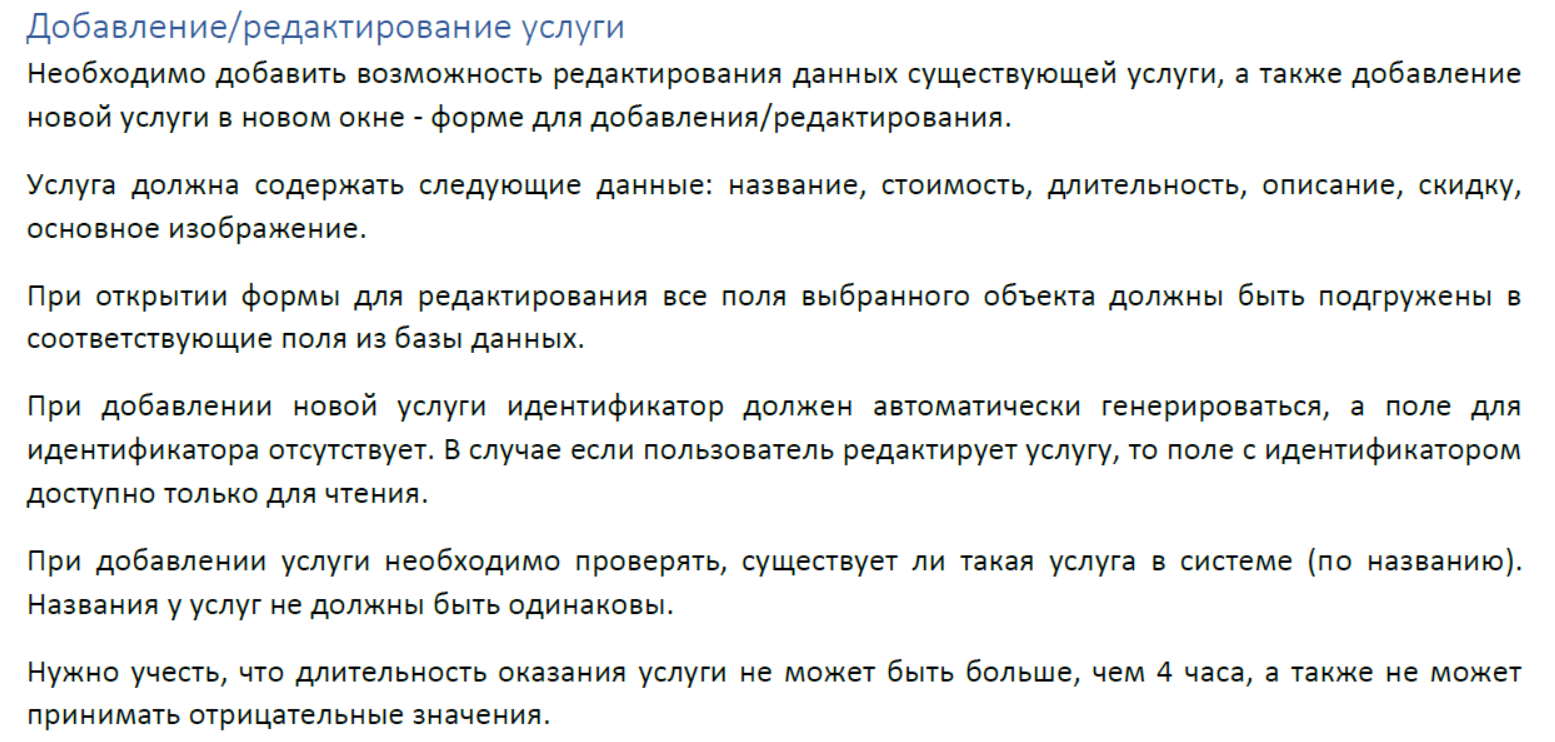
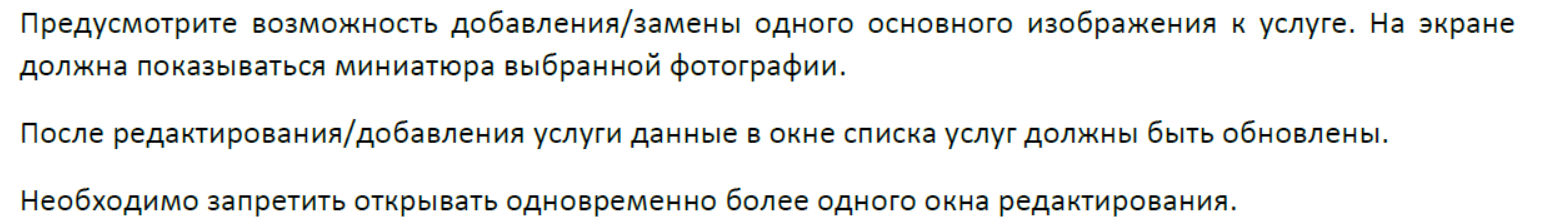
Разработка формы «Добавление/редактирование услуг»

**1. Клонировать с гитхаба предыдущую версию проекта (итог лабы5)**

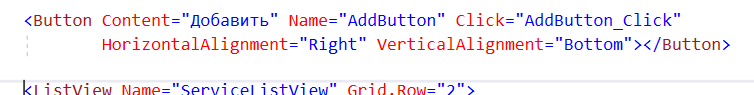
2. Посмотрим Задание из файла Сессия1.pdf:



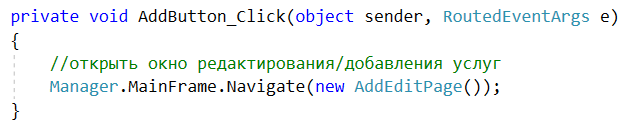


**Начнем с добавления услуги.**

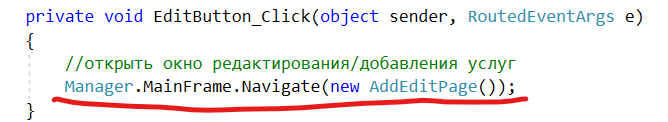
На странице ServicePage.xaml добавим элемент управления – кнопку «Добавить» перед листвью:



Пропишем код обработчика:



3. Также сделаем переход сюда через кнопку Редактировать:



4. Далее работать будем на странице AddEditPage.xaml.

Разместим на странице необходимые элементы управления (удалить с нее первоначальную надпись не забудьте)):

a) Сверстаем сетку, состоящую из 5 строк и 2 столбцов:

<Grid.ColumnDefinitions>

<ColumnDefinition Width="100"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

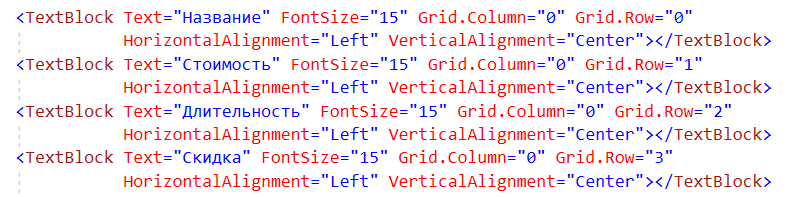
<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

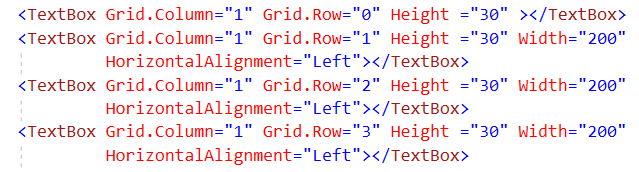
</Grid.RowDefinitions>

b) Разместим элементы для ввода данных:

Текстовые блоки для отображения подсказок, что именно вводить



TextBox для ввода

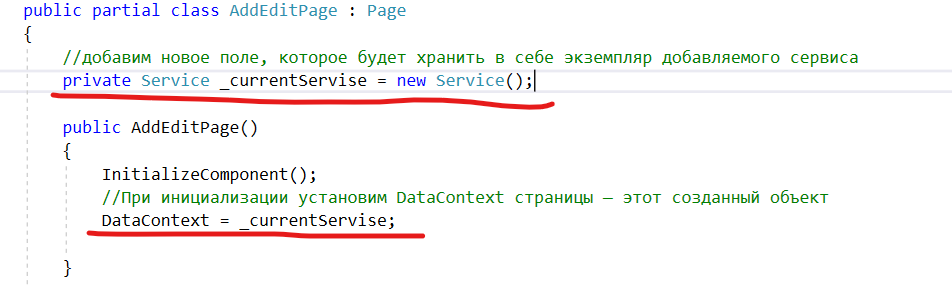


с) Добавим кнопку для сохранения изменений, дадим ей имя и укажем обработчик нажатия:



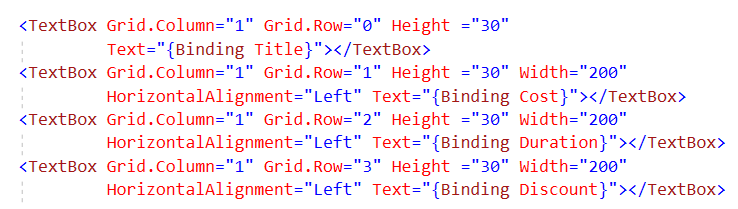
5. На странице кода AddEditPage.xaml.cs добавим новое поле, которое будет хранить в себе экземпляр добавляемого сервиса (услуги).

И при инициализации установим DataContext страницы — этот созданный объект.



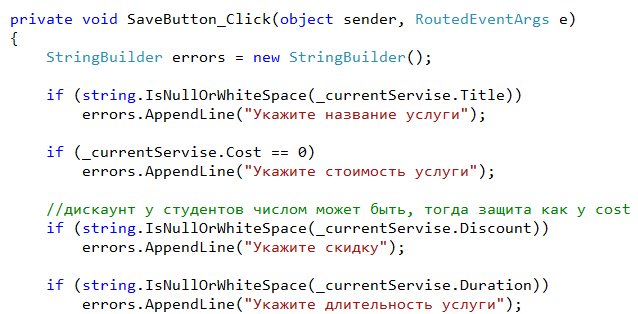
6. Затем, используя привязку данных, укажем, какому свойству обращаться к каждому элементу при загрузке данных.

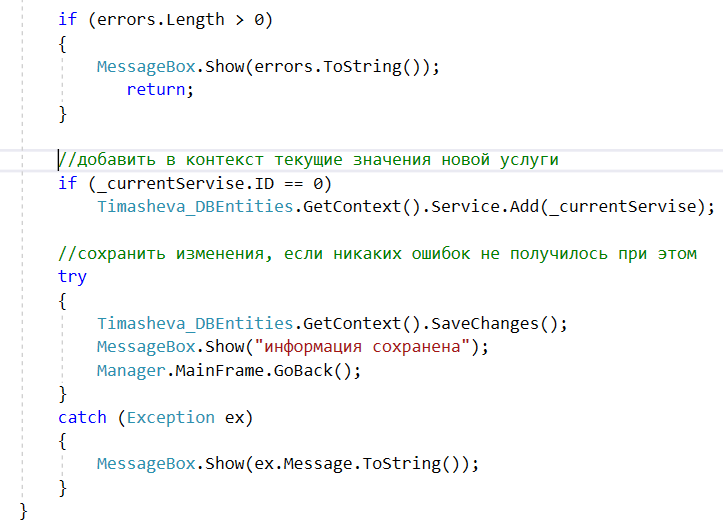
Например, свойство Text у первого TextBox'а будет обращаться к названию услуги (*вам нужно дописать-добавить свойство Text для элементов TextBox*):



7. Далее обработаем нажатие на кнопку Сохранение и в коде пропишем логику обращения к модели данных и добавления нового экземпляра услуги:

Код обработчика:

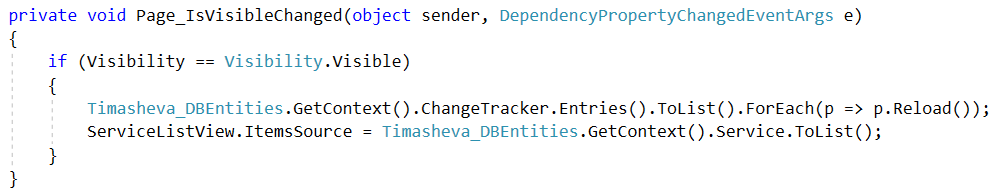




7. Теперь можно реализовать возврат назад для страницы ServicePage.xaml. При возврате на страницу со списком услуг, нам необходимо выводить актуальную информацию, обновляя список в листвью. Для этого будем использовать событие у страницы IsVisibleChange. Оно срабатывает каждый раз, когда страница отображается, либо скрывается. Дополним описание страницы:

Title="Услуги автосервиса" FontFamily="Tw Cen MT" ***IsVisibleChanged="Page\_IsVisibleChanged">***

Перейдем в код:

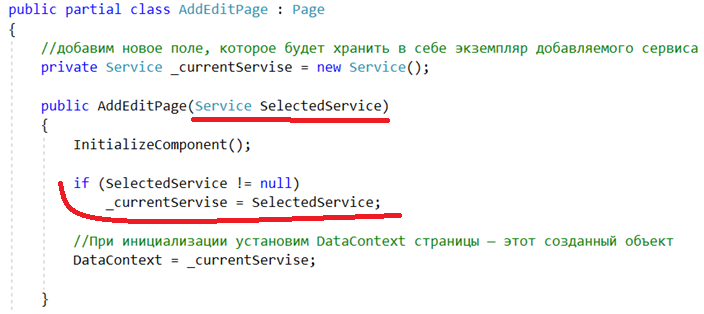


Пояснение: Если видимость страницы isVisible, будем обращаться к контексту с помощью свойства ChangeTracker ко всем сущностям, которые есть. И для каждой из них будем выполнять метод перезагрузки и вывода актуальных данных.

Запустить, проверить работу.

8. Для функции редактирования данных целесообразно использовать ту же страницу, что делали для добавления. Каким образом это будет происходить? В случае, если пользователь намерен изменить информацию об объекте, система будет отображать страницу добавления с информацией о редактируемом объекте. Измененная информация будет фиксироваться в базе данных и отображаться в списке, как было при добавлении.

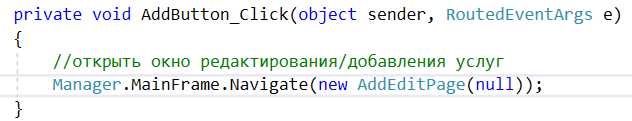
В первую очередь добавим параметр странице AddEditPage.xaml.cs. В нее будем передавать экземпляр выбранной услуги и, в случае если он не пустой, присваивать нашему полю CurrentService:

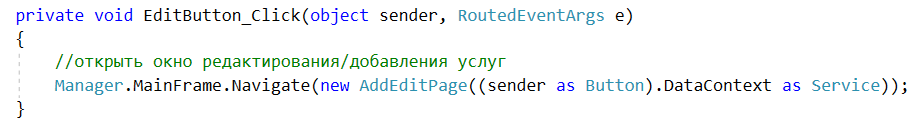


Но не можем сейчас запустить приложение, т. к. возник ряд ошибок.

Вызов страницы AddEditPage теперь требует какого-то аргумента. В случае, если будем делать добавление, то просто пропишем null (отправим пустой экземпляр). При этом для редактирования кнопке EditButton уже будем передавать экземпляр, прописав для этого код. Вместо null будем обращаться к кнопке, на которую нажали, получать ее контекст и знать, что это — услуга.

Поправим обработчики событий в коде ServicePage.xaml.cs:





Запустить, проверить работу. Не забудьте прописать в файле разметки для кнопки «Редактировать» атрибут Click.

Результат выполнения лаб.работы залить на гитхаб под новым коммитом (6 лаба).