Obsah

IModel

Vlastnosti Metody Rozhraní

Obecné rozhraní k aktuálnímu modelu.

Popis
Používejte rozhraní `Model` k získání základních informací k aktuálnímu modelu. `Model` zpøístupòuje všechna dùležitá rozhraní k :

- entitnì relaènímu modelu,
- modelu datových tokù,
- všem šablonám,
- základním informacím k modelu.

IModel - vlastnosti

IModel Legenda

Obecné

- Created
- Modified

Rozhraní k DFD

- CountDataFlows
- CountDataStores
- CountProcesses
- CountTerminators

Rozhraní k ERD

- CountAlterKeys
- CountAlterKeyItems
- CountAttributes
- CountDataTypes
- CountDictTypes
- CountEntities
- CountIndexes
- CountIndexItems
- CountRelations
- CountTriggers
- CountUserRoles
- CountUserPermiss
- CountUsers
- CountUserRoleUsers

Rozhraní k šablonám

- CountDatabaseTemplates
- CountModelTemplates
- CountSystemTemplates
- CountUDFDatabaseTemplates
- CountUDFSystemTemplates

IModel - metody

IModel Legenda

Rozhraní k DFD

—DataFlows
—DataStores
  GetDataFlow
  GetDataStore
  GetProcess
  GetTerminator
—Processes
—Terminators

Rozhraní k ERD

—AlterKeys
—AlterKeyItems
—Attributes
—DataTypes
—DictTypes
—Entities
  GetAlterKey
  GetAlterKeyItem
  GetAttribute
  GetDataType
  GetDictType
  GetEntity
  GetIndex
  GetIndexItem
  GetRelation
  GetTrigger
  GetUserRole
  GetUserPermiss
  GetUserGetUserRoleUser
—Indexes
—IndexItems
—ModelTemplates
—Relations
—Triggers
—UserRoles
—UserPermiss
—Users
—UserRoleUsers

Rozhraní k šablonám

  DatabaseTemplates
  GetModelTemplate
  SystemTemplates
  UDFSystemTemplated
  UDFDatabaseTemplates

IEntity

Vlastnosti Metody Rozhraní

Rozhraní k entitì.

Popis
Používejte rozhraní `IEntity` k získání základních informací k jedné konkretní entitì. `IEntity` zpøístupòuje mimo jiné všechna dùležitá rozhraní ke všem :

- atributùm,
- alternativním klíèùm,
- relacím a
- indexovùm

této entity.

IEntity - vlastnosti

IEntity Legenda

Základní

- Id
- Name

Vlastní

- CountAlterKeys
- CountAttributes
- CountIndexes
- CountRelations

Dependent
Description
Generate
UserTrigger
Storage
PKConstraint
TableName

IEntity - metody

IEntity Legenda

Rozhraní

AlterKeys
Attributes
Indexes
Relations

IAttribute

Vlastnosti Metody Rozhraní

Rozhraní k atributu.

Popis
Používejte rozhraní `IAttribute` k získání základních informací ke konkretnímu atributu.

IAttribute - vlastnosti

IAttribute Legenda

Základní

- Id
- Name

Vlastní

- Check
- CheckConstraint
- ColName
- CountFk
- DataTypeId
- Def
- Def2
- Default
- DefaultConstraint
- Description
- DictTypeId
- FK
- EntityId
- PK
- RoleName
- SQLDataType
- Unique
- UniqueConstraint

Attribute - metody

IAttribute Legenda

Informaèní

AttrFkId
EntityFkId
RelationFkId

IDataType

Vlastnosti Metody Rozhraní

Rozhraní k datovým typùm databáze.

Popis
Používejte rozhraní `IDatatype` k získání základních informací ke všem dostupným datovým typùm aktuální databáze.

IDataType - vlastnosti

IDataType Legenda

Základní

- Id
- Name

Vlastní

DataTypeName
DefaultDecimal
DefaultLength
ExportToId
ExportToLength
ExportToDecimal
IsDecimal
IsLength
MaxDecimal
MaxLength
MinDecimal
MinLength
TransformCode
TransformTo

IDataType - metody

IDataType Legenda

Nejsou

IDictDataType

Vlastnosti Metody Rozhraní

Rozhraní ke slovníkovím datovým typùm databáze.

Popis
Používejte rozhraní `IDictDatatype` k získání základních informací ke všem uživatelem definovaným slovníkovým datovým typùm k aktuálnímu modelu.

IDictDataType - vlastnosti

IDictDataType Legenda

Základní

- Id
- Name

Vlastní

Check
DataTypeId
Decimal
Def
Def2
Default
Description
DictName
Length

IDictDataType - metody

Nejsou

IRelation

Vlastnosti Metody Rozhraní

Rozhraní k relacím.

Popis
Používejte rozhraní `IRelation` k získání základních informací ke všem relacím aktuálního modelu.

IRelation - vlastnosti

IRelation Legenda

Základní

- Id
- Name

Vlastní

ChildDeleteIntegrity:Integer
- ChildEntityId:Integer; ID child entity
ChildInsertIntegrity:Integer
ChildUpdateIntegrity:Integer
Description: WideString Popis
FKConstraint: WideString Název constraint cizího klíèe
ParentDeleteIntegrity:Integer
- ParentEntityId:Integer; ID Parent entity.
ParentInsertIntegrity:Integer
ParentUpdateIntegrity:Integer
PartialityParent: Integer
PartialityChild: Integer Viz. ParcialityParent
- Type:Integer; Typ relace.

IRelation - metody

IRelation Legenda

Nejsou

IIndex

Vlastnosti Metody Rozhraní

Rozhraní k databázovým indexùm.

Popis
Používejte rozhraní `IIndex` k získání základních informací ke všem databázovým indexùm aktuálního modelu.

IIndex - vlastnosti

IIndex Legenda

Základní

- Id
- Name

Vlastní

- CaseSensitive
- Clustered
- Descending
- EntityId
- ExprType
- Expression
- Filter
- Storage
- Unique

IIndex - metody

IIndex Legenda

Nejsou

IIndexItem

Vlastnosti Metody Rozhraní

Rozhraní k položkám databázových indexùm.

Popis
Používejte rozhraní `IIndexItem` jako seznam vazeb mezi všemi databázovými indexy a atributy aktuálního modelu.

IIndexItem - vlastnosti

IIndexItem Legenda

Základní

- Id
- Name

Vlastní

- AttributeId
- IndexId
- Descending

IIndexItem - metody

IIndexItem Legenda

Nejsou

ITrigger

Vlastnosti Metody Rozhraní

Rozhraní k uživatelským triggerùm, procedurám a pohledùm.

Popis
Používejte rozhraní `ITrigger` k získání všech základních informací ke všem uživatelským triggerùm, procedurám a pohledùm aktuálního modelu.

ITrigger - vlastnosti

ITrigger Legenda

Základní

- Id
- Name

Vlastní

- Enabled
- Source
- Type

ITrigger - metody

ITrigger Legenda

Nejsou

ITemplate

Vlastnosti Metody Rozhraní

Obecné rozhraní k šablonám.

Popis
Používejte rozhraní `ITrigger` k získání všech základních informací ke všem šablonám všech úrovní a všech kategorií.

ITemplate - vlastnosti

ITemplate Legenda

Základní

- Id
- Name

Vlastní

- Author
- Category
- Company
- Created
- Description
- Enabled
- EnityId
- Level
- MainScript
- Modified
- ScriptLanguage
- Source
- Version

ITemplate - metody

ITemplate Legenda

Nejsou

IAlternateKey

Vlastnosti Metody Rozhraní

Rozhraní k alternativním klíèùm.

Popis
Používejte rozhraní `IAlternateKey` k získání všech základních informací ke všem altertnativním klíèùm aktuálního modelu.

IAlternateKey - vlastnosti

IAlternateKey Legenda

Základní

- Id
- Name

Vlastní

- AlterKeyName
- EntityId

IAlternateKey - metody

IAlternateKey Legenda

Nejsou

IAlternateKeyItem

Vlastnosti Metody Rozhraní

Rozhraní k položkám alternativních klíèù.

Popis
Používejte rozhraní `IAlternateKeyItem` jako seznam vazeb mezi všemi alternativními klíèí a atributy aktuálního modelu.

IAlternateKeyItem - vlastnosti

IAlternateKeyItem Legenda

Základní

- Id
- Name

Vlastní

- AlterKeyId
- AttributeId

IAlternateKeyItem - metody

IAlternateKeyItem Legenda

Nejsou

IUserRole

Vlastnosti Metody Rozhraní

Rozhraní k uživatelským rolím.

Popis
Používejte rozhraní `IUserRole` k získání všech základních informací ke všem uživatelským rolím aktuálního modelu.

IUserRole - vlastnosti

IUserRole Legenda

Základní

- Id
- Name

Vlastní

Description
Generate
UserRoleName

IUserRole - metody

IUserRole Legenda

Nejsou

IUser

Vlastnosti Metody Rozhraní

Rozhraní k definovaným uživatelùm.

Popis
Používejte rozhraní `IUser` k získání všech základních informací ke všem definovaným databázovým uživatelùm aktuálního modelu.

IUser - vlastnosti

IUser Legenda

Základní

- Id
- Name

Vlastní

Description
Generate
UserName

IUser - metody

IUser Legenda

Nejsou

IPermission

Vlastnosti Metody Rozhraní

Rozhraní k databázovým právùm.

Popis
Používejte rozhraní `IPermission` k získání všech základních informací ke všem definovaným databázovým právùm uživatelù k jednotlivým objektùm aktuálního modelu.

Note
Práva lze pøidìlit dle možností databáze k:

- entitám,
- procedurám,
- triggerùm a

pohledùm.

IPermission - vlastnosti

IPermission Legenda

Základní

- Id
- Name

Vlastní

- CanDelete
- CanDRI
- CanExecute
- CanInsert
- CanSelect
- CanUpdate
- ObjectId
- ObjectType
- OwnerId
- OwnerType

Prává

Delete
DRI
Execute
Insert
Select
Update

IPermission - metody

Nejsou

IUserRoleUser

Rozhraní k seznamu vazeb mezi uživatelí a uživatelskými rolí.

Popis
Používejte rozhraní `IUserRoleUser` jako seznam vazeb mezi všemi databázovými uživatelí a databázovými uživatelskými rolí aktuálního modelu.

IUserRoleUser - vlastnosti

IUserRoleUser Legenda

Základní

- Id
- Name

Vlastní

- RoleId
- UserId

IUserRoleUser - metody

IUserRoleUser Legenda

Nejsou

IProcess

Vlastnosti Metody Rozhraní

Rozhraní k procesu.

Popis
Používejte rozhraní `IProcess` k získání všech základních informací ke všem definovaným procesùm všech úrovní aktuálního modelu datových tokù.

IProcess - vlastnosti

IProcess Legenda

Základní

- Id
- Name

Vlastní

- Description
- LowestLevel
- ParentId
- ProcessId
- Specification

IProcess - metody

IProcess Legenda

Nejsou

IDataStore

Vlastnosti Metody Rozhraní

Rozhraní k datastoru.

Popis
Používejte rozhraní `IDataStore` k získání všech základních informací ke všem definovaným datastorùm všech úrovní aktuálního modelu datových tokù.

IDataStore - vlastnosti

IDataStore Legenda

Základní

- Id
- Name

Vlastní

- Description
- EntityId
- ParentId
- ProcessId

IDataStore - metody

IDataStore Legenda

Nejsou

ITerminator

Vlastnosti Metody Rozhraní

Rozhraní k terminátoru.

Popis
Používejte rozhraní `ITerminator` k získání všech základních informací ke všem definovaným terminátorùm všech úrovní aktuálního modelu datových tokù.

ITerminator - vlastnosti

ITerminator Legenda

Základní

- Id
- Name

Vlastní

- Description
- ParentId
- ProcessId

ITerminator - metody

ITerminator Legenda

Nejsou

IDataFlow

Vlastnosti Metody Rozhraní

Rozhraní k datovému toku.

Popis
Používejte rozhraní `IDataFlow` k získání všech základních informací ke všem definovaným datovým tokùm všech úrovní aktuálního modelu datových tokù.

IDataFlow - vlastnosti

IDataFlow Legenda

Základní

- Id
- Name

Vlastní

- Description
- FromId
- FromObjType
- ParentId
- ProcessId
- ToId
- ToObjType
- Type

IDataFlow - metody

IDataFlow Legenda

Nejsou

ISysUtils

Vlastnosti Metody Rozhraní

Rozhraní k pomocným funkcím.

Popis
Používejte rozhraní `ISysUtils` jako pøístup ke knihovnì funkcí pro všeobecné použití.

Poznámka
Funkce se volají bez explicitního urèení rozhraní, takže staèí napø. napsat `AnsiUpperCase(x)`, nemusíte psát `SysUtils.AnsiUpperCase(x)`. Samozøejmì fungují oba zpùsoby.

ISysUtils - vlastnosti

ISysUtils Legenda

Nejsou

ISysUtils - metody

ISysUtils Legenda

- AnsiUpperCase
- Copy
- Date
- DateTimeToStr
- DateToStr
- Day
- Delete
- Insert
- LowerCase
- Month
- Now
- IntToStr
- ReadAnsiLowerCase
- StrToInt
- Time
- TimeToStr
- Trim
- TrimLeft
- TrimRight
- UpperCase
- UserVarToBool
- Year

ITextStream

Vlastnosti Metody Rozhraní

Rozhraní k textovému souboru.

Popis
Používejte rozhraní `ITextStream pro jednoduchý pøístup k textovým souborùm. Rozhraní Vám umožní vytváøet, otevírat anebo ukládat textové soubory.`

Poznámka
Rozhraní se využívá napø. pøi generování SQL skriptù, RTF anebo HTML dokumentù.

ITextStream - vlastnosti

ISysUtils Legenda

FileName
Text

ITextStream - metody

ISysUtils Legenda

- Clear
- Load
- LoadFromFile
- Save
- SaveToFile
- Write
- Writeln

ILog

Vlastnosti Metody Rozhraní

Rozhraní k protokolovacímu souboru.

Popis
Používejte rozhraní `ILog` k vytváøení protokolovacího souboru prùbìhu zpracování skriptu. Všechny událostí zapsané do souboru se prùbìžnì zobrazují v aplikaèním oknì Log.

ILog - vlastnosti

ILog Legenda

Nejsou

ILog - metody

ILog Legenda

- Clear
- SaveToFile
- Write
- WriteIn

IRtf

Vlastnosti Metody Rozhraní

Rozhraní k tvorbì Rtf souborù.

Popis
Používejte rozhraní `IRtf` k snadnému vytvoøení vlastních RTF souborù (rich text files).

Poznámky
Rtf soubory mají výhodu, že lze vytvoøit formátovaný text, ktery je možno si prohlídnout anebo upravit na velké vìtšinì textových editorù. (Napr. MS Word, WordPad, 602Text a mnoha dalších) .

IRtf - vlastnosti

IRtf Legenda

Text

IRtf - metody

IRtf Legenda

Základní

- New
- SaveToFile
- Write
- Writeln

Styly

- DefineStyle
- SetStyle

Tabulky

- DefineCell
- DefineColumn
- DefineTable
- DrawTable

Záložky

- CallBookmark
- DefineBookmark
- SetBookmark

IVariables

Vlastnosti Metody Rozhraní

Rozhraní ke správì promìnných.

Popis

Používejte rozhraní `IVariables` k definici vlastních uživatelských promìnných k danému skriptu. Takto definované promìnné se objeví vždy ve složce "Uživatelské promìnné skriptu" a proto mùže koncový uživatel skriptu jejich hodnotu lehce zmìnit. Promìnné jsou dostupné bìhem zpracování celého skriptu.

Poznámky

Promìnné se musí ve skriptu definovat vždy ve funkci DefineVariables. Každá definovaná promìnná se stává automaticky vlastností rozhraní. (Napø. Variables.MojePromenna = 5).

IVariables - vlastnosti

IVariables Legenda

Type

IVariables - metody

IVariables Legenda

Základní

- DefineVariable

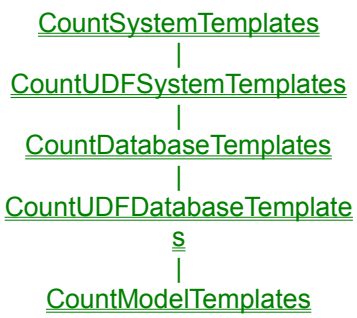
Vlastnosti

- urèená pouze ke ètení
- urèená pouze k zápisu

Vlastnosti

- metoda nevrací výsledek

Hiearchie šablon



ID

property ID:integer

Identifikátor objektu.

Name

Název objektu

Model.CountEntities

Model Pøíbuzné Pøíklad

Vrací poèet entit v modelu.

```
property CountEntities: Integer; readonly;
```

Popis
Použijte vlastnost `CountEntities`, k získání celkového poètu entit v aktuálním ER-modelu.

Entities

Model

function Entities(index:integer):IDispatch;

Vrací rozhraní na entitu podle poøadí. Poøadí je od 0 do Model.CountEntities-1

pøíklad, který vypíše všechny názvy entit v modelu.

```
function Main()
{
for (i=0; i<Model.CountEntities; i++)
   {
   Ent = Model.Entities(i);
   Log.Writeln(Ent.Name);
   }
}
```

Pøíklad k Entity, CountEntities

Pøíklad v JavaScriptu vypíše názvy entit aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountEntities; i++)
   {
   Entity = Model.Entities(i);
   Log.Writeln(Entity.Name);
   };
};
```

Model.CountAttributes

Model Pøíbuzné Pøíklad

Vrací poèet atributù v modelu.

```
property CountAttributes: Integer; readonly;
```

Popis
Použijte vlastnost `CountAttributes`, k získání celkového poètu atributù všech entit v aktuálním modelu.

Pøíklad k Attributes, CountAttributes

Pøíklad v JavaScriptu vypíše názvy všech atributù aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountAttributes; i++)
   {
   Atr = Model.Attributes(i);
   Log.Writeln(Atr.Name);
   };
};
```

Model.Attributes

Vrací rozhraní na atribut podle poøadí. Poøadí je od 0 do Model.CountAttributes-1

pøíklad:

```
function Main()
{
for (i=0; i<Model.CountAttributes; i++)
   {
   Atr = Model.Attributes(i);
   Log.Writeln(Atr.Name);
   }
}
```

Model.CountDataTypes

Vrací poèet datových typù.

```
property CountDataTypes: Integer; readonly;
```

Popis
Použijte vlastnost `CountDataTypes`, k získání celkového poètu datových typù definovaných k databázi aktuálního ER-modelu.

Model.DataTypes

function DataTypes(index:integer):IDispatch;

Vrací rozhraní na datový typ podle poøadí. Poøadí je od 0 do Model.CountDataTypes-1;

pøíklad vypíše seznam všech typù.

```
function Main()
{
for (i=0; i<Model.CountDataTypes; i++)
   {
   t = Model.DataTypes(i);
   Log.Writeln(t.Name);
   }
}
```

Model.DataTypes

Model Pøíbuzné Pøíklad{keepn}

Vrací poèet atributù v modelu.

```
property CountAttributes: Integer; readonly;
```

Popis
Použijte vlastnost `CountAttributes`, k získání celkového poètu atributù všech entit v aktuálním modelu.

Pøíklad k DataTypes, CountDataTypes

Pøíklad v JavaScriptu vypíše názvy všech typù definovaných k databázi aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountDataTypes; i++)
   {
   DataType = Model.DataTypes(i);
   Log.Writeln(DataType.Name);
   };
};
```

Model.CountDictTypes

Model Pøíbuzné

Vrací poèet slovníkových datových typù.

```
property CountDictTypes: Integer; readonly;
```

Popis
Použijte vlastnost `CountDictTypes`, k získání poètu slovníkových datových typù definovaných uživatelem k aktuálnímu ER-modelu.

DictTypes

function DictTypes(index:integer):IDispatch;

Vrací rozhraní na slovníkový datový typ podle poøadí. Poøadí je od 0 do Model.DictTypesCount-1

Pøíklad:

```
function Main()
{
for (i=0; i<Model.CountDictTypes; i++)
   {
   d = Model.DictTypes(i);
   Log.Writeln(d.Name);
   }
}
```

Model.CountRelations

Model Pøíbuzné Pøíklad

Vrací poèet relací v modelu.

```
property CountRealtions: Integer; readonly;
```

Popis
Použíjte vlastnost `CountRelations`, k získání celkového poètu relací v aktuálním modelu.

Relations

function Relations(index:integer):IDispatch

Vrací rozhraní na relaci podle poøadí. Poøadí je od 0 do Model.CountRelations-1

Pøíklad:

Pøíklad k Relations, CountRelations

Pøíklad v JavaScriptu vypíše názvy všech relací aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountRelations; i++)
   {
   RelShips = Model.Relations(i);
   Log.Writeln(RelShips.Name);
   };
};
```

Model.CountIndexes

Model Pøíbuzné Pøíklad

Vrací poèet databázových indexù v modelu.

```
property CountIndexes: Integer; readonly;
```

Popis
Použijte vlastnost `CountIndexes`, k získání celkového poètu databázových indexù všech entit v aktuálním modelu.

Indexes

function Indexes(index:integer):IDispatch;

Vrací rozhraní na databázový index podle jeho poøadi. Poøadí je od 0 do Model.CountIndexes-1

pøíklad:

```
function Main()
{
for (i=0; i<Model.CountIndexes; i++)
   {
   ind = Model.Indexes(i);
   Log.Writeln(ind.Name);
   }
}
```

Pøíklad k Indexes, CountIndexes

Pøíklad v JavaScriptu vypíše názvy všech indexù všech entit aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountIndexes; i++)
   {
   Idx = Model.Indexes(i);
   Log.Writeln(Idx.Name);
   };
};
```

Model.CountIndexItems

Vrací poèet položek databázových indexù.

```
property CountIndexItems: Integer; readonly;
```

Popis
Použijte vlastnost `CountIndexItems`, k získání celkového poètu položek všech databázových indexù v aktuálním modelu.

IndexItems

function IndexItems(index:integer):IDispatch;

Vrací rozhraní na indexovou položku podle poøadí. Poøadí je od 0 do Model.CountIndexItems-1

pøíklad, ktery vypíše všechny indexy a jejich položky:

```
function Main()
{
for (i=0; i<Model.CountIndexes; i++)
   {
   ind = Model.Indexes(i);
   Log.Writeln(ind.Name);
   for (j=0; j<Model.CountIndexItems; j++)
      {
      ii = Model.IndexItems(j);
      if (ind.id == ii.IndexId)
            {
            Atr = Model.GetAttribute(ii.AttributeId);
            Log.Writeln(Atr.Name);
            }
      }
   }
}
```

Model.CountTriggers

Model Pøíbuzné Pøíklad

Vrací poèet uživatelských triggerù, procedur a pohledù.

```
property CountTriggers: Integer; readonly;
```

Popis
Použijte vlastnost `CountTriggers`, k získání celkového poètu všech uživatelských triggerù, procedur a pohledù aktuálního entitnì relaèního modelu.

Triggers

function Triggers(index:integer):IDispatch;

Vrací trigger podle poøadí. Poøadí je od 0 do Model.CountTriggers-1

Pøíklad, ktery vypíše všechny uživatelské triggry, procedury a pohledy i s názvy.

```
function Main()
{
for (i=0; i<Model.CountTriggers; i++)
   {
   Tr = Model.Triggers(i);
   Log.Writeln(Tr.Name);
   Log.Writeln(Tr.Source);
   }
}
```

Pøíklad k Triggers, CountTriggers

Pøíklad v JavaScriptu vypíše všechny názvý uživatelských triggerù, procedur a pohledù a jejich obsah, ke všem entitám aktuálního ER-model do okna Log.

```
function Main()
{
for (i=0; i<Model.CountIndexes; i++)
   {
   Trigger = Model.Triggers(i);
   Log.Writeln(Trigger.Name);
   Log.Writeln(Trigger.Source);
   };
};
```

CountModelTemplates

property CountModelTemplates:integer; readonly;

Poèet uživatelských šablon v modelu. Jsou tam zahrnuty i šablony patøící jednotlivým entitám.

ModelTemplates

function ModelTemplates(index:integer):IDispatch;

Vrací rozhraní na šablonu podle jejího poøadí. Poøadí je od 0 do Model.CountModelTemplates-1.

pøíklad:

```
function Main()
{
for (i=0; i<Model.CountModelTemplates; i++)
   {
   Templ = Model.ModelTemplates(i);
      if (Templ.Level == 5)
      {
      Ent = Model.GetEntity(Templ.EntityId);
      Log.Writeln('Entita '+Ent.Name);
      }
   Log.Writeln(Templ.Name);
   Log.Writeln(Templ.Source);
   }
}
```

Model.CountDataFlows

Vrací poèet alternativních klíèù v modelu.

```
property CountAlterKeys: Integer; readonly;
```

Popis
Použíjte vlastnost `CountAlterKeys`, k získání celkového poètu alternativních klíèù v aktuálním modelu.

AlterKeys

function AlterKeys(index:integer):IDispatch;

Vrací rozhraní na alternativní klíè podle poøadí. Poøadí je od 0 do Model.CountAlterKeys-1.

Model.CountAlterKeyItems

Vrací poèet položek alternativních klíèù v modelu.

```
property CountAlterKeyItems: Integer; readonly;
```

Popis
Použijte vlastnost `CountAlterKeyItems`, k získání celkového poètu položek všech alternativních klíèù v aktuálním modelu.

AlterKeyItems

function AlterKeyItems(index:integer):IDispatch;

Vrací rozhraní na položku alternativního klíèe podle poøadí. Poøadí je od 0 do Model.CountAlterKeyItems-1. Pracuje se s ní podobnì jako s IndexItems.

Model.CountUserRoles

Vrací poèet uživatelských rolí v modelu.

```
property CountUserRoles: Integer; readonly;
```

Popis
Použijte vlastnost `CountUserRoles`, k získání celkového poètu uživatelských rolí definovaných k aktuálnímu modelu.

UserRoles

function UserRoles(index:integer):IDispatch;

Vrací rozhraní na databázovou uživatelskou roli podle poøadí. Poøadí je od 0 do Model.CountUserRoles-1.

Model.CountUserPermiss

Model Pøíbuzné

Vrací poèet uživatelských práv v modelu.

```
property CountUserPermiss: Integer; readonly;
```

Popis
Použijte vlastnost `CountUserPermiss`, k získání celkového poètu definovaných databázových uživatelských práv v aktuálním ER modelu.

Poznámka
Uživatelské práva mùžou být, dle možností používané databáze, definované k entitì, proceduøe, triggeru anebo pohledu. Seznam uživatelských práv definuje vztah mezi uživatelem nebo uživatelskou rolí a daným objektem.

UserPermiss

function UserPermiss(index:integer):IDispatch;

Vrací rozhraní na databázové uživatelské právo podle poøadí. Poøadí je od 0 do Model.CountUserPermiss-1.

Model.CountUsers

Model Pøíbuzné

Vrací poèet databázových uživatelù v modelu.

```
property CountUsers: Integer; readonly;
```

Popis
Použijte vlastnost `CountUsers`, k získání celkového poètu definovaných databázových uživatelù v aktuálním ER modelu.

Users

function Users(index:integer):IDispatch;

Vrací rozhraní na databázového uživatele podle poøadí. Poøadí je od 0 do Model.CountUsers-1.

Model.CountUserRoleUsers

Model Pøíbuzné

Vrací poèet definovaných vztahù uživatelù k uživatelským rolím v modelu.

```
property CountUserRoleUsers: Integer; readonly;
```

Popis
Použijte vlastnost `CountUserRoleUsers`, k získání celkového poètu definovaných vztahù databázových uživatelù k uživatelským rolím v aktuálním ER modelu.

Poznámka
Každému uživateli lze pøiøadit libovolný poèet rolí, které mu umožòují získat rùzné práva k urèitým objektùm. Pokud nadefinujete napøíklad ve vašem modelu ètyøi uživatelé a každému z nich pøiøadíte dvì role, tak Vám vrátí metoda `CountUserRoleUsers` hodnotu osm.

UserRoleUsers

function UserRoleUsers(index:integer):IDispatch;

Vrací rozhraní na UserRoleUsers podle poøadí. Poøadí je od 0 do Model.CountUserRoleUsers.

Model.Created

Model Pøíbuzné

Vrací datum vytvoøení modelu.

```
property Created: TDataTime; readonly;
```

Popis
Použíjte vlastnost `Created`, k získání data vytvoøení modelu. V modelu naleznete tuto informaci napøíklad na razítku. Návratová hodnota je typu Double.

Model.Modified

Model Pøíbuzné

Vrací datum poslední modifikace modelu.

```
property Modified: TDataTime; readonly;
```

Popis
Použijte vlastnost `Modified`, k získání data poslední modifikace modelu. V modelu naleznete tuto informaci napøíklad na razítku. Datum se mìní automaticky pøi uložení modelu. Návratová hodnota je typu Double.

Model.CountProcesses

Model Pøíbuzné

Vrací poèet procesù v modelu.

```
property CountProcesses: Integer; readonly;
```

Popis
Použijte vlastnost `CountProcesses`, k získání poètu všech procesù v aktuálním modelu. Do výsledkù jsou zahrnuté procesy všech úrovní.

Process

function Process(index:integer):IDispatch;
Vrací rozhraní na proces v DFD podle poøadí. Poøadí je od 0 do Model.CountProcess-1.

Model.CountDataStores

Vrací poèet datastorù v modelu.

```
property CountDataStores: Integer; readonly;
```

Popis
Použijte vlastnost `CountDataStores`, k získání poètu všech datastorù v aktuálním modelu. Do výsledkù jsou zahrnuté datastory procesù všech úrovní.

Poznámka
Pokud se jeden a tentýž datastore vyskytuje v nìkolika úrovních modelu, tak se každý jednotlivý vnoøený datastore zapoèítá do výsledkù. Tzn. pokud máte v modelu jeden datastore, který je až ve tøetí úrovní procesu, tak vám metoda `CountDataStores` jako výsledek vrátí hodnotu tøi.

DataStores

function DataStored(index:integer):IDispatch;

Vrací rozhraní na datastore v DFD podle poøadí. Poøadí je od 0 do Model.CountDataStores-1.

Model.CountTerminators

Model Pøíbuzné

Vrací poèet terminátorù v modelu.

```
property CountTerminators: Integer; readonly;
```

Popis
Použijte vlastnost `CountTerminators`, k získání poètu všech terminátoru v aktuálním modelu. Do výsledkù jsou zahrnuté terminátory z procesù všech úrovní.

Poznámka
Pokud se jeden a tentýž terminátor vyskytuje v nìkolika úrovních modelu, tak se každý jednotlivý vnoøený terminátor zapoèítá do výsledkù. Tzn. pokud máte v modelu jeden terminátor, který je až ve tøetí úrovní procesu, tak vám metoda `CountTerminators` jako výsledek vrátí hodnotu tøi.

Terminators

function Terminators(index:integer):IDispatch;

Vrací rozhraní na terminátor z DFD podle poøadí. Poøadí je od 0 do Model.CountTerminators-1.

Model.CountDataFlows

Model Pøíbuzné

Vrací poèet datových tokù v modelu.

```
property CountDataFlows: Integer; readonly;
```

Popis
Použijte vlastnost `CountDataFlows`, k získání poètu všech datových tokù v aktuálním modelu. Do výsledkù jsou zahrnuté datové toky procesù všech úrovní.

Poznámka
Pokud se jeden a tentýž datový tok vyskytuje v nìkolika úrovních modelu, tak se každý jednotlivý vnoøený datový tok zapoèítá do výsledkù. Tzn. pokud máte v modelu jeden datový tok, který je až ve tøetí úrovní procesu, tak vám metoda `CountDataFlows` jako výsledek vrátí hodnotu tøi.

DataFlows

function DataFlows(index:integer):IDispatch; readonly;

Vrací rozhraní na datový tok v DFD podle poøadí. Poøadí je od 0 do Model.CountDataFlows-1.

ReadOnly

property ReadOnly:boolean;

Urèuje, zda je zakázáno zapisovat do klíèových údajù v modelu pøes TLB knihovnu. Implicitní hodnota je true.

Model.CountSystemTemplates

Vrací poèet systémových šablon.

```
property CountSystemTemplates: Integer; readonly;
```

Popis
Použijte vlastnost `CountSystemTemplates`, k získání poètu systémových šablon, které jsou nezávislé na modelu nebo databází v systému k dispozici.

Poznámka
Do seznamu se poèítají systémové šablony ze všech dostupných kategorií (HTML,RTF,...). Tyto šablony jsou urèené pouze ke ètení.

SystemTemplates

Model.CountDatabaseTemplates

Vrací poèet databázových šablon.

```
property CountDatabseTemplates: Integer; readonly;
```

Popis
Použijte vlastnost `CountDatabaseTemplates`, k získání poètu systémových databázových šablon pro aktuální databází.

Poznámka
Do seznamu se poèítají systémové databázové šablony ze všech dostupných kategorií (HTML,RTF,...). Tyto šablony jsou pouze ke ètení.

Model.CountUDFDatabaseTemplates

Hierarchie Model Pøíbuzné

Vrací poèet uživatelských databázových šablon.

```
property CountUDFDatabseTemplates: Integer; readonly;
```

Popis
Použijte vlastnost `CountUDFDatabaseTemplates`, k získání poètu uživatelsých systémových databázových šablon pro aktuální databází.

Poznámka
Do seznamu se poèítají uživatelské systémové databázové šablony ze všech dostupných kategorií (HTML,RTF,...).

Model.CountUDFSystemTemplates

Hierarchie Model Pøíbuzné

Vrací poèet uživatelských systémových šablon.

```
property CountUDFSystemTemplates: Integer; readonly;
```

Popis
Použijte vlastnost `CountUDFSystemTemplates`, k získání poètu uživatelských systémových šablon, které jsou nezávislé na modelu nebo databází v systému k dispozici.

Poznámka
Do seznamu se poèítají uživatelské systémové šablony ze všech dostupných kategorií (HTML,RTF,...).

Model.CountModelTemplates

Vrací poèet uživatelských šablon k modelu.

```
property CountModelTemplates: Integer; readonly;
```

Popis
Použijte vlastnost `CountModelTemplates`, k získání poètu uživatelských šablon, které jsou definované pouze k aktuálnímu modelu.

Poznámka
Do seznamu se poèítají šablony ze všech dostupných kategorií (HTML,RTF,...). Návratová hodnota obsahuje všechny uživatelské šablony k modelu, tzn. i šablony definované k entitì.

Kombinace MS Scriptu a šablon

V CASE Studio mùžete s úspìchem kombinovat šablony fungující podle staré koncepce se šablonami napsanými ve skriptovacích jazycích. Mùžete z klasické šablony volat proceduru nebo funkci umístìnou v šalonì napsané ve skriptovacím jazyku. **Naopak ne**.
Používají se k tomu procedury:

```
Script(TemplateName);
```

V šablonì TemplateName spustí proceduru Main.

```
ScriptProc(TemplateName,ProcName,[par1,par2...]);
```

V šablonì TemplateName spustí procedure ProcName s parametry par1,par2 ....

Pøíklad:

Šablona CreateDatabase

```
@ForTable("","",ScriptProc(scr1,proc1,Table.Id),"","")
```

Projde všechny tabulky v modelu, a pro každou vyvolá proceduru `proc1`, jako parametr použije ID entity.

Pozn: ID použije, protože šablony neumí pracovat s promìnnými typu rozhraní.

Šablona scr1

```
function proc1(x)
{
s = '';
e = Model.GetEntity(x);                    // Podle ID získá rozhraní na
entitu.
s = 'Create table ' + e.TableName + '(';
c = e.CountAttributes;                     // do c poèet atributù v entitì
                                  // kvùli rychlosti
for (i=0; i < c; i++)                      // prochází všechny atributy v
entitì
   {
   a = e.Attributes(i);
   s = s + '\n' + a.ColName + ' ' + a.SQLDataType;
   if (i+1 != c)                           // pokud to není poslední atribut
      {
      s = s +',';                          // pøidá za nìho èárku
      }
   }
s = s + '\n)\n\n';
return (s);
}
```

Výsledek funkce `return (s)` se pøipojí na konec výstupního textového souboru.

Optimalizace

Nìkolik zásad pro lepší práci MS Scriptu. Pøíklady oznaèené jako špatné jsou plnì funkèní, ale neefektivní.

1. Pokud napø. v entitì zjiš□ujete poèet atributù, a tuto hodnotu použijete vícekrát, uložte si ji do promìnné.

Správnì

```
c = ent.CountAttributes;
for (i=0; i<c; i++)
   {
   if (i+1 = c)
      {
      }
   }
```

Špatnì

```
for (i=0; i<ent.CountAttributes; i++)
   {
   if (i+1 = ent.CountAttributes)
      {
      }
   }
```

Vysvìtlení: Entita nevlastní atributy, atributy vlastní model. Pokud se ve scriptu zeptáte entity, kolik má atributù, entita se zeptá modelu, kolik je atributù s EntityID = Ent.Id. Což je samozøejme velmi neefektivní. Toto platí pro všechny seznamy v entitì, indexu a alternativním klíèi.

2. Pokud napø. pøistupujete k rozhraní atributu pøes entitu, a rozhraní téhož atributu použijete vícekrát, uložte si rozhraní do promìnné.

Správnì

```
a = ent.Attributes(i);
s = a.ColName + ' ' + a.SQLDataType;
```

Špatnì

```
s = ent.Attributes(i).ColName + ' ' + ent.Attributes(i).SQLDataType;
```

Vysvìtlení: Podobnì jako u bodu 1, v pøípadì požadavku na ent.Attributes(i) se entita dotazuje modelu, a ten opìt prochází všechny atributy, a vrací rozhraní na i-ty atribut s EntityID = Ent.Id, což je opìt nefektivní.

3. Pokud opakovanì voláme funkci z jiné šablony (napø. ve smyèce), uložíme si rozhraní na jinou šablonu do promìnné.

Správnì

```
t = Scripting.Modul1;
for (i=0; i<1000; i++)
   {
   t.Show();
   (
```

Špatnì

```
for (i=0; i<1000; i++)
   {
   Scripting.Modul1.Show();
   }
```

Vysvìtlení: Pøi uložení rozhraní šablony do promìnné se šablona nemusí opakovanì vyhledávat v seznamu šablon. (Ve skuteènosti se vyhledává jenom pøi prvním použití, pak se uloží do poolu a pøi dalších použítích se vyhledává tam, ale i tak je rychlejsí použít rozhraní uložené v promìnné).

Vyvolání procedury nebo funkce z jiné šablony

Pokud chcete z jedné šablony se skriptovacím jazykem zavolat funkci nebo proceduru z jiné šablony, použijte k tomu

Scripting.<TemplateName>.<ProcName(par1,par2,par3,...)>

pø:
Pokud chcete ze šablony MyTools zavolat funkce DPH, napište to takto:

```
s = Scripting.MyTools.DPH(x,22);
```

Pøi volání funkcí a procedur v šablonách se nemusíte starat, ve kterém skriptovacím jazyce je volaná šablona napsaná, systém se sám postará o správnou funkci. Proto mùžete napø. ze šablony napsané v JScriptu volat funkce umístìné šablonì napsané ve VBScriptu.

Case sensitive

Citlivost na velká a malá písmena závisí na možnostech instalovaných skriptovacích jazykù. Napø. JScript je case sensitive, VBScript je case insensitive. Výjimkou jsou názvy metod a properties v rozhraních, ty jsou vždy case insensitive.

TerminatorsSkriptovací jazyky od tøetích firem

MS Scripting umožòuje kromì JScriptu a VBScriptu používat i skriptovací jazyky od tøetích firem.

Pø.

Pokud chcete v CASE Studiu používat napø. Perl, musíte

1. Stáhnìte si z http://www.activestate.com podporu Perlu pro MS Windows, a nainstalujte ji.

2. Najdìte na disku soubor `ScriptList.ini` (je v adresáøí bin), a opravte ho, tøeba takto:

```
 [ScriptList]
;1 = JScript
;2 = VBScript
3 = PerlScript

101 = OtherScript
```

První 2 øádky nechte zakomentáøované, JScript a VBScript jsou definované pøímo v systému. **Perl od ActiveState má vyhrazeno èíslo 3**. Pokud chcete použít jiný skriptovací jazyk, použíjte jako identifikátor èíslo vìtší než 100. Èísla menší než 100 smìjí použít jen autoøi programu.