РАЗРАБОТКА КОМПОНЕНТА TREEVIEWMANAGER ДЛЯ СИСТЕМЫ ПРОТОКОЛИРОВАНИЯ СОБЫТИЙ

Плахин Д.В.
Научный руководитель: С.Г. Цапко
Томский политехнический университет, г.Томск, пр. Ленина, 30
pdv-mail@mail.ru

Введение

DataAnalyzer – проект, целью которого является упростить протоколирование событий. В программе Data Analyzer пользователь описывает для своего текущего проекта протоколы, содержащие события с определённым числом полей, после чего по запросу пользователя программа генерирует динамическую библиотеку logLibrary.dll для C# или C++, содержащую класс описываемого проекта. Класс предназначен для взаимодействия с удалённым сервисом Data Analyzer Service. Этот сервис взаимодействует с единой базой данных, сохраняя информацию о произошедших событиях и предоставляя эту информацию по запросу. Библиотека logLibrary.dll может быть разной для разных программ, однако суть её работы одна и та же, а структура схожа для всех таких библиотек. Благодаря этому свойству библиотеки база данных остаётся одной и той же для всех разрабатываемых приложений.

После создания библиотеки её можно подключить к создаваемому проекту, и после этого в код программы можно добавлять функции протоколирования событий, обращаясь к соответствующему классу. Все данные, полученные от программ, использующих библиотеку, будут поступать на сервер и впоследствии могут быть просмотрены. Функция просмотра на данный момент отсутствует в проекте и в будущем будет реализована автором данной статьи.

Разработка компонента

На данном этапе работы над проектом Data Analyzer основной задачей было переделать редактор классов. Класс проекта содержит протоколы. Каждый протокол содержит события, а каждое событие – поля. Ранее создание класса проекта предусматривало заполнение трёх форм соответственно – списка протоколов, списка событий (для каждого протокола вызывается и заполняется отдельно) и списка полей (также для каждого события вызывается и заполняется отдельно). При этом на первом уровне можно было редактировать только сам список - добавлять и удалять протоколы. На втором можно было редактировать информацию об этом протоколе (описание и другие поля), а также редактировать список событий. На третьем уровне можно было редактировать информацию о событии, список полей события и свойства полей.

Таким образом, на последнем уровне был список полей и связанный с ним компонент PropertyGrid [1], в котором отображались свойства каждого поля (имя, формат отображения и т.д.). Для упрощения работы с редактором нужно было сделать то же самое, только в одной форме для всех уровней. Это предполагало создание компонента, который способен заполнять стандартный компонент TreeView, представляющий собой дерево элементов. Верхний уровень дерева – протоколы, средний – события, нижний – поля событий. При щелчке на элемент дерева в зависимости от уровня меняется таблица свойств и свойства элемента (протокола, события или поля) отображаются в PropertyGrid. Результат разработки представлен на рисунке 1.

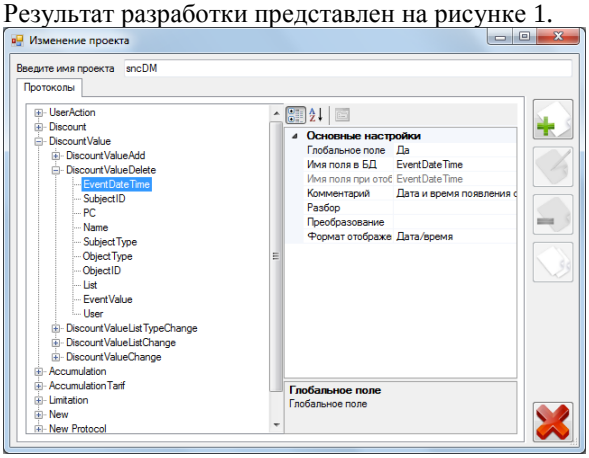

Рис. 1. Полученное окно Data Analyzer

Для заполнения PropertyGrid уже существует разработанный в организации компонент PropertyListViewManager. Компонент получает на вход таблицу полей для отображения. Каждая строка соответствует определённому свойству элемента. Свойства описываются такими параметрами (столбцы таблицы) как Name - отображаемое имя свойства, Value - значение свойства, Type - тип значения, Params - параметры отображения свойства, (например, свойство Type может иметь тип перечисления, и тогда отображаться будет выпадающий список, задаваемый полем Params), Visible - видимость, ReadOnly - можно ли изменять значение, и некоторые другие. Таким образом, для каждого уровня TreeView необходимо было создать только одну такую таблицу (так как свойства всех протоколов/событий/полей соответственно одинаковы, меняются лишь их значения) и правильно в нужный момент её изменять. Изменение предполагает редактирование полей Name, Value и, в некоторых случаях, Params (составлять значения для выпадающего списка, и иногда изменять, в зависимости от значения свойства «Глобальное поле» на уровне полей).

Описываемый в данной статье разработанный компонент TreeViewManager содержит такие таблицы для каждого уровня, а также таблицы со значениями свойств для каждого элемента уровня. Другими словами, столбцы таблицы уровня содержат такие поля как ID (идентификатор элемента), ParentID (идентификатор

родительского элемента), Name (отображаемое имя), а также столбцы, аналогичные строкам в таблицах уровней, для хранения значений свойств каждого элемента. Имена отображаемых в PropertyGrid свойств являются именами этих столбцов и указываются в таблице свойств в столбце StringKey.

В программе Data Analyzer, использующей разработанный компонент, данные загружаются из файла *.db SQL-запросами [2], и записываются по закрытию формы – изменённые поля обновляются, новые – вставляются, неизменные поля программа не трогает, удалённые – удаляются из базы, если они там были. Полученные данные отображаются в привязанном к компоненту TreeView. При выборе элемента происходит событие, передающее в обрабатывающую программу заполненную таблицу свойств. Это сделано, чтобы не связывать жёстко TreeViewManager и PropertyListViewManager. В программе событие ItemSelected обрабатывается и в PropertyListViewManager передаётся нужная таблица, отображающая свойства выбранного элемента.

Все изменения, сделанные в PropertyGrid через PropertyViewManager передаются при вызываемом событии изменения элемента в TreeViewManager. Таким образом, все данные своевременно изменяются, включая имя элемента (протокола, события или поля), которое также изменяется при отображении в TreeView.

Компонент позволяет не переключаться между формами, ведя всю разработку в одном окне. Для удобства работы также создан компонент MultiSelectTreeView, унаследованный от стандартного TreeView. Стандартный компонент не позволяет выбирать сразу несколько элементов, что не вполне удобно для поставленной задачи. Переработанный TreeView позволяет выбрать несколько элементов на одном уровне и переместить, скопировать или удалить сразу несколько объектов. Также компонент предоставляет удобное для выделения и перемещения по дереву управление с клавиатуры. Из-за описанных преимуществ TreeViewManager работает не со стандартным TreeView, а с разработанным MultiSelectTreeView.

Разработанный компонент предоставляет функции удаления, добавления или перемещения узла дерева для любого уровня, производя эти операции не только над построенным деревом, но и над таблицами уровней, содержащими его структуру. При добавлении или перемещении производится проверка соответствия уровня элемента новому месту его расположения. Благодаря этому, в приложении Data Analyzer реализована возможность перемещения и копирования узлов дерева при помощи мыши.

Стоит отметить, что в проекте DataAnalyzer компонент применяется также для редактирования представлений. Описание протокола содержит большой объём событий. С помощью создания представлений можно группировать нужные поля событий необходимым образом и выводить в удобном для пользователя виде при запросе из базы данных. Из пользовательской программы разработчик сможет вызывать генерацию отчёта для данного представления и посмотреть события, удовлетворяющие указанным условиям.

При редактировании представлений дерево содержит четыре уровня – уровень протоколов, уровень представлений, уровень событий и уровень полей. Форма редактирования аналогична представленной на рисунке 1, но она не позволяет редактировать протоколы и события. Существует возможность лишь указать формат отображения конкретного поля в представлении, а также редактировать свойства самих представлений. При удалении полей или событий они удаляются только из представления, но не из протокола. Добавить же поле или событие можно только из списка уже созданных полей и событий соответственно. Таким образом, форма не нуждается во всех возможностях разработанного компонента, однако способность отображать всю структуру протоколов, включая их представления, свойства и поля, является необходимой для удобного редактирования.

Для использования компонента следует перетащить его на форму разрабатываемого приложения. Далее следует передать компоненту нужные таблицы уровней и указать MultySelectTreeView, в котором будут отображаться данные. Затем следует указать столбцы ID и ParentID. Стоит отметить, что во всех таблицах они должны быть одинаковы.

После этого, если компонент используется в связке с PropertyListViewManager, следует указать PropertyGrid для отображения данных и задать таблицы свойств для каждого уровня. Когда завершены эти операции, нужно реализовать событие ItemSelected. Здесь нужно указать таблицу свойств для компонента PropertyListViewManager и вызвать операцию его обновления.

Заключение

Благодаря возможности отображать множество различных данных в одном окне разработанный компонент TreeViewManager позволяет значительно упростить и ускорить многие выполняемые задачи. В связке с компонентом MultiSelectTreeView, включённым в библиотеку TreeViewManager, он предоставляет гибкий инструмент по выполнению операций над несколькими элементами одновременно. Разработанный компонент будет применяться также в других проектах организации, дополняясь различными функциями и изменяясь в соответствии с нуждами разработчиков.

Список литературы

1. Liberty J. Программирование на C# - Символ-Плюс, 2003. - 690с.

2. Кузнецов М, Симдянов И. MySQL 5. – БХВПетербург, 2010. - 1007с.