Jarosław Kuchta

Projekt ModelGen

data pierwszego opracowania: 27.10.2023

data ostatniej modyfikacji: 02.01.2024

# Konfiguracja i opcje przetwarzania

W programie ModelGen wydzielono komponent ModelGenDataConfig odpowiedzialny za konfigurowanie modelu danych aplikacji. Konfiguracja polega na określeniu, które przestrzenie nazw z bibliotek źródłowych, typy i ich właściwości są akceptowane do przetwarzania i w jaki sposób są przetwarzane. Oprócz tego w komponencie głównym ModelGen zdefiniowano opcje przetwarzania, czyli np. nazwy, rodzaje i położenie wynikowych elementów programowych (rozwiązania, projektu, plików).

Moduł główny aplikacji (ModelGenApp) umożliwia edycję tych ustawień przez użytkownika.

## Konfiguracja modelu danych

W wymaganiach systemowych podano, że konfiguracja jest oparta na regułach. Są dwa rodzaje reguł: reguły kwalifikacji oraz reguły transformacji.

Reguły kwalifikacji to zbiór list zawierających nazwy przetwarzanych elementów zaliczające je do określonej kategorii. Nie wszystkie listy zawierają reguły kwalifikacji. Niektóre są potrzebne do definiowania nowych nazw elementów tworzonych w procesie przetwarzania.

Reguły transformacji to zbiór list par typu (nazwa źródłowa, nazwa docelowa) przypisujących elementy jednego rodzaju do elementów innego rodzaju.

Istnieje kilka grup reguł (i definicji):

* dotyczące bibliotek DLL:
  + SourceLibraries – lista nazw bibliotek źródłowych, tj. tych bibliotek DLL, których elementy są wczytywane w procesie przetwarzania. Biblioteki źródłowe muszą być dołączone do programu ModelGen.
  + TargetLibraries – lista definiująca nazwy bibliotek docelowych, tj. tych bibliotek DLL, których kod jest generowany w procesie przetwarzania, a na końcu są kompilowane.
  + UsedLibraries – lista nazw tych bibliotek DLL, które są wykorzystywane w procesie przetwarzania (np. definiują typy podstawowe). Tu kwalifikują się też biblioteki systemowe. Wszystkie one muszą być dołączone do programu ModelGen.
* dotyczące przestrzeni nazw:
  + SourceNamespaces – lista źródłowych przestrzeni nazw, tj. przestrzeni deklarowanych w bibliotekach źródłowych. Domyślnie wszystkie przestrzenie nazw z bibliotek źródłowych są przestrzeniami źródłowymi.
  + TargetNamespaces – lista definiująca docelowych przestrzeni nazw, tj. przestrzeni generowanych w bibliotekach wynikowych. Domyślna docelowa przestrzeń nazw jest określona w opcjach programu ModelGen. Jeśli jednak konstruktor będzie chciał podzielić tę domyślną przestrzeń nazw na podprzestrzenie, to będzie musiał je zdefiniować samodzielnie.
  + UsedNamespaces – lista tych przestrzeni nazw, które są wykorzystywane w procesie przetwarzania. Domyślnie zaliczają się tu wszystkie przestrzenie nazw, które są zdefiniowane w innych bibliotekach.
  + TargetNamespacesToLibraries – lista odwzorowań docelowych przestrzeni nazw w docelowe biblioteki.
  + AcceptedNamespaces – lista akceptowanych źródłowych przestrzeni nazw. Akceptowana przestrzeń nazw to taka, której wszystkie typy są zakwalifikowane do dalszego przetwarzania (z wyjątkiem typów określonych jako odrzucone).
  + RejectedNamespaces – lista odrzuconych źródłowych przestrzeni nazw. Odrzucona przestrzeń nazw to taka, której wszystkie typy nie są dalej przetwarzane (z wyjątkiem typów określonych jako akceptowane).
  + NamespacesShortcuts – lista wzajemnie jednoznacznych odwzorowań przestrzeni nazw i ich skrótów.
  + NamespacesTranslations – lista odwzorowań źródłowych przestrzeni nazw na docelowe.
* dotyczące typów:
  + AcceptedTypes – lista nazw typów indywidualnie akceptowanych. Są to takie typy, które są zakwalifikowane do dalszego przetwarzania, nawet jeśli ich przestrzeń nazw jest odrzucona.
  + RejectedTypes – lista nazw typów indywidualnie odrzuconych. Są to takie typy, które nie są dalej przetwarzane, nawet jeśli ich przestrzeń nazw jest akceptowana.
  + TypesTranslations – lista odwzorowań nazw typów akceptowanych w docelowe z uwzględnieniem źródłowych i docelowych przestrzeni nazw.
  + TypeConverters – lista przypisująca typy konwerterów do typów przetwarzanych wg ich nazw z uwzględnieniem źródłowych i docelowych przestrzeni nazw.
* dotyczące właściwości:
  + RejectedProperties – lista nazw właściwości indywidualnie odrzuconych. Są to takie właściwości, które nie są dalej przetwarzane, nawet jeśli ich typy są akceptowane. Wszystkie pozostałe właściwości typów akceptowanych są akceptowane.
  + PropertiesTranslations – lista odwzorowań nazw właściwości akceptowanych w docelowe z uwzględnieniem źródłowych i docelowych typów i przestrzeni nazw.
  + PropertyConverters – lista przypisująca typy konwerterów do właściwości przetwarzanych wg ich nazw z uwzględnieniem źródłowych i docelowych typów i przestrzeni nazw.
* dotyczące wartości wyliczeniowych:
  + RejectedEnums – lista nazw wartości wyliczeniowych indywidualnie odrzuconych. Są to takie wartości, które nie są dalej przetwarzane, nawet jeśli ich typy są akceptowane. Wszystkie pozostałe wartości wyliczeniowe typów akceptowanych są akceptowane.
  + EnumTranslations – lista odwzorowań nazw wartości wyliczeniowych akceptowanych w docelowe z uwzględnieniem źródłowych i docelowych typów i przestrzeni nazw.
  + TargetEnumValues – lista odwzorowań nazw wartości wyliczeniowych docelowych w wartości całkowite z uwzględnieniem docelowych typów i przestrzeni nazw.

### Użycie symboli wieloznacznych

Nazwy w regułach kwalifikacji mogą zawierać symbole wieloznaczne – gwiazdkę (‘\*’) dla oznaczenia dowolnej sekwencji znaków i znak zapytania (‘?’) dla oznaczenia dowolnego znaku.

Kropka (‘.’) jest używana do tworzenia nazw złożonych – typów poprzedzanych nazwą przestrzeni nazw, właściwości i wartości wyliczanych poprzedzanych nazwami typów poprzedzanych nazwami przestrzeni nazw.

Można też stosować ograniczony podzbiór wyrażeń regularnych .NET. Ponieważ są dopasowywane tylko nazwy, czyli ciągi znaków składające się z liter, cyfr dziesiętnych i znaku podkreślenia, więc nie potrzeba stosowania składni unikowej (escape). Występowanie wyrażeń regularnych jest rozpoznawane przez występowanie znaków specjalnych: ‘\*’, ‘?’, ‘[’, ‘]’, ‘^’, ‘#’, ‘(’, ‘)’. Gwiazdka i znak zapytania po znakach specjalnych ma inne znaczenie. Zestawienie przedstawia tab. 1.

Tab. . Zestawienie wyrażeń regularnych we wzorcach nazw wieloznacznych

|  |  |  |  |
| --- | --- | --- | --- |
| **Podwyrażenie** | Znaczenie | Przykład | Pasuje do nazw |
| [chs] | Pasuje do pojedynczego znaku wymienionego między nawiasami kwadratowymi. Uwzględniana wielkość liter. | Gr[ae]y | Gray  Grey |
| [^chs] | Pasuje do pojedynczego znaku **niewymienionego** między nawiasami kwadratowymi. Uwzględniana wielkość liter. | \*[^\_] | wszystkich nazw z wyjątkiem kończących się podkreśleniem |
| [first-last] | Pasuje do pojedynczego znaku w zakresie od znaku pierwszego do ostatniego. Uwzględniana wielkość liter. | [A-Z]\* | Document  ale nie  document |
| [^first-last] | Pasuje do pojedynczego znaku **spoza** zakresu od znaku pierwszego do ostatniego. Uwzględniana wielkość liter. | [^A-Z]\* | document  ale nie  Document |
| # | Pasuje do dowolnej cyfry dziesiętnej w miejscu występowania. Zastępuje  „[0-9]” | M## | M02  M03 |
| #\* | Pasuje do ciągłej sekwencji cyfr dziesiętnych. Zastępuje „[0-9]\*” | Office#\* | Office2020  Office2021 |
| [chs]? | Dopuszcza opcjonalne wystąpienie znaku wymienionego między nawiasami kwadratowymi. Uwzględniana wielkość liter. | Chart[s]? | Chart  Charts |
| ( ) | Reprezentuje złożone podwyrażenie regularne | Event(Class)? | Event  EventClass |

W nazwach na listach definiujących nie można używać symboli wieloznacznych ani wyrażeń regularnych. Można natomiast używać kropek jako elementu składowego nazwy złożonej.

W parach odwzorowań w regułach transformacji można stosować nazwy wieloznaczne zarówno po stronie źródłowej, gdzie symbol gwiazdki pełni rolę symbolu wieloznacznego, jak i docelowej, gdzie pełni rolę symbolu zastępczego (w to miejsce wstawiany jest łańcuch znaków dopasowanych do symbolu zastępczego. Dla przykładu transformacja

(„DocumentFormat.OpenXml.\*”, „DocumentModel.\*”)

oznacza odwzorowanie zarówno:

(„DocumentFormat.OpenXml.Math”, „DocumentModel.Math”)

jak i:

(„DocumentFormat.OpenXml.Drawing”, „DocumentModel.Drawing”)

### Unikatowość nazw i jednoznaczność odwzorowań

Wszystkie nazwy na jednej liście reguł kwalifikujących muszą być unikatowe.

Jeśli stosowane są reguły zawierające symbole wieloznaczne i jednoznaczne, to stosuje się priorytetyzację dopasowania. Najpierw próbuje się dopasować konkretną nazwę do reguł niezawierających symbolu wieloznacznego. W razie niepowodzenia próbuje się dopasować konkretną nazwę do reguł zawierających jeden symbol wieloznaczny.