

Kurs programowania aplikacji bazodanowych

Wykład 8

Paweł Rajba

Instytut Informatyki
Uniwersytet Wrocławski

Plan wykładu

- Wprowadzenie do generowania kodu
- Podstawy CodeSmith
- Dostęp do schematu bazy danych
- Aplikacje w CodeSmith

Wprowadzenie do generowania kodu

- Automatyczne generowanie kodu znacznie przyspiesza tworzenie aplikacji
 - szczególnie, jeśli np. klasy są tworzone wg jednego schematu
- Kod może być generowany na podstawie
 - istniejącej bazy danych
 - schematu UML
- Tworzenie schematu bazy danych na podstawie struktury klas zapewnia większość systemów ORM
- W centrum naszych zainteresowań będzie program CodeSmith
 - strona domowa produktu: <http://www.codesmithtools.com/>
- Przegląd innych produktów (nie tylko dla .NET) jest na stronie pod adresem <http://www.codegeneration.net/>

Podstawy CodeSmith

W skład aplikacji wchodzi

- Szablony
- Pliki z kodem pomocniczym (*behind code*)

Jak to działa?

- Szablon jest kompilowany, a następnie wykonywany
- Wyniki są generowane na „stdout” lub do pliku

Uwaga:

- W CodeSmith używamy języków z .NET (C#, VB lub JScript), natomiast możemy generować kod dowolnego języka programowania

Podstawy CodeSmith

- Budowa szablonu jest podobna do strony w PHP lub JSP
 - Są fragmenty statyczne, które zostaną przepisane do pliku wynikowego
 - Mamy też fragmenty dynamiczne, które zostaną wykonane i do pliku wynikowego zostanie przekierowany wynik ich działania
 - Struktura jest następująca:
 - Deklaracja szablonu `<%@ CodeTemplate ...%>`
 - Deklaracje właściwości `<%@ Property ...%>`
 - Możemy je ustawiać w osobnym widoku w CodeSmith Studio lub z poziomu kodu (np. w innym skrypcie)
 - Za ich pomocą możemy sterować wykonaniem skryptu
- Są czymś w rodzaju parametrów skryptu

Podstawy CodeSmith

- Struktura c.d.
 - Deklaracje *Assembly* i import przestrzeni nazw, np.
 - `<%@ Assembly Name="System.Data" %>`
 - `<%@ Import Namespace="System.Data" %>`
 - Treść dynamiczną umieszczamy:
 - `<%= To zostanie przepisane do wyniku %>`
 - `<% Tutaj umieszczamy instrukcje %>`
 - `<script runat="template">`
Tutaj deklaracje metod
`</script>`
 - Komentarze:
 - `<%-- Komentarz --%>`

- FirstExample
 - Przy okazji przeglądamy co jest dostępne w
Edit → Insert Content
- HtmlExample

Dostęp do schematu bazy danych

- Informacje o zdefiniowanych bazach danych przechowuje *Schema Explorer*
- Z jego poziomu możemy zarządzać połączeniami
 - Dodajemy nowe połączenie do bazy KursPBD
 - *ConnectionString* możemy wprowadzić ręcznie lub skorzystać z kreatora
 - Patrzymy na pozostałe funkcje zarządcy połączeniami

Dostęp do schematu bazy danych

Główne obiekty reprezentujące obiekty w bazie danych

- DatabaseSchema
- TableSchema
- ColumnSchema
- TableKeySchema
- IndexSchema

Dostęp do schematu bazy danych

DatabaseSchema (istotniejsze właściwości)

- `ConnectionString`
- `Name`
- `Provider`
- `Tables, Views`

IndexSchema (istotniejsze właściwości)

- `Database, Table`
- `IsPrimaryKey, IsUnique`
- `MemberColumns`
- `Name`

ColumnSchema (istotniejsze właściwości)

- AllowDBNull
- Database, Table
- DataType, NativeType, SystemType
- IsForeignKeyMember, IsPrimaryKeyMember, IsUnique
- Name
- Precision
- Scale
- Size

TableKeySchema (istotniejsze właściwości)

- Database
- ForeignKeyMemberColumns
- ForeignKeyTable
- Name
- PrimaryKey
- PrimaryKeyMemberColumns
- PrimaryKeyTable

Dostęp do schematu bazy danych

TableSchema (istotniejsze właściwości)

- Database
- Columns, NonKeyColumns
- NonForeignKeyColumns, NonPrimaryKeyColumns
- ForeignKeyColumns, ForeignKeys
- Indexes
- Keys
- Name
- HasPrimaryKey, PrimaryKey, PrimaryKeys

Przykłady

- SimpleGenerator
- RelationGenerator

Aplikacje w CodeSmith

W CodeSmith możemy:

- Dodawać behind code dla szablonów
- Wywoływać jedne szablony w innych
- Tworzyć projekty

Przykłady

- CodeBehind
- MasterTemplate
- XpoProject