ICommand2

ICommand2 je základní interface programu CASE Studio 2.0.

function LoadFromFile(const FileName: WideString): IModel;
function LoadModelFromFile(const FileName: WideString): IModel;
Naète ze souboru FileName model a otevøe ho. Vrací interface IModel. Obì funkce jsou shodné.

procedure Cascade;
Srovná okna pøes sebe.

procedure Tile;
Srovná okna vedle sebe.

procedure Exit;
Ukonèí program. O uložení modelù se musí postarat programátor ještì pøed ukonèením !!!

procedure Show;
Zobrazí hlavní okno programu. (pøi spuštìní programu jako COM server je hlavní okno ukryté)

function ModelCount:integer;
Poèet modelù.

function Model:IModel;
Vrací interface Imodel;

procedure Hide;.
Ukryje hlavní okno programu.

IModel

function AttribCount: Integer;
Poèet atributù v modelu.

function Attrib(i: Integer): IAttribute;
Vrací interface atributu IAtribut. i mùže být v intervalu 0 .. AttribCount-1

function DatType(OidAttr: Integer): WideString;
Poskládá datový typ atributu do formátu `typ[(delka[,dec])]`. OidAttr je OID atributu.

function EntityCount: Integer;
Poèet entit v modelu.

function Entities(i: Integer): IEntity;
Vrací interface entity IEntity. i mùže být v intervalu 0..EntityCount-1

function RelCount: Integer;
Poèet relací v modelu.

function Rel(i: Integer): IRelation;
Vrací interface relace IRelation.

function DictCount: Integer;
Poèet uživatelských typù ve slovníku modelu.

function DictIt(i: Integer): IDictType;
Interface uživatelského typu IDictType. i v intervalu 0..DictCount-1

function TypeCount: Integer;
Poèet datových typù v aktuální šablonì.

function TypeIt(i: Integer): IDataType;
Interface datového typu IDataType ze šablony. i v intervalu 0..TypeCount-1

function IndexCount: Integer;
Poèet indexù v modelu.

function Index(i: Integer): IIndex;
Interface indexu IIndex. i v intervalu 0..IndexCount-1

function IndexItemsCount: Integer;
Poèet indexových položek v modelu.

function IndexItem(i: Integer): IIndexItem;
Interface indexove položky IIndexItem. i v intervalu 0..IndexItemsCount-1

function AlterKeysCount: Integer;
Poèet alternativních klíèù v modelu.

function AlterKey(i: Integer): IAlterKey;
Rozhraní alternativního klíèe IAlterKey. i v intervalu 0..AlterKeysCount-1

function AlterKeyItemsCount(i: Integer): Integer;

Poèet položek alternarivních klíèù v modelu.

function AlterKeyItem(i: Integer): IAlterKeyIt;
Rozhraní položky alternativního klíèe IAlterKeyIt. i v intervalu 0..AlterKeyItemsCount-1

function TriggersCount: Integer;
Poèet triggrù v modelu.

function Trigger(i: Integer): ITrigger;
Rozhraní triggru ITrigger. i v intervalu 0..TriggersCount-1. V triggrech jsou i uložené procedury a pohledy.

procedure SaveToFile(const fname: WideString);
Uloží model do souboru fname.

proceudre Save;
Uloží model do aktuálního souboru ModelFileName.

function Changed: WordBool;
V modelu byla zmìna, model je tøeba pøed ukonèením programu uložit.

function FindEntOid(i: Integer): IEntity;
Najde entitu podle jejího OID.

function FindAttrOid(i: Integer): IAttribute;
Najde atribut podle jejího OID.

function FindRelOid(i: Integer): IRelation;
Najde relaci podle jejího OID.

function FindDictItOid(i: Integer): IDictType;
Najde uživatelský typ podle jeho OID.

function FindTypeItOID(i: Integer): IDataType;
Najde datový typ podle jeho OID.

function FindIndexOid(i: Integer): IIndex;
Najde index podle jeho OID.

function FindIndexItOid(i: Integer): IIndexItem;
Najde položku indexu podle jejiho jejího OID.

property ModelFileName: WideString;
Název souboru, ve kterém je uložen model.

property ScriptBefore: WideString;
Text, který se vloží pøed generovaný SQL skript.

property ScriptAfter: WideString;
Text, který se vloží za generovaný SQL skript.

procedure ModelClose;
Uzavøe model. O uložení modelu se musí postarat programátor !!!

property ModelName: WideString;
property ProjectName: WideString;
property Version: WideString;

property Author: WideString;
property Company: WideString;
Promìnné, ve kterých jsou hodnoty, které znáte z dialogového okna "Vlastnosti modelu"

property ModelCreated: Double;
Datum vytvoøení modelu

property ModelModified: Double;
Datum poslední úpravy modelu.

IAttribute

property OID: Integer;
Identifikátor atributu.

property Name: WideString;
Název atributu.

property OIDEnt: Integer;
OID entity, které patøí atribut.

property OIDFkRel: Integer;
Pokud je atribut cizím klíèem, je v promìnné OID relace, pomocí které atribut migroval z parent entity. Jinak 0

property OidFKattr: Integer;
Pokud je atribut cizím klíèem, je v promìnné OID atributu, který je primární klíèem v parent entitì. Jinak 0.

property OidFkEnt: Integer;
Pokud je atribut cizím klíèem, je v promìnné OID parent entity. Jinak 0.

property AttrRoleName: WideString;
Název Role pro atribut, pokud je cizím klíèem.

property attrColName: WideString;
Fyzický název atribut (název sloupce)

property AttrDefault: WideString;
Defaultní hodnota atributu.

property AttrCheck: WideString;
Omezující podmínka atributu.

property AttrDef: WideString;
Volnì použitelná textová položka. V editoru atributù jako Def.

property AttrDef2: WideString;
Volnì použitelná textová položka. V editoru atributù jako Def2.

property AttrDef3: WideString;
Volnì použitelná textová položka. Pro možné budoucí využití.

property AtrDescription: WideString;
Poznámka.

property AttrPk: WordBool;
atribut je primárním klíèem.

property AttrNotNull: WordBool;
atribut je NOT NULL

property AttrLength: Integer;
Délka atributu (pokud to typ atributu vyžaduje)

property AttrDecimal;
Poèet desetinných míst v atributu. (pokud to typ atributu vyžaduje)

property OidType;
OID datového typu atributu. Prostøednictvím funkce `FindTypeItOID` z interface IModel získáte pomocí OidType rozhraní na IDataType. Pokud je atribut uživatelského typu, je v property 0.

property OidDict;
OID uživatelského typu. Prostøednictvím funkce `FindDictItOid` z interface IModel získáte pomocí OidDict rozhraní na IDictType. Pokud atribut není uživatelského typ, ale normálního datového typu, je v property 0.

property AttrUnique;
atribut má Unique integritní omezení.

property AttrUniqueConstraint;
Název constraint, pod kterým bude vygenerován.

property AttrCheckConstraint;
Název Check constraint atributu.

property AttrFk: WordBool; readonly
atribut je cizím klíèem.

IEntity

property OID: Integer;
Identifikátor entity.

property Name: WideString;
Název entity.

property EntTableName: WideString;
Fyzický název entity. Vygeneruje se jako název tabulky.

property Trigger: WideString;
Trigger, který patøí k entitì. Také zde mùžete zapsat nìkteré složitìjší constrainty.

property Storage;
Parametry fyzického uložení databáze. Napø. Tablespace.

property Description: WideString;
Poznámka, jakýkoliv text.

property Gener: WordBool;
Entita se bude generovat do SQL skriptu. Jinak to je v checkboxu v editoru entit, a v checklistboxu v dialogovém oknì Generování SQL skriptu.

IRelation

property OidEntParent: Integer;
OID parent entity.

property OidEntChild: Integer;
OID child entity.

property RelType: tTypeRelation;
Typ relace.

```
 TypeRelIdent = 0;
TypeRelNonIdent = 1;
TypeRelInfo = 2;
```
Identifikaèní relace.
Neidentifikaèní relace.
Informativní relace.

property RelParcParent: tTypeParc;
Parcialita parent entity vùèi child entitì.

```
Mandatory = 0;
Optional = 1;
```
Povinná
Nepovinná

property RelParcChild: tTypeParc;
Parcialita child entity vùèi parent entitì. Typ je stejný jako u RelParcParent.

property RelRefParentUpdate: tTypeIntegrity;
Typ referenèní integrity pro parent pøi update

```
  tTypeNone = 0;
  tTypeSetNull = 1;
  tTypeRestrict = 2;
  tTypeCascade = 3;
  tTypeSetDefault = 4;
```

property RelRefParentInsert: tTypeIntegrity;
Typ referenèní integrity pro parent pøi insertu.
Je stejný jako u RelRefParentUpdate.

property RelRefParentDelete: tTypeIntegrity;
Typ referenèní integrity pro parent pøi delete.

property RelRefChildUpdate: tTypeIntegrity;
Typ referenèní integrity pro child pøi update.

property RelRefChildInsert: tTypeIntegrity;
Typ referenèní integrity pro child pøi insertu.

property RelRefChildDelete: tTypeIntegrity;
Typ referenèní integrity pro child pøi delete.

property ParentKeyType: tTypeParentKey;
Typ klíèe v parent entitì.

`tTypePk = 0;` Klíèem je primární klíè.
`tTypeUniqueAttr = 1;` Klíèem je Unique atribut.
`tTypeAlterKey = 2;` Klíèem je alternativní klíè.

property OidParentkey: Integer read;
Pokud je v parent entitì klíèem primární klíè, je v této promìnné 0.
Pokud je v parent entitì klíèem unique atribut, je v této promìnné OID tohoto atributu.
Pokud je v parent entitì klíèem alternativní klíè, je v této promìnné OID tohoto alternativního klíèe.

property ConstraintFkName: WideString;
Fyzický název relace. Pokud to SQL server podporuje, generuje se jako `Constraint <ConstraintFkName> Foreign key`

property RelDescription: WideString;
Poznámka. Libovolný text.

ITrigger

Triggry ze seznamu triggrù. Jsou tu taky uložené procedury a pohledy. Viz TypeTrigger

property OID: Integer;
Identifikátor triggeru.

property Name: WideString;
Název triggeru.

property Source: WideString;
Zdrojový kod triggru (procedury, pohledu)

property TypeTrigger: tTypeTrigger;
Urèuje, zda jde o trigger, uloženou procedure, nebo pohled. Mùže nabývat tìchto hodnot:
tTypeTrig = 0;
tTypeProc = 1;
tTypeView = 2;

property Enabled: WordBool read Get_Enabled write Set_Enabled;
Pokud je nastaven na True, bude se trigger (procedura, pohled) generovat do SQL skriptu.

IDataType

Všechny promìnné jsou pouze ke ètení.

property OID: Integer;
Identifikátor datového typu.

property Name: WideString;
Název datového typu.

property NamePh: WideString;
Fyzický název datového typu.

property LengthExists: WordBool;
Datový typ podporuje délku. Napø. Char(20)

property DecimalExists: WordBool;
Datový typ podporuje i poèet desetinných míst. Napø. Decimal(10,5). Pokud datový typ nepodporuje délku, nemá tato promìnná význam.

property LengthMax: Integer;
Maximální délka datového typu.

property LengthMin: Integer;
Minimální délka datového typu.

property LengthDefault: Integer;
Defaultní délka datového typu. Tato délka se použije jako základní pøi napø. pøi vytvoøení atributu.

property DecimalMax: Integer;
Maximální poèet desetinných míst datového typu.

property DecimalMin: Integer;
Minimální poèet desetinných míst datového typu.

property DecimalDefault: Integer;
Defaultní poèet desetinných míst.

IDictType

property OID: Integer;
Identifikátor slovníkového typu.

property Name: WideString;
Název slovníkového typu.

property NamePh: WideString;
Fyzický název slovníkového typu. Využije se pøedevším tehdy, pokud SQL server podporuje domény.

property OidType: Integer;
OID datového typu, viz IDataType

property DictLength: Integer;
Délka slovníkového typu.

property DictDecimal: Integer;
Poèet desetinných míst slovníkového typu.

property DictDefault: WideString;
Defaultní hodnota slovníkového typu. Použije se pøi vytvoøení domény, nebo ji pøevezme atribut.

property DictCheck: WideString;
Check slovníkového typu.Použije se pøi vytvoøení domény, nebo ji pøevezmr atribut.

property DictDef: WideString;
Volnì použitelná textová položka.
Viz také

property DictDef2: WideString;
Volnì použitelná textová položka.

property DictDef3: WideString;
Volnì použitelná textová položka. Zatím se nepoužívá.

property DictDescription: WideString;
Poznámka. Jakýkoliv text.

IIndex

property OID: Integer;
Identifikátor indexu.

property Name: WideString;
Název indexu.

property OIDEnt: Integer;
OID entity, které index patøí. Viz také IEntity.

property IndexUnique: WordBool;
Index je Unique.

property IndexDescend: WordBool;
Index je sestupný.

property IndexClustered: WordBool;
Index je klastrovaný.

property IndexExpression: WideString;
Index je tvoøen indexovým výrazem. Zatím použito pouze pro Clipper.

property IndexFilter: WideString;
Index je filtrovaný. Zatím nepoužito.

property IndexCaseSensitive;
Index rozlišuje velká a malá písmena. Zatím nepoužito.

property Storage: WideString;
Parametry fyzického uložení indexu. Napø. tablespace.

IIndexItem

Co to je indexová položka ?

V pøíkazu `Create Index OsobaJm on Osoba(Jmeno,Prijmeni)` jsou indexovými položkami `Jmeno` a `Prijmeni`

property OID: Integer;
Identifikátor indexové položky.

property Name: WideString;
Název indexové položky. V nìkterých pøípadech (po editaci atributu) mùže být v promìnné nesprávná hodnota. Doporuèuji název položky najít pomocí funkce `FindAtrOid` z rozhraní IModel.

property OidInd: Integer;
OID indexu, kterému patøí indexová položka. Viz také.

property OidAttr;
OID atributu, ze kterého byla indexová položka vytvoøena. Viz také.

property IndItDescend;
Indexová položka je sestupná. Pokud to SQL server podporuje.

IAlterKey

property OID: Integer;
Identifikátor alternativního klíèe.

property Name: WideString;
Název alternativního klíèe.

property NamePh: WideString;
Fyzický název alternativního klíèe. Pokud to SQL server podporuje, generuje se jako `Constraint Unique`.

IAlterKeyIt

property OID: Integer;
Identifikátor položky alternativního klíèe.

property Name: WideString;
Název položky alternativního klíèe. V nìkterých pøípadech (po editaci atributu) mùže být v promìnné nesprávná hodnota. Doporuèuji název položky najít pomocí funkce `FindAtrOid` z rozhraní IModel.

property OidAlterKey:Integer;
OID alternativního klíèe, kterému patøí položka alternativního klíèe. Viz také.

property OidAttr:Integer;
OID atributu, ze kterého je alternativní položka vytvoøena. Viz také.

Pøíklad

Výpis seznamu entit a všech jejích atributù.

```
var i,j:integer;
    c:ICommand2;
    m:IModel;
    e:IEntity;
    a:IAtribut;
begin
c := CoCommand2.Create;                          // spuštìní COM serveru
c.Show;                                          // zobrazení programu
(nepovinné)
m := c.LoadFromFile('c:\proj\test.dm2');         // naètení modelu ze souboru
for i := 0 to m.EntityCount-1 do                 // prochází všechny entity
   begin
   e := m.Entities(i);                           // získá interface na entitu s
indexem i
   writeln(e.Name);                              // vypíše její název
   for j := 0 to m.AttribCount-1 do              // prochází všechny atributy
      begin
      a := m.Attrib(j);                          // získá interface na atribut
      if a.OIDEnt = e.OID then                   // pokud atribut patøí k
entitì,
         begin
         writeln(a.Name);                        // vypíše její název
         end;
      end;
   end;
end;
```

Pokud jsou interface **`c`** a **`m`** lokální, model a COM server se automaticky zruší po odchodu programu z oblasti platnosti tìchto promìnných. Pokud c a m deklarujete jako globální promìnné, nebo promìnné objektu, musíte model uzavøít `m.ModelClose` a celý COM server uzavøete pøíkazem `c.Exit;`

Pokud ve vašem vývojovém prostøedí nemáte pøístup k TLB knihovnì, použijte tento postup.

Pøíklad 2

Pokud ve svém vývojovém prostøedí nemáte pøístup k TLB knihovnì, mùžete použít pøístup pøes. Pøíklad je napsán pro Delphi, ale tímto zpùsobem mùžete CASE Studio ovládat z Accessu, VB, MS Wordu apod.

```
var i,j:integer;
    c:Variant;
    m:Variant;
    e:Variant;
    a:Variant;
begin
c := CreateOleServer('CaseStudio.Command2');     // spuštìní COM serveru
c.Show;                                          // zobrazení programu
(nepovinné)
m := c.LoadFromFile('c:\proj\test.dm2');         // naètení modelu ze souboru
for i := 0 to m.EntityCount-1 do                 // prochází všechny entity
   begin
   e := m.Entities(i);                           // získá interface na entitu s
indexem i
   writeln(e.Name);                              // vypíše její název
   for j := 0 to m.AttribCount-1 do              // prochází všechny atributy
      begin
      a := m.Attrib(j);                          // získá interface na atribut
      if a.OIDEnt = e.OID then                   // pokud atribut patøí k
entitì,
         begin
         writeln(a.Name);                        // vypíše její název
         end;
      end;
   end;
end;
```

Seznam SQL serverù

OID	Název	
1	Paradox	
10	Interbase 4	
20	Clipper 5	(ve vývoji)
30	Oracle	
40	Sybase Anywhere	
60	MS SQL	
70	Ingres	
80	Interbase 5	
90	Informix	