Лабораторная работа 1. Вариант 2

Механизм привязки данных в Windows Presentation Foundation

В лабораторной работе надо создать пользовательский интерфейс приложения для работы с коллекцией **V2MainCollection** (из лабораторных работ прошлого семестра). Пользовательский интерфейс приложения дает возможность добавлять в коллекцию новые элементы, удалять элементы, сохранять коллекцию в файле, загружать коллекцию из файла.

В среде VisualStudio создать решение (solution) с двумя проектами:

- тип одного проекта – библиотека классов (class library), в которой находятся все типы из лабораторных работ прошлого семестра − **DataItem, Grid, V2Data, V2DataOnGrid, V2DataCollection, V2MainCollection;**
- тип второго проекта – приложение **Windows Presentation Foundation (WPF).**

В класс **V2MainCollection** надо добавить

- реализацию интерфейса **System.Collections.Specialized.INotifyCollectionChanged**;
- открытое свойство булевского типа для информации о том, что пользователь внес изменения в коллекцию после сохранения в файле;
- открытый метод **void Save(string filename)** для сохранения в файле данных класса с помощью сериализации;
- открытый метод **void Load(string filename)** для восстановления из файла данных класса с помощью десериализации.

Метод **void Save(string filename)**

- сериализует объект **List<V2Data>** в файл с именем **filename**;
- если файл с именем filename существует, приложение его перезаписывает; если такого файла нет, приложение его создает;
- метод бросает исключение, если в процессе сериализации или при создании/открытии файла произошла ошибка;
- независимо от того, как завершилась сериализация, все файловые потоки должны быть закрыты в блоке finally.

Метод **void Load(string filename)**

- десериализует объект **List<V2Data>** из файла с именем **filename**;
- метод бросает исключение, если в процессе десериализации или при открытии файла произошла ошибка;
- независимо от того, как завершилась десериализация, все файловые потоки должны быть закрыты в блоке **finally**.

Замечания.

1. При сериализации данных из **V2MainCollection** необходимо использовать нестандартную сериализацию (лекция 12 прошлого семестра, слайды 10-12), так как к типам **System.Numerics.Vector2** и **System.Numerics.Vector3** не прикреплен атрибут **[Serializable]**.
2. К полям типа **event** надо прикрепить атрибут **[field:NonSerialized]**.

Пользовательский интерфейс программы

Главное окно приложения содержит меню с элементами

- **File** (с элементами **New**, **Open**, **Save**);
- **Edit** (с элементами **Add Defaults, Add Default V2DataCollection, Add Default V2DataOnGrid**, **Add Element from File**, **Remove**.

Реакция приложения на выбор пользователем элементов меню File:

New – создается новый объект **V2MainCollection.**

Open – пользователь выбирает имя файла в стандартном диалоге **Microsoft.Win32**.**OpenFileDialog**. Если пользователь сделал выбор (закрыл диалог кнопкой **Open**) выполняется десериализация данных в **V2MainCollection**.

Save – пользователь выбирает имя файла в стандартном диалоге **Microsoft.Win32**.**SaveFileDialog**. Если пользователь сделал выбор (закрыл диалог кнопкой **Save**), данные **V2MainCollection** сериализуются в файл с именем, который выбрал пользователь.

Если перед выбором элементов меню **New** или **Open** или перед выходом из приложения пользователь изменил коллекцию **V2MainCollection** и не сохранил ее (не сериализовал в файл), он получает предупреждение о том, что данные будут потеряны**.** Предупреждение выводится с помощью стандартного диалога **System.Windows.MessageBox.** Пользователю предлагается выбрать − сохранить в файле измененные данные или выполнить соответствующую операцию без сохранения результатов. Если пользователь выбрал сохранение данных, то вызывается стандартный диалог **Microsoft.Win32**.**SaveFileDialog** для выбора имени файла, в который будут сериализованы данные объекта **V2MainCollection**.

При **завершении работы приложения** проверку, что пользователь сохранил в файле коллекцию, в которую внес изменения, надо выполнить обработчике события **Closed** главного окна приложения.

Реакция приложения на выбор пользователем элементов меню Edit

- **Add Defaults** – в коллекцию **V2MainCollection** добавляется несколько элементов с данными по умолчанию;
- **Add Default V2DataCollection** – в коллекцию добавляется один элемент **V2DataCollection** с данными по умолчанию;
- **Add Default V2DataOnGrid** – в коллекцию добавляется один элемент **V2DataOnGrid** с данными по умолчанию;

- **Add Element from File** – в коллекцию добавляется новый элемент **V2DataOnGrid** (для варианта 2-1 прошлого семестра) или **V2DataCollection** (для варианта 2-2 прошлого семестра**),** данные которого читаются из файла;
- **Remove** – из коллекции удаляется элемент, который пользователь выбрал в **ListBox** со всей коллекцией **V2MainCollection**.

В обработчиках элементов меню **Edit** вызываются соответствующие методы класса **V2MainCollection**.

При добавлении в коллекцию нового элемента **V2DataCollection** или **V2DataOnGrid,** данные которого читаются из файла, пользователь выбирает имя файла в стандартном диалоге **Microsoft.Win32**.**OpenFileDialog**. Если пользователь сделал выбор (закрыл диалог кнопкой **Open**) и данные правильно прочитались из файла, в коллекцию добавляется новый элемент. В противном случае в окне сообщений **MessageBox** пользователь получает сообщение об ошибке и может продолжить работу.

Главное окно приложения содержит следующие элементы управления**:**

- **ListBox** (с именем lisBox_Main) для вывода всех элементов из списка **List<V2Data>** коллекции **V2MainCollection**;
- **ListBox** (с именем lisBox_ DataCollection) для вывода элементов списка **List<V2Data>** из **V2MainCollection**, которые имеют тип **V2DataCollection**;
- **ListBox** (с именем lisBox_ DataOnGrid) для вывода элементов списка **List<V2Data>** из **V2MainCollection**, которые имеют тип **V2DataOnGrid**;
- **TextBlock,** в который выводится среднее значение модуля поля для всех результатов измерений в коллекции **V2MainCollection**.

Замечание. Имена для элементов управления **ListBox** добавлены, чтобы было проще ссылаться на элементы управления в тексте лабораторной работы. В программе можно использовать другие имена для элементов управления **ListBox.**

Все перечисленные выше элементы управления надо связать с данными **V2MainCollection** с помощью механизма привязки.

Привязку можно реализовать разными способами. Например, в класс главного окна приложения можно добавить закрытое поле типа **V2MainCollection.** Свойству **DataContext** главного окна приложения присвоить ссылку на объект типа **V2MainCollection** и использовать **DataContext** как источник данных в привязках.

Вывод данных из подмножества элементов коллекции V2MainCollection, которые имеют тип V2DataCollection

В элемент управления lisBox_DataCollection выводится подмножество коллекции, состоящее из элементов типа **V2DataCollection.**

Для привязки к элементу управления **ListBox**, в который выводится подмножество коллекции, надо создать **представление коллекции**.

Главное окно приложения содержит еще один элемент управления **ListBox** (с именем listBox_details) для вывода данных из элемента **V2DataCollection**, который

пользователь выбрал в lisBox_DataCollection**.** В элемент управленния listBox_details выводятся все элементы списка **List<DataItem>.**

При выводе используется шаблон данных **DataTemplate**. Шаблон **DataTemplate** содержит два элемента управления **TextBlock**, каждый из которых использует в привязке пользовательский преобразователь типа. Пользовательский преобразователь типа для первого элемента **TextBlock** формирует строку, содержащую координаты точки измерения поля. Пользовательский преобразователь типа для второго элемента **TextBlock** формирует строку с комплексным значением поля. Шаблон данных **DataTemplate** надо определить в объектных ресурсах приложения.

Элемент listBox_details надо связать с **V2DataCollection** с помощью привязки. В приложении **не должно быть** обработчиков события **SelectionChanged** для элементов управления **ListBox**.

Вывод подмножества элементов коллекции V2MainCollection, которые имеют тип V2DataOnGrid

В элемент управления lisBox_DataOnGrid выводится подмножество коллекции, состоящее из элементов типа **V2DataOnGrid.**

Для привязки к элементу управления **ListBox**, в который выводится подмножество коллекции, надо создать **представление коллекции**.

Главное окно приложения содержит элемент управления **TextBlock** для вывода информации об элементе **V2DataOnGrid,** который пользователь выбрал в lisBox_DataOnGrid.

В элемент **TextBlock** выводятся минимальное и максимальное значения модуля поля в выбранном элементе **V2DataOnGrid**. В привязке используется пользовательский преобразователь типа.

В классе **V2DataOnGrid** надо определить открытые свойства, возвращающие максимальное и минимальное значения модуля поля на сетке.

Обновление элементов управления при изменении коллекции V2MainCollection

При изменении коллекции **V2MainCollection** (добавление нового элемента, удаление элемента из коллекции или замена ссылки на элемент в **List<V2Data>**) должна обновляться информация в элементах пользовательского интерфейса – в элементах **ListBox** с коллекцией и ее представлениями и элементе **TextBlock**, в который выводится информация о том, что пользователь изменил коллекцию и не сохранил данные.

Для этого в классе **V2MainCollection** необходимо реализовать интерфейс **System.Collections.Specialized.INotifyCollectionChanged.** Методы класса, которые изменяют коллекцию, сообщают о том, что в коллекции произошли изменения с помощью события **CollectionChanged. WPF** обновляет данные в элементах управления, если коллекция или ее представление используется как источник данных в привязке.

Событие **CollectionChanged** надо использовать для обновления значения булевского поля, в котором хранится информация о том, что коллекция изменилась в процессе работы приложения. Для этого в конструкторе **V2MainCollection** надо подписаться на событие **CollectionChanged** и в обработчике этого события (методе класса **V2MainCollection**) изменять значение булевского поля.

Для того, чтобы обновлялись элементы управления **TextBlock,** к свойствам которого выполнена привязка, в типах, которые используются как источник привязки, должен быть реализован интерфейс **System.ComponentModel.INotifyPropertyChanged**. Свойства типа, которые используются в привязке, должны сообщать о своем изменении с помощью события **PropertyChanged.**

Обработка исключений

Все исключения, которые могут возникать при обработке некорректного ввода пользователя, должны обрабатываться приложением.

Приложение должно оставаться в рабочем состоянии до тех пор, пока пользователь не закроет главное окно приложения.

Срок сдачи лабораторной работы 14 марта