Лабораторная работа № 5

В связи с внезапно надвигающийся сессией, вам необходимо разработать приложение «Паника» - электронный справочник учета сданных / не сданных предметов. Паника позволит вам не запутаться в том, какие дисциплины вы сдали, а какие предстоит еще сдать, и тем самым доведёт вас до нервного срыва раньше чем закончится сессия!

Техническое задание на разработку

1. Разработать информационную систему, позволяющую работать со списком дисциплин текущей сессии (рис. 1).
2. Реализовать фильтрацию дисциплин по статусу (рис. 2).
3. Визуализировать список дисциплин следующим образом: заменять логическое свойство Статус цветом.
4. Список дисциплин хранится в файле и считываться асинхронно при старте приложения из файла (рис. 3)
5. Приложение должно позволять добавлять, удалять дисциплины, а также менять Статус на противоположный.
6. Сохранять текущий отображаемый список дисциплин в файл.
7. Приложение необходимо реализовать посредствами технологии WPF.

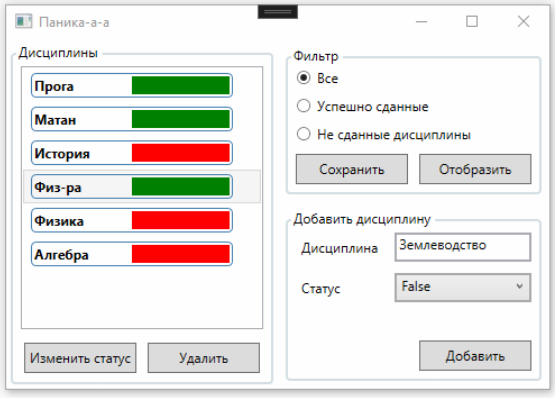

Рис 1.

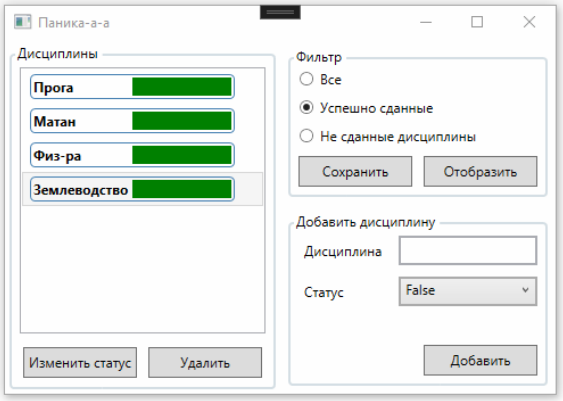

Рис 2.

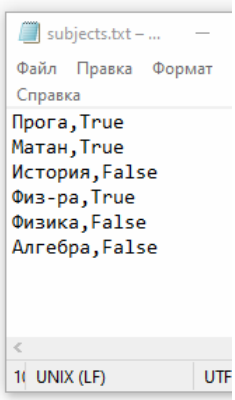

Рис 3.

Рекомендации к коду:

1. Всю программную логику (коллекции, методы и свойства) реализовать в отдельном классе - `Logics`. Для того, чтобы в дальнейшем ваше окно могло забирать данные с объекта этого класса, нужно в codebehind окна `MainWindow` в его конструкторе, сразу после инициализации компонентов, определить контекст (источник) данных.
   Пример:
```
public partial class MainWindow : Window {
     public MainWindow() {
         InitializeComponent(); DataContext = new Logics ();}}
```
2. Данные для отображения берутся посредством технологии привязки данных - `Binding`. Следующий пример демонстрирует как элемент отображения списков (`ListView`) получает данные по предметам привязываясь к соответствующему полю (`Subjects`) своего `DataContext`. `DataContext` – свойство `MainWindow` в который мы поместили объект класса `Logics`. Аналогичным образом мы передаем в `DataContext` информацию с визуального окна о выбранном предмете через свойство `SelectedSubject`, которое берет это значение в свойстве `SelectedItem` контрола `ListView`. Посредством Binding `DataContext` узнает о визуальном состоянии экранной формы и может реагировать на действия пользователя, выполняя те или иные команды, которые объявлены в классе `Logics`, который в свою очередь лежит в `DataContext`.
   Пример:
```
<ListView ItemsSource="{Binding Subjects}" SelectedItem="{Binding SelectedSubject}"/>
```
3. Логика вашего приложения будет менять свойства объектов, отображаемых на форме: статус у дисциплины. Для автоматической перерисовки информации на форме, необходимо, что бы те свойства, которые подвержены изменениям умели сообщать об этом посредством интерфейса

`INotifyPropertyChanged` и вызовом соответствующего события через метод `NotifyPropertyChanged()`. Если подразумевается изменение отображаемых коллекций, то нужно использовать тип `ObservableCollection`. Такая коллекция автоматически сообщит экранной форме о своих внутренних изменениях, которые вы делаете через методы: Add и Remove. Но обратите внимание, если идет подмена коллекции, например пересоздается заново коллекция (у вас это возможно будет при фильтрации), то тут меняется свойство класса (не содержимое коллекции), а следовательно это свойство должно также уведомить экранную форму посредством INotifyPropertyChanged.

Пример:

```
class Subject: INotifyPropertyChanged
{
    public string Title {set {_title = value; NotifyPropertyChanged(); }
}
```

4. Для преобразования логического значения в соответствующий цвет заливки многоугольника используйте интерфейс `IValueConverter` при реализации класса-конвертера в котором принимаемое логическое значение будет преобразовываться в указанный вами цвет.
   Пример: `if ((bool)value) {return Brushes.Green;} return Brushes.Red;`
   Применить конвертер необходимо в разметке, в измененном шаблоне (`ItemTemplate`) у `ListView:`
   `<Rectangle Fill ="{Binding Status, Converter={StaticResource BoolToColorConverter}}"/>`

5. Для связи кнопки на экранной форме с выполнением необходимой функции из `DataContext` (там находится объект класса `Logics`) применяйти тип танных `RelayCommand` (скачивается из сети через менеджер пакетов)
   Пример:
   В разметке формы
   ```
   <Button Command="{Binding ChangeCommand}" />
   ```
   В классе `Logics`
   ```
   ChangeCommand = new RelayCommand (changeSubject);
   private void changeSubject() { SelectedSubject.Status = !SelectedSubject.Status; }
   ```

6. Для преобразования логического значения в соответствующий цвет заливки многоугольника используйте интерфейс `IValueConverter` при реализации класса-конвертера в котором принимаемое логическое значение будет преобразовываться в указанный вами цвет.
   Пример: `if ((bool)value) {return Brushes.Green;} return Brushes.Red;`
   Применить конвертер необходимо в разметке, в измененном шаблоне (`ItemTemplate`) у `ListView:`
   `<Rectangle Fill ="{Binding Status, Converter={StaticResource BoolToColorConverter}}"/>`