*Государственное образовательное бюджетное учреждение высшего профессионального образования*

*«Национальный исследовательский университет Высшая Школа Экономики»*

*Факультет бизнеса и менеджмента - Школа бизнес-информатики*

**Кафедра управления информационными системами и цифровой инфраструктурой**

**Проект по дисциплине «Программирование»**

VDeditor

**Выполнили:**

Ковальчук Д.А.

Годунова В.И.

**Руководитель**старший преподаватель, к.т.н.

С.Г. Ефремов

Москва 2016

**Оглавление**

[**Общая информация** 3](#_Toc446248541)

[**Классы** 4](#_Toc446248542)

[**Скриншоты интерфейса** 6](#_Toc446248543)

**Общая информация**

Название проекта: Графический редактор VDeditor

В настоящее время существует великое множество графических редакторов, доступных как в оффлайн, так и в онлайн режимах. Все они обладают примерно одинаковым набором доступных функций, поэтому в этой сфере достаточно сложно придумать что-то новое.

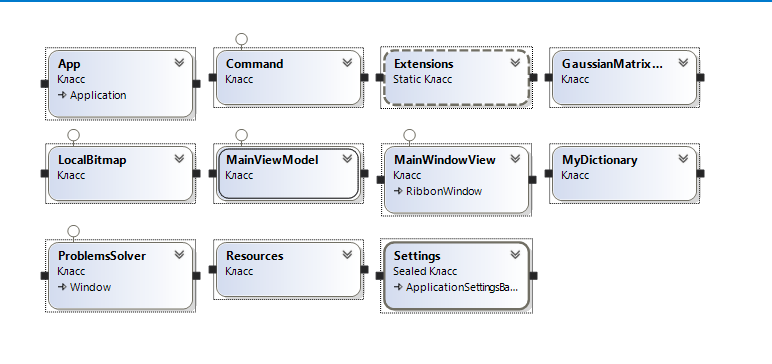
Целью проекта было создание графического редактора с интуитивно понятным интерфейсом, необходимым базовым функционалом и дополнительной опцией сохранения истории изменений в файл. На первый взгляд опция кажется совершенно ненужной, и лучше бы вторым потоком шло открытие пасьянса «Паук». Однако задумывались ли Вы над тем, что зачастую приходится выполнять одни и те же действия, к примеру, при обработке собственных фотографий? Чтобы избавить пользователей от однообразного повторения, мы подготовили первоначальный функционал для создания в дальнейшем аналога макросов в фоторедакторе, когда достаточно просто загрузить файл с записанными действиями, и фотография будет изменена привычным образом.

Репозиторий: <https://github.com/mEss1anh/photoEditorApp.git>

Участники проекта: Ковальчук Дмитрий – разработка общей логики приложения, реализация многопоточности;

Годунова Виктория – работа с интерфейсом, вспомогательные действия в логике приложения, тестирование.

**Классы**



Список классов

View

MainWindowView.xaml

Главное окно проекта, в котором располагаются основные элементы функционала: меню открытия/сохранения, кнопки фильтров, список изменений. Поскольку в дальнейшем планируется развивать и улучшать проект, было решено использовать ленту в качестве основного меню, так как она более других подходит для этих целей. Так же в xaml-коде прописаны основные связи свойств (Data binding).

MainWindowView.xaml.cs

Ввиду реализации принципов MVVM данный класс остается относительно пустым и содержит в себе методы реализации чисто графических задумок

ProblemsSolver.xaml

Стандартное окошко для вывода ошибок в WPF весьма скучно и примитивно. Мы решили предложить свой вариант вспомогательного окна.

ProblemsSolver.xaml.cs

Класс, отвечающий за закрытие вспомогательного окна.

ViewModel

Command.cs (наследует ICommand)

Использование команд удобно в случаях, когда у программы много функциональных кнопок и связанных с ними методов, что делает код более структурированным и легко воспринимаемым.

Реализация была взята с сайта <http://svyatoslavpankratov.blogspot.ru/2011/11/mvvm-pattern-1.html>,

так как она универсальна и часто используется, то не было необходимости прописывать ее самостоятельно.

MainViewModel.cs (наследует INotifyPropertyChanged)

Главный класс раздела ViewModel, содержащий поля, свойства и методы реализации фильтров, вращения и увеличения изображения, и являющийся связующим звеном между видом и моделью нашего проекта.

Методы реализации фильтров были взяты с сайтов

<http://softwarebydefault.com/2013/06/09/image-blur-filters/>

<http://softwarebydefault.com/2013/05/18/image-median-filter/>

Model

LocalBitmap.cs (наследует INotifyPropertyChanged)

Основная модель проекта, описывающая главную сущность – изображение, которое в дальнейшем будет обработано.

GaussianMatrixForBlur.cs

Здесь собраны кальки цветовых матриц для фильтра, чтобы не усложнять основной код программы.

MyDictionary.cs

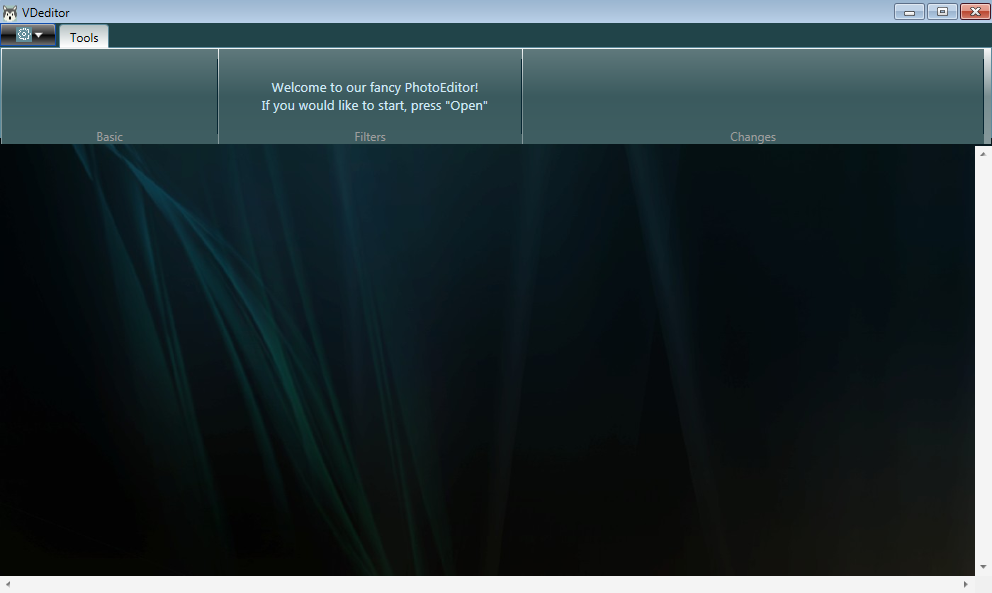
Словарь для унификации записей в истории изменений редактируемого изображения.

Extensions.cs

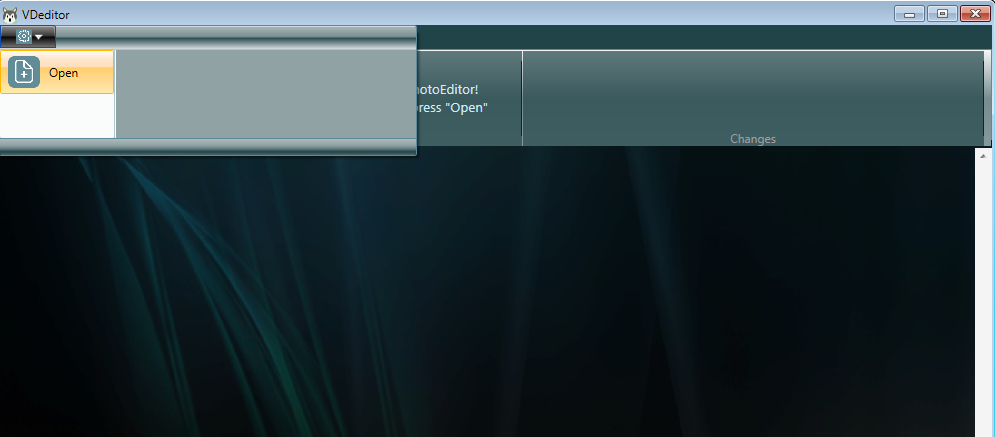
Расширенный метод работы с растровым изображением был вынесен в отдельный класс во избежание загруженности кода основного класса.

**Скриншоты интерфейса**

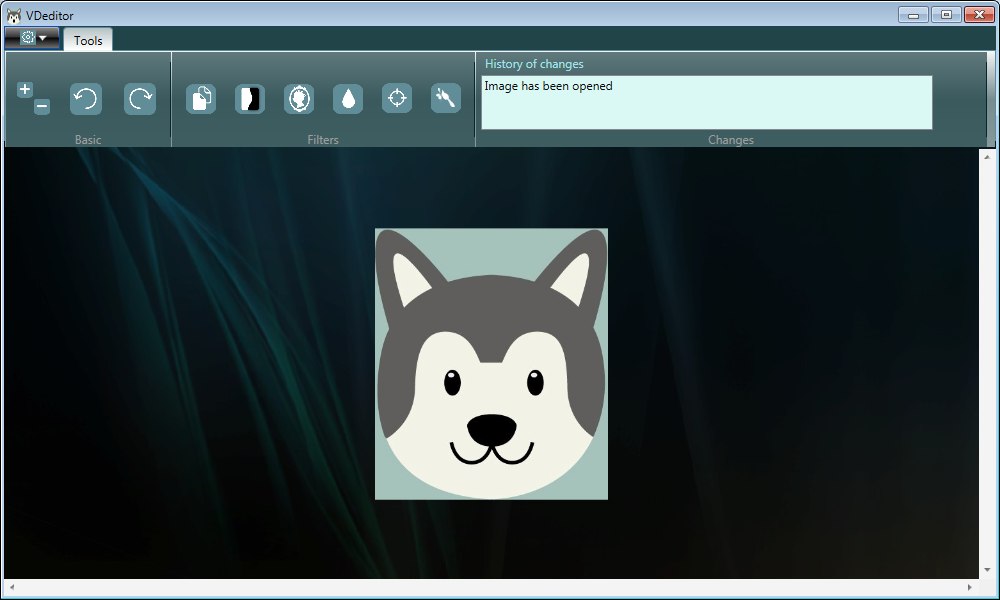
Отсутствие доступа к функционалу приложения до открытия изображения



Меню открытия изображения



Общий вид интерфейса



Появление исключения в случае нажатия кнопки филтра при не открытом изображении

