Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Магнитогорский государственный технический университет

им. Г. И. Носова»

Многопрофильный колледж

Отделение №2 «Информационные технологии и транспорт»

**КУРСОВОЙ ПРОЕКТ**

**РАСЧЁТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

По ПМ.02. ОСУЩЕСТВЛЕНИЕ ИНТЕГРАЦИИ ПРОГРАММНЫХ МОДУЛЕЙ

МДК.02.02. Инструментальные средства разработки программного обеспечения

на тему: Разработка программного продукта для создания покадровой анимации

Исполнитель: Щелутченко Александр Дмитриевич, студент 4 курса, группа ИСпП-20-1

Руководитель: Тутарова В.Д, преподаватель МпК

Работа допущена к защите “ ” 2023г.

(подпись)

Работа защищена “ ” 20 г. с оценкой

(оценка) (подпись)

Магнитогорск, 2023

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Магнитогорский государственный технический университет   
им. Г. И. Носова»

Многопрофильный колледж

**КУРСОВОЙ ПРОЕКТ**

Тема: Разработка программного продукта для создания покадровой анимации

Студента: Щелутченко А. Д.

Задание

Исходные данные по заданной предметной области:

интернет-источники, литература

Состав и содержание проекта:

1 Теоретико-информационный анализ предметной области

* 1. 1.1 Анализ предметной области
  2. 1.2 Анализ и оценка существующих программных продуктов
  3. 1.3 Постановка задачи

2 Проектирование программного обеспечения

* 1. 2.1 Моделирование программного обеспечения
  2. 2.2 Обоснование выбора средств разработки
  3. 2.3 Разработка базы данных
  4. 2.4 Проектирование меню и интерфейса программного обеспечения
  5. 2.5 Спецификация программы
  6. 2.6 Описание запросов при разработке программного обеспечения
  7. 2.7 Проектирование отчетов
  8. 2.8 Администрирование программного обеспечения

3 Описание работы программного обеспечения

* 1. 3.1 Руководство пользователя
  2. 3.2 Руководство программисту

Срок сдачи: «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023\_\_г.

Руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /В.Д. Тутарова

«\_\_\_\_\_» \_\_\_\_\_\_\_\_202\_\_г.

Задание получил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / А. Д. Щелутченко

«\_\_\_\_\_» \_\_\_\_\_\_\_\_202\_\_г.

Магнитогорск, 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc163106415)

[1. ТЕОРЕТИКО-ИНФОРМАЦИОННЫЙ АНАЛИЗ ПОКАДРОВОЙ АНИМАЦИИ 5](#_Toc163106416)

[1.1 Анализ покадровой анимации 5](#_Toc163106417)

[1.2 Анализ и оценка существующих программных продуктов 12](#_Toc163106418)

[1.3 Постановка задачи 19](#_Toc163106419)

[1.3.1 Цель разработки 19](#_Toc163106420)

[1.3.2 Задачи разработки программного продукта 19](#_Toc163106421)

[2. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 20](#_Toc163106422)

[2.1 Моделирование программного обеспечения 20](#_Toc163106423)

[2.1.1 Диаграмма прецедентов 20](#_Toc163106424)

[2.1.2 Функциональная модель 21](#_Toc163106425)

[2.2 Обоснование выбора средств разработки 22](#_Toc163106426)

[2.3 Разработка базы данных 24](#_Toc163106427)

[2.4 Проектирование интерфейса программного обеспечения 24](#_Toc163106428)

[2.5 Спецификация программы 27](#_Toc163106429)

[3 ОПИСАНИЕ РАБОТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 29](#_Toc163106430)

[3.1 Руководство пользователя 29](#_Toc163106431)

[3.2 Руководство программисту 29](#_Toc163106432)

[ЗАКЛЮЧЕНИЕ 30](#_Toc163106433)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 31](#_Toc163106434)

[ПРИЛОЖЕНИЕ А 32](#_Toc163106435)

[ПРИЛОЖЕНИЕ Б 45](#_Toc163106446)

# **ВВЕДЕНИЕ**

В современном информационном обществе графическая анимация становится всё более значимой и популярной. Она используется в различных областях, начиная от развлечений и рекламы до образования и презентаций. Кроме того, с появлением новых технологий и онлайн платформ, использование анимированной графики становится еще более широким и доступным.

Актуальность проекта обусловлена сложностью создания покадровой анимации для нового пользователя, так как современные программы интуитивно непонятны и переполнены ненужными функциями для обычного пользователя.

Цель проекта заключается в разработке программного продукта, который позволяет пользователям создавать простейшую анимацию и просматривать видеоролики, демонстрирующие процесс её создания.

Задачи проекта:

* теоретико-информационный анализ предметной области;
* анализ предметной области;
* анализ и оценка существующих программных продуктов;
* постановка задачи;
* проектирование программного обеспечения;
* моделирование программного обеспечения;
* обоснование выбора средств разработки;
* проектирование меню и интерфейса программного обеспечения;
* спецификация программы;
* описание запросов при разработке программного обеспечения;
* проектирование отчетов;
* администрирование программного обеспечения;
* описание работы программного обеспечения;
* руководство пользователя;
* руководство программисту.

# **ТЕОРЕТИКО-ИНФОРМАЦИОННЫЙ АНАЛИЗ ПОКАДРОВОЙ АНИМАЦИИ**

# **Анализ покадровой анимации**

Анимация – это процесс создания иллюзии движения путем последовательного отображения статичных изображений (кадров) с определенной скоростью. Она является мощным средством визуальной коммуникации, используемым в кино, телевидении, играх, рекламе и других областях. Одной из основных техник анимации является покадровая анимация.

Покадровая анимация – это метод создания движущихся изображений путем последовательного отображения серии кадров, каждый из которых немного отличается от предыдущего. Каждый кадр представляет собой отдельное изображение, зафиксированное в определенный момент времени. Когда эти кадры отображаются с высокой скоростью, они создают иллюзию плавного и непрерывного движения.

Изображение в анимации – это графическое представление объекта, персонажа или сцены в каждом конкретном кадре анимационного проекта. Изображение может быть двухмерным или трехмерным, в зависимости от используемой техники и программного обеспечения.

В анимации изображение является основным строительным блоком, который составляет каждый кадр анимации. Оно может быть создано путем рисования, моделирования или комбинации различных визуальных элементов в соответствии с задумкой и концепцией проекта. Затем изображения последовательно отображаются на экране с определенным интервалом времени, чтобы создать иллюзию движения.

Каждое изображение в анимации видоизменяется или изменяется незначительно по сравнению с предыдущим изображением. Такие изменения могут включать изменение позиции, формы, цвета или состояния объекта, или персонажа. Комбинация этих последовательных изображений создает восприятие непрерывного движения и жизненности в анимации.

Важно отметить, что изображение в анимации зависит от выбранного стиля и техники, а также от творческого видения и предпочтений аниматора. Оно может быть реалистичным или стилизованным, простым или сложным, в зависимости от целей и намерений создателя.

Изображение в анимации играет важную роль в передаче идеи, эмоции и намерений анимации. Качество и выразительность изображений напрямую влияют на визуальное восприятие анимации и способность передать информацию, моменты действия или эмоции зрителю. Поэтому создание и обработка изображений в анимации является неотъемлемой частью процесса создания анимационного проекта.

Покадровая анимация имеет свои особенности и принципы. Она строится на основе закона покадровой съемки и создания последовательности кадров, которые воспроизводятся с определенным интервалом времени и создают иллюзию движения. Для достижения непрерывного потока движения, каждый кадр должен быть незначительно изменен по сравнению с предыдущим, чтобы обеспечить плавную и естественную анимацию.

Процесс создания покадровой анимации включает несколько этапов. Сначала определяется и разрабатывается концепция или идея анимации. Затем создается сценарий или описание последовательности событий, которые будут происходить в анимации. Далее, художник или аниматор создает иллюстрации или рисунки для каждого кадра анимации, учитывая изменения в позиции, форме или состоянии объектов. Процесс рисования каждого кадра может быть выполнен вручную на бумаге, на графическом планшете или с использованием специализированных программных средств. Когда все кадры готовы, они собираются в последовательность и воспроизводятся с определенной скоростью, чтобы создать движение.

Покадровая анимация широко применяется в различных областях, включая кино и мультфильмы, рекламу, компьютерные игры, обучение и другие формы визуальной коммуникации. Она имеет преимущества, такие как полный контроль над каждым кадром, возможность создания высококачественной и детализированной анимации, а также возможность представления различных стилей и эффектов.

Основные принципы, которые характеризуют покадровую анимацию, включают 1антиципацию (подготовка зрителя к предстоящему действию), 2разгоно-задержку (увеличение или уменьшение скорости движения),3 склейку (плавный переход от одного кадра к другому), антиципацию (показ начала действия или движения), эксагерацию (усиление или преувеличение действия), линзовое движение и принцип оверлея.

Развитие технологий компьютерной графики, программного обеспечения и специализированных инструментов позволяет создавать покадровую анимацию с еще большей эффективностью и точностью. Эта техника анимации остается актуальной и широко используется в современной графической анимации, позволяя создавать удивительные и привлекательные движущиеся изображения.

Однако ещё есть техники анимации такие как:

* рисованная анимация: также известна как традиционная или рукотворная анимация. Она основана на создании каждого кадра вручную на бумаге или других материалах. Аниматор рисует изображения, изменяющиеся с каждым кадром, и затем последовательно переходит от одного кадра к другому, создавая иллюзию движения;
* кукольная анимация: в этой технике используются трехмерные или двухмерные куклы, манипуляциями с которыми создается движение. Куклы могут быть сделаны из разных материалов, таких как дерево, пластик или ткань, и могут иметь гибкие части;
* пластилиновая анимация: в этой технике используется пластилин, который моделируется в различные формы и позы. Каждый измененный пластилин переходит на следующий кадр, создавая впечатление движения;
* песочная анимация: это техника, при которой песок используется для создания изображений и сцен. Аниматор двигает и формирует песок на стеклянной поверхности, создавая плавные и органические движения;
* трехмерная компьютерная анимация: в этой технике создается анимация с помощью компьютерных моделей и виртуальных сцен. Аниматор создает или импортирует трехмерные модели, назначает им анимацию и требуемые параметры, а затем использует программное обеспечение для создания движения и рендеринга этих моделей.

Каждая из этих техник имеет свои особенности и подходы, и выбор техники зависит от требований проекта, стиля и предпочтений аниматора. В современной анимации часто используется комбинация нескольких техник, чтобы достичь определенных эффектов и результатов.

Также анимации подразделяются на виды:

* традиционная анимация (рис. 1): также известная как рисованная анимация или рукотворная анимация. Этот вид анимации включает создание отдельных кадров вручную на бумаге или других материалах. Кадры рисуются и затем переставляются для создания движения. Традиционная анимация была основным методом создания анимации до появления компьютеров;

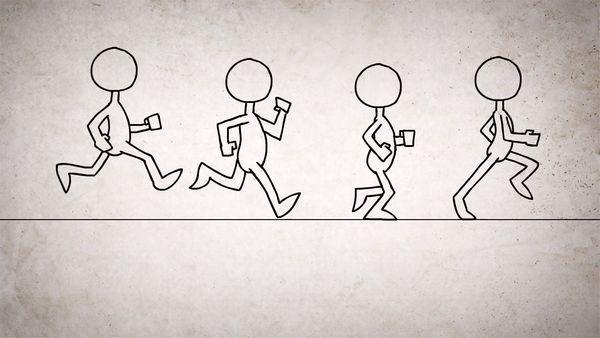


Рисунок 1 – Пример традиционной анимации

* компьютерная анимация (рис. 2): этот вид анимации создается с использованием компьютерных программ и специализированного программного обеспечения. Он включает создание и редактирование анимации с помощью компьютерных графических инструментов и программного обеспечения. Компьютерная анимация может быть двух- или трехмерной и использовать различные техники, такие как моделирование или рендеринг;



Рисунок 2 – Пример компьютерной анимации

* спецэффекты или визуальные эффекты VFX (рис. 3): этот вид анимации используется для добавления впечатляющих и реалистичных спецэффектов в кино, телевизионные шоу и видео. Специалисты в области визуальных эффектов используют компьютерные программы и инструменты, чтобы создать анимацию и добавить ее к реальным или снятым материалам;



Рисунок 3 – Пример визуального эффекта

* стоп-моушн анимация (рис. 4): это метод анимации, в котором фотографии или объекты фиксируются на кадре, а затем перемещаются или меняют свою позицию перед следующим кадром. Этот процесс повторяется множество раз, создавая иллюзию движения. Стоп-моушн анимация может быть основана на использовании различных объектов, таких как фигурки или куклы, или даже на живых образах для создания анимации;

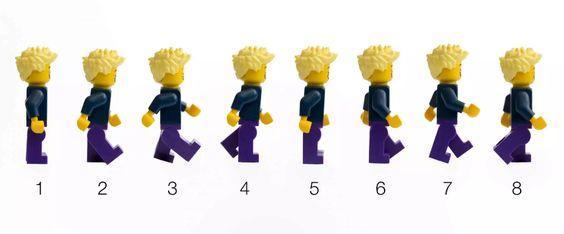


Рисунок 4 – Пример стоп-моушн анимации

* 3D-анимация (рис. 5): этот вид анимации основан на использовании трехмерных моделей и компьютерных графических технологий. 3D-анимация создает виртуальные объекты, которые манипулируются и анимируются в трехмерном пространстве. Она широко применяется в кино, компьютерных играх, рекламе и других областях, где требуется высокий уровень детализации и реалистичности.

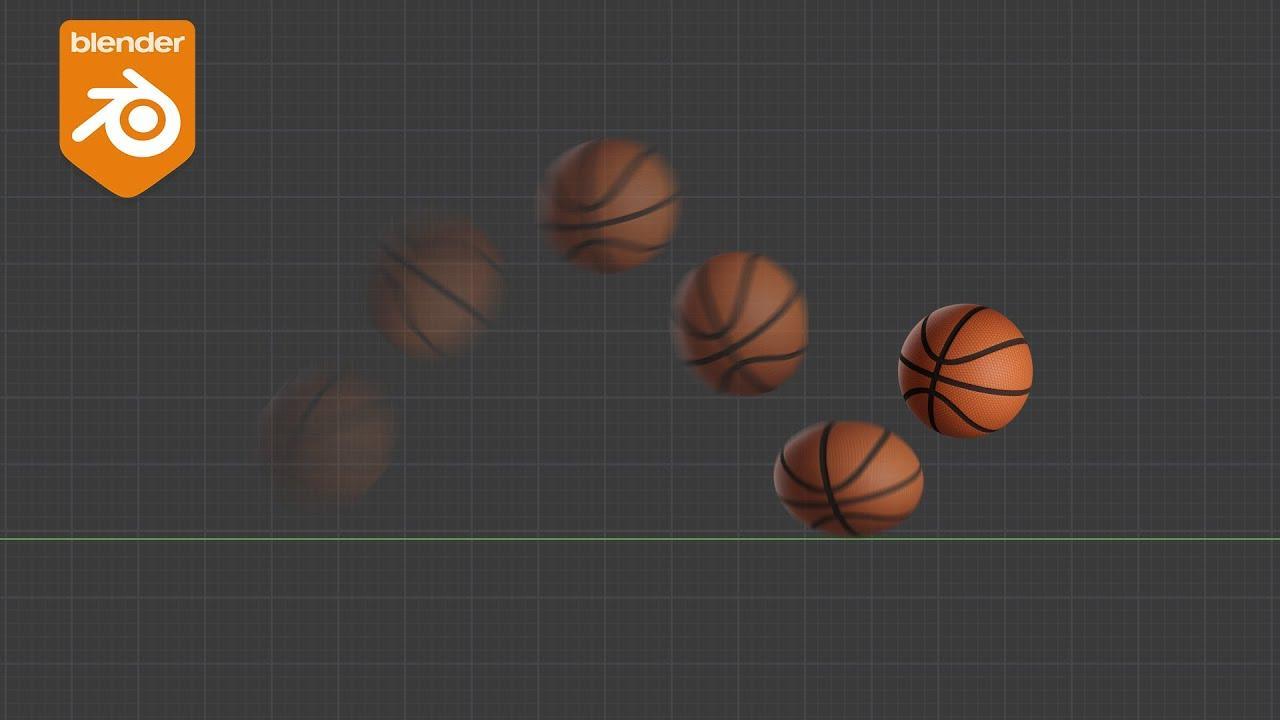


Рисунок 5 – Пример 3D-анимации

История появления покадровой анимации берет свое начало в далекие времена, когда люди начали экспериментировать с созданием движущихся изображений. Самыми ранними отметками можно считать пещерные рисунки, где дикие животные изображались с небольшими различиями в позах, что подразумевало их движение. Рисунки – это один из древнейших видов искусства, которое позволяло сохранять изображения на различных поверхностях и передавать информацию и истории от поколения к поколению. Покадровая анимация восходит к давним временам, когда люди обнаружили, что, путешествуя или пролистывая серию изображений быстро и последовательно, они могут создавать иллюзию движения.

Однако первым реальным шагом к созданию покадровой анимации было развитие феномена морфинга. Морфинг – это явление, когда серия изображений наращивается таким образом, что при их быстром прокручивании человеческое глаз не различает моменты, когда каждое изображение исчезает и следующие появляются. Именно благодаря этому явлению родилась кинематография и анимация.

Первыми искусствами, которые использовали покадровую анимацию, были лепка и живопись. Анимационные кадры, созданные из глины или других материалов, размещались в определенных позах, а затем фиксировались на камере. Получившееся изображение воспроизводило движение объектов или персонажей. Этот метод использовали такие мастера, как Франсуа Лаверджер, который создал первый скульптурный фильм «Кукольный спектакль» в 1896 году.

Следующим важным этапом стала разработка киноаппарата. Братья Люмьер, Жак и Луи, работали над созданием аппарата, который мог бы запечатлеть движение. Ими был изобретен киноаппарат «кинетограф», который позволял снимать и показывать фильмы. В 1895 году они демонстрировали свое изобретение в Париже, показав первый кинематографический фильм «Проезд поезда» на открытии кинотеатра.

Этот фильм стал отправной точкой для массового распространения покадровой анимации. С появлением доступного воспроизводящего устройства кинетоскопа, зрители могли сами просматривать анимационные фильмы в кинотеатрах или у себя дома. Вскоре художники и кинематографисты начали экспериментировать с анимацией, создавая различные символы, персонажи и истории на кадрах.

Важный момент в развитии покадровой анимации связан с появлением мультфильмов. Первым успешным мультфильмом стал «Греко-римский бой» (1910), созданный американским художником Винсентом Уолтером. Он использовал технику покадровой анимации, создавая серию рисунков, которые были проекцией на дисках, и двигались в процессе проецирования на экране. Этот метод позволил ему создать иллюзию движения и оживить рисунки.

Одним из пионеров в области покадровой анимации стал Ула Бурдекен, который в 1920-х годах начал создавать свои фильмы, используя концепцию «осцилляционной анимации». Этот метод заключался в создании анимации в движении с помощью легких электрических импульсов. Это позволило ему создавать разнообразные эффекты и движения, которые были невозможны для создания вручную.

В 1928 году Уолт Дисней создал первого мультяшного персонажа – Микки Мауса, который стал одним из самых известных и любимых персонажей мультфильмов. Дисней внедрил новые техники анимации, такие как смена фона и многоплановая камера, которые значительно улучшили качество и реализм анимации.

Однако настоящим прорывом в области покадровой анимации стало внедрение цветной анимации. В 1932 году австралийский художник и режиссер Герберт Соотсман разработал процесс под названием «техниколор», в результате которого цветовые фильтры использовались для создания цветных изображений при покадровой анимации. Это привлекло внимание кинорежиссеров и аниматоров, открывая новые возможности для анимации и кинематографии.

Со временем, появление компьютеров и развитие компьютерной графики полностью преобразили графическую анимацию. В 1963 году Иван Сазерленд разработал программу «Sketchpad», которая позволяла создавать и редактировать изображения с использованием графического интерфейса. Это стало отправной точкой для развития компьютерной графики и компьютерной анимации.

Одним из важных моментов в развитии графической анимации было появление компьютерной анимации в конце 1960-х - начале 1970-х годов. В это время была разработана первая система компьютерной графики, позволяющая создавать двухмерную и трехмерную анимацию. Это открыло новые горизонты для аниматоров и способствовало развитию графической анимации.

В 1972 году Эд Кэтмулл и Фред Парке, работая в лаборатории компьютерной графики при Университете Юты, создали первый компьютерный анимационный фильм «Hummingbird». Он был создан с помощью программы «KeyFrame», разработанной Кэтмуллом, и включал в себя современную технику компьютерной графики.

В 1982 году появился первый компьютерно-анимационный фильм полного метра «Трон», который стал первым коммерческим успехом в области компьютерной анимации. Фильм полностью создан с использованием компьютерной графики и представляет оригинальное сочетание живых актеров и виртуальной реальности.

В 1995 году компания Pixar выпустила полнометражный анимационный фильм «История игрушек», который стал первым полностью компьютерно-анимированным фильмом, став кассовым хитом и начал новую эру в индустрии анимации. Этот успех привел к созданию множества других компьютерно-анимированных фильмов и установил Pixar в качестве лидера в этой области.

С развитием компьютеров и программного обеспечения для создания покадровой анимации, таких как Autodesk 3ds Max, Adobe After Effects, Blender и другие, покадровая анимация стала еще более доступной и широко применяемой в различных сферах, включая кино, рекламу, компьютерные игры и визуализацию научных данных. Также покадровая анимация тесно связана с развитием интернета, мобильных устройств и социальных сетей. Множество приложений, игр и сайтов используют анимацию для привлечения внимания пользователей и создания интерактивного и увлекательного опыта. Также покадровая анимация широко применяется в обучении, создании тренировочных видео, инструкций и симуляторов, что позволяет повышать эффективность обучения и легко визуализировать сложные концепции или процессы.

Особенно в последние годы развитие покадровой анимации было связано с развитием технологий виртуальной и дополненной реальности. Эти технологии позволяют создавать совершенно новый уровень визуального опыта, оживляя изображения и взаимодействуя с реальным миром. Такие анимированные виртуальные миры могут использоваться в различных областях, включая игры, образование, моделирование и дизайн.

В настоящее время графическая анимация продолжает развиваться и становиться все более реалистичной и интерактивной. С появлением технологий виртуальной и дополненной реальности, а также новых методов анимации, таких как кинематографический и реального времени, покадровая анимация продолжает эволюционировать и исследовать новые возможности, включая разработку анимации с использованием искусственного интеллекта, машинного обучения и алгоритмов генерации контента. Это позволяет создавать более реалистичные, динамические и интерактивные анимационные сцены и обеспечивает широкий спектр возможностей для создания впечатляющих визуальных эффектов.

# **Анализ и оценка существующих программных продуктов**

Существует множество программных продуктов, которые позволяют создавать анимированную графику, например, такие как:

Adobe After Effects (рис. 6) – это мощный программный продукт, разработанный компанией Adobe. Он используется для создания профессиональной анимированной графики, спецэффектов и композитинга. Adobe After Effects имеет широкий спектр инструментов и функций для создания анимации, включая возможность работы со слоями, временную шкалу и различные эффекты. Однако, программа может быть сложной в освоении и требует определенного уровня знаний и опыта. История создания After Effects насчитывает несколько десятилетий и связана с развитием компьютерной графики.

В конце 1980-х годов компания CoSA Digital разрабатывала программу под названием «RenderMorphics», которая предназначалась для создания эффектов и анимации в реальном времени. В 1991 году CoSA Digital выпустила программу под названием «After Effects» для Apple Macintosh. Эта программа была первой в своем роде и предоставляла возможность комбинировать видео и графические элементы, добавлять эффекты и создавать анимацию.

В 1993 году Adobe Systems приобрела CoSA Digital и интегрировала After Effects в свой ассортимент программного обеспечения для дизайна и мультимедиа. С течением времени Adobe продолжала развивать и совершенствовать After Effects, добавляя новые функции и инструменты для создания разнообразных видеоэффектов.

В 2002 году Adobe выпустила Adobe After Effects 5.0, который представил множество новых функций, включая поддержку трехмерной графики и расширенные возможности масок. С каждым последующим выпуском After Effects становился все более мощным и гибким инструментом для создания визуальных эффектов.

Плюсы Adobe After Effects:

* богатый набор функций: After Effects предлагает широкий спектр инструментов и возможностей для создания сложных визуальных эффектов, композиции и анимации. Это включает в себя возможность работы со слоями, масками, трекингом движения, цветокоррекцией, трехмерной графикой и многими другими;
* интеграция с другими программами Adobe: After Effects легко интегрируется с другими программами Adobe, такими как Photoshop и Illustrator. Это дает возможность импортировать графику и создавать комбинированные композиции, а также использовать привычные инструменты и рабочие процессы;
* открытость для сторонних плагинов: After Effects поддерживает установку сторонних плагинов, которые расширяют его функциональность и предлагают новые возможности в области визуальных эффектов и анимации. Это позволяет пользователю настроить программу под свои потребности и сделать работу более эффективной;
* анимационный редактор: After Effects предоставляет удобный анимационный редактор, в котором пользователю доступны различные кривые и параметры для управления движениями и трансформациями объектов. Это позволяет создавать плавные и реалистичные анимации.

Минусы Adobe After Effects:

* высокая стоимость: Абонентская модель и стоимость After Effects могут быть достаточно высокими для отдельных пользователей или небольших студий. Это может быть проблемой для тех, кто только начинает свою работу в сфере визуальных эффектов или имеет ограниченный бюджет;
* сложность обучения: Изучение и освоение After Effects может быть вызовом из-за его множества функций и глубокой конфигурации. Необходимо время и усилия, чтобы овладеть программой и использовать ее на полную мощность;
* требовательность к системным ресурсам: After Effects является достаточно ресурсоемким программным обеспечением, требующим мощного компьютера для плавной и эффективной работы. Это может быть проблемой для пользователей с ограниченными вычислительными мощностями или устаревшим оборудованием;
* время рендеринга: Создание сложных визуальных эффектов и анимации может занимать много времени на рендеринг. Это может стать проблемой при работе с большими проектами или при соблюдении крайних сроков.



Рисунок 6 – Логотип Adobe After Effects

Blender (рис. 7) – это бесплатное и открытое программное обеспечение для создания 3D-графики и анимации. Blender имеет широкий набор инструментов для моделирования, анимации и рендеринга, что делает его идеальным для создания профессиональных анимаций.

История Blender начинается в 1995 году, когда нидерландский программист Тон Розендал (Ton Roosendaal) основал компанию «Not a Number» (NaN) с целью создания свободной, открытой и доступной программы для 3D-моделирования и анимации. В 1998 году «Not a Number» представила первую версию программы Blender, которая была совместима с операционной системой Windows.В 2000 году «Not a Number» столкнулась с финансовыми проблемами и объявила о банкротстве. Однако, вместо закрытия программы, Тон Розендал провел успешную кампанию по сбору средств среди сообщества пользователей Blender, и в 2002 году программа стала свободной и открытой под лицензией GNU General Public License (GPL). Это открытие программы значительно способствовало ее популяризации и развитию. Blender начал привлекать множество талантливых разработчиков и художников со всего мира, которые вносили свой вклад в развитие программы и создавали потрясающие проекты.

С течением времени Blender стал все более мощным и гибким инструментом для 3D-моделирования и анимации. Улучшения и новые возможности, такие как поддержка текстур, редактор анимации, система частиц и другие, были внедрены в различных версиях программы.

В 2009 году Фонд Blender (Blender Foundation) был создан Тоном Розендалом, что позволило организовать разработку Blender как коммерческого и некоммерческого проекта. Фонд Blender успешно собирает средства на основе пожертвований сообщества и спонсоров для поддержания и развития программы.

Плюсы Blender:

* бесплатность и открытый исходный код: Одним из основных преимуществ Blender является то, что он является полностью бесплатным и доступным для всех без необходимости покупки лицензии. Кроме того, Blender имеет открытый исходный код, что позволяет разработчикам и художникам создавать собственные инструменты, плагины и расширения для программы;
* широкий функционал: Blender предлагает обширный набор функций для моделирования, анимации, текстурирования, рендеринга и много другого. Он имеет мощные инструменты и возможности, которые позволяют создавать сложные и реалистичные 3D-сцены и анимацию;
* активное сообщество пользователей: Blender обладает большим и активным сообществом пользователей, которые делают большой вклад в разработку программы, создают обучающие ресурсы и работают над проектами совместно. Это обеспечивает поддержку и обмен опытом между пользователями, а также быструю реакцию на вопросы и проблемы;
* переносимость: Blender совместим с различными операционными системами, включая Windows, macOS и Linux. Это позволяет пользователям работать на платформе, которая наиболее удобна для них и облегчает совместную работу в команде.

Минусы Blender:

* сложность обучения: изучение Blender может быть вызовом из-за его обширного функционала и глубокой настройки. Для полного освоения программы может потребоваться время и усилия, особенно для новичков;
* некоторые функции могут быть нестабильными: поскольку Blender активно разрабатывается и обновляется, некоторые функции или расширения могут быть менее стабильными и могут вызывать проблемы или неожиданные ошибки;
* отсутствие интеграции с другими программами: Blender имеет ограниченную интеграцию с некоторыми популярными программами в индустрии 3D-графики, такими как Autodesk Maya или 3ds Max. Это может затруднить совместную работу с коллегами, которые предпочитают другие программы.



Рисунок 7 – Логотип Blender

Stop Motion Studio (рис. 8) – это простой и удобный программный продукт для создания покадровой анимации. Он идеально подходит для начинающих, так как имеет простой интерфейс и набор инструментов, который позволяет создавать анимацию без необходимости знания сложных программ. Однако, Stop Motion Studio может не иметь достаточного функционала для профессиональных пользователей.

Stop Motion Studio была разработана и выпущена компанией Cateater LLC. В 2010 году разработчики Cateater LLC, лидеры в области создания технологий для мобильных устройств, осознали потенциал стоп-моушн анимации и был создан проект Stop Motion Studio.

Первая версия Stop Motion Studio для iOS была выпущена в 2010 году и получила широкое признание среди пользователей. Программа предоставляла простой интерфейс и интуитивно понятные инструменты для создания стоп-моушн анимации на iPhone и iPad.

Плюсы Stop Motion Studio:

простота использования: Stop Motion Studio предлагает интуитивно понятный интерфейс и простые инструменты, что делает его доступным для широкого круга пользователей. Даже новички без опыта в анимации могут быстро освоить программу и начать создавать свои анимационные проекты;

возможность создания высококачественных анимаций: Stop Motion Studio позволяет создавать стоп-моушн анимацию в высоком разрешении и сохранными картинками, обеспечивая высокое качество визуальных эффектов;

расширенный функционал: программа предлагает различные дополнительные функции и инструменты, такие как масштабирование, редактирование, добавление эффектов и музыки, изменение скорости воспроизведения и другие. Это позволяет пользователям настраивать и улучшать свою анимацию по своему вкусу и потребностям;

интеграция и обмен: Stop Motion Studio позволяет легко делиться своими анимациями с помощью функций экспорта, а также интеграции социальных сетей. Это дает возможность пользователю показать свои творческие работы широкой аудитории.

Минусы Stop Motion Studio:

* ограниченная функциональность: по сравнению с профессиональными программами для стоп-моушн анимации, Stop Motion Studio может иметь ограниченный набор функций и инструментов. Программа может не подходить для более сложных и продвинутых проектов;
* неидеальная стабильность: иногда Stop Motion Studio может сталкиваться с некоторыми проблемами стабильности, такими как вылеты или непредвиденные ошибки. Это может быть неприятно, особенно при работе над длительными проектами;
* ограниченная поддержка на других платформах: Stop Motion Studio вначале была разработана для мобильных устройств и может иметь ограниченную поддержку на других платформах, таких как Windows или Mac.



Рисунок 8 – Логотип Stop Motion Studio

OpenToonz (рис. 9) – программа для создания анимации, которая была разработана на основе Toonz, профессионального программного обеспечения для создания анимации.

Toonz была изначально разработана и выпущена итальянской компанией Digital Video S.p.A. в конце 1990-х годов как коммерческое программное обеспечение для создания анимации. Это была мощная и популярная программа, использовавшаяся в кинопроизводстве, телевидении и анимационной индустрии.

В 2016 году японская компания Dwango Co., Ltd. и студия Studio Ghibli объединились, чтобы создать OpenToonz - свободную и открытую версию программы Toonz. Целью проекта было предоставление мощного инструмента для аниматоров, доступного всем без необходимости покупки лицензии.

Запущенный в апреле 2016 года, OpenToonz предоставил пользователю полный доступ к исходному коду программы. Это открыло возможность для разработчиков по всему миру улучшать программу, вносить изменения и предлагать новые функции, чтобы удовлетворить нужды сообщества.

Плюсы OpenToonz:

* бесплатность и открытый исходный код: OpenToonz предоставляет полностью бесплатный доступ, что делает его доступным для широкой аудитории аниматоров. Кроме того, поскольку это проект с открытым исходным кодом, пользователи и разработчики могут вносить изменения и дополнять функциональность программы в соответствии со своими потребностями;
* богатый набор функций: OpenToonz предлагает широкий спектр инструментов и возможностей для создания анимации. Он включает в себя функции рисования, анимации, колоризации, трекинга движений, эффектов и многое другое. Благодаря этому пользователи могут создавать разнообразные виды анимации со сложными и качественными эффектами;
* гибкость и настраиваемость: OpenToonz позволяет пользователям настраивать и адаптировать программу под свои потребности. Они могут создавать собственные инструменты, шаблоны и расширения или использовать плагины сторонних разработчиков;
* продвинутый рендеринг и комбинирование 2D и 3D анимации: OpenToonz предлагает возможность комбинирования 2D и 3D элементов в рамках одного проекта анимации. Это дает больше гибкости для создания разнообразных сцен и эффектов.

Минусы OpenToonz:

* сложность обучения: OpenToonz может быть сложным для новичков и тех, кто только начинает свой путь в анимации. Программа имеет множество функций и инструментов, и пользователи могут потребоваться время и усилия для изучения и освоения всех возможностей программы;
* отсутствие широкой поддержки и руководства пользователя: Поскольку OpenToonz является программным обеспечением с открытым исходным кодом, иногда может быть сложно найти обширные руководства и качественную поддержку по использованию программы. Однако есть сообщество пользователей, которые обмениваются опытом и предоставляют ресурсы для помощи другим пользователям;
* возможны ошибки и нестабильность: В некоторых случаях OpenToonz может иметь некоторые проблемы с нестабильностью и ошибками, особенно при работе с сложными проектами или использовании дополнительных функций и плагинов.

# **Постановка задачи**

# **1.3.1 Цель разработки**

Целью разработки программного продукта является создание удобного и простого инструмента для создания покадровой анимации с возможностью просмотра уроков по созданию первой анимации.

# **1.3.2 Задачи разработки программного продукта**

Анализируя популярные программы для создания покадровой анимации, был сделан вывод что пользователям нужен удобный, интуитивно понятный и главное простой редактор для создания покадровой анимации, который будет не переполнен ненужными функциями.

Таким образом в ходе разработки были поставлены следующие задачи:

* реализовать интуитивно понятный пользовательский интерфейс;
* реализовать библиотеку графических элементов и эффектов;
* обеспечит возможность создания и редактирования анимации;
* реализовать экспорт готовой анимации в различные форматы;
* предоставить просмотр обучающих видеороликов.

Аудитория назначения: начинающие пользователи, которые только начинают изучать основы анимации.

Планируемый функционал:

* создание и редактирование анимированных объектов;
* управление анимацией: скорость, направление, длительность;
* применение различных эффектов (наложение слоёв, прозрачность и т.д.);
* экспорт анимации в GIF.

# **ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

# **Моделирование программного обеспечения**

Для моделирования программы по созданию покадровой анимации были выбраны следующие диаграммы: диаграмма прецедентов и IDEF. Данный выбор обеспечивает полное раскрытие различных аспектов проекта.

# **Диаграмма прецедентов**

Диаграмма вариантов использования в UML (рис. 9) – это схема, описывающая связи между участниками (актёрами) и сценариями, которая является составной частью модели сценариев и помогает описать систему на концептуальном уровне. В случае программы, диаграмма прецедентов полностью раскрывает функциональные возможности пользователя, такие как создание рисунка и анимации, выбора нужных инструментов и т. д. Диаграмма прецедентов позволяет визуализировать сценарии использования системы и помогает в понимании функциональных возможностей программы по созданию покадровой анимации.

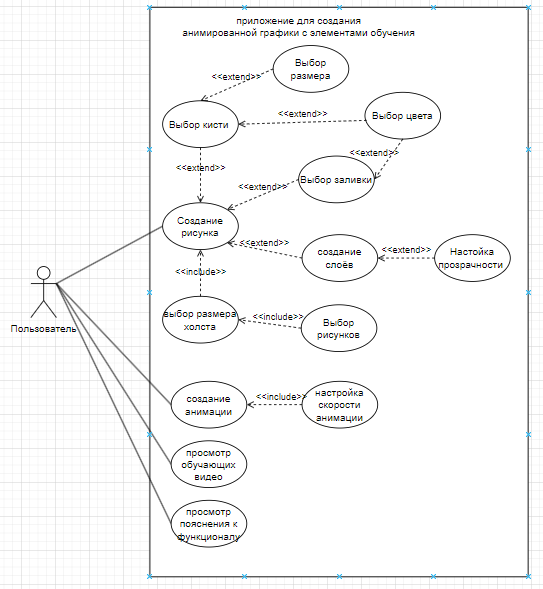


Рисунок 9 – Диаграмма прецедентов

# **Функциональная модель**

IDEF0 (Functional Flow Block Diagram) — это метод структурного анализа и моделирования, который используется для создания функциональной модели, помогающей организациям определить требования к информационной системе. Он представляет собой графическое описание функций системы, информации и документов, а также потоков материалов и средств между ними. IDEF0 может быть использован для анализа, проектирования и разработки систем, а также для улучшения существующих систем.

IDEF0 разделяется на контекстную диаграмму и диаграмму декомпозиции первого уровня.

Контекстная диаграмма (рис. 10) — это диаграмма в методологии IDEF0, которая показывает взаимодействие системы с окружающей средой. Она включает в себя один функциональный блок, который представляет систему, и стрелки, которые изображают входы, выходы, управления и механизмы. Контекстная диаграмма помогает понять границы системы и ее основные функции.

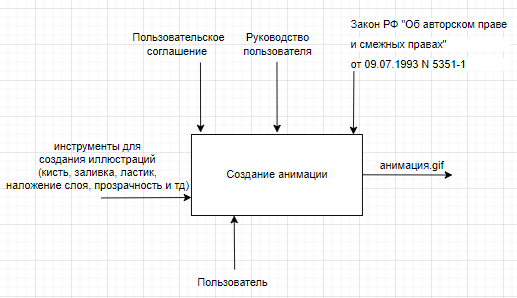


Рисунок 10 - Контекстная диаграмма IDEF0

Диаграмма декомпозиции первого уровня (рис. 11) — это диаграмма в методологии IDEF0, которая разбивает систему на основные подсистемы или функции. Она включает в себя несколько функциональных блоков, которые представляют подсистемы, и связи между ними. Диаграмма декомпозиции первого уровня помогает понять структуру системы и основные процессы, которые происходят в ней.

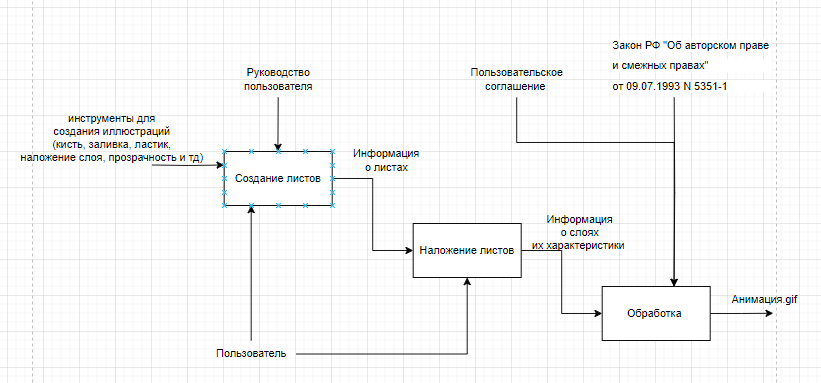


Рисунок 11 - Диаграмма декомпозиции первого уровня IDEF0

# **Обоснование выбора средств разработки**

При выборе Microsoft Visual Studio для разработки десктопного приложения, следует обосновать это решение, учитывая характеристики разрабатываемого программного обеспечения, организационно-технические условия его разработки и параметры выбранного инструмента. Важно также рассмотреть проблемную область, в которой будет использоваться разрабатываемое программное обеспечение.

Microsoft Visual Studio – это интегрированная среда разработки (Integrated Development Environment, IDE) от компании Microsoft, предназначенная для создания различных типов программного обеспечения. Она предоставляет разработчикам широкий набор инструментов и возможностей для создания, отладки, тестирования и развертывания приложений на различных платформах.

Visual Studio поддерживает несколько языков программирования, включая C++, C#, Visual Basic, F#, Python и другие. Он предлагает различные шаблоны проектов для разработки настольных, веб- и мобильных приложений, а также возможности для создания игр, баз данных, облачных приложений и других типов программного обеспечения.

Среда Visual Studio предлагает функции, облегчающие процесс разработки, такие как автозаполнение кода, навигация по проекту, инструменты рефакторинга, отладка с поддержкой точек останова, анализ кода, система контроля версий и другие. Она также интегрируется с другими продуктами и инструментами Microsoft, такими как Azure, SQL Server, Office и Team Foundation Server, облегчая взаимодействие и интеграцию приложений.

Visual Studio имеет обширное сообщество разработчиков и доступ к множеству ресурсов, таких как документация, учебные материалы, форумы и расширения, которые помогают разработчикам улучшать свои навыки и решать проблемы.

Среда разработки Microsoft Visual Studio является мощным инструментом для разработки разнообразных приложений. Она обладает широкими функциональными возможностями, гибкостью и широкой поддержкой со стороны Microsoft, что делает ее популярным выбором для многих разработчиков по всему миру.

Выбор средства разработки на Microsoft Visual Studio WPF C# может быть обоснован следующими факторами:

* широкие возможности разработки: Microsoft Visual Studio предоставляет обширный набор инструментов и библиотек для разработки приложений, включая Windows Presentation Foundation (WPF) для создания графического интерфейса пользователя. WPF позволяет создавать богатые, эффективные и удобные интерфейсы для приложений на платформе Windows;
* продуктивная среда разработки: Visual Studio обладает мощной интегрированной средой разработки (IDE), которая предоставляет разработчикам широкий набор инструментов для ускорения и упрощения процесса разработки. Это включает функции автозаполнения кода, отладки, инструменты рефакторинга и многое другое;
* широкое сообщество разработчиков: разработка на Microsoft Visual Studio и использование C# и WPF предлагает доступ к большому сообществу разработчиков. Это означает, что есть множество ресурсов, форумов, документации и обучающих видео, которые помогут вам изучать и решать проблемы в процессе разработки;
* интеграция с другими продуктами Microsoft: Microsoft Visual Studio плотно интегрирован с другими продуктами Microsoft, такими как Azure, SQL Server, Office и многие другие. Это облегчает интеграцию приложения с другими системами и расширение его функциональности;
* широкая поддержка и длительный срок службы: Visual Studio и язык программирования C# имеют широкую поддержку и аллокацию ресурсов со стороны Microsoft. Обновления и исправления ошибок регулярно выпускаются, и Microsoft продолжает активно развивать эти инструменты, обеспечивая долговечность и стабильность среды разработки.

# **Разработка базы данных**

База данных – это организованная и структурированная коллекция связанных данных, которые хранятся и управляются с целью эффективного доступа, модификации и использования. Она представляет собой структурированное хранилище, в котором информация хранится в упорядоченном и систематизированном виде для обеспечения эффективной обработки данных. Приложение по созданию покадровой анимации взаимодействует с Microsoft SQL Server.

В базе данных находится таблица TrainingVideos (рис. 12). Таблица хранит в себе видеоролик, описание в виде текста, и состояние (пройдено/не пройдено).

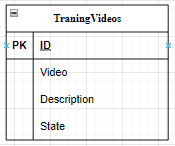


Рисунок 12 – таблица TrainingVideos

# **Проектирование интерфейса программного обеспечения**

Интерфейс приложения – это внешний вид и функциональность, через которые пользователь взаимодействует с приложением. Он предоставляет пользователю средства для ввода и вывода информации, а также управления функциями и операциями приложения. Интерфейс приложения включает в себя элементы дизайна, такие как кнопки, поля ввода, меню, окна и другие компоненты, а также логику взаимодействия с этими элементами.

При разработке интерфейса было выбрано следующее цветовое решение интерфейса:

* светло-серый и серый. Данные оттенки серого обеспечивают сбалансированный и нейтральный фон, который не отвлекает пользователя от содержания интерфейса и позволяет акцентировать внимание на других элементах.
* белый. Данный цвет используется для выделения важных элементов и привлечения внимания пользователя. Например, кнопки, текст и ссылки

Было выбрано данное цветовое решение так как светло-серый и серый цвет имеет ряд преимуществ для интерфейсов. Во-первых, он обеспечивает хорошую контрастность с другими цветами, что позволяет легко читать текст и различать элементы интерфейса. Во-вторых, светло-серый или серый цвет обычно считается спокойным и профессиональным, создавая впечатление надежности и ненавязчивости. Кроме того, серый можно комбинировать с другими цветами, чтобы создавать контрастные акценты или подчеркивать определенные элементы интерфейса.

Проект включает несколько страниц, каждая из которых выполняет определенную функцию (рис. 13).

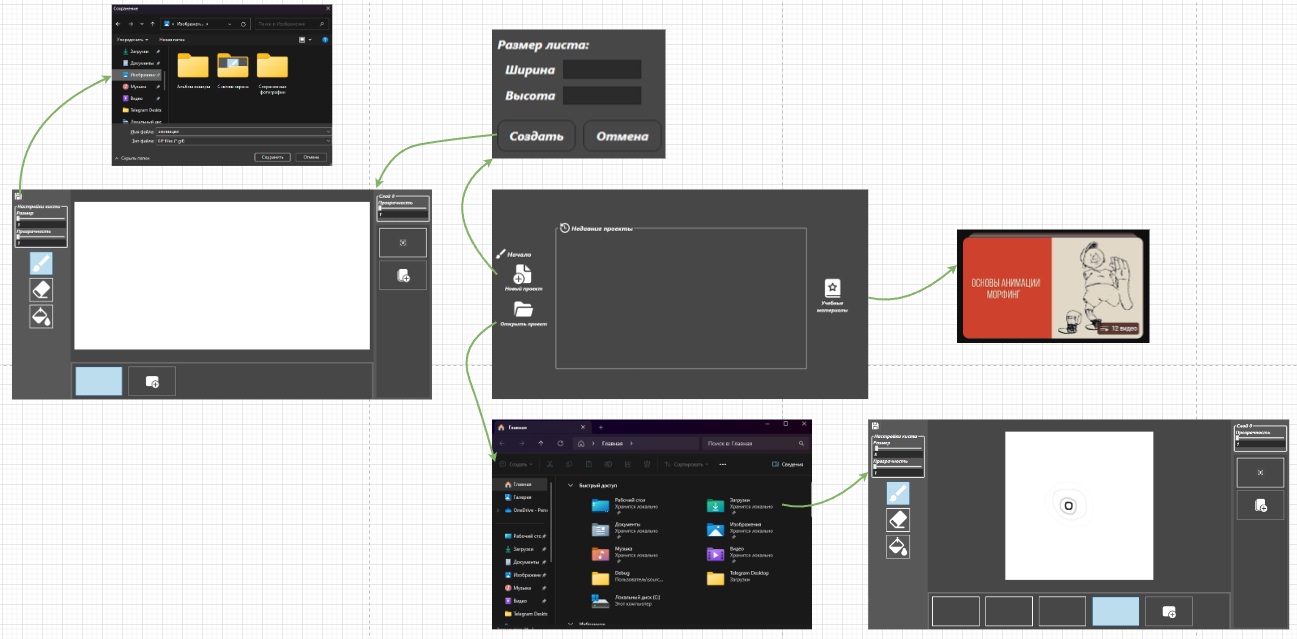


Рисунок 13 – Проектирование интерфейса

Рассмотрим подробнее каждое из окон:

На главном окне (рис. 14) пользователь может создать новый проект, открыть проекты, открыть недавние проекты или просмотреть учебные материалы. Чтобы создать новый проект нужно нажать на кнопку Новый проект, после этого откроется настройка листа для нового проекта. Если же нужно открыть существующий проект необходимо нажать на кнопку Открыть проект, после этого откроется проводник, где хранятся проекты. А нажав на кнопку Учебные материалы откроется окно с учебными материалами.



Рисунок 14 – Главное окно

После нажатия на кнопку создать проект откроется окно настроек листа (рис. 15), в котором необходимо указать высоту и ширину будущего листа. В поле ширина и высота можно указать размер от 144 до 3840. Также на окне настройки листа находятся две кнопки Создать и Отменить. При корректном вводе размера листа и последующем нажатии на кнопку Создать откроется окно проекта, а предыдущие окна закроются. Если же нажать на кнопку отменить, то окно настроек листа просто закроется.

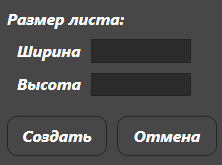


Рисунок 15 – Окно настроек листа

После нажатия на кнопку Создать откроется окно проекта (рис. 16). На этом окне в левой части расположены кисти ластик заливка и их настройка в виде прозрачности и толщены, палитры, а также в верхнем углу кнопка настройки проекта. В правой части экрана расположены слои листа, где можно поменять прозрачность, видимость или добавить новый слой. В нижней части экрана располагаются листы, и кнопка добавить лист. По центру находится лист, на котором можно рисовать. Чтобы сохранить проект необходимо нажать на шестерёнку в левом верхнем углу. Откроется меню с расстановкой времени воспроизведения каждого листа и необходимо будет ввести название проекта для сохранения проекта.

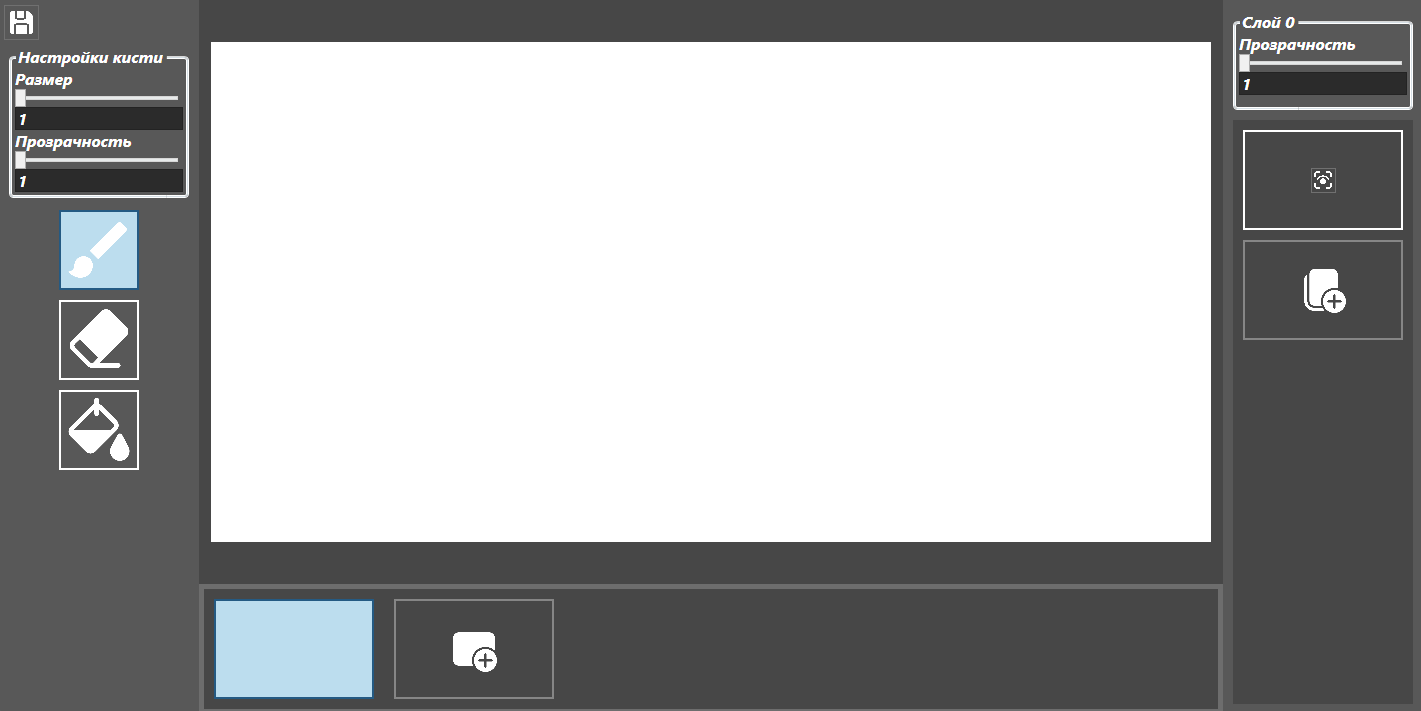


Рисунок 16 – Окно проекта

# **2.5 Спецификация программы**

Для разработки приложения используется Microsoft Visual Studio, для БД используется Microsoft SQL Server. В приложение есть Окна xaml, которые отвечают за разный функционал приложения.

Рассмотрим подробнее содержание проекта (рис. 17):

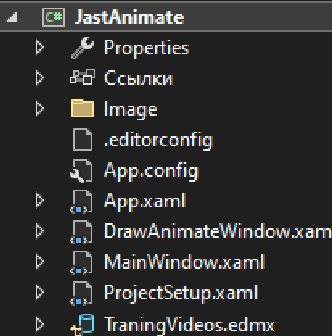


Рисунок 17 – Содержание проекта

– папка Image (рис. 18). Она содержит в себе все изображения, используемые в проекте;

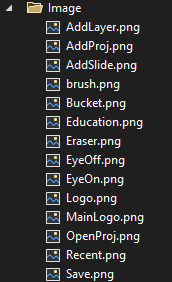


Рисунок 18 – Содержание папки Image

– файл MainWindow.xaml (рис. 19). Он содержит в себе функционал главной страницы и её содержание страниц. С помощью этого файла можно создать проект, открыть прошлый проект или открыть страницу с уроками;



Рисунок 19 – Содержание файла MainWindow.xaml

– файл ProjectSetup.xaml (рис. 20). Он содержит настройки создаваемого листа, то есть с помощью него устанавливается ширина и высота листа;



Рисунок 20 – Содержание файла MainWindow.xaml

– файл DrawAnimateWindow.xaml (рис. 21). Он содержит в себе интерфейс и функционал проекта. С помощью него можно рисовать, менять настройки листа и кисти, редактировать и сохранять анимацию.



Рисунок 20 – Содержание файла DrawAnimateWindow.xaml

# **3 ОПИСАНИЕ РАБОТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

# **3.1 Руководство пользователя**

Руководство пользователя представлено в Приложении А.

# **3.2 Руководство программисту**

Руководство программисту представлено в Приложении Б.

# **ЗАКЛЮЧЕНИЕ**

В рамках данного курсового проекта была разработана программа для покадровой анимации на платформе WPF с использованием языка программирования C#. В результате проделанной работы были достигнуты следующие цели:

– реализация создания анимации: был разработан алгоритм, позволяющий объединить пять картинок с листов InkCanvas в анимацию формата GIF. Для этого использовались методы из библиотеки AnimatedGIF, которые позволяют работать с изображениями и создавать анимацию;

– визуализация анимации: с использованием платформы WPF и языка C# был создан пользовательский интерфейс, который позволяет выбрать пять картинок и просмотреть анимацию в режиме воспроизведения;

– надежность и эффективность: в процессе работы над проектом был уделен особый акцент на обработку ошибок и оптимизацию кода, чтобы обеспечить стабильную работу программы и эффективное использование ресурсов.

Проделанная работа по созданию программы для покадровой анимации на WPF с использованием C# позволила достичь поставленных целей и получить следующие результаты:

– создана программа, которая позволяет объединять пять картинок с листов InkCanvas в анимацию формата GIF;

– реализован пользовательский интерфейс, который обеспечивает удобный просмотр и управление анимацией;

– обработаны возможные ошибки и осуществлена оптимизация кода.

В ходе работы над проектом был получен ценный опыт в области программирования на WPF и C#, а также приобретены навыки работы с библиотекой AnimatedGIF для создания и работы с анимациями. Этот опыт может быть полезен для разработки иных проектов, связанных с анимацией и визуализацией данных.

В целом, курсовой проект по созданию программы для покадровой анимации на WPF с использованием C# является успешным. Полученные результаты соответствуют поставленным целям, а программа имеет потенциал для дальнейшего развития и расширения функциональности.

Для улучшения программного продукта можно добавить новые функции, такие как поддержка 3D-графики, возможность создания интерактивных анимаций и расширенные возможности по работе с текстом и шрифтами.

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Freeman, E., & Freeman, E. (2013). WPF 4.5 Unleashed. Sams Publishing. [дата обращения: 15.09.2023].

2. MacDonald, M., & Szpuszta, M. (2010). Pro WPF in C# 2010: Windows Presentation Foundation in .NET 4. Apress. [дата обращения: 29.09.2023].

3. Petzold, C. (2006). Applications = Code + Markup: A Guide to the Microsoft Windows Presentation Foundation. Microsoft Press. [дата обращения: 13.12.2023].

4. Eden, D. (2008). WPF Control Development Unleashed: Building Advanced User Experiences. Sams Publishing. [дата обращения: 20.12.2023].

5. Loshin, P. (2012). Windows Presentation Foundation Development Cookbook. Packt Publishing. [дата обращения: 04.01.2024].

6. Franklin, J. (2017). Mastering Windows Presentation Foundation. Packt Publishing. [дата обращения: 06.01.2024].

7. Sells, C. (2007). Programming WPF. O'Reilly Media. [дата обращения: 13.06.2023].

8. Gerhardt, C. (2008). Windows Presentation Foundation Recipes in C# 2008: A Problem-Solution Approach. Apress. [дата обращения: 13.06.2023].

9. Card, S., MacKinlay, J., & Shneiderman, B. (1999). Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann. [дата обращения: 13.06.2023].

10. Clark, A. (2011). Moving to WPF: Windows Presentation Foundation and XAML. Apress. [дата обращения: 13.06.2023].

# **ПРИЛОЖЕНИЕ А**

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Магнитогорский государственный технический университет

им. Г. И. Носова»

Многопрофильный колледж

Отделение №2 «Информационные технологии и транспорт»

УТВЕРЖДАЮ

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_ Тутарова В. Д.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023

Программа «Just animate»

на основе WPF

Руководство пользователя

Лист утверждения

А.В.00001-01 33 01-ЛУ

Руководитель разработки

Руководитель

\_\_\_\_\_\_\_\_\_\_\_ Тутарова В. Д.

«\_\_\_\_\_»

\_\_\_\_\_\_\_\_\_\_\_\_2023

2023

|  |  |
| --- | --- |
| УТВЕРЖДЕН  А.В.00001-01 33 01-ЛУ |  |

Программа «Just animate»

на основе WPF

Руководство пользователя

5177746378431.05.11.01.001

Листов 9

2023

# **АННОТАЦИЯ**

Настоящий документ является руководством пользователя Программа «Just animate», созданного при помощи Microsoft Visual Studio WPF (.NET Framework) на языке c#.

В документе приведены сведения о назначении и условиях применения приложения, подготовительных действиях и операциях, которые выполняет пользователь при работе с программой.

Документ разработан в соответствии с требованиями следующих документов:

− ГОСТ Р 59795-2021 «Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов»;

− ГОСТ Р 59853-2021 «Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения»;

− ГОСТ 2.105-95 «Единая система конструкторской документации. Общие требования к текстовым документам».

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 37](#_heading=h.4k668n3)

[1. НАЗНАЧЕНИЕ И УСЛОВИЕ ПРИМЕНЕНИЯ 38](#_heading=h.2zbgiuw)

[1.1. Виды деятельности, функции, для автоматизации которых предназначено данное средство автоматизации 38](#_heading=h.1egqt2p)

[1.2. Условия, при соблюдении которых обеспечивается применение средства автоматизации в соответствии с назначением. 38](#_heading=h.3ygebqi)

[1.3. Требования к программному обеспечению 38](#_heading=h.2dlolyb)

[1.4. Требования к техническому обеспечению 38](#_heading=h.sqyw64)

[2. ПОДГОТОВКА К РАБОТЕ И ПРОВЕРКА РАБОТОСПОСОБНОСТИ 39](#_heading=h.3cqmetx)

[3. ОПИСАНИЕ ОПЕРАЦИЙ 40](#_heading=h.1rvwp1q)

[4. АВАРИЙНЫЕ СИТУАЦИИ 45](#_heading=h.4bvk7pj)

# **ВВЕДЕНИЕ**

Область применения

Программа по созданию покадровой анимации предназначено в первую очередь для начинающих пользователей в области покадровой анимации, с целью создания покадровой анимации и обучению с помощью видеороликов. Однако программа подходит для всех пользователей.

Краткое описание возможностей

Программа предоставляет набор инструментов и функций для создания простейшей покадровой анимации, а также просмотр обучающих материалов по созданию анимации. что помогает пользователям без опыта создать свою первую анимацию.

Уровень подготовки пользователя

Программа по созданию покадровой анимации доступен для пользователей всех уровней подготовки, от начинающих до опытных пользователей. Функционал программы является интуитивно понятным и легко доступным для использования.

Перечень эксплуатационной документации:

* руководство пользователя – данное руководство предоставляет подробные инструкции по использованию приложения и его функциям;
* политика конфиденциальности – в данном документе описываются правила обработки и хранения пользовательских данных и информации, собираемых при использовании приложения.

# **1. НАЗНАЧЕНИЕ И УСЛОВИЕ ПРИМЕНЕНИЯ**

# **1.1. Виды деятельности, функции, для автоматизации которых предназначено данное средство автоматизации**

Виды деятельности, функции, для автоматизации которых предназначено данное средство:

* создание и редактирование анимаций: программное средство позволяет пользователям создавать и настраивать различные графические элементы, такие как векторные фигуры, текст, изображения, а также анимировать их движение и трансформацию;
* управление слоями: пользователи могут работать со сложными многослойными композициями, управляя порядком и прозрачностью слоев, а также добавлять и редактировать отдельные элементы на каждом слое;
* обучение: продукт предлагает обучающие материалы, форумы помощи пользователям в освоении программного обеспечения и решении возникающих вопросов.

# **1.2. Условия, при соблюдении которых обеспечивается применение средства автоматизации в соответствии с назначением.**

Чтобы использовать все возможности приложения, необходимо установить его на совместимое устройство и убедиться, что операционная система соответствует системным требованиям программы. Пользователь должен использовать функциональность приложения в соответствии с ее назначением и предназначением. Таким образом, гарантируется корректна работа и хорошая производительность приложения.

# **1.3. Требования к программному обеспечению**

Для использования программы необходим компьютер с операционной системой Windows. Для получения наилучшего опыта рекомендуется обновлять операционную систему и приложение до последних версий.

# **1.4. Требования к техническому обеспечению**

Для работы с приложением должны удовлетворяться минимальные требования к конфигурации аппаратного обеспечения:

* ОС Windows 7 или более новая;
* 8-ядерный процессор;
* оперативная память 2 ГБ и выше;
* свободное место накопителя не менее 500 Мб.

# **2. ПОДГОТОВКА К РАБОТЕ И ПРОВЕРКА РАБОТОСПОСОБНОСТИ**

Перед началом работы с программой «Just animate» необходимо убедиться, что устройство соответствует минимальным требованиям. При запуске программы на устройстве следует проверить его работоспособность и убедиться, что оно функционирует без проблем. Рекомендуется проверить следующие аспекты:

* минимальные системные требования. Необходимо убедится, что устройство соответствует системным требованиям программного продукта. Это может включать операционную систему, процессор, память и графические возможности;
* запуск приложения. Нужно запустить программу и убедитесь, что она запускается без ошибок и проблем;
* проверка функциональности. Проверка основных функций программного продукта, таких как создание и редактирование анимации, добавление эффектов, импорт и экспорт файлов;
* проверка работоспособности обучающих видео, что они корректно отображаются.

Проверка работоспособности программы «Just animate» перед его использованием позволит быть уверенными, что оно работает должным образом и будет соответствовать потребностям при создании анимации.

# **3. ОПИСАНИЕ ОПЕРАЦИЙ**

Для того, чтобы запустить программу, необходимо нажать по ярлыку, который находится на рабочем столе или в меню пуск, с названием «Just animate» (рис. 16).



Рисунок 16 – Ярлык программы «Just animate»

После входа в систему открывается главное меню (рис. 17). Оно содержит кнопки для создания новых проектов, открытия проектов, а также просмотра обучающих материалов.



Рисунок 17 – Окно главное меню

При нажатии на кнопку новый проект откроется окно настроек листа (рис. 18), в котором находятся поля, в которых необходимо указать высоту и ширину будущего листа и далее нажать на кнопку создать.

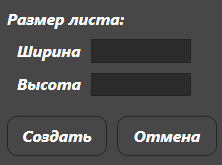


Рисунок 18 – Окно настроек листа

После нажатия на кнопку Создать откроется окно проекта (рис. 19).

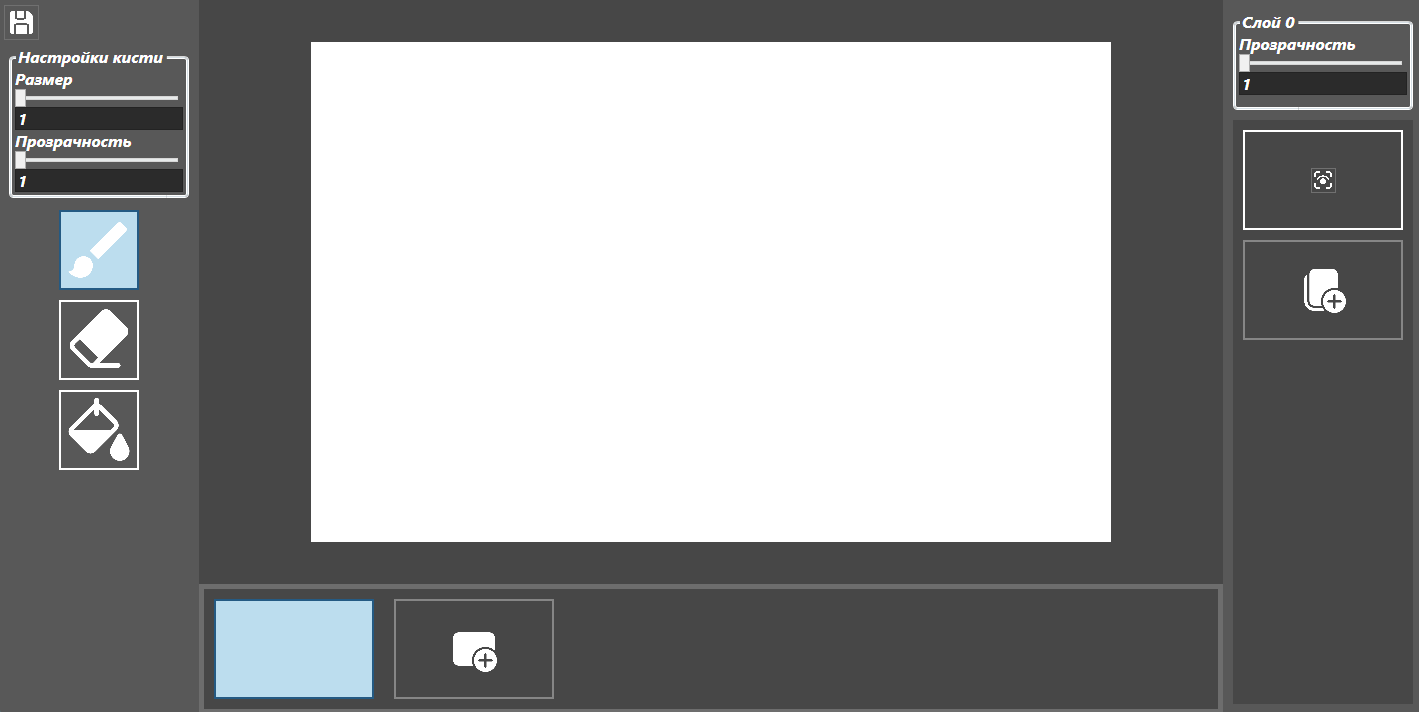


Рисунок 19– Окно проекта

На этом окне в левой части расположена панель инструментов (рис 20), на ней находятся кисти, ластик, заливка и их настройка в виде прозрачности и толщены, а также в верхнем углу кнопка сохранить проект проекта.

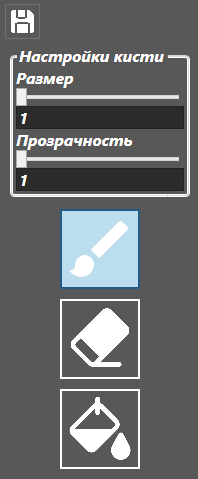


Рисунок 20 – Панель инструментов

В правой части экрана расположены настройки листа (рис 21), на котором находятся слои листа, где можно поменять прозрачность, видимость или добавить новый слой.

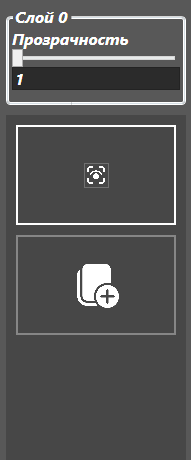


Рисунок 21 – Настройки листа

В нижней части экрана располагаются листы, и кнопка добавить лист (рис 22).

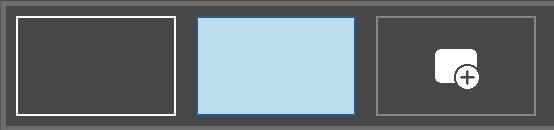


Рисунок 21 – Панель с листами

По центру находится лист (рис. 22), на котором можно рисовать.

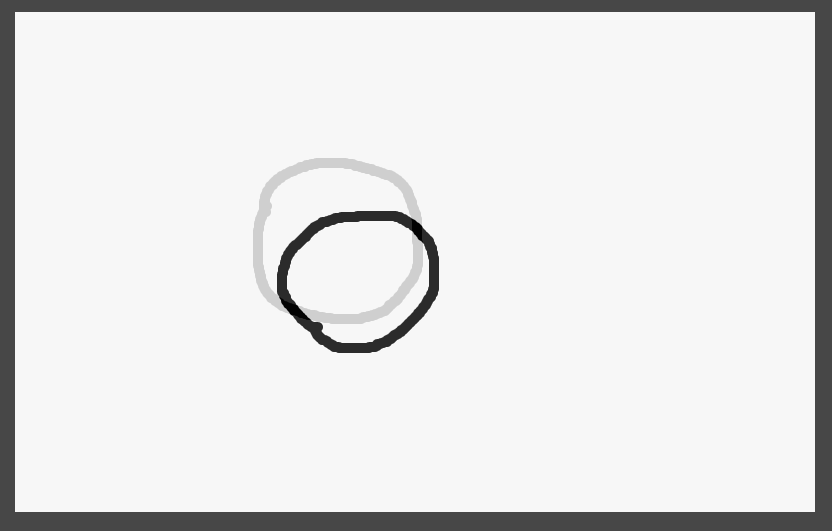


Рисунок 22 – Лист проекта

Чтобы сохранить проект необходимо нажать на кнопку сохранить проект (рис. 23).



Рисунок 23 – Кнопка сохранить проект

Откроется проводник, в котором необходимо будет ввести название проект (рис. 24).

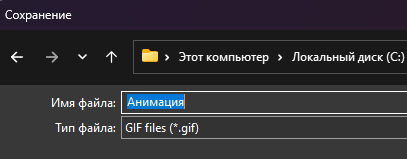


Рисунок 24 – Сохранение проекта

# **4. АВАРИЙНЫЕ СИТУАЦИИ**

Действия в случае несоблюдения условий выполнения технологического процесса, в том числе при длительных отказах технических средств:

– перезапустить программу и/или компьютер, чтобы устранить временные сбои;

– проверить подключение и работоспособность необходимых технических средств, таких как графический планшет, мышь, клавиатура и т.д;

– обратиться к документации к программе или поставщику технических средств, чтобы получить инструкции по устранению поломки или замене неисправного оборудования.

Действия по восстановлению программ и/или данных при отказе носителей данных или обнаружении ошибок в данных:

– если есть резервная копия данных, использовать ее для восстановления информации;

– если нет резервной копии, обратится к специалистам по восстановлению данных, которые могут помочь восстановить утраченные файлы;

– если ошибка возникла во время сохранения проекта, попробовать сохранить его под другим именем или в другом формате;

– если ошибка связана с отдельными элементами проекта, попробовать удалить их или воссоздать;

Действия в случаях обнаружения несанкционированного доступа к данным:

– проверить журналы системы или логи программы для выявления подозрительной активности;

– сообщить о случившемся ответственным лицам или IT-службе вашей организации.

Действия при возникновении других аварийных ситуаций, не описанных выше, рекомендуется обратиться к документации программы или поставщику технической поддержки:

– проверить обновления программы и установите последние исправления, если они доступны;

– перезапустить программу или перезагрузить компьютер, чтобы исправить временные сбои.

В случае возникновения аварийной ситуации, не рекомендуется пытаться самостоятельно вносить изменения в программу или ее настройки, так как это может усугубить проблему. Вместо этого, описанные действия помогут минимизировать последствия и найти решение с помощью специалистов.

# **ПРИЛОЖЕНИЕ Б**

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Магнитогорский государственный технический университет

им. Г. И. Носова»

Многопрофильный колледж

Отделение №2 «Информационные технологии и транспорт»

УТВЕРЖДАЮ

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_ Тутарова В. Д.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023

Программа «Just animate»

на основе WPF

Руководство системного программиста

ЛИСТ УТВЕРЖДЕНИЯ

17933380.425180.10002647.32-ЛУ

Руководитель

\_\_\_\_\_\_\_\_\_\_\_ Тутарова В. Д.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023

2023

|  |  |
| --- | --- |
| УТВЕРЖДЕН  17933380.425180.10002647.32-ЛУ |  |

Программа «Just animate»

на основе WPF

Руководство системного программиста

17933380.425180.10002647.32

Листов 9

2023

**СОДЕРЖАНИЕ**

[1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ 49](#_heading=h.3whwml4)

[1.1 Функциональная архитектура программы 49](#_heading=h.2bn6wsx)

[1.2 Архитектура приложения 49](#_heading=h.1664s55)

[1.3 Требования к программному обеспечению 53](#_heading=h.3q5sasy)

[1.4 Требования к техническому обеспечению 54](#_heading=h.25b2l0r)

[2. СТРУКТУРА ПРОГРАММЫ 55](#_heading=h.3as4poj)

[3. НАСТРОЙКА ПРОГРАММЫ 56](#_heading=h.1pxezwc)

[3.1. Состав и содержание дистрибутивного носителя данных 56](#_heading=h.49x2ik5)

[3.2 Порядок загрузки программы 56](#_heading=h.2p2csry)

[4. ПРОВЕРКА ПРОГРАММЫ 57](#_heading=h.kgcv8k)

[5. СООБЩЕНИЯ 58](#_heading=h.3o7alnk)

# **1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ**

# **1.1 Функциональная архитектура программы**

Функциональная архитектура определяет организацию и взаимосвязи между различными модулями и компонентами программы «Just animate», которые обеспечивают его функциональность. Основные компоненты функциональной архитектуры приложения:

– интерфейс пользователя: это основная часть программы, которая предоставляет пользователю удобный и интуитивно понятный интерфейс для создания и редактирования анимации. Интерфейс состоит из различных окон, элементов управления и панелей инструментов, которые позволяют пользователю взаимодействовать с программой;

– модуль управления: модуль управления отвечает за обработку пользовательских действий, управление анимацией и обмен данными с другими модулями программы. Он содержит логику и алгоритмы, необходимые для выполнения задач, таких как создание, удаление, редактирование и визуализация анимации;

– модуль работы с данными: модуль работы с данными отвечает за сохранение и загрузку созданных анимации в файлы. Он обеспечивает возможность сохранять проекты анимации в определенном формате и восстанавливать их для дальнейшей работы. Этот модуль также может предоставлять функции импорта и экспорта анимации для обмена данными с другими программами.

– модуль отображения: модуль отображения отвечает за визуализацию созданных анимации в режиме реального времени. Он может использовать возможности графического процессора и библиотеки WPF для обеспечения плавной и качественной визуализации анимации на экране. Включает инструменты для управления отображением элементов анимации, их позицией, размером и прозрачностью.

# **1.2. Архитектура приложения**

Для того, чтобы запустить программу, необходимо нажать по ярлыку, который находится на рабочем столе или в меню пуск, с названием «Just animate» (рис. 25).



Рисунок 25 – Ярлык программы «Just animate»

После входа в систему открывается главное меню (рис.26). Оно содержит кнопки для создания новых проектов, открытия проектов, а также просмотра обучающих материалов.



Рисунок 26 – Окно главное меню

Для создания проекта нужно нажать на кнопку новый проект, откроется окно настроек листа (рис. 27), в котором необходимо указать высоту и ширину будущего листа и далее нажать на кнопку создать.

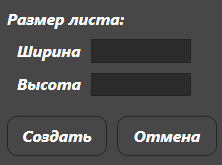


Рисунок 27 – Окно настроек листа

После нажатия на кнопку Создать откроется окно проекта (рис. 28).

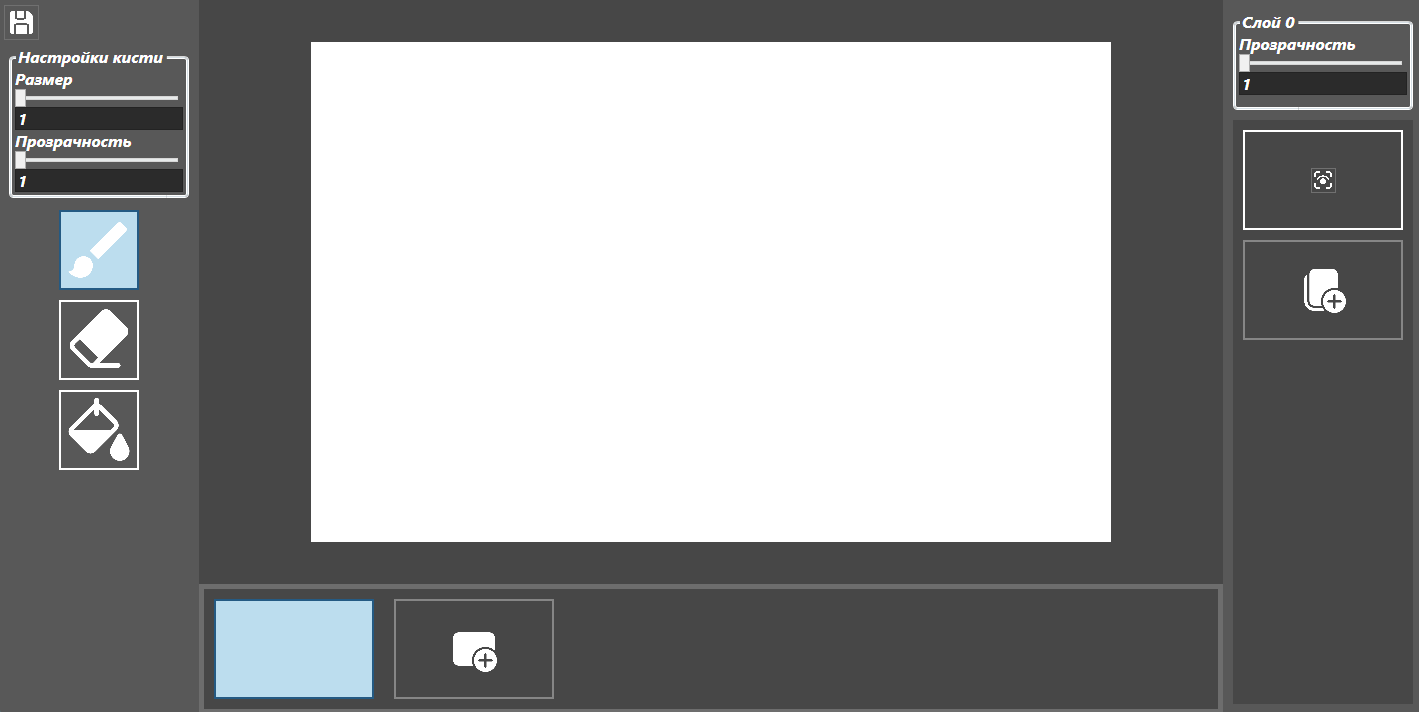


Рисунок 28– Окно проекта

На этом окне в левой части расположена панель инструментов (рис 29), на ней находятся кисти, ластик, заливка и их настройка в виде прозрачности и толщены. Чтобы рисовать нужно нажать на кнопку кисть, если же нарисованное нужно убрать необходимо нажать на ластик и провести курсором по нарисованной линии.

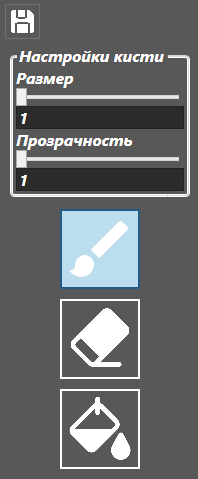


Рисунок 29 – Панель инструментов

В правой части экрана расположены настройки листа (рис 30), на котором находятся слои листа. Чтобы добавить слой на текущий лист необходимо нажать на кнопку добавить новый слой. Если же нужно скрыть какой-либо из слоёв необходимо нажать на кнопку скрыть слой.

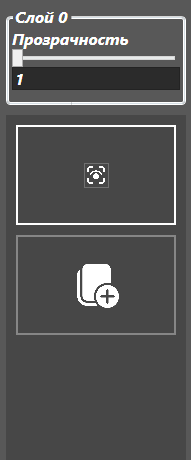


Рисунок 30 – Настройки листа

В нижней части экрана располагаются листы, нужно нажать на кнопку добавить лист, чтобы начать рисовать на другом листе (рис 31).

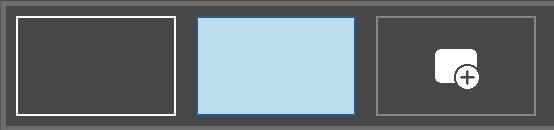


Рисунок 31 – Панель с листами

По центру находится сам лист (рис. 32), на котором можно рисовать.

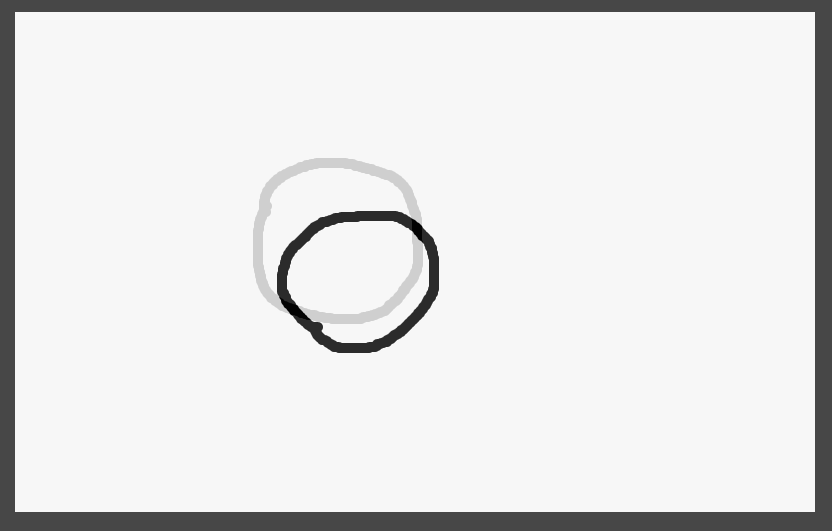


Рисунок 33 – Лист проекта

Чтобы сохранить проект необходимо нажать на кнопку сохранить проект (рис. 34).



Рисунок 34 – Кнопка сохранить проект

В открывшемся проводнике, необходимо ввести название проект и тип файла (рис. 35).

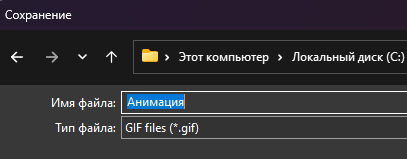


Рисунок 35 – Сохранение проекта

# **1.3. Требования к программному обеспечению**

Для использования программы необходим компьютер с операционной системой Windows. Для получения наилучшего опыта рекомендуется обновлять операционную систему и приложение до последних версий.

# **1.4. Требования к техническому обеспечению**

Для работы с приложением должны удовлетворяться минимальные требования к конфигурации аппаратного обеспечения:

* ОС Windows 7 или более новая;
* 8-ядерный процессор;
* оперативная память 2 ГБ и выше;
* свободное место накопителя не менее 500 Мб.

# **2. СТРУКТУРА ПРОГРАММЫ**

Программа «Just animate» имеет в своем составе следующие функциональные модули:

– модуль управления;

– модуль работы с данными;

– модуль отображения.

Связи между составными частями программы определяются их взаимодействием и зависимостями друг от друга. Модуль управления может обращаться к модулю работы с данными для сохранения и загрузки анимации, а модуль отображения может получать данные из модуля управления для визуализации анимации. Интерфейс пользователя взаимодействует с модулем управления через события и команды, передавая пользовательские действия и получая обновленные данные для отображения на экране.

# **3. НАСТРОЙКА ПРОГРАММЫ**

# **3.1. Состав и содержание дистрибутивного носителя данных**

В состав и содержание дистрибутивного носителя данных для программы «Just animate» входят следующие элементы:

– установочный файл. Установочный файл программы представляет собой исполняемый файл (.exe) или архив (.zip), который содержит все необходимые файлы и компоненты для установки и запуска программы на компьютере пользователя;

– документация и руководство пользователя;

– документация по установке, настройке и использованию приложения;

– ресурсы и файлы программы. В дистрибутивном носителе данных также должны содержаться все необходимые ресурсы и файлы программы, такие как исходные коды (.cs файлы), исполняемые файлы, библиотеки и компоненты (.dll файлы) и изображения, используемые в интерфейсе программы.

– лицензионная информация и соглашения об использовании приложения;

# **3.2 Порядок загрузки программы**

Пользователь загружает и устанавливает программу на свое устройство. После установки необходимо прочитать и принять лицензионное соглашения, после этого можно будет пользоваться программой.

# **4. ПРОВЕРКА ПРОГРАММЫ**

Для проверки работоспособности программы «Just animate» выполнить следующие действия:

1. установить программу «Just animate» на компьютер;
2. открыть программу и принять пользовательское соглашение;
3. изучить предоставленные материалы по обучению покадровой анимации;
4. создать новый проект, указав нужный размер листа;
5. создать анимации с предоставленных функций;
6. изменить начальных настроек кисти;
7. сохранить созданную анимацию, указав имя файла и тип;

# **5. СООБЩЕНИЯ**

В процессе работы программного продукта по созданию покадровой анимации могут возникнуть различные аварийные ситуации. Ниже приведены некоторые из них и возможные способы их устранения:

Сообщения в ходе настройки программы:

1) «Ошибка: не удалось найти библиотеку или компонент»

– содержание: это сообщение указывает на отсутствие необходимых библиотек или компонентов, которые требуются для работы программы;

– действия: проверить, правильно ли установлены все требуемые компоненты программы. Убедится, что необходимые файлы и библиотеки присутствуют и правильно настроены.

2) «Предупреждение: не удалось загрузить настройки»

– содержание: это сообщение указывает на проблемы с загрузкой сохраненных настроек программы;

– действия: проверить, правильно ли указан путь к файлу настроек. Убедится, что файл существует и доступен для чтения. Если файл поврежден или отсутствует, можно попробовать восстановить его из резервной копии.

Сообщения в ходе проверки программы:

1) «Ошибка: Неверное значение параметра»

– содержание: это сообщение указывает на некорректное значение одного из параметров программы;

– действия: проверить введенные значения параметров и убедится, что они соответствуют допустимым значениям. Исправить значение параметра и повторить проверку.

2) «Предупреждение: Некорректная настройка параметра»

– содержание: это сообщение указывает на некорректную настройку одного из параметров программы, которая может привести к неправильному результату.

– действия: проверить настройки параметра и убедится, что они соответствуют требованиям программы. Откорректировать настройки и повторить проверку.

Сообщения в ходе выполнения программы:

1) «Сохранение проекта»

– содержание: это сообщение указывает на процесс сохранения текущего проекта анимации.

– действия: дождаться завершения сохранения проекта. Убедится, что все изменения были успешно сохранены.

2) «Ошибка: не удалось открыть файл»

– содержание: это сообщение указывает на проблему с открытием указанного файла или проекта анимации.

– действия: проверить, правильно ли указан путь к файлу или проекту. Убедится, что файл существует и доступен для чтения. Если файл поврежден или неподдерживаемого формата, попробовать восстановить его из резервной копии или создать новый файл проекта.

**ПРИЛОЖЕНИЕ В**

Листинг программного кода:

namespace WpfApp1

{

    /// <summary>

    /// Логика взаимодействия для DrawAnimateWindow.xaml

    /// </summary>

    public partial class DrawAnimateWindow : Window

    {

        List<ToggleButton> ButtonLayers = new List<ToggleButton> {  };

        List<InkCanvas> Layers = new List<InkCanvas>();

        int LayerWidth;

        int LayerHeight;

        public DrawAnimateWindow(int WidthWindow, int HeightWIndow)

        {

            InitializeComponent();

            ButtonLayers.Add(FirstButton);

            LayerHeight = HeightWIndow;

            LayerWidth = WidthWindow;

            FirstLayer.Height = HeightWIndow;

            FirstLayer.Width = WidthWindow;

            Layers.Add(FirstLayer);

        }

        private void DistributeLayers(ToggleButton thisButton)

        {

            int thisPosition = ButtonLayers.IndexOf(thisButton);

            for (int i = 0; i < thisPosition; i++) //Слои до нужного становятся прозрачнее

            {

                Layers[i].Opacity = 0.8 - (Math.Abs(i - thisPosition) \* 0.02);

                Layers[i].IsEnabled = false;

                Panel.SetZIndex(Layers[i], i+2);

            }

            Layers[thisPosition].Opacity = 0.8;

            Layers[thisPosition].IsEnabled = true;

            Layers[thisPosition].DefaultDrawingAttributes.Width = size\_slider.Value;

            Layers[thisPosition].DefaultDrawingAttributes.Height = size\_slider.Value;

            if ((bool)Eraser.IsChecked)

            {

                Layers[thisPosition].DefaultDrawingAttributes.Color = Colors.White;

            }

            Panel.SetZIndex(Layers[thisPosition], ButtonLayers.Count+1);

            for (int i = thisPosition+1; i < Layers.Count;i++)

            {

                Layers[i].Opacity = 0;

                Layers[i].IsEnabled = false;

                Panel.SetZIndex(Layers[i], 1);

            }

            foreach (var item in ButtonLayers)

            {

                if (item != thisButton)

                {

                    item.IsChecked = false;

                    continue;

                }

                item.IsChecked = true;

            }

        }

        private void SliderSize\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

        {

            size\_slider.Value = Math.Round(size\_slider.Value);

            ChangedSize(size\_slider.Value);

        }

        void ChangedSize(double value)

        {

            for (int i = 0; i < ButtonLayers.Count; i++)

            {

                if ((bool)ButtonLayers[i].IsChecked)

                {

                    Layers[i].DefaultDrawingAttributes.Width = value;

                    Layers[i].DefaultDrawingAttributes.Height = value;

                }

            }

        }

        private void TextBoxSize\_TextChanged(object sender, TextChangedEventArgs e)

        {

            double value;

            if (double.TryParse(size\_textBox.Text, out value))

            {

                if (size\_slider != null && value >= size\_slider.Minimum && value <= size\_slider.Maximum)

                {

                    size\_slider.Value = value;

                    ChangedSize(value);

                }

            }

        }

        private void SliderTransparent\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

        {

        }

        private void TextBoxTransparent\_TextChanged(object sender, TextChangedEventArgs e)

        {

            double value;

            if (double.TryParse(transparent\_textBox.Text, out value))

            {

                if (transparent\_slider != null && value >= transparent\_slider.Minimum && value <= transparent\_slider.Maximum)

                {

                    transparent\_slider.Value = value;

                }

            }

        }

        private void ToggleButton\_Checked(object sender, RoutedEventArgs e)

        {

            ToggleButton clickedButton = (ToggleButton)sender;

            if (clickedButton.IsChecked == true)

            {

                foreach (UIElement element in stackPanel.Children)

                {

                    if (element is ToggleButton toggleButton && toggleButton != clickedButton)

                    {

                        toggleButton.IsChecked = false;

                    }

                }

            }

        }

        private void AddCanvasButton\_Click(object sender, RoutedEventArgs e)

        {

            InkCanvas newCanvas = new InkCanvas();

            newCanvas.Width = LayerWidth;

            newCanvas.Height = LayerHeight;

            Layers.Add(newCanvas);

            DrawMesto.Children.Add(newCanvas);

            ToggleButton newButton = new ToggleButton();//(ToggleButton)sender;

            newButton.Height = 100;

            newButton.Width = 160;

            newButton.Background = System.Windows.Media.Brushes.Transparent;

            newButton.BorderBrush = System.Windows.Media.Brushes.White;

            newButton.BorderThickness = new Thickness(2);

            newButton.Margin =  new Thickness(10, 10, 10, 10);

            newButton.Click += ChooseLayer\_Click;

            int index = ButtonPanel.Children.Count - 1;

            ButtonPanel.Children.Insert(index, newButton);

            ButtonLayers.Add(newButton);

            DistributeLayers(newButton);

        }

        private void ChooseLayer\_Click(object sender, RoutedEventArgs e)

        {

            //FirstLayer.IsEnabled = !FirstLayer.IsEnabled;

            DistributeLayers((ToggleButton)sender);

        }

        private void Save\_Click(object sender, RoutedEventArgs e)

        {

            //MagickImageCollection animation = new MagickImageCollection();

            //IMagickImageCollection

            List<Bitmap> images = new List<Bitmap>();

            foreach (InkCanvas canvas in Layers)

            {

                canvas.Opacity = 1;

                RenderTargetBitmap renderBitmap = new RenderTargetBitmap(

                    (int)canvas.ActualWidth, (int)canvas.ActualHeight,

                    96d, 96d, PixelFormats.Default);

                renderBitmap.Render(canvas);

                // Convert WPF RenderTargetBitmap to GDI Bitmap

                using (MemoryStream outStream = new MemoryStream())

                {

                    BitmapEncoder encoder = new BmpBitmapEncoder();

                    encoder.Frames.Add(BitmapFrame.Create(renderBitmap));

                    encoder.Save(outStream);

                    images.Add(new Bitmap(outStream));

                }

            }

            SaveFileDialog saveFileDialog = new SaveFileDialog();

            saveFileDialog.Filter = "GIF files (\*.gif)|\*.gif|All files (\*.\*)|\*.\*";

            if (saveFileDialog.ShowDialog() == true)

            {

                string filePath = saveFileDialog.FileName;

                using (var gif = AnimatedGif.AnimatedGif.Create(filePath, 100))

                {

                    foreach (var image in images)

                    {

                        gif.AddFrame(image, delay: -1, quality: GifQuality.Bit8);

                    }

                }

            }

            foreach (var button in ButtonLayers)

            {

                if ((bool)button.IsChecked)

                {

                    DistributeLayers((button));

                }

            }

            MessageBox.Show("Анимация сохранена");

        }

        void ChangeColor(System.Windows.Media.Color color)

        {

            for (int i = 0; i < ButtonLayers.Count; i++)

            {

                if ((bool)ButtonLayers[i].IsChecked)

                {

                    Layers[i].DefaultDrawingAttributes.Color = color;

                }

            }

        }

        private void Brush\_Click(object sender, RoutedEventArgs e)

        {

            Eraser.IsChecked = false;

            ChangeColor(System.Windows.Media.Colors.Black);

        }

        private void Eraser\_Click(object sender, RoutedEventArgs e)

        {

             Brush.IsChecked = false;

            ChangeColor(System.Windows.Media.Colors.White);

        }

    }

}

namespace WpfApp1

{

    /// <summary>

    /// Логика взаимодействия для MainWindow.xaml

    /// </summary>

    public partial class MainWindow : Window

    {

        public MainWindow()

        {

        }

        private void Create\_new\_project(object sender, RoutedEventArgs e)

        {

            ProjectSetup projectSetup = new ProjectSetup();

            projectSetup.SetPreviousWindow(this);

            projectSetup.Show();

        }

        private void OpenBrowserButton\_Click(object sender, RoutedEventArgs e)

        {

            string url = "https://drive.google.com/drive/folders/1jBbGnao82p4BLO5jLsC0cyoDbQPw2Dh\_?usp=sharing"; // Замените URL на нужный вам

            Process.Start(new ProcessStartInfo

            {

                FileName = url,

                UseShellExecute = true

            });

        }

    }

}

using System;

using System.Collections.Generic;

namespace WpfApp1

{

    /// <summary>

    /// Логика взаимодействия для ProjectSetup.xaml

    /// </summary>

    public partial class ProjectSetup : Window

    {

        private Window previousWindow;

        public double WidthValue { get; set; }

        public double HeightValue { get; set; }

        public ProjectSetup()

        {

            InitializeComponent();

        }

        private void Button\_Close(object sender, RoutedEventArgs e)

        {

            this.Close();

        }

        public void SetPreviousWindow(Window window)

        {

            previousWindow = window;

        }

        private void Button\_Click(object sender, RoutedEventArgs e)

        {

                DrawAnimateWindow drawAnimateWindow = new DrawAnimateWindow(Convert.ToInt32(widthTextBox.Text), Convert.ToInt32(heightTextBox.Text));

                drawAnimateWindow.Show();

                previousWindow.Close();

                this.Close();

                 Close(); // Закрытие второго окна

        }

    }

}