longint ):TipoHora; Overload;`

DifHora_Retorne_Horas_Fracao

Declaração `public class function DifHora_Retorne_Horas_Fracao(const HAtu,HAnt : TipoHora ):Double; Overload;`

DifHora_Retorne_Horas_Fracao

Declaração `public class function DifHora_Retorne_Horas_Fracao(const HAtu,HAnt : Longint ):Double; Overload;`

DifHora_Retorne_Minutos

Declaração `public class function DifHora_Retorne_Minutos(Const HAtu,HAnt : TipoHora ):Longint; Overload;`

DifHora_Retorne_Minutos

Declaração `public class function DifHora_Retorne_Minutos(Const HAtu,HAnt : Longint ):Longint; Overload;`

DifHora_Retorne_Time

Declaração `public class function DifHora_Retorne_Time(Const HAtu,HAnt : TipoHora ):Longint; Overload;`

DifHora_Retorne_Time

Declaração `public class function DifHora_Retorne_Time(Const HAtu,HAnt : Longint ):Longint; Overload;`

SegundosEmHora

Declaração `public class function SegundosEmHora(Const Segundos:Longint):String;`

New_Lista_Str_Meses

Declaração `public class function New_Lista_Str_Meses:  PSitem;`

getDateStr

Declaração `public class function getDateStr:tstring ;`

getTimeStr

Declaração `public class function getTimeStr:tstring ;`

Chapter 24

Unit mi.rtl.Objects.Methods.Db.Tb_Access

24.1 Descrição

24.2 Uses

- classes
- SysUtils
- Dos
- crt
- strings
- Memory
- mi.rtl.types(??)
- mi.rtl.objects.consts.MI_MsgBox
- mi.rtl.Consts.StrError(??)
- mi.rtl.files(??)
- mi.rtl.objects.Methods.Exception(??)
- mi.rtl.objects.methods.StreamBase.Stream(??)
- mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMemory(??)
- mi.rtl.objects.methods.Collection.FilesStreams(??)

- mi.rtl.objects.methods.Collection(??)
- mi.rtl.objects.methods.Collection.SortedCollection(??)
- mi.rtl.objects.methods.Collection.SortedCollection.StringCollection.CollectionString(??)
- mi.rtl.objects.methods.StreamBase.Stream.FileStream(??)
- mi.rtl.objects.Methods.dates(??)
- mi.rtl.objects.Methods.System(??)

24.3 Visão Geral

TTb_Access_types Classe

TTb_Access_consts Classe

TDataFile Classe

TTb_Access Classe

TFilesOpens Classe

24.4 Classes, Interfaces, Objetos e Registros

TTb_Access_types Classe

Hierarquia

TTb_Access_types > TObjectsSystem(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

A classe TTb_Access_types é usada para declarar todos os types da classe TTb_Access(??)

Campos

maxKeyLen

Declaração `public const maxKeyLen = 254 ;`

Descrição Tamanho máximo da Chave

pageSize

Declaração `public const pageSize = 510 ;`

Descrição Número máximo de chaves permitido em uma página

order

Declaração `public const order = 255 ;`

Descrição Número mínimo de chaves permitido em uma página

maxHeight

Declaração `public const maxHeight = 4 ;`

Descrição Número máximo de níveis na árvore B+

maxDataRecSize

Declaração `public const maxDataRecSize = High (SmallWord)-1 ;`

TTb_Access_consts Classe

Hierarquia

TTb_Access_consts > `TTb_Access_types`(??) > `TObjectsSystem`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

A classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`(??)

Campos

ErroDOS

Declaração `public const ErroDOS : TErroDOS = (Status:  0);`

FileName_Transaction

Declaração `public const FileName_Transaction : PathStr = '';`

ok_Debug_Transaction

Declaração `public const ok_Debug_Transaction : Boolean = False ;`

OkTransaction

Declaração `public const OkTransaction : Boolean = True;`

Descrição False = desabilita transação

OkAddRecFirstFree

Declaração `public const OkAddRecFirstFree : Boolean = True;`

Descrição A Constante `OkAddRecFirstFree` é usado para habilitar aproveitamento do espaço deletado.

- **NOTA**
  - True = O procedimento AddRec aproveita o espaço dos registros deletados.
  - False = O procedimento AddRec Não aproveita o espaço dos registros deletados. ou melhor o novo registro e adicionado no final do arquivo.

MaxFilesOpens

Declaração `public const MaxFilesOpens : Byte = 20;`

Descrição Máximo de arquivo que o DOS pode abrir sem a chamada da Interrupção No. $67

MemoriaLivreEmTaPageStack

Declaração `public const MemoriaLivreEmTaPageStack = 64 *1024;`

Descrição 16K de Memória fica livre em setBufIndex

MsgOkDuplicidade

Declaração `public const MsgOkDuplicidade :  Boolean = True;`

Descrição Indica se deve dar mensagem de chave em duplicidade

Neterr

Declaração `public const Neterr :  SmallInt = 0 ;`

Descrição Indica se houve erro na rede

MaxFiles

Declaração `public const MaxFiles :  Byte = 254;`

OkTestaAberturaDeArquivo

Declaração `public const OkTestaAberturaDeArquivo :  Boolean = true;`

MaxPageEmMemoria

Declaração `public const MaxPageEmMemoria = 20;`

Descrição 10 ficou mais lento do que 20; 100 ficou mais lento do que 20

MinPageEmMemoria

Declaração `public const MinPageEmMemoria = 3;`

Descrição Usado no Buffer dos índice

NoDuplicates

Declaração `public const NoDuplicates = 0;`

Duplicates

Declaração `public const Duplicates = 1;`

FileHeaderSize

Declaração `public const FileHeaderSize = sizeof(TsImagemHeader);`

Descrição A constante **#name** é usado para guardar o número de chaves do Indice

MinDataRecSize

Declaração `public const MinDataRecSize = FileHeaderSize;`

ItemOverhead

Declaração `public const ItemOverhead = SizeOf(TaItem) - Sizeof(TaKeyStr ) + 1;`

PageOverhead

Declaração `public const PageOverhead = SizeOf(TaPage) - SizeOf(TItemArray );`

TaRecBuf

Declaração `public const TaRecBuf : TaRecordBufPtr = nil;`

Const_Ext_Tabela

Declaração `public const Const_Ext_Tabela = '.Tb';`

Const_Ext_Indice_da_tebela

Declaração `public const Const_Ext_Indice_da_tebela = '.Ix';`

Const_Ext_Tabela_com_a_copia_da_versao_anterior_da_tabela

Declaração `public const Const_Ext_Tabela_com_a_copia_da_versao_anterior_da_tabela = '.Tb_';`

Const_Ext_Tabela_de_objetos_vinculados_a_tabela

Declaração `public const Const_Ext_Tabela_de_objetos_vinculados_a_tabela = '.TbO';`

Const_Ext_Tabela_com_os_registro_duplicados

Declaração `public const Const_Ext_Tabela_com_os_registro_duplicados = '.Tb1';`

Const_Ext_Tebela_com_as_Tabelas

Declaração `public const Const_Ext_Tebela_com_as_Tabelas = '.TbT';`

Const_Ext_Indice_da_Tabela_das_tabelas

Declaração `public const Const_Ext_Indice_da_Tabela_das_tabelas = '.IxT';`

Const_Ext_Tabela_com_os_Indices

Declaração `public const Const_Ext_Tabela_com_os_Indices = '.TbI';`

Const_Ext_Indice_da_tebala_de_Indices

Declaração `public const Const_Ext_Indice_da_tebala_de_Indices = '.IxI';`

Const_Ext_Tabela_com_os_Relationships

Declaração `public const Const_Ext_Tabela_com_os_Relationships = '.TbR';`

Const_Ext_Indice_da_tebala_dos_Relationships

Declaração `public const Const_Ext_Indice_da_tebala_dos_Relationships = '.IxR';`

Const_Ext_Tabela_com_todos_os_campos_de_todas_as_tabelas

Declaração `public const Const_Ext_Tabela_com_todos_os_campos_de_todas_as_tabelas = '.TbC';`

Const_Ext_Indice_da_tabela_com_todos_os_campos

Declaração `public const Const_Ext_Indice_da_tabela_com_todos_os_campos = '.IxC';`

Const_Ext_Tabela_de_Parametros

Declaração `public const Const_Ext_Tabela_de_Parametros = '.TbP';`

Const_Ext_Tabela_de_Usuarios

Declaração `public const Const_Ext_Tabela_de_Usuarios = '.TbU';`

Const_Ext_Indice_da_Tabela_de_Usuarios

Declaração `public const Const_Ext_Indice_da_Tabela_de_Usuarios ='.IxU';`

Const_Ext_Backup_da_Tabela

Declaração `public const Const_Ext_Backup_da_Tabela = '.TbK';`

Const_Ext_Banco_de_dados_Access

Declaração `public const Const_Ext_Banco_de_dados_Access = '.Mdb';`

Const_Ext_Banco_de_dados_Access_Secundario

Declaração `public const Const_Ext_Banco_de_dados_Access_Secundario = '.ldb';`

Const_Ext_Banco_de_dados_Interbase

Declaração `public const Const_Ext_Banco_de_dados_Interbase = '.GDB';`

Const_Ext_Tabela_Paradox

Declaração `public const Const_Ext_Tabela_Paradox = '.Db';`

Const_Ext_Tabela_Paradox_Px

Declaração `public const Const_Ext_Tabela_Paradox_Px = '.Px';`

Const_Ext_Tabela_Paradox_Yx

Declaração `public const Const_Ext_Tabela_Paradox_Yx = '.Yx';`

Const_Ext_Tabela_DBF

Declaração `public const Const_Ext_Tabela_DBF = '.DBF';`

Const_Ext_Tabela_DBF_Ndx

Declaração `public const Const_Ext_Tabela_DBF_Ndx = '.Ndx';`

Const_Ext_Tabela_DBF_Idx

Declaração `public const Const_Ext_Tabela_DBF_Idx = '.Idx';`

Const_Ext_Tabela_Word

Declaração `public const Const_Ext_Tabela_Word = '.Doc';`

Const_Ext_Tabela_Excel

Declaração `public const Const_Ext_Tabela_Excel = '.Xls';`

Const_Ext_Array

Declaração
```
public const Const_Ext_Array :  Array[1..24] of string[4] = (
 Const_Ext_Tabela, Const_Ext_Indice_da_tebela,
 Const_Ext_Tabela_com_a_copia_da_versao_anterior_da_tabela,
 Const_Ext_Tabela_de_objetos_vinculados_a_tabela,
 Const_Ext_Tabela_com_os_registro_duplicados,
 Const_Ext_Tebela_com_as_Tabelas, Const_Ext_Indice_da_Tabela_das_tabelas,
 Const_Ext_Tabela_com_os_Indices, Const_Ext_Indice_da_tebala_de_Indices,
 Const_Ext_Tabela_com_os_Relationships,
 Const_Ext_Indice_da_tebala_dos_Relationships,
 Const_Ext_Tabela_com_todos_os_campos_de_todas_as_tabelas,
 Const_Ext_Indice_da_tabela_com_todos_os_campos,
```

```
Const_Ext_Tabela_de_Parametros,
Const_Ext_Tabela_de_Usuarios, Const_Ext_Indice_da_Tabela_de_Usuarios,
Const_Ext_Backup_da_Tabela,
Const_Ext_Banco_de_dados_Access, Const_Ext_Banco_de_dados_Access_Secundario,
Const_Ext_Tabela_DBF, Const_Ext_Tabela_DBF_Ndx, Const_Ext_Tabela_DBF_Idx,
Const_Ext_Tabela_Word, Const_Ext_Tabela_Excel );
```

TDataFile Classe

Hierarquia

TDataFile > TTb_Access_consts(??) > TTb_Access_types(??) > TObjectsSystem(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

A classe TDataFile é usada para acessar arquivos em disco TTb_Access(??)

Campos

DataFile

Declaração `public DataFile:  ^DataFile;`

Ok_CloseDataFile

Declaração `public Ok_CloseDataFile:  Boolean;`

Métodos

Create

Declaração
```
public Constructor Create(const aDataFile :  DataFile;Const
aOk_CloseDataFile:Boolean); virtual;
```

Destroy

Declaração `public Destructor Destroy; Override;`

SetDataFile

Declaração `public procedure SetDataFile(const aDataFile : DataFile;Const aOk_CloseDataFile : Boolean);`

TTb_Access Classe

Hierarquia

TTb_Access > `TTb_Access_consts`(??) > `TTb_Access_types`(??) > `TObjectsSystem`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

no description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`(??)

Métodos

Create

Declaração `public class Procedure Create;`

Destroy

Declaração `public class Procedure Destroy;`

FExisteCodigo

Declaração `public class function FExisteCodigo(Var IxF:IndexFile; Const Codigo:tString):Boolean;`

CreateTAccess

Declaração `public class Procedure CreateTAccess;`

DestroyTAccess

Declaração `public class Procedure DestroyTAccess;`

EscrevaTurboError

Declaração `public class function EscrevaTurboError(DatF : DataFile;Const NR : Longint;Error:SmallWord):Boolean;`

TAIOCheck

Declaração `public class function TAIOCheck(VAR DatF : DataFile;Const R : LONGINT):Boolean;`

SincronizaPosChave

Declaração `public class function SincronizaPosChave(Var datFIx:IndexFile;Const NrCurrent:Longint; KeyCurrent : TaKeyStr):Boolean;`

GetRec

Declaração `public class function GetRec(var DatF : DataFile;Const R : Longint;var Buffer ):Boolean; overload;`

GetRecBlock

Declaração `public class function GetRecBlock(VAR DatF : DataFile; Const R : LONGINT; delta:Word;Var BlocksRead:Word ;VAR Buffer ):Boolean;`

PutRec

Declaração `public class function PutRec(var DatF : DataFile;Const R : Longint;var Buffer;Const Transaction_Current : T_TTransaction):Boolean; overload;`

PutRec

Declaração `public class function PutRec(var DatF : DataFile;Const R : Longint;var Buffer ):Boolean; overload;`

MakeFile

Declaração `public class function MakeFile(var DatF : DataFile;Const FName : FileName;Const RecLen :  SmallWord):Integer; overload;`

FDelStrBrancos

Declaração `public class function FDelStrBrancos(S:tString):tString;`

FileNameTemp_Ext

Declaração `public class function FileNameTemp_Ext(const aPath:PathStr;Var NomeArqTemp :  PathStr;const Extencao :  PathStr):Boolean; overload;`

FileNameTemp_Ext

Declaração `public class function FileNameTemp_Ext(Var NomeArqTemp : PathStr;const Extencao :  PathStr):Boolean; Overload;`

FileNameTemp

Declaração `public class function FileNameTemp(const Extencao : PathStr):PathStr;`

FileName_Seq

Declaração `public class function FileName_Seq(Const aName:PathStr;Const aExt : PathStr):PathStr;`

IsPortLocal

Declaração `public class function IsPortLocal(WPort: TTb_Access.tString):Boolean;`

DelFile

Declaração `public class function DelFile( Const Nome : NameStr):Boolean;`

SetOkAddRecFirstFree

Declaração `public class function SetOkAddRecFirstFree(Const aOkAddRecFirstFree: Boolean):Boolean;`

TestaSePodeAbrirArquivo

Declaração `public class function TestaSePodeAbrirArquivo(Const FName : PathStr): Byte;`

FileShared

Declaração `public class function FileShared(Const FName : PathStr) : Boolean;`

FTrocaExtencao

Declaração `public class function FTrocaExtencao(Const NomeArq:NameStr; Extencao:PathStr) : PathStr;`

Ren

Declaração `public class function Ren(NomeFonte,NomeDestino: PathStr) : Boolean;`

OkRecSizeMismatch

Declaração `public class function OkRecSizeMismatch(Const FName : FileName;Const RecLenBufferRecord : SmallWord):Boolean;`

Descrição A class método `OkRecSizeMismatch` retorna true se o Tamanho do registro em arquivo é maior que o tamanho do buffer do registro em Memória.

ModifyStructurFile

Declaração `public class function ModifyStructurFile(Const FName:FileName;Const RecLenDest : SmallWord ):Boolean; virtual; abstract;`

OpenFile

Declaração `public class function OpenFile(var DatF:DataFile; Const FName : FileName; Const RecLen:SmallWord):Integer;`

ReadHeader

Declaração `public class function ReadHeader(VAR DatF : DataFile):Boolean;`

PutFileHeader

Declaração `public class function PutFileHeader(VAR DatF : DataFile):Boolean;`

NaoMuDOuHeader

Declaração `public class function NaoMuDOuHeader(VAR DatF : DataFile) : BOOLEAN;`

MudouHeaderEmMemoria

Declaração `public class function MudouHeaderEmMemoria(VAR DatF : DataFile) : BOOLEAN;`

aCloseFile

Declaração `public class function aCloseFile(VAR DatF : DataFile):boolean;`

CloseFile

Declaração `public class function CloseFile(VAR DatF : DataFile):boolean; Overload;`

CloseFile

Declaração `public class function CloseFile(VAR DatF : DataFile;OkCondicional:Boolean):boolean; Overload;`

FlushFile

Declaração `public class function FlushFile(VAR DatF : DataFile):Boolean; overload;`

TraveRegistro

Declaração `public class function TraveRegistro(VAR DatF : DataFile; Const R : LONGINT):Boolean;`

DestraveRegistro

Declaração `public class function DestraveRegistro(Var DatF : DataFile;Const R : Longint):Boolean;`

TraveHeader

Declaração `public class function TraveHeader(VAR DatF : DataFile):Boolean;`

DestraveHeader

Declaração `public class function DestraveHeader(VAR DatF : DataFile):Boolean;`

IoResult

Declaração `public class function IoResult(Var DatF : DataFile) :  Integer; overload;`

FileSize

Declaração `public class function FileSize(VAR DatF : DataFile):Longint; Overload;`

NewRec

Declaração `public class Procedure NewRec(VAR DatF : DataFile;VAR R : LONGINT );`

AddRec

Declaração `public class function AddRec(var DatF : DataFile;var R : Longint;var Buffer ):Boolean; overload;`

DeleteRecord

Declaração `public class function DeleteRecord(VAR DatF : DataFile;Const R : LONGINT; Var Buffer ):Boolean; overload;`

DeleteRec

Declaração `public class function DeleteRec(var DatF : DataFile;Const R : Longint):Boolean; overload;`

FileLen

Declaração `public class function FileLen(VAR DatF : DataFile) : LONGINT; overload;`

UsedRecs

Declaração `public class function UsedRecs(VAR DatF : DataFile) : LONGINT; Overload;`

UsedRecs

Declaração `public class function UsedRecs(VAR DatF : DataFile;OK_GetHeader : Boolean) : LONGINT; Overload;`

UsedRecs

Declaração `public class function UsedRecs(Var IxF :IndexFile;OK_GetHeaderDoIndice : Boolean) : LONGINT; Overload;`

UsedRecs

Declaração `public class function UsedRecs(Var IxF :IndexFile) : LONGINT; Overload;`

UsedRecs

Declaração `public class function UsedRecs(Const FileName:PathStr) : Longint; Overload;`

TaPack

Declaração `public class Procedure TaPack(VAR Page : TaPage;Const KeyL : BYTE);`

TaUnpack

Declaração `public class Procedure TaUnpack(VAR Page : TaPage; Const KeyL : BYTE);`

Multiplo_Mais_proximo_de_N

Declaração `public class function Multiplo_Mais_proximo_de_N(Const K,N:Longint): Longint;`

MakeIndex

Declaração `public class function MakeIndex(var IxF : IndexFile;Const FName : FileName;Const KeyLen,S : byte):Integer; overload;`

OpenIndex

Declaração `public class function OpenIndex(var IxF : IndexFile;Const FName : FileName;Const KeyLen,S : byte):Integer; overload;`

LeiaHeaderDoIndice

Declaração `public class Procedure LeiaHeaderDoIndice( VAR IxF : IndexFile);`

aCloseIndex

Declaração `public class function aCloseIndex(VAR IxF : IndexFile):Boolean; overload;`

CloseIndex

Declaração `public class function CloseIndex(VAR IxF : IndexFile):boolean; Overload;`

CloseIndex

Declaração `public class function CloseIndex(VAR IxF : IndexFile;OkCondicional:Boolean):boolean; Overload;`

FlushIndex

Declaração `public class function FlushIndex(VAR IxF : IndexFile):boolean; overload;`

EraseFile

Declaração `public class function EraseFile(VAR DatF : DataFile):boolean; overload;`

EraseIndex

Declaração `public class function EraseIndex(VAR IxF : IndexFile):boolean; overload;`

TaGetPage

Declaração `public class function TaGetPage(VAR IxF : IndexFile;Const R : LONGINT;VAR PgPtr : TaPagePtr):boolean;`

TaNewPage

Declaração `public class Procedure TaNewPage(VAR IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr);`

TaDeletePage

Declaração `public class Procedure TaDeletePage(var IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr);`

ClearKey

Declaração `public class Procedure ClearKey(VAR IxF : IndexFile); overload;`

NextKey

Declaração `public class function NextKey(VAR IxF : IndexFile; VAR DataRecNum : LONGINT; VAR ProcKey ):Boolean; overload;`

PrevKey

Declaração `public class function PrevKey(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey ):Boolean; overload;`

TaXKey

Declaração `public class Procedure TaXKey(VAR K:TaKeyStr; Const KeyL : BYTE);`

TaCompKeys

Declaração `public class function TaCompKeys(Const K1 ,K2; DR1,DR2 : LONGINT; Const Dup : BOOLEAN ) : Shortint;`

TaFindKey

Declaração `public class function TaFindKey(VAR IxF : IndexFile;VAR DataRecNum : LONGINT;VAR ProcKey ):boolean;`

FindKey

Declaração `public class function FindKey(var IxF : IndexFile;var DataRecNum : Longint;var ProcKey ):Boolean; overload;`

FindKeyTop

Declaração `public class function FindKeyTop(var IxF : IndexFile;var DataRecNum : Longint;var ProcKey ):Boolean; overload;`

SearchKey

Declaração `public class function SearchKey(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey:TaKeyStr):Boolean; overload;`

SearchKeyTop

Declaração `public class function SearchKeyTop(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey:TaKeyStr;Const Okequal : Boolean):Boolean; overload;`

TaUpdatePage

Declaração `public class Procedure TaUpdatePage(VAR IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr; Const Transaction_Current : T_TTransaction); overload;`

AddKey_Search_Insert

Declaração `public class Procedure AddKey_Search_Insert(var IxF : IndexFile; Var PrPgRef1 : LONGINT; VAR PrPgRef2,c : LONGINT; VAR PagePtr1,PagePtr2 : TaPagePtr; VAR ProcItem1, ProcItem2 : TaItem; vAR PassUp, okAddKey : BOOLEAN; Const ProcKey : TaKeyStr; Const DataRecNum : Longint; VAR K,L : SmallInt; Var R : SmallInt);`

AddKey_Search_Init_ProcItem1

Declaração `public class Procedure AddKey_Search_Init_ProcItem1(Const ProcKey : TaKeyStr; Const DataRecNum : Longint; vAR PassUp : BOOLEAN; VAR ProcItem1 : TaItem);`

AddKey_Search

Declaração `public class Procedure AddKey_Search(var IxF : IndexFile; PrPgRef1 : LONGINT; VAR PrPgRef2,c : LONGINT; VAR PagePtr1,PagePtr2 : TaPagePtr; VAR ProcItem1, ProcItem2 : TaItem; vAR PassUp, okAddKey : BOOLEAN; Const ProcKey : TaKeyStr; Const DataRecNum : Longint; VAR K,L : SmallInt );`

AddKey

Declaração `public class function AddKey(var IxF : IndexFile; Const DataRecNum : Longint; Const ProcKey : TaKeyStr):Boolean; overload;`

DeleteKey

Declaração `public class function DeleteKey(var IxF : IndexFile;Const DataRecNum : Longint;var ProcKey:TaKeyStr ):Boolean; OVERLOAD;`

FGetHeaderDataFile

Declaração `public class function FGetHeaderDataFile(Const FileName: PathStr;Var Header : TsImagemHeader;Var aFileSize : Longint):Boolean;`

FTamRegDataFile

Declaração `public class function FTamRegDataFile(Const FileName: PathStr):SmallWord;`

NewFileName

Declaração `public class function NewFileName(FileName,Extencao:PathStr):PathStr;`

FTb

Declaração `public class function FTb(Const FileName:PathStr):PathStr;`

FObj

Declaração `public class function FObj(Const FileName:PathStr):PathStr;`

FIx

Declaração `public class function FIx(Const FileName:PathStr):PathStr;`

FDup

Declaração `public class function FDup(Const FileName:PathStr):PathStr;`

AssignDataFile

Declaração `public class Procedure AssignDataFile(Var DatF :DataFile; Const aFileName:PathStr; aBaseSize, aRecSize:SmallWord; aF :TStream; WTipo : AnsiChar ); Overload;`

AssignDataFile

Declaração `public class Procedure AssignDataFile(Var DatF :DataFile; Const aFileName:PathStr; aBaseSize, aRecSize:SmallWord); Overload;`

AssignIndexFile

Declaração `public class Procedure AssignIndexFile(Var IxF : IndexFile; Const aFileName :  PathStr; aBaseSize, aRecSize :  SmallWord); Overload;`

UpperCase

Declaração `public class function UpperCase(str:AnsiString):AnsiString;`

FMinuscula

Declaração `public class function FMinuscula(str:AnsiString):AnsiString;`

Int2str

Declaração `public class function Int2str(Const L : LongInt) : TTb_Access.tString;`

Beep

Declaração `public class procedure Beep;`

spc

Declaração `public class function spc(Const campo:AnsiString;Const tam :Longint):AnsiString;`

SetOkTransaction

Declaração `public class function SetOkTransaction(Const aOkTransaction : BOOLEAN):BOOLEAN;`

StartTransaction

Declaração `public class function StartTransaction(Const aDelta : SmallWord):Integer; Overload;`

StartTransaction

Declaração `public class function StartTransaction(Const DatF : DataFile ; Var aok_Set_Transaction : Boolean): Integer; Overload;`

COMMIT

Declaração `public class function COMMIT:Boolean; Overload;`

COMMIT

Declaração `public class function COMMIT(Const Wok_Set_Transaction : Boolean):Boolean; Overload;`

Rollback

Declaração `public class Procedure Rollback;`

SetTransaction

Declaração `public class function SetTransaction(const OnOff:Boolean; Var WOK : Boolean ):Boolean; Overload;`

SetTransaction

Declaração `public class function SetTransaction(const OnOff:Boolean; Var WOK, Wok_Set_Transaction:Boolean):Boolean; Overload;`

GetFileName_Transaction

Declaração `public class function GetFileName_Transaction(): tString;`

Assign_Transaction

Declaração `public class function Assign_Transaction(Const aFileName : PathStr):SmallWord;`

TransactionPendant_Error

Declaração `public class function TransactionPendant_Error:Boolean;`

TransactionPendant

Declaração `public class function TransactionPendant:Boolean;`

Truncate

Declaração `public class Procedure Truncate(Var DatF: DataFile;NR : LongInt);`

CopyFrom

Declaração `public class Procedure CopyFrom(Font_DatF: DataFile ;Var Dest_DatF: DataFile); Overload;`

CopyFrom

Declaração `public class Procedure CopyFrom(Font_IxF : IndexFile ;Var Dest_IxF : IndexFile); Overload;`

TFilesOpens Classe

Hierarquia

TFilesOpens > `TSortedCollection`(??) > `TCollection`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

- `FilesOpens`(??) é uma coleção que mantém todos os arquivos abertos `at`(??)é o momento com objetivo de fecha-los nos casos exceção.

Campos

okFlushAllFiles

Declaração `public okFlushAllFiles:  Boolean;`

Métodos

Create

Declaração `public constructor Create;`

destroy

Declaração `public destructor destroy; override;`

Compare

Declaração `public function Compare(Key1, Key2: Pointer): Sw_Integer; Override;`

Set_OkFlushAllFiles

Declaração `public Function Set_OkFlushAllFiles(wOkFlushAllFiles:boolean):Boolean;`

ListaTabelas

Declaração `public Procedure ListaTabelas;`

Insert

Declaração `public procedure Insert(Item: Pointer); Override;`

FreeItem

Declaração `public procedure FreeItem(Item: Pointer); Override;`

FOkCodigo

Declaração `public Function FOkCodigo(NomeIxF:PathStr;Const Codigo:tString):Boolean;`

FlushIndexs

Declaração `public Procedure FlushIndexs;`

FlushAllFiles

Declaração `public Procedure FlushAllFiles;`

OpenAllFiles

Declaração `public Function OpenAllFiles:Boolean;`

CloseAllFiles

Declaração `public Function CloseAllFiles:Boolean;`

EstatisticasDosArquivosAbertos

Declaração `public Procedure EstatisticasDosArquivosAbertos;`

SaveCurrentState

Declaração `public Procedure SaveCurrentState;`

RestoreCurrentState

Declaração `public Procedure RestoreCurrentState;`

MaxTamReg

Declaração `public Function MaxTamReg:SmallWord;`

24.5 Variáveis

FilesOpens

Declaração `FilesOpens:  TFilesOpens = nil;`

Chapter 25

Unit mi.rtl.Objects.Methods.Db.Tb__Access

25.1 Descrição

Esta unit `mi.rtl.Objects.Methods.Db.Tb__Access` é usada para criar banco de dados local usando estrutura **Type Record End**;

25.2 Uses

- `Classes`
- `SysUtils`
- `dos`
- `mi.rtl.Objects.Methods.Paramexecucao.Application`(??)
- `mi.rtl.objects.Methods.Exception`(??)
- `mi.rtl.objects.Methods.dates`(??)
- `mi.rtl.objects.methods.ParamExecucao`(??)
- `mi.rtl.objects.methods.db.tb_access`(??)

25.3 Visão Geral

`TTb__Access_types Classe`

`TTb__Access_consts Classe`

`TTb__Access Classe`

25.4 Classes, Interfaces, Objetos e Registros

TTb__Access_types Classe

Hierarquia

TTb__Access_types > `TTb_Access`(**??**) > `TTb_Access_consts`(**??**) > `TTb_Access_types`(**??**) > `TObjectsSystem`(**??**) > `TObjectsMethods`(**??**) > `TObjectsConsts`(**??**) > TObjectsTypes

Descrição

no description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`(**??**)

TTb__Access_consts Classe

Hierarquia

TTb__Access_consts > `TTb__Access_types`(**??**) > `TTb_Access`(**??**) > `TTb_Access_consts`(**??**) > `TTb_Access_types`(**??**) > `TObjectsSystem`(**??**) > `TObjectsMethods`(**??**) > `TObjectsConsts`(**??**) > TObjectsTypes

Descrição

no description available, TTb__Access_types description followsno description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`(**??**)

Campos

EndClearAll

Declaração `public const EndClearAll :  TipoProc = Nil;`

EndOpenFiles

Declaração `public const EndOpenFiles :  TipoProc = Nil;`

EndCloseFiles

Declaração `public const EndCloseFiles :  TipoProc = Nil;`

OkDeveReindexarDB

Declaração `public const OkDeveReindexarDB : Boolean = false;`

Descrição True = O sistema deve reindexar todos os arquivos

OkDeveRepararDB

Declaração `public const OkDeveRepararDB : Boolean = false;`

Descrição True = O Sistema deve executar a rotina para reparar as consistências entre tabelas

NRecAux

Declaração `public const NRecAux :  Longint = 0;`

Descrição A constante `NRecAux` é o número do registro corrente temporário.

- **NOTA**
  - É usado para manter a compatibilidade com o passado.

NRec

Declaração `public const NRec :  Longint = 0;`

Descrição A constante `NRec` é o número do registro corrente

- **NOTA**
  - É usado para manter a compatibilidade com o passado.

WCursorLigado

Declaração `public const WCursorLigado :  Boolean = true;`

WEndCloseFiles

Declaração `public const WEndCloseFiles:  TipoProc = nil;`

WEndOpenFiles

Declaração `public const WEndOpenFiles :  TipoProc = nil;`

TTb__Access Classe

Hierarquia

TTb__Access > `TTb__Access_consts`(??) > `TTb__Access_types`(??) > `TTb_Access`(??) > `TTb_Access_consts`(??) > `TTb_Access_types`(??) > `TObjectsSystem`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

no description available, TTb__Access_consts description followsno description available, TTb__Access_types description followsno description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`**(??)**

Métodos

StartTransaction

Declaração `public class function StartTransaction(Const DatF : TMI_DataFile ;`
`Var aok_Set_Transaction :  Boolean):  Integer; Overload;`

FileSize

Declaração `public class function FileSize(Var MI_DataFile :`
`TMI_DataFile):Longint; Overload;`

Init_MI_DataFile

Declaração `public class Procedure Init_MI_DataFile(Var MI_DataFile :`
`TMI_DataFile; NomeArquivo :  PathStr; tamanhoRegistro :  SmallWord;`
`NumeroDeArqIndice :  byte ); Overload;`

Init_MI_DataFile

Declaração `public class Procedure Init_MI_DataFile(Var MI_DataFile : TMI_DataFile; NomeArquivo : PathStr; tamanhoRegistro : TTb_Access.SmallWord; NumeroDeArqIndice : byte; wOkTemporario : Boolean); Overload;`

Init_IxF

Declaração `public class Procedure Init_IxF(Const Indice : Byte; Var IxF : TMI_IndexFile; Const CNomeArqIndice : PathStr; Const CRepeteChave : Byte; Const StrCondicao : tString );`

MakeFile

Declaração `public class function MakeFile(Const FileName:PathStr;Const TamArq:Longint):Integer; overload;`

MakeFile

Declaração `public class function MakeFile(var DatF : TMI_DataFile ):Integer; overload;`

MakeIndex

Declaração `public class function MakeIndex(Const FileName:PathStr;Const RepeteChave,TamChave:Byte):Integer; overload;`

MakeIndex

Declaração `public class function MakeIndex(var IxF : TMI_IndexFile):Integer; Overload;`

OpenFile

Declaração `public class function OpenFile(var DatF : TMI_DataFile;OkCreate : Boolean):Integer; Overload;`

OpenFile

Declaração `public class function OpenFile(var DatF : TMI_DataFile ):Integer; Overload;`

OpenIndex

Declaração `public class function OpenIndex(var IxF : TMI_IndexFile):Integer; Overload;`

CloseFile

Declaração `public class function CloseFile(var DatF : TMI_DataFile ):Integer; overload;`

CloseIndex

Declaração `public class function CloseIndex(var IxF : TMI_IndexFile):Boolean; overload;`

MakeArq

Declaração `public class Procedure MakeArq(VAR DatF : TMI_DataFile; VAR Buff );`

Descrição A class procedure `MakeArq` é usado criar aquivo sem o registro 0

OpenArq

Declaração `public class Procedure OpenArq(VAR DatF : TMI_DataFile; VAR Buff );`

Descrição A class procedure `OpenArq` é usado abrir aquivo sem o registro 0

AbreArqSemHeader

Declaração `public class Procedure AbreArqSemHeader(VAR Arqdados:TMI_DataFile ; VAR Buff );`

CloseArqSemHeader

Declaração `public class Procedure CloseArqSemHeader(VAR DatF : TMI_DataFile);`

GetAddRec

Declaração `public class function GetAddRec(Const Title : tString; Const NomeFonte:PathStr; Var RegFonte; Const TamFonte : SmallWord; Const NomeDestino : PathStr; Var regDestino; Const TamDestino : SmallWord; Const AtualizaDestino : TFuncGetAddRec; Const OkMakeFile :Boolean) : Boolean;`

ModifyStructurFile

Declaração `public class function ModifyStructurFile(Const FName:FileName;Const RecLenDest : SmallWord ):Boolean; override;`

PrimeiroLivre

Declaração `public class function PrimeiroLivre(VAR DatF: TMI_DataFile) : LONGINT;`

TraveArq

Declaração `public class function TraveArq(Var DatF : TMI_DataFile):Boolean;`

DestraveArq

Declaração `public class function DestraveArq(Var DatF : TMI_DataFile):Boolean;`

UsedRecs

Declaração `public class function UsedRecs(var DatF : TMI_DataFile ) : Longint; Overload;`

GetRec

Declaração `public class function GetRec(var DatF : TMI_DataFile ;Const R : Longint;var Buffer ):Boolean; overload;`

PutRec

Declaração `public class function PutRec(var DatF : TMI_DataFile ;Const R : Longint;var Buffer ):Boolean; overload;`

AddRec

Declaração `public class function AddRec(var DatF : TMI_DataFile ;var R : Longint;var Buffer ):Boolean; overload;`

DeleteRec

Declaração `public class function DeleteRec(var DatF : TMI_DataFile ;Const R : Longint):Boolean; overload;`

FileLen

Declaração `public class function FileLen(var DatF : TMI_DataFile ) :  Longint; overload;`

MakeIndex

Declaração `public class function MakeIndex(var IxF : TMI_IndexFile;Exclusivo:Boolean ):Integer; Overload;`

OpenIndex

Declaração `public class function OpenIndex(var IxF : TMI_IndexFile;Exclusivo :Boolean ):Integer; Overload;`

OpenIndex

Declaração `public class function OpenIndex(var IxF : TMI_IndexFile;Exclusivo,OkCreate:Boolean ):Integer; Overload;`

ClearKey

Declaração `public class function ClearKey(var IxF : TMI_IndexFile) :Boolean; overload;`

NextKey

Declaração `public class function NextKey(var IxF : TMI_IndexFile; var ProcDatRef : Longint; var ProcKey ):Boolean; overload;`

PrevKey

Declaração `public class function PrevKey(var IxF : TMI_IndexFile; var ProcDatRef : Longint; var ProcKey ):Boolean; overload;`

FindKeyTop

Declaração `public class function FindKeyTop(var IxF : TMI_IndexFile ; var ProcDatRef : Longint; var ProcKey ):Boolean; overload;`

FindKey

Declaração `public class function FindKey(var IxF : TMI_IndexFile; var ProcDatRef : Longint; var ProcKey ):Boolean; overload;`

SearchKey

Declaração `public class function SearchKey(var IxF : TMI_IndexFile; var ProcDatRef : Longint; var ProcKey:TaKeyStr ):Boolean; overload;`

SearchKeyTop

Declaração `public class function SearchKeyTop(var IxF : TMI_IndexFile ; var ProcDatRef : Longint; var ProcKey :TaKeyStr; Const Okequal : Boolean ):Boolean; overload;`

AddKey

Declaração `public class function AddKey(var IxF : TMI_IndexFile; Const ProcDatRef : Longint; Const ProcKey : TaKeyStr ):Boolean; overload;`

DeleteKey

Declaração `public class function DeleteKey(var IxF : TMI_IndexFile; Const ProcDatRef : Longint; Const ProcKey : TaKeyStr):Boolean; overload;`

FlushFile

Declaração `public class procedure FlushFile(var DatF :TMI_DataFile ); overload;`

FlushIndex

Declaração `public class procedure FlushIndex(var IxF : TMI_IndexFile ); overload;`

Seek

Declaração `public class function Seek(Var DatF : TMI_DataFile;Const R : Longint ):SmallInt; overload;`

CloseFilesOpens

Declaração `public class Procedure CloseFilesOpens; virtual;`

MyDestroyMemory

Declaração `public class Procedure MyDestroyMemory;`

MyCreateMemory

Declaração `public class Procedure MyCreateMemory;`

MyDestroyMemorySemVideo

Declaração `public class Procedure MyDestroyMemorySemVideo;`

MyCreateMemorySemVideo

Declaração `public class Procedure MyCreateMemorySemVideo;`

ExecCommand

Declaração `public class function ExecCommand(FileName:PathStr;Flags: Longint;aExecAsync : Longint): Byte; Overload;`

ExecCommand

Declaração `public class function ExecCommand(FileName:PathStr;Flags: Longint): Byte; Overload;`

ExecDos

Declaração `public class function ExecDos(Const Path: PathStr; Const ComLine: ComStr): Byte;`

Descrição A classe método `ExecDos` executa um programa externo de form assíncrona.

- **EXEMPLO**

```
ExecDos('/usr/bin/gnome-terminal','ls');
```

FindKey

Declaração `public class function FindKey(var IxF : TMI_IndexFile; var ProcDatRef : Longint; ProcKey : TaKeyStr):Boolean ; overload;`

AdicioneChave

Declaração `public class function AdicioneChave(var IxF : TMI_IndexFile ; Const ProcDatRef : Longint; Const ProcKey : TaKeyStr):Boolean;`

EliminaChave

Declaração `public class function EliminaChave(var IxF : TMI_IndexFile ; Const ProcDatRef : Longint; Const ProcKey : TaKeyStr):Boolean;`

NomeDaEstacao

Declaração `public class function NomeDaEstacao:tString;`

ValidFileName

Declaração `public class function ValidFileName(Const Name : PathStr):Byte;`

FConcatNomeArq

Declaração `public class function FConcatNomeArq(Nome,Extencao:PathStr) : PathStr;`

CriaArqTemp

Declaração `public class function CriaArqTemp(Var ArqF : TMI_DataFile; Const TamArqTemp : SmallWord; Const NumeroDeIndice : Byte ):Boolean;`

CriaArqTempI

Declaração `public class function CriaArqTempI(Var IxF: TMI_IndexFile; Const RepeteChave : Byte; Const EndChaveNoRegistro: tString):Boolean;`

EspacoEmDisco

Declaração `public class function EspacoEmDisco(NomeFonte,DriveDestino:PathStr) :boolean;`

TTraveRegistro

Declaração `public class function TTraveRegistro(Var DatF : TMI_DataFile;Const R : Longint):Boolean;`

TDestraveRegistro

Declaração `public class function TDestraveRegistro(Var DatF : TMI_DataFile;Const R : Longint):Boolean;`

FPackDataFile

Declaração `public class function FPackDataFile(NomeArq :PathStr):Boolean;`

FLeiaGrave

Declaração `public class function FLeiaGrave(Const MsgStr : tString; Const NomeArqDados : PathStr; Var RegBuff ; Const TamRegBuff : SmallWord; Const OkFunc : TipoFuncao) : Boolean;`

LeiaGrave

Declaração `public class function LeiaGrave(Const MsgStr : tString; Var ArqDados : TMI_DataFile; Var RegBuff ; Const OkFunc : TipoFuncao) : Boolean;`

FLeieEGraveRegistro

Declaração `public class function FLeieEGraveRegistro(Const NomeFonte:PathStr; Var RegFonte; Const TamFonte : SmallWord; Const NomeDestino : PathStr; Var regDestino; Const TamDestino : SmallWord; AtualizaDestino : TipoFuncao; Const OkMakeFile :Boolean) : Boolean;`

StrDataEmQueFoiAlterado

Declaração `public class function StrDataEmQueFoiAlterado(Const NomeArquivo :PathStr) : tString;`

StrDateFile

Declaração `public class function StrDateFile(Const NomeArquivo : PathStr;Const Ch:AnsiChar) : tString;`

CreateLst

Declaração `public class function CreateLst:Boolean;`

DestroyLst

Declaração `public class Procedure DestroyLst;`

GeraSequencia

Declaração `public class function GeraSequencia(Var ArqI:IndexFile) :Longint;`

TestaAberturaDeArquivo

Declaração `public class function TestaAberturaDeArquivo(MaxFile : Byte; Var NumMaxPossivel:Byte ): Boolean;`

AssingLst

Declaração `public class function AssingLst(Const WopcaoRedireciona : AnsiChar; Const aNomeRedireciona : PathStr):Boolean;`

redirecionaParaNul

Declaração `public class procedure redirecionaParaNul;`

Create

Declaração `public class Procedure Create;`

Destroy

Declaração `public class Procedure Destroy;`

Chapter 26

Unit mi.rtl.objects.methods.db.tb__access_test

26.1 Uses

- Classes
- SysUtils
- mi.rtl.Consts(??)
- mi.rtl.objectss(??)

26.2 Visão Geral

TAluno Classe

26.3 Classes, Interfaces, Objetos e Registros

TAluno Classe

Hierarquia

TAluno > TObject

Descrição

A class TAluno desmonstra o uso da classe TObjectss.Ttb__access(??)

Campos

NR_Bof

Declaração `public const NR_Bof : Longint = 0 ;`

NR_Current

Declaração `public const NR_Current : Longint = 0 ;`

NR_Eof

Declaração `public const NR_Eof : Longint = 0 ;`

OkBof

Declaração `public const OkBof : boolean = true;`

OkEof

Declaração `public const OkEof : Boolean = false;`

RecordSelected

Declaração `public const RecordSelected : Boolean = false;`

ArqAluno

Declaração `public var ArqAluno: TObjectss.Ttb__access.TMI_DataFile; static;`

RegAluno

Declaração `public var RegAluno: TRegAluno; static;`

Métodos

Init_ArqAluno

Declaração `public class procedure Init_ArqAluno;`

Create_ArqAluno

Declaração `public class function Create_ArqAluno:Boolean;`

DoOnNewRecord

Declaração `public class procedure DoOnNewRecord(aNome:AnsiString);`

AddRec

Declaração `public class function AddRec(aNome:AnsiString):Boolean;`

NextRec

Declaração `public class function NextRec:Boolean;`

GoBof

Declaração `public class Function GoBof:  Boolean ;`

Cadastra

Declaração `public class function Cadastra:Boolean;`

Lista_ordem_crescente

Declaração `public class Procedure Lista_ordem_crescente;`

GoEof

Declaração `public class Function GoEof:  Boolean ;`

PrevRec

Declaração `public class function PrevRec:Boolean;`

Lista_ordem_Decrescente

Declaração `public class Procedure Lista_ordem_Decrescente;`

DeleteRec

Declaração `public class function DeleteRec:Boolean;`

Test_SetTransaction

Declaração `public class Function Test_SetTransaction:Boolean;`

Chapter 27

Unit mi.rtl.Objects.Methods.Db.Tb__Access

27.1 Descrição

Esta unit `mi.rtl.Objects.Methods.Db.Tb__Access` é usada para criar banco de dados local usando estrutura **Type Record End**;

- **NOTA**
  - Comandos parecidos com clipper
  - Está obsoleto não recomendo seu uso.

27.2 Uses

- `Classes`
- `SysUtils`
- `dos`
- `mi.rtl.Objects.Methods.Paramexecucao.Application`(??)
- `mi.rtl.objects.Methods.Exception`(??)
- `mi.rtl.objects.Methods.dates`(??)
- `mi.rtl.objects.methods.ParamExecucao`(??)
- `mi.rtl.objects.methods.db.tb_access`(??)
- `mi.rtl.objects.methods.db.tb__access`(??)

27.3 Visão Geral

`TTb___Access_types Classe`

`TTb___Access_consts Classe`

`TTb___Access Classe`

27.4 Classes, Interfaces, Objetos e Registros

TTb___Access_types Classe

Hierarquia

TTb___Access_types > `TTb__Access(??)` > `TTb__Access_consts(??)` > `TTb__Access_types(??)` > `TTb_Access(??)` > `TTb_Access_consts(??)` > `TTb_Access_types(??)` > `TObjectsSystem(??)` > `TObjectsMethods(??)` > `TObjectsConsts(??)` > TObjectsTypes

Descrição

no description available, TTb__Access description followsno description available, TTb__Access_consts description followsno description available, TTb__Access_types description followsno description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access(??)`

Campos

MaxBase

Declaração `public const MaxBase = 135;`

Descrição Uma Base permite `um(??)` arquivo de dados + `Um(??)` arquivo de indice

TTb___Access_consts Classe

Hierarquia

TTb___Access_consts > `TTb___Access_types(??)` > `TTb__Access(??)` > `TTb__Access_consts(??)` > `TTb__Access_types(??)` > `TTb_Access(??)` > `TTb_Access_consts(??)` > `TTb_Access_types(??)` > `TObjectsSystem(??)` > `TObjectsMethods(??)` > `TObjectsConsts(??)` > TObjectsTypes

Descrição

no description available, TTb___Access_types description followsno description available, TTb__Access description followsno description available, TTb__Access_consts description followsno description available, TTb__Access_types description followsno description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access(??)`

Campos

NBase

Declaração `public const NBase : Byte = 0;`

MatPFile

Declaração
```
public const MatPFile : TipoBaseDeDados =
(Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,
Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil,Nil );
```

TTb___Access Classe

Hierarquia

TTb___Access > TTb___Access_consts(??) > TTb___Access_types(??) > TTb__Access(??) > TTb__Access_consts(??) > TTb__Access_types(??) > TTb_Access(??) > TTb_Access_consts(??) > TTb_Access_types(??) > TObjectsSystem(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

no description available, TTb___Access_consts description followsno description available, TTb___Access_types description followsno description available, TTb__Access description followsno description available, TTb__Access_consts description followsno description available, TTb__Access_types description followsno description available, TTb_Access description followsno description available, TTb_Access_consts description followsA classe `TTb_Access_consts` é usada para declarar todas as constantes da classe `TTb_Access`(??)

Métodos

OpenFile

Declaração `public class function OpenFile(var DatF : TMI_DataFile ; var IxF : TMI_IndexFile; Const condicao : NumeroDeArquivos ) : Integer; overload;`

CloseFile

Declaração `public class procedure CloseFile(var DatF : TMI_DataFile ; var IxF : TMI_IndexFile ; Const condicao : NumeroDeArquivos ); overload;`

LocRegUse

Declaração `public class function LocRegUse(Const P : TipoPonteiroBD): Byte;`

NRecSkip

Declaração `public class function NRecSkip(Var Buff) : Longint;`

Replace

Declaração `public class Procedure Replace(Var Buff);`

Locate

Declaração `public class Procedure Locate(Var Buff;NRec:Longint);`

ReplaceUnLock

Declaração `public class Procedure ReplaceUnLock(Var Buff);`

TrocaChave

Declaração `public class Procedure TrocaChave(Var Buff; Var IxF : TMI_IndexFile; ChaveAnterior, ChaveAtual : TaKeyStr);`

FPosicao

Declaração `public class function FPosicao(Posicao : Byte ) : Byte;`

IndiceSele

Declaração `public class function IndiceSele(Var Buff) : PathStr;`

ClearSkipAll

Declaração `public class Procedure ClearSkipAll;`

Use

Declaração `public class function Use(Var DatF : TMI_DataFile ; Var IxF : TMI_IndexFile ; Var Buff ):Boolean;`

UseC

Declaração `public class Procedure UseC(Var Buff);`

Skip

Declaração `public class Procedure Skip(Var Buff; Var Chave : TaKeyStr; Const ModoDePesquisa : TipoSkip );`

FSkip

Declaração `public class function FSkip(Var Buff; Var Chave : tString; Const ModoDePesquisa : TipoSkip ):Boolean;`

FSkipSearch

Declaração `public class function FSkipSearch(Var Buff; Chave : tString):Boolean;`

SeekRec

Declaração `public class Procedure SeekRec(Var Buff; Var IxF : TMI_IndexFile ; Var Chave : TaKeyStr; Const ModoDePesquisa : TipoSkip );`

SkipLock

Declaração `public class Procedure SkipLock(Var Buff; Var Chave : tString; Const ModoDePesquisa : TipoSkip );`

LocRegSkip

Declaração `public class function LocRegSkip(P : TipoPonteiroBD): Byte;`

Sele

Declaração `public class Procedure Sele(Var Buff; Var IxF : TMI_IndexFile );`

AAddRec

Declaração `public class Procedure AAddRec(Var Buff;Var IxF : TMI_IndexFile ; Var Chave : tString); overload;`

ADeleteRec

Declaração `public class Procedure ADeleteRec(Var Buff;Var IxF : TMI_IndexFile ;Var Chave : tString); overload;`

CloseFilesOpens

Declaração `public class Procedure CloseFilesOpens; override;`

EscrevaDataFile

Declaração `public class procedure EscrevaDataFile(Var DatF : TMI_DataFile; Var IxF : TMI_IndexFile);`

Chapter 28

Unit mi.rtl.objects.Methods.db.types.consts.Meth

28.1 Descrição

- A unit `mi.rtl.objects.Methods.db.types.consts.Methods` implementa a classe `TDb_Methods(??)` do pacote **mi.rtl.db**.
  - **NOTAS** -
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **01/12/2021**
      - 10:15 a ?? : Criar a unit mi.rtl.db_Methods.pas
  - **CÓDIGO FONTE**:
    -

28.2 Uses

- `Classes`
- `SysUtils`

- `mi.rtl.objects.Methods.db.types.consts`
- `mi.rtl.objects.methods.StreamBase.Stream`(??)
- `mi.rtl.objects.TException`

28.3 Visão Geral

`TDb_Methods Classe`

28.4 Classes, Interfaces, Objetos e Registros

TDb_Methods Classe

Hierarquia

TDb_Methods > TDbConsts

Descrição

- A classe `TDb_Methods` implementa os método de classe comum a todas as classes de TDB do pacote **mi.rtl.db**.

Métodos

Create

Declaração `public constructor Create(aowner:TComponent); Overload; Override;`

Destroy

Declaração `public destructor Destroy; override;`

FExisteCodigo

Declaração `public function FExisteCodigo(Var IxF:IndexFile; Const Codigo:tString):Boolean;`

CreateTAccess

Declaração `public procedure CreateTAccess;`

DestroyTAccess

Declaração `public procedure DestroyTAccess;`

EscrevaTurboError

Declaração `public function EscrevaTurboError(DatF : DataFile;Const NR : Longint;Error:SmallWord):Boolean;`

TAIOCheck

Declaração `public function TAIOCheck(VAR DatF : DataFile;Const R : LONGINT):Boolean;`

SincronizaPosChave

Declaração `public function SincronizaPosChave(Var datFIx:IndexFile;Const NrCurrent:Longint; KeyCurrent : TaKeyStr):Boolean;`

GetRec

Declaração `public function GetRec(var DatF : DataFile;Const R : Longint;var Buffer ):Boolean;`

GetRecBlock

Declaração `public function GetRecBlock(VAR DatF : DataFile; Const R : LONGINT; delta:Word;Var BlocksRead:Word ;VAR Buffer ):Boolean;`

_PutRec

Declaração `public function _PutRec(var DatF : DataFile;Const R : Longint;var Buffer;Const Transaction_Current : T_TTransaction):Boolean;`

PutRec

Declaração `public function PutRec(var DatF : DataFile;Const R : Longint;var Buffer ):Boolean;`

MakeFile

Declaração `public function MakeFile(var DatF : DataFile;Const FName : TFileName;Const RecLen :  SmallWord):Integer;`

FMakeFile

Declaração `public function FMakeFile(Const FileName:PathStr;Const TamArq:Longint):Integer;`

AtualizaDestino

Declaração `public function AtualizaDestino(Var RegFonte; Const TamFonte:SmallWord; var RegDestino; Const TamDestino :  SmallWord) : BOOLEAN;`

FDelStrBrancos

Declaração `public function FDelStrBrancos(S:tString):tString;`

FileNameTemp_Ext

Declaração `public function FileNameTemp_Ext(const aPath:PathStr;Var NomeArqTemp :  PathStr;Extencao :  PathStr):Boolean; overload;`

FileNameTemp_Ext

Declaração `public function FileNameTemp_Ext(Var NomeArqTemp : PathStr;Extencao :  PathStr):Boolean; Overload;`

FileNameTemp

Declaração `public function FileNameTemp(Extencao : PathStr):PathStr;`

FileName_Seq

Declaração `public function FileName_Seq(Const aName:PathStr;Const aExt : PathStr):PathStr;`

IsPortLocal

Declaração `public function IsPortLocal(WPort: tString):Boolean;`

DelFile

Declaração `public function DelFile( Const Nome : TFileName):Boolean;`

SetOkAddRecFirstFree

Declaração `public function SetOkAddRecFirstFree(Const aOkAddRecFirstFree: Boolean):Boolean;`

TestaSePodeAbrirArquivo

Declaração `public function TestaSePodeAbrirArquivo(Const FName : PathStr): Byte;`

FileShared

Declaração `public function FileShared(Const FName : PathStr) : Boolean;`

FTrocaExtencao

Declaração `public function FTrocaExtencao(Const NomeArq:TFileName; Extencao:PathStr) : PathStr;`

Ren

Declaração `public function Ren(NomeFonte,NomeDestino: PathStr) : Boolean;`

OkRecSizeMismatch

Declaração `public function OkRecSizeMismatch(Const FName : TFileName;Const RecLenBufferRecord : SmallWord):Boolean;`

ModifyStructurFile

Declaração `public function ModifyStructurFile(Const FName:TFileName;Const RecLenDest : SmallWord ):Boolean;`

OpenFile

Declaração `public function OpenFile(var DatF:DataFile; Const FName : TFileName; Const RecLen:SmallWord; AFileMode:Word ):Integer;`

ReadHeader

Declaração `public function ReadHeader(VAR DatF : DataFile):Boolean;`

PutFileHeader

Declaração `public function PutFileHeader(VAR DatF : DataFile):Boolean;`

NaoMuDOuHeader

Declaração `public function NaoMuDOuHeader(VAR DatF : DataFile) :  BOOLEAN;`

MudouHeaderEmMemoria

Declaração `public function MudouHeaderEmMemoria(VAR DatF : DataFile) : BOOLEAN;`

aCloseFile

Declaração `public function aCloseFile(VAR DatF : DataFile):boolean;`

CloseFile

Declaração `public function CloseFile(VAR DatF : DataFile):boolean; Overload;`

CloseFile

Declaração `public function CloseFile(VAR DatF : DataFile;OkCondicional:Boolean):boolean; Overload;`

FlushFile

Declaração `public function FlushFile(VAR DatF : DataFile):Boolean;`

TraveRegistro

Declaração `public function TraveRegistro(VAR DatF : DataFile; Const R : LONGINT):Boolean;`

DestraveRegistro

Declaração `public function DestraveRegistro(Var DatF : DataFile;Const R : Longint):Boolean;`

TraveHeader

Declaração `public function TraveHeader(VAR DatF : DataFile):Boolean;`

DestraveHeader

Declaração `public function DestraveHeader(VAR DatF : DataFile):Boolean;`

FileSize

Declaração `public function FileSize(VAR DatF : DataFile):Longint; Overload;`

NewRec

Declaração `public procedure NewRec(VAR DatF : DataFile;VAR R : LONGINT );`

AddRec

Declaração `public function AddRec(var DatF : DataFile;var R : Longint;var Buffer ):Boolean;`

DeleteRecord

Declaração `public function DeleteRecord(VAR DatF : DataFile; Const R : LONGINT; Var Buffer ):Boolean;`

DeleteRec

Declaração `public function DeleteRec(var DatF : DataFile;Const R: Longint):Boolean;`

FileLen

Declaração `public function FileLen(VAR DatF : DataFile) :  LONGINT;`

UsedRecs

Declaração `public function UsedRecs(VAR DatF : DataFile) :  LONGINT; Overload;`

UsedRecs

Declaração `public function UsedRecs(VAR DatF : DataFile;OK_GetHeader : Boolean) :  LONGINT; Overload;`

UsedRecs

Declaração `public function UsedRecs(Var IxF :IndexFile;OK_GetHeaderDoIndice : Boolean) :  LONGINT; Overload;`

UsedRecs

Declaração `public function UsedRecs(Var IxF :IndexFile) :  LONGINT; Overload;`

UsedRecs

Declaração `public function UsedRecs(Const FileName:PathStr) :  Longint; Overload;`

TaPack

Declaração `public procedure TaPack(VAR Page :  TaPage;Const KeyL : BYTE);`

TaUnpack

Declaração `public procedure TaUnpack(VAR Page :  TaPage; Const KeyL : BYTE);`

Multiplo_Mais_proximo_de_N

Declaração `public function Multiplo_Mais_proximo_de_N(Const K,N:Longint): Longint;`

MakeIndex

Declaração `public function MakeIndex(var IxF : IndexFile;Const FName : TFileName;Const KeyLen,S : byte):Integer;`

FMakeIndex

Declaração `public function FMakeIndex(Const FileName:PathStr;Const RepeteChave,TamChave:Byte):Integer;`

LeiaHeaderDoIndice

Declaração `public procedure LeiaHeaderDoIndice( VAR IxF : IndexFile);`

aCloseIndex

Declaração `public function aCloseIndex(VAR IxF : IndexFile):Boolean;`

CloseIndex

Declaração `public function CloseIndex(VAR IxF : IndexFile;OkCondicional:Boolean):boolean; Overload;`

CloseIndex

Declaração `public function CloseIndex(VAR IxF : IndexFile):boolean; Overload;`

FlushIndex

Declaração `public function FlushIndex(VAR IxF : IndexFile):boolean;`

EraseFile

Declaração `public function EraseFile(VAR DatF : DataFile):boolean;`

EraseIndex

Declaração `public function EraseIndex(VAR IxF : IndexFile):boolean;`

TaGetPage

Declaração `public function TaGetPage(VAR IxF : IndexFile;Const R : LONGINT;VAR PgPtr : TaPagePtr):boolean;`

TaNewPage

Declaração `public procedure TaNewPage(VAR IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr);`

TaDeletePage

Declaração `public procedure TaDeletePage(var IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr);`

ClearKey

Declaração `public procedure ClearKey(VAR IxF : IndexFile);`

NextKey

Declaração `public function NextKey(VAR IxF : IndexFile; VAR DataRecNum : LONGINT; VAR ProcKey ):Boolean;`

PrevKey

Declaração `public function PrevKey(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey ):Boolean;`

TaXKey

Declaração `public procedure TaXKey(VAR K:TaKeyStr; Const KeyL : BYTE);`

TaCompKeys

Declaração `public function TaCompKeys(Const K1 ,K2; DR1,DR2 : LONGINT; Const Dup : BOOLEAN ) : Shortint;`

TaFindKey

Declaração `public function TaFindKey(VAR IxF : IndexFile;VAR DataRecNum : LONGINT;VAR ProcKey ):boolean;`

FindKey

Declaração `public function FindKey(var IxF : IndexFile;var DataRecNum : Longint;var ProcKey ):Boolean;`

FindKeyTop

Declaração `public function FindKeyTop(var IxF : IndexFile;var DataRecNum : Longint;var ProcKey ):Boolean;`

SearchKey

Declaração `public function SearchKey(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey:TaKeyStr):Boolean;`

SearchKeyTop

Declaração `public function SearchKeyTop(var IxF : IndexFile; var DataRecNum : Longint; var ProcKey:TaKeyStr;Const Okequal : Boolean):Boolean;`

TaUpdatePage

Declaração `public procedure TaUpdatePage(VAR IxF : IndexFile; VAR R : LONGINT; VAR PgPtr : TaPagePtr; Const Transaction_Current : T_TTransaction);`

AddKey_Search_Insert

Declaração `public procedure AddKey_Search_Insert(var IxF : IndexFile; Var PrPgRef1 : LONGINT; VAR PrPgRef2,c : LONGINT; VAR PagePtr1,PagePtr2 : TaPagePtr; VAR ProcItem1, ProcItem2 : TaItem; vAR PassUp, okAddKey : BOOLEAN; Const ProcKey : TaKeyStr; Const DataRecNum : Longint; VAR K,L : SmallInt; Var R : SmallInt);`

AddKey_Search_Init_ProcItem1

Declaração `public procedure AddKey_Search_Init_ProcItem1(Const ProcKey : TaKeyStr; Const DataRecNum : Longint; vAR PassUp : BOOLEAN; VAR ProcItem1 : TaItem);`

AddKey_Search

Declaração `public procedure AddKey_Search(var IxF : IndexFile; PrPgRef1 : LONGINT; VAR PrPgRef2,c : LONGINT; VAR PagePtr1,PagePtr2 : TaPagePtr; VAR ProcItem1, ProcItem2 : TaItem; vAR PassUp, okAddKey : BOOLEAN; Const ProcKey : TaKeyStr; Const DataRecNum : Longint; VAR K,L : SmallInt );`

AddKey

Declaração `public function AddKey(var IxF : IndexFile; Const DataRecNum : Longint; Const ProcKey : TaKeyStr):Boolean;`

DeleteKey

Declaração `public function DeleteKey(var IxF : IndexFile;Const DataRecNum : Longint;var ProcKey:TaKeyStr ):Boolean;`

FGetHeaderDataFile

Declaração `public function FGetHeaderDataFile(Const FileName: PathStr;Var Header : TsImagemHeader;Var aFileSize : Longint):Boolean;`

FTamRegDataFile

Declaração `public function FTamRegDataFile(Const FileName: PathStr):SmallWord;`

NewFileName

Declaração `public function NewFileName(FileName,Extencao:PathStr):PathStr;`

FTb

Declaração `public function FTb(Const FileName:PathStr):PathStr;`

FObj

Declaração `public function FObj(Const FileName:PathStr):PathStr;`

FIx

Declaração `public function FIx(Const FileName:PathStr):PathStr;`

FDup

Declaração `public function FDup(Const FileName:PathStr):PathStr;`

AssignDataFile

Declaração
```
public procedure AssignDataFile(Var DatF :DataFile; Const
 aFileName:PathStr; aBaseSize, aRecSize:SmallWord; Const AFileMode:  Word; aF
 :TStream; WTipo :  AnsiChar ); Overload;
```

AssignDataFile

Declaração
```
public procedure AssignDataFile(Var DatF :DataFile;Const
 aFileName:PathStr;aBaseSize,aRecSize:SmallWord); Overload;
```

AssignIndexFile

Declaração `public procedure AssignIndexFile(Var IxF : IndexFile; Const aFileName : PathStr; aBaseSize, aRecSize : SmallWord); Overload;`

UpperCase

Declaração `public function UpperCase(str:AnsiString):AnsiString;`

FMinuscula

Declaração `public function FMinuscula(str:AnsiString):AnsiString;`

Int2str

Declaração `public function Int2str(Const L : LongInt) : tString;`

spc

Declaração `public function spc(Const campo:AnsiString;Const tam :Longint):AnsiString;`

SetOkTransaction

Declaração `public function SetOkTransaction(Const aOkTransaction : BOOLEAN):BOOLEAN;`

StartTransaction

Declaração `public function StartTransaction(Const aDelta : SmallWord):Integer; Overload;`

StartTransaction

Declaração `public function StartTransaction(Const DatF : DataFile ; Var aok_Set_Transaction : Boolean): Integer; Overload;`

COMMIT

Declaração `public function COMMIT:Boolean; Overload;`

COMMIT

Declaração `public function COMMIT(Const Wok_Set_Transaction : Boolean):Boolean; Overload;`

Rollback

Declaração `public procedure Rollback;`

SetTransaction

Declaração `public function SetTransaction(const OnOff:Boolean; Var WOK : Boolean ):Boolean; Overload;`

SetTransaction

Declaração `public function SetTransaction(const OnOff:Boolean; Var WOK, Wok_Set_Transaction:Boolean):Boolean; Overload;`

GetFileName_Transaction

Declaração `public function GetFileName_Transaction(): tString;`

Assign_Transaction

Declaração `public function Assign_Transaction(Const aFileName : PathStr):SmallWord;`

TransactionPendant_Error

Declaração `public function TransactionPendant_Error:Boolean;`

TransactionPendant

Declaração `public function TransactionPendant:Boolean;`

Truncate

Declaração `public procedure Truncate(Var DatF: DataFile;NR : LongInt);`

CopyFrom

Declaração `public procedure CopyFrom(Font_DatF: DataFile ;Var Dest_DatF: DataFile); Overload;`

CopyFrom

Declaração `public procedure CopyFrom(Font_IxF : IndexFile ;Var Dest_IxF : IndexFile); Overload;`

Is_TFileOpen

Declaração `public Function Is_TFileOpen(const a_TFile :  TStream):Boolean;`

Chapter 29

Unit mi.rtl.Objects.Methods.Exception

29.1 Descrição

-A unit `mi.rtl.Objects.Methods.Exception` implementa a classe `TException`(??) do pacote `mi.rtl`(??).

- **VERSÃO**:
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
    - 2021-12-14
      - 11:00 a 11:30 - Criado a unit `mi.rtl.Objects.Methods.Exception` e implementação da classe `TException`(??)
    - 2021-12-15
      - 15:00 a 18:42 - T12 Criar a classe `TException`(??)
      - 21:25 a 22:40 - Troquei o nome de constructor create para que fique equivalente as mensagem de `TStrError.ErrorMessage`(??).

29.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.Consts.StrError(??)`
- `mi.rtl.objects.Methods(??)`

29.3 Visão Geral

`TException Classe`

29.4 Classes, Interfaces, Objetos e Registros

TException Classe

Hierarquia

TException > `TObjectsMethods(??)` > `TObjectsConsts(??)` > TObjectsTypes

Descrição

- A classe `TException` é usada com a palavra reservada **raise** para mostrar o erro, sua localização e em seguida salva no dispositivo definido em **TObjectss.Logs.LogType**.
  - **NOTA**
    - LogType = TLogType = (ltSystem,ltFile,ltStdOut,ltStdErr);
      - ltSystem = Arquivo definido pelo sistema operacional;
      - ltFile = Arquivo definido pela aplicação;
      - ltStdOut,ltStdErr = Terminal do sistema operacional.
  - **EXEMPLO DE USO**:

```
procedure TMi_Rtl_Tests.Action_test_TExceptionExecute(Sender: TObject);
begin
  with TMI_ui_types do begin
    logs.EnableWriteIdentificao := true;
    try
      raise TException.Create(5);
    except
```

```
end;

try
  raise TException.Create('Acesso ao arquivo negado');
except
end;

try
  raise TException.Create(Self, 'Action_test_TExceptionExecute','aFileName','AFieldName',5
except
end;

try
  raise TException.Create(Self, 'Action_test_TExceptionExecute','aFileName','AFieldName','
except
end;


try
  raise TException.Create(Self, 'Action_test_TExceptionExecute',5);
except
end;

try
  raise TException.Create(Self, 'Action_test_TExceptionExecute','Acesso ao arquivo negado'
except
end;


// Os exemplos abaixo são mantidos para manter a compatibilidade com o passado.


  try
   raise TException.Create4('aModule', 'aUnit', 'Procedure_or_Function',   'ParamResult');
 except
 end;

 try
   raise TException.Create4('aModule', 'aUnit', 'Procedure_or_Function',   5);
 except
 end;

 try
   raise TException.Create5('aModule', 'aUnit','ObjectName', 'aMethodName',   'aMsgError'
 except
 end;
```

```
      try
        raise TException.Create5('aModule', 'aUnit','ObjectName', 'aMethodName',   5);
      except
      end;

      try
        raise TException.Create6('aModule', 'ObjectName', 'aMethodName','aFileName','AFieldNam
      except
      end;

      try
        raise TException.Create7('aModule', 'aUnit','ObjectName', 'aMethodName','aFileName','A
      except
      end;

      try
        raise TException.Create7('aModule', 'aUnit','ObjectName', 'aMethodName','aFileName','A
      except
      end;

      try
        raise TException.Create8('aModule', 'aUnit','ObjectName', 'aMethodName','aFileName','A
      except
      end;
    end;
  end;
```

Propriedades

Message

Declaração `public property Message:  Ansistring read FMessage write FMessage;`

Métodos

Create

Declaração `public constructor Create(const Msg:  Ansistring); Overload;`

Create

Declaração `public constructor Create(const aCodError:SmallInt); Overload;`

Create

Declaração `public constructor Create(const Sender: TObject;Const aMethodName, aFileName, AFieldName:AnsiString;aCodError:integer ); Overload;`

Create

Declaração `public constructor Create(const Sender: TObject;Const aMethodName, aFileName, AFieldName:AnsiString;aMsg:AnsiString); Overload;`

Create

Declaração `public constructor Create(const Sender: TObject;Const aMethodName:AnsiString;aCodError:SmallInt ); Overload;`

Create

Declaração `public constructor Create(const Sender: TObject;Const aMethodName:AnsiString;aMsg:AnsiString ); Overload;`

Create4

Declaração `public constructor Create4(Const aModule, aUnit, Procedure_or_Function, aMessage:AnsiString); Overload;`

Create4

Declaração `public constructor Create4(Const aModule, aUnit, Procedure_or_Function:AnsiString; aCodError:SmallInt); Overload;`

Create5

Declaração `public constructor Create5(aModule, aUnit, aObjectName, aMethodName :AnsiString; aCodError:SmallInt); Overload;`

Create5

Declaração `public constructor Create5(aModule, aUnit, aObjectName, aMethodName :AnsiString; aMsgError:AnsiString); Overload;`

Create6

Declaração `public constructor Create6(aModule, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aCodError:SmallInt); Overload;`

Create7

Declaração `public constructor Create7(aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aCodError:SmallInt); Overload;`

Create7

Declaração `public constructor Create7(aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName:AnsiString; aMessage:AnsiString); Overload;`

Create8

Declaração `public constructor Create8(aModule, aUnit, aObjectName, aMethodName, aFileName, AFieldName, aMessage, aProcedure_or_Function :AnsiString); Overload;`

Chapter 30

Unit mi.rtl.Objects.Methods.Paramexecucao

30.1 Uses

- Classes
- SysUtils
- dos
- mi.rtl.objects.Methods(??)
- mi.rtl.objects.Methods.Exception(??)
- mi.rtl.objects.Methods.dates(??)

30.2 Visão Geral

TR_ParamExecucao_Local Registro

TParamExecucao_types Classe

TParamExecucao_consts Classe

TParamExecucao Classe

30.3 Classes, Interfaces, Objetos e Registros

TR_ParamExecucao_Local Registro

Campos

_ParamExecucao

Declaração `public _ParamExecucao: TParamExecucao;`

_Destoy_ParamExecucao

Declaração `public _Destoy_ParamExecucao: Boolean;`

TParamExecucao_types Classe

Hierarquia

TParamExecucao_types > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

no description available, TObjectsMethods description follows

- A classe `TObjectsMethods` implementa os método de classe comum a todas as classes de TObjects do pacote `mi.rtl`(??).

TParamExecucao_consts Classe

Hierarquia

TParamExecucao_consts > `TParamExecucao_types`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

no description available, TParamExecucao_types description followsno description available, TObjectsMethods description follows

- A classe `TObjectsMethods` implementa os método de classe comum a todas as classes de TObjects do pacote `mi.rtl`(??).

Campos

_Set_NomeDeArquivosGenericos

Declaração `public var _Set_NomeDeArquivosGenericos: TSet_NomeDeArquivosGenericos; static;`

Descrição Usado para inicializar os paths da sessão.

Set_NomeDeArquivosGenericos_Global

Declaração `public const Set_NomeDeArquivosGenericos_Global : TSet_NomeDeArquivosGenericos = nil;`

TParamExecucao Classe

Hierarquia

TParamExecucao > `TParamExecucao_consts(??)` > `TParamExecucao_types(??)` > `TObjectsMethods(??)` > `TObjectsConsts(??)` > TObjectsTypes

Descrição

no description available, TParamExecucao_consts description followsno description available, TParamExecucao_types description followsno description available, TObjectsMethods description follows

- A classe `TObjectsMethods` implementa os método de classe comum a todas as classes de TObjects do pacote `mi.rtl(??)`.

Propriedades

HostName

Declaração `public property HostName : AnsiString Read Get_HostName Write SetDominioHost;`

Descrição O campo `HostName` contem o nome ou o ip do banco de dados

- **REFERÊNCIA** [nomes de host da Internet](https://networkencyclopedia.com/hostname/#: :t

Campos

okCreate

Declaração `protected Var okCreate:Boolean;`

NomeDeArquivosGenericos

Declaração `public NomeDeArquivosGenericos: TNomeDeArquivosGenericos;`

PathRaiz

Declaração `public PathRaiz: AnsiString;`

Tipo_de_Execucao

Declaração `public Tipo_de_Execucao: TParamExecucao_Tipo_de_Execucao;`

Identificacao

Declaração `public Identificacao: TIdentificacao;`

Command

Declaração `public Command: SmallInt;`

Modulo

Declaração `public Modulo: SmallInt;`

Acao_Form

Declaração `public Acao_Form: AnsiString;`

DataAtual

Declaração `public DataAtual: TDates.typeData;`

DatabaseNameCharSet

Declaração `public DatabaseNameCharSet: AnsiString;`

List_Value_Default

Declaração `public List_Value_Default:  AnsiString;`

Métodos

Create

Declaração `public Constructor Create(aPathRaiz:Ansistring); overload; Virtual;`

Create

Declaração `public Constructor Create(aOwner:TComponent); overload; override;`

Destroy

Declaração `public Destructor Destroy; override;`

Set_NomeDeArquivosGenericos

Declaração `public Function Set_NomeDeArquivosGenericos() :boolean;`

Acao_Form_Is_Event

Declaração `public Function Acao_Form_Is_Event:Boolean;`

Acao_Form_Is_Mb_Bit

Declaração `public Function Acao_Form_Is_Mb_Bit:Boolean;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial,WUsuario,wPassword: tString); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial:Byte;WUsuario:SmallInt;wPassword:tString); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial,WUsuario,wPassword,wCommand : tString); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial:Byte;WUsuario:SmallInt;wPassword:tString;wCommand : SmallInt); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial:Byte;WUsuario:SmallInt;wNome_Compreto_do_Usuario : tString;wPassword:tString); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial:Byte;WUsuario:SmallInt;wNome_Compreto_do_Usuario : tString;wPassword:tString;aUsername:  tString); Overload;`

Set_ParamStr

Declaração `public procedure Set_ParamStr(wFilial,WUsuario,wPassword,wModulo,wCommand :  tString); Overload;`

Set_ParamStr

Declaração public procedure Set_ParamStr(wFilial,WUsuario,wPassword,wModulo,wCommand,a_DataAtual : tString); Overload;

Set_ParamStr

Declaração public procedure Set_ParamStr(wFilial,WUsuario,wPassword,wModulo,wCommand,a_DataAtual : tString;wAcao_Form: AnsiString); Overload;

Set_ParamStr

Declaração public procedure Set_ParamStr(wFilial,WUsuario,wPassword,wModulo,wCommand,a_DataAtual : tString;wAcao_Form: AnsiString;wList_Value_Default:AnsiString); Overload;

Set_ParamStr

Declaração public procedure Set_ParamStr(wFilial:Byte; WUsuario:SmallInt; wPassword:tString; wModulo:SmallInt; wCommand : SmallInt); Overload;

Set_ParamStr

Declaração public procedure Set_ParamStr(wFilial:AnsiString; WUsuario:SmallInt; wPassword:tString; wModulo:SmallInt; wCommand : SmallInt); Overload;

Check_Se_Comando_autorizado

Declaração public procedure Check_Se_Comando_autorizado;

Get_Password_do_Comando

Declaração `public Function Get_Password_do_Comando(aModulo: Byte; aComando : SmallInt):tString;`

Param_Execucao

Declaração `public class Function Param_Execucao: TParamExecucao;`

Set_ParamExecucao

Declaração `public class Procedure Set_ParamExecucao(aParamExecucao : TParamExecucao);`

Chapter 31

Unit mi.rtl.Objects.Methods.Paramexecucao.App

31.1 Uses

- Classes
- SysUtils
- mi.rtl.types(??)
- mi.rtl.applicationabstract(??)
- mi.rtl.objects.consts.MI_MsgBox
- mi.rtl.objects.consts.progressdlg_if(??)
- mi.rtl.objects.consts.logs(??)
- mi.rtl.objects.methods.ParamExecucao(??)

31.2 Visão Geral

TApplication_type Classe

TApplicationConsts Classe

TApplication Classe

application

Setapplication

31.3 Classes, Interfaces, Objetos e Registros

TApplication_type Classe

Hierarquia

TApplication_type > `TApplicationAbstract`(**??**) > TCustomApplication

Descrição

A class *`TApplication_type`** é usada para capsular todas as variáveis globais do projeto e gerenciar o ciclo de vida do aplicativo

TApplicationConsts Classe

Hierarquia

TApplicationConsts > `TApplication_type`(**??**) > `TApplicationAbstract`(**??**) > TCustomApplication

Descrição

no description available, TApplication_type description followsA class *`TApplication_type`** é usada para capsular todas as variáveis globais do projeto e gerenciar o ciclo de vida do aplicativo

Campos

origin

Declaração `public origin: TTypes.TPoint;`

Descrição Ponto inferior a esqueda da aplicação

Size

Declaração `public Size: TTypes.TPoint;`

Descrição Ponto superior a direta da aplicação

MI_MsgBox

Declaração `public MI_MsgBox: TMI_MsgBox;`

Logs

Declaração `public Logs: TFilesLogs;`

Descrição

- `Logs` é inicializado em Initialization e destruído em finalization

TApplication Classe

Hierarquia

TApplication > `TApplicationConsts`(**??**) > `TApplication_type`(**??**) > `TApplicationAbstract`(**??**) > TCustomApplication

Descrição

no description available, TApplicationConsts description followsno description available, TApplication_type description followsA class *`TApplication_type`** é usada para capsular todas as variáveis globais do projeto e gerenciar o ciclo de vida do aplicativo

Campos

ParamExecucao

Declaração `public ParamExecucao: TParamExecucao;`

Métodos

Create

Declaração `public constructor Create(AOwner: TComponent); override;`

Destroy

Declaração `public destructor Destroy; override;`

FileOptions_CommandEnabled

Declaração `public function FileOptions_CommandEnabled(aCommand: AnsiString): Boolean; Virtual;`

Descrição O método `FileOptions_CommandEnabled` deve ser redefinido na aplicações filhas para indicar se o comando a ser executado está habilitado no arquivo de opções.

EnableCommands

Declaração `public procedure EnableCommands(aCommands: TCommandSet); virtual;`

DisableCommands

Declaração `public procedure DisableCommands(aCommands: TCommandSet); virtual;`

31.4 Funções e Procedimentos

application

Declaração `function application: TApplication;`

Setapplication

Declaração `Procedure Setapplication(aApplication : TApplication);`

Chapter 32

Unit mi.rtl.Objects.Methods.StreamBase

32.1 Descrição

- A unit `mi.rtl.Objects.Methods.StreamBase` implementa a classe `TStreamBase`(??) do pacote `mi.rtl`(??).
  - **NOTAS**
    - O Use da classe `mi.rtl.Objects.Methods.StreamBase` não deve ser instanciada antes de implementar os métodos abstratos;
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **19/11/2021** 21:25 a 23:15 Criar a unit mi.rtl.objects.methods.StreamBase.pas
      - **20/11/2021** 14:02 a 15:19 Documentação da classe e agrupar métodos virtuais, métodos não virtuais e proteger os métodos abstratos.
  - **CÓDIGO FONTE**:
    -

32.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.types`(??)
- `mi.rtl.objects.Methods`(??)

32.3 Visão Geral

`TStreamBase Classe`

32.4 Classes, Interfaces, Objetos e Registros

TStreamBase Classe

Hierarquia

TStreamBase > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

- A class `TStreamBase` é uma classe abstrata para implementação de streams.

Campos

Status

Declaração `public Status: Integer;`

Descrição Stream `status`

StreamSize

Declaração `public StreamSize: int64;`

Descrição Stream current size

Position

Declaração `public Position:  Int64;`

Descrição Current `position`

Alias

Declaração `public Alias:  AnsiString;`

Métodos

Create

Declaração `public constructor Create; overload; virtual;`

Destroy

Declaração `public destructor Destroy; Override;`

Open

Declaração `protected procedure Open; overload; Virtual;`

Close

Declaração `public procedure Close; Virtual;`

Rewrite

Declaração `protected procedure Rewrite; Overload; Virtual;`

Flush

Declaração `protected procedure Flush; Virtual;`

Truncate

Declaração `protected procedure Truncate; Overload; Virtual;`

Read

Declaração `protected procedure Read(Var Buf; Count: Sw_Word); Overload; Virtual;`

Write

Declaração `public procedure Write(Var Buf; Count: Sw_Word); Overload; Virtual;`

ReadStr

Declaração `public function ReadStr: ptstring;`

Get

Declaração `public function Get: TClass;`

StrRead

Declaração `public function StrRead: PAnsiChar;`

Put

Declaração `public procedure Put(P: TClass);`

StrWrite

Declaração `public procedure StrWrite(P: PAnsiChar);`

WriteStr

Declaração `public procedure WriteStr(P: ptstring);`

CopyFrom

Declaração `public procedure CopyFrom(Var S: TStreamBase; Count:  LongInt);`

GetPos

Declaração `public function GetPos:  LongInt; Virtual;`

GetSize

Declaração `public function GetSize:  LongInt; Virtual;`

Reset

Declaração `public procedure Reset; Overload; Virtual;`

Seek

Declaração `public procedure Seek(Pos:  LongInt); overload; Virtual;`

Seek

Declaração `public procedure Seek(NR: LongInt;a_RecSize:Longint); Overload; Virtual;`

Error

Declaração `public procedure Error(Code, Info:  Integer); Virtual;`

GetDriveType

Declaração `public function GetDriveType:TDriveType; overload; virtual;`

Chapter 33

Unit mi.rtl.Objects.Methods.StreamBase.Stream

33.1 Descrição

- A Unit `mi.rtl.Objects.Methods.StreamBase.Stream` implementa a classe `TStream(??)` do pacote `mi.rtl(??)`.
  - **NOTAS**
    - Está unit foi testada nas plataformas: win32, win64 e linux.
    - Como o linux não tem opção de travar a região de uma arquivo eu removi as classes **_TRecLock** e **TCollRecsLocks**.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **20/11/2021** - 09:10 a ??:?? Criar a unit mi.rtl.objects.methods.StreamBase.Stream.pas
      - **22/11/2021**
      - 09:44 a 12:05 Adaptar `TStream(??)` ao free pascal;
      - 14:10 a 19:05 Adaptar **_TStream** e `TStream(??)` ao free pascal;

    -
  - **CÓDIGO FONTE**:
    -

33.2 Uses

- Classes
- SysUtils
- mi.rtl.types(??)
- mi.rtl.consts(??)
- mi.rtl.files(??)
- mi.rtl.objects.types(??)
- mi.rtl.objects.Methods(??)
- mi.rtl.objects.methods.StreamBase(??)

33.3 Visão Geral

TStream Classe

33.4 Classes, Interfaces, Objetos e Registros

TStream Classe

Hierarquia

TStream > TStreamBase(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

- A class TStream é a classe base da classes **_TStream** do pacote mi.rtl(??).

Propriedades

BaseSize

Declaração published property BaseSize: Longint Read _BaseSize write SetBaseSize;

RecSize

Declaração published property RecSize: Longint Read _RecSize write SetRecSize;

Ok_Aguarde

Declaração `published property Ok_Aguarde: Boolean Read _Ok_Aguarde write Set_Ok_Aguarde;`

FileMode

Declaração `published property FileMode : Word Read _FileMode write SetFileMode;`

ShareMode

Declaração `published property ShareMode : Cardinal read _ShareMode write SetShareMode;`

FileName

Declaração `published property FileName : AnsiString read GetFileName write SetFileName;`

Campos

_Base

Declaração `protected _Base: Pointer;`

_Rec

Declaração `protected _Rec: Pointer;`

Status_Rewrite

Declaração `public Status_Rewrite: Byte;`

ClockBegin

Declaração `public ClockBegin:  DWord;`

Last_Mode

Declaração `public Last_Mode:  TLast_Mode_Read_Write;`

State

Declaração `public var State:  Longint;`

Ok_FreeMem_Rec

Declaração `protected Ok_FreeMem_Rec:Boolean;`

_FileName

Declaração `protected _FileName:  AnsiString;`

Métodos

Set_BaseSize

Declaração `public procedure Set_BaseSize(a_Base :
Pointer;a_BaseSize:Longint); Overload; Virtual;`

SetBaseSize

Declaração `protected procedure SetBaseSize(a_BaseSize : Longint); Overload;
Virtual;`

Set_RecSize

Declaração `public procedure Set_RecSize(a_Rec : Pointer;a_RecSize:Longint); Overload; Virtual;`

SetRecSize

Declaração `protected procedure SetRecSize(a_RecSize : Longint); Overload; Virtual;`

Calc_Pos

Declaração `public Function Calc_Pos(NR: LongInt;a_RecSize:Longint):Longint;`

FileSize

Declaração `public function FileSize: Longint; overload; Virtual;`

Seek

Declaração `public procedure Seek(NR: LongInt;a_RecSize:Longint); Overload; override;`

Create

Declaração `public constructor Create(); overload; override;`

Destroy

Declaração `public destructor Destroy; Override;`

Set_Ok_Aguarde

Declaração `protected procedure Set_Ok_Aguarde(a_Ok_Aguarde:  Boolean); Virtual;`

CloseOpen

Declaração `public function CloseOpen:Integer; VIRTUAL;`

Flush_Disk

Declaração `public function Flush_Disk:Integer; Virtual;`

Flush

Declaração `public procedure Flush; Override;`

Read

Declaração `public procedure Read(Var Buf; Count:  Sw_Word;Var BytesRead:Sw_Word) ; Overload; Virtual;`

Write

Declaração `public procedure Write(Var Buf; Count:  Sw_Word;Var BytesWrite:Sw_Word); Overload; Virtual;`

SetFileMode

Declaração `public procedure SetFileMode(Const aFileMode:Word); virtual;`

SetShareMode

Declaração `public procedure SetShareMode(Const aShareMode:Cardinal); virtual;`

SetStateFileMode

Declaração `public function SetStateFileMode(Const AState: Longint; Const Enable: boolean):Boolean;`

GetStateFileMode

Declaração `public function GetStateFileMode(Const AState: Longint): Boolean;`

Reset

Declaração `public procedure Reset; overload; Override;`

Reset

Declaração `public procedure Reset(aFileMode: Word;ShareMode : Cardinal); overload; Virtual; abstract;`

Rewrite

Declaração `public procedure Rewrite; overload; override;`

Rewrite

Declaração `public procedure Rewrite(aFileMode: Word;ShareMode : Cardinal); Overload; Virtual; abstract;`

SetBufSize

Declaração `public function SetBufSize(Const aBufSize : Sw_Word):Sw_Word; Overload; Virtual;`

IsFileOpen

Declaração `public function IsFileOpen:Boolean; Virtual;`

GetRecBase

Declaração `public function GetRecBase(Var RecBase):Integer; Overload; Virtual;`

PutRecBase

Declaração `public function PutRecBase(Var RecBase):Integer; Overload; Virtual;`

GetRecBase

Declaração `public function GetRecBase:Integer; Overload; Virtual;`

PutRecBase

Declaração `public function PutRecBase:Integer; Overload; Virtual;`

GetRec

Declaração `public function GetRec(Nr:  Longint;Var Rec):Integer; Overload; Virtual;`

PutRec

Declaração `public function PutRec(Nr:  Longint;Var Rec):Integer; Overload; Virtual;`

GetRec

Declaração `public function GetRec(Nr:  Longint):Integer; Overload; Virtual;`

PutRec

Declaração `public function PutRec(Nr:  Longint):Integer; Overload; Virtual;`

BlockRead

Declaração `public function BlockRead(Nr:  Longint;Var Blocks ; Const Count: Longint):Longint; Virtual;`

BlockWrite

Declaração `public function BlockWrite(Nr:  Longint;Var Blocks ; Const Count: Longint):Longint; Virtual;`

Error

Declaração `public procedure Error(Code, Info:  Integer); Override;`

Truncate

Declaração `public procedure Truncate(Pos:  LongInt); Overload; Virtual;`

CopyFrom

Declaração `public procedure CopyFrom(Var S: TStream; Count:  LongInt); Overload; Virtual;`

CopyFrom

Declaração `public procedure CopyFrom(Var S: TStream ); Overload; Virtual;`

Bof

Declaração `public function Bof:Boolean; Virtual;`

Eof

Declaração `public function Eof:Boolean; Virtual;`

goBof

Declaração `public function goBof:Boolean;`

goEof

Declaração `public function goEof:Boolean;`

SetFileName

Declaração `protected procedure SetFileName(a_FileName:  AnsiString); Virtual;`

GetFileName

Declaração `protected function GetFileName:  AnsiString; Virtual;`

Chapter 34

Unit mi.rtl.Objects.Methods.StreamBase.Stream.

34.1 Descrição

- A Unit `mi.rtl.Objects.Methods.StreamBase.Stream.FileStream` implementa a classe `TFileStream`(**??**) do pacote `mi.rtl`(**??**).
  - **NOTAS**
    - Implementa banco um fluxo de dados em disco.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **22/11/2021**
      - 17:00 a 19:45 Criar a unit `mi.rtl.Objects.Methods.StreamBase.Stream.FileStream`
      - **29/11/2021**
      - 10:10 a 12:01 - t12 Documentar a classe `TFileStream`(**??**). Exemplo 01: Test_FileStream_com_header
      - 13:50 a 14:33 - t12 Documentar a classe `TFileStream`(**??**). Exemplo 01: Test_FileStream_sem_header
      - **21/12/2021**
      - 15:30 a 16:20 - T12 Transferir os métodos de TDosStream para TFileStrem.
  - **CÓDIGO FONTE**:
    -

34.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.types`(??)
- `mi.rtl.consts`(??)
- `mi.rtl.files`(??)
- `mi.rtl.objects.types`(??)
- `mi.rtl.objects.Methods`(??)
- `mi.rtl.objects.methods.StreamBase`(??)
- `mi.rtl.objects.methods.StreamBase.Stream`(??)
- `mi.rtl.objects.consts.MI_MsgBox`

34.3 Visão Geral

`TFileStream Classe`

34.4 Classes, Interfaces, Objetos e Registros

TFileStream Classe

Hierarquia

TFileStream > `TStream`(??) > `TStreamBase`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

- A classe `TFileStream` é usada para criar um fluxo de dados em disco usada no banco de dados maricarai, onde é possível adicionar um registro no início do arquivo de tamanho maior ou menor que os registros seguintes ao registro zero.
  - **EXEMPLOS**
    - Test_FileStream_sem_header

```
Procedure TMi_Rtl_Tests.Test_FileStream_sem_header;
  type
   TAluno = record
              Id : integer;
              nome : string[35];
            end;

  var
    FileStream_Alunos : TObjectss.TFileStream;
    aluno : TAluno;//Registro de aluno

    nr : longint; //Número do registro.
     n  : longint; //Contador

begin
  with TObjectss do
  try
    fillchar(aluno,sizeof(aluno),' ');

    if TObjectss.FileExists(expandFileName('aluno.txt'))
    then FileStream_Alunos := TFileStream.Create(expandFileName('aluno.txt'),fileMode)
    else FileStream_Alunos := TFileStream.Create(expandFileName('aluno.txt'),fileMode,fmCreate

    with aluno,FileStream_Alunos do
    if status = StOk then
    begin
      //Define o tamanho do registro
      recSize := sizeof(aluno);

      //Adiciona o registro 0;
      if status = StOk then
      begin
        n := 0;
        Id:= n;
        nome:= 'Paulo Sergio';
        PutRec(n,aluno);
      end;

      //Adiciona o registro 1;
      if status = StOk then
      begin
        inc(n);
        Id:= n;
        nome:= 'George Bruno';
```

```
        PutRec(n,aluno);
      end;

      // Ler e imprime os registros salvos acima.
       if status = StOk then
      begin
        for nr := 0 to n do
        begin
          GetRec(nr,aluno);
          if Status = Stok
          then begin
                 SysMessageBox('Nr ='+intToStr(nr)+    '; id ='+intToStr(aluno.id)+'; Aluno =
                               ,'Test_FileStream_sem_header'
                               ,false);

               end
          else break;
        end;
      end;//if
    end; //with

    with FileStream_Alunos do
      if status <> StOk
      then SysMessageBox(ErrorMessage(ErrorInfo),'Test_FileStream_sem_header',true);

  finally
    FileStream_Alunos.Destroy;
  end;

end;
```

* Test_FileStream_com_header

```
procedure TMi_Rtl_Tests.Test_FileStream_com_header;

  type

   // Tipo de registro 1 ao final do arquivo:
    TAluno = record
               Id : integer;
               nome : string[35];
             end;
```

```
   // Tipo de registro a ser usado no registro zero do arquivo.

    THeadAlunos = record
                    TotalDeAlunos:longint;
                  end;

  var
    FileStream_Alunos : TObjectss.TFileStream;
    aluno     : TAluno;//Registro de aluno
    headAluno : THeadAlunos;

    nr : longint; //Número do registro.
     n  : longint; //Contador

//Início da procedure
begin
 with TObjectss do
 try
   fillchar(aluno,sizeof(aluno),' ');

   if TObjectss.FileExists(expandFileName('aluno.txt'))
   then FileStream_Alunos := TFileStream.Create(expandFileName('aluno.txt'),fileMode)
   else FileStream_Alunos := TFileStream.Create(expandFileName('aluno.txt'),fileMode,fmCreate

   with aluno,FileStream_Alunos do
   if status = StOk then
   begin
     //Define o tamanho do registro zero
     baseSize := sizeof(headAluno);

     //Define o tamanho do registro
     recSize := sizeof(aluno);

     headAluno.TotalDeAlunos := 0;
     PutRecBase(headAluno);

     //Adiciona o registro 0;
     if status = StOk then
     begin
       inc(headAluno.TotalDeAlunos);
       n := headAluno.TotalDeAlunos;
       Id:= n;
       nome:= 'Paulo Sergio';
       PutRec(n,aluno);
       PutRecBase(headAluno);
     end;
```

```
    //Adiciona o registro 1;
    if status = StOk then
    begin
      inc(headAluno.TotalDeAlunos);
      n := headAluno.TotalDeAlunos;
      nome:= 'George Bruno';
      PutRec(n,aluno);
      PutRecBase(headAluno);
    end;

    // Ler e imprime os registros salvos acima.
     if status = StOk then
    begin
      getRecBase(headAluno);
      if status = StOk then
      begin
        SysMessageBox('Número de registros='+intToStr(headAluno.TotalDeAlunos)
                     ,'Test_FileStream_sem_header'
                     ,false);

        for nr := 1 to headAluno.TotalDeAlunos do
        begin
          GetRec(nr,aluno);
          if Status = Stok then
          begin
            SysMessageBox('Nr ='+intToStr(nr)+    '; id ='+intToStr(aluno.id)+'; Aluno ='+al
                         ,'Test_FileStream_sem_header'
                         ,false);

          end else break;
        end;
      end;
    end;
  end; //with

  with FileStream_Alunos do
    if status <> StOk
    then SysMessageBox(ErrorMessage(ErrorInfo),'Test_FileStream_sem_header',true);

 finally
   FileStream_Alunos.Destroy;
 end;
end;
```

Campos

_ShareModeAnt

Declaração `protected _ShareModeAnt: CARDINAL;`

Handle

Declaração `public Handle: THandle;`

Descrição DOS file `handle`

Métodos

SetShareMode

Declaração `protected procedure SetShareMode(Const a_ShareMode:CARDINAL); override;`

SetFileName

Declaração `protected procedure SetFileName(a_FileName: AnsiString); Override;`

Create

Declaração `public CONSTRUCTOR Create(aFName: AnsiString; aFileMode: Word;aShareMode:Cardinal); overload; virtual;`

Create

Declaração `public CONSTRUCTOR Create(aFName: AnsiString; aFileMode: Word); overload; virtual;`

Create

Declaração `public CONSTRUCTOR Create(aFileName: AnsiString; aFileMode: Word;Size: Sw_Word;a_BaseSize,a_RecSize:Longint); overload; virtual;`

GetDriveType

Declaração `public Function GetDriveType:TDriveType; Override;`

Destroy

Declaração `public DESTRUCTOR Destroy; Override;`

Truncate

Declaração `public PROCEDURE Truncate; Overload; Override;`

Seek

Declaração `public procedure Seek(NR: LongInt;a_RecSize:Longint); Overload; override;`

Open

Declaração `public PROCEDURE Open; overload; Override;`

Open

Declaração `public PROCEDURE Open(aFileMode: Word;aShareMode:Cardinal); Overload; Virtual;`

Close

Declaração `public PROCEDURE Close; Override;`

Reset

Declaração `public PROCEDURE Reset; Overload; Override;`

Reset

Declaração `public PROCEDURE Reset(aFileMode: Word;aShareMode : Cardinal); Overload; override;`

Rewrite

Declaração `public PROCEDURE Rewrite; Overload; Override;`

Rewrite

Declaração `public procedure Rewrite(aFileMode: Word;aShareMode : Cardinal); Overload; Override;`

Read

Declaração `public PROCEDURE Read(Var Buf; Count: Sw_Word); Overload; Override;`

Write

Declaração `public PROCEDURE Write(Var Buf; Count: Sw_Word); Overload; Override;`

GetSize

Declaração `public FUNCTION GetSize:  LongInt; Override;`

CloseOpen

Declaração `public Function CloseOpen:Integer; Override;`

Descrição

- O método `CloseOpen` é usado para obrigar o windows a descarregar o buffer do arquivo.
  - **NOTA**
    - O linux não tem a função dulicateHandle.

Flush_Disk

Declaração `public Function Flush_Disk:Integer; Override;`

Flush

Declaração `public PROCEDURE Flush; Override;`

IsFileOpen

Declaração `public Function IsFileOpen:Boolean; Override;`

DeleteFile

Declaração `public Procedure DeleteFile;`

CreateFileStream

Declaração `public function CreateFileStream(aFName:  AnsiString; aFileMode: Word) :  TFileStream; Virtual;`

SaveToFile

Declaração `public function SaveToFile(aFileName:AnsiString):Boolean; Overload; Virtual;`

SaveToFile

Declaração `public function SaveToFile:Boolean; Overload; Virtual;`

LoadFromFile

Declaração `public function LoadFromFile(aFileName:AnsiString):Boolean; Overload; virtual;`

Chapter 35

Unit mi.rtl.Objects.Methods.StreamBase.Stream.

35.1 Descrição

- A Unit `mi.rtl.Objects.Methods.StreamBase.Stream.MemoryStream` implementa a classe `TMemoryStream`(??) do pacote `mi.rtl`(??).
  - **NOTAS**
    - Implementa um fluxo de dados em memória.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **23/11/2021**
      - 06:10 a 07:17 - Criar a unit `mi.rtl.Objects.Methods.StreamBase.Stream.MemoryStream`
      - 07:43 a - Documentar a unit `mi.rtl.Objects.Methods.StreamBase.Stream.MemoryStream`.
  - **CÓDIGO FONTE**:
    -

35.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.files`(??)
- `mi.rtl.objects.methods.StreamBase.Stream`(??)
- `mi.rtl.objects.methods.StreamBase.Stream.FileStream`(??)

35.3 Visão Geral

`TMemoryStream Classe`

35.4 Classes, Interfaces, Objetos e Registros

TMemoryStream Classe

Hierarquia

TMemoryStream > `TStream`(??) > `TStreamBase`(??) > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

- A classe `TMemoryStream` é usada para gerenciar um fluxo de dados em memória.
  - **NOTA**
    - Todas as alterações aqui devem ser completamente transparentes para os códigos existentes. Basicamente, os blocos de memória não precisam ser segmentos de base mas isso significa que nossa lista se torna blocos de memória em vez de segmentos. O stream também se expandirá como os outros streams padrão

Campos

BlkCount

Declaração `public BlkCount: Sw_Word;`

Descrição Number of segments

BlkSize

Declaração `public BlkSize: Word;`

Descrição Memory block size

MemSize

Declaração `public MemSize: LongInt;`

Descrição Memory alloc size

BlkList

Declaração `public BlkList: PPointerArray;`

Descrição Memory block list

Handle

Declaração `public Handle: THandle;`

Descrição Quando `Handle`=HANDLE_INVALID o bloco de memória não foi alocado

Métodos

LoadFromFile

Declaração
```
protected function LoadFromFile(aFileName:AnsiString):Boolean;
Virtual;
```

SetBufSize

Declaração
```
public Function SetBufSize(Const aBufSize : Sw_Word):Sw_Word;
Override;
```

SetBufSize

Declaração `public Function SetBufSize(ALimit: LongInt; ABlockSize: Word):Sw_Word; Overload; Virtual;`

SetFileName

Declaração `public Procedure SetFileName(a_FileName: AnsiString); Override;`

Create

Declaração `public CONSTRUCTOR Create(ALimit, ABlockSize: Longint); overload; virtual;`

Destroy

Declaração `public DESTRUCTOR Destroy; Override;`

Truncate

Declaração `public PROCEDURE Truncate; Override;`

Read

Declaração `public PROCEDURE Read(Var Buf; Count: Sw_Word;Var BytesRead:Sw_Word); Overload; override;`

Read

Declaração `public PROCEDURE Read(Var Buf; Count: Sw_Word); Overload; Override;`

Write

Declaração `public PROCEDURE Write(Var Buf; Count: Sw_Word;Var BytesWrite:Sw_Word); Overload; override;`

Write

Declaração `public PROCEDURE Write(Var Buf; Count: Sw_Word); Overload; Override;`

Chapter 36

Unit mi.rtl.objects.methods.StreamBase.Stream.M

36.1 Descrição

- A Unit `mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMemory` implementa a classe `TBufferMemory`(**??**).
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **23/11/2021**
      - 12:55 a 14:30 - Criar a unit `mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMe`
      - 14:30 a 19:35 - Criar um exemplo de como usar a classe `TBufferMemory`(**??**)
      - 21:35 a 22:44 - Documentar a classe `TBufferMemory`(**??**)
      - **29/11/2021**
      - 14:45 a 15:10
      - Criar exemplo TMi_Rtl_Tests.Test_TBufferMemory_sem_header;
      - Criar exemplo TMi_Rtl_Tests.Test_TBufferMemory_com_header;
  - **CÓDIGO FONTE**:
    -

36.2 Uses

- Classes
- SysUtils
- mi.rtl.objects.methods.StreamBase.Stream.MemoryStream(??)

36.3 Visão Geral

TBufferMemory Classe

36.4 Classes, Interfaces, Objetos e Registros

TBufferMemory Classe

Hierarquia

TBufferMemory > TMemoryStream(??) > TStream(??) > TStreamBase(??) > TObjectsMethods(??) > TObjectsConsts(??) > TObjectsTypes

Descrição

- A class TBufferMemory cria um **array of record** em memória usando os métodos os seek(??), PutREc(??), GetRec(??)
  - **NOTA**
    - Uso a classe TBufferMemory para criar arquivos em memória no banco de dados **Tb_Access.pas**
  - **EXEMPLO**
    - Exemplo de como gravar um registro **sem header em memória**.

```
Procedure TMi_Rtl_Tests.Test_TBufferMemory_sem_header;
  type
   TAluno = record
              Id : integer;
              nome : string[35];
            end;

  var
   FileMemory_Alunos : TObjectss.TBufferMemory;
   Aluno          : TAluno;
```

```
   nr : longint;
   n  : longint;

begin
  with TObjectss do
  try
    FileMemory_Alunos := TObjectss.TBufferMemory.Create(sizeof(aluno));
    with aluno,FileMemory_Alunos do
    begin
      if status = StOk then
      begin
        n := 1;
        Id:= n;
        nome:= 'Paulo Sérgio';
        PutRec(id,aluno);
      end;

      if status = StOk then
      begin
        inc(n);
        Id:= n;
        nome:= 'George Bruno';
        PutRec(id,aluno);
      end;

      if status = StOk then
      begin
        for nr := 1 to n do
        begin
          GetRec(nr,aluno);
          if status = StOk
          then SysMessageBox('Nr ='+intToStr(nr)+
                             '; id ='+intToStr(Aluno.id)+
                             '; Aluno ='+Aluno.nome
                             ,'Test_FileStream_sem_header',false)
          else break;
        end;
      end;

    end;

  finally
    FileMemory_Alunos.Destroy;
  end;

end;
```

∗ Exemplo de como gravar um registro **com header em memória**.

```
Procedure Test_TBufferMemory_com_header;
  type
    TAluno = record
               Id : integer;
               nome : string[35];
             end;
  type
    THeadAlunos = record
                    TotalDeAlunos:longint;
                  end;

  var
    TBufferMemory_Alunos : TObjectss.TBufferMemory;
    HeadAlunos : THeadAlunos;
    Aluno              : TAluno;
    nr : longint;
    n  : longint;

begin
  with TObjectss do
  try
    TBufferMemory_Alunos := TBufferMemory.Create(sizeof(HeadAlunos),sizeof(aluno));

    with aluno,TBufferMemory_Alunos do
    if status = StOk then
    begin
      HeadAlunos.TotalDeAlunos:= 0;
      PutRecBase(HeadAlunos); // Grava o header

      if status = StOk then
      begin
        inc(HeadAlunos.TotalDeAlunos);
        Id:= HeadAlunos.TotalDeAlunos;
        nome:= 'Paulo Sérgio da Silva Pacheco';
        PutRec(id,aluno);
        if status = StOk
        then PutRecBase(HeadAlunos); // Grava o header
      end;


      if status = StOk then
```

```
begin
  inc(HeadAlunos.TotalDeAlunos);
  Id:= HeadAlunos.TotalDeAlunos;
  nome:= 'George Bruno Melo Pacheco';

  PutRec(id,aluno);
  if status = StOk
  then PutRecBase(HeadAlunos); // Grava o header
end;

if status = StOk then
begin
  GetRecBase(n);
  if status = StOk
  then
  begin
    //Imprime o número de elemntos adicionado ao stream
    SysMessageBox('Número de registros: '+intToStr(n)
                  ,
                  'Test_FileStream_sem_header',false);

    // Ler e imprime os registros.
     for nr := 1 to n do
    begin
        GetRec(nr,aluno);
        if status = StOk
        then SysMessageBox('Nr ='+intToStr(nr)+
                           '; id ='+intToStr(Aluno.id)+
                           '; Aluno ='+Aluno.nome
                           ,
                           'Test_FileStream_sem_header',false)
        else Break;
    end;

    if status <> StOk
    then SysMessageBox(errorMessage(errorInfo)
                       ,
                       'Test_FileStream_sem_header',false)

  end;
end;

if status <> StOk
then SysMessageBox(errorMessage(errorInfo)
                   ,
                   'Test_FileStream_sem_header',false)
```

```
    end;

  finally
    TBufferMemory_Alunos.Destroy;
  end;

end;
```

Métodos

Create

Declaração `public CONSTRUCTOR Create(a_BaseSize,a_RecSize:Longint); overload; override;`

Descrição

- O constructor `Create` cria um stream de um **array of record** em memória onde a mesma será gravado após o header passado pelo parâmetro a_BaseSize.;
  - **PARÂMETROS**
    - **a_BaseSize** - Tamanho do registro usado no registro de posição zero
    - **a_RecSize** - Tamanho do registro depois do registro usado na posição depois da base;

Create

Declaração `public CONSTRUCTOR Create(a_RecSize:Longint); overload; virtual;`

Descrição

- O constructor `Create` cria um stream de um **array of record** em memória onde a mesma será gravado após ao início do bloco em memória obs: `BaseSize`(**??**)=0.;

Seek

Declaração `public PROCEDURE Seek(NR: LongInt); Overload; override;`

Error

Declaração `public PROCEDURE Error(Code, Info:  Integer); Override;`

Chapter 37

Unit mi.rtl.Objects.Methods.System

37.1 Descrição

SISTEMA : Nort Soft Data Base MODULO : MARICARAY AUTOR : Paulo Pacheco ——— HISTORIA ——— DATA HARA HORA OCORRENCIA ——— —— —— ——————————————————————— —— 01/08/02 08:00 Implementacao inicial 08/08/02 23:00 Implementacao Final 25/01/22 Convertido para lazarus *

37.2 Uses

- `Classes`
- `SysUtils`
- `crt`
- `mi.rtl.objects.Methods`(??)
- `mi.rtl.objects.Methods.Exception`(??)
- `mi.rtl.objects.methods.StreamBase.Stream`(??)
- `mi.rtl.objects.methods.StreamBase.Stream.FileStream`(??)
- `mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMemory`(??)
- `mi.rtl.objects.methods.Collection.FilesStreams`(??)

37.3 Visão Geral

`TObjectsSystem Classe`

37.4 Classes, Interfaces, Objetos e Registros

TObjectsSystem Classe

Hierarquia

TObjectsSystem > `TObjectsMethods`(??) > `TObjectsConsts`(??) > TObjectsTypes

Descrição

no description available, TObjectsMethods description follows

- A classe `TObjectsMethods` implementa os método de classe comum a todas as classes de TObjects do pacote `mi.rtl`(??).

Campos

BlocksRead

Declaração `public const BlocksRead :  Word = 0;`

BlocksWrite

Declaração `public const BlocksWrite :  Word = 0;`

Métodos

FlushDOSFile

Declaração `public class function FlushDOSFile(VAR F : File):Boolean;`

BlockRead

Declaração `public class function BlockRead(var F: File; var Buf; Count: Word):Word;`

BlockWrite

Declaração `public class function BlockWrite(var F: File; var Buf; Count: Word):Word;`

Seek

Declaração `public class function Seek(VAR F:FILE ;Const NR: Longint):SmallInt ;`

AppendText

Declaração `public class function AppendText(VAR F:Text;AFileMode:Word):SmallInt ;`

Rewrite

Declaração `public class function Rewrite(VAR F:Text;AFileMode:SmallWord):SmallInt ; overload;`

Rewrite

Declaração `public class function Rewrite(VAR F:File;Const aRecLen:Integer;AFileMode:SmallWord):SmallInt ; overload;`

Close

Declaração `public class function Close(Var F:File):Boolean; Overload;`

Close

Declaração `public class function Close(Var F:Text):Boolean; Overload;`

Reset

Declaração `public class function Reset(VAR F:FILE ;Const aRecLen:Integer;Const AFileMode:SmallWord):Integer ; Overload;`

OpenText

Declaração `public class function OpenText(VAR F:Text;Mode:  Word):SmallInt ;`

Reset

Declaração `public class function Reset(VAR F:Text ;Const AFileMode:SmallWord):SmallInt ; Overload;`

FileFlushBuffers

Declaração `public class function FileFlushBuffers(VAR F : File):Boolean; overload;`

FileFlushBuffers

Declaração `public class function FileFlushBuffers(VAR F : File; const Ok_FileFlushBuffers :  Boolean):Boolean; Overload;`

Size_LinFeed_Text

Declaração `public class function Size_LinFeed_Text(aFileName : AnsiString):SmallInt ;`

FTempoDeTentativas

Declaração `public class procedure FTempoDeTentativas(Const HcHelp:SmallInt);`

Is_TFileOpen

Declaração `public class function Is_TFileOpen(const a_TFile : TStream):Boolean;`

CopyFiles

Declaração `public class function CopyFiles(const SourceName, TargetName: AnsiString):Integer;`

DeleteFiles

Declaração `public class function DeleteFiles(const SourceName:AnsiString):Integer;`

Existe_Espaco_em_Dobro

Declaração `public class function Existe_Espaco_em_Dobro:  Boolean;`

Chapter 38

Unit mi.rtl.objects.types

38.1 Descrição

- A Unit `mi.rtl.objects.types` implementa a classe **TObjectsTypes** .
  - **NOTAS**
    - Esta unit foi testada nas plataformas: win32, win64 e linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **17/11/2021** 20:30 a 22:49 - Criada a classe **TObjectsTypes**. Falta conclui...
      - **18/11/2021** 09:05 - Concluir a classe **TObjectsTypes**.
      - **15/12/2021** 15:00 a 15:15 - Revisar a documentação da unidade.

38.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.files`(??)

38.3 Visão Geral

`DummyClass Classe`

38.4 Classes, Interfaces, Objetos e Registros

DummyClass Classe

Hierarquia

DummyClass > TObject

Descrição

- Internal Class

Campos

Data

Declaração `public Data:  Record`

Chapter 39

Unit mi.rtl.Objectss

39.1 Descrição

- A Unit `mi.rtl.Objectss` reune todas as classes base pacote `mi.rtl(??)`.
  - **NOTAS**
    - Esta unit foi testada nas plataformas: win32, win64 e linux.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **20/11/2021** 9:10 a ??: Criar a unit mi.rtl.objects.pas -
  - **CÓDIGO FONTE**:
    -

39.2 Uses

- `Classes`
- `SysUtils`
- `mi.rtl.Objects.Methods.Paramexecucao.Application(??)`
- `mi.rtl.types(??)`

- mi.rtl.Consts(??)
- mi.rtl.Consts.StringListBase(??)
- mi.rtl.Consts.StringList(??)
- mi.rtl.files(??)
- mi.rtl.objects.types(??)
- mi.rtl.objects.consts(??)
- mi.rtl.objects.consts.MI_MsgBox
- mi.rtl.objects.consts.progressdlg_if(??)
- mi.rtl.objects.Methods(??)
- mi.rtl.objects.Methods.Dates(??)
- mi.rtl.objects.methods.ParamExecucao(??)
- mi.rtl.objects.Methods.Exception(??)
- mi.rtl.objects.methods.StreamBase(??)
- mi.rtl.objects.methods.StreamBase.Stream(??)
- mi.rtl.objects.methods.StreamBase.Stream.MemoryStream(??)
- mi.rtl.objects.methods.StreamBase.Stream.MemoryStream.BufferMemory(??)
- mi.rtl.objects.methods.StreamBase.Stream.FileStream(??)
- mi.rtl.objects.methods.Collection(??)
- mi.rtl.objects.methods.Collection.SortedCollection(??)
- mi.rtl.objects.methods.Collection.SortedCollection.StrCollection(??)
- mi.rtl.objects.methods.Collection.SortedCollection.stringCollection(??)
- mi.rtl.objects.methods.Collection.SortedCollection.stringcollection.CollectionString(??)
- mi.rtl.objects.methods.Collection.FilesStreams(??)
- mi.rtl.objects.methods.db.tb_access(??)
- mi.rtl.objects.methods.db.tb__access(??)
- mi.rtl.objects.methods.db.tb___access(??)

39.3 Visão Geral

TObjectss Classe

39.4 Classes, Interfaces, Objetos e Registros

TObjectss Classe

Hierarquia

TObjectss >

Campos

Application

Declaração `public const Application : TApplication = nil;`

Métodos

Set_MI_MsgBox

Declaração `public Class Procedure Set_MI_MsgBox(aMI_MsgBox: TMI_MsgBox); Virtual;`

ProcStreamError

Declaração `public class Procedure ProcStreamError(Const S: TStreambase);`

StrToSItem

Declaração `public class Function StrToSItem(Const StrMsg:AnsiString; Colunas : byte;Alinhamento:TAlinhamento):PSItem;`

WriteSItems

Declaração `public class procedure WriteSItems(var S: TCollectionString; Const Items: PSItem);`

PSItem_ListaDeMsgErro

Declaração `public class Function PSItem_ListaDeMsgErro:PSItem; override;`

MessageError

Declaração `public class Procedure MessageError; override;`

Chapter 40

Unit mi.rtl.Types

40.1 Descrição

- A Unit `mi.rtl.Types` reune os tipos globais usados pelo pacote `mi.rtl(??)`. Esta unit foi testada nas plataformas: no linux.
  - **NOTA**
    - O Método **TTypes.TPointer.Get_Mem** ignora alocação de memória real porque não sei como fazer nas plataformas diferentes do Windows.
  - **VERSÃO**
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - **Period** : June to September of 2001)
      - **14/09/2001** : I begin of the version: Windows 98
      - **29/10/2021** : Portado para o compilador free pascal para os sistemas operacionais: 1. x86_64-linux 2. x86_64-win64 3. i386-win32
      - **02/11/2021** : Trabalhei na documentação com pasdoc.
      - **12/11/2021**

· A Unit **mi.rtl.types** foi convertida para unit **mi.types**.

· Criado a class `TTypes(??)` com todos os tipos definidos em **mi.rtl.types** com objetivo de encapsular os tipos globais do pacote `mi.rtl`(**??**).

· **13/11/2021**

· Documentação da unit `mi.rtl.Types`.

· **15/12/2021**

· Criado o tipo registro TIndentificação.

40.2 Uses

- `Classes`
- `Dos`
- `SysUtils`

40.3 Visão Geral

`TTypes Classe`

40.4 Classes, Interfaces, Objetos e Registros

TTypes Classe

Hierarquia

TTypes > TComponent

Descrição

A classe `TTypes` declara todos os tipos globais do pacote MarIcarai

Campos

Alias

Declaração `public Alias:  AnsiString;`

ok_Set_Transaction

Declaração `public const ok_Set_Transaction : BOOLEAN = false;`

Descrição

- A constant `ok_Set_Transaction` indica se o processo está dentro de uma transação.

MAX_BYTE

Declaração `public const MAX_BYTE = high(SmallWord);`

MAX_ARRAY_BYTE

Declaração `public const MAX_ARRAY_BYTE = MAX_BYTE div sizeof(byte);`

MAX_INT

Declaração `public const MAX_INT = high(Integer);`

MAX_ARRAY_INT

Declaração `public const MAX_ARRAY_INT = MAX_INT div sizeof(integer);`

MAX_SMALL_INT

Declaração `public const MAX_SMALL_INT = high(SmallInt);`

MAX_ARRAY_SMALL_INT

Declaração
```
public const MAX_ARRAY_SMALL_INT = MAX_SMALL_INT div
sizeof(SmallInt);
```

MAX_LONG_INT

Declaração `public const MAX_LONG_INT = high(LongInt);`

MAX_ARRAY_LONG_INT

Declaração `public const MAX_ARRAY_LONG_INT = MAX_LONG_INT div sizeof(Longint);`

MAX_WORD

Declaração `public const MAX_WORD = high(Word);`

MAX_ARRAY_WORD

Declaração `public const MAX_ARRAY_WORD = MAX_WORD div sizeof(word);`

MAX_SMALL_WORD

Declaração `public const MAX_SMALL_WORD = high(system.word);`

MAX_ARRAY_SMALL_WORD

Declaração
```
public const MAX_ARRAY_SMALL_WORD = MAX_SMALL_WORD div
 sizeof(system.word);
```

MAX_LONG_WORD

Declaração `public const MAX_LONG_WORD = high(LongWord);`

MAX_ARRAY_LONG_WORD

Declaração
```
public const MAX_ARRAY_LONG_WORD = MAX_LONG_WORD div
 sizeof(LongWord);
```

MAX_POINTER

Declaração `public const MAX_POINTER = MAX_ARRAY_WORD;`

Descrição O ideal seria memAvail, porém esta função não é multiplataforma;

MAX_ARRAY_PTR

Declaração `public const MAX_ARRAY_PTR = MAX_POINTER div sizeof(Pointer);`

FileNameLen

Declaração `public const FileNameLen :  integer = Dos.FileNameLen;`

Descrição Usado para compatibilidade com o passado;

evNothing

Declaração `public const evNothing = $0000;`

evMouseDown

Declaração `public const evMouseDown = $0001;`

evMouseUp

Declaração `public const evMouseUp = $0002;`

evMouseMove

Declaração `public const evMouseMove = $0004;`

evMouseAuto

Declaração `public const evMouseAuto = $0008;`

evKeyDown

Declaração `public const evKeyDown = $0010;`

evCommand

Declaração `public const evCommand = $0100;`

evBroadcast

Declaração `public const evBroadcast = $0200;`

EvAplCliSvr

Declaração `public const EvAplCliSvr = $0400;`

evMouse

Declaração `public const evMouse = $000F;`

evKeyboard

Declaração `public const evKeyboard = $0010;`

evMessage

Declaração `public const evMessage = $FF00;`

SizeOffldCluster

Declaração `public const SizeOffldCluster : TSizeOffldCluster = sizeof(TSizeOffldCluster);`

SizeOffldDbCluster

Declaração `public const SizeOffldDbCluster = 50;`

Métodos

Create

Declaração `public constructor Create(aowner:TComponent); Overload; Override;`

CheckEmpty

Declaração `public class procedure CheckEmpty(Var Rect: TTypes.TRect);`

Chapter 41

Unit mi.ui.dialogs

41.1 Descrição

- A unit `mi.ui.dialogs` implementa a classe `TDialogs`(**??**) do pacote mi.ui.
  - **VERSÃO**:
    - Alpha - 0.5.0.687
  - **CÓDIGO FONTE**:
    -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco e-mail: paulosspacheco@yahoo.com.br
      - 2021-12-02
      - 23:00 a 23:35 - Criado a unit `mi.ui.dialogs` e implementação da classe `TDialogs`(**??**)
    - 2021-12-03
      - 09:40 a 12:00
      - Criar método de classe Confirm();
      - Criar método de classe Prompt();
      - Criar método de classe Password();
    - **2021-12-04**
      - 15:11 a 16:40

- Criar exemplo TForm1.Test_tobjects_dlgs_Confirm;
- Criar exemplo TForm1.Test_tobjects_dlgs_Prompt;
- Criar exemplo TForm1.Test_tobjects_dlgs_password;

41.2 Uses

- `Classes`
- `SysUtils`
- `Forms`
- `Dialogs`
- `Graphics`
- `StdCtrls`
- `mi.rtl.objects.consts`(??)
- `mi.rtl.objects.Methods`(??)
- `mi.rtl.objects.consts.dialogs`

41.3 Visão Geral

`TDialogs Classe`

41.4 Classes, Interfaces, Objetos e Registros

TDialogs Classe

Hierarquia

TDialogs > mi.rtl.objects.consts.dialogs.TDialogs

Métodos

Create

Declaração `public constructor Create(aOwner:  TObjectsConsts); overload; override;`

CreateMessageDialog

Declaração `public function CreateMessageDialog( aCaption, aMsg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): integer; overload;`

Descrição

- O método `CreateMessageDialog` mostra uma mensagem formatada onde a função reconhece ^M para passagem de linha, ^J retorno do carro e ^C.
  - **NOTA**
    - O texto entre ^C vai ficar alinhado no centro do topo do formulário.
  - **EXEMPLO**

Alert

Declaração `public Procedure Alert(aTitle: AnsiString;aMsg:AnsiString); override;`

Descrição

- A procedure `Alert` executa um dialogo com botão **OK**

Confirm

Declaração `public Function Confirm(aTitle: AnsiString;aPergunta:AnsiString):Boolean; override;`

Descrição

- A procedure `Confirm` executa um diálogo com dois botões: **OK** e **Cancel**
  - **RETORNA:**
    - **True** : Se o botão **OK** foi pressionando;
    - **False** : Se o botão **Cancel** foi pressionando.

– **EXEMPLO**

```
procedure TForm1.Test_tobjects_dlgs_Confirm;
begin
  with TObjectss.dlgs do
    if Confirm('Test_tobjects_dlgs_Confirm','Continua o processament
     then Alert('Test_tobjects_dlgs_Confirm','Confirmado a ação!')
     else Alert('Test_tobjects_dlgs_Confirm','Não confirmado a ação!
end;
```

Prompt

Declaração `public Function Prompt(aTitle: AnsiString;aPergunta:AnsiString;Var aResult: AnsiString):Boolean; override;`

Descrição

- A função `Prompt` mostra um dialogo com dois botões **OK** e **Cancel** e uma entrada de dados solicitando que o usuário digite um valor.

  – **RETORNA:**

    * **True** : Se o botão **ok** foi pŕessionando;
    * **False** : Se o botão **cancel** foi pŕessionando.
    * **aResult** : Retorna a string digitada no formulário;

  – **EXEMPLO**

```
procedure TForm1.Test_tobjects_dlgs_Prompt;
  var idade,fmt : string;
begin
  idade := '';
  with TObjectss.dlgs do
    if Prompt('Test de Dlgs.Prompt','Qual a sua idade',idade)
    then begin
           fmt := format('Idade digitada: %s   '+^M+
                         'Idade de meu pai é %d ',[idade,102]);
           Alert('Test de Dlgs.Prompt',fmt) //
         end
    else Alert('Test de Dlgs.Prompt','Ok. Respeito sua privacidade.
end;
```

GetPassword

Declaração `public Function GetPassword(aTitle: AnsiString; var apassword:AnsiString):Boolean; Overload; override;`

Descrição

- A função `GetPassword` mostra um diálogo para receber um valor sem mostrar o que foi digitado. O formulário possui dois botões **OK** e **Cancel**
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pressionado;
    - **False** : Se o botão **cancel** foi pressionado.
    - **apassword** : Retorna a string com a senha do usuário.

GetPassword

Declaração `public Function GetPassword(aTitle: AnsiString; var aUsername:AnsiString; var apassword:AnsiString):Boolean; Overload; override;`

Descrição

- A função `GetPassword` mostra um dialogo solicitando o login do usuário e a senha e dois botões **OK** e **Cancel**
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pŕessionando;
    - **False** : Se o botão **cancel** foi pŕessionando.
    - **aUsername** : Retorna a string com nome do usuário.
    - **apassword** : Retorna a string com a senha do usuário.
  - **EXEMPLO**

```
procedure TForm1.Test_tobjects_dlgs_password;
  Var
    s,u : string;
begin
  s := '';
  with TObjectss.dlgs do
    if GetPassword('Password',u,s)
    then Alert('Password','A senha digitada é: '+S)
    else Alert('Password','Senha não informada');

end;
```

41.5 Constantes

_Dialogs

Declaração `_Dialogs : mi.ui.Dialogs.TDialogs = nil;`

Chapter 42

Unit mi.ui.lcl.form

42.1 Uses

- uMi_ui_scrollbox_lcl(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)
- umi_ui_bitbtn_lcl(??)
- umi_ui_button_lcl(??)
- umi_ui_checkbox_lcl(??)
- umi_ui_radiogroup_lcl(??)
- uMi_ui_ComboBox_lcl(??)
- uMi_Ui_DBCheckBox_Lcl(??)
- uMi_Ui_DbComboBox_lcl(??)
- uMI_ui_DbEdit_LCL(??)
- umi_ui_dblookupComboBox_lcl(??)
- uMI_ui_DbRadioGroup_Lcl(??)
- uMi_ui_Label_lcl(??)
- uMi_ui_maskedit_lcl(??)
- uMi_ui_Dmxscroller_form_lcl
- umi_ui_InputBox_lcl(??)
- umi_ui_dmxscroller_form_lcl_ds(??)
- LazarusPackageIntf

Chapter 43

Unit mi_rtl_ui_custom_application

43.1 Descrição

A unit `mi_rtl_ui_custom_application` implementa a classe `TMI_ui_Custom_Application`(??).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **PENDÊNCIAS**
- **REFERÊNCIA**

-CHARSET-SUPPORTED (https://www.postgresql.org/docs/current/multibyte.html#MULTIBYTE-CHARSET-SUPPORTED)

icial do componente **sqldb** (https://www.freepascal.org/docs-html/fcl/sqldb/index.html)

Exemplos de uso do **SqlDb** (https://www.freepascal.org/docs-html/fcl/sqldb/usingsqldb.html)

SqlDBHowto (https://wiki.freepascal.org/SqlDBHowto)

tsqlquery.insertsql (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.insertsql.html)

- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
    - **2022-03-29 16:06**
      - Criar a unit `mi_rtl_ui_custom_application` e analisar o que preciso fazer para integrar com a unit `mi_ui_Dmxscroller_sql`(??)

* **2022-04-06 15:40**

    - Implementar o evento Get_ParametersCloseQuery e salvar o formulário quando ele for executado.
    - No evento Get_ParametersCloseQuery Checar se o usuário é válido.
    - Criar método DoOnValidUser
    - Criar método **Get_ParametersCloseQuery** para executar o evento DoOnValidUser.

* **2022-04-07 08:43**

    - Cada banco de dados SQL tem alguns parâmetros básicos para sua conexão:
    - Criar as propriedades de `TMI_ui_Custom_Application`(**??**) para que o usuário informe esses parâmetros:
    - Banco de dados PostgresSQL
    - CharSet = 'UTF8';
    - ConnectorType:='PostgreSQL';
    - HostName := '127.0.0.1';
    - UserName := 'postgres';
    - Password := 'masterkey';
    - DatabaseName:= 'maricarai';
    - DirDatabaseName:= './';
    - connected :Boolean
    - Options : TSQLConnectorOptions
    - Documentar as propriedade criadas hoje .

* **2022-04-08**

    - **09:00**
    - Escrever a descrição da classe `TMI_ui_Custom_Application`(**??**).
    - **11:18**
    - Alterar o nome da propriedade **Options** para **SQLConnectorOptions**.
    - Criar a propriedade **SQLTransactionOptions**

- **14:22**
- Documentar as propriedades da classe `TMi_ui_Custom_Application`(**??**).
- **21:40**
- Em `TMI_ui_Custom_Application.Get_ParametersCloseQuery`(**??**) antes de checar se os parãmetros são válidos, transferir os campos do formulários para as propriedades equivalentes.

* **2022-04-14 14:58**
  - Criar a constante **OkCreateDataBase** e o método **CreateDataBase**.
* **2022-04-15 10:00**
  - Criar método **NameDataBase** que retorna o nome do database porque o nome do dataBase é diferente em cada banco de dados. O postres usa um nome simples e o ip para acessar o banco, o SqLite3 usa o nome da pasta + nome do database + ext.

43.2 Uses

- `Classes`
- `SysUtils`
- `SqlDb`
- `DB`
- `BufDataset`
- `PQConnection`
- `CustApp`
- `mi.rtl.Types`(**??**)
- `mi_rtl_ui_Dmxscroller`(**??**)
- `mi.rtl.Objects.Methods.Paramexecucao.Application`(**??**)

43.3 Visão Geral

`TMI_ui_Custom_Application Classe`

`Mi_ui_Custom_Application`

`Set_Mi_ui_Custom_Application`

43.4 Classes, Interfaces, Objetos e Registros

TMI_ui_Custom_Application Classe

Hierarquia

TMI_ui_Custom_Application > `TApplication`(??) > `TApplicationConsts`(??) > `TApplication_type`(??) > `TApplicationAbstract`(??) > TCustomApplication

Descrição

no description available, TApplication description followsno description available, TApplicationConsts description followsno description available, TApplication_type description followsA class *`TApplication_type`** é usada para capsular todas as variáveis globais do projeto e gerenciar o ciclo de vida do aplicativo

Propriedades

SQLConnectorOptions

Declaração `published property SQLConnectorOptions : TSQLConnectionOptions Read _SQLConnectorOptions write SetSQLConnectorOptions default [];`

Descrição A propriedade `SQLConnectorOptions` é usada para controlar o comportamento do SqlDb para esta conexão.

- As seguintes opções podem ser definidas:
  - Type TSQLConnectionOption = (scoExplicitConnect, scoApplyUpdatesChecksRowsAffected);
  - **ONDE**:
    - **scoExplicitConnect** :
      - Quando definido, a conexão deve ser feita explicitamente.
      - O comportamento padrão é para **TSQLQuery** abrir implicitamente a conexão conforme necessário.
    - **scoApplyUpdatesChecksRowsAffected** :
      - Quando definido, sempre que uma instrução SQL de atualização é executada durante **ApplyOptions** de um conjunto de dados, **RowsAffected** é verificado e deve ser igual a 1.

- **REFERÊNCIAS**

tsqltransaction.options (https://www.freepascal.org/docs-html/fcl/sqldb/tsqltransaction.options.html)

SQLTransactionOptions

Declaração `published property SQLTransactionOptions : TSQLTransactionOptions Read _SQLTransactionOptions write SetSQLTransactionOptions default [];`

Descrição A propriedade `SQLTransactionOptions` é usada para controlar o comportamento do SqlDb para esta transação.

- As seguintes opções podem ser definidas:
  - Type TSQLTransactionOption = (stoUseImplicit, stoExplicitStart);
  - **ONDE**:
    - **stoUseImplicit** :
      - Use o suporte a transações implícitas do mecanismo de banco de dados. Isso significa que nenhum comando explícito de início e parada de transação será enviado ao servidor quando os métodos **Commit** ou **Rollback** forem chamados (tornando-os efetivamente sem operação no nível do banco de dados).
    - **stoExplicitStart**
      - Quando definido, sempre que uma instrução SQL é executada, a transação deve ter sido iniciada explicitamente. O comportamento padrão é que **TSQLStatement** ou **TSQLQuery** iniciem a transação conforme necessário.

- **REFERÊNCIAS**

tsqltransaction.options (https://www.freepascal.org/docs-html/fcl/sqldb/tsqltransaction.options.html)

Connected

Declaração `published property Connected : Boolean Read GetConnected write SetConnected;`

Descrição A propriedade `Connected` conecta ao banco de dados selecionado.

- True = Conecta ao banco;
- False = Desconecta do banco;

ConnectorType

Declaração `published property ConnectorType : TUiDmxScroller.TConnectorType Read _ConnectorType write _ConnectorType;`

Descrição O evento `ConnectorType` seleciona o tipo de banco de dados a ser conectado

HostName

Declaração `published property HostName : AnsiString Read GetHostName write SetHostName;`

Descrição A propriedade `HostName` informa ao `SQLConnector(??)` o **IP** ou domínimo onde o banco de dados foi hospedado.

DirDataBaseName

Declaração `published property DirDataBaseName : AnsiString Read GetDirDataBaseName write SetDirDataBaseName;`

Descrição A propriedade `DirDataBaseName` contém a pasta do HD do servidor onde o banco de banco foi hospedado.

- **Não foi implementado ainda**
  - Preciso de mais informações de como alterar a pasta dos bancos de dados PostgreSQL e SQLite3.

DatabaseName

Declaração `published property DatabaseName : AnsiString Read GetDatabaseName write SetDatabaseName;`

Descrição A propriedade `DatabaseName` contém o nome do Banco de Dados dentro do PostegresSQL ou do SQLite3.

UserName

Declaração `published property UserName : AnsiString Read GetUserName write SetUserName;`

Descrição A propriedade `UserName` contém o nome do usuário conectado ao banco de dados.

Password

Declaração `published property Password : AnsiString Read GetPassword write SetPassword;`

Descrição A propriedade `Password` contém a senha do usuário conectado ao banco de dados.

CharSet

Declaração `published property CharSet : AnsiString Read GetCharSet write SetCharSet;`

Descrição A propriedade `CharSet` é usada para definir o tipo de caractere do banco de dados.

- **NOTA**
  - Deve ser informado em tempo de designe do projeto.
- **REFERÊNCIAS**

`CHARSET`-TABLE (https://www.postgresql.org/docs/current/multibyte.html#CHARSET-TABLE)

onValidUser

Declaração `published property onValidUser : TOnValidUser Read _OnValidUser write _onValidUser;`

Descrição O evento `onValidUser` é disparado toda vez que o `TUiDmxScroller`(**??**) ativado.

SQLConnector

Declaração `published property SQLConnector : TSQLConnector read _SQLConnector;`

Descrição A propriedade `SQLConnector` é um componente conector de banco de dados versátil para uso com qualquer banco de dados suportado.

- A incluir uma aplicação **TMi_UI_Application** na aplicação corrente automáticamente é disponibilizado um conector de acesso o banco de dados.
- **REFERÊNCIAS**

TSQLConnector (https://wiki.freepascal.org/TSQLConnector)

sqldb/tsqlconnector (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlconnector.html)

SQLTransaction

Declaração `published property SQLTransaction : TSQLTransaction read _SQLTransaction;`

Descrição A propriedade `SQLTransaction` representa uma transação no banco de dados na qual um TSQLQuery é tratado.

- Na prática, pelo menos uma transação precisa estar ativa para um banco de dados, mesmo que você a utilize apenas para leitura de dados.
- **NOTAS**
  - Ao usar uma única transação, defina a propriedade TConnection. Transaction para a transação para definir a transação padrão para o banco de dados; a propriedade TSQLTransaction.Database correspondente deve apontar automaticamente para a conexão.

- Ao ativar uma TSQLTransaction o método **StartTransaction** inicia uma transação; chamar o método **Commit** ou o método **RollBack** confirma (salva) ou reverte (esquece/aborta) a transação.
    - Você deve cercar suas transações de banco de dados com eles, a menos que use as propriedades **Autocommit** ou **CommitRetaining**.

- **REFERÊNCIAS**

tsqltransaction (https://www.freepascal.org/docs-html/fcl/sqldb/tsqltransaction.html)

Get_Parameters

Declaração `published property Get_Parameters : TUiDmxScroller read _Get_Parameters;`

Descrição A propriedade `Get_Parameters` contém o formulário para ler os parâmetros de conexão com o banco de dados.

Campos

BufDataSet1

Declaração `public BufDataSet1: TBufDataSet;`

Descrição O atributo `BufDataSet1` é usado para salvar em disco local no arquivo `FileName_Parameters(??)` os parâmetros informados pelo formulário `Get_Parameters(??)`

DataSource1

Declaração `public DataSource1: TDataSource;`

Descrição O atributo `DataSource1` permite integrar os dados da classe **TMiDmxScroller** com os componentes da LCL com DbGrid, DbEdit etc...

OkCreateDataBase

Declaração `public const OkCreateDataBase : boolean = false;`

Descrição A constante `OkCreateDataBase` se **true** executa o método `CreateDataBase(??)` se Existe-CreateDataBase = false

Const_ConnectorType

Declaração `public const Const_ConnectorType : Array[TUiDmxScroller.TConnectorType] of AnsiString =('PostgreSQL','SqLite3');`

Descrição A constante `Const_ConnectorType` contém a lista de nomes dos tipo de bancos de dados testados pelo componente `TMI_ui_Custom_Application(??)`

FileName_Parameters

Declaração `public const FileName_Parameters : AnsiString = '';`

Descrição A constante `FileName_Parameters` contém o nome do arquivo de parâmetros

- A constante `FileName_Parameters` é inicializado em `TMI_ui_Custom_Application.create(??)` onde:
  - `FileName_Parameters` := ParamStr(0)+'_Parameters.bds';

_Get_Parameters

Declaração `protected _Get_Parameters: TUiDmxScroller;`

Descrição Este atributo é usado pelas classes filhas para implementar classes herdadas de `TUiDmxScroller(??)`.

- No momento (08/04/22 a classe que herdade é: **TUiDmxScroller_form**)

Métodos

ExistDataBase

Declaração `public function ExistDataBase:Boolean;`

Descrição O Método `ExistDataBase` retorna **true** se o banco de dados existe e **false** se não existir.

CreateDataBase

Declaração `public function CreateDataBase:boolean;`

Descrição O método `CreateDataBase` cria o banco de dados se a constante `OkCreateDataBase`(**??**) = true

Get_ParametersEnter

Declaração `public procedure Get_ParametersEnter(aDmxScroller: TUiDmxScroller);`

Descrição O método `Get_ParametersEnter` é usado pela classe `Get_Parameters`(**??**).

- Esse evento cria o arquivo de parâmetros usando os dados das propriedades de `TMI_ui_Custom_Application`(**??**) definidas no tempo de projeto.

Get_ParametersExit

Declaração `public procedure Get_ParametersExit(aDmxScroller: TUiDmxScroller);`

Get_ParametersCloseQuery

Declaração `public Procedure Get_ParametersCloseQuery(aDmxScroller:TUiDmxScroller; var CanClose:boolean);`

Descrição O método `Get_ParametersCloseQuery` é usado para confirmar o fechamento do formulário `Get_Parameters`(**??**) com botão **MrOK** caso os campos de `Get_Parameters`(**??**) sejam válidos.

- **NOTA**
  - Método `Get_ParametersCloseQuery` executa o evento `DoOnValidUser`(**??**), se o mesmo for assinalado na aplicação com objetivo de não permitir fechar o formulário modal com botão MrOK caso `DoOnValidUser`(**??**) retornar false.
  - Pode ser usado para checar se usuário e senha são válidos bem como se os parâmetros estão compatíveis com os bancos de dados instalados.

Login_GetTemplate

Declaração `public Function Login_GetTemplate( aNext :  PSItem ) :  PSItem;`

Descrição O método `Login_GetTemplate` retorna um Template usado para criar o formulário de entrada de dados para a conexão.

NameDataBase

Declaração `published Function NameDataBase:AnsiString;`

Descrição O método `NameDataBase` retorna o nome do banco de dados de acordo com o tipo de banco de dados.

DoOnValidUser

Declaração `public function DoOnValidUser(aDmxScroller:TUiDmxScroller;aUserName:AnsiString;aPassword:AnsiString):boolean virtual;`

Descrição O método `DoOnValidUser` executa o evento `onValidUser(??)` se o mesmo for assinalado na aplicação ou retorna true se `onValidUser(??)` = **nil**

Create_Get_Parameters

Declaração `protected procedure Create_Get_Parameters; virtual; Abstract;`

Descrição O método `Create_Get_Parameters` deve ser implementado para criar classe TUiDmxScroller_form_lcl ou TUiDmxScroller_form_HTML_Angular4.

Create

Declaração `public constructor Create(AOwner:  TComponent); override;`

Descrição O constructor `Create` cria os componentes `SQLConnector(??)`, `SQLTransaction(??)`, `BufDataSet1(??)`, `DataSource1(??)`, Inicia a constante `FileName_Parameters(??)`, executa o método `Create_Get_Parameters(??` inicializa `charSet(??)` e liga os componentes `SQLConnector(??)` com `SQLTransaction(??)` e os componentes **DataSource1.DataSet :=** `BufDataset1(??)`.

Destroy

Declaração `public destructor Destroy; override;`

Descrição O destructor `Destroy` destrói as classes criadas pelo constructor da classe

43.5 Funções e Procedimentos

Mi_ui_Custom_Application

Declaração `function Mi_ui_Custom_Application: TMI_ui_Custom_Application;`

Descrição A função `Mi_ui_Custom_Application` retorna a ultima instância de `TMI_ui_Custom_Application(??)` criada no sistema

Set_Mi_ui_Custom_Application

Declaração
```
Function Set_Mi_ui_Custom_Application(aMi_ui_Custom_Application :
TMI_ui_Custom_Application): TMI_ui_Custom_Application;
```

Descrição A função `Set_Mi_ui_Custom_Application` seta a ultima instância de `TMI_ui_Custom_Application(??)` criada no sistema e retorna aplicação selecionada anteriormente.

43.6 Tipos

TOnValidUser

Declaração
```
TOnValidUser = function
(aDmxScroller:TUiDmxScroller;aUserName:AnsiString;aPassword:AnsiString):boolean
of Object;
```

Descrição O tipo `TOnValidUser` é usado no evento OnValidUser

Chapter 44

Unit mi_rtl_ui_Dmxscroller

44.1 Descrição

A unit `mi_rtl_ui_Dmxscroller` implementa a classe `TUiDmxScroller`(**??**) e registro `TDmxFieldRec`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **HISTÓRICO**
  -
- **CÓDIGO FONTE**:
  -
  - **PENDÊNCIAS**
    - T12 Quando uma linha em um label tem muitos caracteres de 2 bytes os últimos não são interpretados.
    - T12 Implementar o campo FldLink. (Esse campo executa um ação usando controle TStaticText.
    - T12 O controle TComboBox da LCL alterar o tamanho da fonte courie New caso o tema do sistema mude.
      - Pesquisar sobre o assunto.
    - T12 No método SetString em caso de erro de gera exceção informando valor máximo do campo e não o valor digitado.

* T12 Implementar o evento OnChange em todos os controles, visto que o mesmo é mais `f(??)`ácil criar lógica de negócios visto que o mesmo só é executado se o campo for modificado.
* T12 Implementar a possibilidade das fontes do label ser personalizada baseado em um estilo que pode ser uma variável global.
    - Suponha que ^Z = <h1> Título e ^D = <B> de negrito então o sistema informa a TDmxFieldRec.Style = nome do estilo onde nome do estilo = 'Font = FonteX; Size= XX; etc.. '
    - Exemplo:

      ^ZCADASTRO DE ALUNOS

      ^DÑome do Aluno: \sssssssss
* T12 Na construção do formulário LCL setar o campo PDmxFieldRec.LinkEdit;
* T12 Implementar o método: function FieldByNum(aFieldnum:Integer):`PDmxFieldRec(??)`;
* T12 Implementar a edição `FldBoolean(??)`.
    - Os campo Boolean deve ser editados como uma campo enumerado onde:
    - 0 - False; não
    - 1 = True; sim
* T12 O campo `fld_LHora(??)` não inicializado antes de compactar a hora.
* T12 Quando o usuário teclar tab para passar o campo e o campo seguinte não estiver visível o sistema deve passar a página do controle parent.
* T12 Implementar a edição de campo `FldMemo(??)`.
* T!2 Implementar a campo `fldBLOb(??)`;
* t12 Implementar a edição de `fldHexValue(??)`.
    - O campo Hexadecimal deve ser campo longint mais a edição é uma string comum . `FldStr(??)`
* T12 Implementar a propriedade AlignmentLabels := taCenter; AlignmentLabels := taLeftJustify; AlignmentLabels := taRightJustify ;
* T12 Implementar a execução do evento do tipo CharExecProc quando a tecla F7 é pressionada.
* T12 Criar opção para gerar cliente HTML a partir de TDmxScroller

        - Referência: [Componente que espoe dados para o browser](https://wiki.freepascal.org/SqlDbRestBridg

    * T12 O grupo `TMi_RadioGroup_Lcl`(**??**) não é selecionado com a tecla na tecla **TAB**

        - Quando os botões TRadioButton estão dentro do TRadioGroup a propriedade TRadioGroup.TabStop não funciona.

    * T12 Nosso código só é executado com o editor de propriedade. Se não estamos no editor de propriedade então não temos controle do código no modo design. Qual o meu problema: O formulário deve ser criado em tempo de execução, porém eu queria ver como ele estava ficando sem precisar compilar e executar o código, por isso coloquei o código em um stringList e ao ativar o objeto, o formulário é criado. Porém esses objetos criados no designer não podem ficar no arquivo de recursos porque quando for executado vai haver duplicidade.

        - Quando eu desativo o objetos todos os objetos que ele criou são excluídos do arquivo de recursos.

        - Isso eu já faço agora, mais quando distribuir o componente as pessoas vão deixar esses componente usado no teste e ao executar vai haver error.

        - Por isso eu queria que caso a propriedade active tivesse em true eu queria que ela ficasse em false.

- **CONCLUÍDO**

    - T12 O campo FldCheckBox não está funcionando o flag `charHint`(**??**) .

    - T12 Implementar o controle ChatHint no Template para seja possível passar um documento markdown pelo Template; .

    - T12 Ao executar o evento OnExit é necessário o redraw em de todos os campo caso haja alteração ao retorna da chamada. .

    - T12 O componente TMi_ui_Button_lcl não está na lista dos campos selecionados na tecla tab.

    - T12 Os campos `FldEnum`(**??**) não estão mostrando o help.

    - T12 Criar a propriedade Locked;

    - T12 No pacote mi.rtl.ui, transferir toda dependência do pacote LCL para o pacote mi.rtl.form.

44.2 Uses

- `Classes`
- `SysUtils`
- `db`

- BufDataset
- SqlDb
- mi.rtl.Objects.Consts.Mi_MsgBox
- mi.rtl.objects.Methods.dates(??)
- mi_rtl_ui_Types(??)
- mi.rtl.Consts(??)
- mi_rtl_ui_methods(??)

44.3 Visão Geral

TFldEnum_Lookup Classe

TDmxFieldRec Registro

TUiDmxScroller Classe

44.4 Classes, Interfaces, Objetos e Registros

TFldEnum_Lookup Classe

Hierarquia

TFldEnum_Lookup > TComponent

Descrição

A classe TFldEnum_Lookup é usada para implementar campo ComboBox quando TDmxScroller estiver TDataSource <> nil porque o Lazarus espera em campos ComboBox um string e não o índice da lista de strings.

Propriedades

DmxFieldRec

Declaração public property DmxFieldRec : pDmxFieldRec read _DmxFieldRec Write SeTDmxFieldRec;

Descrição A propriedade DmxFieldRec contém o campo comboBox se ser editado

Campos

BufDataSet

Declaração `public BufDataSet: TBufDataSet;`

Descrição O atributo `BufDataSet` contém o arquivo em memória das opções do campo ComboBox sendo editado.

DataSource

Declaração `public DataSource: TDataSource;`

Descrição O atributo `DataSource` é a fonte de dados associado a `TFldEnum_Lookup.BufDataSet`(**??**) do campo sendo editado.

KeyField

Declaração `public KeyField: AnsiString;`

Descrição O atributo `KeyField` contém o nome do campo chave da tabela associada.

ListField

Declaração `public ListField: AnsiString;`

Descrição O atributo `ListField` contém o nome do campo da tabela associada a ser visualizado.

Métodos

create

Declaração `public constructor create(aDmxFieldRec : pDmxFieldRec); overload;`

Descrição O constructor `create` cria os campos TBufDataSet e TDataSource do campo `TFldEnum_Lookup`(**??**)

destroy

Declaração `public destructor destroy; override;`

Descrição O destructor `destroy` destrói os campos TBufDataSet e TDataSource do campo `TFldEnum_Lookup`(**??**)

TDmxFieldRec Registro

Descrição

O registro `TDmxFieldRec` é usado para guardar as informações passadas pelos Templates das strings.

- **REFERÊNCIA**

Estrutura record e object https://wiki.freepascal.org/Record

- A aparência padrão dessas visualizações geralmente é orientada por coluna/linha, com exceção de exibições do tipo formulário e campos únicos.
- Você declara uma estrutura de registro para o procedimento de inicialização do **tvDMX** em um modelo string – que também determina o formato de exibição. (Você verá mais tarde como o **tvDMX** pode ser usado para trabalhar com formulários ou editores de campo.)
- **EXEMPLO**
  - O `Template`(**??**) '\ sssssssss'sssssssssss \ iiii \ rrr.rr' representa o registro:
    - **CÓDIGO PASCAL**

```
type

  TRecord = Record
              nome : String [20];
              Ano  : Integer;
              Valor : Real;
            end;
```

    - **NOTA:**
      - A letra (**s**) minúsculo aceita qualquer número e letras maiúsculas e minúsculas;
      - A letra (**i**) representa um número inteiro com 2 bytes com edição em 4 posições (0 a 9999);

- A letra (**r**) representa um número real com 8 bytes com edição em 5 posições (0 a 999.99)
- O símbolo (**‘**) crase é usado para informar que a parte do texto depois deste sinal deve ser omitida da visão.
- A símbolo (**’ \ ’**) barra invertida deve ser usada como delimitador de campo e é exibida como um espaço em branco.
- O símbolo ( ) til deve ser usado para separar rótulos dos campos de dados.

- **ATENÇÃO**
  - O registro `TDmxFieldRec` não pode ser **class** e nem conter **métodos virtuais**, porque este registro e alocado com as funções **new** e **dispose**.

Propriedades

FieldName

Declaração `public property FieldName : AnsiString read _FieldName write SetFieldName;`

Descrição O campo `FieldName` guarda o nome do campo e deve ser inicializado em CreateStruct

ID_Dynamic

Declaração `public property ID_Dynamic : AnsiString Read _ID_Dynamic Write _ID_Dynamic;`

owner

Declaração `public property owner : TUiDmxScroller read _owner write Set_owner;`

FieldAltered

Declaração `public property FieldAltered : Boolean read GetFieldAltered write _FieldAltered;`

Descrição A propriedade `FieldAltered` Indica que o campo foi alterado. Deve ser atualizado na visão caso a tabela esteja em modo de edição.

OkSpc

Declaração `public property OkSpc : Boolean read _OkSpc write SetOkSpc;`

OkMask

Declaração `public property OkMask : Boolean read _OkMask write _okMask;`

Descrição O método `OkMask` é usado para habilitar ou não em GetString a mascara em campos numéricos.

AsString

Declaração `public property AsString : AnsiString read GetAsString write SetAsString;`

Value

Declaração `public property Value : Variant Read GetValue write SetValue;`

FldOrigin_Y

Declaração `public property FldOrigin_Y: Integer Read GetFldOrigin_Y Write _FldOrigin_Y;`

FldOrigin

Declaração `public property FldOrigin : TPoint read getFldOrigin;`

vidis_OnEnter

Declaração `public property vidis_OnEnter: Boolean Read Getvidis_OnEnter Write Setvidis_OnEnter;`

Descrição A propriedade `vidis_OnEnter` usado para evitar reentrância do evento `DoOnEnter`(**??**)()

vidis_OnExit

Declaração `public property vidis_OnExit: Boolean Read Getvidis_OnExit Write Setvidis_OnExit;`

Descrição A propriedade `vidis_OnExit` é usado para evitar reentrância do evento `DoOnExit`(**??**)()

Campos

LinkEdit

Declaração `public LinkEdit: TComponent;`

Descrição Componente corrente que está editando esse campo.

Alias

Declaração `public Alias: AnsiString;`

Descrição O campo `Alias` é usado para associar label ao corrente campo.

- **NOTA**
  - Esse campo foi necessário para implementar campos do tipo boolean [X] por que o mesmo sempre vem associado a um rótulos amigável e o controle checkbox precisa dele.
- **EXEMPLO**
  - `Template`(**??**) de um botão checkbox:

Resourcestring

```
tmp_Aceita = '\X Aceita o contrato +ChFN+'Aceita_contrato'+CharHi
```

Template_org

Declaração `public Template_org: AnsiString;`

Descrição O campo `Template_org` guarda o modelo original do `Template`(**??**) e deve ser inicializado em CreateStruct

Next

Declaração `public Next: pDmxFieldRec;`

Descrição Próximo campo

RSelf

Declaração `public RSelf: pDmxFieldRec;`

Descrição Usado para referenciar-se a si mesmo.

Prev

Declaração `public Prev: pDmxFieldRec;`

Descrição Campo anterior

access

Declaração `public access: byte;`

Descrição read-only, hidden, skip, accSpecX

Fieldnum

Declaração `public Fieldnum: Integer;`

Descrição Número do campo, varia de 1 a totalFields (Se zero (0) é porque trata-se um rótulos)

ScreenTab

Declaração `public ScreenTab: integer;`

Descrição Override column num.

ColumnWid

Declaração `public ColumnWid: byte;`

Descrição width of Field column

ShownWid

Declaração `public ShownWid: byte;`

Descrição visible width of column

TypeCode

Declaração `public TypeCode: AnsiChar;`

Descrição 's', 'r', etc.

FldEnum_Lookup

Declaração `public FldEnum_Lookup:TFldEnum_Lookup;`

FillValue

Declaração `public FillValue: AnsiChar;`

Descrição If the Field is numeric, fill in with '#0' if it's alphanumeric, fill in with ' '

UpperLimit

Declaração `public UpperLimit: byte;`

Descrição maximum `value`(**??**) limit

ShowZeroes

Declaração `public ShowZeroes: boolean;`

Descrição display zero values

TrueLen

Declaração `public TrueLen: byte;`

Descrição unformatted text length

Parenthesis

Declaração `public Parenthesis: boolean;`

Descrição '('/')' AnsiCharacters

Decimals

Declaração `public Decimals: byte;`

Descrição decimal point or cluster `value`(**??**)

FieldSize

Declaração `public FieldSize: integer;`

Descrição sizeof (datatype)

DataTab

Declaração `public DataTab: integer;`

Descrição position in record

Template

Declaração `public Template: ptString;`

Descrição Field `Template`

ListComboBox

Declaração `public ListComboBox: PSItem;`

Descrição O atributo `ListComboBox` contém uma lista de opções possíveis para o campo.

- Nota:
  - Após caractere **CharListComboBox** contém um ponteiro para uma lista de opções do mesmo tipo de campo.
    - Exemplo:

```
Const
  ' Dia de vencimento: \Ssssss'+ChFN+'Dia'+CreateOptio
     NewSItem('Dia 10',
     NewSItem('Dia 15',
     NewSItem('Dia 20',
     NewSItem('Dia 25',
              nil)))));
```

ListComboBox_Default

Declaração `public ListComboBox_Default: Longint;`

Descrição O Atributo `ListComboBox_Default`* é usado guardar o valor padrão para a lista do BomboBox ou LookupBox

- Exemplo para selecionar "Dia 20" da lista.

– O número **2** representa o terceiro item da lista.

```
Const
  ' Vencimento: \Ssssss'+ChFN+'Dia'+CreateOptions(accNormal, 2,
     NewSItem('Dia 10',
     NewSItem('Dia 15',
     NewSItem('Dia 20',
     NewSItem('Dia 25',
              nil)))));
```

ExecAction

Declaração `public ExecAction: AnsiString;`

Descrição O campo `ExecAction` é inicializado no interpretador de `Template`(??) quando o caractere `CharExecAction`(??) é encontrado.

- **EXEMPLO DE USO DE AÇÕES NO** `TEMPLATE`(??)

1. Se o atributo `Fieldnum`(??) do campo for diferente de zero, então o **rótulo** do botão associado a ação será o caracteres e a ação pode atualizar o buffer do campo.

   – No exemplo a seguir a função CreateExecAction retorna a string `chFN`(??)+**aFieldName+' '+ChEA+(aFieldNa**

   – O interpretador de `Template`(??) atualiza a string `LinkExecAction`(??) caso o o ponto seja encontrado no `ExecAction` do Label.

```
Result := NewSItem(' Cliente: '+'\LLLLL'+CreateExecActi
```

2. Se o atributo `Fieldnum`(??) do campo for igual a zero, então a rótulo do botão será o rótulo do campo.

   – No exemplo a seguir um rótulo de novo cliente (icons) e um botão ok (icons)

```
NewSItem('   &Novo cliente: '+CharExecAction+Action_Nov
         '         '+CharExecAction+Action_Ok.name)
```

LinkExecAction

Declaração `public LinkExecAction: pDmxFieldRec;`

Descrição O atributo `LinkExecAction` é atualizado com o ponteiro do campo passado por `execAction(??)`.

- O Interpretador de `Template(??)` deve pegar o campo usando a função FieldByName(aFieldName passado em `execAction(??)`), quando `execAction(??)` tiver um ponto antes do nome da ação.
  - Ex: **(aFieldName.aExecAction)**.

```
Result := NewSItem(' Cliente: '+'\LLLLL'+CreateExecAction('Cliente',Pesquisa.Name),nil);
```

CharShowPassword

Declaração `public CharShowPassword: AnsiChar;`

_DateMask

Declaração `public _DateMask: TDates.TDateMask;`

_HourMask

Declaração `public _HourMask: TDates.THourMask;`

QuitFieldAltomatic

Declaração `public QuitFieldAltomatic: Boolean;`

CurPos

Declaração `public CurPos: integer;`

Descrição Posição do curso quando este campo estiver sendo editado.

SelStart

Declaração `public SelStart: Integer;`

Descrição Posição do início da seleção quando este campo estiver sendo editado.

SelEnd

Declaração `public SelEnd: Integer;`

Descrição Posição do fim da seleção quando este campo estiver sendo editado.

_FieldAltered

Declaração `public _FieldAltered: Boolean;`

HelpCtx_Hint

Declaração `public HelpCtx_Hint: AnsiString;`

Descrição O campo `HelpCtx_Hint` contém a documentação resumida do registro.

HelpCtx_Porque

Declaração `public HelpCtx_Porque: AnsiString;`

Descrição Por que preciso deste campo?

HelpCtx_Onde

Declaração `public HelpCtx_Onde: AnsiString;`

Descrição Onde esse campo será usado?

_OkSpcAnt

Declaração `public _OkSpcAnt: Boolean;`

Descrição Salva o valor de _OkSpc antes de setar com aOkSpc

ProviderFlags

Declaração `public ProviderFlags: TUiTypes.TMiProviderFlags;`

Descrição O atributo `ProviderFlags` é usado nos métodos **TUiDmxScroller_sql.CreateTables** e **TUiDmxScroller_sql.CreateBufDataset_FieldDefs** para integração do componente **TDmxScroller** com o componente TSqlDbConnector.

ForeignKey

Declaração `public ForeignKey: TuiTypes.TForeignKey;`

Descrição O atributo `ForeignKey` é usado para criar chave estrangeira e os relacionamentos

KeyForeign

Declaração `public KeyForeign: AnsiString;`

Descrição O atributo `KeyForeign` contém uma string com o nome da tabela estrangeira e a lista de campos relacionados

- **EXEMPLO**
  - CIDADES,ESTADO;CIDADE
    - CIDADES = tabela estrangeira
    - ESTADO = Estado da cidade.
    - CIDADE = Cidade do estado.

Métodos

StrNumberValid

Declaração `public function StrNumberValid(S: AnsiString):AnsiString;`

GetAsStringFromBuffer

Declaração `public Function GetAsStringFromBuffer(aWorkingData : pointer):AnsiString;`

SetAsString

Declaração `public Procedure SetAsString(S:AnsiString);`

GetAsString

Declaração `public Function GetAsString:AnsiString;`

IsInputText

Declaração `public Function IsInputText:Boolean;`

SItemsLen

Declaração `public function SItemsLen(S: PSItem) : SmallInt;`

MaxItemStrLen

Declaração `public function MaxItemStrLen(AItems: PSItem) : integer;`

GetMaxLength

Declaração `public Function GetMaxLength():integer;`

IsStaticText

Declaração `public function IsStaticText:Boolean;`

IsInputRadio

Declaração `public function IsInputRadio:Boolean;`

IsInputDbRadio

Declaração `public function IsInputDbRadio:Boolean;`

IsInputCheckbox

Declaração `public function IsInputCheckbox:Boolean;`

isInputPassword

Declaração `public function isInputPassword:Boolean;`

IsInputHidden

Declaração `public function IsInputHidden:Boolean;`

IsSelect

Declaração `public function IsSelect:Boolean;`

Descrição O objeto filho que implementar um ISelect deve anular e retornar a interface ISelect;

IsComboBox

Declaração `public function IsComboBox:Boolean;`

Descrição Usado quando trata-se de campos enumerados.

FirstField

Declaração `public function FirstField:  pDmxFieldRec;`

LastField

Declaração `public function LastField:  pDmxFieldRec;`

NextField

Declaração `public function NextField:  pDmxFieldRec;`

PrevField

Declaração `public function PrevField:  pDmxFieldRec;`

SelectFirstField

Declaração `public Function SelectFirstField:  pDmxFieldRec;`

SelectLastField

Declaração `public Function SelectLastField:  pDmxFieldRec;`

Select

Declaração `public Procedure Select;`

GetCount_Cluster

Declaração `public Function GetCount_Cluster:Integer;`

GetValue_Cluster

Declaração `public Function GetValue_Cluster(aItem:  Integer):AnsiString;`

SetValue_Cluster

Declaração `public Procedure SetValue_Cluster(aItem:Integer;wValue:AnsiString);`

GetChecked_Cluster

Declaração `public Function GetChecked_Cluster( aItem:  Integer):Boolean;`

SetChecked_Cluster

Declaração `public Procedure SetChecked_Cluster( aItem :`
`Integer;aValue:Boolean);`

GetCount_InputRadio

Declaração `public Function GetCount_InputRadio:Integer;`

GetValue_InputRadio

Declaração `public Function GetValue_InputRadio(aItem:  Integer):AnsiString;`

SetValue_InputRadio

Declaração `public Procedure`
`SetValue_InputRadio(aItem:Integer;aValue:AnsiString);`

GetChecked_InputRadio

Declaração `public Function GetChecked_InputRadio( aItem:  Integer):Boolean;`

SetChecked_InputRadio

Declaração `public Procedure SetChecked_InputRadio( aItem : Integer;aValue:Boolean);`

get_Item_Focused_InputRadio

Declaração `public Function get_Item_Focused_InputRadio:Longint;`

GetCount_InputCheckbox

Declaração `public Function GetCount_InputCheckbox:Integer;`

Descrição Construção da propriedade Count

- Objetivo: Retorna o numero de items da lista onde os itens devem ser acessados com index 0 a count-1

GetValue_InputCheckbox

Declaração `public Function GetValue_InputCheckbox(aItem:  Integer):AnsiString;`

Descrição Construção da propriedade `Value(??)`

- Objetivo: Ler o label associado a opção ou trocar seu valor.
- Sintaxe: Setando = `Value(??)`[1] = 'Sim'; `Value(??)`[2] = 'Nao'; `Value(??)`[1] = 'Yes' Lendo = If LowerCase(`Value(??)`[1]) = 'SIM' Then;

SetValue_InputCheckbox

Declaração `public Procedure SetValue_InputCheckbox(aItem: Integer;aValue:AnsiString);`

GetChecked_InputCheckbox

Declaração `public Function GetChecked_InputCheckbox( aItem: Integer):Boolean;`

Descrição Construção da propriedade Checked - Sintaxe: 1 = If Checked[1] then; 2 = Checked[1] := True.

- Objetivo: Selecionar um item da lista de opções ou checar se a opção está selecionada

SetChecked_InputCheckbox

Declaração `public Procedure SetChecked_InputCheckbox( aItem : Integer;aValue:Boolean);`

GetCount_Select

Declaração `public Function GetCount_Select:Variant;`

Descrição Construção da propriedade Count de campos enumerados

- Objetivo: Retorna o numero de items da lista onde os itens devem ser acessados com index 0 a count-1

GetSize_Select

Declaração `public Function GetSize_Select():Variant;`

Descrição Número de Linhas a ser mostrada no box. Usado em campos enumerados.

GetValue_Select

Declaração `public Function GetValue_Select(aItem: Integer):AnsiString;`

Descrição Construção da propriedade `Value(??)`

- Objetivo: Ler o label associado a opção ou trocar seu valor.
- Sintaxe: Setando = `Value(??)`[1] = 'Sim'; `Value(??)`[2] = 'Nao'; `Value(??)`[1] = 'Yes' Lendo = If LowerCase(`Value(??)`[1]) = 'SIM' Then;

SetValue_Select

Declaração `public Procedure SetValue_Select(aItem: Integer;aValue:AnsiString);`

GetChecked_Select

Declaração `public Function GetChecked_Select( aItem:  Integer):Boolean;`

Descrição Construção da propriedade Checked - Sintaxe: 1 = If Checked[1] then; 2 = Checked[1] := True.

- Objetivo: Selecionar um item da lista de opções ou checar se a opção está selecionada

SetChecked_Select

Declaração `public Procedure SetChecked_Select( aItem : Integer;aValue:Boolean);`

IsNumber

Declaração `public Function IsNumber:Boolean;`

Descrição O método `IsNumber` retorna true se o campo é numérico e false se alfanumérico

IsNumberReal

Declaração `public Function IsNumberReal:Boolean;`

IsNumberInteger

Declaração `public Function IsNumberInteger:Boolean;`

IsData

Declaração `public function IsData:  Boolean;`

IsHora

Declaração `public function IsHora:  Boolean;`

GetLeft

Declaração `public Function GetLeft:Integer;`

GetTop

Declaração `public Function GetTop:Integer;`

GetWidth

Declaração `public Function GetWidth:Integer;`

GetHeight

Declaração `public Function GetHeight:Integer;`

SetAccess

Declaração `public Function SetAccess(aaccess :  byte):Byte;`

Valid

Declaração `public function Valid(Command:  Word):  Boolean;`

DoOnEnter

Declaração `public procedure DoOnEnter(Sender: TObject);`

Descrição O método `DoOnEnter` é executado toda vez antes do controle ler do buffer do campo.

- Se o `TUiDmxScroller.OnEnterField`(**??**) tiver assinalado o método `DoOnEnter` o executa.

DoOnExit

Declaração `public procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` é executado toda vez antes do controle gravar no buffer do campo.

- Se o `TUiDmxScroller.OnExitField`(**??**) tiver assinalado o método `DoOnExit` o executa.

TUiDmxScroller Classe

Hierarquia

TUiDmxScroller > `TUiMethods`(**??**) > TUiConsts

Descrição

A classe `TUiDmxScroller` tem como objetivo criar um formulário baseado em uma lista do tipo ShortString.

- **NOTAS**
  - O método `createStruct`(**??**) criar uma lista de campo tipo `TDmxFieldRec`(**??**) com todas as informações necessárias para criar uma tabela ou um formulário.
  - O formulário é criado com apena uma linha.
- **EXEMPLO**:
  - Template := ' Nome \SSSSSSSSSSSSSSSSSSS Idade: \BB'
    - A classe cria a lista de campos:
      - Label1: Nome
      - Field1: campo ShortString com 20 posições maiúsculas
      - Label2: Idade
      - Field2: Campo byte com duas posições

Propriedades

CurrentRecord

Declaração `public property CurrentRecord : Longint read _CurrentRecord write SetCurrentRecord;`

Strings

Declaração `published property Strings : TMiStringList Read GetStrings Write SetStrings;`

Descrição A propriedade `Strings` o `Strings` é usada para editar o Template em tempo de projeto.

TableName

Declaração `public property TableName : String Read _TableName Write SetTableName;`

Descrição A propriedade `TableName` contém o nome da tabela ou consulta no banco de dados.

- **NOTA**
  - A propriedade `TableName` é usado no método **SetSqlBufDataset** para criação da propriedade **TCustomSQLQuery.SQL** e no método **AlterTable** para criação da tabela ou consulta no banco de dados.

Appending

Declaração `public property Appending : Boolean read GetAppending write SetAppending;`

Descrição A propriedade `Appending` é usada para saber se está editando um novo registro ou não

- **NOTA**
  - TRUE = Indica que um novo registro esta sendo editado.
  - False = Indica que um registro existente está sendo editado ou visualizado.
  - Obs: Deve ser atualizado na visão caso a tabela está em edição.

DoOnNewRecord_FillChar

Declaração `public property DoOnNewRecord_FillChar : Boolean Read _DoOnNewRecord_FillChar Write SetDoOnNewRecord_FillChar default True;`

RecordSelected

Declaração `protected property RecordSelected : boolean read GetRecordSelected Write SetRecordSelected default false;`

OnNewRecord

Declaração `public property OnNewRecord : TOnNewRecord read _OnNewRecord Write _OnNewRecord;`

Descrição A propriedade `OnNewRecord` é executada em `DoOnNewRecord`(**??**) se a mesma for assinalada.

onCloseQuery

Declaração `public property onCloseQuery : TOnCloseQuery Read _OnCloseQuery write _onCloseQuery;`

Descrição O evento `onCloseQuery` é disparado toda vez que o `TUiDmxScroller`(**??**) é desativado.

- **NOTA***
  - **Este evento é disparado antes de desativar a classe **`TUiDmxScroller`(**??**)**
  .
  - Obs: Se o parâmetro **CanClose** for **false**, então o formulário não é desativado.

onEnter

Declaração `public property onEnter : TOnEnter Read _OnEnter write _onEnter;`

Descrição O evento `onEnter` é disparado toda vez que o `TUiDmxScroller`(**??**) ativado.

onExit

Declaração `public property onExit : TOnExit Read _OnExit write _onExit;`

Descrição O evento `onExit` é disparado toda vez que o `TUiDmxScroller`(**??**) é destivado.

onEnterField

Declaração `public property onEnterField : TOnEnterField Read _OnEnterField write _onEnterField;`

Descrição O evento `onEnterField` é disparado toda vez que o controle corrente recebe o foco.

onExitField

Declaração `public property onExitField : TOnExitField Read _OnExitField write _onExitField;`

Descrição O evento `onExitField` é disparado toda vez que o controle corrente perde o foco.

onGetTemplate

Declaração `public property onGetTemplate : TonGetTemplate Read _onGetTemplate Write _onGetTemplate;`

Descrição O evento `onGetTemplate` substitui o método `getTemplate`(**??**) caso `OnGetTemplate`<>nil

onAddTemplate

Declaração `public property onAddTemplate: TonAddTemplate read _onAddTemplate write _onAddTemplate;`

Active

Declaração `public property Active : Boolean Read _Active Write SetActive;`

CurrentField

Declaração `public property CurrentField : pDmxFieldRec read _CurrentField write SetCurrentField;`

Descrição A propriedade `CurrentField` contem um ponteiro para o campo selecionado

AlignmentLabels

Declaração `public property AlignmentLabels : TAlignment read _AlignmentLabels write _AlignmentLabels;`

BufDataset

Declaração `protected property BufDataset : TBufDataset read GetBufDataset write SetBufDataset;`

Descrição A propriedade `BufDataset` é usada com objetivo de integração dos dados do formulário TVDmx e os controle decentes de TDataSet.

DataSource

Declaração `public property DataSource : TDataSource Read _DataSource Write _DataSource;`

Descrição A propriedade `DataSource` permite que controles da **LCL** (Lazarus Componentes Library) possam usar os dados do componente **TDmxScroller**.

- **NOTA**
  - Essa integração permite que **TDmxScroller** utilize todos os componentes de banco de dados do Free Pascal.

Locked

Declaração `public property Locked : Boolean read _Locked write SetLocked;`

Descrição A propriedade `Locked` deve ser redefinida na interface filha desta classe para travar o formulário se aLocked = true e destravar se aLocked = false;

Campos

Fields

Declaração `public Fields: TFPList;`

Descrição O atributo `Fields` contém uma lista `pDmxFieldRec`(**??**) cujo **Fieldnum<>0**.

- Essa lista é atualizada em `createStruct`(**??**)

Limit

Declaração `public Limit: TPoint;`

CreateValid

Declaração `public CreateValid: boolean ;`

Descrição Deve ser true ao criar a classe

WorkingData

Declaração `protected WorkingData: pointer;`

WorkingDataOld

Declaração `protected WorkingDataOld: pointer;`

DataBlockSize

Declaração `protected DataBlockSize: longint;`

ActualRecordNum

Declaração `public ActualRecordNum: longint;`

DMXField1

Declaração `public DMXField1: pDmxFieldRec;`

Descrição O atributo `DMXField1` contém o primeiro campo da lista encandeada

TotalFields

Declaração `public TotalFields: integer;`

Descrição O atributo `TotalFields` contém o total de campos da lista apontada por `DMXField1`(**??**)

RecordSize

Declaração `public RecordSize: integer;`

Descrição O atributo `RecordSize` contém o tamanho do buffer calculado por `CreateStruct`(**??**)

FieldData

Declaração `public FieldData: pointer;`

Descrição O atributo `FieldData` contém o ponteiro do buffer do corrente campo calculado pela propriedade `CurrentField`(**??**)

RecordData

Declaração `public RecordData: pointer;`

WidthChar

Declaração `public WidthChar:byte;`

Descrição O atributo `WidthChar` deve ser iniciado quando este controle for incluído em um TScrolling-WinControl. em um controle gráfico.

- **EXEMPLO**
  - `WidthChar` := ((Owner as TScrollingWinControl).Canvas.TextWidth(UiDmxScrol
    div Length(UiDmxScroller.CharAlfanumeric));

HeightChar

Declaração `public HeightChar:byte;`

Descrição O atributo `HeightChar` deve ser iniciado quando este controle for incluído em um TScrolling-WinControl.

- **EXEMPLO**
  - `HeightChar` := (Owner as TScrollingWinControl).Canvas.TextHeight(CharAlfanu

KeyAltered

Declaração `protected KeyAltered: Boolean ;`

Descrição O atributo `KeyAltered` indica se algum campo da chave foi alterado é setado em changeMade

keysPrimaryKeyComposite

Declaração `protected keysPrimaryKeyComposite: AnsiString;`

Descrição O atributo `keysPrimaryKeyComposite` contém a lista de campos que pertence a chave primária

- **EXEMPLOS**:
  - **Chave simples**:
    - 'Matricula'.
  - **Chave composta**:
    - 'Estado,Cidade'.
- **NOTA**
  - Se pos(',',`keysPrimaryKeyComposite`) <> 0 indica que a chave é composta.

flagPrimaryKey_AutoIncrement

Declaração `protected flagPrimaryKey_AutoIncrement:Boolean;`

_OnCloseQuery

Declaração `protected _OnCloseQuery: TOnCloseQuery;`

_OnEnter

Declaração `protected _OnEnter: TOnEnter;`

_OnExit

Declaração `protected _OnExit: TOnExit;`

_OnEnterField

Declaração `protected _OnEnterField: TOnEnterField;`

_OnExitField

Declaração `protected _OnExitField: TOnExitField;`

_Active

Declaração `protected var _Active: Boolean;`

Descrição O atributo **name** é usado para criar a propriedade `active`(**??**) do componente

- NOTAS
  - O componente só pode estar ativo se o `GetTemplate`(**??**) <> nil .
  - O método **CreateFormLCL** só pode ser executado uma vêz.
  - Caso o `active`(**??**) esteja **true** e o usuário seta para false o controle Owner deve ficar invisível.

_CurrentField

Declaração `protected _CurrentField: pDmxFieldRec;`

_BufDataset

Declaração `protected _BufDataset: TBufDataset;`

_DataSource

Declaração `protected _DataSource: TDataSource;`

_Locked

Declaração `public _Locked: Boolean;`

Métodos

SetHelpCtx_Hint

Declaração
```
public Function
SetHelpCtx_Hint(aFldNum:Integer;a_HelpCtx_Hint:AnsiString):pDmxFieldRec;
virtual; overload;
```

Descrição O método `SetHelpCtx_Hint` inicia a documentação resumida do campo. aFldNum

SetHelpCtx_Hint

Declaração
```
public Procedure
SetHelpCtx_Hint(apDmxFieldRec:pDmxFieldRec;a_HelpCtx_Hint:AnsiString);
virtual; overload;
```

Descrição O método `SetHelpCtx_Hint` inicia a documentação resumida do campo passado em :apDmx-FieldRec

SetCurrentRecord

Declaração `protected Procedure SetCurrentRecord(aCurrentRecord : Longint ); Virtual;`

ShowControlState

Declaração `protected procedure ShowControlState; Virtual;`

UpdateBuffers

Declaração `public Procedure UpdateBuffers; Virtual;`

Refresh

Declaração `public procedure Refresh; VIRTUAL;`

GetAppending

Declaração `protected Function GetAppending:Boolean; VIRTUAL;`

SetAppending

Declaração `protected Procedure SetAppending(aAppending:Boolean); VIRTUAL;`

SetOnCalcRecord

Declaração `protected Function SetOnCalcRecord(Const WOnCalcRecordEnable:Boolean):Boolean;`

GetRecordSelected

Declaração `protected Function GetRecordSelected: boolean; Virtual;`

SetRecordSelected

Declaração `protected Procedure SetRecordSelected(a_RecordSelected : boolean); Virtual;`

ChangeMadeOnOff

Declaração `protected procedure ChangeMadeOnOff(const aValue:Boolean);`

Descrição O método `ChangeMadeOnOff` seta os atributos indicativos de que o objeto foi alterado ou não.

DoOnNewRecord

Declaração `public Procedure DoOnNewRecord; overload; Virtual;`

Descrição O método `DoOnNewRecord` é usado para inicializa os parâmetros de um novo registro

SetState

Declaração `protected Function SetState(Const AState: Int64; Const Enable: boolean):Boolean; virtual;`

Descrição O método `SetState` seta o bit passado no parâmetro aState e retorna o estado anterior do mapa de bits passado por aState

GetState

Declaração `public function GetState(Const AState: Int64): Boolean; Virtual;`

Descrição O Método `GetState` recebe um mapa de bits e retorna:

- false : Se o bite estiver desligado;
- true ; Se o bit estiver ligado

FieldByName

Declaração `public function FieldByName(aName:String):PDmxFieldRec;`

Descrição O método `FieldByName` retorna o campo passado por aName.

FieldByNumber

Declaração `public function FieldByNumber(aFieldNum:Integer):PDmxFieldRec;`

CancelBuffers

Declaração `public function CancelBuffers: Boolean;`

Descrição O método `CancelBuffers` copia o buffer do registro anterior para o buffer do registro atual

GetBuffers

Declaração `protected function GetBuffers:Boolean; Virtual;`

Descrição O método `GetBuffers` copia o buffer do registro atual para o buffer do registro anterior

- **OBSERVAÇÃO:**
  - O método `GetBuffers` deve ser anulado para ler o buffer dos campos dos arquivos associados a classe `TUiDmxScroller(??)` para o buffer dos campos da classe `TUiDmxScroller(??)`

PutBuffers

Declaração `protected function PutBuffers:Boolean; Virtual;`

Descrição O método `PutBuffers` deve ser anulado para grava o buffer dos campos da classe `TUiDmxScroller(??)` para o buffer dos campos dos arquivos associados a classe `TUiDmxScroller(??)`

DoOnCloseQuery

Declaração `public Procedure DoOnCloseQuery(aDmxScroller:TUiDmxScroller ; var CanClose:boolean ); overload;`

DoOnCloseQuery

Declaração `public Procedure DoOnCloseQuery(var CanClose:boolean ); overload;`

Scroll_it_inview

Declaração `public procedure Scroll_it_inview(AControl:  pDmxFieldRec); virtual;`

Descrição O método `Scroll_it_inview` é usado para da o scroller na janela onde esse componente for inserido.

- **NOTA**
  - A LCL não rola a tela com a tecla tab e o controle não estiver visível.

DoOnEnter

Declaração `public Procedure DoOnEnter(aDmxScroller:TUiDmxScroller); Virtual;`

Descrição O método `DoOnEnter` Executa o evento `onEnter`(**??**) se o mesmo estiver assinalado.

DoOnExit

Declaração `public Procedure DoOnExit(aDmxScroller:TUiDmxScroller);`

BeforeDestruction

Declaração `public procedure BeforeDestruction; override;`

Descrição Executado antes de construir o componente

Create

Declaração `public constructor Create(aOwner:TComponent); Override;`

Descrição Constrói o componente

destroy

Declaração `public destructor destroy; override;`

Descrição Destrói o componente

CreateStruct

Declaração `protected procedure CreateStruct(var ATemplate : TString); virtual; overload;`

Descrição A procedure `CreateStruct` é executado para construir a lista apontada por `DMXField1`(??) baseado no Template do tipo `TString`(??).

- **NOTA**
  - O parâmetro aTemplate é um string com 255 caracteres, porém o mesmo pode ser encandeado usando a função `CreateAppendFields`(??).
  - A função `CreateAppendFields`(??) retorna a constante `fldAPPEND`(??) mais o endereço da string a ser concatenada.
    - **EXEMPLO**

```
Var
  S1,s2,Template : TString;
begin
  S1 := ' Nome do Aluno....: \sssssssssssssssssssssss
  s2 := ' Endereço do aluno: \sssssssssssssssssssssss
  Template := S1+CreateAppendFields(s2);
end;
```

CreateStruct

Declaração `protected procedure CreateStruct(var ATemplate : PSItem); virtual; overload;`

Descrição A procedure `CreateStruct` é executado para construir a lista apontada por `DMXField1`(**??**) baseado na lista `PSItem`(**??**).

- **NOTA**
  - O parâmetro aTemplate é uma lista `PSitem`(**??**).
  - A função `CreateTSItemFields`(**??**) retorna uma lista de `PSItem`(**??**).
    - **EXEMPLO**

```
Var
  Template : PSItem;
begin
  Template := CreateTSItemFields(
                          NewSItem('ccccccccccccccccccccc
                          NewSItem('ccccccccccccccccccccc
                          NewSItem('ccccccccccccccccccccc
                          NewSItem('ccccccccccccccccccccc
                          NewSItem('ccccccccccccccccccccc
end;
```

CreateStruct

Declaração `protected procedure CreateStruct(); virtual; overload;`

Descrição A procedure `CreateStruct` interpreta o Template obtido em `getTemplate`(**??**) e cria a lista de `pDmxFieldRec`(**??**) associada ao Template.

- **Nota**
  - O Template pode ser obtido pela propriedade Template se Template<> " ou retornado pelo evento `onGetTemplate`(**??**).

DestroyStruct

Declaração `protected procedure DestroyStruct; virtual;`

Descrição A procedure `DestroyStruct` destrói as lista criada por `CreateStruct`(**??**) acima.

CreateBufDataset_FieldDefs

Declaração `protected Procedure CreateBufDataset_FieldDefs; virtual;`

Descrição O método `CreateBufDataset_FieldDefs` é usado para criar os campos de `BufDataset`**(??)**

CreateData

Declaração `protected procedure CreateData; Virtual;`

Descrição A procedure `CreateData` é usada para alocar (`RecordSize`(**??**)) memória para o buffer (`WorkingData`(**??**)) do registro calculado por `createStruct`(**??**)

DestroyData

Declaração `protected procedure DestroyData; virtual;`

Descrição A procedure `DestroyData` é usada para desalocar memória do buffer do registro criado por `CreateData`(**??**)

GetRecordData

Declaração `public Function GetRecordData:  Pointer; virtual;`

Descrição A função `GetRecordData` retorna o atributo `WorkingData`(**??**)

SetLimit

Declaração `protected Procedure SetLimit(X, Y: Integer); virtual;`

GetTemplate

Declaração `protected function GetTemplate(aNext: PSItem) : PSItem; overload; virtual;`

Descrição O método `GetTemplate` é usado para atualizar o atributo _onGetTemplate com o modelo informado pelo usuário caso o `onGetTemplate`(**??**) seja nil.

- **NOTA**

  1. O Evento _onGetTemplate só é iniciado em tempo de execução por isso o formulário não pode ser criado em tempo de projeto usando o evento `onGetTemplate`(**??**).
  2. As `strings`(**??**) do formulário também pode ser desenhado usando o evento `OnAddTemplate`(**??**).
  3. O evento `OnGetTemplate`(**??**) tem prioridade em relação ao evento `OnAddTemplate`(**??**);

SetActive

Declaração `protected procedure SetActive(aActive : Boolean); virtual;`

Descrição A procedure `SetActive` seta a propriedade `active`(**??**) e criar um formulário LCL ou HTML dependendo do tipo de aplicação

SetCurrentField

Declaração `protected Procedure SetCurrentField(aCurrentField : pDmxFieldRec);`

PutString

Declaração `public Function PutString(Const OkSpc:Boolean;Const S:tString) : SmallInt; virtual; overload;`

Descrição A função `PutString` salva a string S no `currentField`(**??**)

- **PARÂMETROS**
  - OkSpc : campo lógico e se **true** salva o campo preenchendo com espaço para completar a máscara.
  - S : String do tipo ShortString com conteúdo do campo.

PutString

Declaração `public function PutString(Const aFieldName:tString;S : ShortString):SmallInt; virtual; overload;`

Descrição O método `PutString` salva um string no campo passado por aFieldName.

GetString

Declaração `public function GetString(const aFieldName: tString):AnsiString; virtual; overload;`

Descrição O método `GetString` retorna um string do campo passado por aFieldName.

GetString

Declaração `public Function GetString(Const OkSpc:Boolean) : TString; virtual; overload;`

Descrição A função `GetString` retorna a string com o valor do `currentField`(**??**)

- **PARÂMETROS**
  - OkSpc : campo lógico e se **true** retorna o campo preenchendo com espaço para completar a máscara.

GetString

Declaração `public Function GetString: TString; virtual; overload;`

Descrição A função `GetString` retorna a string com o valor do `currentField`(**??**) sem preencher com espaço para completar a máscara.

PutString

Declaração `public function PutString(const S : ShortString):SmallInt; virtual; overload;`

Descrição A função `PutString` salva a string **S** no `currentField`(**??**) usando **okspc** = false;

- **PARÂMETROS**
  - S : String do tipo ShortString com conteúdo do campo.

Get_MaskEdit_LCL

Declaração `public Function Get_MaskEdit_LCL(aTemplate : ShortString; out Size_TypeFld, aLength_Buffer : SmallWord; out OkMask : Boolean) : AnsiString; overload;`

Descrição O método `Get_MaskEdit_LCL` receber a máscara do DmxScroller e retorna a mascara do componente LCL.

- **Nota**
  - Em **Size_TypeFld** retorno o tamanho do tipo de dados da mascara;
  - Em **OkMask** retorna **true** se tiver mascara e **false** caso contrário

Get_MaskEdit_LCL

Declaração `public Function Get_MaskEdit_LCL(aTemplate : ShortString; out OkMask : Boolean) : AnsiString; overload;`

Descrição O método `Get_MaskEdit_LCL` receber a máscara do Dmx e retorna a mascara do componente LCL.

DoAddRec

Declaração `public Function DoAddRec:Boolean; virtual;`

IfEqual

Declaração `public Function IfEqual(Const Ofset_Inicial:Word;Const PAnt,PAtu : Pointer; Const Len:Word):Boolean;`

Descrição O atributo `IfEqual` retorna true se o buffer apontado por PAnt for igual ao buffer apontado por PAtu.

RecordAltered

Declaração `public function RecordAltered: Boolean ;`

Descrição O método `RecordAltered` retorna true se o registro atual for diferente do registro anterior

CreateExecAction

Declaração `public class function CreateExecAction(Const aFieldName:AnsiString;const aExecAction: AnsiString) : AnsiString;`

Descrição A classe método `CreateExecAction` é usado para adicionar a chamada de um procedimento quando a tecla F7 é pressionada;

add

Declaração `public procedure add(aTemplate:AnsiString);`

SetLocked

Declaração `protected Procedure SetLocked(aLocked:Boolean); Virtual;`

44.5 Tipos

PSItem

Declaração `PSItem = TUiMethods.PSItem;`

tString

Declaração `tString = TUiMethods.tString;`

ptString

Declaração `ptString = TUiMethods.Ptstring;`

TDates

Declaração `TDates = TUiMethods.TDates;`

PValue

Declaração `PValue = TUiMethods.PValue;`

TValue

Declaração `TValue = TUiMethods.TValue;`

TOnGetTemplate

Declaração `TOnGetTemplate = function (aNext: PSItem) : PSItem of Object unimplemented;`

Descrição Usado para criar modelos de formulários dinamicamente usando como parâmetro listas de PSItems.

TOnAddTemplate

Declaração `TOnAddTemplate = Procedure(const aUiDmxScroller:TUiDmxScroller) of Object unimplemented;`

Descrição O tipo `TOnAddTemplate` é usado para criar modelos de formulários dinamicamente usando o método add

- **EXEMPLO**

```
Procedure AddTemplate(const aUiDmxScroller:TUiDmxScroller);
begin
  with aUiDmxScroller do
  begin
    add(' EXEMPLO DE TEMPLATE ');
    add('');
    add(' Alfanumérico maiúscula com 15 posições: \SSSSSSSSSSSSSS');
    add(' Alfanumérico maiúscula e minuscula com 30 posições: ');
    add('  \ssssssssssssssssssssssssssssssssssssss');
    add(' Alfanumérico com a primeira letra maiúscula: \Sssssssssssss');
    add(' Valor double.......: \RRR,RRR.RR');
    add(' Valor SmalInt......: \II,III');
    add(' Valor Byte.........: \BBB');
    add(' Valor Smallword....: \WW,WWW');
    add(' Sexo..............: '+ CreateEnumField(TRUE, accNormal, 0,
                                NewSItem(' indefinido ',
                                NewSItem(' Masculino',
                                NewSItem(' Feminino',
                                        nil)))));
    add(' Estado Civil              \KA Indefinido  '+chFN+'Sexo');
    add('  \X Casado?               \KA Masculino    ');
    add('  \X Pretende se divorciar? \KA Feminino    ');
    add('  \X Tens filhos?          ');
    add('');
  end;
end;

procedure TForm1.DmxScroller_Form1AddTemplate(const aUiDmxScroller: TUiDmxScrol
begin
  AddTemplate(aUiDmxScroller);
end;
```

pDmxFieldRec

Declaração `pDmxFieldRec = ^TDmxFieldRec;`

Descrição O tipo `pDmxFieldRec` aponta para o campo do tipo `TDmxFieldRec`(**??**)

TEndProc

Declaração `TEndProc = Procedure(Const AOwner:TUiDmxScroller; Const ADmxFieldRec:PDmxFieldRec);`

Descrição O tipo `TEndProc` é usado para fazer pesquisa genérica no banco de dados quando a tecla F7 é pressionada.

TOnEnter

Declaração `TOnEnter = Procedure(aDmxScroller:TUiDmxScroller) of Object;`

Descrição O tipo `TOnEnter` é usado para implementar evento onEnter da classe `TUiDmxScroller`(**??**)

TOnExit

Declaração `TOnExit = Procedure(aDmxScroller:TUiDmxScroller) of Object;`

Descrição O tipo `TOnExit` é usado para implementar evento onExit da classe `TUiDmxScroller`(**??**)

TOnNewRecord

Declaração `TOnNewRecord = Procedure(aDmxScroller:TUiDmxScroller) of Object;`

Descrição O tipo `TOnNewRecord` é usado para implementar evento onNewRecord da classe `TUiDmxScroller`(**??**)

TOnCloseQuery

Declaração `TOnCloseQuery = Procedure(aDmxScroller:TUiDmxScroller; var CanClose:boolean) of Object;`

Descrição O tipo `TOnCloseQuery` é usado para implementar evento OnCloseQuery da classe `TUiDmxScroller`(**??**)

- **NOTA***
  - **Este evento é disparado antes de desativar a classe **`TUiDmxScroller`(**??**)**
  .
  - Obs: Se o parâmetro **CanClose** for **false**, então a classe `TUiDmxScroller`(**??**) não é desativado.

TOnEnterField

Declaração `TOnEnterField = Procedure(aField:pDmxFieldRec) of Object;`

Descrição O tipo `TOnEnterField` é usado no evento OnEnterField

TOnExitField

Declaração `TOnExitField = Procedure(aField:pDmxFieldRec) of Object;`

Descrição O tipo `TOnExitField` é usado no evento OnExitField

SmallWord

Declaração `SmallWord = TUiDmxScroller.SmallWord;`

44.6 Constantes

AccNormal

Declaração `AccNormal = TUiMethods.AccNormal;`

LF

Declaração `LF = TConsts.LF;`

Chapter 45

Unit mi_rtl_ui_DmxScroller_Buttons

45.1 Descrição

A unit `mi_rtl_ui_DmxScroller_Buttons` implementa a classe `TUiDmxScroller_Buttons`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
  - **PENDÊNCIAS**
  - **CONCLUÍDO**
    - Criar classe `TUiDmxScroller_Buttons`(**??**)
    - Criar constructor Create.
    - Criar Function Create_RCommands.
    - Commands_Buttons_High : Byte;
    - Commands_Buttons : Array[0..Max_List_Buttons] of TRCommand;
    - Max_List_Buttons = sizeof(Longint);
    - Commands_Buttons_Mb : Longint;
    - Function Add_RCommands_Buttons;
    - Function Create_RCommands_Buttons;
    - Function Set_Commands_Buttons_Mb(Const aMb_Bits:Longint):Longint;

    * Documentar os atributos abaixo:
        · Commands_Buttons_High : Byte;
        · Commands_Buttons : Array[0..Max_List_Buttons] of TRCommand;

– **HISTÓRICOS**

    * **DIAS ANTERIORES**
        .
    * **DO DIA**
        · **2022-07-07**
        · **09:45**
        · Documentar o método **Add_RCommands_Buttons**
        · **14:55**
        · Documentar o método **Add_RCommands_Buttons**

45.2 Uses

- `Classes`
- `SysUtils`
- `mi_rtl_ui_methods`(**??**)
- `mi_rtl_ui_Dmxscroller`(**??**)

45.3 Visão Geral

`TUiDmxScroller_Buttons Classe`

45.4 Classes, Interfaces, Objetos e Registros

TUiDmxScroller_Buttons Classe

Hierarquia

TUiDmxScroller_Buttons > `TUiMethods`(**??**) > TUiConsts

Descrição

A classe `TUiDmxScroller_Buttons` tem como objetivo registrar os dados necessários para criar os botões de navegação e edição de uma tabela quando TDataSource for <> nil.

- **EXEMPLO USO**

Propriedades

Commands_Buttons_High

Declaração `public property Commands_Buttons_High : Byte Read _Commands_Buttons_High;`

Descrição A propriedade `Commands_Buttons_High` contém o número de linhas inicializadas da matriz `Commands_Buttons(??)`, ou seja: é igual o número de linhas criadas em: `Create_RCommands_Buttons(??)`.

Commands_Buttons_Mb

Declaração `public property Commands_Buttons_Mb : Longint read _Commands_Buttons_Mb;`

Descrição O atributo `Commands_Buttons_Mb` contém o mapa de bits dos botões que serão criados no formulário.

- **NOTA**
  - O mapa de bits é do tipo longint (4 bytes) por isso pode conter no máximo (4x8=32) botões.

Campos

UiDmxScroller

Declaração `public UiDmxScroller: TUiDmxScroller;`

Descrição O atributo `UiDmxScroller` deve ser passado em constructor `create(??)`

OkCmmNewRecord

Declaração `public Var OkCmmNewRecord: boolean;`

Descrição O atributo `OkCmmNewRecord` indica se o registro pode ser incluído ou não, ou seja: é o estado inicial da ação incluir informada pelo usuário.

- **NOTA**
  - True : O registro pode ser incluído. Obs: DataSet.Append pode ser executado.
  - False: O registro não pode ser incluído. Obs: DataSet.Append não pode ser executado.
  - Esse atributo é usado nos seguintes métodos:
    - Create_RCommands_Edit
    - No método DoOnNewRecord
    - Action Novo
- **EXEMPLO**

```
// Tirado do código: Function TRecord.Create_RCommands_Edit

OkCmmNewRecord := Application.FileOptions_CommandEnabled(Module,aCmNovo);
if aCmNovo<=255 then
  if OkCmmNewRecord
  then Application.EnableCommands([aCmNovo])
  Else Application.DisableCommands([aCmNovo]);
```

OkCmmDbLocaliza

Declaração `public Var OkCmmDbLocaliza: boolean;`

Descrição O atributo `OkCmmDbLocaliza` indica se o registro pode ser localizado ou não, ou seja: é o estado inicial da ação **pesquisar** informada pelo usuário.

- **NOTA**

- True : O registro pode ser pesquisado. DataSet.Locate pode ser executado.
- False: O registro não pode ser pesquisado. DataSet.Locate não pode ser executado.
- Esse atributo é usado nos seguintes métodos:
  - Create_RCommands_Edit
  - No método DoOnNewRecord
  - Action Pesquisa

- **EXEMPLO**

```
// Tirado do código: Function TRecord.Create_RCommands_Edit

if OkCmmNewRecord or OkCmmEvaluateRecord or OkCmmZeroizeRecord
then begin
       self.OkCmmDbLocaliza := true;
       self.Locked := false;
     end
else begin
        self.OkCmmDbLocaliza := Application.FileOptions_CommandEnabled(Module,A
        self.Locked := True;
      end;
```

OkCmmZeroizeRecord

Declaração `public Var OkCmmZeroizeRecord:  boolean;`

Descrição O atributo `OkCmmZeroizeRecord` indica se o registro pode ser excluído ou não, ou seja: é o estado inicial da ação excluir informada pelo usuário.

- **NOTA**
  - **True** : O registro pode ser deletado.
  - **False**: O registro não pode ser deletado.
  - Esse atributo é usado nos seguintes métodos:
    - Create_RCommands_Edit

* Nos métodos DeleteRec
* Action Delete

- **EXEMPLO**

```
// Tirado do código: Function TRecord.Create_RCommands_Edit

OkCmmZeroizeRecord  := Application.FileOptions_CommandEnabled(Module,ACmExclusa
if ACmExclusao<=255 then
  if OkCmmZeroizeRecord
  then Application.EnableCommands([ACmExclusao])
  Else Application.DisableCommands([ACmExclusao]);
```

Max_List_Buttons

Declaração `public const Max_List_Buttons = sizeof(Longint)*8;`

Descrição A constante `Max_List_Buttons` contém o número máximo de comandos da matriz `Commands_Buttons`(**??**)

Commands_Buttons

Declaração `public Commands_Buttons:  Array[0..Max_List_Buttons] of TRCommand;`

Descrição O atributo `Commands_Buttons` contém os dados necessários para criar os botões de ações da classe de acesso a arquivos.

- **EXEMPLO DE USO DESTA MATRIZ**

```
Commands_Buttons[1] := Create_RCommand(CmOk,'Ok'  ,'',KbEnter,AHelpCtx,Flag)^;
Commands_Buttons[2] := Create_RCommand(CmOk,'Next','',Kbno   ,AHelpCtx,Flag)^;
```

Métodos

Create

Declaração `public constructor Create(aOwner:TComponent); Override;`

Descrição O construtor `Create` é usado para iniciar o atributo **_UiDmxScroller** com o cast (aOwner as `TUiDmxScroller`(**??**))

Set_Commands_Buttons_Mb

Declaração `public Function Set_Commands_Buttons_Mb(Const aMb_Bits:Longint):Longint;`

Descrição O Método `Set_Commands_Buttons_Mb` seta _Commands_Buttons_Mb e retorna o mapa de bits `Commands_Buttons_Mb`(??) anterior.

EXEMPLO DE USO

```
//*** Seta as propriedades do fornecedor ***
With ArqFornecedor do
Begin
  Alias := sgc('Parâmetros para pesquisa de duplicatas');
  SetExpandable(False); //Não permite Inclusões
  SetLocked(False); //false = Não travado porque a janela filha pode ser alterada e expand
  SetOkWriteRec(False);  //Desabilita a alteração.
   Set_Commands_Buttons_Mb(Mb_Cm_Bof_Prev_Next_Eof);
end;
```

Create_RCommand

Declaração `public Procedure Create_RCommand(Const aStrCommand:tString; Const aName,aParam :AnsiString; Const aKeyCode:Word; Const aAHelpCtx:Word; Const aFlag : Byte; Const aMb_Bits : Longint; Const aFlags_Buttons : Byte; var RCommand_Temp : TRCommand );`

Descrição O método `Create_RCommand` é usado para iniciar os elementos da matriz `Commands_Buttons`(??)

- **EXEMPLO E USO**

```
Function Create_RCommands_Buttons(Var aCommands : Array of TRCommand):SmallWor
Begin
  If High(aCommands) < 2
  Then Raise TException.Create(self,'Create_RCommands_Buttons()',ParametroInva

  Create_RCommand('CmDbGoBof'  ,CmDbGoBof   ,'&Inicio'   ,'',kbNoKey,0,0,Mb_Cm_
  Create_RCommand('CmDbPrevRec',CmDbPrevRec ,'&Anterior' ,'',kbNoKey,0,0,Mb_Cm_

end;
```

Create_RCommands_Buttons

Declaração `protected Function Create_RCommands_Buttons(aCmNovoStr:AnsiString;aCmAlteracaoStr:AnsiString;aCmExclusaoStr:Ans overload; Virtual;`

Descrição O método `Create_RCommands_Buttons` retorna em **aCommands** a matriz aberta de TRCommand e em **result** retorna o número de elementos adicionados em **aCommands**.

- **EXEMPLO E USO**

```
Var
  Commands_Buttons_High : Byte; //Numero de comandos de Commands
  Commands_Buttons      : Array[0..Max_List_Buttons] of TRCommand;
Begin
  Commands_Buttons_High := Create_RCommands_Buttons(Commands_Buttons);
end;
```

Add_RCommands_Buttons

Declaração `protected function Add_RCommands_Buttons(aStrCommand: tString; aName: AnsiString; aParam: tString; aKeyCode: Word; aAHelpCtx: Word; aState: Byte; aFlags_Buttons: Byte): Longint;`

Descrição O método `Add_RCommands_Buttons` adiciona um botão na posição `Commands_Buttons`(**??**)[`Commands_Buttons_H`

Length_Button_Name_Actives

Declaração `protected Function Length_Button_Name_Actives:Smallint;`

Descrição O método `Length_Button_Name_Actives` retorna a soma do número de caracteres do campo **TRCommand.name** dos botões ativos

Get_Commands_Mb_i

Declaração `public Function Get_Commands_Mb_i(Const aMb_Bits:Longint):Longint;`

Descrição O método `Get_Commands_Mb_i` retorna a posição na matriz `Commands_Buttons`(**??**) do mapa de bit passado por aMb_Bits.

- **Nota**
  - A posição deve ser a mesma do Mapa: _Commands_Buttons_Mb

Get_Commands_Mb_StrCommand

Declaração `public Function Get_Commands_Mb_StrCommand(Const aMb_Bits:Longint):AnsiString;`

Descrição Retorna o nome do comando passado per aMb_Bits.

Chapter 46

Unit mi_rtl_ui_dmxscroller_form

46.1 Descrição

A unit `mi_rtl_ui_dmxscroller_form` implementa a classe `TDmxScroller_Form`(??).

- Primeiro autor: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**:
  -
- **PENDÊNCIAS**
  - T12 Documentar a unit.
  - T12 Criar propriedade UiDmxScroller_Buttons:`TUiDmxScroller_Buttons`(??)
- **CONCLUÍDO**
  - Criar atributo private FirstDataRow : integer;
  - Criar atributo private PrevRec : integer;
  - Criar atributo protected DMXFields : TFPList;
  - Criar atributo protected FldRadioButtonsAdicionados:TStringList;

- Criar atributo Public Function SetHelpCtx_Hint
- Criar atributo Public Procedure SetHelpCtx_Hint
- Criar constructor Create(aOwner:TComponent);Override;
- Criar método public procedure AfterConstruction; override;
- Criar public destructor destroy;override;
- Criar método protected procedure ShowControlState;override;
- Criar método protected procedure CreateStruct
- Criar método Protected procedure DestroyStruct; Override;
- Criar método procedure Scroll_it_inview_LCL
- Criar método public procedure Scroll_it_inview
- Criar método protected procedure CreateFormLCL
- Criar método public function GetTemplate(aNext: `PSItem(??)`)
- Criar método protected procedure UpdateBuffers_Controls;virtual;
- Criar método public procedure UpdateBuffers;override;
- Criar método public procedure Refresh;override;
- Criar método protected procedure SetActiveTarget(aActive : Boolean);override;
- Criar método protected procedure SetActive(aActive : Boolean);override;

46.2 Uses

- `Classes`
- `SysUtils`
- `typInfo`
- `mi.rtl.Consts(??)`
- `mi_rtl_ui_Dmxscroller(??)`

46.3 Visão Geral

`TDmxScroller_Form_Atributos Classe`

`TDmxScroller_Form Classe`

`Register`

46.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Atributos Classe

Hierarquia

TDmxScroller_Form_Atributos > `TUiDmxScroller`(**??**) > `TUiMethods`(**??**) > TUiConsts

Descrição

no description available, TUiDmxScroller description followsA classe `TUiDmxScroller` tem como objetivo criar um formulário baseado em uma lista do tipo ShortString.

- **NOTAS**
  - O método `createStruct`(**??**) criar uma lista de campo tipo `TDmxFieldRec`(**??**) com todas as informações necessárias para criar uma tabela ou um formulário.
  - O formulário é criado com apena uma linha.
- **EXEMPLO**:
  - Template := ' Nome \SSSSSSSSSSSSSSSSSSS Idade: \BB'
    - A classe cria a lista de campos:
      - Label1: Nome
      - Field1: campo ShortString com 20 posições maiúsculas
      - Label2: Idade
      - Field2: Campo byte com duas posições

Campos

DMXFields

Declaração `protected DMXFields:  TFPList;`

Descrição O atributo `DMXFields` salva todos os rótulos e campos da lista de Templates

- **MOTIVO**
  - A classe mãe `TUiDmxScroller`(**??**) mãe da classe `TDmxScroller_Form`(**??**) cria Template de apenas uma linha, a lista `DMXFields` salva todas as linhas para geração de um Template do tipo formulário.

FldRadioButtonsAdicionados

Declaração `protected FldRadioButtonsAdicionados:TStringList;`

Descrição Usado para evitar que RadiosButton sejam adicionados mais de uma vês em radiosgroups diferentes.

- Mais informações veja campos FldRadioGrous.

Métodos

SetHelpCtx_Hint

Declaração
```
public Function
SetHelpCtx_Hint(aFldNum:Integer;a_HelpCtx_Hint:AnsiString):pDmxFieldRec;
override;
```

Descrição O método `SetHelpCtx_Hint` inicia a documentação resumida do campo. aFldNum

SetHelpCtx_Hint

Declaração
```
public Procedure
SetHelpCtx_Hint(apDmxFieldRec:pDmxFieldRec;a_HelpCtx_Hint:AnsiString);
override; overload;
```

Descrição O método `SetHelpCtx_Hint` inicia a documentação resumida do campo passado em :apDmx-FieldRec

TDmxScroller_Form Classe

Hierarquia

TDmxScroller_Form > `TDmxScroller_Form_Atributos(??)` > `TUiDmxScroller(??)` > `TUiMethods(??)` > TUiConsts

Descrição

A classe `TDmxScroller_Form` implementa a construção de formulários usando uma lista de Templates do tipo TDmxScroller

Propriedades

name

Declaração `published property name;`

Alias

Declaração `published property Alias;`

Strings

Declaração `published property Strings;`

OnAddTemplate

Declaração `published property OnAddTemplate;`

OnNewRecord

Declaração `published property OnNewRecord;`

onCloseQuery

Declaração `published property onCloseQuery;`

onEnter

Declaração `published property onEnter;`

onExit

Declaração `published property onExit;`

onEnterField

Declaração `published property onEnterField;`

onExitField

Declaração `published property onExitField;`

onGetTemplate

Declaração `published property onGetTemplate;`

Active

Declaração `published property Active;`

AlignmentLabels

Declaração `published property AlignmentLabels;`

Métodos

Create

Declaração `public constructor Create(aOwner:TComponent); Override;`

Descrição Constrói o componente

AfterConstruction

Declaração `public procedure AfterConstruction; override;`

destroy

Declaração `public destructor destroy; override;`

Descrição Destrói o componente

CreateStruct

Declaração `protected procedure CreateStruct(var ATemplate : PSItem); override; overload;`

Descrição O método `CreateStruct` interpreta uma lista de `strings`(??) do tipo `PSItem(??)` e adiciona os layout de cada campo em `pDmxFieldRec`(??), em seguida adiciona `pDmxFieldRec(??)` em `DMXFields(??)` com todos os campos de formação de formulário visual.

DestroyStruct

Declaração `protected procedure DestroyStruct; Override;`

Descrição O método `DestroyStruct` destrói os dados criados em `CreateStruct(??)()`.

GetTemplate

Declaração `public function GetTemplate(aNext: PSItem) : PSItem; overload; override;`

Descrição O método `GetTemplate` retorna uma lista de `PSItem(??)` (Lista de `strings`(??)) com o modelo usado para criar a tela.

- **NOTA**
  - O Evento `onGetTemplate`(??) só é iniciado em tempo de execução, por isso o formulário não pode ser criado em tempo de desenho do aplicativo.
  - Caso o evento `onGetTemplate`(??) seja nil, então não posso ativar a tela.

- Esse método pode ser anulado, caso se queira ignorar o evento `onGetTemplate`(**??**) e definir o Template em uma método pai herdado desta classe.
- O modelo cria um formulário usando os tipos de dados primitivos.

- **EXEMPLO**

```
// Implementação de um modelo no Alvo LCL
function TDMAlunos.DmxScroller_Form_AlunoGetTemplate(aNext: PSItem): PSItem;
begin
  with DmxScroller_Form_Aluno do
  begin
    // AlignmentLabels:= taCenter;
    AlignmentLabels := taLeftJustify;
    // AlignmentLabels := taRightJustify ;
    Result :=
      NewSItem('      Matrícula  \LLLLL'+CharFieldName+'matricula'+CharAccRead(
      NewSItem(' Nome do aluno:  \sssssssssssssssssss‘sssssss’+CharFieldName+'
      NewSItem('',
      NewSItem('      Endereço:  \sssssssssssssssssss‘sssssssssss’+CharFieldNa
      NewSItem(' P. Referência:  \sssssssssssssssssss‘sssss’+CharFieldName+’Pc
      NewSItem('           Cep:  \##.###-###’+CharFieldName+’cep’,
      NewSItem('        Estado:  \SS’+CharFieldName+’Estado’+CharForeignKeyN_Um
      NewSItem('        Cidade:  \sssssssssssssssssss‘sssss’+CharFieldName+’ci
      NewSItem('',
      aNext)))))))));
  end;
end;
```

UpdateBuffers_Controls

Declaração `protected procedure UpdateBuffers_Controls; virtual;`

UpdateBuffers

Declaração `public procedure UpdateBuffers; override;`

SetActiveTarget

Declaração `protected procedure SetActiveTarget(aActive:  Boolean); virtual;`

Descrição A procedure `SetActiveTarget` seta a propriedade `active`(**??**) e criar um formulário na plataforma alvo

SetActive

Declaração `protected procedure SetActive(aActive :  Boolean); override;`

Descrição A procedure `SetActive` seta a propriedade `active`(**??**) e criar um formulário LCL ou HTML dependendo do tipo de aplicação

46.5 Funções e Procedimentos

Register

Declaração `procedure Register;`

46.6 Tipos

TDmxFieldRec

Declaração `TDmxFieldRec = mi_rtl_ui_Dmxscroller.TDmxFieldRec;`

pDmxFieldRec

Declaração `pDmxFieldRec = mi_rtl_ui_Dmxscroller.pDmxFieldRec;`

SmallWord

Declaração `SmallWord = TUiDmxScroller.SmallWord;`

46.7 Constantes

accDelimiter

Declaração `accDelimiter = TConsts.accDelimiter ;`

Descrição A constante `accDelimiter` informa que o campo é delimitador de campos no Template.

accHidden

Declaração `accHidden = TConsts.accHidden ;`

Descrição A constante `accHidden` (Const `accHidden` = 2;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o mesmo é invisível.

- **EXEMPLO**
  - Como usar o mapa de bits `accHidden` para saber se o campo está invisível.

```
with pDmxFieldRec^ do
  If (access and accHidden <> 0)
  then begin
         ShowMessage(Format('O campo %s está invisível'),[CharField
       end;
```

AccNormal

Declaração `AccNormal = TConsts.AccNormal;`

Descrição A constante `AccNormal` (Const `AccNormal` = 0;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se que o campo pode ser editado.

- **EXEMPLO**

– Como usar o mapa de bits `accNormal` para saber se o campo pode ser editado.

```
with pDmxFieldRec^ do
  If (access and accNormal <> 0)
  then begin
         ShowMessage(Format('O campo %s pode ser editado'),[FieldNa
       end;
```

accReadOnly

Declaração `accReadOnly = TConsts.accReadOnly ;`

Descrição A constante `accReadOnly` (Const ReadOnly = 1;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o campo é somente para leitura.

- **EXEMPLO**

  – Como usar o mapa de bits ReadOnly para saber se o campo não pode ser editado.

```
with pDmxFieldRec^ do
  If (access and ReadOnly <> 0)
  then begin
         ShowMessage(Format('O campo %s não pode ser editado'),[Fie
       end;
```

accSkip

Declaração `accSkip = TConsts.accSkip ;`

Descrição A constante `accSkip` (Const `accSkip` = 4;) é um mapa de bits usado para identificar o bit do campo `TDmxFieldRec.access`(**??**) que informa se o campo pode receber o focus.

- **EXEMPLO**

  – Como usar o mapa de bits `accSkip` para saber se o campo não pode receber o focus.

```
with pDmxFieldRec^ do
 If (access and accSkip <> 0)
 then begin
        ShowMessage(Format('O campo %s não pode receber o focus'),
      end;
```

CharHint

Declaração `CharHint = TConsts.CharHint;`

Descrição A constante `CharHint` é usado para documentar o campo e indica que todo o texto até o próximo caractere de controle será o conteúdo do campo HelpCtx_Hint.

- **EXEMPLO**

Resourcestring

```
tmp_Alunos_Idade = '\BB'+ChFN+'idade'+CharUpperlimit+#64+
                   CharHint+'A idade do aluno. Valores válidos 1 a 64'+
                   CharHintPorque+'Este campo é necessário para que se agrupe
                   CharHintOnde+'Ele será usado pelo coordenador ao classific


tmp_Alunos_Matricula = \IIII'+ChFN+'matricula'+CharHint+'A matricula do aluno

tmp_Alunos = '      Idade: %s'+lf+
             '  Matricula: %s
```

CharHintOnde

Declaração `CharHintOnde = TConsts.CharHintOnde;`

Descrição A contante `CharHintOnde` informa que todo texto até o próximo delimitador contém informações para o campo HelpCtx_Onde

CharHintPorque

Declaração `CharHintPorque = TConsts.CharHintPorque;`

Descrição A contante `CharHintPorque` informa que todo texto até o próximo delimitador contém informações para o campo HelpCtx_Porque

fld_LData

Declaração `fld_LData = TConsts.fld_LData ;`

Descrição A constante `fld_LData` é do tipo TDateTime e guarda a data compactada 'dd/dd/dd'

fld_LHora

Declaração `fld_LHora = TConsts.fld_LHora ;`

Descrição A constante `fld_LHora` é do tipo TDateTime e guarda a hora compactada ##:##:##

fldAnsiChar

Declaração `fldAnsiChar = TConsts.fldAnsiChar ;`

Descrição A constante `fldAnsiChar` (Const `fldAnsiChar` = 'C') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'C' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere maiúsculo.

- **EXEMPLO**
  - Representação de um AnsiString de 10 dígitos em um buffer de 11 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

```
Const
  Nome := '\CCCCCCCCCC'; //PAULO SÉRG
```

fldAnsiChar_Minuscula

Declaração `fldAnsiChar_Minuscula = TConsts.fldAnsiChar_Minuscula;`

Descrição A constante `fldAnsiChar_Minuscula` (Const `fldAnsiChar(??)` = 'c') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'c' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere minúsculo.

- **EXEMPLO**

  - Representação de um AnsiString de 10 dígitos em um buffer de 11 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

Const

```
Nome := '\ccccccccc'; //paulo Sérg
Nome := '\Ccccccccc'; //Paulo Sérg
```

fldAnsiCharNUM

Declaração `fldAnsiCharNUM = TConsts.fldAnsiCharNUM ;`

Descrição A constante `fldAnsiCharNUM` (Const `fldAnsiChar(??)` = '0') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '0' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere numérico ['0'..'9']] .

- **EXEMPLO**

  - Representação de um AnsiString de 11 dígitos em um buffer de 12 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

Const

```
telefone := '\(00) 0 0000-0000' //85 9 9702 4498
```

fldAnsiCharVAL

Declaração `fldAnsiCharVAL = TConsts.fldAnsiCharVAL ;`

Descrição A constante `fldAnsiCharVAL` (Const `fldAnsiChar(??)` = '0') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '0' após o caractere "\" representa no buffer do formulário um tipo AnsiString que só aceita caractere numérico ['0'..'9']] com formatação dbase.

- **EXEMPLO**
  - Representação de um AnsiString de 11 dígitos em um buffer de 12 bytes onde o ultimo byte contém o caractere #0 informando o fim da string;

Const

```
telefone := '\(NN) N NNNN-NNNN' //85 9 9702 4498
```

fldAPPEND

Declaração `fldAPPEND = TConsts.fldAPPEND ;`

Descrição A constante `fldAPPEND` é usada para concatenar duas listas do tipo `PSItem(??)`.

- A constante `fldAPPEND` é necessário porque DmxScroller trabalha com string curta e a mesma tem um tamanho de 255 caracteres, onde o tamanho está na posição 0.
- Como usar a constante `fldAPPEND`:
  - A função **CreateAppendFields** retorna a constante `fldAPPEND` mais o endereço da string a ser concatenada.
    - **EXEMPLO**

```
procedure Template : ShortString;
  Var
    S1,s2,Template : TString;
begin
  S1 := ' Nome do Aluno....: \sssssssssssssssssssssss
  s2 := ' Endereço do aluno: \sssssssssssssssssssssss
  result := S1+CreateAppendFields(s2);
end;
```

* **NOTA**

  - A contante `fldAPPEND` foi criada porque o projeto inicial foi para turbo pascal e ambiente console.
  - A versão atual podemos usar AnsiString visto que o limite do mesmo é a memória.
  - Para usar AnsiString é necessário converter para `PSitem`(**??**) com a função: **StringToSItem**.
  - **EXEMPLO:**

```
function TMI_UI_InputBox.DmxScroller_Form1Ge
begin
  with DmxScroller_Form1 do
  begin
    if _Template  <> ''
    then Result := StringToSItem(_Template,

//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 40
//    Result := StringToSItem(_Template, 80

    else result := nil;
  end;
end;
```

fldBLOb

Declaração `fldBLOb = TConsts.fldBLOb ;`

Descrição A constante `fldBLOb` indica que o campo é não formatado podendo ser um Record, porém a edição do mesmo será feito por outros meios.

- **NOTA**
  - Para informar ao buffer do registro que o campo é `fldBLOb`, a função **CreateBlobField** é necessário.
  - A **class function** `TUiMethods.CreateBlobField(??)`**(Len: integer; AccMode,Default: byte) : DmxIDstr;** reserva espaço para o mesmo.
  - Pendência: Preciso criar um exemplo de uso deste tipo de informação.

fldBYTE

Declaração `fldBYTE = TConsts.fldBYTE ;`

Descrição A constante `fldBYTE` (Const `fldBYTE` = 'B') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'B' após o caractere "\" representa no buffer do formulário um tipo byte.

- **EXEMPLO**

Const

```
idade := '\BB' //Os dois dígitos estarão em um buffer de 1 byte;
```

fldBoolean

Declaração `fldBoolean = TConsts.fldBoolean;`

Descrição A constante `fldBoolean` (`fldBoolean` = 'X') indica que o campo é do tipo byte e só pode ter dois valores.

- **NOTA**
  - Valores possíveis:

        * 0 - False; não
        * 1 = True; sim
    - A forma de editá-los deve ser com o componente checkbox.

- **EXEMPLO**

Resourcestring

```
tmp_Aceita = '\[X]'+ChFN+'Aceita_contrato'+CharHint+'Aceita os termos do cont
Template = tmp_Aceita+' Aceita os termos do contrato ';
```

fldCONTRACTION

Declaração `fldCONTRACTION = TConsts.fldCONTRACTION ;`

Descrição A constante `fldCONTRACTION` ...

fldData

Declaração `fldData = TConsts.fldData ;`

Descrição A constante `fldData` ...

FldDateTimeDos

Declaração `FldDateTimeDos = TConsts.FldDateTimeDos ;`

Descrição A constante `FldDateTimeDos` ...

fldENUM

Declaração `fldENUM = TConsts.fldENUM ;`

Descrição A constante `fldENUM` (`fldENUM`=^E) é um campo do tipo byte(0..255) que contém uma lista de string que são selecionadas por um componente tipo ComboBox.

- **EXEMPLO**

```
  Const tmpMidia : PSitem = nil;
begin
  tmpMidia := CreateEnumField(TRUE, accNormal, 0,
                              NewSItem(' indefinido ',
                              NewSItem(' PenDriver  ',
                              NewSItem(' SSD        ',
                              nil))))+CharFieldName+'Midia;

  Template = NewSItem('   Eu uso  '+ tmpMidia + '  em meu computador.
```

CharExecAction

Declaração `CharExecAction = TConsts.CharExecAction ;`

Descrição A contante `CharExecAction` é usado para associar ao campo atual uma classe **TAction**.

- **NOTA**
  - O interpretador de Templates associa a ação do Template ao corrente campo.
- **EXEMPLO DE USO DE AÇÕES NO TEMPLATE**
  1. Se o atributo **Fieldnum** do campo for diferente de zero, então o **rótulo** do botão associado a ação será o caracteres e a ação pode atualizar o buffer do campo.
     - No exemplo a seguir um rótulo e um campo de cliente:

```
NewSItem(' Cliente: '+'\LLLLL'+CharExecAction+CreateExe
```

  2. Se o atributo **Fieldnum** do campo for igual a zero, então a rótulo do botão será o rótulo do campo.
     - No exemplo a seguir um rótulo de novo cliente (icons) e um botão ok (icons)

```
NewSItem('   &Novo cliente: '+CharExecAction+CreateExec
         '         '+CharExecAction+CreateExecAction(Ac
```

fldExtended

Declaração `fldExtended = TConsts.fldExtended ;`

Descrição A constante `fldExtended` (`fldExtended`='E') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'E' após o caractere "\" representa no buffer do formulário um tipo Extended.

- **EXEMPLO**

```
Const
   Valor := '\EEE,EEE,EEE,EEE,EE' //Todos os números editados nesta
                                   //mascara, estarão em um buffer de 10 bytes;
```

CharFieldName

Declaração `CharFieldName = TConsts.CharFieldName ;`

Descrição A constante `CharFieldName` usada para informar o nome do campo informado antes deste caractere.

- **EXEMPLO**

```
Const
  idade := '\BB'+CharFieldName+'idade'
```

ChFN

Declaração `ChFN = TConsts.ChFN ;`

Descrição A constante `ChFN` é igual a `CharFieldName`(??). Foi criada para facilitar seu uso.

fldHexValue

Declaração `fldHexValue = TConsts.fldHexValue ;`

Descrição A constante `fldHexValue` ...

fldLData

Declaração `fldLData = TConsts.fldLData ;`

Descrição A constante `fldLData` ...

fldLHora

Declaração `fldLHora = TConsts.fldLHora ;`

Descrição A constante `fldLHora` ...

fldLONGINT

Declaração `fldLONGINT = TConsts.fldLONGINT ;`

Descrição A constante `fldLONGINT` ...

FldMemo

Declaração `FldMemo = TConsts.FldMemo ;`

Descrição A constante `FldMemo` ...

FldOperador

Declaração `FldOperador = TConsts.FldOperador ;`

Descrição A constante `FldOperador` ...

FldRadioButton

Declaração `FldRadioButton = TConsts.FldRadioButton ;`

Descrição A constante `FldRadioButton` ...

fldReal4

Declaração `fldReal4 = TConsts.fldReal4 ;`

Descrição A constante `fldReal4` ...

fldReal4P

Declaração `fldReal4P = TConsts.fldReal4P ;`

Descrição A constante `fldReal4P` ...

fldReal4Positivo

Declaração `fldReal4Positivo = TConsts.fldReal4Positivo ;`

Descrição A constante `fldReal4Positivo` ...

fldReal4PPositivo

Declaração `fldReal4PPositivo = TConsts.fldReal4PPositivo ;`

Descrição A constante `fldReal4PPositivo` ...

fldRealNum

Declaração `fldRealNum = TConsts.fldRealNum ;`

Descrição A constante `fldRealNum` ...

fldRealNum_Positivo

Declaração `fldRealNum_Positivo = TConsts.fldRealNum_Positivo;`

Descrição A constante `fldRealNum_Positivo` ...

FldSData

Declaração `FldSData = TConsts.FldSData ;`

Descrição A constante `FldSData` ...

FldSDateTimeDos

Declaração `FldSDateTimeDos = TConsts.FldSDateTimeDos ;`

Descrição A constante `FldSDateTimeDos` ...

FldSHora

Declaração `FldSHora = TConsts.FldSHora ;`

Descrição A constante `FldSHora` ...

fldSHORTINT

Declaração `fldSHORTINT = TConsts.fldSHORTINT ;`

Descrição A constante `fldSHORTINT` ...

fldSItems

Declaração `fldSItems = TConsts.fldSItems ;`

Descrição A constante `fldSItems` ...

fldSmallInt

Declaração `fldSmallInt = TConsts.fldSmallInt ;`

Descrição A constante `fldSmallInt` ...

fldSmallWORD

Declaração `fldSmallWORD = TConsts.fldSmallWORD ;`

Descrição A constante `fldSmallWORD` ...

fldSTR

Declaração `fldSTR = TConsts.fldSTR ;`

Descrição A constante `fldSTR` (Const `fldStr` = 'S') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 'S' após o caractere "\" representa no buffer do formulário um tipo ShortString que só aceita caractere maiúsculo.

- **EXEMPLO**
  - Representação de um string de 10 dígitos em um buffer de 11 bytes onde o byte zero contém o tamanho da string;

```
Const
  Nome := '\SSSSSSSSSS'  //PAULO SÉRG
```

fldSTR_Minuscula

Declaração `fldSTR_Minuscula = TConsts.fldSTR_Minuscula;`

Descrição A constante `fldSTR_Minuscula` (Const `fldSTR_Minuscula` = 's') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres 's' após o caractere "\" representa no buffer do formulário um tipo ShortString que só aceita caractere minúscula.

- **EXEMPLO**

  - Representação de um string de 10 dígitos em um buffer de 11 bytes onde o byte zero contém o tamanho da string;

```
Const
  Nome := '\sssssssss' //paulo sérg
  Nome := '\Sssssssss' //Paulo sérg
```

fldSTRNUM

Declaração `fldSTRNUM = TConsts.fldSTRNUM ;`

Descrição A constante `fldSTRNUM` (Const `fldSTRNUM` = '#') usado na máscara do Template, informa ao componente `TUiDmxScroller(??)` que a sequência de caracteres '#' após o caractere "\" representa no buffer do formulário um tipo ShortString que só aceita caractere numérico.

- **EXEMPLO**

  - Representação de um string de 11 dígitos em um buffer de 12 bytes onde o byte zero contém o tamanho da string;

```
Const
  telefone := '\(##) # ####-####' //85 9 9702 4498
```

CharUpperlimit

Declaração `CharUpperlimit = TConsts.CharUpperlimit ;`

Descrição A constante `CharUpperlimit` (`CharUpperlimit`=^U) permite informar um limite superior para campos do tipo byte.

- O gerador de formulário deve usar o conteúdo do campo pDmxFieldRec.Upperlimit para criticar se o valor do campo está na faixa entre 1 e pDmxFieldRec.Upperlimit.
- O valor zero significa que o campo está nulo.
- **EXEMPLO**

– Um campo onde o seu conteúdo não ultrapasse um byte, pode ser informado no Template da seguinte forma:

Const
```
idade := '\BBB+CharUpperlimit+#130+CharHint+'Não existe humanos c
```

fldZEROMOD

Declaração `fldZEROMOD = TConsts.fldZEROMOD ;`

Descrição A constante `fldZEROMOD` ...

CharShowPassword

Declaração `CharShowPassword = TConsts.CharShowPassword ;`

Descrição A constante `CharShowPassword` informa para controle que os caracteres não devem ser visualizado.

- **NOTA**
  - Usados no campos tipo senha.

CharShowPasswordChar

Declaração `CharShowPasswordChar = TConsts.CharShowPasswordChar ;`

Descrição A constante `CharShowPasswordChar` ...

TypeDate

Declaração `TypeDate = TConsts.TypeDate ;`

Descrição A constante `TypeDate` ...

_TypeDate

Declaração `_TypeDate = TConsts._TypeDate ;`

Descrição A constante `_TypeDate` ...

TypeHora

Declaração `TypeHora = TConsts.TypeHora ;`

Descrição A constante `TypeHora` ...

TypeMemo

Declaração `TypeMemo = TConsts.TypeMemo ;`

Descrição A constante `TypeMemo` ...

Chapter 47

Unit mi_rtl_ui_Dmxscroller_sql

47.1 Descrição

A unit `mi_rtl_ui_Dmxscroller_sql` implementa a classe `TUiDmxScroller_sql`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
  - **PENDÊNCIAS**
    - T12 Falta implementar chave estrangeira em createTable;
    - T12 Em `TUiDmxScroller_sql.DoOnNewRecord`(**??**); está executando o método (CustomBuf-Dataset as TSQLQuery).Append; antes do componenente `TUiDmxScroller_sql`(**??**) está visível e isto está gerando exceção.
    - T12 ANÁLISE

Estudar os procedimentos armazenados (https://www.w3schools.com/sql/sql_stored_procedures.asp)

Estudar as restrições SQL (https://www.w3schools.com/sql/sql_constraints.asp)

- Como saber se um campo é uma chave que liga outra tabela?
- https://www.w3schools.com/sql/sql_foreignkey.asp (SQL FOREIGN KEY Constraint)

```
/*Não, podemos permitir que os registros das pessoas que possuim camisetas
  lavando sejam apagados, para garantir a integridade da informação.
   Para isso devemos utilizar o as chaves estrangeiras que acusarão
  um erro quando tentarmos deletar uma pessoa que possuir camisetas.
   Veja em código:
 */

CREATE TABLE Pessoa(
    IdPessoa INT NOT NULL PRIMARY KEY IDENTITY(1,1),
    Nome VARCHAR(20) NOT NULL
)

CREATE TABLE Camiseta(
    IdCamiseta INT NOT NULL PRIMARY KEY IDENTITY(1,1),
    Descrição VARCHAR(20) NOT NULL,
    IdPessoa INT NOT NULL
    CONSTRAINT FK_Camiseta_Pessoa FOREIGN  KEY(IdPessoa) REFERENCES Pessoa(IdPessoa)
)

INSERT INTO Pessoa VALUES ('HeyJoe')
INSERT INTO Pessoa VALUES ('Caique')


INSERT INTO Camiseta VALUES ('Azul', 1)
INSERT INTO Camiseta VALUES ('Amarela', 1)
INSERT INTO Camiseta VALUES ('Preta', 2)

SELECT * FROM Pessoa, Camiseta WHERE Pessoa.IdPessoa = Camiseta.IdPessoa
```

  - Como saber o tipo de relacionamento que os campos de outra tabela tem com a tabela atual?

* T12 A opção CreateTable está dando mensagem de erro quando a coluna já existe.

  - Encontrar uma forma de não gerar exceção ou ignorar as exceções nesta rotina.

* T12 Em `TUiDmxScroller_sql.AlterTable`(**??**) checar:

  - T12 Criar código para todos os tipos reconhecidos por marIcaraí.

  - T12 Debugar para saber se está tudo funcionando.

  - T12 Permitir adicionar uma nova coluna mesmo que a tabela já exista.

* T12 Em SetTableName(aTableName:String) criticar o nome aTableName é um nome válido para a tabela.

– **HISTÓRICO**

* Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
* **2022-03-13**
  - **09:00**
  - T12 Implementar a criação de código SQL baseado nos dados de `TUiDmxScroller`(**??**).
  - **ANÁLISE:**
  - Onde pegar o nome da tabela ou consulta?
  - TUiDmxScroller.CustomBufDataset.FileName;
  - Onde pegar o nome dos campos da tabela CustomBufDataset.Filename?
  - A lista de campos encontra-se em : TUiDmxScroller_Atributos.Fields : TFPList;
  - Como saber se `TUiDmxScroller`(**??**) é uma tabela ou a uma consulta?
  - Se todos os TUiDmxScroller_Atributos.Fields[].FieldName não contém '—' é porque é FileName é uma tabela.
  - Se pelo menos um TUiDmxScroller_Atributos.Fields[].FieldName contém '—' é porque é FileName é uma consulta envolvendo mais de uma tabela.
  - Como saber se uma tabela ou consulta existe do banco de dados?
  - O SQL do **postegres** e do **sqlite3** tem a clausula **IN NOT EXISTS** no comando CREATE TABLE:
  - EXEMPLO:

```
CREATE TABLE IF NOT EXISTS TEST01 ();
```

  - **REFERÊNCIAS**
  - https://en.wikipedia.org/wiki/SQL:2016 (SQL:2016)
  - (PostgresSQL aceita 160 das 169 especificação 2016)(https://www.postgresql.org/docs/12/features.htn

bancos de dados x conformidade SQL (https://en.wikipedia.org/wiki/SQL_compliance)

lientes de bancos de dados opensource (https://medevel.com/17-sql-client-open-source/)

- https://dbeaver.io/ (Instalei programa cliente SQL DBeaver)
- Obs: Não deu certo. Ele é escrito em java e não funcionou o básico.

sqlite create database if not exists (https://www.codegrepper.com/code-examples/sql/sqlite+create+database+if+not+exists)

* **2022-03-14**
  - **08:22**
  - T12 Criar a unit mi_ui_Dmxscroller_sql.pas com a classe `TUiDmxScroller_sql`(**??**) com objetivo de concentrar a integração do TDmxScroller com o componente **TSQLQuery**
  - **20:00**
  - T12 Na Construção de TFields atualizar a propriedade **TField.ProviderFlags** com o tipo de acesso definido em `TDmxFieldRec.Access`(**??**)
  - **21:12**
  - T12 Criar propriedade **TableName**
  - **21:27**
  - T12 Criar Function SetSqlCustomBufDataset:Boolean;Virtual;
  - CustomBufDataset.SQL := **SELECT * FROM X** onde X será definido pela propriedade **TableName**
* **2022-03-15**
  - **09:11**
  - Depurar o que fiz ontem para fazer funciona a atualização do banco de dados SQL.
  - **11:36**
  - Criar método `TUiDmxScroller_sql.AlterTable`(**??**) : Boolean;
  - **14:38**
  - T12 Atualizar TSQLQury.TFields.ProviderFlags com TUiDmxScroller.MiProviderFlags
* **2022-03-16**
  - **16:23**
  - T12 Em `TUiDmxScroller_sql.CreateCustomBufDataset_FieldDefs`(**??**), atualizar **TField.ProviderFlags** com os dados do campo `TDmxFieldRec.ProviderFlags`(**??**).

- **16:54**
- Em `TUiDmxScroller_sql.AlterTable`(**??**) usar os flags `TDmxFieldRec.ProviderFlags`(**??**) para criação da tabela.

* **2022-03-17**
  - **10:48**
  - T12 Os flags indicando que se trata de chave primária não está sendo atualizado em createStructor, por isso não está criando a chave primária.

* **2022-03-18**
  - **10:40**
  - T12 Ao criar uma tabela SQL em **AlterTable** adicionar colunas ao invés de criar a tabela toda.
  - **Motivo**:
  - Permitir que o banco de dados fique compatível com o Template.
  - Alterar um coluna de forma automática não é bom, porque o que está feito gera dependências que produzirão erros ao fazer essas alterações.

* **2022-03-21**
  - **08:57**
  - T12 Criar function SQL_AddkeysPrimaryKeyComposite(I : Integer):Boolean;
  - Esta função adiciona chave primária composta na tabela.
  - **REFERÊNCIA**

a usando a expressão ALTER TABLE (https://www.techonthenet.com/postgresql/primary_keys.php#: :text=In%20PostgreSQL%2C%20a%

  - **15:40**
  - T12 Em AlterTable criar a restrição de chave estrangeira no TDmxScroller_sql.
  - Nome da função: function AddKeyForeigns(I : Integer):Boolean;

* **2022-03-22**
  - **09:00**
  - T12 Documentar as units `TuiTypes`(**??**) e TUIConsts.
  - **10:00**

- T12 Criar os relacionamentos entre tabelas (restrições entre tabelas)
- **14:14**
- T12 Depurar os relacionamentos entre tabelas.
- **18:47**
- O Componente CustomBufDataset não está entrando no modo edit.
- O problema estava nos eventos TScrollBoxDMX.DoOnEnter e TScrollBoxDMX.DoOnExit;]

* **2022-03-22**

  - **20:27**
  - T12 Analisar como criar os comandos CmIncluir, cmAlterar, cmExcluir, cmConsulta para a tabela TDmxScroller
  - Criar os comandos:
  - Public Procedure DoOnNewRecord;overload;override; //Usado para inicializa os parametros de um novo registro
  - Public Procedure PutRec;Override;//Grava o buffer no arquivo memo
  - Public Procedure GetRec;Override;//O primeiro registro esta gravado em Value
  - Public Function DeleteRec:Boolean;Override;
  - Function UpdateRec: Boolean;Override;
  - Function UpdateRec_if_RecordAltered:Boolean;Override;
  - Function PrevRec : Boolean;overload;override;
  - Function NextRec : Boolean;overload;override;

* **2022-03-23**

  - Criar método Public Function AddRec:Boolean;Override;
  - Para que DoAddrec possa adicionar o registro é necessário que o registro esteja selecionando, ou seja no modo edit.
  - Obs: Está com problema.

* **2022-03-25**

  - https://wiki.freepascal.org/Firebird#Creating_objects_programmatically (Estudar página sobre o banco de dados firebird)

* **2022-03-28**

  - Em `TUiDmxScroller_sql.DoOnNewRecord(??)`; está executando o método (CustomBufDataset as TSQLQuery).Append; antes do componenente `TUiDmxScroller_sql(??)` está visível e isto está gerando exceção. -

* **2022-03-30**

  - Implementar a conexão com o banco de dados usando o componente Mi_Application.

* **2022-04-14**

  - Debugar o método `TUiDmxScroller_sql.AlterTable(??)`.

* **2022-04-15**

  - O método `TUiDmxScroller_sql.AlterTable(??)` precisa reconhecer a sintaxe do banco de dados selecionado.

  - O postgresSQL sintaxe:

  - CREATE TABLE [IF NOT EXISTS] table_name (column1 datatype(length) column_contraint, column2 datatype(length) column_contraint, column3 datatype(length) column_contraint, table_constraints);

  - **REFERÊNCIA**

postgresql-create-table (https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table/)

  - O sqLite3 sintaxe:

  - CREATE TABLE [IF NOT EXISTS] [schema_name].table_name (column_1 data_type PRIMARY KEY, column_2 data_type NOT NULL, column_3 data_type DEFAULT 0,table_constraints) [WITHOUT ROWID];

  - **REFERÊNCIA:**

sqlite-create-table (https://www.sqlitetutorial.net/sqlite-create-table/)

47.2 Uses

- `Classes`
- `SysUtils`
- `BufDataset`
- `db`

- SqlDb
- mi.rtl.Types(??)
- mi_rtl_ui_types(??)
- mi_rtl_ui_consts
- mi_rtl_ui_Dmxscroller(??)
- umi_rtl_ui_custom_application

47.3 Visão Geral

TDmxScroller_sql_Atributos Classe

TUiDmxScroller_sql Classe

47.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_sql_Atributos Classe

Hierarquia

TDmxScroller_sql_Atributos > TUiDmxScroller(??) > TUiMethods(??) > TUiConsts

Descrição

A class TDmxScroller_sql_Atributos contém os atributos da class TDmxScroller_sql

Campos

CustomBufDataset

Declaração `public CustomBufDataset: TCustomBufDataset;`

Descrição O atributo pública CustomBufDataset é definida em **CreateCustomBufDataset_FieldDefs** que é executado em **TDmxScroller.CreateData** baseado na estrutura do Template passado por GetTemplate(??).

- **NOTA**
  - O atributo CustomBufDataset deve ser passado por **DataSource.DataSet**.
  - Em **CreateCustomBufDataset_FieldDefs** é criado os campo da propriedade CustomBufDataset se a propriedade (DataSource(??)<>nil) e (DataSource.DataSet <> nil).

- Se a propriedade DataSource.DataSet = nil então a propriedade `CustomBufDataset`=**nil**
- O método **CreateCustomBufDataset_FieldDefs** reconhece duas possibilidade para os descendentes de `CustomBufDataset` quais sejam:

  TBufDataset (https://www.freepascal.org/docs-html/fcl/bufdataset/tbufdataset.

  2. https://www.freepascal.org/docs-html/fcl/sqldb/tcustomsqlquery.h (TCustomSQLQuery)

     * Preciso das propriedades de acesso a banco de dados SQL.
     * O evento `OnGetTemplate`(**??**) deve setar as propriedades customizadas de **TCustomSQLQuery**.

- **REFERNCIA**:

tcustombufdataset (https://www.freepascal.org/daily/packages/fcl-db/bufdataset/tcustombufdataset 14.html)

tcustomsqlquery (https://www.freepascal.org/docs-html/fcl/sqldb/tcustomsqlquery.html)

- https://www.freepascal.org/docs-html/fcl/bufdataset/tcustombufdataset.html (TCustomBufDataset);

TBufDataSet (https://wiki.freepascal.org/How_to_write_in-memory_database_applications_in_La

tstatementtype.html (https://www.freepascal.org/docs-html/fcl/sqltypes/tstatementtype.html)

tsqlquery (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.html)

tdatasetstate (https://www.freepascal.org/docs-html/fcl/db/tdatasetstate.html)

How_to_connect_to_a_database_server (https://wiki.freepascal.org/SqlDBHowto#How_to_connect_to_a_database_server.3

Example:_reading_data_from_a_table (https://wiki.freepascal.org/SqlDBHowto#Example:_reading_data_from_a_table)

How_to_execute_direct_queries.2Fmake_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_execute_direct_queries.2Fmake

How_to_read_data_from_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_read_data_from_a_table.3F)

Why_does_TSQLQuery.RecordCount_always_return (https://wiki.freepascal.org/SqlDBHowto#Why_does_TSQLQuery.RecordCount_a

Como usar SQLDb no Lazarus (https://wiki.freepascal.org/SqlDBHowto#Lazarus)

Trabalhando com tabelas relacionadas (https://wiki.freepascal.org/MasterDetail)

How_to_change_data_in_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_change_data_in_a_table.3F)

How_does_SqlDB_send_the_changes_to_the_database_server (https://wiki.freepascal.org/SqlDBHowto#How_does_SqlDB_send_the_changes_to_

How_to_handle_Errors (https://wiki.freepascal.org/SqlDBHowto#How_to_handle_Errors)

How_to_execute_a_query_using_TSQLQuery (https://wiki.freepascal.org/SqlDBHowto#How_to_execute_a_query_using_TSQLQ

How_to_use_parameters_in_a_query (https://wiki.freepascal.org/SqlDBHowto#How_to_use_parameters_in_a_query.3F)

Select_query (https://wiki.freepascal.org/SqlDBHowto#Select_query)

Exemplo de SQLQuery com parãmetros (https://wiki.freepascal.org/SqlDBHowto#Example)

- https://wiki.freepascal.org/SqlDBHowto#Troubleshooting:_TSQLConnection_log (Troubleshooting:_TSQLConnection_logging)

Exemplo de log (https://wiki.freepascal.org/SqlDBHowto#FPC_.28or:_the_manual_

TUiDmxScroller_sql Classe

Hierarquia

TUiDmxScroller_sql > `TDmxScroller_sql_Atributos(??)` > `TUiDmxScroller(??)` > `TUiMethods(??)` > TUiConsts

Descrição

A classe `TUiDmxScroller_sql` implementa o acesso ao banco de dados usando o atributo `CustomBufDataset(??)`

- **NOTA**
  - O atributo `CustomBufDataset(??)` pode ser **TBufDataset** não conectado a banco de dados sql e **TCustomSQLQuery** conectado ao banco de dados SQL.
- **REFERÊNCIA**

Working_With_TSQLQuery (https://wiki.freepascal.org/Working_With_TSQLQuery)

Parameters_in_TSQLQuery (https://wiki.freepascal.org/Working_With_TSQLQuery#Parameters_in_TSQLQuery.SQL)

sql-basico (https://www.devmedia.com.br/sql-basico/28877)

Propriedades

DataSource

Declaração `published property DataSource : TDataSource Read _DataSource Write _DataSource;`

Descrição A propriedade `DataSource` permite que controles da **LCL** (Lazarus Componenents Library) possam usar os dados do componenente **TDmxScroller**.

- **NOTA**
  - Essa integração permite que **TDmxScroller** utilize todos os componentes de banco de dados do Free Pascal.

Campos

_DataSource

Declaração `protected _DataSource: TDataSource;`

Métodos

SetDataBase

Declaração `protected procedure SetDataBase;`

Create

Declaração `public constructor Create(aOwner:TComponent); Override;`

Descrição Constrói o componente

GetkeysPrimaryComposite

Declaração `public function GetkeysPrimaryComposite(I : Integer):AnsiString;`

Descrição O método `GetkeysPrimaryComposite` retorna a lista de campos pertencentes a chave composta primária.

GetKeysPrimary

Declaração `public function GetKeysPrimary:AnsiString;`

Descrição A função `GetKeysPrimary` retorna a chave primária composta ou não na tabela.

- **Como TSQLQuery trata os campos de chave primária**
  - Ao atualizar registros, TSQLQuery precisa saber quais campos compõem a chave primária que pode ser usada para atualizar o registro e quais campos devem ser atualizados: com base nessas informações, ele constrói um comando SQL UPDATE, INSERT ou DELETE.
  - A construção da instrução SQL é controlada pela propriedade UsePrimaryKeyAsKey e pelas propriedades ProviderFlags .
  - A propriedade Providerflags é um conjunto de 3 sinalizadores:
    - pfInkey : O campo faz parte da chave primária
    - pfInWhere : O campo deve ser utilizado na cláusula WHERE das instruções SQL.
    - pfInUpdate : Atualizações ou inserções devem incluir este campo. Por padrão, ProviderFlags consiste apenas em pfInUpdate .
    - **NOTA***
      - **Se sua tabela tiver uma chave primária (conforme descrito acima), você só precisará definir a propriedade **UsePrimaryKeyAsKey**

      como True e tudo será feito para você. Isso definirá o sinalizador pfInKey para os campos de chave primária.

- **REFERÊNCIA**

Working With TSQLQuery e Primary_key_Fields (https://wiki.freepascal.org/Working_With_TSQLQuery#Primary_key_Fields)

CreateTable

Declaração `public Function CreateTable: Boolean;`

Descrição A função `CreateTable` cria a tabela se a mesma não existir

AlterTable

Declaração `public Function AlterTable:  Boolean; Virtual;`

Descrição O método `AlterTable` cria a tabela ou consulta `TableName(??)` no banco de dados caso a propriedade `TableName(??)` não existe no banco de dados e `TableName(??)` seja diferente de vazio.

- O método `TUiDmxScroller_sql.AlterTable` precisa reconhecer a sintaxe do banco de dados selecionado.
  - O postgresSQL sintaxe:
    - `CREATE(??)` TABLE [IF NOT EXISTS] table_name (column1 datatype(length) column_contraint, column2 datatype(length) column_contraint, column3 datatype(length) column_contraint, table_constraints);
    - **REFERÊNCIA**

    postgresql-`create(??)`-table (https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table/)
  - O sqLite3 sintaxe:
    - `CREATE(??)` TABLE [IF NOT EXISTS] [schema_name].table_name (column_1 data_type PRIMARY KEY, column_2 data_type NOT NULL, column_3 data_type DEFAULT 0,table_constraints) [WITHOUT ROWID];
    - **REFERÊNCIA:**

    lang_createtable.html (https://www.sqlite.org/lang_createtable.html)

    sqlite-`create(??)`-table (https://www.sqlitetutorial.net/sqlite-create-table/)

    lang_createtable.html (https://www.sqlite.org/lang_createtable.html)
- **NOTAS**
  - As tabelas só são criadas automaticamente caso a constante AlterTableQL = true.

- Ao adiciona uma coluna que já exista no banco de dados o sistema trata a exceção e tenta adicionar a próxima coluna. Motivo: Poder expandir a tabela dinâmicamente.
- O comportamento do Banco de dados SqLite ao criar uma tabela é diferente do postgres.
    * O sqLite não permite criar tabela vazia.

SetSqlCustomBufDataset

Declaração `public Function SetSqlCustomBufDataset:Boolean; Virtual;`

Descrição O método `SetSqlCustomBufDataset` inicializa as propriedades SQLs de `CustomBufDataset`(**??**)

- **PROPRIEDADES OBRIGATÓRIAS SEREM INICIALIZADAS:**
    - CustomBufDataset.SQL;
- **PROPRIEDADES OPCIONAIS SEREM INICIALIZADAS:**
    - CustomBufDataset.InsertSQL;
    - CustomBufDataset.UpdataSQL;
    - CustomBufDataset.DeleteSQL;
    - CustomBufDataset.RefreshSQL;
- **GERAÇÃO AUTOMÁTICA DE INSTRUÇÃO SQL DE ATUALIZAÇÃO**
    - O **SqlDb** (mais em particular, **TSQLQuery**) pode gerar automaticamente instruções de atualização para os dados que busca. Para isso, ele irá varrer a instrução propriedade **CustomBufDataset.SQL** e determinar a tabela principal na consulta: esta é a **primeira tabela** encontrada na parte **FROM** da instrução **SELECT** .
        * Exemplo:

```
SELECT * FROM ALUNOS
```

· Alunos será a tabela selecionada para uso dos campos de https://www.freepascal.org/docs-html/fcl/db/tField.html (TField).

- Para operações **INSERT** e **UPDATE**, a propriedade instrução SQL gerada inserirá e atualizará todos os campos que possuim **pfInUpdate** em sua propriedade **TField.ProviderFlags**.

  * Os campos somente leitura não serão adicionados à instrução SQL.

  * Os campos que são NULL não serão adicionados a uma consulta de inserção, o que significa que o servidor de banco de dados inserirá o que estiver na cláusula DEFAULT da definição de campo correspondente.

- O campos de chave primária

  * Ao atualizar registros, **TSQLQuery** precisa saber quais campos compõem a chave primária que pode ser usada para atualizar o registro e quais campos devem ser atualizados: com base nessas informações, ele constrói os comandos **SQL UPDATE, INSERT ou DELETE**.

  * A construção da instrução **SQL** é controlada pela propriedade **UsePrimaryKeyAsKey** e pelas propriedades **ProviderFlags**.

  * A propriedade TField.ProviderFlag é um conjunto de 6 sinalizadores:

    · **pfInUpdate** : As alterações no campo devem ser propagadas para o banco de dados..

    · **pfInWhere** : O campo deve ser usado na cláusula WHERE de uma instrução de atualização no caso de upWhereChanged.

    · **pfInKey** : Campo é um campo chave e usado na cláusula WHERE de uma instrução de atualização.

    · **pfHidden** : O valor deste campo deve ser atualizado após a inserção.

- **pfRefreshOnInsert** : O valor deste campo deve ser atualizado após a inserção.
- **pfRefreshOnUpdate** : O valor deste campo deve ser atualizado após a atualização.

- **REFERNCIAS**

TSQLQuery Introdução (https://wiki.freepascal.org/Working_With_TSQLQuery#General)

TSQLQuery exemplos (https://wiki.freepascal.org/TSQLQuery)

- https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.html
- https://wiki.freepascal.org/Working_With_TSQLQuery (Trabalhando com TSQLQuery);
- https://www.freepascal.org/docs-html/fcl/sqldb/updatesqls.html (updatesqls.html);

CreateCustomBufDataset_FieldDefs

Declaração `public Procedure CreateCustomBufDataset_FieldDefs; override;`

Descrição O método `CreateCustomBufDataset_FieldDefs` é usado para criar os campos de `CustomBufDataset(??)`

GetTemplate

Declaração `public function GetTemplate(aNext: PSItem) : PSItem; overload; override;`

Descrição O método `GetTemplate` retorna uma lista de `PSItem(??)` (Lista de `strings`(??)) com o modelo usado para criar a tela.

- **NOTA**
  - O Evento `onGetTemplate`(??) só é iniciado em tempo de execução, por isso o formulário não pode ser criado em tempo de desenho do aplicativo.
  - Caso o evento `onGetTemplate`(??) seja nil, então não posso ativar a tela.
  - Esse método pode ser anulado, caso se queira ignorar o evento `onGetTemplate`(??) e definir o Template em uma método pai herdado desta classe.

GetBuffers

Declaração `public function GetBuffers:Boolean; Override;`

Descrição O método `GetBuffers` ler o buffer dos campos dos arquivos associados a classe `TUiDmxScroller_sql(??)` para o buffer dos campos da classe `TUiDmxScroller(??)`

PutBuffers

Declaração `public function PutBuffers:Boolean; override;`

Descrição O método `PutBuffers` grava o buffer dos campos da classe `TUiDmxScroller_sql(??)` para o buffer dos campos dos arquivos associados a classe `TUiDmxScroller_sql(??)`

SetActiveLCL

Declaração `public procedure SetActiveLCL(aActive:  Boolean); override;`

DoOnNewRecord

Declaração `public Procedure DoOnNewRecord; Override;`

Descrição O método `DoOnNewRecord` seleciona o registro para adição de um novo registro

- NOTA
  - Está gerando exceção.?????

DoAddRec

Declaração `public Function DoAddRec:Boolean; override;`

Descrição O método `DoAddRec` adicione o registro editado no banco de dados. = **OBSERVAÇÂO**

- O método `DoAddRec` só funciona se o registro atender as seguintes condições:

- `appending`(**??**) =true;
- Mb_St_Insert habilidado
- `CustomBufDataset`(**??**) <> nil
- CustomBufDataset.Active = true;

- **REFERÊNCIA**

tsqlquery.options (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.options.html)

Chapter 48

Unit mi_rtl_ui_interfaces

48.1 Descrição

- A unit `mi_rtl_ui_interfaces` é usada para implementar as interfaces do pacote mi.ui com propopósito de permitir que se possa criar as interfaces com usuário independente do pacote gráfico instalado.
  - **NOTA**
    - O IDE Lazarus cria automaticamente o número da interface. Tecla: **Crt+Alt+G**

48.2 Tipos

TEnum_HelpCtx_StrCurrentCommand_Topic_Content_run

Declaração `TEnum_HelpCtx_StrCurrentCommand_Topic_Content_run = (...);`

Descrição

Valores `HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_Indefinido`

`HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_indicator`

`HelpCtx_StrCurrentCommand_Topic_Content_run_Parameter_File`

48.3 Constantes

IITable

Declaração `IITable : TGUID = '{937B4AC1-A9B5-437C-A2ED-7EFF6CEEA919}';`

IIInputText

Declaração `IIInputText : TGUID = '{CBEFA72F-A283-4374-AED4-8A62C05335D9}';`

Chapter 49

Unit mi_rtl_ui_methods

49.1 Uses

- Classes
- SysUtils
- db
- Variants
- UTF8Process
- System.UITypes
- mi_rtl_ui_consts

49.2 Visão Geral

TUiMethods Classe

49.3 Classes, Interfaces, Objetos e Registros

TUiMethods Classe

Hierarquia

TUiMethods > TUiConsts

Métodos

CreateAppendFields

Declaração `public class function CreateAppendFields(ATemplate: ptString) : DmxIDstr;`

Descrição A class function `CreateAppendFields` é usado para encandear Templates do tipo `TString`(**??**)

CreateBlobField

Declaração `public class function CreateBlobField(Len: integer; AccMode,Default: byte) : DmxIDstr;`

Descrição A class function `CreateBlobField` é usado para encandear campos do tipo blob

CreateEnumField

Declaração `public class function CreateEnumField(ShowZ: boolean; AccMode,Default: LongInt;AItems: PSItem) : DmxIDstr;`

Descrição A class function `CreateEnumField` é usado para encandear Templates do tipo enumerado

CreateCheckBoxField

Declaração `public class function CreateCheckBoxField(CharNumberField: AnsiChar;ShowZ: boolean; AccMode,Default: byte;AItems: PSItem) : AnsiString;`

Descrição A class function `CreateCheckBoxField` é usado para encandear Templates do tipo checkbox

CreateTSItemFields

Declaração `public class function CreateTSItemFields(ATemplates: PSItem) : DmxIDstr;`

Descrição A class function `CreateTSItemFields` é usado para encandear Templates do tipo `PSItem`(**??**)

CreateOptions

Declaração `public class function CreateOptions(Default: LongInt;AItems: PSItem) : DmxIDstr;`

Descrição A class function `CreateOptions` é usado para informar uma lista de opções para o campo.

- **NOTA** O campo que pode receber uma lista pode ser de qualquer tipo, exceto os tipos:
  - `FldEnum`(??),`FldBoolean`(??) e `FldRadioButton`(??).
- **EXEMPLO DE USO**

```
with aUiDmxScroller do
begin
  add(' _EXEMPLO DE TEMPLATE_____________________________________ ');
  add('');
  add(' Vencimento: \Ssssss'+ChFN+'Vencimento'+CreateOptions(1,NewSItem('Dia 10
                                                             NewSItem('Dia 15
                                                             NewSItem('Dia 20
                                                             NewSItem('Dia 25
                                                             nil)))))+'  dias

  add('      Prazo: \BB'+ChFN+'Dias'+CreateOptions(2,NewSItem('30',
                                                   NewSItem('60',
                                                   NewSItem('90',
                                                   NewSItem('120',
                                                   nil)))))+'  dias ');

  add('');
end;
```

GetMaxTViRect

Declaração `public class Function GetMaxTViRect: TViRect;`

AnsiString_to_TCollectionString

Declaração `public class Function AnsiString_to_TCollectionString(Msg: AnsiString): TCollectionString;`

MsgDlgButtons_To_MsgDlgBtn

Declaração `public class Function MsgDlgButtons_To_MsgDlgBtn(Buttons: TMI_MsgBox.TMsgDlgButtons): TMI_MsgBox.TArray_MsgDlgBtn;`

FStrSelection

Declaração `public class function FStrSelection(S:AnsiString):AnsiString;`

EliminaTilDeTodasAsStrings

Declaração `public class Procedure EliminaTilDeTodasAsStrings(ATCollectionString: TCollectionString;Var aFrist_Item_Valid : Integer);`

GetModalResult

Declaração `public class function GetModalResult(ButtonDefault: TMI_MsgBox.TMsgDlgBtn):TModalResult;`

isValueDbChanged

Declaração `public class function isValueDbChanged(Sender: TComponent): Boolean; virtual;`

Descrição O método `isValueDbChanged` verifica se o componente fornecido tem uma relação com **db** e seu conteúdo foi alterado

FMb_Bits

Declaração `public function FMb_Bits(const aBit: Byte): Longint;`

Descrição O método `FMb_Bits` retorna o mapa de bits da posição aBit. Ou seja: a função move o bit para a esquerda aBits posição.

- **NOTA**
  - Como o mapa de bits possui 4 bytes este método gera exceção se aBit for maior que 32.
- Example:
  - Command is: 00000100 shl 2 (shift left 2 bits)
  - Action is: 00000100 <- 00 (00 gets added to the right of the value; left 00 "disappears")
  - Result is: 00010000

Chapter 50

Unit mi_rtl_ui_types

50.1 Descrição

A unit `mi_rtl_ui_types` implementa a classe `TUiTypes`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
  - **HISTÓRICO**
    - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
    - **2022-03-16**
      - **19:49**
      - Criar o tipo TStrSQL com objetivo de criar sql para qualquer banco de dados conhecido pelo sistema.

50.2 Uses

- `Classes`
- `SysUtils`
- `db`
- `mi.rtl.types`(**??**)

- mi.rtl.Consts(??)
- mi.rtl.files(??)
- mi.rtl.objects.consts.mi_msgbox
- mi.rtl.Objects.Methods(??)
- mi.rtl.objects.Methods.dates(??)
- mi.rtl.Objects.Methods.ParamExecucao.Application(??)
- mi.rtl.Objects.Methods.Exception(??)
- mi.rtl.Objectss(??)

50.3 Visão Geral

TUiTypes Classe

50.4 Classes, Interfaces, Objetos e Registros

TUiTypes Classe

Hierarquia

TUiTypes > TObjectss(??) >

Descrição

A class TUiTypes concentra todos os tipo do pacote mi.ui.

Chapter 51

Unit mi_ui_Dmxscroller_sql

51.1 Descrição

A unit `mi_ui_Dmxscroller_sql` implementa a classe `TUiDmxScroller_sql`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
  - **PENDÊNCIAS**
    - T12 Falta implementar chave estrangeira em createTable;
    - T12 Em `TUiDmxScroller_sql.DoOnNewRecord`(**??**); está executando o método (CustomBuf-Dataset as TSQLQuery).Append; antes do componenente `TUiDmxScroller_sql`(**??**) está visível e isto está gerando exceção.
    - T12 ANÁLISE

Estudar os procedimentos armazenados (https://www.w3schools.com/sql/sql_stored_procedures.asp)

Estudar as restrições SQL (https://www.w3schools.com/sql/sql_constraints.asp)

- Como saber se um campo é uma chave que liga outra tabela?
- https://www.w3schools.com/sql/sql_foreignkey.asp (SQL FOREIGN KEY Constraint)

```
/*Não, podemos permitir que os registros das pessoas que possuim camisetas
  lavando sejam apagados, para garantir a integridade da informação.
   Para isso devemos utilizar o as chaves estrangeiras que acusarão
  um erro quando tentarmos deletar uma pessoa que possuir camisetas.
   Veja em código:
 */

CREATE TABLE Pessoa(
    IdPessoa INT NOT NULL PRIMARY KEY IDENTITY(1,1),
    Nome VARCHAR(20) NOT NULL
)

CREATE TABLE Camiseta(
    IdCamiseta INT NOT NULL PRIMARY KEY IDENTITY(1,1),
    Descrição VARCHAR(20) NOT NULL,
    IdPessoa INT NOT NULL
    CONSTRAINT FK_Camiseta_Pessoa FOREIGN  KEY(IdPessoa) REFERENCES Pessoa(IdPessoa)
)

INSERT INTO Pessoa VALUES ('HeyJoe')
INSERT INTO Pessoa VALUES ('Caique')


INSERT INTO Camiseta VALUES ('Azul', 1)
INSERT INTO Camiseta VALUES ('Amarela', 1)
INSERT INTO Camiseta VALUES ('Preta', 2)

SELECT * FROM Pessoa, Camiseta WHERE Pessoa.IdPessoa = Camiseta.IdPessoa
```

  - Como saber o tipo de relacionamento que os campos de outra tabela tem com a tabela atual?

* T12 A opção CreateTable está dando mensagem de erro quando a coluna já existe.

  - Encontrar uma forma de não gerar exceção ou ignorar as exceções nesta rotina.

* T12 Em `TUiDmxScroller_sql.AlterTable`(**??**) checar:

  - T12 Criar código para todos os tipos reconhecidos por marIcaraí.

  - T12 Debugar para saber se está tudo funcionando.

  - T12 Permitir adicionar uma nova coluna mesmo que a tabela já exista.

    * T12 Em SetTableName(aTableName:String) criticar o nome aTableName é um nome válido para a tabela.

– **HISTÓRICO**

    * Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)

    * **2022-03-13**

        · **09:00**

        · T12 Implementar a criação de código SQL baseado nos dados de `TUiDmxScroller`(**??**).

        · **ANÁLISE:**

        · Onde pegar o nome da tabela ou consulta?

        · TUiDmxScroller.CustomBufDataset.FileName;

        · Onde pegar o nome dos campos da tabela CustomBufDataset.Filename?

        · A lista de campos encontra-se em : TUiDmxScroller_Atributos.Fields : TFPList;

        · Como saber se `TUiDmxScroller`(**??**) é uma tabela ou a uma consulta?

        · Se todos os TUiDmxScroller_Atributos.Fields[].FieldName não contém '—' é porque é FileName é uma tabela.

        · Se pelo menos um TUiDmxScroller_Atributos.Fields[].FieldName contém '—' é porque é FileName é uma consulta envolvendo mais de uma tabela.

        · Como saber se uma tabela ou consulta existe do banco de dados?

        · O SQL do **postegres** e do **sqlite3** tem a clausula **IN NOT EXISTS** no comando CREATE TABLE:

        · EXEMPLO:

```
CREATE TABLE IF NOT EXISTS TEST01 ();
```

        · **REFERÊNCIAS**

        · https://en.wikipedia.org/wiki/SQL:2016 (SQL:2016)

        · (PostgresSQL aceita 160 das 169 especificação 2016)(https://www.postgresql.org/docs/12/features.htn

bancos de dados x conformidade SQL (https://en.wikipedia.org/wiki/SQL_compliance)

lientes de bancos de dados opensource (https://medevel.com/17-sql-client-open-source/)

- https://dbeaver.io/ (Instalei programa cliente SQL DBeaver)
- Obs: Não deu certo. Ele é escrito em java e não funcionou o básico.

sqlite create database if not exists (https://www.codegrepper.com/code-examples/sql/sqlite+create+database+if+not+exists)

* **2022-03-14**
  - **08:22**
  - T12 Criar a unit mi_ui_Dmxscroller_sql.pas com a classe `TUiDmxScroller_sql`(**??**) com objetivo de concentrar a integração do TDmxScroller com o componente **TSQLQuery**
  - **20:00**
  - T12 Na Construção de TFields atualizar a propriedade **TField.ProviderFlags** com o tipo de acesso definido em `TDmxFieldRec.Access`(**??**)
  - **21:12**
  - T12 Criar propriedade **TableName**
  - **21:27**
  - T12 Criar Function SetSqlCustomBufDataset:Boolean;Virtual;
  - CustomBufDataset.SQL := **SELECT * FROM X** onde X será definido pela propriedade **TableName**
* **2022-03-15**
  - **09:11**
  - Depurar o que fiz ontem para fazer funciona a atualização do banco de dados SQL.
  - **11:36**
  - Criar método `TUiDmxScroller_sql.AlterTable`(**??**) : Boolean;
  - **14:38**
  - T12 Atualizar TSQLQury.TFields.ProviderFlags com TUiDmxScroller.MiProviderFlags
* **2022-03-16**
  - **16:23**
  - T12 Em `TUiDmxScroller_sql.CreateCustomBufDataset_FieldDefs`(**??**), atualizar **TField.ProviderFlags** com os dados do campo `TDmxFieldRec.ProviderFlags`(**??**).

- **16:54**
- Em `TUiDmxScroller_sql.AlterTable`(**??**) usar os flags `TDmxFieldRec.ProviderFlags`(**??**) para criação da tabela.

* **2022-03-17**
  - **10:48**
  - T12 Os flags indicando que se trata de chave primária não está sendo atualizado em createStructor, por isso não está criando a chave primária.

* **2022-03-18**
  - **10:40**
  - T12 Ao criar uma tabela SQL em **AlterTable** adicionar colunas ao invés de criar a tabela toda.
  - **Motivo**:
  - Permitir que o banco de dados fique compatível com o Template.
  - Alterar um coluna de forma automática não é bom, porque o que está feito gera dependências que produzirão erros ao fazer essas alterações.

* **2022-03-21**
  - **08:57**
  - T12 Criar function SQL_AddkeysPrimaryKeyComposite(I : Integer):Boolean;
  - Esta função adiciona chave primária composta na tabela.
  - **REFERÊNCIA**

a usando a expressão ALTER TABLE (https://www.techonthenet.com/postgresql/primary_keys.php#: :text=In%20PostgreSQL%2C%20a%

  - **15:40**
  - T12 Em AlterTable criar a restrição de chave estrangeira no TDmxScroller_sql.
  - Nome da função: function AddKeyForeigns(I : Integer):Boolean;

* **2022-03-22**
  - **09:00**
  - T12 Documentar as units `TuiTypes`(**??**) e TUIConsts.
  - **10:00**

- T12 Criar os relacionamentos entre tabelas (restrições entre tabelas)
- **14:14**
- T12 Depurar os relacionamentos entre tabelas.
- **18:47**
- O Componente CustomBufDataset não está entrando no modo edit.
- O problema estava nos eventos TScrollBoxDMX.DoOnEnter e TScrollBoxDMX.DoOnExit;]

* **2022-03-22**

  - **20:27**
  - T12 Analisar como criar os comandos CmIncluir, cmAlterar, cmExcluir, cmConsulta para a tabela TDmxScroller
  - Criar os comandos:
  - Public Procedure DoOnNewRecord;overload;override; //Usado para inicializa os parametros de um novo registro
  - Public Procedure PutRec;Override;//Grava o buffer no arquivo memo
  - Public Procedure GetRec;Override;//O primeiro registro esta gravado em Value
  - Public Function DeleteRec:Boolean;Override;
  - Function UpdateRec: Boolean;Override;
  - Function UpdateRec_if_RecordAltered:Boolean;Override;
  - Function PrevRec : Boolean;overload;override;
  - Function NextRec : Boolean;overload;override;

* **2022-03-23**

  - Criar método Public Function AddRec:Boolean;Override;
  - Para que DoAddrec possa adicionar o registro é necessário que o registro esteja selecionando, ou seja no modo edit.
  - Obs: Está com problema.

* **2022-03-25**

  - https://wiki.freepascal.org/Firebird#Creating_objects_programmatically (Estudar página sobre o banco de dados firebird)

* **2022-03-28**

  - Em `TUiDmxScroller_sql.DoOnNewRecord(??)`; está executando o método (CustomBufDataset as TSQLQuery).Append; antes do componenente `TUiDmxScroller_sql(??)` está visível e isto está gerando exceção. -

* **2022-03-30**

  - Implementar a conexão com o banco de dados usando o componente Mi_Application.

* **2022-04-14**

  - Debugar o método `TUiDmxScroller_sql.AlterTable(??)`.

* **2022-04-15**

  - O método `TUiDmxScroller_sql.AlterTable(??)` precisa reconhecer a sintaxe do banco de dados selecionado.

  - O postgresSQL sintaxe:

  - CREATE TABLE [IF NOT EXISTS] table_name (column1 datatype(length) column_contraint, column2 datatype(length) column_contraint, column3 datatype(length) column_contraint, table_constraints);

  - **REFERÊNCIA**

postgresql-create-table (https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table/)

  - O sqLite3 sintaxe:

  - CREATE TABLE [IF NOT EXISTS] [schema_name].table_name (column_1 data_type PRIMARY KEY, column_2 data_type NOT NULL, column_3 data_type DEFAULT 0,table_constraints) [WITHOUT ROWID];

  - **REFERÊNCIA:**

sqlite-create-table (https://www.sqlitetutorial.net/sqlite-create-table/)

51.2 Uses

- `Classes`
- `SysUtils`
- `BufDataset`
- `db`

- SqlDb
- mi.rtl.Types(??)
- mi_ui_types
- mi_ui_consts
- mi_ui_Dmxscroller
- uMi_ui_custom_application

51.3 Visão Geral

TDmxScroller_sql_Atributos Classe

TUiDmxScroller_sql Classe

51.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_sql_Atributos Classe

Hierarquia

TDmxScroller_sql_Atributos > TUiDmxScroller(??) > TUiMethods(??) > TUiConsts

Descrição

A class TDmxScroller_sql_Atributos contém os atributos da class TDmxScroller_sql

Campos

CustomBufDataset

Declaração `public CustomBufDataset: TCustomBufDataset;`

Descrição O atributo pública CustomBufDataset é definida em **CreateCustomBufDataset_FieldDefs** que é executado em **TDmxScroller.CreateData** baseado na estrutura do Template passado por GetTemplate(??).

- **NOTA**
  - O atributo CustomBufDataset deve ser passado por **DataSource.DataSet**.
  - Em **CreateCustomBufDataset_FieldDefs** é criado os campo da propriedade CustomBufDataset se a propriedade (DataSource(??)<>nil) e (DataSource.DataSet <> nil).

- Se a propriedade DataSource.DataSet = nil então a propriedade `CustomBufDataset`=**nil**
- O método **CreateCustomBufDataset_FieldDefs** reconhece duas possibilidade para os descendentes de `CustomBufDataset` quais sejam:

TBufDataset (https://www.freepascal.org/docs-html/fcl/bufdataset/tbufdataset.

2. https://www.freepascal.org/docs-html/fcl/sqldb/tcustomsqlquery.h (TCustomSQLQuery)
    * Preciso das propriedades de acesso a banco de dados SQL.
    * O evento `OnGetTemplate`(**??**) deve setar as propriedades customizadas de **TCustomSQLQuery**.

- **REFERNCIA**:

tcustombufdataset (https://www.freepascal.org/daily/packages/fcl-db/bufdataset/tcustombufdataset 14.html)

tcustomsqlquery (https://www.freepascal.org/docs-html/fcl/sqldb/tcustomsqlquery.html)

- https://www.freepascal.org/docs-html/fcl/bufdataset/tcustombufdataset.html (TCustomBufDataset);

TBufDataSet (https://wiki.freepascal.org/How_to_write_in-memory_database_applications_in_La

tstatementtype.html (https://www.freepascal.org/docs-html/fcl/sqltypes/tstatementtype.html)

tsqlquery (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.html)

tdatasetstate (https://www.freepascal.org/docs-html/fcl/db/tdatasetstate.html)

How_to_connect_to_a_database_server (https://wiki.freepascal.org/SqlDBHowto#How_to_connect_to_a_database_server.3

Example:_reading_data_from_a_table (https://wiki.freepascal.org/SqlDBHowto#Example:_reading_data_from_a_table)

How_to_execute_direct_queries.2Fmake_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_execute_direct_queries.2Fmake

How_to_read_data_from_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_read_data_from_a_table.3F)

Why_does_TSQLQuery.RecordCount_always_return (https://wiki.freepascal.org/SqlDBHowto#Why_does_TSQLQuery.RecordCount_a

Como usar SQLDb no Lazarus (https://wiki.freepascal.org/SqlDBHowto#Lazarus)

Trabalhando com tabelas relacionadas (https://wiki.freepascal.org/MasterDetail)

How_to_change_data_in_a_table (https://wiki.freepascal.org/SqlDBHowto#How_to_change_data_in_a_table.3F)

How_does_SqlDB_send_the_changes_to_the_database_server (https://wiki.freepascal.org/SqlDBHowto#How_does_SqlDB_send_the_changes_to_

How_to_handle_Errors (https://wiki.freepascal.org/SqlDBHowto#How_to_handle_Errors)

How_to_execute_a_query_using_TSQLQuery (https://wiki.freepascal.org/SqlDBHowto#How_to_execute_a_query_using_TSQLQ

How_to_use_parameters_in_a_query (https://wiki.freepascal.org/SqlDBHowto#How_to_use_parameters_in_a_query.3F)

Select_query (https://wiki.freepascal.org/SqlDBHowto#Select_query)

Exemplo de SQLQuery com parãmetros (https://wiki.freepascal.org/SqlDBHowto#Example)

- https://wiki.freepascal.org/SqlDBHowto#Troubleshooting:_TSQLConnection_log (Troubleshooting:_TSQLConnection_logging)

Exemplo de log (https://wiki.freepascal.org/SqlDBHowto#FPC_.28or:_the_manual_

TUiDmxScroller_sql Classe

Hierarquia

TUiDmxScroller_sql > `TDmxScroller_sql_Atributos(??)` > `TUiDmxScroller(??)` > `TUiMethods(??)` > TUiConsts

Descrição

A classe `TUiDmxScroller_sql` implementa o acesso ao banco de dados usando o atributo `CustomBufDataset(??)`

- **NOTA**
  - O atributo `CustomBufDataset(??)` pode ser **TBufDataset** não conectado a banco de dados sql e **TCustomSQLQuery** conectado ao banco de dados SQL.
- **REFERÊNCIA**

Working_With_TSQLQuery (https://wiki.freepascal.org/Working_With_TSQLQuery)

Parameters_in_TSQLQuery (https://wiki.freepascal.org/Working_With_TSQLQuery#Parameters_in_TSQLQuery.SQL)

sql-basico (https://www.devmedia.com.br/sql-basico/28877)

Propriedades

DataSource

Declaração `published property DataSource : TDataSource Read _DataSource Write _DataSource;`

Descrição A propriedade `DataSource` permite que controles da **LCL** (Lazarus Componenents Library) possam usar os dados do componenente **TDmxScroller**.

- **NOTA**
  - Essa integração permite que **TDmxScroller** utilize todos os componentes de banco de dados do Free Pascal.

Campos

_DataSource

Declaração `protected _DataSource: TDataSource;`

Métodos

SetDataBase

Declaração `protected procedure SetDataBase;`

Create

Declaração `public constructor Create(aOwner:TComponent); Override;`

Descrição Constrói o componente

GetkeysPrimaryComposite

Declaração `public function GetkeysPrimaryComposite(I : Integer):AnsiString;`

Descrição O método `GetkeysPrimaryComposite` retorna a lista de campos pertencentes a chave composta primária.

GetKeysPrimary

Declaração `public function GetKeysPrimary:AnsiString;`

Descrição A função `GetKeysPrimary` retorna a chave primária composta ou não na tabela.

- **Como TSQLQuery trata os campos de chave primária**
  - Ao atualizar registros, TSQLQuery precisa saber quais campos compõem a chave primária que pode ser usada para atualizar o registro e quais campos devem ser atualizados: com base nessas informações, ele constrói um comando SQL UPDATE, INSERT ou DELETE.
  - A construção da instrução SQL é controlada pela propriedade UsePrimaryKeyAsKey e pelas propriedades ProviderFlags .
  - A propriedade Providerflags é um conjunto de 3 sinalizadores:
    - pfInkey : O campo faz parte da chave primária
    - pfInWhere : O campo deve ser utilizado na cláusula WHERE das instruções SQL.
    - pfInUpdate : Atualizações ou inserções devem incluir este campo. Por padrão, ProviderFlags consiste apenas em pfInUpdate .
    - **NOTA***
      - **Se sua tabela tiver uma chave primária (conforme descrito acima), você só precisará definir a propriedade **UsePrimaryKeyAsKey**

      como True e tudo será feito para você. Isso definirá o sinalizador pfInKey para os campos de chave primária.

- **REFERÊNCIA**

Working With TSQLQuery e Primary_key_Fields (https://wiki.freepascal.org/Working_With_TSQLQuery#Primary_key_Fields)

CreateTable

Declaração `public Function CreateTable: Boolean;`

Descrição A função `CreateTable` cria a tabela se a mesma não existir

AlterTable

Declaração `public Function AlterTable:  Boolean; Virtual;`

Descrição O método `AlterTable` cria a tabela ou consulta `TableName(??)` no banco de dados caso a propriedade `TableName(??)` não existe no banco de dados e `TableName(??)` seja diferente de vazio.

- O método `TUiDmxScroller_sql.AlterTable` precisa reconhecer a sintaxe do banco de dados selecionado.
  - O postgresSQL sintaxe:
    - `CREATE(??)` TABLE [IF NOT EXISTS] table_name (column1 datatype(length) column_contraint, column2 datatype(length) column_contraint, column3 datatype(length) column_contraint, table_constraints);
    - **REFERÊNCIA**

    postgresql-`create(??)`-table (https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-create-table/)
  - O sqLite3 sintaxe:
    - `CREATE(??)` TABLE [IF NOT EXISTS] [schema_name].table_name (column_1 data_type PRIMARY KEY, column_2 data_type NOT NULL, column_3 data_type DEFAULT 0,table_constraints) [WITHOUT ROWID];
    - **REFERÊNCIA:**

    lang_createtable.html (https://www.sqlite.org/lang_createtable.html)

    sqlite-`create(??)`-table (https://www.sqlitetutorial.net/sqlite-create-table/)

    lang_createtable.html (https://www.sqlite.org/lang_createtable.html)
- **NOTAS**
  - As tabelas só são criadas automaticamente caso a constante AlterTableQL = true.

- Ao adiciona uma coluna que já exista no banco de dados o sistema trata a exceção e tenta adicionar a próxima coluna. Motivo: Poder expandir a tabela dinâmicamente.
- O comportamento do Banco de dados SqLite ao criar uma tabela é diferente do postgres.
    * O sqLite não permite criar tabela vazia.

SetSqlCustomBufDataset

Declaração `public Function SetSqlCustomBufDataset:Boolean; Virtual;`

Descrição O método `SetSqlCustomBufDataset` inicializa as propriedades SQLs de `CustomBufDataset`(**??**)

- **PROPRIEDADES OBRIGATÓRIAS SEREM INICIALIZADAS:**
    - CustomBufDataset.SQL;
- **PROPRIEDADES OPCIONAIS SEREM INICIALIZADAS:**
    - CustomBufDataset.InsertSQL;
    - CustomBufDataset.UpdataSQL;
    - CustomBufDataset.DeleteSQL;
    - CustomBufDataset.RefreshSQL;
- **GERAÇÃO AUTOMÁTICA DE INSTRUÇÃO SQL DE ATUALIZAÇÃO**
    - O **SqlDb** (mais em particular, **TSQLQuery**) pode gerar automaticamente instruções de atualização para os dados que busca. Para isso, ele irá varrer a instrução propriedade **CustomBufDataset.SQL** e determinar a tabela principal na consulta: esta é a **primeira tabela** encontrada na parte **FROM** da instrução **SELECT** .
        * Exemplo:

```
SELECT * FROM ALUNOS
```

            · Alunos será a tabela selecionada para uso dos campos de https://www.freepascal.org/docs-html/fcl/db/tField.html (TField).

- Para operações **INSERT** e **UPDATE**, a propriedade instrução SQL gerada inserirá e atualizará todos os campos que possuim **pfInUpdate** em sua propriedade **TField.ProviderFlags**.

    * Os campos somente leitura não serão adicionados à instrução SQL.

    * Os campos que são NULL não serão adicionados a uma consulta de inserção, o que significa que o servidor de banco de dados inserirá o que estiver na cláusula DEFAULT da definição de campo correspondente.

- O campos de chave primária

    * Ao atualizar registros, **TSQLQuery** precisa saber quais campos compõem a chave primária que pode ser usada para atualizar o registro e quais campos devem ser atualizados: com base nessas informações, ele constrói os comandos **SQL UPDATE, INSERT ou DELETE**.

    * A construção da instrução **SQL** é controlada pela propriedade **UsePrimaryKeyAsKey** e pelas propriedades **ProviderFlags**.

    * A propriedade TField.ProviderFlag é um conjunto de 6 sinalizadores:

        · **pfInUpdate** : As alterações no campo devem ser propagadas para o banco de dados..

        · **pfInWhere** : O campo deve ser usado na cláusula WHERE de uma instrução de atualização no caso de upWhereChanged.

        · **pfInKey** : Campo é um campo chave e usado na cláusula WHERE de uma instrução de atualização.

        · **pfHidden** : O valor deste campo deve ser atualizado após a inserção.

- **pfRefreshOnInsert** : O valor deste campo deve ser atualizado após a inserção.
- **pfRefreshOnUpdate** : O valor deste campo deve ser atualizado após a atualização.

- **REFERNCIAS**

TSQLQuery Introdução (https://wiki.freepascal.org/Working_With_TSQLQuery#General)

TSQLQuery exemplos (https://wiki.freepascal.org/TSQLQuery)

- https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.html
- https://wiki.freepascal.org/Working_With_TSQLQuery (Trabalhando com TSQLQuery);
- https://www.freepascal.org/docs-html/fcl/sqldb/updatesqls.html (updatesqls.html);

CreateCustomBufDataset_FieldDefs

Declaração `public Procedure CreateCustomBufDataset_FieldDefs; override;`

Descrição O método `CreateCustomBufDataset_FieldDefs` é usado para criar os campos de `CustomBufDataset(??)`

GetTemplate

Declaração `public function GetTemplate(aNext: PSItem) : PSItem; overload; override;`

Descrição O método `GetTemplate` retorna uma lista de `PSItem(??)` (Lista de `strings`(??)) com o modelo usado para criar a tela.

- **NOTA**
  - O Evento `onGetTemplate`(??) só é iniciado em tempo de execução, por isso o formulário não pode ser criado em tempo de desenho do aplicativo.
  - Caso o evento `onGetTemplate`(??) seja nil, então não posso ativar a tela.
  - Esse método pode ser anulado, caso se queira ignorar o evento `onGetTemplate`(??) e definir o Template em uma método pai herdado desta classe.

GetBuffers

Declaração `public function GetBuffers:Boolean; Override;`

Descrição O método `GetBuffers` ler o buffer dos campos dos arquivos associados a classe `TUiDmxScroller_sql(??)` para o buffer dos campos da classe `TUiDmxScroller(??)`

PutBuffers

Declaração `public function PutBuffers:Boolean; override;`

Descrição O método `PutBuffers` grava o buffer dos campos da classe `TUiDmxScroller_sql(??)` para o buffer dos campos dos arquivos associados a classe `TUiDmxScroller_sql(??)`

SetActiveLCL

Declaração `public procedure SetActiveLCL(aActive:  Boolean); override;`

DoOnNewRecord

Declaração `public Procedure DoOnNewRecord; Override;`

Descrição O método `DoOnNewRecord` seleciona o registro para adição de um novo registro

- NOTA
  - Está gerando exceção.?????

DoAddRec

Declaração `public Function DoAddRec:Boolean; override;`

Descrição O método `DoAddRec` adicione o registro editado no banco de dados. = **OBSERVAÇÂO**

- O método `DoAddRec` só funciona se o registro atender as seguintes condições:

- `appending`(**??**) =true;
- Mb_St_Insert habilidado
- `CustomBufDataset`(**??**) <> nil
- CustomBufDataset.Active = true;

- **REFERÊNCIA**

tsqlquery.options (https://www.freepascal.org/docs-html/fcl/sqldb/tsqlquery.options.html)

Chapter 52

Unit mi_ui_mi_msgbox_dm

52.1 Uses

- Classes
- SysUtils
- Dialogs
- Graphics
- StdCtrls
- System.UITypes
- mi.rtl.objects.consts.mi_msgbox

52.2 Visão Geral

TMi_ui_mi_msgBox Classe

52.3 Classes, Interfaces, Objetos e Registros

TMi_ui_mi_msgBox Classe

Hierarquia

TMi_ui_mi_msgBox > TDataModule

Descrição

TMi_ui_mi_msgBox

- EXEMPLO

```
Var
  S : String[10] = '';
begin
   if MI_MsgBox.InputBox('InputBox Test','Gual a sua indade? ',s,'sssssssss') = Mrok
   then ShowMessage('Sua idade é: 's);
```

Campos

MI_MsgBox1

Declaração `public MI_MsgBox1: TMI_MsgBox;`

Métodos

MI_MsgBox1InputBox

Declaração `public function MI_MsgBox1InputBox(const aTitle, ALabel: AnsiString; var Buff; Template: AnsiString): TModalResult;`

MI_MsgBox1InputPassword

Declaração `public function MI_MsgBox1InputPassword(const aTitle: AnsiString; var aPassword: AnsiString): TModalResult;`

MI_MsgBox1InputValue

Declaração `public function MI_MsgBox1InputValue(const aTitle, aLabel: AnsiString; var aValue: Variant): TModalResult;`

MI_MsgBox1MessageBox

Declaração `public function MI_MsgBox1MessageBox(const aMsg: AnsiString): TModalResult;`

MI_MsgBox1MessageBox_03

Declaração `protected function MI_MsgBox1MessageBox_03(const aMsg: AnsiString;DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TModalResult;`

MI_MsgBox1MessageBox_04

Declaração `protected function MI_MsgBox1MessageBox_04(aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn): TModalResult;`

MI_MsgBox1MessageBox_04_PSItem

Declaração `protected function MI_MsgBox1MessageBox_04_PSItem(aPSItem: TMI_MsgBoxTypes.PSItem; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn ): TModalResult;`

MI_MsgBox1MessageBox_05

Declaração `protected function MI_MsgBox1MessageBox_05(ATitle: AnsiString; aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn ): TModalResult;`

MI_MsgBox1MessageBox_ListBoxRec_PSItem

Declaração `protected function MI_MsgBox1MessageBox_ListBoxRec_PSItem(Atitulo: AnsiString; APSItem: TMI_MsgBoxTypes.PSItem; itemSelection: longint; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn ): TModalResult;`

create

Declaração `public constructor create(aOwner:TComponent); Override;`

52.4 Variáveis

Mi_ui_mi_msgBox

Declaração `Mi_ui_mi_msgBox: TMi_ui_mi_msgBox;`

Chapter 53

Program project1

53.1 Uses

- `Interfaces`
- `Forms`
- `Unit1`(??)

Chapter 54

Unit testForm

54.1 Descrição

A unit `testForm` implementa o formulário `TMI_UI_InputBox`(**??**) usado para criar formulário baseado em Template `PSITem`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.693
- **CÓDIGO FONTE**:
  -
- **PENDÊNCIAS**
  - T12 A classe Mi_ScrollBox_LCL1 deve ser criada em tempo de execução para quenão tenha problema na instalação.
  - T12 A a classe DmxScroller_Form_Lcl1 deve ser criada em tempo de execução para que não tenha problema na instalação.
  - T12 A a classe ButtonPanel1 deve ser criada em tempo de execução para que não tenha problema na instalação.
  - T12 A propriedade autosize deve ser true após o form for criado.
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
  - **2022-05-17**
    - T12 Análise de como será a classe `TMI_UI_InputBox`(**??**).

    * T12 Criar a unit `testForm`.
    * T12 Criar formulário `TMI_UI_InputBox`(**??**);
    * T12 Adicionar o componente ButtonPanel1 e habilitar os botões ok e cancel;
    * T12 Adicionar o componente Mi_ScrollBox_LCL1 ;
    * T12 Criar evento: function DmxScroller_Form_Lcl1GetTemplate;
    * T12 Criar atributo protected _FormSItem : `PSitem`(**??**);
    * T12 Criar propriedade Template:AnsiString;
        - Criar método Set_Template(aTemplate:AnsiString);

- **2022-05-18**
    * **10:51**
        - As alterações que fiz ontem no método `TObjectsMethods.StringToSItem`(**??**)() criou efeito colateral.
        - Corrigido.
    * **14:28**
        - Criar função:
        - function `InputBox`(**??**)(): TModalResult;

- **2022-05-19**
    * **11:13**
        - Criar os eventos
        - OnEnterLocal
        - OnExistLocal
        - onEnterFieldLocal
        - OnExitFieldLocal
        - Criar função:
        - function MI_MsgBox1MessageBox_04_PSItem(aPSItem: TMI_MsgBoxTypes.PSItem; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn): TModalResult;

- **2022-06-27**
  - **09:30**
    - T12 A classe Mi_ScrollBox_LCL1 deve ser criada em tempo de execução para quenão tenha problema na instalação. .
  - **10:25**
    - T12 A a classe DmxScroller_Form_Lcl1 deve ser criada em tempo de execução para que não tenha problema na instalação.
  - **10:41**
    - T12 A a classe ButtonPanel1 deve ser criada em tempo de execução para que não tenha problema na instalação.
- **2022-06-28**
  - **15:52**
    - Criar unit `umi_ui_inputbox_lcl_test(??)` para desmostrar o uso de `InputBox(??)`.

54.2 Uses

- `Classes`
- `SysUtils`
- `Forms`

54.3 Visão Geral

`ShowFormF`

54.4 Funções e Procedimentos

ShowFormF

Declaração `Procedure ShowFormF;`

54.5 Variáveis

f

Declaração `f:Tform;`

Chapter 55

Unit uDmxScroller_Form_Lcl_add_test

55.1 Descrição

A unit `uDmxScroller_Form_Lcl_add_test` implementa a classe TTForm1 do pacote mi.ui.

55.2 Uses

- `Classes`
- `SysUtils`
- `Forms`
- `Controls`
- `Graphics`
- `Dialogs`
- `LMessages`
- `StdCtrls`
- `DBCtrls`
- `ExtCtrls`
- `ButtonPanel`
- `uMi_ui_DmxScroller_Form_Lcl`
- `uMi_ui_scrollbox_lcl`(??)
- `mi_rtl_ui_Dmxscroller`(??)

55.3 Visão Geral

TDmxScroller_Form_Lcl_add_test Classe

55.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_add_test Classe

Hierarquia

TDmxScroller_Form_Lcl_add_test > TForm

Campos

ButtonPanel1

Declaração `public ButtonPanel1: TButtonPanel;`

DmxScroller_Form_Lcl1

Declaração `public DmxScroller_Form_Lcl1: TDmxScroller_Form_Lcl;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

ButtonPanel1Click

Declaração
```
public procedure ButtonPanel1Click(Sender: TObject);
```

DmxScroller_Form_Lcl1AddTemplate

Declaração
```
public procedure DmxScroller_Form_Lcl1AddTemplate(const
 aUiDmxScroller: TUiDmxScroller );
```

DmxScroller_Form_Lcl1Enter

Declaração `public procedure DmxScroller_Form_Lcl1Enter(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form_Lcl1EnterField

Declaração `public procedure DmxScroller_Form_Lcl1EnterField(aField: pDmxFieldRec);`

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

destroy

Declaração `public destructor destroy; override;`

55.5 Variáveis

DmxScroller_Form_Lcl_add_test

Declaração `DmxScroller_Form_Lcl_add_test: TDmxScroller_Form_Lcl_add_test;`

Chapter 56

Unit uDmxScroller_Form_Lcl_add_test2

56.1 Uses

- Classes
- SysUtils
- Forms
- Controls
- Graphics
- Dialogs
- ButtonPanel
- uMi_ui_scrollbox_lcl(??)
- uMi_ui_DmxScroller_Form_Lcl
- mi_rtl_ui_Dmxscroller(??)

56.2 Visão Geral

TDmxScroller_Form_Lcl_add_test2 Classe

56.3 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_add_test2 Classe

Hierarquia

TDmxScroller_Form_Lcl_add_test2 > TForm

Campos

ButtonPanel1

Declaração `public ButtonPanel1: TButtonPanel;`

DmxScroller_Form_Lcl1

Declaração `public DmxScroller_Form_Lcl1: TDmxScroller_Form_Lcl;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

DmxScroller_Form_Lcl1GetTemplate

Declaração
```
public function DmxScroller_Form_Lcl1GetTemplate(aNext: PSItem):
 PSItem;
```

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

56.4 Variáveis

DmxScroller_Form_Lcl_add_test2

Declaração `DmxScroller_Form_Lcl_add_test2: TDmxScroller_Form_Lcl_add_test2;`

Chapter 57

Unit uDmxScroller_Form_Lcl_test

57.1 Descrição

A unit `uDmxScroller_Form_Lcl_test` implementa o teste dos componentes TUiConsts.MI_MsgBox, mi_scrollbox_LCL1 e TDmxScroller_Form_Lcl onde os mesmos são ligados no evento `TDmxScroller_Form_Lcl_test.FormCreate(??)`

- **NOTAS**
  - A constante **TUiConsts.MI_MsgBox** precisa se iniciada com o atributo `TMi_ui_mi_msgBox.MI_MsgBox1(??)` da unit `mi_ui_mi_msgbox_dm(??)` para que os diálogos internos do componente **DmxScroller_Form_Lcl1** possa gerar mensagens sem depender diretamente da **LCL**, ou seja: Será possível implementar dialogs em outros frameworks visuais tais como **html**, **angula 4**, etc alterando o método SetActive().
    - O método SetActive seleciona os método DmxScroller_Form_Lcl1.CreateFormLCL ou o método DmxScroller_Form_Lcl1.CreateFormHTML conforme o tipo de aplicação.
  - O evento DmxScroller_Form_Lcl1.onGetTemplate precisa se iniciado em OnCreate do form porque a propriedade **onGetTemplate** ainda não foi lida do arquivo de recursos e precisamos da mesma para executar o método DmxScroller_Form_Lcl1.SetParentLcl.
- **CÓDIGO PASCAL**

```
procedure TForm_Mi_Ui_Test.FormCreate(Sender: TObject);
begin
   TUiConsts.MI_MsgBox := get_MI_MsgBox.MI_MsgBox1;
   DmxScroller_Form_Lcl1.onGetTemplate:= DmxScroller_Form_Lcl1GetTemplate;
   DmxScroller_Form_Lcl1.SetParentLcl(mi_scrollbox_LCL1);
end;
```

57.2 Uses

- Classes
- SysUtils
- DB
- BufDataset
- memds
- Forms
- Controls
- Graphics
- Dialogs
- typInfo
- MaskEdit
- StdCtrls
- ExtCtrls
- DBGrids
- ButtonPanel
- ActnList
- DBCtrls
- Spin
- Buttons
- DBExtCtrls
- EditBtn
- SpinEx
- SynEdit
- TAChartExtentLink
- SQLite3Conn
- SqlDb
- mi.rtl.Types(??)

- mi_rtl_ui_consts
- mi_rtl_ui_Dmxscroller(??)
- uMi_ui_scrollbox_lcl(??)
- uMi_Ui_DbComboBox_lcl(??)
- uMI_ui_DbEdit_LCL(??)
- uMi_ui_maskedit_lcl(??)
- uMi_ui_ComboBox_LCL(??)
- uMi_BitBtn_LCL
- uMi_ui_DmxScroller_Form_Lcl
- uMi_ui_mi_msgbox_dm(??)
- uMi_ui_DmxScroller_Form_Lcl_DS_Test(??)
- umi_ui_InputBox_lcl(??)
- uDmxScroller_Form_Lcl_add_test(??)
- uMi_ui_DmxScroller_Form_Lcl_ds_test2(??)
- umi_ui_inputbox_lcl_test(??)
- uDmxScroller_Form_Lcl_add_test2(??)

57.3 Visão Geral

TDmxScroller_Form_Lcl_test Classe

57.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_test Classe

Hierarquia

TDmxScroller_Form_Lcl_test > TForm

Campos

Form_ds_test2

Declaração `public Form_ds_test2: TAction;`

Action_Form_ds_test

Declaração `public Action_Form_ds_test: TAction;`

AddTemplate

Declaração `public AddTemplate: TButton;`

Button1

Declaração `public Button1: TButton;`

GetTemplate

Declaração `public GetTemplate: TButton;`

Novo

Declaração `public Novo: TAction;`

Gravar

Declaração `public Gravar: TAction;`

Excluir

Declaração `public Excluir: TAction;`

Pesquisar

Declaração `public Pesquisar: TAction;`

Pesquisa

Declaração `public Pesquisa: TAction;`

ActionList1

Declaração `public ActionList1: TActionList;`

Button_ModifyFontsAll_LCL

Declaração `public Button_ModifyFontsAll_LCL: TButton;`

InputBox

Declaração `public InputBox: TButton;`

form_ds_Test

Declaração `public form_ds_Test: TButton;`

Button_Cidades

Declaração `public Button_Cidades: TButton;`

ButtonPanel1

Declaração `public ButtonPanel1: TButtonPanel;`

DmxScroller_Form_Lcl1

Declaração `public DmxScroller_Form_Lcl1: TDmxScroller_Form_Lcl;`

GroupBox1

Declaração `public GroupBox1: TGroupBox;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Panel1

Declaração `public Panel1: TPanel;`

StaticText1

Declaração `public StaticText1: TStaticText;`

Métodos

Form_ds_test2Execute

Declaração `public procedure Form_ds_test2Execute(Sender: TObject);`

Action_Form_ds_testExecute

Declaração `public procedure Action_Form_ds_testExecute(Sender: TObject);`

AddTemplateClick

Declaração `public procedure AddTemplateClick(Sender: TObject);`

GetTemplateClick

Declaração `public procedure GetTemplateClick(Sender: TObject);`

Button_ModifyFontsAll_LCLClick

Declaração `public procedure Button_ModifyFontsAll_LCLClick(Sender: TObject);`

form_ds_TestClick

Declaração `public procedure form_ds_TestClick(Sender: TObject);`

InputBoxClick

Declaração `public procedure InputBoxClick(Sender: TObject);`

Descrição O método `InputBoxClick` demonstra o uso da função MsgBox_Form

DmxScroller_Form_Lcl1Enter

Declaração `public procedure DmxScroller_Form_Lcl1Enter(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form_Lcl1EnterField

Declaração `public procedure DmxScroller_Form_Lcl1EnterField(aField: pDmxFieldRec);`

DmxScroller_Form_Lcl1Exit

Declaração `public procedure DmxScroller_Form_Lcl1Exit(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form_Lcl1ExitField

Declaração `public procedure DmxScroller_Form_Lcl1ExitField(aField: pDmxFieldRec);`

DmxScroller_Form_Lcl1GetTemplate

Declaração `public function DmxScroller_Form_Lcl1GetTemplate(aNext: PSItem): PSItem;`

DmxScroller_Form_Lcl1NewRecord

Declaração `public procedure DmxScroller_Form_Lcl1NewRecord(aDmxScroller: TUiDmxScroller);`

ExcluirExecute

Declaração `public procedure ExcluirExecute(Sender: TObject);`

FormClose

Declaração `public procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);`

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

GravarExecute

Declaração `public procedure GravarExecute(Sender: TObject);`

mi_scrollbox_LCL1Enter

Declaração `public procedure mi_scrollbox_LCL1Enter(Sender: TObject);`

NovoExecute

Declaração `public procedure NovoExecute(Sender: TObject);`

PesquisarExecute

Declaração `public procedure PesquisarExecute(Sender: TObject);`

PesquisaExecute

Declaração `public procedure PesquisaExecute(Sender: TObject);`

StaticText1Click

Declaração `public procedure StaticText1Click(Sender: TObject);`

StaticText2Click

Declaração `public procedure StaticText2Click(Sender: TObject);`

destroy

Declaração `public destructor destroy; override;`

57.5 Constantes

tmp_Alunos_Idade

Declaração
```
tmp_Alunos_Idade = '\BB'+ChFN+'idade'+CharUpperlimit+#64+
 CharHint+'A idade do aluno.  Valores válidos 1 a 64'+ CharHintPorque+'Este
 campo é necessário para que se agrupe o alunos baseado em sua faixa etária'+
 CharHintOnde+'Ele será usado pelo coordenador ao classificar a turma';
```

tmp_Alunos_Matricula

Declaração `tmp_Alunos_Matricula = '\IIII'+ChFN+'matricula'+CharHint+'A matricula do aluno é um campo sequencial e calculado ao incluir o registro';`

tmp_Alunos

Declaração `tmp_Alunos = '  Idade:   %s'+TDmxScroller_Form_Lcl.lf+ '  Matricula:   %s'+TDmxScroller_Form_Lcl.lf;`

57.6 Variáveis

DmxScroller_Form_Lcl_test

Declaração `DmxScroller_Form_Lcl_test:  TDmxScroller_Form_Lcl_test;`

Chapter 58

Unit umi_ui_bitbtn_lcl

58.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- Buttons
- ActnList
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

58.2 Visão Geral

TMi_BitBtn_LCL Classe

Register

58.3 Classes, Interfaces, Objetos e Registros

TMi_BitBtn_LCL Classe

Hierarquia

TMi_BitBtn_LCL > TBitBtn

Descrição

A classe `TMi_BitBtn_LCL` é necessária para que se possa selecionar o controle associado ao botão criado pelo método: `pDmxFieldRec(??)^.createExecAction`.

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_BitBtn_LCL(??)`.

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString(??)`)

Métodos

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

58.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 59

Unit umi_ui_button_lcl

59.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- ActnList
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

59.2 Visão Geral

TMI_Button_LCL Classe

Register

59.3 Classes, Interfaces, Objetos e Registros

TMI_Button_LCL Classe

Hierarquia

TMI_Button_LCL > TButton

Descrição

A classe `TMI_Button_LCL` é necessária para que se possa selecionar o controle associado ao botão criado pelo método: `pDmxFieldRec(??)^.createExecAction`.

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_Button_LCL(??)`.

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString(??)`)

Métodos

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

59.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 60

Unit umi_ui_checkbox_lcl

60.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- ActnList
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

60.2 Visão Geral

TMI_CheckBox_LCL Classe

Register

60.3 Classes, Interfaces, Objetos e Registros

TMI_CheckBox_LCL Classe

Hierarquia

TMI_CheckBox_LCL > TCheckBox

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_Button_LCL`(**??**).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(**??**))

Métodos

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec`(**??**)

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec`(**??**) para o controle (Self).

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e pDmxFieldRec.DoOnExit(Self).

60.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 61

Unit uMi_ui_ComboBox_lcl

61.1 Uses

- Windows
- Messages
- SysUtils
- Classes
- Graphics
- Controls
- Forms
- Dialogs
- StdCtrls
- LResources
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

61.2 Visão Geral

TMI_ComboBox_LCL Classe

Register

61.3 Classes, Interfaces, Objetos e Registros

TMI_ComboBox_LCL Classe

Hierarquia

TMI_ComboBox_LCL > TComboBox

Descrição

A classe `TMI_ComboBox_LCL` permite edita um campo enumerado do componente `TDmxFieldRec`(??)

- **NOTA**
  - O item zero contm a string selecionada e caso a mesma seja editada o valor digitado passa ser o filtro de pesquisa.

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes: TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessrios para criar o componente `TMI_MaskEdit_LCL`(??).

- **NOTA**
  - Esses dados devem ser criados pelo mtodo TDmxScroller_Form_Lcl_attributes.Crea ATemplate : `TString`(??))

Value

Declaração `public property Value: String read GetValue write SetValue;`

ImgIndexes

Declaração `published property ImgIndexes: TStringList read FImgIndexes write SetImgIndexes;`

Images

Declaração `published property Images: TImageList read FImages write SetImages;`

ShowImages

Declaração `published property ShowImages: Boolean read FShowImages write SetShowImages;`

Color

Declaração `published property Color;`

Align

Declaração `published property Align;`

AutoComplete

Declaração `published property AutoComplete;`

AutoDropDown

Declaração `published property AutoDropDown;`

AutoSelect

Declaração `published property AutoSelect;`

OnEditingDone

Declaração `published property OnEditingDone;`

AutoSize

Declaração `published property AutoSize;`

text

Declaração `published property text;`

ItemIndex

Declaração `published property ItemIndex;`

DragMode

Declaração `published property DragMode;`

DragCursor

Declaração `published property DragCursor;`

DropDownCount

Declaração `published property DropDownCount;`

Enabled

Declaração `published property Enabled;`

Font

Declaração `published property Font;`

ItemHeight

Declaração `published property ItemHeight;`

Items

Declaração `published property Items;`

MaxLength

Declaração `published property MaxLength;`

ParentColor

Declaração `published property ParentColor;`

ParentFont

Declaração `published property ParentFont;`

ParentShowHint

Declaração `published property ParentShowHint;`

PopupMenu

Declaração `published property PopupMenu;`

ShowHint

Declaração `published property ShowHint;`

Sorted

Declaração `published property Sorted;`

TabOrder

Declaração `published property TabOrder;`

TabStop

Declaração `published property TabStop;`

Visible

Declaração `published property Visible;`

OnChange

Declaração `published property OnChange;`

OnClick

Declaração `published property OnClick;`

OnDblClick

Declaração `published property OnDblClick;`

OnDragDrop

Declaração `published property OnDragDrop;`

OnDragOver

Declaração `published property OnDragOver;`

OnDrawItem

Declaração `published property OnDrawItem;`

OnDropDown

Declaração `published property OnDropDown;`

OnEndDrag

Declaração `published property OnEndDrag;`

OnEnter

Declaração `published property OnEnter;`

OnExit

Declaração `published property OnExit;`

OnKeyDown

Declaração `published property OnKeyDown;`

OnKeyPress

Declaração `published property OnKeyPress;`

OnKeyUp

Declaração `published property OnKeyUp;`

OnMeasureItem

Declaração `published property OnMeasureItem;`

OnStartDrag

Declaração `published property OnStartDrag;`

Anchors

Declaração `published property Anchors;`

Métodos

DrawItem

Declaração
```
protected procedure DrawItem(Index:  Integer; Rect:  TRect; State:
 TOwnerDrawState); override;
```

Create

Declaração `public constructor Create(AOwner:TComponent); override; overload;`

Create

Declaração `public constructor`
`Create(aOwner:TComponent;aDmxScroller_Form_Lcl_attributes:`
`TDmxScroller_Form_Lcl_attributes); overload; overload;`

Destroy

Declaração `public destructor Destroy; override;`

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` transfere os dados do controle para o componente TMI_rtl_ui_DmxScroller.

- **NOTA**
  - Quando **pDmxFieldRec.ListComboBox** <> nil usar `Value(??)` e se **pDmxFieldRec.ListComboBox**=nil usar `ItemIndex`(??).
  - A propriedade `Value(??)` pode ser qualquer valor.

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` transfere os dados do controle para o componente TMI_rtl_ui_DmxScroller.

- **NOTA**
  - Quando **pDmxFieldRec.ListComboBox** <> nil usar `Value(??)` e se **pDmxFieldRec.ListComboBox**=nil usar `ItemIndex`(??).

DoOnMouseDown

Declaração `protected procedure DoOnMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);`

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Select

Declaração `protected procedure Select; override;`

DoOnKeyPress

Declaração `protected procedure DoOnKeyPress(Sender: TObject; var Key: system.Char);`

Clear

Declaração `public procedure Clear; Override;`

AddValue

Declaração `public procedure AddValue(aString:String);`

WMPaint

Declaração `protected procedure WMPaint(var Message:  TLMPaint); message LM_PAINT;`

61.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 62

Unit uMi_Ui_DBCheckBox_Lcl

62.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- DBCtrls
- ActnList
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

62.2 Visão Geral

TMi_Ui_DBCheckBox_Lcl Classe

Register

62.3 Classes, Interfaces, Objetos e Registros

TMi_Ui_DBCheckBox_Lcl Classe

Hierarquia

TMi_Ui_DBCheckBox_Lcl > TDBCheckBox

Descrição

A classe `TMi_Ui_DBCheckBox_Lcl` permite edita um campo boolean do registro `TDmxFieldRec(??)`

- **NOTA** -

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

DmxFieldRec

Declaração `public property DmxFieldRec:  pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_MaskEdit_LCL(??)`.

- **NOTA**
  - Esses dados devem ser criados pelo método TDmxScroller_Form_Lcl_attributes.Cre ATemplate : `TString(??)`)

Métodos

Create

Declaração `public constructor Create(AOwner:TComponent); override;`

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e pDmxFieldRec.DoOnExit(Self).

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec`(**??**)

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec`(**??**) para o controle (Self).

62.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 63

Unit uMi_Ui_DbComboBox_lcl

63.1 Uses

- Windows
- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- DBCtrls
- StdCtrls
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

63.2 Visão Geral

TMi_DbComboBox_LCL Classe

Register

63.3 Classes, Interfaces, Objetos e Registros

TMi_DbComboBox_LCL Classe

Hierarquia

TMi_DbComboBox_LCL > TDBComboBox

Descrição

A classe `TMi_DbComboBox_LCL` permite edita um campo enumerado do registro `TDmxFieldRec(??)`

- **NOTA**
  - O item zero contém a string selecionada e caso a mesma seja editada o valor digitado passa ser o filtro de pesquisa.

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_MaskEdit_LCL(??)`.

- **NOTA**
  - Esses dados devem ser criados pelo método TDmxScroller_Form_Lcl_attributes.Cre ATemplate : `TString(??)`)

Value

Declaração `public property Value: String read GetValue write SetValue;`

ImgIndexes

Declaração `published property ImgIndexes: TStringList read FImgIndexes write SetImgIndexes;`

Images

Declaração `published property Images: TImageList read FImages write SetImages;`

ShowImages

Declaração `published property ShowImages: Boolean read FShowImages write SetShowImages;`

Color

Declaração `published property Color;`

Align

Declaração `published property Align;`

AutoComplete

Declaração `published property AutoComplete;`

AutoDropDown

Declaração `published property AutoDropDown;`

AutoSelect

Declaração `published property AutoSelect;`

OnEditingDone

Declaração `published property OnEditingDone;`

AutoSize

Declaração `published property AutoSize;`

text

Declaração `published property text;`

ItemIndex

Declaração `published property ItemIndex;`

DragMode

Declaração `published property DragMode;`

DragCursor

Declaração `published property DragCursor;`

DropDownCount

Declaração `published property DropDownCount;`

Enabled

Declaração `published property Enabled;`

Font

Declaração `published property Font;`

ItemHeight

Declaração `published property ItemHeight;`

Items

Declaração `published property Items;`

MaxLength

Declaração `published property MaxLength;`

ParentColor

Declaração `published property ParentColor;`

ParentFont

Declaração `published property ParentFont;`

ParentShowHint

Declaração `published property ParentShowHint;`

PopupMenu

Declaração `published property PopupMenu;`

ShowHint

Declaração `published property ShowHint;`

Sorted

Declaração `published property Sorted;`

TabOrder

Declaração `published property TabOrder;`

TabStop

Declaração `published property TabStop;`

Visible

Declaração `published property Visible;`

OnChange

Declaração `published property OnChange;`

OnClick

Declaração `published property OnClick;`

OnDblClick

Declaração `published property OnDblClick;`

OnDragDrop

Declaração `published property OnDragDrop;`

OnDragOver

Declaração `published property OnDragOver;`

OnDrawItem

Declaração `published property OnDrawItem;`

OnDropDown

Declaração `published property OnDropDown;`

OnEndDrag

Declaração `published property OnEndDrag;`

OnEnter

Declaração `published property OnEnter;`

OnExit

Declaração `published property OnExit;`

OnKeyDown

Declaração `published property OnKeyDown;`

OnKeyPress

Declaração `published property OnKeyPress;`

OnKeyUp

Declaração `published property OnKeyUp;`

OnMeasureItem

Declaração `published property OnMeasureItem;`

OnStartDrag

Declaração `published property OnStartDrag;`

Anchors

Declaração `published property Anchors;`

Métodos

DrawItem

Declaração
```
protected procedure DrawItem(Index:  Integer; Rect:  TRect; State:
 TOwnerDrawState); override;
```

Create

Declaração `public constructor Create(AOwner:TComponent); override; overload;`

Create

Declaração
```
public constructor
Create(aOwner:TComponent;aDmxScroller_Form_Lcl_attributes :
TDmxScroller_Form_Lcl_attributes); overload; overload;
```

Destroy

Declaração `public destructor Destroy; override;`

PutBuffer

Declaração `public Procedure PutBuffer;`

GetBuffer

Declaração `public Procedure GetBuffer;`

DoOnMouseDown

Declaração
```
protected procedure DoOnMouseDown(Sender:  TObject; Button:
TMouseButton; Shift:  TShiftState; X, Y: Integer);
```

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender:  TObject);`

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Select

Declaração `protected procedure Select; override;`

DoOnKeyPress

Declaração `protected procedure DoOnKeyPress(Sender: TObject; var Key: system.Char);`

Clear

Declaração `public procedure Clear; Override;`

AddValue

Declaração `public procedure AddValue(aString:String);`

WMPaint

Declaração `protected procedure WMPaint(var Message: TLMPaint); message LM_PAINT;`

63.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 64

Unit uMI_ui_DbEdit_LCL

64.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- DBCtrls
- LMessages
- LCLType
- db
- Variants
- mi.rtl.objects.Methods.dates(??)
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

64.2 Visão Geral

`TMI_DbEdit_LCL Classe`

`Register`

64.3 Classes, Interfaces, Objetos e Registros

TMI_DbEdit_LCL Classe

Hierarquia

TMI_DbEdit_LCL > TDBEdit

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `public property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário a ser criado em owner

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_DbEdit_LCL`(**??**).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(**??**))

DisplayFormat

Declaração `public property DisplayFormat : AnsiString Read _DisplayFormat;`

Descrição A propriedade `DisplayFormat` a mascara de saída usada para setar Field.DisplayFormat

MaskEdit

Declaração `public property MaskEdit : AnsiString Read _MaskEdit write SetMaskEdit;`

Descrição A propriedade `MaskEdit` contém o modelo e os cálculos do formulário a ser criado em owner

Métodos

Create

Declaração `public constructor Create(AOwner:TComponent); overload; override;`

Create

Declaração `public constructor Create(aOwner:TComponent;aDmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes); overload;`

SeTDmxFieldRec

Declaração `protected Procedure SeTDmxFieldRec(apDmxFieldRec : pDmxFieldRec ); Overload;`

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec`(**??**)

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec`(**??**) para o controle (Self).

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

Descrição O método `DoOnEnter` ao receber o foco executa os métodos `GetBuffer(??)` e pDmxFieldRec.DoOnEnter(Self).

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e pDmxFieldRec.DoOnExit(Self).

DoEditNumberKeyPress

Declaração `protected procedure DoEditNumberKeyPress(Sender: TObject; var Key: char);`

Descrição O método `DoEditNumberKeyPress` edita os campos números de 1 a 10 bytes

GetHelpCtx_Hint

Declaração `protected FUNCTION GetHelpCtx_Hint(): AnsiString;`

Descrição O método `GetHelpCtx_Hint` captura a Idocumentação do campo definido na classe onde o campo for criado.

- Com o programa **pasdoc** a Idocumentação não precisa está no arquivo de recursos, por isso, para obter o link para o campo preciso saber apenas o endereço do link.

GetMaxLength

Declaração `protected FUNCTION GetMaxLength(): Variant;`

SetMaxLength

Declaração `protected PROCEDURE SetMaxLength(aMaxLength: Variant);`

GetSize

Declaração `protected FUNCTION GetSize(): Variant;`

SetSize

Declaração `protected PROCEDURE SetSize(aSize: Variant);`

SetAlias

Declaração `protected procedure SetAlias(const aAlias: AnsiString);`

GetName

Declaração `protected FUNCTION GetName(): AnsiString;`

GetAlias

Declaração `protected FUNCTION GetAlias: AnsiString;`

WMSetFocus

Declaração `protected procedure WMSetFocus(var Message: TLMSetFocus); message LM_SETFOCUS;`

GetDataLink

Declaração `protected function GetDataLink:TFieldDataLink;`

WMPaint

Declaração `protected procedure WMPaint(var Message:  TLMPaint); message LM_PAINT;`

ValidateEdit

Declaração `public procedure ValidateEdit; override;`

64.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 65

Unit umi_ui_dblookupComboBox_lcl

65.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- DBCtrls
- StdCtrls
- db
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

65.2 Visão Geral

TMi_ui_DBLookupComboBox_Lcl Classe

Register

65.3 Classes, Interfaces, Objetos e Registros

TMi_ui_DBLookupComboBox_Lcl Classe

Hierarquia

TMi_ui_DBLookupComboBox_Lcl > TDBLookupComboBox

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_MaskEdit_LCL`(??).

- **NOTA**
  - Esses dados devem ser criados pelo método TDmxScroller_Form_Lcl_attributes.Cre ATemplate : `TString`(??))

Métodos

Create

Declaração `public constructor Create(AOwner:TComponent); override; overload;`

Create

Declaração `public constructor Create(aOwner:TComponent;aDmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes); overload; overload;`

Destroy

Declaração `public destructor Destroy; override;`

PutBuffer

Declaração `public Procedure PutBuffer;`

GetBuffer

Declaração `public Procedure GetBuffer;`

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender:  TObject);`

DoOnExit

Declaração `protected procedure DoOnExit(Sender:  TObject);`

65.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 66

Unit uMI_ui_DbRadioGroup_Lcl

66.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- DBCtrls
- ActnList
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

66.2 Visão Geral

TMI_ui_DbRadioGroup_Lcl Classe

Register

66.3 Classes, Interfaces, Objetos e Registros

TMI_ui_DbRadioGroup_Lcl Classe

Hierarquia

TMI_ui_DbRadioGroup_Lcl > TDBRadioGroup

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_Button_LCL`(**??**).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(**??**))

Métodos

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec`(**??**) para o controle (Self).

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec(??)`

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e `pDmxFieldRec(??)^`.DoOnExit(Self).

66.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 67

Unit umi_ui_dmxscroller_form_lcl_attributes

67.1 Descrição

A unit `umi_ui_dmxscroller_form_lcl_attributes` implementa a classe `TDmxScroller_Form`(**??**).

- Primeiro autor: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **HISTÓRICO**:
  - T12 Criado unit com objetivo de separar o pacote `mi.rtl`(**??**) do pacote **LCL** com objetivo de criar aplicação web independente de gráficos locais.
- **PENDÊNCIAS**
  - T12 Documentar a unit.
  - T12 Criado unit com objetivo de separar o pacote `mi.rtl`(**??**) do pacote **LCL** com objetivo de criar aplicação web independente de gráficos locais.
  - T12 Criar classe `TDmxScroller_Form_Lcl_attributes`(**??**) e mover todos os atributos da classe TDmxScroller_Form_Lcl para ela.

67.2 Uses

- Classes
- SysUtils
- controls
- StdCtrls
- forms
- typInfo
- types
- Graphics
- ActnList
- Dialogs
- mi.rtl.Consts(??)
- mi_rtl_ui_DmxScroller_Form(??)
- uMi_ui_scrollbox_lcl(??)

67.3 Visão Geral

TDmxScroller_Form_Lcl_attributes Classe

67.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_attributes Classe

Hierarquia

TDmxScroller_Form_Lcl_attributes > TDmxScroller_Form(??) > TDmxScroller_Form_Atributos(??) > TUiDmxScroller(??) > TUiMethods(??) > TUiConsts

Descrição

no description available, TDmxScroller_Form description followsA classe TDmxScroller_Form implementa a construção de formulários usando uma lista de Templates do tipo TDmxScroller

Propriedades

ActionList

Declaração `public property ActionList : TActionList Read _ActionList Write _ActionList;`

Descrição A propriedade `ActionList` permite que ações do formulário LCL possam ser executados a partir do componente **TDmxScroller**.

- **NOTA**
  - O interpretador de Template inicia o campo `TDmxFieldRec.ExecAction(??)` e o campo LinkExecAction.
  - O gerador de formulário ao encontrar `TDmxFieldRec.ExecAction(??)` cria um botão para que se possa executar a ação.
- **EXEMPLO**

```
with DmxScroller_Form1 do
begin
  Result := NewSItem('        &Primeiro '+CharExecAction+Novo.Name+
                     '  P&róximo '+CharExecAction+(Gravar.Name)+
                     ' &Anterior '+CharExecAction+Pesquisar.Name+
                     '  &Ultimo '+CharExecAction+(Excluir.Name),nil);
end;
```

ParentLCL

Declaração `published property ParentLCL : TScrollingWinControl Read _ParentLCL write SetParentLCL;`

Descrição A propriedade `ParentLCL` informa a janela que será desenhada o formulário

Campos

_ActionList

Declaração `protected _ActionList: TActionList;`

Chapter 68

Unit umi_ui_dmxscroller_form_lcl_base

68.1 Uses

- Classes
- SysUtils

Chapter 69

Unit umi_ui_dmxscroller_form_lcl_ds

69.1 Descrição

A unit `umi_ui_dmxscroller_form_lcl_ds` implementa a classe TDmxScroller_Form_Lcl_attributes_Form_ds.

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
  - **PENDÊNCIAS**
    - T12 O Componente TMi_ui_DmxScroller_Form_ds quando inserido em um dataModule os campos checkBox e RadioButton não estão visíveis.
    - T12 Corrigir problema quando vinculados ao dbGrid os controles do tipo:
      - `fldEnum`(**??**);
      - `FldBoolean`(**??**);
      - `FldRadioButton`(**??**);
    - T12 Criar Método procedure TDmxScroller_Form.CreateFormLCL(aOwner: TScrolling-WinControl);
      - Esse método deve usar as classes da Pallet Data Controls
      - TMI_ui_DbText_LCL
      - TMI_ui_DbEdit_LCL

        - `TMI_ui_DbLookupComboBox_LCL`(**??**)
        - TMI_ui_DbCheck_LCL
        - TMI_ui_DbRadioButton_Lcl
        - TMi_ui_DbData_lcl
        - TMi_ui_DbHora_lcl
        - TMi_ui_DbDataHora_lcl
    * T12 HABILITAR OS EVENTOS DE TDataSource.dataset
        - T12 Executar os eventos do dataSet associado aos controles dbText e Db dbComboBox
        - T12 Dar opção global para habilitar e desabilitar as mascaras dos textos.
        - Criar opção global OkEditMask para usar com a propriedade: TMI_DbEdit_LCL.CustomEditMask;

= **CONCLUÍDO**

- T12 A classe deve herdar de **TDmxScroller_Form_DS**
- **HISTÓRICO**
    * Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
    * **2022-04-27**
        - **14:10**
        - Análise de como implementar a unit;
        - Criar a classe TMI_ui_DbEdit_LCL
    * **2022-06-07**
        - Criar a classe TMI_ui_DbCheck_LCL
        - Em CreateFormLCL inserir classe TMI_ui_DbCheck_LCL se CurrentField for do tipo `FldBoolean`(**??**);
        - Criar a classe `TMi_Ui_DBRadioGroup_Lcl`(**??**)
        - Em CreateFormLCL inserir classe `TMI_ui_DbRadioGroup_LCL`(**??**) se CurrentField for do tipo `FldRadioButton`(**??**);
        - Criar função CreateRadioGroup;

- Em CreateFormLCL inserir classe `TMi_Button_LCL`(??) se DmxFieldRec.ExecAction<>”

* **2022-07-04**

  - **09:38**

  - T12 O Componente TMi_ui_DmxScroller_Form_ds quando inserido em um dataModule os campos checkBox e RadioButton não estão visíveis.

69.2 Uses

- `Classes`
- `SysUtils`
- `controls`
- `StdCtrls`
- `forms`
- `db`
- `mi_rtl_ui_Dmxscroller`(??)
- `uMi_ui_Dmxscroller_form_Lcl`
- `uMI_ui_DbEdit_LCL`(??)
- `uMi_Ui_DbLookupComboBox_lcl`(??)
- `uMi_Ui_DbComboBox_lcl`(??)
- `uMi_Ui_DBCheckBox_Lcl`(??)
- `uMi_Ui_DbRadioGroup_Lcl`(??)
- `umi_ui_button_lcl`(??)
- `uMi_ui_Label_lcl`(??)

69.3 Visão Geral

`TDmxScroller_Form_Lcl_DS Classe`

`Register`

69.4 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_DS Classe

Hierarquia

TDmxScroller_Form_Lcl_DS > TDmxScroller$_{F}orm_{L}cl$

Propriedades

TableName

Declaração `published property TableName;`

DataSource

Declaração `published property DataSource;`

Métodos

CreateFormLCL

Declaração `protected procedure CreateFormLCL(aOwner:TScrollingWinControl); override;`

Descrição O método `CreateFormLCL` desenha um formulário TScrollingWinControl usando várias linha.

- O modelo cria um registro usando os tipos de dados primitivos.
- **EXEMPLO**:

```
function TDMAlunos.DmxScroller_Form_AlunoGetTemplate(aNext: PSItem): PSItem;
begin
  with DmxScroller_Form1_DS do
  begin
    // AlignmentLabels:= taCenter;
    AlignmentLabels := taLeftJustify;
    // AlignmentLabels := taRightJustify ;
    Result :=
      NewSItem('      Matrícula  \LLLLL'+CharFieldName+'matricula'+CharAccRead(
      NewSItem(' Nome do aluno:  \sssssssssssssssssss‘sssssss'+CharFieldName+'
```

```
        NewSItem('',
        NewSItem('      Endereço:  \sssssssssssssssssss'ssssssssss'+CharFieldNa
        NewSItem(' P. Referência:  \sssssssssssssssssss'sssss'+CharFieldName+'Po
        NewSItem('           Cep:  \##.###-###'+CharFieldName+'cep',
        NewSItem('        Estado:  \SS'+CharFieldName+'Estado'+CharForeignKeyN_Um
        NewSItem('        Cidade:  \sssssssssssssssssss'sssss'+CharFieldName+'ci

        NewSItem('',
        aNext)))))))));

    end;
end;
```

UpdateBuffers_Controls

Declaração `protected procedure UpdateBuffers_Controls; override;`

Descrição O método `UpdateBuffers_Controls` ler o buffer dos campos dos arquivos associados a classe **TDmxScroller_Form_Lcl_attributes_sql** para o buffer dos campos da classe `TDmxScroller_Form_Lcl_attri`

SetActiveTarget

Declaração `protected procedure SetActiveTarget(aActive:  Boolean); override;`

69.5 Funções e Procedimentos

Register

Declaração `procedure Register;`

69.6 Tipos

TDmxFieldRec

Declaração `TDmxFieldRec = uMi_ui_Dmxscroller_form_Lcl.TDmxFieldRec;`

pDmxFieldRec

Declaração `pDmxFieldRec = uMi_ui_Dmxscroller_form_Lcl.pDmxFieldRec;`

SmallWord

Declaração `SmallWord = uMi_ui_Dmxscroller_form_Lcl.SmallWord;`

69.7 Constantes

AccNormal

Declaração `AccNormal = mi_rtl_ui_Dmxscroller.AccNormal;`

Chapter 70

Unit uMi_ui_DmxScroller_Form_Lcl_ds_test

70.1 Uses

- mi.rtl.Objects.Consts(??)
- Classes
- SysUtils
- DB
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- DBGrids
- DBCtrls
- uMi_ui_scrollbox_lcl(??)
- uMi_ui_DmxScroller_Form_Lcl_ds(??)
- uMi_ui_Dmxscroller_form_lcl
- mi_rtl_ui_Dmxscroller(??)
- uMi_Ui_DBCheckBox_Lcl(??)

70.2 Visão Geral

`TMi_ui_DmxScroller_Form_Lcl_ds_test Classe`

70.3 Classes, Interfaces, Objetos e Registros

TMi_ui_DmxScroller_Form_Lcl_ds_test Classe

Hierarquia

TMi_ui_DmxScroller_Form_Lcl_ds_test > TForm

Descrição

A class `TMi_ui_DmxScroller_Form_Lcl_ds_test` demonstra a classe `TDmxScroller_Form_Lcl_DS`(??) integrada aos controles associados a `DataSource1`(??).

- **NOTA**
  - A class `TMi_ui_DmxScroller_Form_Lcl_ds_test` cria o dataset associado a `DataSource1`(??) caso DataSource1.DataSet seja nil.
  - Caso DataSource1.dataSet <> nil, então o mesmo precisa ter os mesmo campos passado pelo template.
    - Obs: Se o campos passados em DataSource1.DataSet não sejão iguais aos templates, o sistema vai haver execeção.

Campos

DataSource1

Declaração `public DataSource1:  TDataSource;`

DBGrid1

Declaração `public DBGrid1:  TDBGrid;`

DmxScroller_Form_Lcl_DS1

Declaração `public DmxScroller_Form_Lcl_DS1:  TDmxScroller_Form_Lcl_DS;`

GroupBox1

Declaração `public GroupBox1: TGroupBox;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

DmxScroller_Form_Lcl_DS1GetTemplate

Declaração `public function DmxScroller_Form_Lcl_DS1GetTemplate(aNext: PSItem): PSItem;`

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

70.4 Variáveis

Mi_ui_DmxScroller_Form_Lcl_ds_test

Declaração `Mi_ui_DmxScroller_Form_Lcl_ds_test: TMi_ui_DmxScroller_Form_Lcl_ds_test;`

Chapter 71

Unit uMi_ui_DmxScroller_Form_Lcl_ds_test2

71.1 Uses

- Classes
- SysUtils
- Forms
- Controls
- Graphics
- Dialogs
- DBGrids
- uMi_ui_DmxScroller_Form_Lcl_ds_test2_dm(??)
- uMi_ui_scrollbox_lcl(??)

71.2 Visão Geral

TMi_ui_DmxScroller_Form_Lcl_ds_test2 Classe

71.3 Classes, Interfaces, Objetos e Registros

TMi_ui_DmxScroller_Form_Lcl_ds_test2 Classe

Hierarquia

TMi_ui_DmxScroller_Form_Lcl_ds_test2 > TForm

Campos

DBGrid1

Declaração `public DBGrid1: TDBGrid;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

71.4 Variáveis

Mi_ui_DmxScroller_Form_Lcl_ds_test2

Declaração
```
Mi_ui_DmxScroller_Form_Lcl_ds_test2:
 TMi_ui_DmxScroller_Form_Lcl_ds_test2;
```

Chapter 72

Unit uMi_ui_DmxScroller_Form_Lcl_ds_test2_dm

72.1 Uses

- Classes
- SysUtils
- ActnList
- DB
- uMi_ui_DmxScroller_Form_Lcl_ds(??)
- mi_rtl_ui_Dmxscroller(??)

72.2 Visão Geral

TMi_ui_DmxScroller_Form_Lcl_ds_test2_dm Classe

72.3 Classes, Interfaces, Objetos e Registros

TMi_ui_DmxScroller_Form_Lcl_ds_test2_dm Classe

Hierarquia

TMi_ui_DmxScroller_Form_Lcl_ds_test2_dm > TDataModule

Campos

GoBof

Declaração `public GoBof: TAction;`

Next

Declaração `public Next: TAction;`

Prev

Declaração `public Prev: TAction;`

GoEof

Declaração `public GoEof: TAction;`

ActionList1

Declaração `public ActionList1: TActionList;`

DataSource1

Declaração `public DataSource1: TDataSource;`

DmxScroller_Form_Lcl_DS1

Declaração `public DmxScroller_Form_Lcl_DS1: TDmxScroller_Form_Lcl_DS;`

Excluir

Declaração `public Excluir: TAction;`

Gravar

Declaração `public Gravar: TAction;`

Novo

Declaração `public Novo: TAction;`

Pesquisar

Declaração `public Pesquisar: TAction;`

Métodos

DataModuleCreate

Declaração `public procedure DataModuleCreate(Sender: TObject);`

GoBofExecute

Declaração `public procedure GoBofExecute(Sender: TObject);`

DmxScroller_Form_Lcl_DS1AddTemplate

Declaração
```
public procedure DmxScroller_Form_Lcl_DS1AddTemplate( const
 aUiDmxScroller: TUiDmxScroller);
```

ExcluirExecute

Declaração `public procedure ExcluirExecute(Sender: TObject);`

GoEofExecute

Declaração `public procedure GoEofExecute(Sender: TObject);`

GravarExecute

Declaração `public procedure GravarExecute(Sender: TObject);`

NextExecute

Declaração `public procedure NextExecute(Sender: TObject);`

NovoExecute

Declaração `public procedure NovoExecute(Sender: TObject);`

PesquisarExecute

Declaração `public procedure PesquisarExecute(Sender: TObject);`

PrevExecute

Declaração `public procedure PrevExecute(Sender: TObject);`

72.4 Variáveis

Mi_ui_DmxScroller_Form_Lcl_ds_test2_dm

Declaração
```
Mi_ui_DmxScroller_Form_Lcl_ds_test2_dm:
 TMi_ui_DmxScroller_Form_Lcl_ds_test2_dm;
```

Chapter 73

Unit uMi_ui_DmxScroller_Form_Lcl_ds_test_dm

73.1 Uses

- Classes
- SysUtils
- DB
- uMi_ui_DmxScroller_Form_Lcl_ds(??)
- mi_rtl_ui_Dmxscroller(??)

73.2 Visão Geral

TMi_ui_DmxScroller_Form_Lcl_ds_test_dm Classe

73.3 Classes, Interfaces, Objetos e Registros

TMi_ui_DmxScroller_Form_Lcl_ds_test_dm Classe

Hierarquia

TMi_ui_DmxScroller_Form_Lcl_ds_test_dm > TDataModule

Campos

DataSource1

Declaração `public DataSource1: TDataSource;`

DmxScroller_Form_Lcl_DS1

Declaração `public DmxScroller_Form_Lcl_DS1: TDmxScroller_Form_Lcl_DS;`

Métodos

DmxScroller_Form_Lcl_DS1GetTemplate

Declaração `public function DmxScroller_Form_Lcl_DS1GetTemplate(aNext: PSItem): PSItem;`

73.4 Variáveis

Mi_ui_DmxScroller_Form_Lcl_ds_test_dm

Declaração `Mi_ui_DmxScroller_Form_Lcl_ds_test_dm: TMi_ui_DmxScroller_Form_Lcl_ds_test_dm;`

Chapter 74

Unit uMI_UI_InputBox

74.1 Descrição

A unit `uMI_UI_InputBox` implementa o formulário `TMI_UI_InputBox`(**??**) usado para criar formulário baseado em Template `PSITem`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.693
- **CÓDIGO FONTE**:
  -
- **PENDÊNCIAS**
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
  - **2022-05-17**
    - T12 Análise de como será a classe `TMI_UI_InputBox`(**??**).
    - T12 Criar a unit `uMI_UI_InputBox`.
    - T12 Criar formulário `TMI_UI_InputBox`(**??**);
    - T12 Adicionar o componente ButtonPanel1 e habilitar os botões ok e cancel;
    - T12 Adicionar o componente Mi_ScrollBox_LCL1 ;
    - T12 Criar evento: function DmxScroller_Form_Lcl1GetTemplate;
    - T12 Criar atributo protected _FormSItem : `PSitem`(**??**);

        * T12 Criar propriedade Template:AnsiString;
            · Criar método Set_Template(aTemplate:AnsiString);

- **2022-05-18**
    * **10:51**
        · As alterações que fiz ontem no método `TObjectsMethods.StringToSItem`(**??**)() criou efeito colateral.
        · Corrigido.
    * **14:28**
        · Criar função:
        · function `InputBox`(**??**)(): TModalResult;

- **2022-05-19**
    * **11:13**
        · Criar os eventos
        · OnEnterLocal
        · OnExistLocal
        · onEnterFieldLocal
        · OnExitDieldLocal
        · Criar função:
        · function MI_MsgBox1MessageBox_04_PSItem(aPSItem: TMI_MsgBoxTypes.PSItem; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn): TModalResult;

74.2 Uses

- `Classes`
- `SysUtils`
- `Forms`
- `Controls`
- `Graphics`

- Dialogs
- StdCtrls
- ExtCtrls
- ButtonPanel
- uMi_ui_DmxScroller_Form_Lcl
- mi_rtl_ui_Dmxscroller(??)
- uMi_ui_scrollbox_lcl(??)
- uMi_ui_DmxScroller_Form_Lcl_ds(??)

74.3 Visão Geral

TMI_UI_InputBox Classe

InputBox

74.4 Classes, Interfaces, Objetos e Registros

TMI_UI_InputBox Classe

Hierarquia

TMI_UI_InputBox > TForm

Descrição

A class TMI_UI_InputBox edita uma formulário passado em forma de Template(??) e devolve o formulário LCL caso o botão **OK** seja pressionando ou nil caso o botão **Cancelar** seja pressionando.

- **EXEMPLO**
  - O exemplo abaixo cria um formulário e permite interagir com com os eventos ao entrar e ao sair do formulário.

```
procedure InputBoxOnEnter(aDmxScroller: TUiDmxScroller);
 //Ao entrar no formulário este evento inicia os campos nome, endereço e cep

  procedure SetValue(Field:String;Value:String);
    var
      aField: pDmxFieldRec;
  begin
```

```
    with aDmxScroller do
    begin
      aField := FieldByName(Field);
      if aField<>nil
      Then aField.AsString:= value;
    end;
  end;

begin
  with aDmxScroller do
  begin
    setValue('nome','Paulo Henrique');
    setValue('endereço','Rua Francisco de Souza Oliveira, 15');
    setValue('cep','60310770');
  end;
end;

procedure InputBoxOnEnterField(aField: pDmxFieldRec);
    // Ao entrar no campo 01 e o mesmo for vazio inicializa-o com o nome Paulo Sérgio

begin
  Case aField.Fieldnum of
    1 : begin
          if aField.AsString = ''
          then aField.AsString := 'Paulo Sérgio';
        end;
    2 : begin end;
  end;

end;


Procedure InputBoxOnCloseQuery (aDmxScroller:TUiDmxScroller; var CanClose:boolean);
  //Esta função permite validar o formulário ao pressionar o botão ok.

   var
    idade : byte;
    s : string;
begin
  s := aDmxScroller.FieldByName('idade').AsString;
  idade := StrToInt(s);
  if  idade <> 64
  then begin
         ShowMessage('O campo idade <> de 64!.');
         CanClose := false;
       end
```

```
  else CanClose := true;
end;


Procedure TestInputBox;
    //Criar o formulário

  Var
    MI_UI_InputBox : TMI_UI_InputBox = nil;
begin
  with TDmxScroller_Form_Lcl do
  if InputBox('Dados do aluno',
                ' Nome do Aluno: \Sssssssssssssssssssssss'ssssssssssssss'+ChFN+'Nome'+lf+
                '      Endereço: \Sssssssssssssssssssssss'ssssssssssssss'+ChFN+'endereco'+lf+
                '           Cep: \##-###-###'+ChFN+'cep'+lf+
                '        Bairro: \sssssssssssssssssssssss'+ChFN+'bairro'+lf+
                '        Cidade: \sssssssssssssssssssssss'+ChFN+'cidade'+lf+
                '        Estado: \SS'+ChFN+'estado'+lf+
                '         Idade: \BB'+ChFN+'idade'+lf+
                '     Matricula: \III'+ChFN+'matricula'+lf+
                '     Mensalidade: \R,RRR.RR'+ChFN+'mensalidade',
                InputBoxOnEnter,nil,

                InputBoxOnEnterField,nil,
                InputBoxOnCloseQuery,
                MI_UI_InputBox
           ) = MrOk
  then with MI_UI_InputBox.DmxScroller_Form_Lcl1 do
       begin
         if FieldByName('nome').AsString = ''
         then begin
                ShowMessage('Campo "nome" não pode ser vazio');
              end;
         MI_UI_InputBox.free;
       end;
end;

procedure TDmxScroller_Form_Lcl_test.Button_InputBoxClick(Sender: TObject);
begin
  TestInputBox;
end;
```

Propriedades

Template

Declaração `public property Template: AnsiString read _Template write Set_Template;`

Descrição A propriedade `Template` é usada para criar uma lista de `PSItem`(**??**) para ser usada como modelo do formulário.

- **NOTAS**
  - `Template` é um string comum, onde cada linha é separada com ^J.
  - `Template` tem uma lista de string com formato Dmx.
    - Formato da propriedade `Template`:

```
Template := ' Nome do Aluno: \Sssssssssssssssssssssss
            '      Endereço: \Sssssssssssssssssssssss
            '           Cep: \##-###-###'+ChFN+'cep'+lf
            '        Bairro: \sssssssssssssssssssssss
            '        Cidade: \sssssssssssssssssssssss
            '        Estado: \SS'+ChFN+'estado'+lf+
            '         Idade: \BB'+ChFN+'idade'+FldUpper
            '     Matricula: \III'+ChFN+'matricula'+lf+
            '      Valor da '+lf+
            '     Mensalidade: \R,RRR.RR'+ChFN+'mensali
```

  - **SINTAXE**
    - (til) : Limitador de rótulos do formulário;
    - **s** (s minúsculo) : caracteres alfanumérico incluindo os maiúsculas, minusculas, números e símbolos;
    - **S** (S maiúsculo) : caracteres alfanumérico incluindo os maiúsculas, números e símbolos;
    - # (# cancela) : Aceita somente números de 0 a 99
    - **-** (literal) : Separador de números
    - **B** (B maiúsculo): Campo do tipo byte
    - **FldUpperLimit** : O caractere seguinte indica o limite superior da variável. No exemplo acima é 18 anos;

* **R** : Indica um caractere de um campo do tipo double;
* **I** : Indica um caractere de um campo do tipo interger. Faixa: -32000 a +32000;

Campos

ButtonPanel1

Declaração `public ButtonPanel1: TButtonPanel;`

DmxScroller_Form_Lcl1

Declaração `public DmxScroller_Form_Lcl1: TDmxScroller_Form_Lcl;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

CancelButtonClick

Declaração `public procedure CancelButtonClick(Sender: TObject);`

DmxScroller_Form_Lcl1AddTemplate

Declaração
```
public procedure DmxScroller_Form_Lcl1AddTemplate(const
 aUiDmxScroller: TUiDmxScroller );
```

DmxScroller_Form_Lcl1CloseQuery

Declaração
```
public procedure DmxScroller_Form_Lcl1CloseQuery(aDmxScroller:
 TUiDmxScroller;var CanClose: boolean);
```

DmxScroller_Form_Lcl1Enter

Declaração `public procedure DmxScroller_Form_Lcl1Enter(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form_Lcl1EnterField

Declaração `public procedure DmxScroller_Form_Lcl1EnterField(aField: pDmxFieldRec);`

DmxScroller_Form_Lcl1Exit

Declaração `public procedure DmxScroller_Form_Lcl1Exit(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form_Lcl1ExitField

Declaração `public procedure DmxScroller_Form_Lcl1ExitField(aField: pDmxFieldRec);`

DmxScroller_Form_Lcl1GetTemplate

Declaração `public function DmxScroller_Form_Lcl1GetTemplate(aNext: PSItem): PSItem;`

DmxScroller_Form_Lcl1NewRecord

Declaração `public procedure DmxScroller_Form_Lcl1NewRecord(aDmxScroller: TUiDmxScroller);`

FormCloseQuery

Declaração `public procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);`

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

OKButtonClick

Declaração `public procedure OKButtonClick(Sender: TObject);`

Set_Template

Declaração `protected Procedure Set_Template(aTemplate:AnsiString);`

Descrição O Método `Set_Template` inicia o atributo _Template e criar a lista _FormSItem : `PSitem`(**??**)

- **NOTAS**
  - Criar em TObjectss.TStringList o método

SetAlias

Declaração `public Procedure SetAlias(aTitle:AnsiString);`

74.5 Funções e Procedimentos

InputBox

Declaração
```
function InputBox( aTitle: AnsiString; aTemplate: AnsiString;
aOnEnterLocal:TOnEnterLocal ; aOnExitLocal:TOnExitLocal;
aOnEnterFieldLocal:TOnEnterFieldLocal; aOnExitFieldLocal:TOnExitFieldLocal;
aOnCloseQueryLocal:TOnCloseQueryLocal; out aMi_ui_InputBox : TMI_UI_InputBox
): TModalResult;
```

Descrição A função `InputBox` cria um formulário passado por Template e executa os eventos do formulário passado pelos parâmetros.

- **PARÂMETROS**

- atitle; // Título do formulário;
- aTemplate; // Modelo do formulário cuja a sintaxe segue abaixo:
- aOnEnter; // Executado ao entrar no formulário criado baseado no Template;
- aOnExit; // Executado ao sair do formulário criado baseado no Template;
- aOnEnterField; // Executado ao entrar um campo focado;
- aOnExitField; // Executado ao sair do campo focado;
- aOnCloseQuery // Executado ao fechar o form se CanClose = true;

- **SINTAXE DO MODELO**
  - Exemplo

```
const
  Template := ' Nome do Aluno: \sssssssssssssssssssssss'+lf+
              '      Endereço: \ssssssssssssssssssssssss‘sssssssss
              '           Cep: \##-###-###'+lf+
              '        Bairro: \sssssssssssssssssssssssss'+lf+
              '        Cidade: \sssssssssssssssssssssssss'+lf+
              '        Estado: \sssssssssssssssssssssssss'+lf+
              '         Idade: \BB'+lf+
              '     Mensalidade: \R,RRR.RR';
```

  - Tipos de dados do formulário:
    - s = Char alfanumérico
    - # = Char numérico
    - R = Double
    - B = Byte

- Exemplo de chamada da função:

```
if InputBox('Dados do aluno',
             ' Nome do Aluno: \sssssssssssssssssssssss'+lf+
             '      Endereço: \sssssssssssssssssssssss‘ssssssssssssss’+lf+
             '           Cep: \##-###-###'+lf+
             '        Bairro: \sssssssssssssssssssssss'+lf+
             '        Cidade: \sssssssssssssssssssssss'+lf+
             '        Estado: \sssssssssssssssssssssss'+lf+
             '         Idade: \BB'+lf+
             '     Mensalidade: \R,RRR.RR',
             nil,nil,nil,nil,nil
        ) = MrOk
then begin
     end;
```

74.6 Tipos

TOnEnterLocal

Declaração `TOnEnterLocal = Procedure(aDmxScroller:TUiDmxScroller);`

Descrição O tipo `TOnEnterLocal` é usado para implementar evento onEnterLocal do atributo **Mi_ScrollBox_LCL1**

TOnExitLocal

Declaração `TOnExitLocal = Procedure(aDmxScroller:TUiDmxScroller);`

Descrição O tipo `TOnExitLocal` é usado para implementar evento onExitLocal do atributo **Mi_ScrollBox_LCL1**

TOnEnterFieldLocal

Declaração `TOnEnterFieldLocal = Procedure(aField:pDmxFieldRec);`

Descrição O tipo `TOnEnterFieldLocal` é usado para implementar evento OnEnterFieldLocal do atributo **Mi_ScrollBox_LCL1**

TOnExitFieldLocal

Declaração `TOnExitFieldLocal = Procedure(aField:pDmxFieldRec);`

Descrição O tipo `TOnExitFieldLocal` é usado para implementar evento OnExitFieldLocal do atributo **Mi_ScrollBox_LCL1**

TOnCloseQueryLocal

Declaração `TOnCloseQueryLocal = Procedure(aDmxScroller:TUiDmxScroller; var CanClose:boolean);`

Descrição O tipo `TOnCloseQueryLocal` é usado para implementar evento OnCloseQueryLocal do atributo **Mi_ScrollBox_LCL1**.

- **NOTA***
  - **Este evento é disparado antes de desativar a classe **`TUiDmxScroller`(??)**

  .

  - Obs: Se o parâmetro **CanClose** for **false**, então a classe `TUiDmxScroller`(??) não é desativado.

Chapter 75

Unit umi_ui_InputBox_lcl

75.1 Descrição

A unit `umi_ui_InputBox_lcl` implementa o formulário `TMI_UI_InputBox`(**??**) usado para criar formulário baseado em Template `PSITem`(**??**).

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **PENDÊNCIAS**
  - T12 A propriedade autosize deve ser true após o form for criado.
  - T12 Dar
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
  - **2022-05-17**
    - T12 Análise de como será a classe `TMI_UI_InputBox`(**??**).
    - T12 Criar a unit `umi_ui_InputBox_lcl`.
    - T12 Criar formulário `TMI_UI_InputBox`(**??**);
    - T12 Adicionar o componente ButtonPanel1 e habilitar os botões ok e cancel;

    * T12 Adicionar o componente Mi_ScrollBox_LCL1 ;
    * T12 Criar evento: function DmxScroller_Form1GetTemplate;
    * T12 Criar atributo protected _FormSItem : `PSitem`(**??**);
    * T12 Criar propriedade Template:AnsiString;
        - Criar método Set_Template(aTemplate:AnsiString);

- **2022-05-18**
    * **10:51**
        - As alterações que fiz ontem no método `TObjectsMethods.StringToSItem`(**??**)() criou efeito colateral.
        - Corrigido.
    * **14:28**
        - Criar função:
        - function `InputBox`(**??**)(): TModalResult;

- **2022-05-19**
    * **11:13**
        - Criar os eventos
        - OnEnterLocal
        - OnExistLocal
        - onEnterFieldLocal
        - OnExitFieldLocal
        - Criar função:
        - function MI_MsgBox1MessageBox_04_PSItem(aPSItem: TMI_MsgBoxTypes.PSItem; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn): TModalResult;

75.2 Uses

- Classes
- SysUtils
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- ExtCtrls
- ButtonPanel
- uMi_ui_Dmxscroller_form
- mi_rtl_ui_Dmxscroller(??)
- uMi_ui_scrollbox_lcl(??)
- uMi_ui_Dmxscroller_form_ds

75.3 Visão Geral

TMI_UI_InputBox Classe

InputBox

75.4 Classes, Interfaces, Objetos e Registros

TMI_UI_InputBox Classe

Hierarquia

TMI_UI_InputBox > TForm

Descrição

A class TMI_UI_InputBox edita uma formulário passado em forma de Template(??) e devolve o formulário LCL caso o botão **OK** seja pressionando ou nil caso o botão **Cancelar** seja pressionando.

- **EXEMPLO**
  - O exemplo abaixo cria um formulário e permite interagir com com os eventos ao entrar e ao sair do formulário.

```
procedure InputBoxOnEnter(aDmxScroller: TUiDmxScroller);
 //Ao entrar no formulário este evento inicia os campos nome, endereço e cep

  procedure SetValue(Field:String;Value:String);
    var
      aField: pDmxFieldRec;
  begin
    with aDmxScroller do
    begin
      aField := FieldByName(Field);
      if aField<>nil
      Then aField.AsString:= value;
    end;
  end;

begin
  with aDmxScroller do
  begin
    setValue('nome','Paulo Henrique');
    setValue('endereço','Rua Francisco de Souza Oliveira, 15');
    setValue('cep','60310770');
  end;
end;

procedure InputBoxOnEnterField(aField: pDmxFieldRec);
    // Ao entrar no campo 01 e o mesmo for vazio inicializa-o com o nome Paulo Sérgio

begin
  Case aField.Fieldnum of
    1 : begin
          if aField.AsString = ''
          then aField.AsString := 'Paulo Sérgio';
        end;
    2 : begin end;
  end;

end;


Procedure InputBoxOnCloseQuery (aDmxScroller:TUiDmxScroller; var CanClose:boolean);
  //Esta função permite validar o formulário ao pressionar o botão ok.

   var
    idade : byte;
```

```
    s : string;
begin
  s := aDmxScroller.FieldByName('idade').AsString;
  idade := StrToInt(s);
  if  idade <> 64
  then begin
         ShowMessage('O campo idade <> de 64!.');
         CanClose := false;
       end
  else CanClose := true;
end;


Procedure TestInputBox;
    //Criar o formulário

  Var
    MI_UI_InputBox : TMI_UI_InputBox = nil;
begin
  with TDmxScroller_Form do
  if InputBox('Dados do aluno',
                ' Nome do Aluno: \Sssssssssssssssssssssss'ssssssssssssss'+ChFN+'Nome'+lf+
                '      Endereço: \Sssssssssssssssssssssss'ssssssssssssss'+ChFN+'endereco'+lf+
                '           Cep: \##-###-###'+ChFN+'cep'+lf+
                '        Bairro: \sssssssssssssssssssssss'+ChFN+'bairro'+lf+
                '        Cidade: \sssssssssssssssssssssss'+ChFN+'cidade'+lf+
                '        Estado: \SS'+ChFN+'estado'+lf+
                '         Idade: \BB'+ChFN+'idade'+lf+
                '     Matricula: \III'+ChFN+'matricula'+lf+
                '   Mensalidade: \R,RRR.RR'+ChFN+'mensalidade',
                InputBoxOnEnter,nil,

                InputBoxOnEnterField,nil,
                InputBoxOnCloseQuery,
                MI_UI_InputBox
           ) = MrOk
  then with MI_UI_InputBox.DmxScroller_Form1 do
       begin
         if FieldByName('nome').AsString = ''
         then begin
                ShowMessage('Campo "nome" não pode ser vazio');
              end;
         MI_UI_InputBox.free;
       end;
end;
```

```
procedure TDmxscroller_form_test.Button_InputBoxClick(Sender: TObject);
begin
  TestInputBox;
end;
```

Propriedades

Template

Declaração `public property Template:  AnsiString read _Template write Set_Template;`

Descrição A propriedade `Template` é usada para criar uma lista de `PSItem`(**??**) para ser usada como modelo do formulário.

- **NOTAS**
  - `Template` é um string comum, onde cada linha é separada com ^J.
  - `Template` tem uma lista de string com formato Dmx.
    - Formato da propriedade `Template`:

```
Template := ' Nome do Aluno: \Sssssssssssssssssssssss
            '      Endereço: \Sssssssssssssssssssssss
            '           Cep: \##-###-###'+ChFN+'cep'+lf
            '        Bairro: \sssssssssssssssssssssss
            '        Cidade: \sssssssssssssssssssssss
            '        Estado: \SS'+ChFN+'estado'+lf+
            '         Idade: \BB'+ChFN+'idade'+FldUpper
            '     Matricula: \III'+ChFN+'matricula'+lf+
            '      Valor da '+lf+
            '     Mensalidade: \R,RRR.RR'+ChFN+'mensali
```

  - **SINTAXE**
    - (til) : Limitador de rótulos do formulário;
    - **s** (s minúsculo) : caracteres alfanumérico incluindo os maiúsculas, minusculas, números e símbolos;

* **S** (S maiúsculo) : caracteres alfanumérico incluindo os maiúsculas, números e símbolos;
* # (# cancela) : Aceita somente números de 0 a 99
* **-** (literal) : Separador de números
* **B** (B maiúsculo): Campo do tipo byte
* **FldUpperLimit** : O caractere seguinte indica o limite superior da variável. No exemplo acima é 18 anos;
* **R** : Indica um caractere de um campo do tipo double;
* **I** : Indica um caractere de um campo do tipo interger. Faixa: -32000 a +32000;

Campos

ButtonPanel1

Declaração `public ButtonPanel1: TButtonPanel;`

DmxScroller_Form1

Declaração `public DmxScroller_Form1: TDmxScroller_Form;`

Mi_ScrollBox_LCL1

Declaração `public Mi_ScrollBox_LCL1: TMi_ScrollBox_LCL;`

Métodos

CancelButtonClick

Declaração `public procedure CancelButtonClick(Sender: TObject);`

CloseButtonClick

Declaração `public procedure CloseButtonClick(Sender: TObject);`

DmxScroller_Form1AddTemplate

Declaração `public procedure DmxScroller_Form1AddTemplate(const aUiDmxScroller: TUiDmxScroller );`

DmxScroller_Form1CloseQuery

Declaração `public procedure DmxScroller_Form1CloseQuery(aDmxScroller: TUiDmxScroller;var CanClose: boolean);`

DmxScroller_Form1Enter

Declaração `public procedure DmxScroller_Form1Enter(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form1EnterField

Declaração `public procedure DmxScroller_Form1EnterField(aField: pDmxFieldRec);`

DmxScroller_Form1Exit

Declaração `public procedure DmxScroller_Form1Exit(aDmxScroller: TUiDmxScroller);`

DmxScroller_Form1ExitField

Declaração `public procedure DmxScroller_Form1ExitField(aField: pDmxFieldRec);`

DmxScroller_Form1GetTemplate

Declaração `public function DmxScroller_Form1GetTemplate(aNext: PSItem): PSItem;`

DmxScroller_Form1NewRecord

Declaração `public procedure DmxScroller_Form1NewRecord(aDmxScroller: TUiDmxScroller);`

FormCloseQuery

Declaração `public procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);`

FormCreate

Declaração `public procedure FormCreate(Sender: TObject);`

OKButtonClick

Declaração `public procedure OKButtonClick(Sender: TObject);`

Set_Template

Declaração `protected Procedure Set_Template(aTemplate:AnsiString);`

Descrição O Método `Set_Template` inicia o atributo _Template e criar a lista _FormSItem : `PSitem`(**??**)

- **NOTAS**
  - Criar em TObjectss.TStringList o método

SetAlias

Declaração `public Procedure SetAlias(aTitle:AnsiString);`

75.5 Funções e Procedimentos

InputBox

Declaração
```
function InputBox( aTitle:  AnsiString; aTemplate:  AnsiString;
aOnEnterLocal:TOnEnterLocal ; aOnExitLocal:TOnExitLocal;
aOnEnterFieldLocal:TOnEnterFieldLocal; aOnExitFieldLocal:TOnExitFieldLocal;
aOnCloseQueryLocal:TOnCloseQueryLocal; out aMi_ui_InputBox :  TMI_UI_InputBox
):  TModalResult;
```

Descrição A função `InputBox` cria um formulário passado por Template e executa os eventos do formulário passado pelos parâmetros.

- **PARÂMETROS**
  - atitle; // Título do formulário;
  - aTemplate; // Modelo do formulário cuja a sintaxe segue abaixo:
  - aOnEnter; // Executado ao entrar no formulário criado baseado no Template;
  - aOnExit; // Executado ao sair do formulário criado baseado no Template;
  - aOnEnterField; // Executado ao entrar um campo focado;
  - aOnExitField; // Executado ao sair do campo focado;
  - aOnCloseQuery // Executado ao fechar o form se CanClose = true;
- **SINTAXE DO MODELO**

- Exemplo

```
const
  Template := ' Nome do Aluno: \sssssssssssssssssssssss'+lf+
              '      Endereço: \sssssssssssssssssssssssss‘ssssssss
              '           Cep: \##-###-###'+lf+
              '        Bairro: \ssssssssssssssssssssssss'+lf+
              '        Cidade: \ssssssssssssssssssssssss'+lf+
              '        Estado: \ssssssssssssssssssssssss'+lf+
              '         Idade: \BB'+lf+
              '   Mensalidade: \R,RRR.RR';
```

- Tipos de dados do formulário:
  - s = Char alfanumérico
  - # = Char numérico
  - R = Double
  - B = Byte

- Exemplo de chamada da função:

```
if InputBox('Dados do aluno',
             ' Nome do Aluno: \sssssssssssssssssssssss'+lf+
             '      Endereço: \sssssssssssssssssssssssss‘sssssssssssssss'+lf+
             '           Cep: \##-###-###'+lf+
             '        Bairro: \ssssssssssssssssssssssss'+lf+
             '        Cidade: \ssssssssssssssssssssssss'+lf+
             '        Estado: \ssssssssssssssssssssssss'+lf+
             '         Idade: \BB'+lf+
             '     Mensalidade: \R,RRR.RR',
             nil,nil,nil,nil,nil
        ) = MrOk
then begin
     end;
```

75.6 Tipos

TOnEnterLocal

Declaração `TOnEnterLocal = Procedure(aDmxScroller:TUiDmxScroller);`

Descrição O tipo `TOnEnterLocal` é usado para implementar evento onEnterLocal do atributo **Mi_ScrollBox_LCL1**

TOnExitLocal

Declaração `TOnExitLocal = Procedure(aDmxScroller:TUiDmxScroller);`

Descrição O tipo `TOnExitLocal` é usado para implementar evento onExitLocal do atributo **Mi_ScrollBox_LCL1**

TOnEnterFieldLocal

Declaração `TOnEnterFieldLocal = Procedure(aField:pDmxFieldRec);`

Descrição O tipo `TOnEnterFieldLocal` é usado para implementar evento OnEnterFieldLocal do atributo **Mi_ScrollBox_LCL1**

TOnExitFieldLocal

Declaração `TOnExitFieldLocal = Procedure(aField:pDmxFieldRec);`

Descrição O tipo `TOnExitFieldLocal` é usado para implementar evento OnExitFieldLocal do atributo **Mi_ScrollBox_LCL1**

TOnCloseQueryLocal

Declaração `TOnCloseQueryLocal = Procedure(aDmxScroller:TUiDmxScroller; var CanClose:boolean);`

Descrição O tipo `TOnCloseQueryLocal` é usado para implementar evento OnCloseQueryLocal do atributo **Mi_ScrollBox_LCL1**.

- **NOTA***
  - **Este evento é disparado antes de desativar a classe **`TUiDmxScroller`(??)**
  .
  - Obs: Se o parâmetro **CanClose** for **false**, então a classe `TUiDmxScroller`(??) não é desativado.

Chapter 76

Unit umi_ui_inputbox_lcl_test

76.1 Uses

- Classes
- SysUtils
- umi_ui_InputBox_lcl(??)

76.2 Visão Geral

TestinputBox

76.3 Funções e Procedimentos

TestinputBox

Declaração `Procedure TestinputBox;`

Chapter 77

Unit uMi_ui_Label_lcl

77.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- StdCtrls
- ActnList
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

77.2 Visão Geral

TMi_ui_Label_lcl Classe

Register

77.3 Classes, Interfaces, Objetos e Registros

TMi_ui_Label_lcl Classe

Hierarquia

TMi_ui_Label_lcl > TLabel

Propriedades

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_BitBtn_LCL`(**??**).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(**??**))

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

Métodos

DoOnClick

Declaração `protected procedure DoOnClick(Sender: TObject);`

77.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 78

Unit uMi_ui_maskedit_lcl

78.1 Uses

- Mask
- Windows
- SysUtils
- Messages
- Classes
- Controls
- StdCtrls
- Forms
- Grids
- dialogs
- LResources
- mi.rtl.Consts.StrError(??)
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

78.2 Visão Geral

TMI_MaskEdit_LCL Classe

Register

78.3 Classes, Interfaces, Objetos e Registros

TMI_MaskEdit_LCL Classe

Hierarquia

TMI_MaskEdit_LCL > TMaskEdit

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição O atributo `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_MaskEdit_LCL`(??).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(??))

Campos

_StringGrid

Declaração `public var _StringGrid: TStringGrid;`

Métodos

Create

Declaração `public constructor Create(AOwner:TComponent); override;`

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec(??)`

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec(??)` para o controle (Self).

DoOnMouseDown

Declaração `protected procedure DoOnMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);`

Descrição O método `DoOnMouseDown` seleciona todas as letras ou número do controle focado.

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

Descrição O método `DoOnEnter` ao receber o foco executa os métodos `GetBuffer(??)` e pDmxFieldRec.DoOnEnter(Self).

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e pDmxFieldRec.DoOnExit(Self).

DoEditNumberKeyPress

Declaração `protected procedure DoEditNumberKeyPress(Sender: TObject; var Key: char);`

Descrição O método `DoEditNumberKeyPress` edita os campos números de 1 a 10 bytes

DoOnKeyPress

Declaração `protected procedure DoOnKeyPress(Sender: TObject; var Key: system.Char);`

Descrição O método `DoOnKeyPress` não usado por enquanto???

GetHelpCtx_Hint

Declaração `protected FUNCTION GetHelpCtx_Hint(): AnsiString;`

Descrição O método `GetHelpCtx_Hint` captura a documentação do campo definido na classe onde o campo for criado.

- Com o programa **pasdoc** a documentação não precisa está no arquivo de recursos, por isso, para obter o link para o campo é preciso saber apenas o endereço do link.

GetSize

Declaração `protected FUNCTION GetSize(): Variant;`

SetSize

Declaração `protected PROCEDURE SetSize(aSize: Variant);`

SetAlias

Declaração `protected procedure SetAlias(const aAlias: AnsiString);`

GetName

Declaração `protected FUNCTION GetName(): AnsiString;`

GetAlias

Declaração `protected FUNCTION GetAlias: AnsiString;`

WMPaint

Declaração `protected procedure WMPaint(var Message: TLMPaint); message LM_PAINT;`

78.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 79

Unit umi_ui_mi_msgbox_dm

79.1 Uses

- Classes
- SysUtils
- Dialogs
- Graphics
- StdCtrls
- System.UITypes
- mi.rtl.objects.consts.mi_msgbox

79.2 Visão Geral

TMi_ui_mi_msgBox Classe

get_MI_MsgBox

79.3 Classes, Interfaces, Objetos e Registros

TMi_ui_mi_msgBox Classe

Hierarquia

TMi_ui_mi_msgBox > TDataModule

Campos

MI_MsgBox1

Declaração `public MI_MsgBox1: TMI_MsgBox;`

Métodos

MI_MsgBox1InputBox

Declaração `public function MI_MsgBox1InputBox(const aTitle, ALabel: AnsiString; var Buff; Template: AnsiString): TModalResult;`

MI_MsgBox1InputPassword

Declaração `public function MI_MsgBox1InputPassword(const aTitle: AnsiString; var aPassword: AnsiString): TModalResult;`

MI_MsgBox1InputValue

Declaração `public function MI_MsgBox1InputValue(const aTitle, aLabel: AnsiString; var aValue: Variant): TModalResult;`

MI_MsgBox1MessageBox

Declaração `public function MI_MsgBox1MessageBox(const aMsg: AnsiString): TModalResult;`

MI_MsgBox1MessageBox_03

Declaração `public function MI_MsgBox1MessageBox_03(const aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons): TModalResult;`

MI_MsgBox1MessageBox_04

Declaração `public function MI_MsgBox1MessageBox_04(aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn): TModalResult;`

MI_MsgBox1MessageBox_04_PSItem

Declaração `public function MI_MsgBox1MessageBox_04_PSItem(aPSItem: TMI_MsgBoxTypes.PSItem; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn ): TModalResult;`

MI_MsgBox1MessageBox_05

Declaração `public function MI_MsgBox1MessageBox_05(ATitle: AnsiString; aMsg: AnsiString; DlgType: TMsgDlgType; Buttons: TMsgDlgButtons; ButtonDefault: TMsgDlgBtn ): TModalResult;`

Alert

Declaração `public Procedure Alert(aTitle: AnsiString;aMsg:AnsiString);`

Descrição

- A procedure `Alert` executa um dialogo com botão **OK**

Confirm

Declaração `public Function Confirm(aTitle: AnsiString;aPergunta:AnsiString):Boolean;`

Descrição

- A função `Confirm` executa um dialogo com os botões **OK** e **Cancel** fazendo uma pergunta.
  - **RETORNA:**
    - **True** : Se o botão **OK** foi pŕessionando;
    - **False** : Se o botão **Cancel** foi pŕessionando.

Prompt

Declaração `public Function Prompt(aTitle: AnsiString;aPergunta:AnsiString;Var aResult: Variant):Boolean;`

Descrição

- A função `Prompt` mostra um dialogo com dois botões **OK** e **Cancel** e um campo input solicitando que o usuário digite um valor.
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pŕessionando;
    - **False** : Se o botão **cancel** foi pŕessionando.
    - **aResult** : Retorna a string digitada no formulário;

InputPassword

Declaração `public Function InputPassword(aTitle: AnsiString;out aUsername:AnsiString;out apassword:AnsiString):Boolean; Overload;`

Descrição

- A função `InputPassword` mostra um dialogo solicitando o login do usuário e a senha e dois botões **OK** e **Cancel**
  - **RETORNA:**
    - **True** : Se o botão **ok** foi pŕessionando;
    - **False** : Se o botão **cancel** foi pŕessionando.
    - **aUsername** : Retorna a string com nome do usuário.
    - **apassword** : Retorna a string com a senha do usuário.

InputPassword

Declaração `public Function InputPassword(aTitle: AnsiString;out apassword:AnsiString):Boolean; Overload;`

create

Declaração `public constructor create(aOwner:TComponent); override;`

79.4 Funções e Procedimentos

get_MI_MsgBox

Declaração `function get_MI_MsgBox:  TMi_ui_mi_msgBox;`

Chapter 80

Unit umi_ui_radiogroup_lcl

80.1 Uses

- Classes
- SysUtils
- LResources
- Forms
- Controls
- Graphics
- Dialogs
- ExtCtrls
- StrUtils
- ActnList
- mi_rtl_ui_DmxScroller(??)
- mi_rtl_ui_DmxScroller_Form(??)
- umi_ui_dmxscroller_form_lcl_attributes(??)

80.2 Visão Geral

TMI_RadioGroup_LCL Classe

Register

80.3 Classes, Interfaces, Objetos e Registros

TMI_RadioGroup_LCL Classe

Hierarquia

TMI_RadioGroup_LCL > TRadioGroup

Propriedades

DmxScroller_Form_Lcl_attributes

Declaração `published property DmxScroller_Form_Lcl_attributes : TDmxScroller_Form_Lcl_attributes Read _DmxScroller_Form_Lcl_attributes write SetDmxScroller_Form_Lcl_attributes;`

Descrição A propriedade `DmxScroller_Form_Lcl_attributes` contém o modelo e os cálculos do formulário

DmxFieldRec

Declaração `public property DmxFieldRec: pDmxFieldRec Read _pDmxFieldRec Write SeTDmxFieldRec;`

Descrição A propriedade `DmxFieldRec` fornece os dados necessários para criar o componente `TMI_Button_LCL`(??).

- **NOTA**
  - Esses dados devem ser criados pelo método DmxScroller_Form_Lcl_attributesr.Crea ATemplate : `TString`(??))

Métodos

GetBuffer

Declaração `public Procedure GetBuffer;`

Descrição O método `GetBuffer` ler os dados da propriedade `pDmxFieldRec`(??) para o controle (Self).

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

PutBuffer

Declaração `public Procedure PutBuffer;`

Descrição O método `PutBuffer` salva os dados do controle (Self) para a propriedade `pDmxFieldRec`(**??**)

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição O método `DoOnExit` ao perder o foco executa os métodos PuttBuffer e `pDmxFieldRec`(**??**)^.DoOnExit(Self).

80.4 Funções e Procedimentos

Register

Declaração `procedure Register;`

Chapter 81

Unit uMi_ui_scrollbox_lcl

81.1 Descrição

A unit `uMi_ui_scrollbox_lcl` implementa a classe `TMi_ScrollBox_LCL`(**??**) para ser usado como container para UiDmxScroller.

- **VERSÃO**
  - Alpha - 0.5.0.687
- **CÓDIGO FONTE**:
  -
- **PENDÊNCIAS**
  - O evento onGetTemplate não está funcionando de forma automática, preciso setá-lo.
  - Antes de executar onGetTemplate definir um nome de fonte padrão e checar se o mesmo precede a fonte editada na IDE.
- **HISTÓRICO**
  - Criado por: Paulo Sérgio da Silva Pacheco paulosspacheco@yahoo.com.br)
    - **2022-02-21**
      - Data em que essa unity foi criada.
    - **2022-02-22 14:00**
      - Documentar a unidade. -

* **2022-02-24 21:00**

  - Implementar os enventos OnEnter e OnExit de `TMi_ScrollBox_LCL`(**??**) para executar os eventos OnEnter e onExit de UiDmxScroller

* **2022-03-01 11:00**

  - Em `TMi_ScrollBox_LCL.Create`(**??**) inicializar a fonte fixa **Courie New** se windows ou **DejaVu Sans Mono** se linux e em ambas as plataformas foi definido o tamanho da fonte em 12 px.

* **2022-03-22 20:00**

  - Os eventos onEnter e OnExit não estavam executando `TUiDmxScroller.DoOnEnter`(**??**) e `TUiDmxScroller.OnExit`(**??**) caso o usuário não tenha iniciado os eventos OnEnter e OnExit do formulário isso gerava problema em GetBuffer e SetBuffer. Corrigido.

* **2022-06-27 18:19**

  - Redefinir procedure ComputeScrollbars; virtual;

  - Implementei mais meus problema continuaram. Tem algo muito errado no controle ScrollBox do Lazarus.

81.2 Uses

- `Classes`
- `SysUtils`
- `LResources`
- `Forms`
- `Controls`
- `Graphics`
- `Dialogs`
- `LMessages`
- `types`
- `mi_rtl_Ui_Methods`(**??**)
- `mi_rtl_ui_Dmxscroller`(**??**)

81.3 Visão Geral

`TMi_ScrollBox_LCL Classe`

`Register`

81.4 Classes, Interfaces, Objetos e Registros

TMi_ScrollBox_LCL Classe

Hierarquia

TMi_ScrollBox_LCL > TScrollBox

Descrição

A Classe `TMi_ScrollBox_LCL` foi redefinida para que os eventos DmxScroller.OnEnter e DmxScroller.OnExit sejam executados quando o ScrollBox recebe e perde o foco.

- **REFERNCIAS**

TScrollBox (https://lazarus-ccr.sourceforge.io/docs/lcl/forms/tscrollbox.html)

  - https://wiki.freepascal.org/Form_Tutorial (Form_Tutorial) -

Propriedades

UiDmxScroller

Declaração `public property UiDmxScroller : TUiDmxScroller read _UiDmxScroller write SetUiDmxScroller;`

Descrição A propriedade `UiDmxScroller` foi crada para que os eventos DmxScroller.OnEnter e DmxScroller.OnExit sejam executados quando o ScrollBox recebe o foco e perde o foco.

- A propriedade `UiDmxScroller` não deve ser published por que a mesma deve ser inicializada automatiocamente em TUiDmxScroller.SetParentLCL()

Métodos

DoOnEnter

Declaração `protected procedure DoOnEnter(Sender: TObject);`

Descrição A procedure `DoOnEnter` é usado para executar o evento onEnter de DmxScroller

DoOnExit

Declaração `protected procedure DoOnExit(Sender: TObject);`

Descrição A procedure `DoOnExit` é usado para executar o evento onExit de DmxScroller

Create

Declaração `public Constructor Create(aOwner: TComponent); Override;`

Descrição O constructor `Create` foi redefinido para que os eventos OnEnter e OnExit sajam iniciados com `DoOnEnter`(**??**) e `DoOnExit`(**??**) respectivamente.

SetUiDmxScroller

Declaração `protected Procedure SetUiDmxScroller(aUiDmxScroller:TUiDmxScroller);`

Descrição O método `SetUiDmxScroller` é usado para inicializar o atributo _UiDmxScroller

- **NOTA**
  - Caso a propriedade DmxScroller.Active seja **true**, então deve-se toná-la **false** para que perca o vínculo com a visão anterior.

Refresh

Declaração `public Procedure Refresh;`

WidthChar

Declaração `public function WidthChar:Byte;`

HeightChar

Declaração `public function HeightChar:Byte;`

81.5 Funções e Procedimentos

Register

Declaração `procedure Register;`

81.6 Tipos

PSItem

Declaração `PSItem = TUiMethods.PSItem;`

Chapter 82

Unit Unit1

82.1 Uses

- Classes
- SysUtils
- Forms
- Controls
- Graphics
- Dialogs
- uDmxScroller_Form_Lcl_add_test(??)

82.2 Visão Geral

TDmxScroller_Form_Lcl_add_test1 Classe

82.3 Classes, Interfaces, Objetos e Registros

TDmxScroller_Form_Lcl_add_test1 Classe

Hierarquia

TDmxScroller_Form_Lcl_add_test1 > TDmxScroller_Form_Lcl_add_test(??) > TForm

82.4 Variáveis

DmxScroller_Form_Lcl_add_test1

Declaração `DmxScroller_Form_Lcl_add_test1: TDmxScroller_Form_Lcl_add_test1;`